Présentation de travaux en vue de l'obtention de l'habilitation à diriger des recherches de l'Université Paul Sabatier, Toulouse III

Spécialité : Informatique

par

Marco WINCKLER

Engineering Interactive Systems across application domains

C. Freitas	Examinateur	Professeur, Universidade Federal do Rio Grande do Sul
L. Nigay	Rapporteur	Professeur, Université Joseph Fourier, Grenoble 1
N. Stanton	Rapporteur	Professeur, University of Southampton
P. Palanque	Président du Jury	Professeur, Université Paul Sabatier
O. Pastor	Rapporteur	Professeur, Universidad Politécnica de Valencia
G. Rossi	Examinateur	Professeur, Universidad National de la Plata

Interactive Critical Systems (ICS) - Institute of Research in Informatics of Toulouse (IRIT)
University Paul Sabatier (Toulouse 3), 118 route de Narbonne, 31062 Toulouse CEDEX 9, France
Phone: +33 (0)5.61.55.63.59 / E-mail: winckler@irit.fr / Web: http://www.irit.fr/~Marco.Winckler/

Abstract

The development of interactive systems is complex: on one hand we have to consider the users and their needs, and on the other hand we have to deal with the practical aspects of the software development. Users occupy a central place in the development of interactive systems because their interaction with the systems is required to make the system work; if we remove the users (or if we don't take into account their skills, motivations and needs) it will become impossible to operate the systems and thus determine whether (or not) it fulfill the requirements. Thus, the focus on users is a keystone for the research on Human-Computer Interaction (HCI). However, we must acknowledge that members of the development team are also users of interactive tools that are used to build other interactive systems. For that, it is sensible to say that, in addition to the diversity of users and users' needs, approaches for developing interactive systems must also take into account the developer's needs for dealing with the idiosyncrasies of technology and application domains; otherwose, if developers don't have appropriate tool support they will not be able to achieve their job successfully.

In some extension, the research on the engineering of interactive systems is in the crossroad between disciplines such Human-Computer Interaction (HCI), Software Engineering, and more generally on Computer Sciences. Currently, a reasonable amount of research work is motivated by arguments based on idiosyncrasies of particular types of interactive systems. But yet, many questions of non-exclusive practical and theoretical significance can be evenly applied regardless the type of interactive system at concern. The development of reliable, usable and effective interactive systems, their systematically analysis and treatment require appropriate methods, models, tools and approaches. On one hand it is tempting to focus on specialized solutions aimed for a particular type of interactive systems which are defined by unique features (such as system characteristics and users' requirements, etc.). But on the other hand, it might have many convergent points among diverse types of interactive systems for which solutions exist and just require a few adaptations.

Interactive systems can assume many forms and quickly become complex. I have been fascinated by the development of Web applications since I was an undergraduate student in countryside Brazil. At that time, some argued that the technology for building such applications pre-existed the Web (which is somewhat true) and there was nothing really new about it... and yet, I thought there was something worthy being investigated. Hereupon two main questions haunted me: "How to provide an accurate description of features and idiosyncrasies of Web applications I was developing?" and "How to determine whether (or not) that Web applications really fulfilled the needs of the target users?" These questions lead me first to pursue a master degree in Computer Sciences (1997-1999) and then a PhD thesis (2000-2004). I have found in the field of Human-Computer Interaction (HCI) the theoretical background necessary for supporting my reflections about models for specifying the behavior of interactive applications and methods for assessing the usability. In a short run, my research was focused on the field of Human-Computer Interaction while it was also contributing to the emerging field of Web Engineering (WE). Nonetheless, in the meantime I have observed several connections between more diverse types of interactive systems including e-government applications, e-voting systems, ground-segment systems, incident reporting systems, information visualization techniques, mobile interactive applications, multimodal applications, personal information management systems… For that, I have found that the work I was developing could contribute to more than a particular application domain and, in some extensions, generalized and reused. Generalization are useful, but they cannot be interpreted as shortcuts because that would prevent the understanding of socio-technical aspects that determine the applicability and acceptance of method and tools across domains.

In this work I present the results of my reflections for the research on engineering of interactive systems. Rather than arguing about the differences between application domains (and/or providing an exhaustive list of facets that can be used to characterize interactive systems), this work presents my contributions to the field by following a more holistic and yet pragmatic approach. For the purposes of illustration, Web applications will be used to support the discussions about the adaptation of methods to solve particular engineering problems that cannot be solving with more general methods. The choice of Web applications is not arbitrary as it also reflects my personal interests for applications using Web technology and illustrates well how I have oriented by research carrier in earlier years. It is also worthy of mention that we should not forget how influential Web technology is nowadays and therefore it would be unavoidable to include it our reflection about the engineering of interactive systems. The contribution are organized in four main topics: development processes, users of interactive system, model-based approaches, and evaluation methods. We make no claim for covering all aspects of the engineering of interactive systems. However, we assume that these topics are cornerstones for understanding the underlying research problems for that they are expected to settle a long-term research agenda in the field.

Keywords: Human-Computer Interaction (HCI), Engineering Interactive Systems, User-Centered Design process, User Interface design, Model-Based Development Process, Usability Evaluation, Web Engineering.

“A record, if it is to be useful to science, must be continuously extended, it must be stored, and above all it must be consulted” – Vannevar Bush (1890-1974).

“Whenever a theory appears to you as the only possible one, take this as a sign that you have neither understood the theory nor the problem which it was intended to solve” – Karl Popper (1902-1994)

Table of Contents

PART II - Engineering Interactive Systems across application domains

1 Introduction

Personal computers and computing systems create a milestone in 20th century history and certainly can be said to settle a turning point in humankind evolution towards the information society. Indeed, computing systems become omnipresent in almost all aspects of our daily lives including communications, education, health, transport, entertainment … and, if we consider for a moment the daily use of mobile apps and Web applications, we can just wonder how much dependent of interactive technologies we have become.

Few computing systems are completely autonomous nowadays. Indeed, most of computing systems (ranging from the usual applications deployed into desktop computers and mobile phones, to more domain specific systems such as those embedded into aircraft's cockpits, voting machines…) are indeed interactive. The underlying statement for defining an interactive system encompasses a dialog between users and the computer where users not only provides inputs to a computing system but are the ultimate receivers of the corresponding system's outputs [63]. Moreover, inputs and outputs events occur as long as user interacts with the system. Those canonical definitions about user interaction with interactive system are illustrated by Figure 1. Therefore, the construction of interactive systems not only requires building computers and software but ultimately it should consider users at some point.

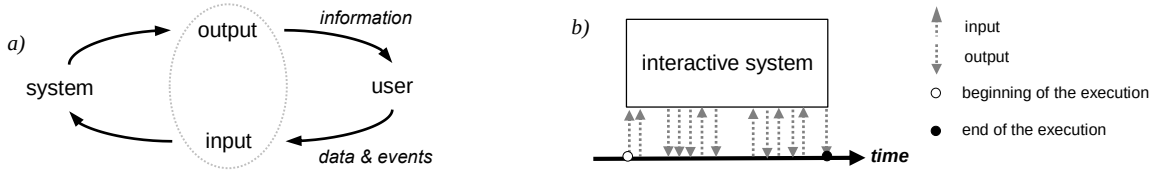


Figure 1. Canonical views on user interaction with interactive system: a) user interaction based on occurrence of input/output events; b) input/output interactions along system execution (adapted from Bastide, Palanque [19]**)**.

The central role of users along the development process of interactive systems was settled down in the 1980s with the emergence of the Human-Computer Interaction (HCI) [44] which thereupon defined dual research agenda in the field: on one hand it aims at understanding how human cognition and behavior might affect (or be affected by) the use of technology; the quest for such knowledge motivates the empirical research on Human Factors which, ultimately, is hoped to inspire the design of innovative interactive systems that better fulfill users' needs. On the other hand, many aspects of the research on HCI concern the development of interactive systems, which means systems that are built to react to users' input (ex. keystrokes, mouse clicks…) and provide them with a meaningful representation (either visual, auditory…) of the results produced by the inner algorithms embedded into computers. Figure 2 illustrates this view accordingly to a multidisciplinary perspective where boundaries between disciplines are blurred to indicate overlapping between research topics and to highlight the central place of users who are the raison d'être of interactive systems.

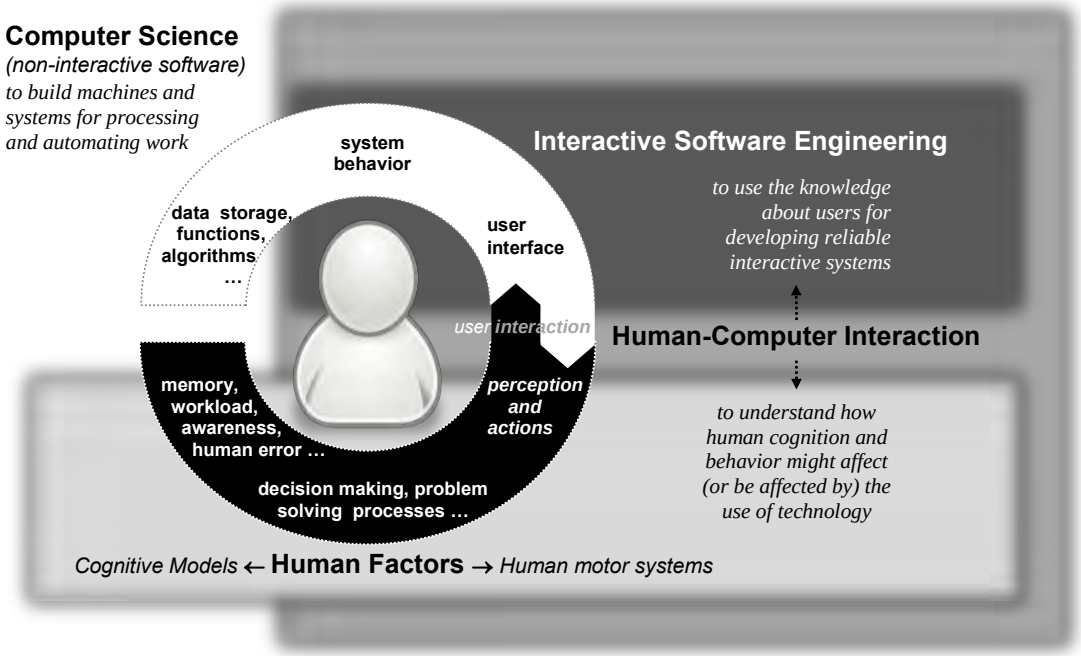


Figure 2. Diagrammatic view of the main disciplines involved in development of interactive systems.

Users occupy a central place in the development of interactive systems because their interaction with the systems is required to make the system work; if we remove the users (or if we don't take into account the users skills and needs) it becomes impossible to operate the systems and thus determine whether (or not) it fulfill the requirements. Currently, it is widely agreed that the quality of interactive systems depend on the users and this view is clearly sustained by standards such as the ISO 9241-11 (1988) [105] (which defines the concept of usability as "*the extent to which a product can be used by the target users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use*") and ISO 13407 (1999) [109] (which advocates the use of User-Centered Design (UCD) processes as a mean for taking users' needs into account and for building usable interactive systems). Nonetheless, users' needs might vary dramatically across the population and/or might be interpreted differently according to diverse context of use. So that the development of many real-life interactive systems becomes a complex endeavor that demands a deep analysis of the tradeoffs between overall system's requirements, users' needs and technical constraints. For example, social acceptance and users' adoption of e-voting applications not only depends on the system usability (i.e. making easy for users to cast votes even none or minimum prior training) but also requires the system to feature other properties such as accessibility (i.e. as for impaired users should have equal voting rights) and privacy (i.e. no link can be made between a vote and a voter) [248]. For safety critical systems such as interactive aircraft cockpits, the usability has implications for the training of pilots but safety is vital for ensuring that the systems will not endanger the passengers' life and/or the environment [223]. Whilst rebooting the system is a reasonable (and yet acceptable) solution for recovering a voting machine from software crash, this is totally unacceptable in a real-time critical systems where people life is at stake [142]. Thus the overall quality of interactive system not only depends on the quality of the software/hardware products but it also should take into account the implications for the user interaction and their which cannot be dissociate.

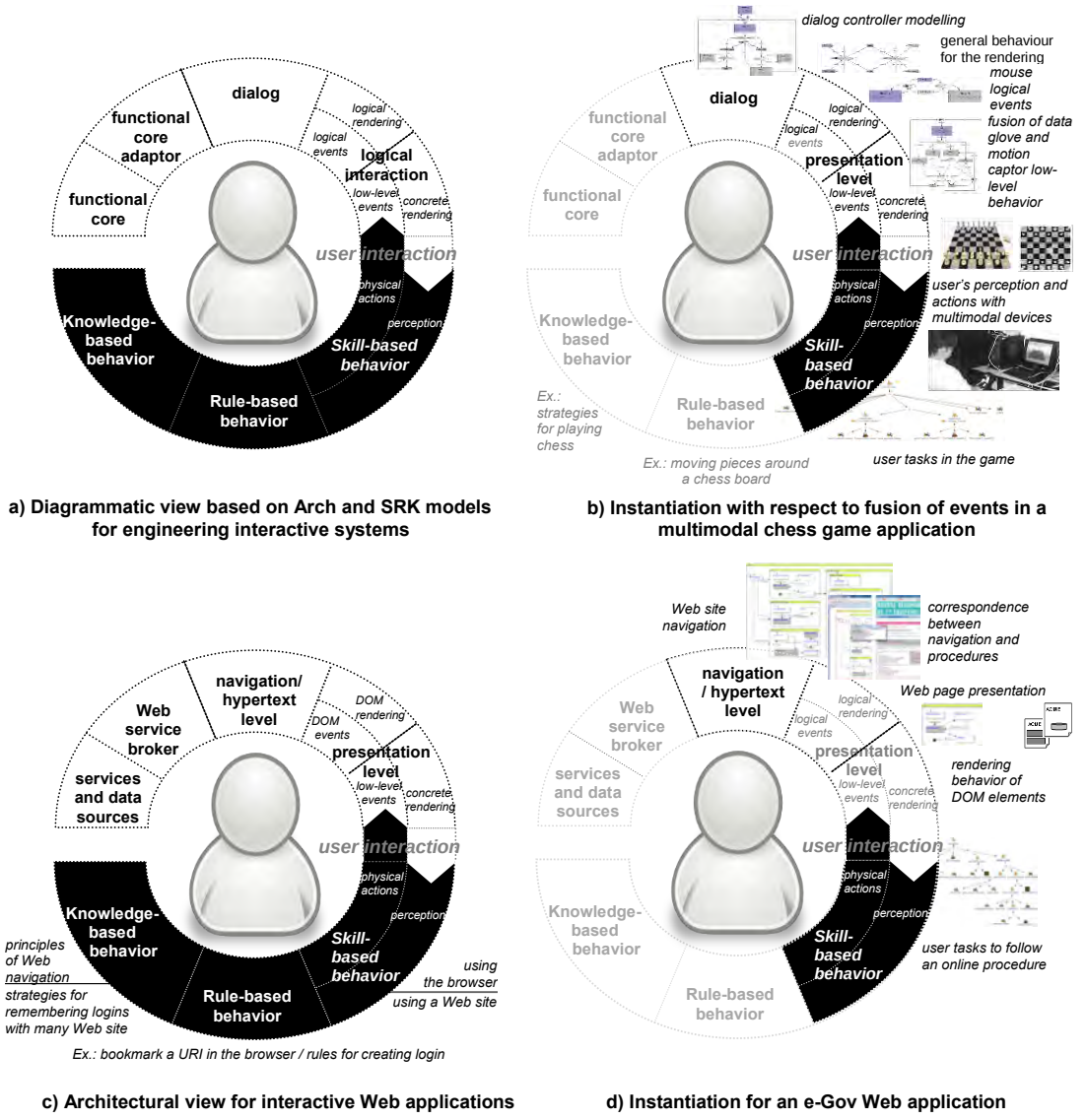
Interactive systems can assume many forms (ex. cockpits, mobile apps, interactive TV, e-government applications, e-voting systems...) and quickly become complex (ex. a simple standalone inventory management system can rapidly evolve to an online retailing application based on worldwide supply chain). Thus, it is sensible to say that, in addition to the diversity of users and users' needs, approaches for developing interactive systems must also take into account idiosyncrasies of technology and application domains. Currently, a reasonable amount of research work is motivated by arguments based on idiosyncrasies of particular types of interactive systems. But yet, many questions of non-exclusive practical and theoretical significance can be evenly applied regardless the type of interactive system at concern. For example: How to provide a meaningful support to describe and analyze users' tasks? How to inform users which tasks are supported by the system and how they can be accomplished? How to embed user's needs for guidance and training as part of the development process of interactive systems? How to optimize and/or find a balance on the allocation of tasks performed by users and the system? How to cope with tasks when users are interrupted? How to provide seamlessly support to users to accomplish their tasks alone or as part of a collaborative work? How to describe the inner behavior of interaction techniques? How to deal with increasing complexity of systems? How make sure the systems support user's work as specified? How to cope with competing and (possibly) conflicting properties that might affect the design of user interfaces? How to mitigate the effect of users' errors, mistakes and any possible harmful actions with the interactive systems? …

Those questions are so important for the development of reliable, usable and effective interactive systems that their systematically analysis and treatment require appropriate methods, models, tools and approaches. On one hand it is tempting to focus on specialized solutions aimed for a particular type of interactive systems which are defined by unique features (such as system characteristics and users' requirements, etc.). But on the other hand, it might have many convergent points among diverse types of interactive systems for which solutions exist and just require a few adaptations. To understand the tradeoffs between properties, problems and design solutions ones has to adopt two interweaving ways of reasoning: abstraction/generalization and specialization/adaptation of suitable alternatives. In order to illustrate this point, Figure 3 provides a series of examples focused on the articulation of models for describing interactive systems. It is worthy of notice that, as far as interactive systems are a concern, are of particular interest models that are able to describes human behavior and models that are able to describe the inner behavior of the system. In Figure 3.a we examine the articulation between the SRK model [194] and the Arch model [17][18]. The SRK model (the lower part of Figure 3.a) is used to describe the human behavior in terms of skills, rules and knowledge and how such behaviors affect user activity with the interactive system. The Arch model (the upper part of Figure 3.a) explains how user inputs are treated by the system as logical events and how these events are processed at the dialog layer, which make calls to inner core functions and send the corresponding output to the user through logical and concrete rendering. The models presented at Figure 3.a are generic. An instantiation of the SRK and Arch models is done at Figure 3.b for describing problems associated with the fusion/fission of events in multimodal applications. The demonstration is done by using a particular a Chess game application that can be operated by using data glove, keyboard and motion captor as described in [157]. Here, the discussion of the fusion/fission problem is mainly covered by the presentation and dialog levels of the interactive system so that the other levels of the Arch model (i.e. functional core and functional core adaptor) are fading in that picture. The same graphical representation is used with human behavior (knowledge-based such as "strategies for playing chess" and rule-based behavior such as "moving pieces around a chess board") that is

not required to test the hypothesis associated to the multimodal interaction. Whilst the overall principles presented at Figure 3.a might be valid to any kind of interactive system, some applications might require adaptations.

Figure 3.c shows an example of such adaptations for dealing with the idiosyncrasies of Web applications. The upper part of Figure 3.c is still very similar to the Arch model but the vocabulary has been changed. For example, the *dialog* is replaced with *navigation/hypertext level* to highlight the fact that navigation is the main concerns for dialog modeling in Web applications. *Web service broker*, *services and data sources* correspond to terms used to designate components that run on Web servers and constitute core elements of information processing of Web applications. The *presentation level* shows the elements of user interaction on the client side, i.e. the Web browser. In Figure 3.c the structure of the SRK model is the same as in previous example but it is decorated with examples that illustrate the knowledge and behavior users might have for using *the browser* (e.g. *bookmark a URI, principles of Web navigation*) and for using *a particular Web site* (e.g. *special rules for creating a login, strategies for remembering logins with many Web sites*).

An instantiation of this model for a particular e-Gov Web application is given Figure 3.d where the focus is given to tasks users have to accomplishing by navigating a Web site to follow an administrative procedure. When we compare models used for specifying the multimodal Chess game application (Figure 3.c) and the e-Gov Web application (Figure 3.c) it is possible to observe that different models have been used. Nonetheless, this is simply another kind of adaptation/tuning required for coping with fine-grained interaction (as in the example of the multimodal Chess game) or the coarse-grained interaction (as shown by the navigation of the e-Gov Web site).



a) Diagrammatic view based on Arch and SRK models for engineering interactive systems

b) Instantiation with respect to fusion of events in a multimodal chess game application

c) Architectural view for interactive Web applications

d) Instantiation for an e-Gov Web application

Figure 3. Examples of articulation of models for describing concerns related to the development of interactive systems: a) generic view based on the articulation of the Arch model [17] and SRK model [194]; b) instantiation of models for highlighting concerns related to fusion/fission of event in a multimodal chess game application; c) specialization of models for copying with Web architecture and user interaction with the Web browser and Web site; d) instantiation describing a particular e-Gov Web application aimed at help users to accomplish an administrative procedure online.

As we shall see in the examples above, the dialog modeling is an aspect (shared by both Web and multimodal applications) that raises fundamental questions about the interactive system behavior. Nonetheless, adaptations of models are required to find the optimal solution to a particular interactive system. Such adaptations are driven by real needs (in terms of description of the systems, evaluation of performance, communication support along the development team, automation of work, etc.) for accomplishing an engineering work.

A substantial part of my research activities have been focused in to find a balance between the reuse of existing knowledge to build interactive systems and the idiosyncrasies of particular types of applications. Along the years I also came to investigate a diverse types of interactive systems including e-government applications, e-voting systems, ground-segment systems, incident reporting systems, information visualization techniques, mobile interactive applications, multimodal applications, personal information management systems… In due time, it was possible to observe some of the connections between these diverse types of interactive systems and look for a more general perspective for the engineering of the interactive system. A general theory about the engineering of interactive systems is beyond the scope of this work, but we hope to contribute to a better understanding of what engineering interactive system is about and what are the challenges for applying, reusing and adapting methods, models, tools and approaches to support the development to interactive systems in different application domains. The identification of such challenges is aimed at setting a full research agenda for future work.

Thus, rather than arguing about the differences between application domains (and/or providing an exhaustive list of facets that can be used to characterize interactive systems), this work presents my contributions to the field by following a more holistic and yet pragmatic approach. For the purposes of illustration, Web applications will be used to support the discussions about the adaptation of methods to solve particular engineering problems that cannot be solving with more general methods. Nonetheless, to explain the adaptations, the problems are first presented in more general terms, followed by an illustration of my contributions to the Engineering of Interactive System and then discussed in terms of specialization/ adaptations that might be required to cope with the idiosyncrasies of Web applications. The choice of Web applications is not arbitrary as it also reflects my personal interests for applications in this domain and illustrates well how I have oriented by research carrier, in particular in earlier years. In addition to that, it is worthy of mention that we should not forget how influential Web technology is nowadays and therefore it would be unavoidable to include it our reflection about the engineering of interactive systems. The rest of the work is organized around four main chapters followed by a long-term research agenda as follows:

- Chapter 2 is focused on the development process of interactive systems and, in particular, user-centered design processes.
- Chapter 3 presents some challenges for collecting and representing users and how to use such knowledge to build interactive systems.
- Chapter 4 discusses the role of model-based approaches for the engineering interactive systems.
- Chapter 5 revisits methods for assessing user interfaces at the light of their potential contributions for some of the challenges developers might have to integrate evacuation outputs into the development process of interactive systems.

Each one of these chapters contains a state-of-the-art, a summary of my contributions (on interactive systems and Web applications) and it proposes a research agenda in the matter. Herein, we make no claim for covering all aspects of the engineering of interactive systems. Indeed, a discussion about architectures, programing languages, interaction techniques and platforms are notably missing in the present work. Nonetheless, these aspects are discussed in the last (but not least) chapter which is aimed at providing a long-term research agenda in the field. The last chapter also discusses some opportunities for cross-pollination of the knowledge obtained with Web applications and other types of interactive systems.

2 Development process of interactive systems

The ultimate goal of theories, methods and techniques in any engineering field is to help developers to deal with the production of quality products that are economical and timely delivered. In Computer Science, the study of development processes is often seem as a mean to understand, describe, assess, automate and improve procedures, policies and techniques used to tackle the inner complexity of the development of software products [82]. When interactive systems are at a concern, complexity is increased by the fact that the some software qualities depend upon how users perceive and interact with the system. This chapter has the ambition of analyzing software development processes and in particular those that we consider suitable for engineering of interactive systems. Hereafter we make some bold claims about the needs people that participate during the development processes (which ultimate includes end-users, clients, stakeholders and members of the development team) might have in terms of guidance to achieve the development of high quality interactive systems. The sections ranging from 2.1 to 2.4 present a summary of existing software processes and discuss some aspects that might affect the adoption of process for engineering interactive systems. The subsequent sections present respectively some of my contributions that are aimed at improving the development process for interactive systems in general (section 2.5) and more contributions for improving processes for dealing with Web-based applications (section 2.6). Lately (in the section 2.7) we propose a research agenda.

2.1 The development processes as an open box

In the last decades there is an increasing awareness of the importance of development processes as a mean to ensure the quality of products in a large sense [55]. Such awareness comes from the fact that if no explicit notion of process is in place, the development of products only can be explained as a kind of black box where only the input (i.e. the requirements) and the output (i.e. the product delivered) are known. As illustrated by Figure 4.a, a black box process does not inform the steps that lead from requirements to the final product thus preventing a systematic analysis of problems and reducing the possibilities of a transparent coordinating the activities among members of the development team; as a consequence, it is difficult to monitor the progress of the development and when the product appears at the end of the process it is often too late to care about the quality. A black box process is even worse for dealing with software products because that development process often starts with informal requirements raised by clients who are not able to translate their perception of the business world into precise requirements [112]. In the beginning of the process, software requirements are often informal, incomplete, they can be sometimes contradictory or not reflecting the customers' needs. In a black box process, there is no guidance for revising initial requirements or to introduce new requirements identified along the way.

Conversely, when an explicit process is in place we can see through complexity: activities can be decomposed into smaller and more manageable steps, members of the development team know the steps which help to coordinate their activities, introduction of errors can be prevented by applying solutions to known problems in a systematic way, and quality can be checked at every step of the process (as illustrated by Figure 4.b). Moreover, progress can be monitored and communicated to clients who can help to clarify, complete, refine and possibly revise requirements that might change during the process, thus increasing the chances that the final product meets the customers' expectations. Thus, whilst is tempting to deal with complexity in software development by separating the product (what is visible to the client/costumers) from processes (how this quality products can be achieved), we should recognize that product and processes are intermingled because it is by controlling processes that developers can inject the required qualities of products, to reduce time to market and to manage development costs. Therefore, it is clear that to achieve quality and software correctness we have to go inside the black box, describe steps into details and make sure that the process is structured in such a way that makes the development systematic and less suitable to include errors.

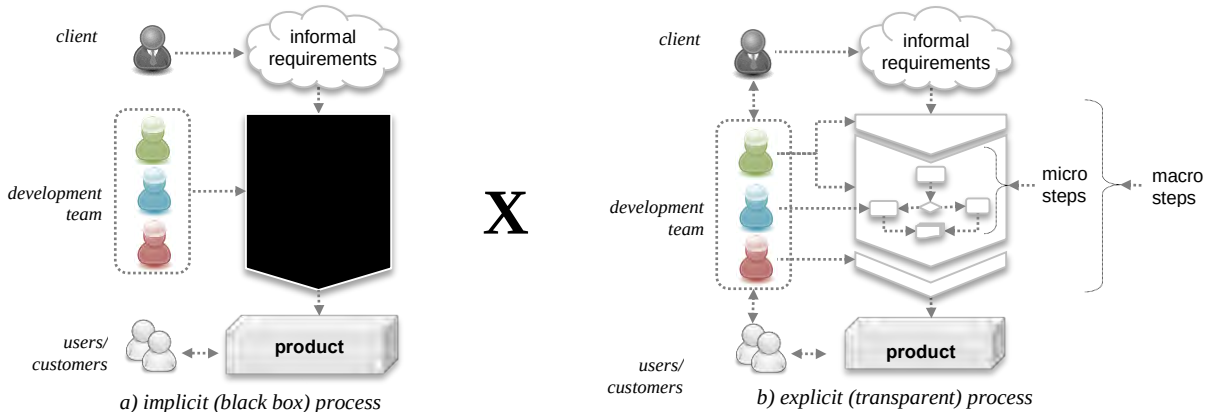


Figure 4. Overview of development process as an implicit black box (a) versus an explicit process (b).

2.2 Development processes models[1]

Every project is unique in terms of the problems that arise, the priorities and resources assigned to it, the environment in which it operates, and the project manager's attitude used to guide and control project activities. Therefore it is not possible to settle upon a single development process that would fit all projects' needs. Nonetheless, some models describing abstract views of the development process can help to represent the activities that should be performed and to pinpoint solutions to questions that underpin software project management [48].

It is out of the scope of this work to provide a comprehensive list of development process models but a few ones are worthy to be cited as they will help us to examine issues that we found relevant for the development of interactive systems. Figure 5 presents a selection of models that are commonly used to describe software development, which includes: the Waterfall model [203], the Spiral model [34], the Start model [102], the "V" Model [80], the ISO human-centered process [109], the agile method SCRUM [215], the IBM's RUP [122] and the Test Driven Development TDD [24]. As we shall see, those models often focus on a particular view about the software development which dramatically reduces the complexity of real-life projects. In the discussion that follows the description of every development process model we highlight the issues that we consider as important limitations for the strict implementation of development process models, at least as they appear in the literature.

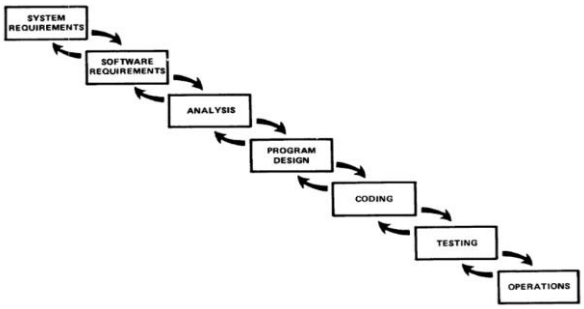
One of the most common views used for describing the development process is to present it as a straightforward lifecycle describing the sequence of steps that allows to go from initial requirements to achieved products. The lifecycle view of software development has been popularized by the waterfall model [203] which organizes activities required to build software as a sequence of steps where one step inherits the artifacts and decisions made in previous steps. Such organization is clearly inspired from manufacture production lines where production follows a well-stablished plan that don't include cycles. We should note that a strict implementation of a sequential process is risky for software development because it does not allow to clarify/introduce new requirements, revise decisions, and/or fix problems found in latter steps of the process. For that, Royce [203] proposed possible variants allowing to go back in the process. One of this variations presented by Figure 5.a, includes return for revising artifacts and decisions made in previous steps. Models based on a lifecycle, like the waterfall model, depicts an idealized process which is based on the assumption that the human process can be standardized once for all. The description of well-defined steps helps people to organize the documentation, the artifacts produced and how to report it afterwards. Nonetheless, as pointed by Parnas [177], lifecycle models are useful abstractions that are unsuitable for describing actual software development activities that rarely occur in a linear form so that they can only be considered valid for describing a process run *a posteriori*.

A closer look on actual developer's activities will reveal that many iterations are often necessary to mature design ideas and to explore design alternatives; such observation contradicts a linear view of software development based on straightforward sequence of steps. The spiral model [34] presented at Figure 5.b is one of the most emblematic processes that highlights the evolving nature of software development. The spiral model is described as a cycle made up of four generic steps encompassing the *definition of objectives, evaluation of alternatives, development* of ideas, and *planning of the next steps*. Those steps are articulated to produce incremental prototypes. Many cycles might be required to achieve a final product but every cycle bring additional knowledge about the requirements thus allowing to adjust the product to meet the client expectation. The central role played by incremental prototypes in the process encourages the exploration of many design alternatives, thus breaking the basic assumption of linear models such as the waterfall model that states that the outcomes of a development process is a single product and the end. The spiral model helps to explain the evolving nature of software development however the level of traceability required to follow the evolution of ideas and artifacts produced along the process might overburden the development team with activities that costly and unbearable for most projects.

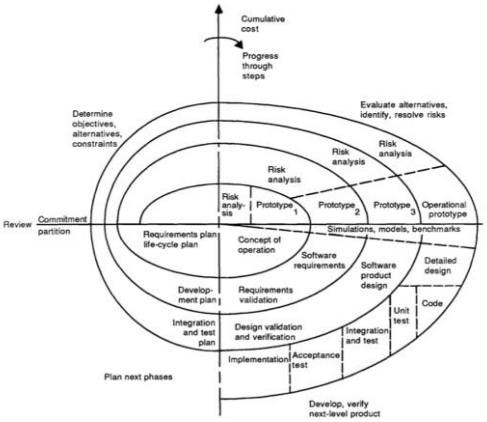
When quality control is a concern, we can identify two possible alternatives views: the first is to adopt continuous evaluation of artefacts; the second is to focus on quality control of the final product before delivery. Whist these views are not contradictory, they have strong implications with respect to the way of activities are organized in a development process. Most of the development processes models adopt either one or another view rather than to employ both strategies for quality control [112]. These two views are better illustrated by two different models, respectively by the star model [102] and by the V-model [80][148].

---

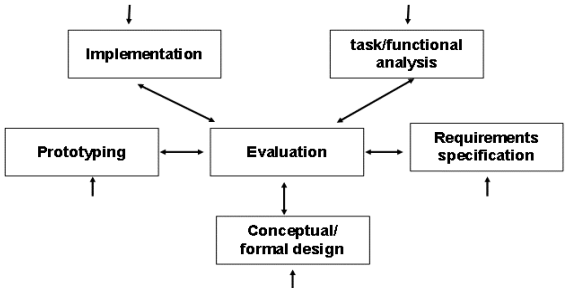
[1] The term *model* usually refers to an abstraction representation of the real world thus reducing its complexity to the aspects that really matter for the resolution of specific problems. Models of interactive systems contain information that is useful to build a software, for example system's core functions, user' tasks, etc. A full description of models that we consider useful for engineering an interactive system is presented at chapter 4. Nonetheless, in this chapter we also employ the term *models* to refer to abstract representation of the process that leads to the development of an interactive systems. To avoid any possible misunderstanding with other types of models, we adopt in this chapter the expression *development process models* when we refer those models describing the development process. The word *models* alone should be understood in a broad meaning. In a similar way, specific models will always be assigned with a complement that indicated the scope of information covered by the model, for example *task models* is used when mentioning abstract representation of the user's tasks.



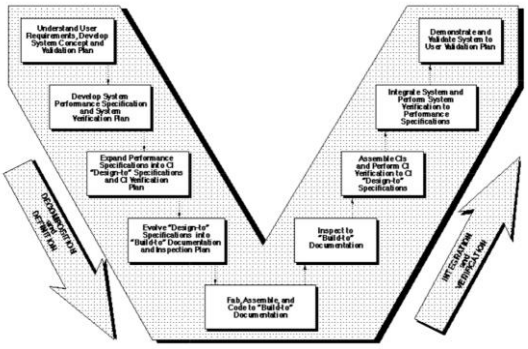
a) Waterfall process (Royce, W. W., 1970)



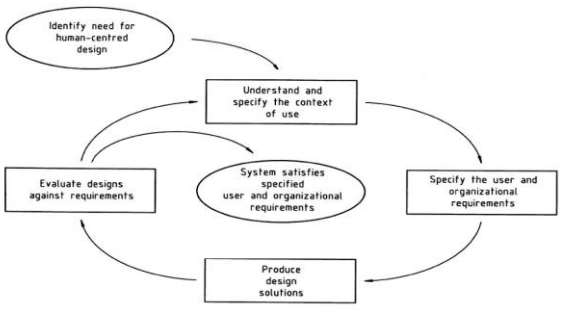
b) Spiral model of the software process (Boehm, B. 1986)



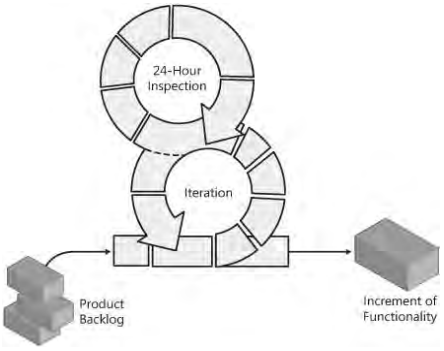
c) The star model (Hartson & Hix, H., 1989)



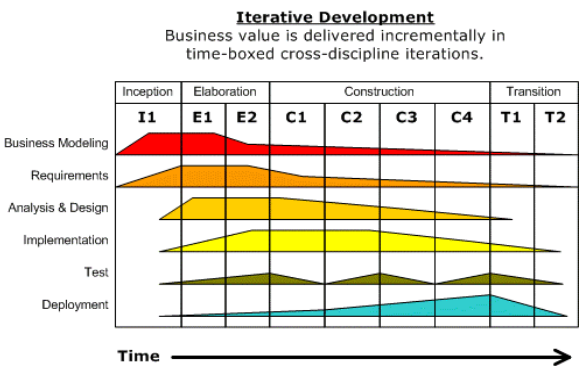
d) V-model (Forsberg, K., Mooz, H., 1991)



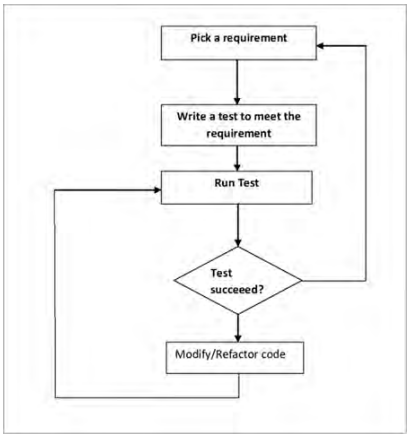
e) Standard for human-centered design processes for interactive systems (ISO 13407, 1999)



f) SCRUM skeleton (Shwaber & Beedle, 2002)



g) Rational Unified Process – RUP (IBM, 2003)



h) TDD approach to software developmen (Beck, 2003)

Figure 5. A samples of models commonly used for describing software development processes.

The star model (Harston & Hix, 1989) [102] is aimed at achieving quality control through continuous evaluation of artifacts produced along the development process. As depicted at Figure 5.c, every activity should be subject of an evaluation. The number of evaluations is not fixed in advance so that multiple evaluations can be performed in a loop for improving artifacts produced in a particular step. It is worthy of notice at Figure 5.c that multiple arrows are depicted to indicate the many possible entry points to the process. On one hand these multiple entries allowed by the start model might give some flexibility to start whenever it is more convenient but, on the other hand, they provide little guidance for streamlining the process. Moreover, the star model does not clearly state which evaluation methods should be used but we can assume that different assessments are run according to the artifacts available. Despite the fact that continuous evaluation can help to improve quality control we should recall that systematic evaluations might be cumbersome, costly and delay the development process as a whole. Evaluation of artifacts produced (ex. use cases, specifications of user tasks and of the systems, prototypes, etc.) help to ensure quality control along the development process but it don't guarantee defect-free products at delivery.

The V-model [80][148] is one of those development processes that is clearly concerned by the quality of products at the end of the process. For that V-model model separates steps of design/specification and evaluation in two branches that are connected in a V-like shape, as presented in Figure 5.d. The branches are connected at the step of coding/implementation. Every step of design/specification is paired by a step of evaluation which obliges the crosschecking of every artifact and decision made during the development process. However, evaluations always take place once the implementation is achieved. The main inconvenient of such as a process is that correction of errors and/or the inclusion of new requirements will force to return to the beginning of the cycle. Some variants of the original V-model explicitly show connections between the paired evaluation and design/specification phases so that it is possible to return to the steps where the problems should be fixed, instead of going back to the beginning. In that configuration of evaluation steps, the V-models help to make sure that every part of product (artifact, documentation, etc.) has been validated. The V process it obviously unsuitable for continuous quality control but it remains popular in projects that are concerned by certification of the final product.

As previously shown in Figure 4.b, development processes are meant to transform client/costumers requirement into products that fulfill user's expectations. So that is sensible to ask how users' needs are taken into account along the development process. The ISO 13407 standard for human-centered design processes for interactive systems [109] (also known as user-centered design – UCD process) tackle this issue by placing users at the center of the development process as depicted by Figure 5.e. The overall process is defined as iterative and supported by incremental prototypes that are expected to evolve and ultimately lead to the development of a final product. The process makes bold claims about the importance of multiple iterations, which should stops only when users and organizational requirements are met. Some of these features (ex. multiple iterations and use of prototypes) resemble some aspects of the spiral model but one of the particularities of the UCD process is to emphasize the continuous focus on users throughout the development process. Indeed, all steps recall for keeping a users' point of view. It is interesting to notice that the UCD process does not inform how to involve users, clients and stakeholders in the process, it does not even counsel methods and/or artifacts for identifying and assessing the design.

The focus on clients and on developers is clearly identified in agile methods [23][136] such as SCRUM [215] and extreme programming [22]. Figure 4.f presents a look at glance of the SCRUM development process also called skeleton. In that process, requirements issued by the client are recorded in a product backlog which is known of all the members of the development team. The work required to produce the software product is organized around several short cycles (the so-called sprints) of three to four weeks. Each cycle focus on one functionality described in the backlog and it encompasses steps of development and evaluation of the software with the clients. Clients are kept in the loop and adjustments can be done at the pace of sprints. The SCRUM method also introduce guidelines to organize the work such as the 24 hours cycles (i.e. the daily SCRUM) which are aimed aim synchronizing the activities and improve the communication among the members of development team. Agile methods became fashionable in the last years probably because short cycles of development and evaluation match clients and developer's needs from fast feedback and visibility on the progress of the project. Nonetheless, the add-valued for improving the overall quality of product is debatable. As far end-users are not the focus of the process, it is difficult to claim that the use of agile methods per se has an impact on the usability of the final software [40]. Moreover, as the underlying principles of agile methods refuse the use of models, artifacts and methods for specifying the software, they leave developers of no means for controlling the reliability of products and even development progress might be hard to follow beyond the advancement of tasks in the backlog.

Models that describe information about the system to be built can contribute in many ways to the development of computing systems for example by reducing ambiguity of requirements, documenting the design and support the communication among members of the development team[2] [64]. In the last decades numerous modeling languages have been proposed to support the design and the development of complex software systems. Quite often, a method featuring a development process follows the description of modeling language. The Rational Unified Process

[2] See chapter 4 for further information about models for engineering interactive systems.

(RUP) is an example of model-based development process that make extensive use of models described using the Unified Modeling Language (UML) [122]. In RUP, use case models that provide a functional view of user requirements are central because they are used to support subsequent development of other models describing the inner system core (ex. class diagrams, state diagrams, etc.). The RUP process consist of iterative cycles of four main phases where each phase has one key objective and milestone at the end that denotes the objective being accomplished. RUP phases are related to the so-called disciplines that are actually the tasks to be performed. The relationship between phases and disciplines can be visualized in a RUP hump chart as illustrated by Figure 4.g. Nonetheless, it would be inaccurate to consider that the RUP provides a seamless flow and consistency between models just because it is advocating a series of iterations in a well-organized phase plan [183].

The research around model-based approaches started with the 80's computer-aided software engineering (CASE) tools. However, after the publication of RUP in 2003 a multitude of development processes based on models have been proposed and diversely categorized [199] as: Model Driven Engineering (MDE), also called Model-Driven Development (MDD), when models are considered the primary artifact of the development process and used to (semi)automatically generate the implementation; the term Model-Driven Architecture or MDA [55] is the OMG's particular vision of MDE that relies on the use of OMG standards; and Model-Based Engineering (MBE) or Model-Based Development (MBD) when software models play an important role although they are not necessarily the key artifacts. These terms are prone to confusion and the discussion between the advantages and inconvenient of each model-based development paradigms has been fierce in the industry and in the research community. It is interesting to notice that discussion also involve partisans of agile methods that are against the use of any model.

Model Driven Engineering [213] is based on the combination of two main technologies: domain-specific languages (DSL) whose function is to formalize the application structure, behavior and requirements within a particular application domain; and transformation engines and generators which are tools dedicated to analyze certain aspects of models and then synthesize various types of artifacts which ultimately include the source code of the final application. MDE approaches employ three types of models: platform independent models (PIM) which should be combined with platform models (PM) and then transformed to obtain the platform specific model (PSM) which serve as inputs to transformation engines and generators to build the final application. On one the main criticism is that to get the full benefits of MDE, everything should be a model, which for interactive systems implies to describe humans and their surrounding environment. Moreover, MDE approaches promote the use of information contained in models to automatically generate the user interface of interactive systems. Whilst the generation of the user interface might be suitable for a few applications, mainly those which are based on simple interaction such as filling Web forms, we claim that it might remove all creativity that could be associate to the development process [159]. Moreover, model-driven development processes focus on models, which dramatically implications for fostering the communication with client, users and other members of the development team. Indeed, rapid prototyping disappear in that process because the construction of a prototype also require modeling, which slow down the whole process.

Any development process that relies on models but does not follow the generative path imposed by MDE could be classified as MBE. We consider MBE much more flexible because it does allow to combine models with informal artifacts and it favors prototyping activities, which we consider the most appropriate approach for building interactive systems. Therefore, many modeling methods can be referred as MBE and the utility of such model-based approaches for engineering interactive systems is conditioned by the inner characteristics of models (such as scope and expressiveness power of notations) as well as the way models are meant to be used (i.e. details about the method) along the development process.

One of the trickiest problems for most of developers is to test the code produced. Testing the software demands considerable effort but they this is downright dangerous to skip it [150]. Test-driven development (TDD) [24] is a software development process that relies on the repetition of a very short development cycle where, at first, the developer writes a failing automated test case that defines a desired improvement or new function, then produces code to pass that test and finally refactors the new code to acceptable standards. An overview of this approach is illustrated at Figure 4.h. TDD is related to the test-first programming concepts of extreme programming, begun in 1999, [22] but more recently has created more general interest in its own right [24]. TDD is primarily a developer's tool to help create well-written unit of code that correctly performs a set of operations. In order ensure that requirements are well-defined, it has been proposed another process called Acceptance Test-Driven Development (ATDD) which is a tool supporting the communication with users and the client. ATDD is also sometimes called Behavior-Driven Development (BDD) [11]. The overall idea behind ATDD/BDD is to write structured narratives and scenarios using plain English language which can be understood by all. Conversely to TDD, ATDD/BDD does not require automated testing but some specialized tools can covert ATDD/BDD scenarios into TDD scripts. The practices of development are worthy of mention because they help developers to ensure that they code is correct with respect to the tests specified. In this sense, the practice of testing the code might ensure overall quality of the software code but does provide a low-level view of the overall activity of the software development. Another limitation factors is user's requirements are only described through ATDD/BDD scenarios, which can hardly take into account all the complexity (and possibly the multiple execution paths) of user interaction with the system.

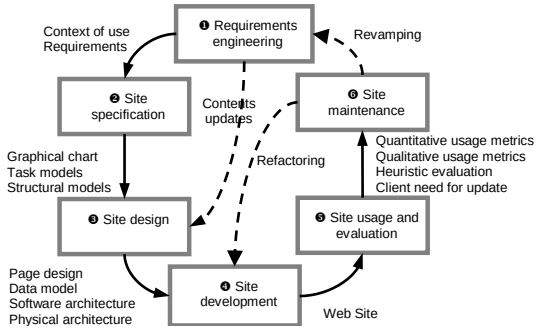
2.3 Diversity of development practices across application domains

Some application domains have idiosyncrasies that require the development of domain-specific development processes. Figure 6 illustrates the diversity of development practices in four applications domains: Web sites, games, airborne systems and safety systems.

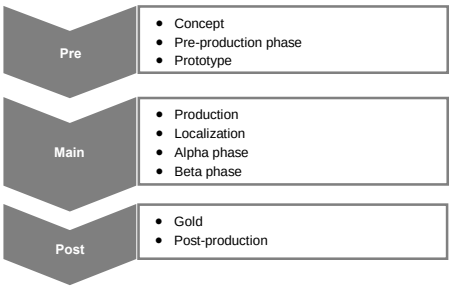
Many application in the Web domain are unique in the sense they require regular maintenance to update contents, refreshing visuals... and all these changes affect the user interaction with the system. The development process proposed by Scapin et al. (2000) [208] recalls that Web sites continue to evolve even when the first cycle of development is complete. An adapted version of the Web development process shows in Figure 6.a three paths for dealing with the evolution of Web site: updating contents (which affect the design of page pages), refactoring (with changes in the code) and revamping (when new requirements are introduced leading to a major new release).

Quite often, a successful game development company does not disclosure any information about the software development process used. Novak [165] reports on a mixture of concepts and methods that support iterative development for games featuring three main phases. Figure 6.b presents an abstract view of a game development process. We do not have represented the iterations but we must distinguished that, in the pre-phase the iterations occurs in a typical UCD life cycle using prototypes, in the main phase the interactions are similar to the V-model and, in the post phase the iterations are closer to the agile processes. It is also interesting to notice that this process include steps that we don't see in other applications domains, for example: the localization (when games that are aimed to be delivered to different markets), alpha phase (when the a game is playable from start to finish allowing different evaluation methods to be applied to better understand aspects like fun, playability and user experience) and beta phase (whose main goal is to fix bugs and promote fine-tuning of the user experience). Games development might have some additional milestones for example, when the product is developed for special consoles, the release candidate is tested and evaluated by the game console manufacturer.

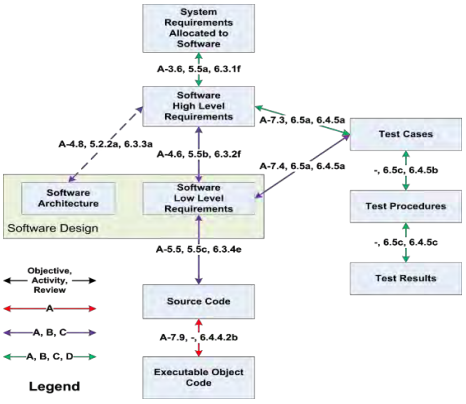
In safety critical systems several quality such as reliability, fault-tolerance and security, deeply influence the development process. But usability should also be regarded as a critical factor especially when 80% to 90% of accidents in industry are attributed to human error [206]. One key element of the development process of safety critical systems is the certification phase that involves an external organization who is charge of assessing the process. Figure 6.c presents a safety process which include the details about the certification activities. It is important to note that the certification phase is more prominent when the system involve citizens. A typical aspect of safety critical systems is the fact that standards processes are available and should be careful followed. Figure 6.c describes the standard DO-178C for airborne systems [191] while Figure 6.d presents another standard, the IEC 61511 dealing specifically with safety issues.



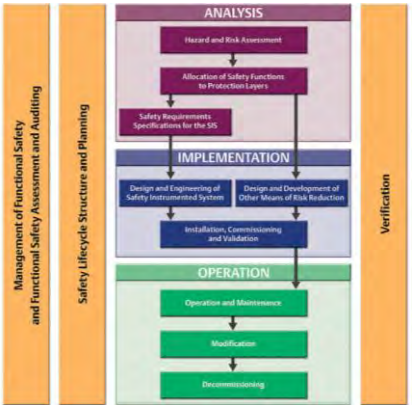
a) Life cycle for Web applications, adapted from (Scapin et al., 2000)



b) Game development process, adapted from (Novak, 2008)



c) Software Considerations in Airborne Systems and Equipment Certification (RTCA DO-178C standard, 2012) by Steven H. VanderLeest, is linced under CC BY-SA 3.0



d) IEC 61511 Functional safety - Safety instrumented systems for the process industry sector.

Figure 6. Example of domain-specific processes: a) Web, b) games, c) aerospace and d) safety system.

2.4 Development processes for engineering interactive systems

Nearly all software nowadays is operated by humans, most of development process do not focus on users and their activities [63]. This is the case of the waterfall model [203], the spiral model [34], the V-model [80] and RUP [122] that are centered on the system and do not contain any step related to understating human activity. This is also the case of the standards of DO-178C [191] the IEC 61511 [107] which are concerned by the certification. Agile methods such as such SCRUM [215] and TDD [24], mistake users by clients, which from a user-centered design perspective is misleading as client do not necessarily interact with the system they contract. Whilst clients might provide important requirements, end-users are more likely to inform the design with requirements grounded on their daily activities. The Start model [102] and the ISO human-centered design process [109] promote the use of prototyping and evaluation as a mean to keep users in the loop and this practice is borrowed by in the domains of Web [208] and games [165]. Nonetheless, many development process models fails to describe how developers should make the transition from understanding users' needs to building suitable products. It is interesting to notice that the development of interactive systems requires the interpretation of users' requirements (which are often informal) and their subsequent translation into some kind of artefacts (such as models and/or code) that should be used for building the systems. Thus, if developers don't have the appropriate tool support and the methodological guidance, they might fail miserably into the building suitable products. The focus on user analysis, prototyping and evaluation are traditional steps in user-centered design processes [63]. In addition to that, we understand that the development processes for engineering interactive systems is an activity driven by humans too, for that members of the development team should collaborate among themselves and use appropriate tools [242].

In previous work [170] we have examined the discrepancies between development process for games and safety critical systems. Despite of the specificities of the two domains it is surprising to find a set of similarities: both domains share the approach that usability as a factor of the interaction with the system might be less important than other software quality factors. Both domains have a set of additional development phases and usability evaluation is typically quite heavy in terms of infrastructure. Being interested in the domain dependent adaptation of usability evaluation, the domain of safety-critical system and the domain of games are promising candidates to understand how usability evaluation methods must be adapted to better fit the domain.

We suggest that the comparison between domains might help to improve development process, although it doesn't seem realistic to expect that a single development process model could deal with the diversity of constraints across diverse domains and the cultural practices already in place. Our experience in the field, lead us to suggest that some activities performed during the development process can help to engineer better interactive systems:

- *Analysis of the users* of the system to be built. Many methods exist for knowing the users (see chapter 3). But the users and their needs evolve so that such as analysis should be a continuous activity in the development process rather than a snapshot phase in the beginning of the process;
- *Prototyping the user interface*. We assume that prototyping is a cost-effective mean to support the exploration of many design alternatives for the user interface. Prototypes are excellent communication tools that can be used as a *lingua franca* with users, clients and members of the development team. Iterations with prototypes has been proved an efficient strategy to tune requirements for the user interface. This recommendation implies that the development process must include iterations around prototyping activities until the design is considered suitable for implementation;
- *User interface modeling*. Appropriate models can help to foster the communication of users' requirements and promote a smooth transition between informal requirements and artifacts used to represent the users and the system to be built. For example, task models, have been demonstrated very helpful for describing users' tasks (see section 3.1) but many other models describing the aspects of interactive systems such as dialog models can also fit the bill (see chapter 4 for a full account about models for engineering interactive systems). Other good reasons for using model-based approaches include: to automation of tasks (guiding for developers), ensure the transition of artefacts between the phases of the development process, support the certification and verification of artifacts produced along the processes.
- *Evaluation with users* of prototypes and/or final systems. For that users should be allowed able to face the prototypes and tell whether or not the proposed solution fits their needs and expectations. It is important that empirical information obtained from users could be traced back to the user interface and, as much as possible, to any other artefacts used to produce the system.
- *Guiding the development team* throughout the process. We assume that understanding user's needs is an important thing, but the development team might also be guided during the translation of user's needs into the artifacts required to build the systems. For that, the development process must include a detailed description of activities they should undertake to that the artifacts (and the final systems) correspond to the users' needs.

2.5 Contributions for improving development processes of interactive systems

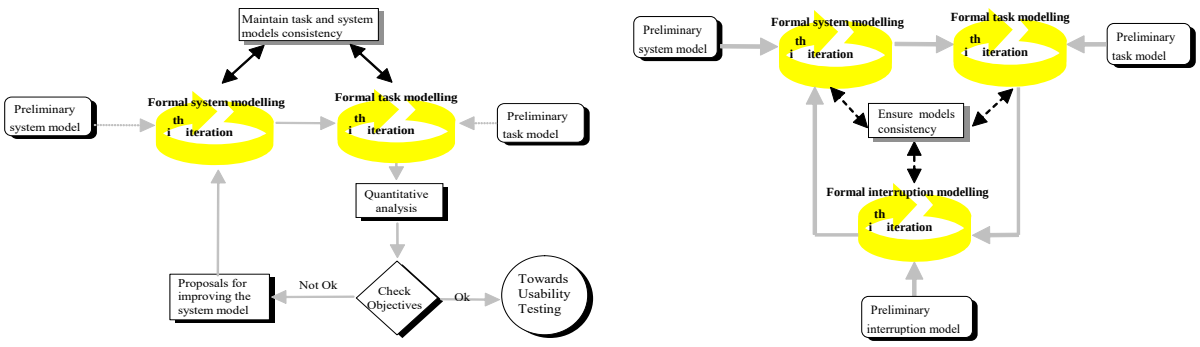
As far as the development process is a concern, my contributions focus on improving existing development processes rather than proposing completely new ones. The rationale behind is that the development processes models are aimed at explaining how problems can be solved; thus if activities required to accomplish a job are not described (or incompletely described), people might forget to do the job or do it in unexpected ways. In every project which we have been involved with, we looked upon the procedures usually applied in domain and then try to make them understandable as much as possible. In doing this, we try do not neglect solutions that might exist in other application domains, as discussed in [170]. Hereafter I present three examples of our contributions in the matter. The first contribution refers to the development of micro processes that are aimed at helping to ensure the coherence between models in a model-based approach [139]. The section 2.5.1 start by explaining the role of macro and micro development processes for engineering, and then explain two micro processes. The second example describe how we have integrated training activities as part of a model-based development process for safety critical systems. The last example investigate development process models for describing the collaboration between Web site owners, third-party developers of Web augmenters and end-users.

2.5.1 A micro processes for assessing the compatibility between interactive system models

It is interesting to notice that development process models have diverse degrees of details, for that we propose a distinction between macro-steps (that provide a global overview about the process) and micro-steps (that are aimed at providing details on how to accomplish specific tasks in a particular step) that guide developers along the development process of interactive systems. Figure 4.b presents an abstract view of the development process which is organized with two levels of details encompassing macro steps and micro steps. The so-called micro-steps refers to part of the process used to inform developers which deliverable should be produced, how problems should be solved in a systematic way, and how quality can be controlled in specific steps of the development process. We call macro-steps those that are used to address parts of the process that are aimed at providing a coherent view of whole without providing details of how tasks should be performed by members of the development team. Development process models presented in section 2.2 are good examples of process that only contain macro steps. Those processes provide a big picture but they lack of details for describing how activities should be accomplished, which is the role of micro steps.

Figure 7 illustrate two micro processes that should be understood as a sequence of micro steps that can be used in a model based-approach. Figure 7.a presents a micro process that is aimed at maintaining the consistency between task and systems models and Figure 7.b presents its variation when interruptions models are used to assess the impact of interruptions. Both processes are generic enough to accommodate diverse modeling notations. Moreover, it does not impose any particular method but they describes the steps that are considered necessary to get the job done. The micro process shown in Figure 7.a have been demonstrated flexible enough to allow its application to the development of many systems that require integration of tasks models and systems models such model-based training [141]. The micro process shown in Figure 7.b recalls that, interruptions might affect the performance of both users and the systems, for that an interruption model must be added to simulate the disruptive effect of interruptions when conceiving interactive systems [173][259]. The micro process are not competing and can be jointly used in conjunction of a macro process that is also built upon the premises of model-based development of interactive systems.

We assume that macro and micro steps are complementary in a development process. Currently, there is very little work trying to analyses the individual function of micro steps in development process. A few examples exist, but they are often associated to a particular modeling method which difficult the analysis and generalization of process. We suggest that the articulation between macro and micro steps is certainly an interesting path that should provide insights for future research in the field of engineering of interactive systems.



a) compatibility of task models and system models b) compatibility of tasks, system and interruption models

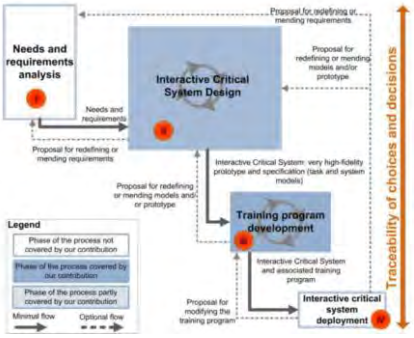
Figure 7. An iterative model-based design life cycle using both task and system models.

2.5.2 Including training as part of the development process

Operation of safety critical systems requires qualified operators that have detailed knowledge about the system they are using and how it should be used. Instructional Design and Technology intends to analyze, design, implement, evaluate, maintain and manage training programs. Among the many methods and processes that are currently in use, the first one to be widely exploited was Instructional Systems Development (ISD) [229] which has been further developed in many ramifications and is part of the Systematic Approach to Training (SAT) [106] instructional design family. One of the key features of these processes (at least when they are refined) is the importance of Instructional Task Analysis, particularly the decomposition of a job in its tasks and sub-tasks in order to decide what knowledge and skills must be acquired by the trainee.

In order to design such training programs and thus to improve human reliability we have proposed a systematic approach using model-based approaches currently used for interactive systems engineering. This work has been developed as part of the PhD thesis of Célia Martinie which I have co-supervised (2011) [137]. A synthesis of the proposal is described in Figure 8 where we explain how task and interactive systems modeling can be bound to job analysis to ensure that each trainee meets the performance goals required. That development process model integrates steps commonly present in process used in the HCI and safety critical domains, including: i) tasks analysis and task modeling as a reference model that binds all phases of the development process; ii) prototyping and subsequent formal modeling of the interactive system; iii) quantitative evaluation of user performance; iv) development of training program as part of the development process; v) traceability of requirements. One of the specificities of this proposal is to build the training program based on artifacts also used for building the system such as the task models, the formal specification of the system and the user interface prototypes. This development process has been duly illustrated in a case study for building a training program for operators of satellite ground segments, which is based on and compatible with the Ground Systems and Operations ECSS standard [141].

a) global view



b) detailed view

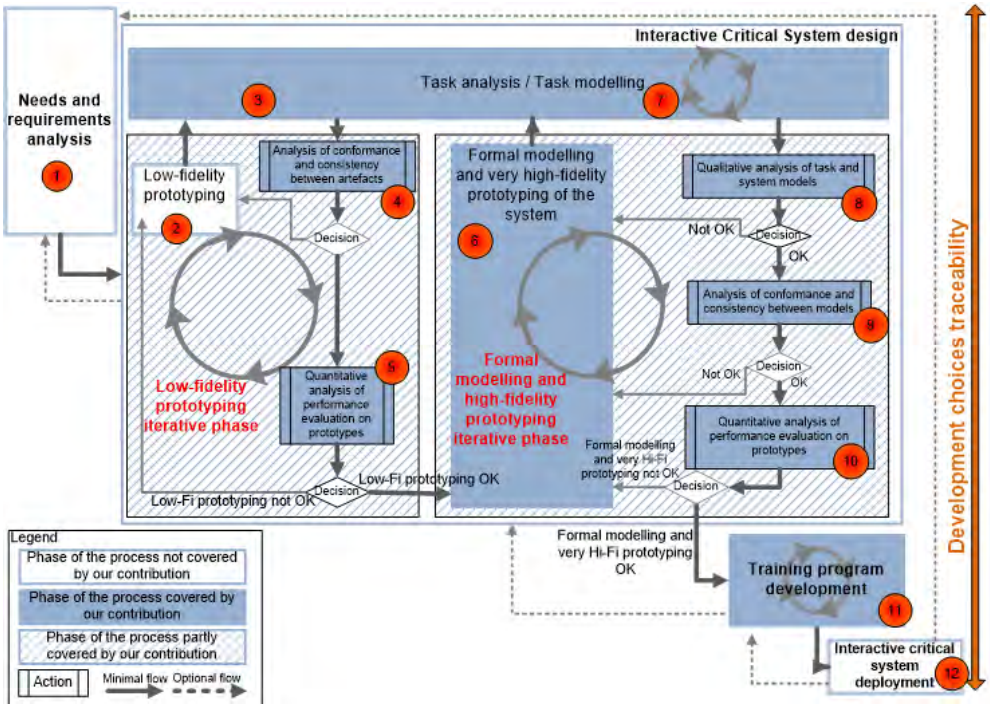


Figure 8. Development process for critical systems including training activities (Célia Martinie. 2011 [137]).

2.5.3 Development processes for integrating diverse expertise

The development of Web sites is a complex and time consuming process that usually involves different kind of expertise from individuals who must coordinate their activities to achieve an application that will fit to evolving needs of a (often) large and varied audience [256]. But in the last decades, we could observer the introduction of Web components (the so-called Web augmenters) that can be developed independently of the Web site and then applied to existing Web sites to adapt the original contents and/or the design. The PhD thesis of Sergio Firmenich (2013) who I have co-supervised at the Universidad de La Plata (Argentina) [71] investigates the development of Web augmenters. That work not only includes a complete study about the development processes for building Web augmenters [73][74] but also describe the development processes describing how to combine Web augmenters to orchestrate complex procedures [75][76]. For the work presented at the International Conference on Web Engineering (ICWE 2011) [74] we receive the best paper award.

It is interesting to notice that the introduction of Web augmenters reduce the control that developers and Web site owners had over the content delivery to the users. Indeed, web augmenters can be built by third party developers and installed by end-users without the consent of Web site owners who are excluded from the adaptation process. Thus, more recently we have started to investigate strategies for introducing Web augmenters in the practice of traditional Web development. We have found out that all existing adaptation strategies remarkably fail to provide collaboration between actors involved [72]. For that, we present a negotiated approach for Web adaptation which relies in three basic principles: first, all actors must find advantages in the collaboration; secondly, actors must collaborate; last but not least, tooling is essential to incite actors to collaborate. The negotiated approach also implies that a kind of commitment can be reached and for that, actors must collaborate. Close collaboration and commitment often demand the implementation of coordination and communication tasks, which require additional resources (in particular in terms of time and cognitive effort to maintain relationships running). To prevent that additional coordination and communication tasks come to plunge the advantages of such collaboration, the negotiated approach proposes that actors can work independently (as much as possible) and only perform the tasks for which they might foresee a direct advantage. To support such as a light-tight collaboration, we rely on a distribution of task among the participants and the existence of appropriate tool support as presented in Figure 9.

The work presented in [72] raises interesting research questions that are of practical and theoretical importance for the development process models in the context of Web applications, such as: In which extension end-users are able to compromise and collaborate with Web site owners and communities of coders? How to describe in the processes user's needs that require adaptation of Web sites evolve overtime? How to prevent those communities of coders can damage the presentation of Web site contents? How to improve trustful relationship between users that have different interests in the adaptation of Web applications? How to ensure long term compatibility between Web sites and external scripts? Whilst these question are quite specific for the development of application in the Web context, we don't exclude the possibility that they might influence development process in domains such as mobile applications and games.

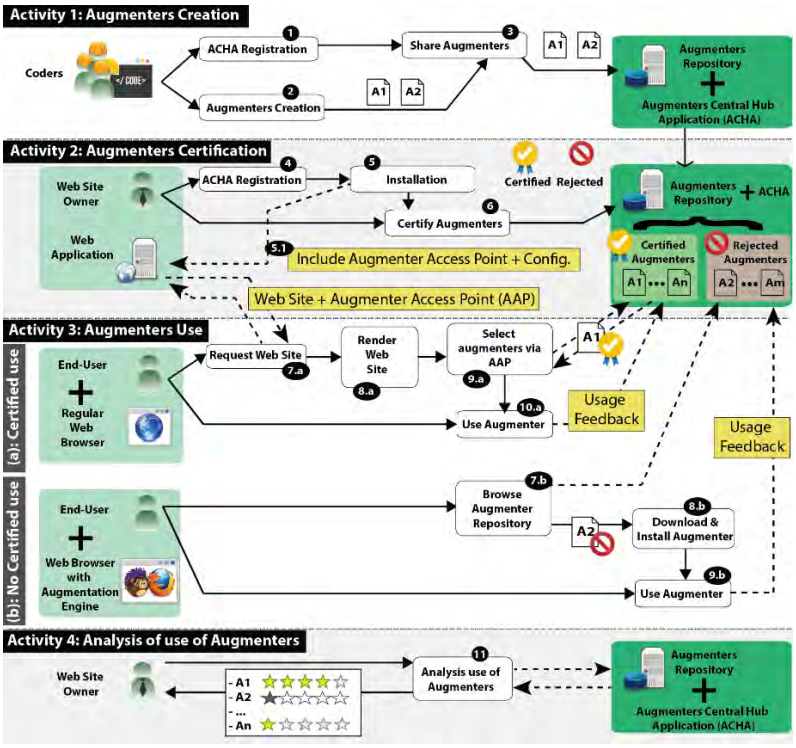


Figure 9. Tasks allocation in a negotiated adaptation based on a Web augmentation approach [72].

2.6 Challenges for improving processes for dealing with Web applications

In spite of the apparent facility to build Web pages given by current visual environments, the development within the Web is complex [243]. The main reasons include the fact that Web applications require regular maintenance in order to update page content, to follow a particular business workflow, to include new features for supporting new tasks and/or users, and so on. Besides, Web development is often performed by several people (from different backgrounds such as graphics designers, marketing, and editorial) at a time and thus in a parallel manner. Another important feature of a Web project is the time to delivery or to update that can be as short as few hours. Designers can change their design almost immediately, which makes Web development projects highly evolutionary in nature. Moreover, the term Web applications is fuzzy in the sense at might refer to many different kind of applications over the Web (for example such as Web sites, service-oriented Web sites, data-intensive Web applications and Web information Systems…) and different types of Web applications might different types of development processes. Thus the challenges for investigating development process for Web applications, in the broad meaning of the term, are many. Nonetheless, we suggest that there are four major challenges that affect the development process in the domain: the diversity of platform on which Web applications can run, the need of frequent updates, the diversity of audience, and the diversity of participants in the development process with does not exclude the possibility that users might directly contribute with contents and developing parts of the application in an end-user development approach.

One of the main challenges for building usable Web applications is that the rendering of the application delivered to users might change across the client Web browser. This problem exist since the origins of the Web. A minimum of interoperability is ensured by W3C standards such as HTML 5 [239] but it does not solve everything. Indeed, interoperability has decreased with the diffusion of smartphones that integrate a lightweight Web browser. To make Web applications run on smartphone, developers have to provide a mobile version of the Web site. The term responsive design was coined to address the issues related to the adaptability of Web application across platforms, from desktop to smartphone. Currently make strategies for developing responsive Web application have been proposed [264]. Nonetheless, very little has been investigate so far with respect to the development process leading to the production and the maintenance of responsive Web applications.

Web applications need frequent maintenance, and every time developers touch the code source they might introduce errors, usability/accessibility flaws. The existing solutions to this problem enforce completely different approaches for the development process. One of the solutions is to employ tools based on guidelines inspection to check whether or not errors have been included after each update [198][268]. Another solution consist in to employ an integrated development environment (IDE) for developing Web applications which will prevent that the code contain errors [267]. Both solutions might work but the inner development process enforced by them have a huge impact on the activity of people involved.

It is very difficult to seize the audience of Web sites at first. Then when an application becomes successful there is a huge amount of users adopting it (we might call it viral), at least for while… With new users, comes new requirements that have not foreseen in the first cycle of development. To keep users connected Web site might be tempted to integrate personalization features as part of the development process. Current Web personalization approaches usually suffer a boundary problem, since most, if not all, work in an individual application basis. When a user needs to deal with two or more applications for performing a particular task, he will face differences in the personalization approach for each of them (if any). Another drawback of personalization mechanisms is that, specified by application's developers, do not necessarily may foresee the requirements of every single application user. Currently, we don't have any development process that guide developers towards the resolution of problems associated to the introduction of personalized features in Web sites. We suggest that this problem could be addressed in the future by looking at existing solutions in the field of gales;

One of the most interesting facets of Web evolution is the end-users interaction with Web content. At first, users could only browse through contents provided by the web site. Later, users could actively contribute with content by using tools (e.g. CMS, wikis) embedded into the web site. More recent technologies provide users with tools allowing them to change the way Web content is presented. For example, using visual Mashups, users can compose content hosted by diverse web sites and they can run Greasemonkey scripts to change third part web applications by adding content and/or controls (e.g. highlight search results in Amazon.com which refer to Kindle) [74]. These tools built under the concept of Web augmentation [71] extend what user can do with Web content but they provided limited support to tasks that require navigation on multiple Web sites. Regardless the extension in which end-users contribute with contents and/or with the personalization of the Web site, we should not neglect the fact that they are active participants that must be considered by development process. Thus development process models should be able to describe in which extension end-users contribute, how and when their participation is required along the process, how conflicts can be solved when different actors of the process (ex. marketing, designers and end-users) don't agree with the development… We assume that if end-users are recognized as actors taking place in the development process, it would be possible to envisage new strategies of development of interactive systems for the Web.

2.7 Research agenda

Modern interactive systems should be characterized as complex systems for (at least) two main reasons: first of all, interactive systems large entities comprising many components with the capability of handling multiple threads of dialog with users and, in many cases, through many communication channels [66]; moreover, the development of interactive systems require diverse skills (ex. on software development, graphical design, databases, network communication, human factors…) for that a multidisciplinary team must to cooperate [256]. Faced to such complexity, members of the development team need help to coordinate their activities and develop quality products in an economic and timely fashion. For that, we cannot conclude this presentation of development processes models without recalling that software development is in fact a human-centered processes where creativity and autonomy play a crucial role [82].

For many years software development process have been considered socio-technical systems, where organizations and humans have a key role to be supported by technology; thus the improved performance is the result of a proper interplay between social aspects and technology [226]. As a consequence, most of activities occurring during the development of interactive systems cannot be fully automated. There are, indeed, certain activities where prescriptions and automation are useful, but it is impossible to force the adoption of a rigid and predefined patterns and procedures. In general, the development of interactive systems is a complex process in which is not possible to enforce a step-by-step compliance with a predefined model. This is one of the main drawbacks against the canonical software development processes presented in this chapter.

We assume that the research aimed at improving development processes can only be achieved by understanding technological constrains and socio-technical barriers that might prevent people to change the way they actual deal with the development of interactive systems. However, we also assume that comparing practices of development across application domain might help to find suitable solutions for enforcing a change. In a short run, our research is concerned by the investigation of how user-centered development processes and in particular the following topics:

- Integrating model-based development using prototypes in a User-Centered Design process. This topic is covered by the PhD thesis of Jean Luc Hak, started in September 2015, which I act as co-supervisor. The primary goal here is to investigate tool support for the creation of prototypes that are formally described using a user interface ontology. We are concerned by the underlying process that guides the prototyping activities, such as the iterations among cycles of design and evaluation, decision-making based on prototypes and prototypes evolution that lead from initial prototypes to final products. Doing so, we expect that developers would be able to justify the final prototype before the implementation, to monitor and document the prototyping activities by covering each prototypes produced. We also expect that this approach would help to improve the communication using prototype for bringing new ideas, for assessing alternative solutions or for detecting problems in earlier phases of the development process. The ultimate goal of the thesis to improve the communication during the development process. We raise the hypothesis that we can help the development team by proposing better communication (annotations and traceability of design choices) and development features (versioning) that fits the UCD practice.
- Investigating the potential of cross-pollination of agile methods, and in particular test-driven development process (TDD), in a User-Centered Design process. This topic is covered by the PhD thesis of Thiago Rocha Silva, started in September 2014, which I act as co-supervisor. This PhD thesis is concerned by methods that could make a compromise between the advantages of agile methods (ex. fast speed in the development process, active involvement of client and end-users, etc.) and model-based approach (ex. unambiguous specifications, automated software test against pre-defined requirements, etc.). For that purpose, the PhD thesis explores the use of methods for formalizing end-users requirements which could be attached to artefacts (such as task models, prototypes…) of the user interface of the software being developed. The specific goals of this PhD thesis include: i) to develop a model-based approach for supporting automated or semi-automated verification of end-user requirements; ii) to support the communication and collaboration between all participants in the development process of Web applications, including designers, developers, clients, etc., and iii) to investigate mechanism for using models in fast cycle of the development processes, as defined in agile methods. The focus of this work is the members of development team. We expect to find solutions to help them to cross-check end-users requirements along the many artefacts they use to build interactive systems.

We are going to pursue our research by comparing strategies for developing interactive system across diverse domains. We truly believe that by understanding the good practices of developers in a particular domain can help to find generic solutions to problems of software development, and in particular interactive software development. We don't expect to define a revolution macro development process that would fit all the idiosyncrasies of all projects, but we hope that we could find solution to micro development process that help the development team to understand the problems they are facing and how to solve them. In a long run, we expect to be able to characterize such as micro development process and then propose specific usages across application domains.

3 Users and the engineering of interactive systems

Users of interactive systems are human beings who, conversely to computing systems, cannot be programed to systematically behave in a particular way. So if interactive systems are aimed to be effective and reliable they should be able to cope with the idiosyncrasies of users' abilities, wishes and needs. Knowing the users and knowing how to employ such knowledge about users during the development process of interactive system is therefore essential. This chapter is aimed at describing models for representing the knowledge about users and how I have employed them in my research. The chapter starts by a view at glance of the literature followed by my personal contributions in the matter and a research agenda.

3.1 State-of-the-art on approaches for representing the knowledge about users

In the early eighties, the birth of the field of Human Computer Interaction (HCI) witnessed a vivid interest in models of user cognition, knowledge, and problem solving strategies. A clear ambition of that era was to shape design processes around the elicitation of such knowledge and its application in the design of user interface software. In the subsequent years, several approaches have been proposed to seize the human characteristics that might contribute to the development of interactive systems. Figure 10 provides some dimensions that we use hereafter to explain the most frequent approaches found in the HCI literature. This classification above includes *analytical models* (e.g. *human models* and *task models*) that are used to explain and/or reason about human behavior and activities, *stereotype-based models* that encompass techniques for describing humans as users of interactive systems, and *empirical models* that cover several dimensions used to portray users.

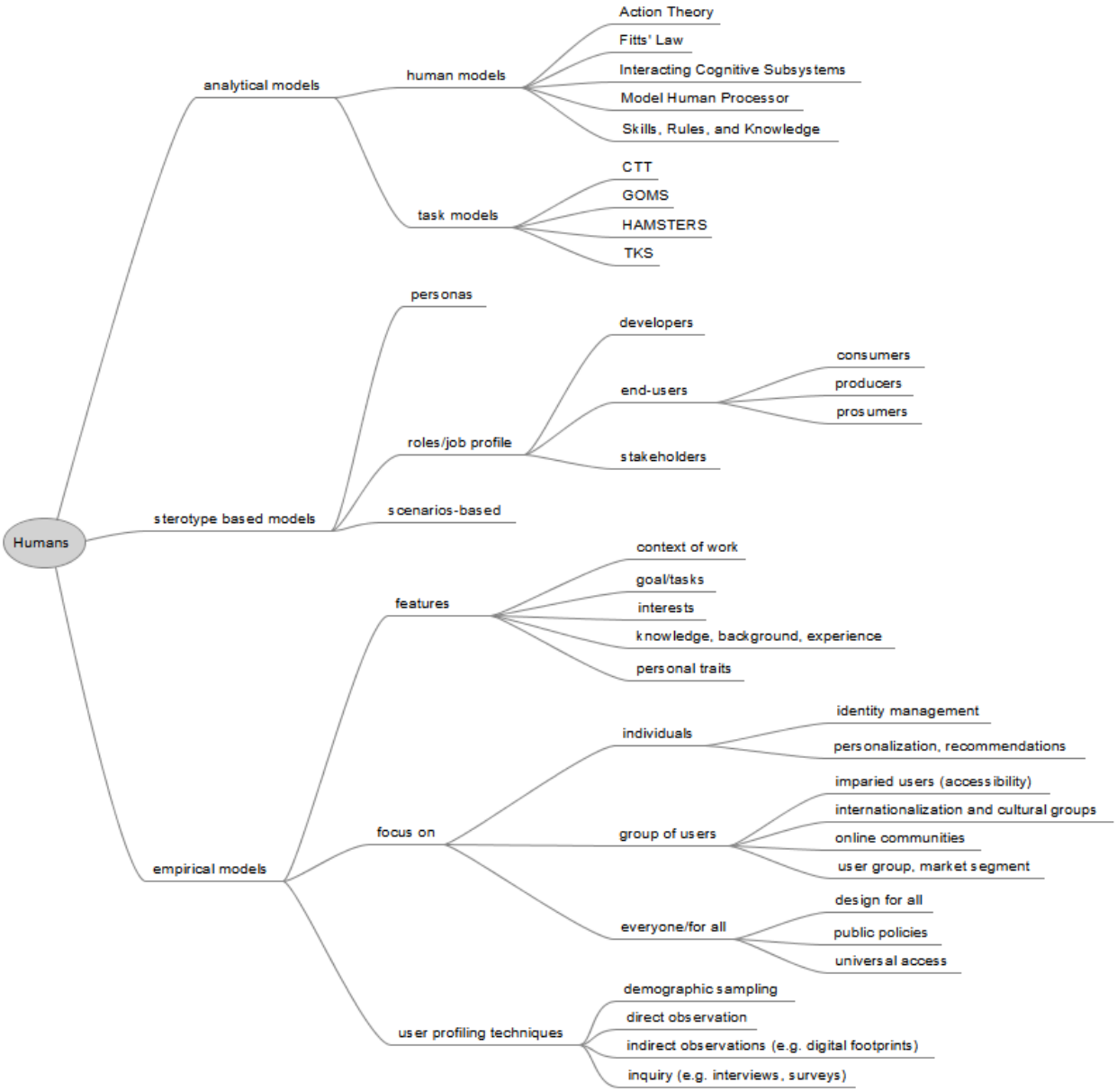


Figure 10. A classification of approaches for representing users' characteristics.

It is interesting to notice that the dimensions presented in Figure 10 are orthogonal and, to be effective in real-life projects, they are often combined; for example a subset of *features* can be used to portray either *individuals* or *group of users* through the means of *personas* whose tasks can be represented by *task model*. Notice that some branches are expanded to illustrate subsidiary questions related to the design of interactive systems, for example, *focus on individuals* might lead to questions related to *identity management* and *personalization*. The list of approaches discussed in this chapter is not exhaustive but it covers the most prominent contributions to the engineering of interactive systems.

3.1.1 Analytical models

Human models and task models are part of the theoretical core knowledge in HCI. These two categories of models share the focus on generic users (as human being) rather than on individuals, which are more generally the subject of profiling techniques.

Most of human models, such as the Model of Human Processor (MHP) [43] which has been influential to development of theories and task models such as GOMS [115], come from the research from Cognitive Psychology started in the 1980s. Fitts' law [76] is an exception, but its accuracy to predict human movement has been proved by experimental user studies. These and many other *human models* constitute the core set of theories used in HCI to explain cognitive process and user behavior triggered when users try to communicate with interactive systems [63]. A complete survey of human models is out of the scope of the present work. Nonetheless, we provide here a brief account of a few models (i.e. Norman's Action Theory [164], Interacting Cognitive Subsystems (ICS) [11] and Rasmussen's SRK model [60]) to illustrate and support the discussion of the relevance of human models to the engineering of interactive systems.

The *Action Theory* of Norman, or Norman's cognitive model [163][164], explains the gulfs that exist in the communication between users and systems; i.e. the execution gulf and the evaluation gulf. The execution gulf is the effort required for a user to express an intention in terms of commands or instructions. In other words, the gulf of execution is the difference between the intentions of the users and what the system allows them to do or how well the system supports those actions. The evaluation gulf refers to the way the results provided by the system are meaningful or understandable by the users, and in accordance with their goals. In other words, the gulf of evaluation is the degree to which the system or artifact provides representations that can be directly perceived and interpreted in terms of the user's expectations and intentions. Thus, if the system does not "present itself" in a way that lets the users derive which sequence of actions will lead to the intended goal or system state, or infer whether previous actions have moved them closer to their goal, there is a large gulf of evaluation. Overall, the gulfs of evaluation and of execution refer to the mismatch between our internal goals on the one side, and, on the other side, the expectations and the availability of information specifying the state of the world (or an artifact) and how we may change it. As suggested in [47], this model can be used as a premise for analyzing the semantic distance and articulatory distance created by the user interaction with a system.

Interacting Cognitive Subsystems (ICS) [11] describes how input and output signals conveyed by communication channels (e.g. audio, visual, and sensorial) are processed by human's brains and body. It is a comprehensive model of human information processing that describes cognition in terms of a collection of subsystems, each of which process different mental representations. There are three sensory subsystems (visual, acoustic, body state), four central subsystems composed of two structural subsystems (morphonolexical, object) and two meaning subsystems (propositional, implicational), and two effectors subsystems (articulatory, limb). These representational subsystems are supplemented by peripheral somatic and visceral response systems. Each subsystem also contains transformation processes to convert incoming data to certain other mental codes. ICS model takes into account the fact the information processing in subsystems are not symmetrical and they might suffer transformation. For example, visual information is not directly translated into the so-called propositional code but must be processed via the object system that addresses spatial structure. Although in principle all processes are continuously trying to generate code, only some of the processes will generate stable output that is relevant to a given task. The applicability of ICS is somewhat limited but it becomes relevant for analyzing low level (almost physiological) user interaction with multimedia and multimodal systems.

The *SRK model* [60] describes human performance in terms of *Skill-, Rule-* and *Knowledge-based* behaviors. Skill-based behavior refers to routine activities conducted automatically and do not require conscious allocation of attentional resources; such behaviors are triggered when human performance is determined by stored, preprogrammed patterns of instructions. Rule-based behaviors are activities controlled by a set of stored rules or procedures. The distinction between skill-based and rule-based behaviors depends on the attention and training level of each individual. Skill-based behaviors typically progress without conscious attention and the individual is seldom able to verbalize how performance on the behavior is controlled or explain the information on which their performance is based. Performance of rule-based behaviors is typically based on specific ability and the rules for performance can be verbalized. Knowledge-based behaviors are those in which stored rules no longer apply and a novel situation is presented for which a plan must be developed to solve a problem. In contrast to set rules, plans are often required to be changed based on the situation. As stated by Rasmussem [60] "*control tasks must be*

described in terms referring to human mental functions rather than system requirements". Attentional resources must be allocated to the behavior and, therefore, the performance of knowledge-based behaviors is goal-controlled. One of the implications of the SRK model is to highlight the fact that human performance is related to level of consciousness: in familiar environments human behavior is not goal-controlled which might imply in to human error due to the lack of attention whilst in non-familiar environments human errors more or prone to occurs by lack of knowledge [195]. Thus, interactive systems might exploit the users' knowledge about the task and the surrounding context to propose the appropriate feedback.

Whilst human models aims at explaining cognitive processes and general user behaviors, task models focus on the actual tasks users perform with a given system. A user task is defined as a goal together with the ordered set of tasks and actions that would satisfy it in the appropriate context. The relationships between tasks and goals are clearly described in Norman's action theory [163]. Task-based models are aimed at providing a representation of user tasks so that it can support the analysis of user activity. Task models have many possible uses for the engineering of interactive systems and they are better discussed in section 4.3 (*Important models for engineering interactive systems)*. The raison why we mention task models in this chapter is to provide evidence of the inner relationships between the representations of the knowledge about users that can be embedded to task models. It is important to recall that task models are supported by task analysis, which is widely recognized as a fundamental way to focus on the specific user needs and to improve the general understanding of how users may interact with a user interface to accomplish a given interactive goal [60]. Further information about task models can be found in chapter 4 but the fundamentals traits that ties together user tasks and to the study of users can be summarized as follows:

- Task models capture knowledge about the end-users' logic of use of an applications, which characterize tasks and users as interlaced concepts [63].
- Task models notations, such as TKS [116], CTT [197] and HAMSTERS [138], might include support the description of user's traits, even if in most cases it limited to the allocation of tasks to user's roles.
- Task models presents construct to operationalize human models; ex. HAMSTERS have been extended to explicitly represent procedural, situational, or strategic knowledge users need to accomplish tasks [143].
- Task models such CTT [197] and HAMSTERS [138] features tools supporting task simulation, so these models can be used to predict users' performance with interactive systems, which can be understood as an indirect (or simulated) knowledge about human behavior.

3.1.2 Stereotype-based models

During the development process of interactive systems it often necessary to discuss the alternative designs options with the whole development team at the light of users in the target population. For that purpose, human models reveal to be poor communication tools. Moreover, discussions oriented to real-life of people might be biased by either by the need of an individual who is not necessarily representative of the population, or by personal information that cannot be easily communicated without compromising privacy. In this, stereotype-based models such as *personas*, *scenarios* and *roles/job profiles* better fulfil this job as communication tools.

One of the most traditional approaches is to describe users in terms of roles/job profiles, or in other words the function users might play with interactive systems. Roles/job profiles modeling often include users' responsibilities, authorizations, tasks and required skills. This approach mask all individuals' features and it consider that user playing a role in the system (or organization) are interchangeable. Such as an approach can be applied at different levels and it can be used to analyze the role of participants (for example as *developers* and related participants of the development team, *stakeholders* or *end*-users) in the development process of interactive systems [256]. It is interesting to notice that roles evolve overtime; for example, the advent of Web 2.0 applications such as Wikipedia allowing user-generated content, the role of end-users as simple consumers of information has evolved to include features of content-providers. Some works, have also demonstrated that with appropriate tool support users can also change functions of application to support their needs [73].

Personas is another wide-spread technique in HCI to inform how users use a tool, product or application, giving a clear picture of how they're likely to use the system, and what they'll expect from it. Often used by the marketing community, it was popularized among technology people by Alan Cooper's book (1999) [53]. Personas are depicted as fictitious users and it includes a concise summary of his characteristics such as experiences, goals and tasks, behaviors, motivations and environmental conditions. When backed by careful statistical analysis of surveys, personas got credibility for inferring real users' expectations [153].

The term scenarios refers to a large family of techniques used to concretely describe the use of the future system by the target users [46][197]. Scenarios might feature narrative descriptions or sequences of activities (more or less elaborated according to the technique employed) of what users do with interactive system. The narratives might be fictitious or ground on true-life histories reported by users, for example during interviews. One of the key distinctions between scenarios and any model is that the former are grounded examples of specific experience, whereas models are more abstract representations of phenomena in the real world [224].

3.1.3 Empirical models

Empirical models focus on the analysis of individuals' characteristics in a population to create *user profiles,* which sometimes also referred in the literature as *user models* [115]. A user profile is a portrait of someone containing the most important or interesting facts about the individual. The motivation of building user profiles is that users differ in their preferences, interests, background and goals when using interactive systems. Discovering these differences is important to characterize and understand people that use (or are potential users of) an interactive systems.

Given the complexity of human being, the number of features that can be user to portray users is huge. Moreover, it is interesting to notice that the relevance of those features might vary according to the application in consideration [212]. For example, features such as topics that a user likes to read, newspaper user usually reads, and frequencies of reading are relevant for online newspaper whilst for a personal agenda application other features about users should come up first such as when users have holidays, work time schedule, the priorities of activity to the user, rescheduling habits… Nonetheless, most of user profiling techniques include the *context of work,* user's *goal/tasks, interests, knowledge, previous experiences* and *personal traits*.

The study of user features can be focused on *individuals, groups of* users or *everyone* in a population. User profiles focused on individuals aim at knowing specific user's characteristics and determine his/her need for *personalization* the user interface [39]. However, as the study of individuals involves real users, it also raises subsidiary questions related to *privacy* and *identity management* [197][261]. The *personalization* implies some kind of adaptation of the user interface to cope with the idiosyncrasies of a user profile [39]. Some *recommender systems* also can make use of user profiles to propose to contents and navigational links that better suits to a particular user [115] but not a not all adaptive systems are driven by user models for example, many adaptations proposed by plastic users interfaces [52] are driven the environment and the platform rather than by specific user's features.

In contrast to individual user profiles, *group profiles* aim at combining individual user profiles to model a group. It is implied that users in a group share the same needs, which is often the case for user suffering from similar disabilities, sharing the same cultural traits, or taking part of the same social network or online communities, or consuming similar products [1]. The study of group profiles become particularly relevant when it is necessary to make personalization and/or recommendations to groups of users rather than to individual users; for example to recommend visit for tourist groups taking into account characteristics of subgroups such as children and disable within that group [9]. More recent studies explore metrics that can be derived from user interact in social networks to portray group of users that share some characteristics that cannot be directly derived from users individuals characteristics. For example, Panigrahy, Najork and Xie (2012) [175] have shown that social affinity (a concept defined as an intermediate measure between the shortest path distance and the number of paths connecting a pair of user in graph depicting a social network) and approximated shortest path between two users had a high correlation with their profile and query similarities, and thus can be inferred through a social network by recommender systems when the target user profile is not known.

Nonetheless, it is also possible to identify features that can be present in a given population, and should be treated fairly as if shared by *everyone,* or in other words *for all* users. The focus on all users is deeply associated with the concept of universal access [126][146] and user interface for all [52], which are used to describe the user interfaces that are aesthetic and usable to the greatest extent (possible by everyone), regardless of their age, ability, or status in life. User interfaces for all users often focus on methods and tools allowing a better insertion of individuals in the information society, which immediate prompts to questions related to accessibility, human diversity, social inclusion and equality. The scope of the research focused on all and everyone is broad and sometimes difficult to seize with traditional methods used for analyzing user's characteristics and needs. Nonetheless, the debate on applications for all users has a strong impact on the definition of public policies [122].

The HCI community has been very prolific in the development of methods for observing and for assessing users' behavior and traits that could be used to support user studies, which means investigations that not only focus on users but also taken into account the use of interactive technology. The discussion about methods involving user testing during the assessment of interactive system is treated in chapter 1. Nonetheless it is worth to recalling that techniques for *user profiling* can be classified as *demographic sampling methods*, *direct observation, indirect observation* and *inquiry*. Methods based on *demographic sampling* try to extracting knowledge about users from data obtained by generic surveys that have been used to assess traits of a population [122]; this kind of methods are more often used in Artificial Intelligence. The other categories of approaches for user profiling are of particular interests to the HCI research as they focus on the observation of real users. Direct observation allows data collecting from users who are aware of being observed during a user testing or ethnographic studies [197]. To avoid any possible observer-expectancy effect, indirect methods based on the collection of digital footprints has also been explored in the literature [222] and in particular with Web applications [111]. Nonetheless, *inquiry* methods based on interviews and questionnaires are broadly used for allowing users to report information about them, which is a valuable source of information for constituting user profiles [128].

3.2 Contributions for understanding and modeling users

My work is focused on building interactive systems rather than deepen the understanding about human cognition and behavior. Notwithstanding, the development of interactive systems requires knowing the users. For that, I have been and avid consumer of any knowledge available about the users of the systems I was investigating and I have systematically employed (alone or in combination) analytical user models, stereotype-based models and empirical modeling techniques. Hereafter, I present three examples of how I have used these approaches for developing interactive systems, what are the shortcomings with existing approaches and what were the solutions we have developed to overcome these shortcomings. The first example concerns a holistic perspective about people that participate in the development process of interactive systems, i.e. users, stakeholders and the development team. The second one refers to the use of task models to support the representation of the knowledge about users; for that we have extended the task model to take into account the need of designer and developers that exploit tasks model to build the applications. The third example describes our attempts to organize user interfaces guidelines, which is a suitable way to systematize knowledge gained from empirical user studies.

3.2.1 Knowing users, stakeholders and the development team

Along users, stakeholders[3] and the development team[4] play an important role in the development process of interactive systems [258]. Historically, the need of having a good understanding of the users is clearly associated to the Norman & Draper's definition of user-centered design (UCD) which states that: "*...User-centred design emphasizes that the purpose of the system is to serve the user, not to use a specific technology, not to be an elegant piece of programming. The need of the users should dominate the design of the interface, and the need of the interface should dominate the design of the rest of the system*" [163].

Whilst UCD has been subject to multiple interpretations and some extensions have been proposed to define the degree of user involvement in the development process [78], it is still more generally refer to the people that use the final version of the system. However, design choices for a system might be affected by indirect factors such as requirements, constraints and needs issued from other needs of stakeholders and/or the development team. In previous work we have approached this problem by examining the tasks, the responsibilities and the roles played by all participants that contribute directly or indirectly to the development process. We have applied this analysis in the context of Web applications [256] and on the development of e-government applications [251][246].

In the case of Web [256] the problem is driven by the fact that different expertise and skills are required to build interactive systems deployed in the Web. For small Web sites the work is often done by a single person but for large Web project a multidisciplinary team is required (e.g. graphical designers, database administrators, client-side and server-side developers, webmasters, content providers, editor in chief, marketing staff, etc.). Moreover, Web applications are evolving in nature and should be regularly updated. Some of the updates can be planned in advance (ex. a conference web site that features a Web portal to advertise the call for contributions in the early phases, might become a live feed application to deliver information when the conference occurs, and then it be transformed in storage place for paper and contacts of participants once the conference ends) but others are contingent upon Web site visitors, for example, which often brings the need of personalization (e.g. Facebook has suffered numberless adaptations to cope with incoming users with diverse needs of connectivity, information, ages, etc.). What it is interesting to notice is that, very often, participants intervene at different phases of the development process and have little information about what the other participant do, thus mishaps and conflicting updates might occurs and comprise the overall quality of the application that is delivered to the end-users. By providing a method for modelling tasks and responsibilities, we sought to help to deepen the understanding about all participants in the development process and highlight the dependencies between tasks performed by different roles and/or individuals.

We have examined the relationship between end-users, stakeholders (ex. deciders, politicians and clerks) and the development team of e-government applications. End-users of e-government applications are citizens with universal needs for accessing interactive applications and they should receive fair treatment without any kind of group discrimination. Whilst administrative procedures (ex. paying taxes, getting copies of official…) should be accessible to all citizens, different levels of administrations (ex. national, regional and local administrations) share the responsibility for procedures; quite often, this leads to a multiplication of Web sites[5] which makes difficult for citizens to create an accurate mental model of services provided. We have investigated this phenomenon in France [251][244] and in Belgium [246] but the situation seems to be the same world-wide. The problem is not the lack of information about end-users (i.e. citizens) but the lack of coordination among development teams. This situation

[3] The term "stakeholder" is meant here to refer to people internal to an organization, administrative clerks and agents who are in charge of collecting and analyzing data provided by users and deciding the outcomes.

[4] The term "development team" designate here any person who ensure some kid of responsibility in the development process which might include programmers, graphical designers, editor in chief, project manager, etc.

[5] The French Ministry of the Economy estimates there are more than 10,000 governmental websites; 700 of them are managed by the national government, while the rest belong to regional and local administrations.

was illustrated in a field study performed at SmalS-MvM where we have found that many usability problems (ex. inconsistent implementations of design options across services deployed at social security Web portal) were mainly due to communication problems between stakeholders and the development team. Stakeholders work on the back end of the system, and as a consequence they have a different view on what is important for the application; they know a lot about a limited set of administrative procedures which refrains the understanding about user's needs for related applications. The problem is aggravated having different development teams in charge of different e-government services. Given the overall context, citizens are of little help as they only knows the services they have been using and they have a limited view about the administration in general. Once we have understood that a holistic approach was needed, we have proposed a user interface pattern language for e-government applications [188][190] that revealed to be a suitable alternative to solve the communication problems between the participants along the development process (i.e. end-users, stakeholders, development team) in that particular domain [187].

3.2.2 Task models the representation of the knowledge about users

I have systematically used task models as a suitable alternative for analyzing the users' logical of operation with the many types of interactive systems I was investigating (ex. Web applications, e-voting systems, ground segment systems, mobile apps…). Whilst investigation task model notations, I've found that despite the fact that there is an obvious relationship between the users and the tasks they perform with the system, a very few notations (such as TKS, GTA, MAD and CTT) provide constructs to represent users; even if such constructs are restricted to the allocation of a task to a particular agent (i.e. a user and or a system) [188]. In order to support a finer-grained analysis of users' activity whilst performing tasks, I have worked with my colleagues in the development of a task model notation called HAMSTERS [138]. HAMSTERS embeds constructs for representing features that might be require to understand users' capabilities for performing a task such as the impact of a task on user perception, working memory ad decision making. Further details about the notation HAMSTERs is given at section 4.5.2.

3.2.3 Management of user interface guidelines

The understanding of how human cognition and behavior is often the product of investigations of the uses and the users of interactive systems. Whilst user studies can be demanding, the knowledge gained from them can be systematized in the form of recommendations and or user interface guidelines that contain implicit information about users. During the development of interactive system, the main source of recommendations for designers and developers are indeed user interface guidelines. The sources of ergonomic knowledge for user interface design are diverse and the problems related to the selection of guidelines from different sources motivated a variety of tools for working with guidelines [232][233].

In previous work we have investigated the management of guidelines sources and, in particular, the occurrence of conflicting guidelines. Indeed, quite often designers have to combine different guidelines sets in order to address the idiosyncrasies of application domains [231]. In the context of the Ubiloop project[6], for example, the development of a mobile app for reporting incidents in cities required to screen guidelines sources for ubiquitous applications, Web technology and guidelines concerning the properties usability, accessibility and security. The combination of different guidelines sources encompassed 177 guidelines that have to be cleaned for removing duplications, guidelines that are not relevant to the project and for solving conflicting guidelines (ex. security guidelines recommending validation steps that contradicts with usability guidelines that recommend minimal actions).

Conflicting guidelines have been previously reported in the literature [124] [236]. Nonetheless, the inner problems related to the occurrence of potentially conflicting guidelines have been poorly documented so far. The resolution of conflicts is a daunting and demanding task that often requires taking into account the trade-offs associated with alternative design choices. Therefore, whenever a good solution for solving conflicts between guidelines is found, it is worth of the efforts for recording and documenting it. Nonetheless, a method and the corresponding tool support are required to support the description of conflicting guidelines. Most of currently existing tools for working with guidelines can handle diverse guidelines sources but they are not able to exhibit conflicts.

In a previous work [146] we have proposed a systematic approach for dealing with trade-offs and design choices associated to guidelines. There, we have provided taxonomy for describing the problems raised when interleaving two guidelines from different sources. Such taxonomy include scenarios for describing guidelines that can be considered equivalent, guidelines that address similar issues but one is more general than another, guidelines that contradict each other either in terms of goals and or recommendations, one guideline that supersedes or that should be used as replacement for another one, unique guidelines that appears once. Our approach [146] is grounded on a rational design approach which forces the justification of potentially conflicts guidelines that might be found overtime when combining guideline sources. The operationalization of the approach is supported by the guideline management tool called Open-HEREDEUX[7] which also acts as a knowledge-base for multiple guidelines sources.

[6] Available at: http://www.irit.fr/recherches/ICS-site/project/ubiloop

[7] Available at: www.grihotools.udl.cat/openheredeux/

3.3 Users and Web applications

It seems obvious that we should design interactive system for users, and the best way for knowing the users is to talk to them. However, one of the trickiest problem for knowing Web users is that they can be virtually anywhere in the planet, which makes difficult to keep in touch with them to seize their needs through direct observation. For that many remote evaluation methods have been developed [252]. However, the observation of remote users is tricky and might pose several challenges for the protection of users' identity (i.e. privacy issues) and accuracy of data collected. In France, the CNIL (the National Committee of Informatics and Freedom; www.cnil.fr) which is in charge of overseeing the protection of citizens' rights concerning the use of their personal information supports recommendations to prevent websites operating in France to keep records of personal data without the explicit authorization of users. In European levels, the EU requires that notice and consent be given for a wide range of data collection, including the use of most cookies by websites (exemptions are granted for some uses of session cookies and other short-term cookies [68]. Even if the protection of personal information is not regarded as equally important world-wide, privacy is an increase concern in the Web domain.

It is interesting to notice that the Web offers a large exposure of contents and services (over 1 billion of Web sites since September 2014[8]) which compete for users' attention. When a Web site finds its public, there is a change that it becomes successful, which can be verified in a first moment by an increasing number of visitors. New users often came with new expectations and needs, so that to keep a significant *revisitation* rate it is important to personalize and/or adapt the user interface [39][155]. An interesting case study about the evolution of users' needs can be illustrated by Facebook which dramatically changed along the years to accommodate the user interface to a large variety of audiences[9]. Multiple audiences is not a privilege of large Web sites and even a simple e-commerce Web site for toys might boast to have kids, parents, grand-parents, friends and teachers among its typical users [78]. From this simple example is also possible to understand how users evolve over time (ex. evolving from kids to parent) thus making the process of identification of users' needs even more dynamic.

The evolution of users can also be observed in terms of what they are able to do with Web applications [61]. In the early days of the Web, users could only browse through contents provided by Web sites. Later, users could actively contribute with content by using tools (e.g. CMS, wikis) embedded into these sites. In this we should notice that the evolution of evolution Web technology also affect users' behavior. Currently, users are interested in adapting contents according to their preferences [39]. Such interests and needs for adaptation of contents can across the boundaries of single Web applications. Users might navigate from one application to another one unpredictably, and some tasks performed in the Web are accomplished by using several applications. This current use of the Web creates new challenges regarding the adaptability on Web applications in order to integrate them in different ways [73]. Collaboration between users using Web application also become common place and this have an huge impact on how users can organize their work [75]. In more recent years, some authors [4][62] in the Web engineering community explore the concept of end-user programming over the Web by empowering users with little program skills with tools allowing them to create their own scripts to adapt the contents of the Web sites accordingly to their need. Whilst these are still early attempts for supporting end-user programming over the Web, these tools are promising and should be regarded in a longer run for understanding users' behavior whilst interacting with Web applications.

3.4 Research agenda

Despite of decades of empirical studies and a vast literature about users and interactive systems, we still have much to learn about users are affected by the use of system and how we develop computing systems that better fulfil users' needs. However, such knowledge about users is difficult to grasp at once, either because the target user population is non-homogeneous, users evolve overtime (ex. increasing his experience with technology in the domain, having news experiences with the product, aging, etc.), or because the context of use imposes constraints on user behavior which require fine analysis (ex. using map on a mobile phone while driving requires more attention that when using sited in back side of a bus).

In order to understand the users and their needs we need appropriate method for describing such knowledge. As far as the engineering of interactive systems is a concern two basic aspects should be addressed: improve the expressiveness of user models and demonstrate how the knowledge obtained from user's models can be correlated to artifacts used to build interactive systems.

In our research we have addressed side-effect problems that might affect the development of interactive systems (such as understanding of the needs of all participants involved in the development process, expressiveness power of tools and notations and in particular task models for better representing the knowledge users). The results

[8] Source: http://www.internetlivestats.com/total-number-of-websites/ (September 2014)

[9] Full illustration of the Facebook case is available at: http://www.memoclic.com/galeries/30-evolution-facebook

obtained so far are promising and motivate the research in the same direction. Beyond that, we also have identified other two important lacunas related to the management of the knowledge about the users that should be addressed in future to help to engineering interactive system:

- Charting the evolution of users (in terms of requirements, behaviors...) and their needs. There is strong evidence that users and their needs for using the system might evolve overtime. However, so far it is very difficult to reason about user evolution due the lack of methods and tools providing the appropriate level of abstraction for such analysis such information about users. Tracing the evolution of users' needs might help designers and developers to better understand the impact of design choices and assess the potential of user's adaptability to the user interface after using the interactive system for a while.
- Embedding user models as part of the specification of interactive systems. Currently, user models are not a mandatory part of the software and there are often absent in the specification of interactive systems. As a consequence, when the software is delivered there is no means to trace the design choices made by the developers to the needs identified in a particular user model. If user models are systematically embedded as part of the system specification it would be possible to compare the premises/rational behind user models and the actual use of the system made by real users. The consequences of this kind of research would be two-folds: on one hand help to improve the representation through user models that are able to capture the characteristics of real users; on the other hand, it might help the development team improve their hypothesis involving the target users for the system they are developing.

These aspects are going to be addressed in the PhD thesis of Thaíse Costa who I co-supervise at the *Universidade Federal da Paraíba*, João Pessoa, Brazil. That PhD thesis if focused on an e-learning platform that is aimed at follow the evolution of users' skills along the time. On one hand we are investigating users' models to describe the skills that should be developed by the user during e-learning activities. On the other had we expect to use that model to monitor actual progress of users, and then understand how design alternatives might help to improve (or not) the development of users' skills.

To investigate models which are able to explain mechanisms used by users to adapt (or not) their work practices after the introduction of a new interactive systems. Rouse (1981) argues that "*humans, if given the choice, would prefer to act as context-specific pattern recognizers rather than attempting to calculate or optimize*". We suggest that long run studies based on user observation and log file analysis would be useful for understanding users' deviation on prescribed tasks with systems. Such kind of research is of particular interests for the engineering of interactive systems as it can help to better plan tasks provided by interactive system, prevent unacceptable users' deviation by adding barrier at the interaction and prevent the occurrence of human errors.

4 Models for engineering interactive systems

After the software crisis in the 60's and since the emergence of Software Engineering as a discipline in Computer Science, models and model-based approaches are regarded as a suitable solution for overcoming diverse problems raised by the increasing complexity of computers and computing systems. Despite the fact that models can be effectively used to describe computing systems in general, the focus of this chapter is to identify models that are able to describe interactive systems. For that we have to highlight the idiosyncrasies of interactive systems which require models to describe the inner behavior of the system but also the representation of its users, the context of use and how users interact with the system. As we shall see, models are omnipresent in all chapters of this work because they contribute in many ways for the engineering of interactive systems. It is the intention of this chapter to provide an overview the models that we consider as important for the engineering of interactive systems and how we can improve the description of inner characteristics of interactive systems by extending existing models and by creating new ones. Models describing the development process are covered by section 2.2. Hereafter we describe the role of models for coping with the idiosyncrasies of interactive systems in diverse application domains and in particular for dealing with the development of Web applications. The last part of this chapter we pin point out our contributions in terms of models for engineering interactive systems and in the specificities of models for describing Web applications. Finally, we propose a research agenda in the field.

4.1 Overview of models for engineering interactive systems

Across the years the term *model* has been overused in the Computer Science literature to refer to *conceptual models*, *notations* and *diagrams*. These multiples uses of the term *models* is an abuse of language to address interwoven aspects, for that a clarification seems needed here. A *conceptual model*, also called meta-model, defines the concepts that describe a particular view of the real world. In order to be properly communicated, *conceptual models* require a representation which is often provided by a language featuring a graphical notation containing the rules for building *diagrams* that are actual representations of the real word problems in a particular application domain. Models are quite often associated to computing tools (such as editors, simulators, code generators, etc.) that allow their operationalization. By the appropriate tools support, *notation* and *diagrams* become the basic building blocks for supporting model-based approaches. It is interesting to notice that it is possible to identify models (at least a conceptual model) behind any computing tool. The process that leads from the interpretation of the real world towards the implementation of interactive is not always straightforward but we can identify some phases that should be accomplished along the way as depicted in Figure 11. These phases are indicated by dotted lines in Figure 11 and are meant to set a permeable transition between modeling aspects.

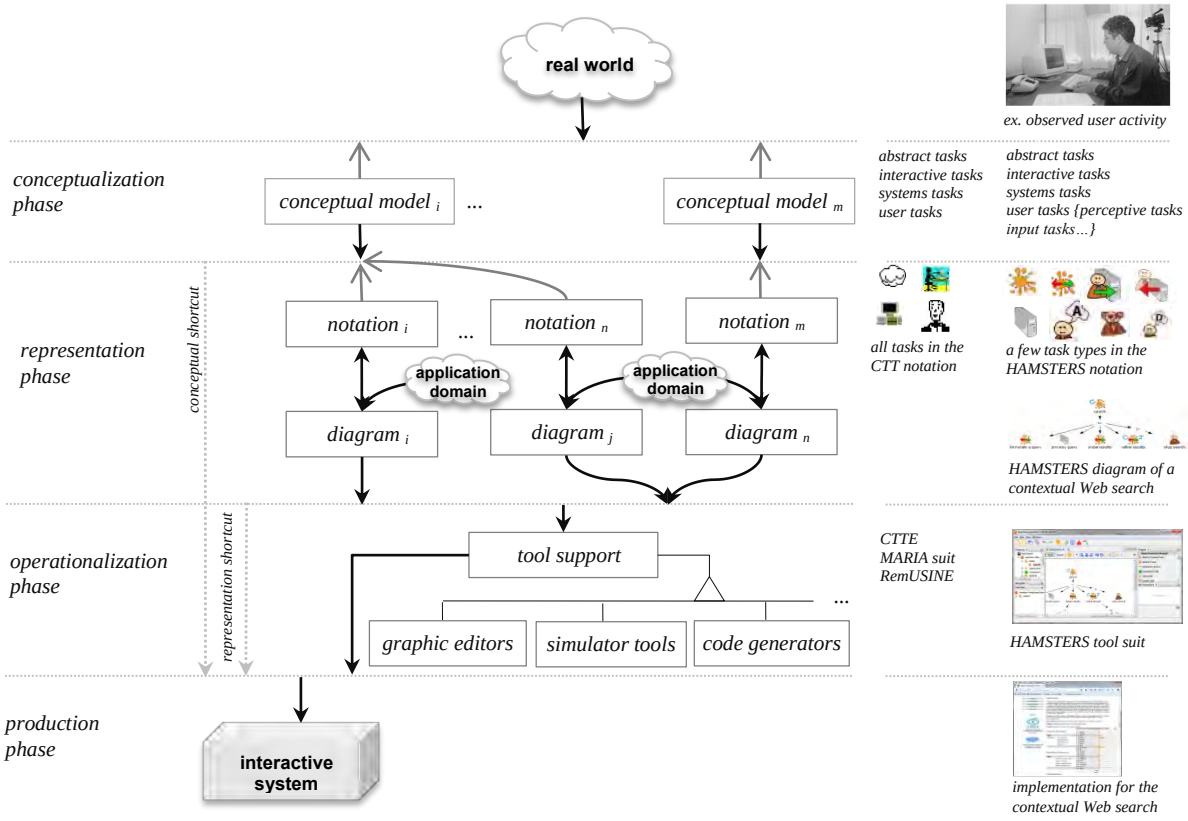


Figure 11. Overview of a models chain for engineering interactive systems.

The *conceptual phase* set the boundaries between something (object, characteristics, features, aspects…) observed in the real world and the concepts created to define it. This is illustrated in Figure 11 by a conceptual task model used for describing user activities with an interactive system that encompasses concepts such as *abstract tasks* (i.e. tasks that can be decomposed), *interactive tasks* (i.e. tasks that require both user and system action), *user tasks* (i.e. task perform by the user) *system tasks* (i.e. tasks performed by the system)... The consequence of the conceptualization process is the creation of a specialized (and yet partial) view of the user activity for that a conceptual *task model* put aside other observable aspects of the real world such as the user's working environment, user's profile, user's emotional state, etc. Therefore, many conceptual models can be created to describe different aspects of observable world. Moreover, as we shall see in Figure 11, the underlying concepts of user tasks can be extended/refined to include additional details; for example a *user task* can be refined in *perceptive tasks, input task*… Indeed, a huge number of models have been proposed in the literature to cope with different types of objects and problems related to the description of tasks [60][131]. Each specialization is a fragmented view of the real world which raises many philosophical questions related to the scope of models such as: How to identify concepts that are not overvalued/undervalued by biased observation of the reality? How to make sure that a model contains all and only the relevant concepts required for solving a problem? Which concepts are relevant for the engineering of interactive systems? How detailed the description of concepts should be? In which extension a conceptual model can be extended to provide a more detailed view of reality? How to prevent that conceptual models become more complex than the necessary for understanding a problem? ...

Once concepts and definitions have been settled, the next step is to provide a representation for them. At the *representation phase,* the formalization of *notations* and the construction of *diagrams* become the keys elements. A *notation* defines the set of graphical elements and rules for combining them, thus featuring a *modeling language*. A *diagram* is a graphical representation created upon rules imposed by a notation and used to describe a particular problem. It is worthy on notice that additional information must be obtained from the application domain to build meaningful diagrams. In Figure 11, CTT [197] and HAMSTERS [138] are used to illustrate notations used for representing concepts in a task model. A diagram featuring the task *conceptual Web search* illustrates the use of some elements provided by the HAMSTERS notation. The creation of diagrams might highlight features that have not been observed and/or represented before, thus defying the representational support provided by notations and requiring a revision of the conceptual model. Idiosyncrasies of application domains might also lead to the development of *Domain Specific Languages* (*DSL).* Indeed, the definition of a diagrams, notation and concepts is strongly bounded, for that Figure 11 features an arrow going upwards from the representation phase back to the conceptual phase. Research questions of interest are: How to interpret graphical representation of models? How reduce/remove ambiguity in diagrams? How to deal with large diagrams? How to assess the expressiveness of notations? When do/don't introduce syntactic sugar in notations? …

The *operationalization phase* is concerned by the development of dedicated tool support for handling diagrams and notations. It distinguishes models that are built by hand from those for which a dedicated tool support exists. The features provided by tools determine the level of operationalization of models including representational purposes only (ex. graphical editors), support to the analysis (ex. model simulators, model checkers, etc.), construction of interactive systems (ex. code generators), etc. Figure 11 illustrates the tool HAMSTERS which supports the eponym notation and it pin points tools such as CTTE, MARIA suit and RemUSINE that exploit CTT models. It is worthy of notice at that some tools can handle more than one type of model/diagram, which is often the case of tools that provide multiple views on the system and/or can support the transformation of a model into another. Whilst the existence of the tool support is not a precondition for using model during the development process of interactive systems, tool support is often seen as a plus for the diffusion of models. Moreover, the availability of tools allows the investigation of research questions such as: How to combine specialized views provided by different models to have a more complete picture of the interactive system? How to automatically inspect models? How to automatically translate a model into another and then determine in which extension they contain similar concepts? How to handle large diagrams for which all representational solutions don't provide an effective solution? How to optimize and tune models to improve performance? …

Last, but not least, the *production phase* implies that a particular model has been used to build an interactive system. An illustration is given for the *implementation of a context Web search systems* which has been previously conceptualized and represented as HAMSTERS task model. The obvious straightforward passage through the *production phase* is by using tools. However, even model-based representation built by hand can be useful during the development process of interactive systems, for example as a simple communication support among the development team; for that reason, a *representation shortcut* has been added to the diagram at Figure 11. Similarly, a *conceptual shortcut* is used to represent the integration of elements of a conceptual model during the development process of an interactive system that have never been properly represented and/or edited by tools. At this point, we can investigate some additional research questions related to the effectiveness of models for supporting the development process of interactive systems; the tradeoff of model based approaches versus had hoc development process, accuracy of models representation in the description of actual system behavior, use of models at runtime to control the execution of the system…

In the present work we assume that *a model is a language representing of some aspects of the real world that is used for solving problems with interactive systems*. In that definition we will find all the elements around the models including a *notation* used to represent a real life problem featuring a *diagram* that describes the inner *concepts.* Such as a definition also encompass the concept of language for describing the user interface, or a *User Interface Description Language (UIDL),* that must be understood as a mean to talk about all the modeling aspects. For the sake of simplicity, we will use the term model in a very broad meaning covering conceptual models, notations and/or diagrams. In case of possible ambiguity, we will use the terms conceptual models, notations, diagrams and modelling tools whereas is needed.

The contributions of models to the engineering of interactive systems are many and they might include the following: the development and/or extension of *conceptual models* and *notations* aimed at describing the system and the user interaction*;* the use of notations to build *diagrams* for representing actual problems in a particular interaction technique; the development of *tool support* for automating and/or better supporting the use of model; the study of strategies employing *model-based approach for building and assessing interactive systems* and the *transformations required* in the model chain to generate user interface from abstract descriptions. The model chain presented at Figure 11 is aimed at presenting dependencies between conceptual models, notations and diagrams and support the discussion around general problems related to the scope and the expressive power of models. By no means have we aimed at describing a canonical process for producing conceptual models, notations and/or diagrams. Indeed, we assume that the development of model is not a straightforward process and quite often the before mentioned aspects are interwoven and developed in parallel.

4.2 Software Engineering models for engineering interactive systems

The research in the field of Software Engineering is rich of examples of models for developing computing systems [4][156]. Hereafter we discuss the limits of existing models, the need for developing and extending new models for engineering interactive systems and degree of formalization require making models fully operational.

Entity-Relationship Modeling (ER models) [50] and Unified Modeling Language (UML) [166] are among the most widely used notations for describing conceptual models (respectively data models and software design) and extensively employed in a variety of model-based approaches for software development. Given the fact that user interfaces represent an essential part of the code of computing systems, it is somewhat natural to expect that models developed in the Software Engineering field would provide a fair support for modelling interactive systems as well. However, not less can be true when the aspects of the system when we want to represent are the composition of user interface elements and how user interact with the system. Other significant drawbacks of UML models include the lack of notations able to describe the user interface in several levels of abstraction and the occurrence of fine-grained system's behaviors such as fusion/fission of events. UML seems to have been developed with little specific attention given to user interface issues [185]. Indeed, a limited number of UML models are able to addresses a few concepts that are relevant for the specification of interactive systems.

A good example of such as limitations can be illustrated by use cases diagrams that are proposed as part of the tool suit of UML models for capturing the intention of users might have towards the system [113]. Use cases are useful because they help to identify user goals and systems functionalities. However, the representation of such models is limited to a simple action providing very little details about the activity performed and actors involved (user and or system). This over simplification of use cases might lead to multiple interpretations of actions from different designers [52]. Interesting enough, use cases share some key concepts with tasks models which emerged from the HCI research in the 80's. Some authors [219] have suggested that use cases and tasks models might share a formal semantics for the key concepts. Although task models provide a more detailed and less ambiguous description of users' goals than use case, task models have never been integrated into the set of UML diagrams [60]. Many authors [215][217] have proposes methods for better integrating HCI and Software Engineering methods and thus harmonizing the development practices of interactive systems. For example, there were many attempts [230] to show how translate task models into UML state machines aimed at representing the dialog of interactive systems. Other authors propose to use tasks models to complete and extend the description of UML use cases [178].

It is still debatable in which extension UML models can be adapted and extended to copy with the idiosyncrasies of interactive systems [185]. The SysML [225] which can be seen as an extended version of UML 2.0, suffers from the same deficiencies with respect to interactive systems. In more recent years (March 2013), the Object Group Management Inc. adopted the Interaction Flow Modelling Language (IFML) [167] as a standard for describing the user interfaces. IFML diagrams are meant to be built on the top of system models describing the data and software infrastructure. Despite the fact that IFML explicitly recognizes that the user interface should be modelled as part of the whole system, it still presents the user interface of a feature supported by the system in development which does not encourage a user-centered design. Nonetheless we can conclude that traditional software models are helpful in many aspects that concerning the development of computing systems (in particular data and in software architecture) but additional views should be added for supporting the development of interactive systems. The need for multiple views of the user interface is better described in the section that follows.

4.3 Important models for engineering interactive systems

By definition an interactive system encompasses a dialog between users and the computer which, according to Green [82], is better modelled as the triplet: (*user*)↔(*user interface*)↔(*system*). This triplet model, which durably influenced the development of user interface management systems (UIMS), was aimed at providing a model-based view of the development of interactive systems. Figure 12 presents an extension of the Green's model by including further views about interactive systems. As we shall see at Figure 12, the view about users distinguishes the representation of users as individuals in a population (the user *profile*) and the user as a human being (the user's *cognition*). For the user interface, models might cover the *presentation* of graphic elements, the *interaction* and the *dialog*. Whilst *presentation, interaction* and *dialog* are interwoven in the final user interface, it is possible to treat those aspects of the user interface separately for which dedicated notations exist. For the system we recognize that models describing the *platform,* the *system architecture* and the *information space* might also influence the design. Figure 12 also includes two orthogonal views of models describing the *application domain* and the *context* which also might have an effect on the design and development of interactive systems. The representation of *tasks* is a particular case between what we represent of the user (in term of generic activities) and the user interface (when tasks are performed with an interactive system).

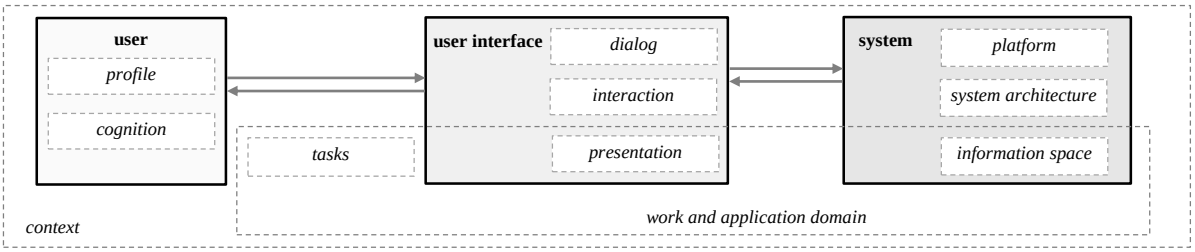


Figure 12. A complimentary view on Green's model for describing interactive systems [82].

As we shall see, the complexity of interactive systems implies a large set of concepts ranging from user-oriented to systems-oriented concepts that can hardly be accommodated in a single model without the risk of increasing complexity and loosing focus on the aspects necessary to solve individual problems. For that many specialized models have been proposed in the literature to cover the different views proposed at Figure 12.

An overview of orientation of models and their underlying concepts is presented by Figure 13. It is worthy of notice that *contextual models* and models describing *work and application domain* are not included in this representation because is not possible to described them as having a clear orientation towards users and/or to system. As for *contextual models* they are often a combination of diverse dynamic features describing the environment where evolve the users and where the user interface and the system is deployed. Models describing *work and application domain* might be useful for many types of applications, even those that do not consider interactive system. However, *task models* are a remarkable exception in this matter because they are helpful for describing what people do (or are supposed to do), which is useful information for characterizing application domains. Interesting enough, when applied to interactive systems, *task models* can act as a bridge between *user models* (by describing users' goals and skills required to complete their activities) and *user interface models* (by indicating the resources used and how tasks can be accomplished with the help of the system). Thus the order of execution of tasks is (or at least should be) reflected by the internal behavior of the interactive systems which are the subject of *dialog models.*

User-oriented ↑	***Scope and concepts of models***
Cognitive models	user cognition, user memory, user behavior, emotions..
User models	user profile based individual characteristics, demographics, personality traits, users' goals,…
Task models	goals, dependency between tasks, types of tasks, pre-/post-conditions, order of transition between tasks…
Dialog models	actions, events, conditions, parameters, states and transitions…
Interaction models	actions, events, resulting rendering on the user interface…
Presentation models	user interface composition, set of widgets, metaphors, icons, color coding, …
Information space	inner organization of data characterizing an information space employed by users to complete their tasks…
Architectural models	data processing and overall software architecture including behavior of the functional system core…
Platform models	characteristics of the devices and the platform where the system is running…
↓ *System -oriented*	

Figure 13. Scope of models for engineering interactive systems.

In the sequence, we provide examples of conceptual models, notations and diagrams that illustrate the classification of models that you consider important to the engineering of interactive systems. Some of the examples are simply conceptual models that don't have a graphical representation form that could be used to build diagrams describing actual interactive systems. The main utility of such conceptual models for the engineering of interactive systems relies if the formalization of concepts that are embedded into particular applications. Rather than to provide a comprehensive review of conceptual models and notations for engineering interactive systems, we propose a classification hereafter of models that might help the reader to understand the articulation between different types of models. We must say that despite the fact that the representation of all views presented at Figure 12 provides a more complete picture of interactive systems; models can be used alone or in combination according to the problems at concern.

Cognitive models are meant to describe concepts around users' cognition and behavior. Quite often these are conceptual models that can be used to understand how users might think and behave when facing interactive systems. Models in this category include the Model of Human Processor (MHP) [43], Norman's Action Theory [156], Interacting Cognitive Subsystems (ICS) [11] and Rasmussen's SRK model [60]. These models correspond to the definition of *analytical user models* described in section 3.1.1. A few concepts of *cognitive models* can be found in other modelling approach, for example the concepts of *goals* and *operators* described by GOMS [115] can be found in tasks model notations such as CMGOMS. GOMS is often referred as a modelling method that can be operationalized by Keystroke Modelling Method (KLM) and supported by tools such as CogTool [226]. Another example is SRK model [60] whose concepts of procedural, situational, or strategic knowledge required by users to accomplish certain tasks have been embedded into the task model HAMSTERS [143]. However, even without a corresponding notation for building diagrams of actual interactive systems, cognitive models in their conceptual models might be helpful for understand the drawbacks in the user interaction with the systems and as such they might play a role in design and evaluation phases of the development process.

User profiles provide a representation of actual users or a target population of users. Models in this category include as *personas*, *scenarios* and *roles/job profiles* which are duly described in section 3.1.2. We should mention here that the representation provide by these models is textual and somewhat informal so that they are prone to subjective interpretation. Thus, whilst it might be hard to provide an accurate mapping between contents of user models and the interactive systems, user models might help the development team to keep in mind that their design has a purpose and it should ultimately fit to the needs of users in a recognizable target population. Conversely, we think it makes senses that the description of target users (at least in terms of required skills to use the systems) should be delivered as part of interactive system.

Task models play a major role the engineering of interactive systems. The investigation of tasks models started in the 80's as a mean for describing and analyzing user activities. Task models have a long tradition in the HCI research [60] culminating with the creation of series of conferences focused on *TAsk MOdels and DIAgrams* (TAMODIA[10]). Some task models such as GOMS [63] is directly derived from the Human Model Processor (Card, Moran & Newell, 1983) [25]. Along the years, many researchers have investigated the underlying concepts around task models and they have proposed conceptual models which are too often accompanied by new notations. Task models such as TKS [116], CTT [197], GOMS [63] and HAMSTERS [138] are among commonly used notations for representing what users can accomplish with interactive systems. However, as wisely suggested by Limbourg, Pribeanu and Vanderdonckt [131], most of key concepts of those tasks model are cross-compatible and could be translated to different notations using a common meta-model. Beyond the importance of tasks models for analyzing users' activities, the very existence task-based notations, and in particular those based on formal methods, must be considered a great step forward to the development of interactive systems because they allows the creation of a bridge between user's goals (expressed in terms of tasks that must be performed) and interactive systems (as users tasks can be associated to components of the user interface that represent resources used to complete a task; moreover, the dialog of usable interactive systems is expected to follow the natural users' logic of execution which is typically described as part of task models) [158].

Dialog models play a major role on the design of interactive systems by capturing the dynamic aspects of the user interaction with the system. It is easy to think *dialog models* in terms of low-level interaction as a mean to describe how users can send a message (the input) to the system (by triggering low-level events such as mouse click) and how the systems replies to the users by changing the rendering of the user interface (the output). However, these elementary concepts of the dialog can be applied at different levels of abstraction of a user interface [261]. For example, it is possible to represent a dialog model featuring transitions between abstract presentation units (typically grouping contents that should be presented together) far before to decide the modality (graphic, textual,

[10] TAMODIA was a series of specialized international conferences organized around the themes of tasks models and diagram. Annual conferences were run from 2002 to 2009 after which, TAMODIA was integrated as part of the *ACM SIGCHI Symposium on Engineering Interactive Computing Systems* which also encompasses the former conferences HCI (*Engineering Human Computer Interaction, sponsored by IFIP 2.7/13.4*), DSV-IS (*International Workshop on the Design, Specification and Verification of Interactive Systems*), and CADUI (*International Conference on Computer-Aided Design of User Interfaces*).

audio, etc.) of contents delivered to the user. *Wireframes* are a good illustration of simplified *dialog models* that can be used to describe some transitions between presentation units in low-fidelity prototypes. Without wireframes, it would be difficult to extract from the designers' mind the sequence of presentation of screen in a prototype and it would also be impossible to build prototype builders for mimicking the systems responses to users interacting with low-fidelity prototypes. Using *dialog models* based on formal method, it is possible provide accurate fine-grained description of any aspects of the system behavior which is not only useful for design the system but allows *dialog models* to be used for controlling at run time the execution of the fully implemented interactive systems [14]. In its full extents, a *dialog model* might include the specification of: relationship between presentation units (e.g. transitions between windows) as well as between user interface elements (e.g. activate/deactivate buttons), events chain (i.e. including fusion/fission of events when multimodal interaction is involved) and integration with the functional core which requires mapping of events to actions according to predefined constraints enabling/disabling actions at runtime [261]. Notice that a *dialog model* does not describe the rendering of the user interface, which is the main scope of *presentation models*, and its simplest form, can be represented as kind of node-edge diagram. Parnas (1969) [176] provided one of the first (if not the very first) example of dialog modelling which typically address user interface issues. Since then, a large number of notations and modelling techniques have been proposed for describing the dialog aspect of the user interface. Some notations are devoted to the dialog aspect of the user interface (example ICO [18] , SCXML [220] and SWC [255]), while other models might also cover the structure and the presentation aspects. Is some cases the description of the dialog is supported by an external language (ex. XUL), however, quite often, the dialog is embedded into the UIDL, such as is the case of UsiXML, XUL and UIML. A full account of dialog modelling notations is beyond the purposes of this work but for a comprehensive review we invite the interested reader to take a look at the work of Navarre et al. [158].

Interaction models are can be said as a particular type of *dialog model* as they also concerned by description of the behavior of the interactive systems. However, contrary to *dialog models, interaction modelling methods* such as the Interaction Object Graphs (IOGs) [45] should feature a description of the system rendering and the low-level systems events. As such, *interaction models* do not support high levels of abstraction about the system behavior. Nonetheless, *interaction models* have been demonstrated helpful for representing fine-grained behavior at the execution time. A good example is ICON [67] that, by the means of tool support, provides a graphical representation of events and devices connected to an application; using the toolkit ICON, users can configure the events (or events combination) that will be used to activate a particular rendering in the application at run time. Another interesting type of *interaction model* can be found in [91] where an extension of the ICO notation has been proposed to support the description of multi-touch interaction techniques. Hamon et al. [91] demonstrate that, using the extended ICO notation and a layered architecture, it is possible to represent the dynamic instantiation of input devices (i.e. finger) which can then be exploited dynamically to offer a multiplicity of interaction techniques that are also dynamically instantiated. *Interaction models* are often complex so that their use is often limited to the description of advanced interaction techniques and are not suitable for describing the complete behavior of the interactive systems.

Presentation models are aimed at describing the rendering (output) of user interfaces. Most of the techniques for prototyping interactive systems employ presentation models that are combined with dialog models to provide a more complete description of the user interface, thus including not only the presentation but also the behavior [21]. The degree of fidelity of presentation might dramatically vary along the development process. In early phases, paper-based mockups can be considered informal, inexpensive and yet suitable presentation models that can be used to communicate basic ideas about the user interface design. However, drawings on paper are informal descriptions that can be subject to interpretations (i.e. ambiguity in the recognition of the graphical elements) and insufficient to describe some design constraints (ex. precise size and position of objects). We consider that the semantic of graphical elements aimed at representing the user interface and the degree of fidelity (or realism) of such representation are two important dimensions for comparing presentation models [119]. The formalization of presentation models is nowadays supported by a large number of tools, such as Balsamiq[11] and Axure[12], which replace the ambiguity of hand-made sketch by the semantically meaningful widgets that can be selected from customized palettes containing the widgets that can be used with a particular platform and/or device. Some specialized tools such as GAMBIT [207] include gesture recognition for automatically transforming hand-made sketches into presentation models. Whilst it is be possible to envisage rendering as having visual, audio, and/or any kind of other sensorial output to the users, it is interesting to notice that most of the existing presentation models focus on graphical features including the description of widgets used in the composition of the user interface, metaphors, icons, color coding, etc. Giving the still growing importance of multimedia and multimodal interaction techniques, we suggest that presentation models must evolve (or be completed by other kind of models) to cover other forms of rendering.

[11] Available at: https://balsamiq.com/

[12] Available at: http://www.axure.com/

By *information space* we address a large set of models aimed at describing the inner organization of information required by the users to complete their tasks with the system. Traditionally, information might be available in the form of *structured data* (i.e. entities in the same group have the same descriptions or attributes and that can be easily represented by a relational-like database schema), *unstructured data* (i.e. data in free form that does not follow any particular format or sequence such as media streaming and free text) and/or *semi-structured data* (i.e. data that can be partially structured by using marks or tags, which is typical of text formats and Web documents). Entity-Relationship (ER) [50] and UML class diagrams [166] are widely known examples of modelling notation for representing structured data. Generally, search engines are used for retrieval of unstructured data via querying on keywords or tokens that are indexed at time of the data ingest; in that case mathematical models can be used to describe the structure of indexes used by algorithms embedded into search engines [31]. XML [238] and HTML [239] are contemporaneous languages that borrow from the Dexter Reference Model [91] and the Standard Generalized Markup Language (SGML) [110] the basic syntax elements allowing the description of semi-structured data [1]. Currently, the term NoSQL [94] is frequently employed to refer modelling techniques for processing unstructured data and semi-structured data. Nonetheless, the main focus of NoQSL modelling techniques is to represent progress data management that meets the needs of modern business applications, such as scaling big data. In this scenario, models for representing metadata and semantic data should also be considered part of the *information space* [151].

Architectural models are useful models for representing the various architectural components of an interactive application and the relationships between them. Architectural models are aimed to describe data processing and overall software architecture including behavior of the functional system core. As far interactive systems are a concern, architectural models of interest should include the description of the modalities of user's input/output [218]. Various architectures for interactive systems have been proposed trying to address specific problems raised by a specific kind of interaction technique in an interactive application. For instance [105] targets at multimodal interactions while [93][160] address specific problems related to touch input. Most of these architectures refine and extend the Arch model [17] proposing additional layers supporting the transformation of low level user events into higher level events ultimately transformed in interactions techniques. What is important is that these architectures feature various properties which are most of time not explicitly mentioned. That diversity of architectures comes precisely from the fact that each architecture takes a specific point of view and favor some properties (such as rapidity of information processing, handling efficiently concurrent input …) while giving up with other ones. The classification proposed in [18] emphasizes the identification of properties of frameworks and architectures for interactive systems. It is interesting to notice that *architectural models* are quite often dependent on the hardware/software integration for that they are associate to specific platforms.

Platform models are aimed at describing the characteristics of the devices and the platform where the system is running, which ultimately might include physical attributes (ex. screen size and resolution, input/output devices available, number of processors, memory an memory capacity, etc.), software platform (ex. operational systems, software architecture issues such as the need of a virtual machine/browser/library for deploying the system, system configuration for diverse aspects including security, etc.) and system performance in particular configuration and/or settings (ex. performance according to the number of connected users, network bandwidth, etc.). Platform models become particularly relevant in the context of multi-target interactive systems, when the user interfaces should be adapted to device's constraints such as screen resolution and input/output devices available [241] (ex. mobile phones, tabletops, etc.), which requires the inclusion of the notion of plasticity in the development process [42]. In this context, platform models are aimed to help with the development of multiples versions of the same applications running in multiple platforms. Whilst the concept of platform might have a broad meaning, the term *platform models* are rooted with the Model Driven Architecture (MDA) approach proposed by the OMG [119]. MDA promote a radical shift in the way to build application by moving from object composition to model transformation [32] where a model-driven engineering framework exploiting a meta-model of the application should be able to manipulate both PIMs (Platform Independent Models) and PSMs (Platform Specific Models) to generate applications according to the target platform [154]. In MDA, platform-independent models (PIMs) are initially expressed in a platform-independent modeling language, such as UML, then it is subsequently translated to a platform-specific model (PSM) by mapping the PIM to some implementation language or platform (e.g., Java) using formal rules such as Object Constraint Language (OCL) [119]. Whilst the description of the platform is an important aspect of the description of the interactive systems, *platform models* are not frequently available outside of a MDA approach. The contributions and limitations of MDA approach for the development of interactive systems is better discussed in section 2.2.

We assume that Figure 13 illustrates the main categories of models for describing most types of interactive systems. Nonetheless, this section would not be complete without a few words about models for describing *contextual models, work and application domains*. We consider these two categories of models as orthogonal to for the development of interactive systems because they assemble information that might be already available in other models.

Contextual models are aimed at providing a description of a situation and/or an entity (i.e. person, place, object...) that is considered relevant to understand the interaction between a user and an application. The meaning of what should be considered *context* and which information should be used to describe context might be depended on the application domain, for that this question is debatable in the community and no agreement has been found yet [59]. The set of attributes that could be used to define a context can be huge and they might evolve overtime. For that, it seems overkill to systematically employ models to describe the context of use for all types of interactive systems. Nonetheless, we recognize that *contextual models* might be useful for some kind of interactive systems in particular adaptive systems such as recommender systems [182], plastic user interfaces [42][58] and adaptable front-end services [82]. A good example in the matter is given though a space problem proposed by Coutaz and Calvary [55] that is aimed at describing dimensions/criteria that define the context for user interface plasticity. Based on the definitions within that space model those authors [55] propose a set of *contextual models* that were used to drive adaptations of user interface in the CAMELEON framework. Another use for contextual models is given by Calvary et al. [41] who employ a problem space model to assess design alternatives for deploying e-government applications in different platforms.

The term *work and application models* is used here in a very broad meaning to address many types of models that are used to seize the characteristics of an application and/or application domain. For example, *workflow models* are used to describe, plan, control, and manage organization's business processes [87]. Models in this category might include different views about organizations, people, processes, user tasks, business polices, etc. It is interesting to notice that some of the information used to describe *work and application models* is also available in other models and in particular in *task models, presentation* and *information space*. For example, a *workflow* is recursively de-composed into processes which are in turn decomposed into tasks; thus *task models* and *workflow* models are indeed close related but yet not quite the same as workflow include business constraints that are out of the scope of user interaction with the system. The study of models for *work and application domain* is an interesting subject of the research in the field with a great potential for cross-pollination between disciplines. On one hand, the observation of user activities and the fully understanding of users tasks has led to the proposal of methods for optimizing business processes described by workflow models [135]. On the other hand, the study of workflow-based applications allows the identification of design patterns that can be used to improve the coherence of user tasks among applications [88]. These early studies are promising but we assume that much still remain to be done before we can fully understand the contributions to the Engineering of Interactive Systems of *work and application models*. We should notice that not all interactive systems need the description of *work and application domain* to be developed. To make it explicit we can recall that not all interactive systems run under a workflow.

4.4 Diversity of models

A quick look in the literature and one will find a huge number of publications talking about conceptual models, notations and modelling languages for engineering interactive systems. Indeed, the research community has been very prolific in the matter and although it is difficult to identify a single cause for justifying the development of models, notations and/or modeling languages, there are a few obvious reasons that worth to be mentioned: create/update the scope of a conceptual model, develop/improve a graphical/textual representation for concepts, amending models for making them usable and operational via tool support. Currently, many researchers and practitioners treat the diversity of models as a problem that can be solved by selecting the model that better suits the resolution of a problem [38], by trying to make notations cross-compatible [131] and/or by developing tools for transforming a model into another [211]. In this section we propose another perspective for supporting the discussion about the diversity of modes that include the analyses of some processes might affect the development and adoption of models, notations and modeling approaches. In order to support the discussion we will mainly focus task models the rationale behind the illustration can be applied to other category of models.

4.4.1 Underlying process leading to models diversification

In section 4.1 we present a models chain featuring a conceptualization phase, a representation phase and an operationalization phase that respectively leads to the definition of conceptual models, to the definition of notations and modeling languages, and to diagrams representing aspects of interactive systems. However, we shall notice, that model chain allow much permeability between phases; for example, new concepts might be discovered quite late in the process only when trying to create diagrams using a notation for represent problems in a real life application. Thus, it seems that the development of models is not a straightforward process. Nonetheless, currently we know very little about the underlying processes that lead to development of models. We consider that understanding these processes might be useful to understand the phenomena of models diversification. We also suggest that the study of these processes might deepen the understanding of the model and model-based approaches in a large meaning. To illustrate model diversification, Figure 14 shows a timeline of publications concerning task models. It is important to say here that timeline only refers to the first public mention of a particular task model and it does not indicate whether (or not) a model is still in use or how important it is for the community. For example, whilst HTA is mainly cited in scientific article for historical reasons it was in use for several decades (at the in the academia) and it had a strong impact on the research on task analysis and inspiring the development of new models such as GOMS, TKS, etc. [60].

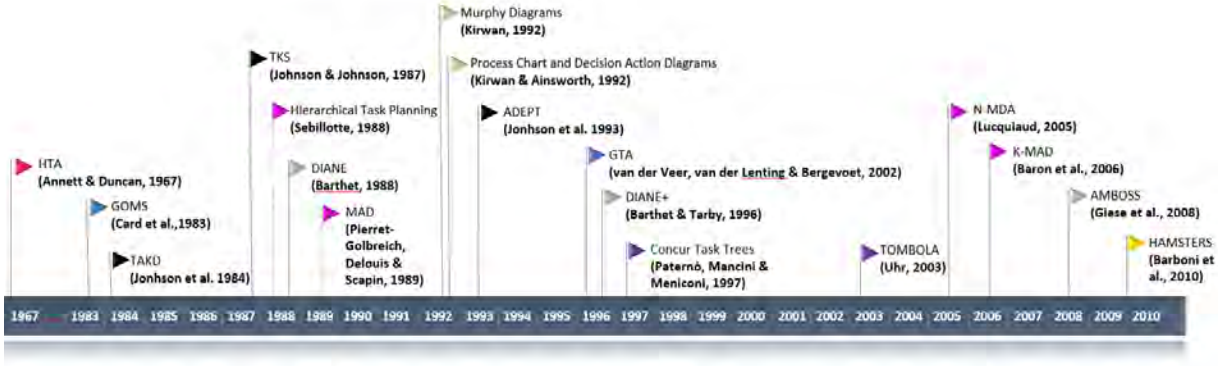


Figure 14. Timeline showing diversity of task model notations as they first appear in the literature.

The series of publications in Figure 14 illustrates a variety of task models. We can identify three main aspects that might be used to justify the diversity of models: the scope of concepts covered by the conceptual model, the many possible representation of concepts and the degree of formalization of concepts. Such as variability might reflect a need for adaptation and/or specialization of models to cover idiosyncrasies of interactive systems as discussed in the introduction. It is worthy of notice that the conceptual model and the definition of the notations might be performed in parallel but they are often published/released at once in the same publication. It is pretty sad that in many cases, publications are delivered without making any explicit mention about the processes that lead to the proposal of extensions and/or completely new models, notations and/or modelling languages.

For an example of processes we had in mind we must cite the discovery and acquisition of concepts. Discovery often occurs when a real life application/case study present elements (never conceptualized before) that should be included in the scope of a model used to describe the problem to be solved. The acquisition of concepts comes to play when concepts can be borrowed from other conceptual models. During the process of acquisition, concepts can be taken as they were previously defined, concepts can be refined (for example to increase the level of detail), or simply discarded if we assume they are not necessary for solving the problem at a concern. It is interesting to notice that discovery is often justified by the authors as a need for specifying problems with case studies they were working on. For example, AMBOSS introduced the concept of risk factor to tasks which has need found necessary for applications in the safety-critical domain. GTA introduced the concept of collaborative tasks when trying to deal with interactive systems. We can also observe a process of acquisition in the repetition of basic concepts such as tasks and tasks decomposition that have been originally set by HTA and systematically reused in other notations; we can purposely wonder which processes lead to a large adoption of the hierarchical organization of tasks in sub-tasks in detriment of other forms of organization of tasks such as cyclic graphs.

As far as task models are at a concern, a wide availability of tool support should be account as an important factor for the adoption of models and notations. For example, whilst the MAD notation has been published as early as 1989 [183] and a tool support (EMAD) existed in 1991, both tool and notation were only available as internal reports at INRIA. Only in 2006 an extension of MAD notation (K-MAD) and the corresponding tool support K-MADe were published in a French-speaking conference [16]. In contrast, availability of the CTTe tool support in 2000 [175] helped a lot for the popularity of the Concur Task Tree (CTT) which originally appeared in 1997.

Another aspect concerns the formalization of conceptual models thought notations and/or modeling languages. Once the scope of concepts for a conceptual model is set, two main questions arise in a tandem: how to represent these concepts? and how much formalization is required? It is interesting to notice that the same conceptual model might benefit of more than one notation and some notations can offer an incomplete coverage of concepts. The formalization of concepts implies a certain degree of freedom for interpreting graphical/textual elements of a notation. Informal models allow much freedom for interpreting diagram; semi-formal models propose a reduce number of possible interpretation and formal models only allows a single way for interpreting diagrams. For examples, it is also interesting to notice the large variability of icons used for representing tasks. Concepts can be taken at many degrees of abstraction according to the artefacts and models used to represent them. With mockups, for example, a task is described as an informal annotation and interpreted as what users can do with the prototype. Task model such as HAMSTERS, allows a semi-formal description of the tasks that include information about the decomposition of the steps required to accomplish a goal and/or the necessary articulation with other tasks that are part of the users' job. We call these task models semi-formal because it still allows some interpretation on how tasks are actually performed with the system; this interpretative problem is solved by formal models. It is interesting to notice that most of modelling language, with a remarkable exception of USIXML [132], have a single level of formalization. Currently we know very little about the process that leads to choosing a particular degree of formalization and articulation with other more/less formalized models occurs. This questions about the degree of formalization of models is of great importance for the engineering of interactive systems for that this discussion is resumed in section 4.4.3.

4.4.2 Dealing with multiple models

As we shall see in Figure 13, the scope of concepts covered by models is diverse and a single model cannot accommodate all views of an interactive system. The processes that involve multiple models represent a particular challenge for the development of interactive systems. A comprehensive description of all is not always necessary to build interactive systems and the models to use depends on the problems to solve, the complexity of the system at a concern, and the people who used the models. For example, for representing e-procurement applications we need concepts related to tasks and workflows, which can be covered by dedicated tasks and workflow models, undifferentiated notations that can accommodate both types of concepts or by weaving models (ex. SWC and CTT) as illustrated by Figure 15. As illustrated in [158] with the implementation of a simple game such as the game of 15, there are many cases when only by combining models it is possible to provide an accurate description of how to successfully develop an interactive system. The question whether to have a single or multiple models to solve a problem is tricky and it is one of the core questions for dealing with the development of interactive systems.

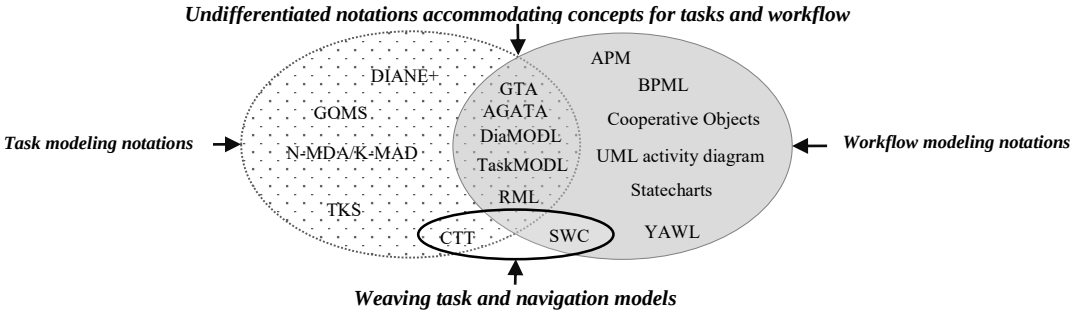


Figure 15. Models covering concepts of e-procurement applications (Pontico, Farenc & Winckler, 2006 [188])

In large software projects often multiple modeling languages are used in order to cover the different domains and views of the application and the language skills of the developers appropriately [68]. The increasing complexity of interactive systems will ultimate require the extension (or development of new) models. The idea of combining models and views has occupied the software engineering community in the last years [119]. Such "multi-modeling" raises many methodological and semantical questions, ranging from semantic consistency of the models written in different notation to the correctness of model transformations between the other notations. Interrelating different modeling notations is a difficult task due to the variety of possible structuring mechanisms and underlying computational paradigm. Moreover, the semantic of models is not something that can be translated in a straightforward process. So far we can identify a few directions for the research in the field:

- Articulating views of multiple modeling languages: The underlying idea is to make the correspondence between models that have concepts in common. Bindings between models are created by developers by making parts (and or concepts) of a model to correspond to another model. Once a correspondence between models is stablished it is possible to follow how changes in parts of one model affect other models attached by a coordinated view. This strategy has been demonstrated the successful exploitation of coordinate views between task models and system models [14][137][158][251]. On one hand models remain independent of the coordinated view and can be used in individual basis whereas only one view of the system is required; on the other hand coordinated views can help to follow the co-evolution of models when adaptations in one model is required to keep the coordination in place with other models. Moreover, coordinated views can ensure a certain level of co-execution of models [13].

- Model transformation: by using appropriate mapping of concepts a model can be translated from a language to another [68]. A model transformation is itself a model and it is described by a specific transformation language. Multi-modeling languages can be reached by successive models transformations combining and/or producing different models. The underlying idea of transforming a model into another is seductive and become a predominant topic of research in Software Engineering in the last years. However, the use of transformation model is not always straightforward for dealing with interactive systems [58]. Many authors have suggested the use of transformation models to deal with the development of user interfaces [87][134][220]. Nonetheless, many models used for engineering interactive systems are informal and/or semi-formal which poses serious problems for the interpretation of concepts and corresponding model transformation which might lead to usability problems that are embedded into the user interfaces generated through model transformations. The trick issues with a model transformation is that, if a user interface should be change for any reason, we should fix the underlying models instead of the user interface as perceived by designers and end-users, which requires a completely different way of reasoning about the development process of user interfaces. Whilst the effective use of transformation models might be somewhat limited to simple types of user interfaces (such as based on Web forms), they have been demonstrated of interests for generating multi-target platforms [151][261].

- Development of unified user interface description languages: This alternative consider the development of a unified language that encompasses multiple modeling notations, each one covering a particular views of interactive systems. This approach might not reduce the number of models and notation but some problems related to the semantic of concepts can be reduced if models are assumed to be part of the same unified language. We might consider this solution as similar to that which in the past has led to the development of Unified Modeling Language (UML) in Software Engineering [166]. USIXML [132] and IFML [167] can be cited as an example of attempt for unifying models for engineering interactive systems. USIXML follows the principle defined by the framework Cameleon [42] and it supports several level of abstractions of models (including *task models*, *abstract user interfaces* (AUI), *concrete user interface* (CUI) and *final user interface* (FUI). By appropriate tool support (which does not exclude model transformation) it is possible to refine abstract user interface elements into more concrete specifications. Multiple levels of abstraction offer a certain degree of flexibility for accommodating constraint and solving design issues without having to focus on a specific target platform.
- Domain Specific Languages (DSL). A Domain Specific Languages (DSL) aims to facilitate the construction of software artifacts thought specialized abstractions and notations that integrate all the concepts required to solve a particular type problem. DSLs are claimed to bring important productivity improvements to developers and the increased usability is regarded as one of the key benefits of DSLs. Nonetheless very few studies about the usability of DLS exists [7].

4.4.3 Degree of formality of models

Another factor of diversity of modelling languages is their degree of formalization. It quite remarkable that the usages of models are somewhat limited by their degree of formality. While informal models such as mock-ups/blueprints are suitable for communicating design options with users, clients and/or stakeholders, they cannot be considered for rigorous analysis [251]. Semi-formal models such as task models still require external/human interpretation to become operational but they have been demonstrated useful for specifying and assessing scenarios that should be supported by interactive systems [251]. Formal methods leave no room for interpretation so that models can tell whether a particular sequence of activities are possible or not, which enable their use for enacting (or controlling) the execution of interactive systems [246].

Although informal and semi-formal models have their utility during the development process, we acknowledge that formal methods are of particular interest for the research on interactive systems. As wisely argued by Alan Dix [65], formal methods make explicit every possible design decision thus helping to prevent errors that can be introduced during the development process if developers have to interpret incomplete or ambiguous specifications. Indeed, a high level is formality is required to build computing tools that present a deterministic behavior. Moreover, we argue that if a design solution is meant to be reused, it is better to be formally described. This statement is valid not only for developers who must a formal specification to be sure they are correctly implementing the design solution, but we suggest it should become a mantra for researchers in the field as it offers a meant for systematizing design solutions and the knowledge acquired after analyzing existing alternatives.

There is a vast literature on formal models for dealing with interactive systems and the foundations can be found in the books of Alan Dix [64] and Harrison and Thimbleby [96]. For a comprehensive example about many possible uses of formal methods, Palanque and Paternò [181] compile a large set of solutions based formal models that have been applied to a case study of describing a Web browser. Former conference series DSV-IS[13], TAMODIA[14], CADUI[15], and now merged into EICS[16], and other workshops offer a venue for articles in the matter.

Despite a significant interests of the scientific community in the 90s', the take up of formal methods for the engineering of interactive systems has been pretty slow. Formal methods have been regarded as being difficult to understand due to unfamiliar symbols and interpretation rules that are not apparent to practitioners [195]. Costs is also a harsh counter-argument quite often used [118] for that the only strong motivation for using formal methods is applications on safety critical domains such as Health, Transportation, Command and Control, and Aerospace. Nonetheless we suggest that those problems can be overcome with development of syntactic sugar and appropriate tool support for hiding and disclosing the complexity underneath formal model speciation according to the needs and or capacity of understanding of practitioners using models. The challenge is to provide a lightweight interface and interaction models that internally embed formal models in such a way they become palatable for user interface designers. We consider that this one of important aspect that could orient the research in the field.

[13] DSVIS: Design, specification, and verification of interactive systems (run from 1994 to 2008).

[14] TAMODIA: Task Models and Diagrams (run from 2002 to 2009).

[15] CADUI: Computer-Aided Design of User Interfaces (run from 1993 to 2008).

[16] EICS: ACM Symposium on Engineering Interactive Computing Systems (started in 2009, see http://www.eics.org).

4.5 Contribution to models for engineering of interactive systems

My interests on models for engineering interactive systems increased during my PhD thesis (2004) [241] and since then models played a central role in my research activities. In the last years I have extensively investigated the use of model-based approaches for improving the communication between stakeholders and members of the development team [244][251], assessing interactive systems [29][168][198][265], specifying complex interaction techniques [157], providing user guidance and supporting training [138], and predicting the effects of interruptions during the execution of interactive systems [172][172][228]. More precisely, my research activities was particularly focused on *dialog models* and *task models*. The next sections describe a selection of three of my contributions on the matter which includes: the research around a dialog modelling notation called StateWebCharts (SWC), the proposal of the task model notation HAMSTERS, and lately the weaving between dialog models and task models.

4.5.1 StateWebCharts: a dialog modelling notation for the Web

StateWebScharts (SWC) is a modelling notation for describing navigation aspects of Web applications [255]. *Navigation* is a special kind of dialog behavior which occurs when a user moves from a document to another, which is commonly found in hypertext-based systems such as the Web. SWC was proposed as part of my PhD [241] as a solution for describing idiosyncrasies of Web navigation which ultimate include a dialog between a Web client (i.e. Web browser) and one or more Web servers, the dynamic generation of contents of Web documents and hyperlinks that allows seamless connectivity of two distinct Web applications. For that I have extended the Harel's statecharts notation [255] by adding two main features: a semantic representation for states for describing the way which Web contents are generated and integrated into a Web page (i.e. static, dynamic, transient and external), and an explicit description of agents triggering the navigation (i.e. the user or the system). With these extensions we are able to describe the navigational behavior of Web applications.

SWC features a contribution not only in terms of a conceptual model but also in terms of modelling language. As a conceptual model, SWC allows to tackle concepts that are specific to Web applications. Moreover, SWC offers a graphical notation featuring a modelling language and a pragmatic approach for representing the navigation of Web applications. The operational semantic of SWC notation is embedded into a tool support called SWCeditor [246] which allows the edition, visualization and simulation of SWC models, as illustrated by Figure 16. The operationalization of SWC models via a tool support allows the investigation of many other interesting research questions about the use of navigational models for engineering Web applications. Indeed, as we have suggested in [247], the information provided by SWC models can be employed by model checker tools to analyze some system properties concerning the usability, analysis of path navigability and comparing alternative navigational paths. Using the SWCeditor we also have explored the use of navigation models for automating the cognitive walkthrough method for inspecting Web applications [246]. We also have developed a dialog controller that is able to process SWC models deployed at the server side [258]; such as an approach allow SWC models to be used to control the access of documents stored on a Web server and it supported our investigation on executable models. The use of SWC models for represented raised several interesting research questions that have been addressed by the PhD thesis of Florence Pontico and Joseph Xiong that I have co-supervised in the period from 2004 to 2008.

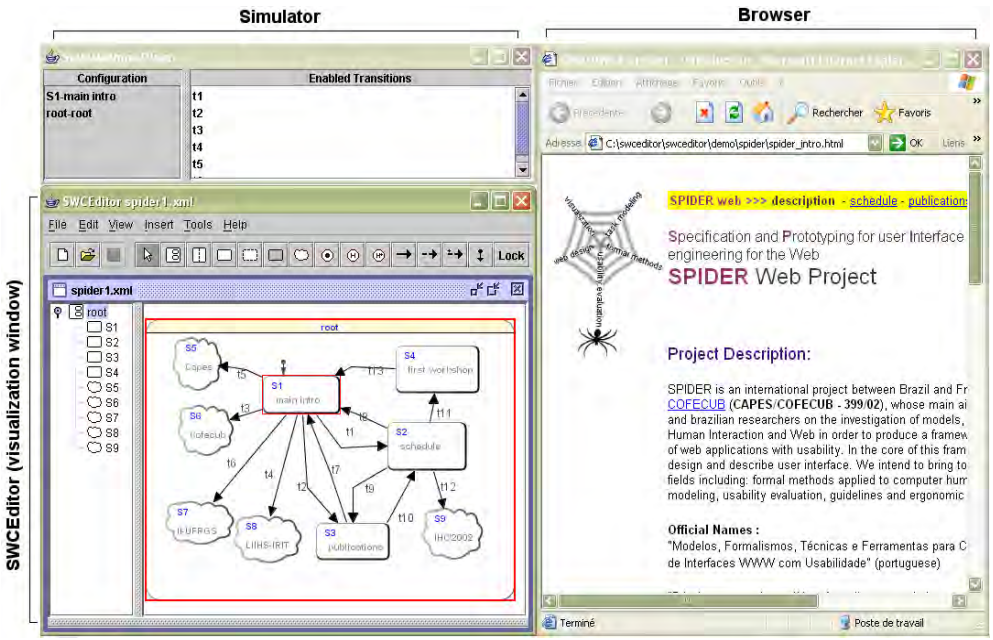


Figure 16. Simulation and execution of Web applications using the SWCeditor.

During the PhD thesis of Florence Pontico [186] it was investigated the use of SWC as a user interface description language for formalizing user interface (ui) patterns. This study was motivated by the fact that ui patterns available in the literature are often delivered without any kind of formalization, allowing diverse interpretations and leading to implementations that cannot be automatically cross-checked (with respect to the recommendations provided by guidelines). Similar applications presenting different navigation might be accountable for inconsistencies and lack of coherence, which are quite well known sources of usability problems [161][162]. In a large case study run during the PhD thesis at the agency SmalS-MvM[17] we have found that inconsistencies between similar applications were indeed due to the lack of guidance provided to designers and developers trying to use ui patterns [249]. For that we have proposed a template for systematically organizing ui patterns [190]. The description proposed for ui patterns is rather classical (advices of implementation and rationale around a given ui design pattern) but it introduces an attribute called *wireframe* which might assume different forms depending on the nature of ui pattern at a concern. For example, whilst ui patterns describing recommendations for pages and/or basic components (ex. buttons, logos, etc.) hold a *wireframe* attribute featuring a very schematic user interface representation, *wireframes* of ui patterns describing navigation contains SWC navigational diagram. The template structure was applied to a large set of ui patterns that were delivered at SmalS-MvM in the form of universal catalogue of ui patterns dedicated to e-government applications [189].

One of the first results of the PhD thesis of Florence Pontico was to demonstrate that SWC models are useful for describing the navigational UI patterns. But the research didn't stop here. By the means of a dedicate tool, which is described in the PhD thesis of Florence PONTICO as eGovPIM [186], we have demonstrated how designers can reused and modify SWC models to generate low fidelity prototypes. We suggest that these rapid prototypes can help the communication with clients and designers in initial phases of the development process for checking initial ideal about the design. Moreover, as those prototypes are generated from a template taken from the universal catalogue we can be considered consistent with respect to the ui pattern. This lead to the investigation of two subsidiary research questions: How to use models for guiding the design of prototypes? And, how to support flexibility into ui patterns when designers have to accommodate an arbitrary number of pages in a navigation trail according to the needs of the application in development? For that, an extension of SWC models was proposed to allow to specify which states and transitions can be modified without broking the navigational principle defined by a UI pattern.

The PhD thesis of Joseph Xiong was focused on methods for automatic inspection of accessibility and usability guidelines at different phases of the development process of Web applications [265]. The underlying motivation for this PhD thesis is that the correction of usability problems found after an application has been deployed is known to be expensive and time-consuming [33]. Our goal was to determine in which extension it would be possible to anticipate the detection of usability and accessibility problem in earlier phases of the development process and in particular during the phase of design. Several artifacts and models, including SWC models, were employed in the study [268]. Giving the fact that a substantial number of usability problems reported to Web applications refer to navigation, SWC models were considered suitable artifacts for the study. Moreover, many usability problems related to the navigation (such as broken links to pages, or long paths between two stats) can be expressed in terms of integrity properties (such as the reachability of states). Thus, we assume that by checking such integrity properties in a SWC model we can infer the occurrence of navigation flaws before the implementation of the system. In order to support the assessment of models, we have generalized the concepts we want to check by the means an Ontology [267]. The concepts in that Ontology were extract after a careful inspection of a large set of ergonomic guidelines for the Web which includes the W3C/WAI Content Accessibility Guidelines and a guidelines compilation issued from the project EvalWeb [209]. After that, all the guidelines were rewritten in the same terms concepts defined in the Ontology. Using mapping tables, the constructs of SWC models have also been associated to concepts of the Ontology. An appropriate tool support as then used to inspect SWC models with respect to the guidelines.

The PhD thesis of Joseph Xiong presented both academic contributions and impact in the industry. First of all, it enlarges the use of conceptual models for building Web applications [267]. That PhD thesis proposed a method and a tool support for dealing with inspection of models along development process [266]. The results show that ergonomic guidelines can be inspected in early phases of the development process [268]. This PhD thesis was funded under a contract CIFRE[18] in cooperation with the company Génigraph which embeds the results into the framework e-citiz[19]. Thanks to the methods developed during the PhD thesis of Joseph Xiong, the framework e-citiz received a prize "*TIC du Salon des Maires et des Collectivités Locales 2005*".

---

[17] SmalS-MvM is a non-profit organization devoted to the design, deployment and handling of public e-Government applications in Belgium. It employs more than 1.000 persons, mainly administrative staff, and software engineering experts. Further information at https://www.smals.be/.

[18] CIFRE - *Conventions Industrielles de Formation par la Recherche* – is a program that fund PhD thesis developed in cooperation with the industry. Further information at: http://www.anrt.asso.fr/fr/espace_cifre/accueil.jsp?p=1

[19] Further information about Génigraph, now Softeam-Cadextan, and the framework e-Citiz at: http://www.e-citiz.fr/

4.5.2 HAMSTERS: contributions to the development of a new task model notation

HAMSTERS is a task model whose acronym stands for Human-centered Assessment and Modeling to Support Task Engineering for Resilient Systems [144]. HAMSTERS was inspired by existing notations such as CTT, K-MAD and AMBOSS, from which it borrow many concepts but it also features several extensions of the conceptual task model to cope with fine-grained description of user tasks. For example, HAMSTERS borrows from CTT the typology of abstract tasks, system tasks, user tasks and interactive tasks. However, those constructs are much more specialized; for example user tasks are decomposable in *motor* tasks (ex. physical activity), a *cognitive* task (ex. decision making), or *perceptive* task (ex. perception of alert).

The research that leads to the development of HAMSTERS started in 2009 as a student project but the major developments were accomplished during the PhD thesis of Célia Martinie (2009-2011) [137] which I co-supervised. Figure 17 presents a timeline with the evolution of the notations HAMSTERS. That timeline covers the period of 2009 to 2015 and only takes into account the appearance of documents describing the evolution of the conceptual model and the corresponding graphical notation. Every subsequent extension of the notation gave place to a publication in a scientific conference such as EICS [13][138], ATACCS [140], ECCE [143], HCSE [79] and INTERACT [70][144]. We should notice that extensions not only affect the conceptual model but also drove the evolution of the tool support which is eponym of the HAMSTERS notation.

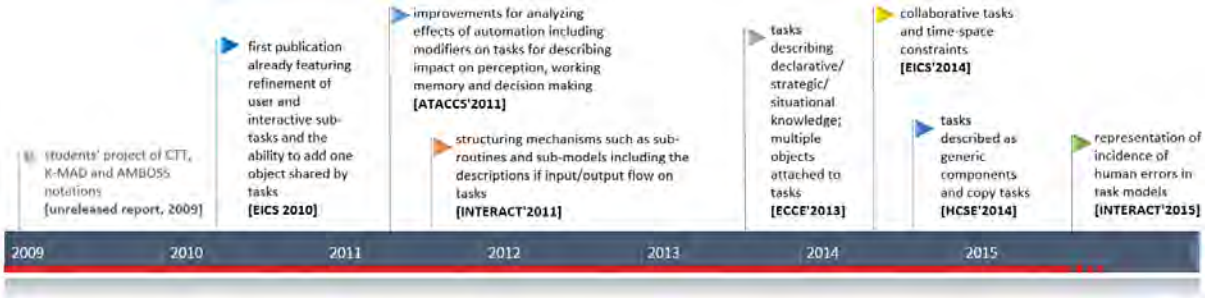


Figure 17. Evolution of the HAMSTERS notation (2009-2015).

I have accompanied the development of HAMSTERS since the beginning but I have particularly contributed with the structuring mechanisms that have been proposed for helping users of task models (mainly the members of the development team) to deal with the complexity of large systems [79][144]. The rationale for these extension is the following: if the activities to be represented are complex, the resulting models might become complex too. Complexity is a recurrent problem with model-based approaches that might require significant availability of resources which is sometimes perceived as too much effort, too long to produce and not being cost effective enough. Besides the development of tool support for editing and simulating HAMSTER models, we have also addressed complexity at the notation level. In previous work [138] we have discussed some basics mechanism allowing abstraction and refinement of task models is we have proposed two mechanisms (i.e. sub-models and sub-routines). More recently in [79], we have revisited these mechanisms and we propose a third one (component) that provides a powerful mean for reusing models parts. These three mechanisms are aimed at supporting rapid task-model development by structuring models and improving reuse of existing models. This work is driven by the premise that the development team might have special needs for working with task models in an efficient, effective and satisfying way. By improving the usability of tools and the task model notations we expect to help developers to focus on the analysis of end-users tasks and free them to the burden of managing complex diagrams.

It is worthy of notice that such as extensions are highly influenced by real life cases studies issued from project with the industry (mainly CNES and AIRBUS) and motivated by the possibilities for improving model-based task analysis. For example, some of the studies around HAMSTER models aim at determining the costs of task migrations (when the responsibility for a task is transferred from an agent to another) which is particularly important for analyzing the tradeoffs of task automation (when a task performed by the user is transferred to the system). At a first sight, automating tasks might discharge users from an activity, but is not always the case: for example, using an ATM system requires users to perform additional some cognitive tasks (e.g. remember password). By decomposing the cost in terms of cognitive, motor and perceptive tasks we expect to have a better understanding of the real costs of task automation for the users. Thus, extension of user task were are required for performing the simulation of user's tasks and thus predicting users' performance with interactive systems [13].

The research around HAMSTERS follows its course. Currently some colleagues in my research group are working on the representation of procedural, situational, or strategic knowledge required by users to accomplish certain tasks [143]. Other studies still in progress focused on the representation of collaborative tasks [138] and human error [70].

.

4.5.3 Weaving dialog models and task models

Dialog models and task models provide two complementary views about how interactive systems work. As presented in section 4.3, a dialog model describes how the system process user's inputs and how it produces the corresponding feedback. For task models, they inform everything users must to do (i.e. to perform, to know, to behave, to cooperate with others, etc.) in order to accomplish their goals with the software. Dialog models and task models are closely related because, at a high level of abstraction, both describe a sequence of actions and/or interactions that might occur when the interactive system is running. Nonetheless, we cannot generate a model from another.

In [159] we have demonstrated that a task model might inform different design options for a system and if we try to generate the dialog from a task model we remove all creativity from the design process. Moreover, despite the fact that some parts of the dialog can be derived from task models using transformation rules [134], task models lack of substantial information (for example for handling events) and designers must intervene at some point to complete the dialog [261]. The other way around have been proposed by few authors [133] who could, at the best, demonstrate the extraction of simple sequences of users actions that are hardly enough to support the analysis of human activity with systems. Nonetheless, the views provided by task and dialog models must converge at some point, otherwise that will be a gap between what users are expected to do and how the systems responds to users' actions. For that strategies for weaving task and dialog models are very important for the engineering of interactive systems.

In our research we start by investigating how use scenarios extracted from task models as a bridge between those two views. That scenario-based approach presented in [169] extends previous the work of Navarre et al. [156] and it has been validated using different dialog notations. In a study concerning applications for Air Traffic Control [169] we have employed the CTT notation for describing tasks and ICO models for describing the dialog. In studies concerning Web applications we have used CTT too but replace ICO models by SWC models [257][260]. These studies demonstrated that scenarios present the necessary concepts for ensuring the correspondence between task and dialog models, regardless the notations we have used to describe the dialog.

A step forward in our research was achieved by supporting the co-execution of models. To this end we have developed a tool support for the task model notation HAMSTERS and used the CASE tool PetShop for building the dialog. We have shown that a full integration of two dedicated tools can be performed and that it provides many benefits both for the verification of the compatibility of the models and at runtime by supporting users' activity to reach their goals providing them with contextual information [13]. A full case study focused on Weather Radar System (WXR) embedded in aircraft new generation interactive cockpits validate the approach. That work represent a dual perspectives for future research in the field: on one hand it allows the investigation of executable models that can control interactive system at run time; on the other hand, it makes possible to visualize parts of models that are involved in the execution of the systems and how changes one model might dynamically affect other models and the system itself.

4.6 Specificities of models for dealing with Web applications

The Web was born as a hypertext system and this can be easily seen nowadays throughout the concepts related to navigation that permeates models used to describe the dialog behavior of applications in the domain. Notwithstanding, many current Web applications follow predefined business logic and complex transactional operations requiring integration with distributed databases and legacy systems. As a result of this hybrid heritage, the development of many Web applications needs to consider aspects typically found in document management systems (i.e. information architecture) and software engineering principles (i.e. functional architecture) [241]. It is interesting to notice that there are many types of Web applications and behind this generic term we might have interactive systems mingling aspects of data-intensive applications (when the interaction pattern *create/read/update/delete* is predominant), mashups applications (ex. an application built by assembling pieces of other Web sites), workflow-based information systems (applications driven by a business process), service-oriented application (such as search engines), informational Web sites (based on the navigation of static documents)...

To deal with the increasing complexity of Web applications several specialized models and model-based approaches have been proposed in the literature (such as OOHDM, OO-H, UWE, WebML…). For that, we have proposed the StateWebCharts (SWC) notation as presented in section 4.5.1. All those modeling methods are aimed at supporting a systematic development process for engineering Web applications. Nonetheless, we should recall another specificity of the Web: the fact that user interfaces should be processed by a Web browser which is charge of displaying contents as they have been transferred from the Web server. Interesting enough, the code source of user interfaces is accessible by the Web browser. The distribution of the Web application between the client and the server, the fact that the code source of the user interface is accessible at the client side and subsequently processed by the Web browsers, induces some challenges for the development of models describing the user interaction over the Web that goes beyond of the dialog defined for Web applications, as illustrated hereafter:

- The independence to the Web applications from the Web browser is one the cornerstones for ensuring the interoperability of the Web across many platforms. As the specificities of diverse platform are thus handled by Web browsers some problems might arise, for example contents of Web applications might be squeezed to fit on diverse screen sizes, interactivity can be degraded by browser that cannot process client-side scripts (ex. lack of JavaScript support), some contents might not be displayed due to client-side restriction (ex. blocking pop-ups)… Following a truly user-centered design approach requires to tune Web applications according to the restrictions of the browser. However, given the large number platforms, fabricants and versions of browsers available in the market, tuning applications to the browser would be cost-prohibitive. Moreover, given to the fact that many applications needs fast updates of contents, it would be an endless work to tune applications by hand. For automating the process, we need to increase the level of abstraction of components they use to build Web applications and use other strategies for tuning the contents according to the requirements of a concrete user interface [147]. Whilst plasticity is a property that can be available in many types of interactive systems [58], it becomes an everyday concern when building Web applications. As we have illustrated in [41], the need of plasticity might have a huge impact on the user tasks and in the dialog and it is important take care for not degrading the usability of Web applications.

- We must consider the multiple layers of dialog that are behind the execution of Web application which might encompass the Web application itself, the Web browser and the presence of add-ons/plugins [62][73]. In the first layer we consider the navigation as is was specified and implemented by the developers of Web applications deployed at the Web server. In the second layer, we have the user interaction with a Web browser that displays the Web application's contents as they have been transferred from the Web server. The use of navigational mechanism available at the browser level (such as history and backtracking mechanisms) might affect the overall user experience with respect to the navigation in Web applications. Finally, we also need to consider that specialized add-ons/plugins might have been installed by the user on the top of the Web browser and they modify the behavior of the Web application and even the behavior of the Web browser. So that, a model aiming to describing the whole user interaction with an application delivered on the Web would better take into account all those three layers.

- Web application are distributed between a client and one or more Web servers and heavily dependent of network connectivity. However, there are many situations where the tasks users have to perform online with a Web application might be interrupted due to, for example, an unexpected loss of connectivity (ex. network failures), temporary unavailability of the Web server (ex. for maintenance purposes), external events that prompt users to interrupt online tasks (ex. mandatory turn off of electronic devices during takeoff/landing…), etc. The availability of local storage of Web site contents previously downloaded by the browser might suggest that users can perform some of their tasks when offline. Nonetheless several technical constraints (such as contents like video streaming that are only accessible online, predefined timestamps for connections, history of navigation, etc.) might prevent users of efficiently resume their tasks over the Web after an interruption. Such scenarios of interruption of connectivity are frequent with Web applications and might dramatically affect the user interaction. For that we are concerned by conceptual models that are able to describe the behavior of Web applications that can partially executable at the client-side (i.e. the Web browser) after an interruption caused by loss of connectivity [6].

- There are many processes over the Web that require a high level of collaboration between users and can only be achieved by orchestrating users' tasks. Indeed, many tasks users engage over the Web involve dealing with different Web sites; for example, planning a simple trip would require the visit of a first Web site for booking a hotel, a second one for booking flights and many more for finding interesting sightseeing at destination... Moreover, once users figure out the procedures for planning a trip, they might want to share it with friends and colleagues. Despite the fact that users would consider such Web navigation as being part of the same task (i.e. planning a trip) most Web sites will run independently with little support to the actual users' concern [73]. This example highlights two important aspects about the use of the Web as a platform: i) users tasks are distributed across many disjoint Web sites; ii) users share information to accomplish a common goal. Orchestration is therefore a special kind of behavior that encompasses intra-application dialog (i.e. actions users perform in a Web application to accomplish a goal) and inter-applications dialog (i.e. information exchange between independent applications). We have proposed in a previous work [76] a Domain Specific Language (DSL) aimed at describing procedures that are aimed to orchestrate user tasks over multiple Web sites. That DSL combines the description of manual task (i.e. simple instructions tell users for to perform a task on a Web site) and automated tasks (i.e. tasks that can be automated by using tools built under the concept of Web augmentation [70] and that support the integration between two independent Web applications). The dependencies created by the orchestration of user tasks between disjoint applications might lead to a complex triangulation between the actors that involved in the development of Web applications as discussed in [71].

This list above is not exhaustive but it is illustrative enough to induce the interests for dedicated models and model-based approaches for engineering Web-based interactive systems.

4.7 Research agenda

Models help to deal with the increasing complexity of systems and for that reason they are at the core the research in the field of the Engineering of Interactive systems. Special needs for representing concepts required to cover a particular application domain and/or improving the usability of notations is often accounted for the evolution of models. However, it is interesting to notice that the development of models should follow the evolution of technology and its usages. For example, with the advent of ubiquitous computing technology new concepts such as positioning and network coverage have been included models for description interactive systems. Evolution of usage of applications such as the fact that users become producers and not only consumer of contents of Web applications also might leads to the redefinition of the concepts for covering end-user programming.

Overall, the list of topics that we could put in a research agenda concerning model and model-based approach is definitely huge. Nonetheless, we are particularly interested in models able to cope with the idiosyncrasies of interactive systems deployed in the Web domain. As we have discussed in section 4.6, there are many reasons that explain why we need dedicated models for describing concepts in the Web domain. In a short run, these concerns can be translated in the following topics:

- The problems related to interruptions caused by loss of connectivity is going to be addressed in the PhD thesis of Félix Albertos Marco who I co-supervise in a collaboration with the *Universidad Castilla-la-Mancha*, Spain. The research questions concern models able to support the description of a modal (online/offline) behavior of Web applications. For that purposes, features of the Web and the capabilities are described and analyzed all together as we have suggested in [6]. On one hand we expect to propose models that allows as to analyze the impact of interruptions in human work over the Web. On the other had the ultimate goal is to support the development of Web applications that are resilient to interruptions caused by the loss of connectivity and as a consequence support at the best the users to perform their tasks even when offline.
- The problems related to the orchestration of user tasks distributed across disjoint Web applications was initiated with the PhD thesis of Sérgio Firmenich [71] in a collaboration with the *Universidad Nationale de La Plata* (La Plata, Argentina) and is going to be pursued in the PhD thesis of Iñigo Aldalur in a collaboration with the *Universidad of País Vasco* (San Sebastian, Spain). The ultimate goal is to provide a support for describing complex user tasks that can only be achieved by orchestration of Web sites. For some of these tasks we assume that users should be Web to program scripts (or at least being able to tune existing scripts according to their needs). We expect to provide models that are able to describe complex user tasks in terms of intra- and inter-applications dialog and support the description of fine-grained behavior of scripts aimed at automate the orchestration of tasks.

In a long run, we expect to contribute with models and model-based approach that can integrate the views about the Web development. In the way, we will pursue the analyses of models used in other applications domains. We believe that looking around models and problems solved by model-based approached in other domains it might be possible to get insights and borrow ideas to solve problems in the Web domain. This is indeed what we have experienced when investigating task models in safety-critical systems who inspired our research on the Web domain. Moreover, in an even long run, we expect to lever our understanding of the use of models for building interactive systems in a border sense, beyond the borders of a particular application domain and/or technology.

5 Assessing interactive systems

According to the standard ISO 13407 [109], the development of user-centered interactive systems can only be achieved by an iterative cycle where phases of design and evaluation alternate during the development process. The ultimate goal of such evaluation cycles is to understand how to improve the systems until its design fulfil users' needs in terms of usability [63] or as defined by to the ISO standard 9241-11 "The extent to which a product can be used by the target users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use" [105]. This chapter examines the overall contributions of usability of evaluation methods for the development of interactive systems, the shortcoming with existent methods and the perspectives for future research. We start by providing a brief overview of the state-of-the art in usability evaluation methods. Then it discuss how other user interfaces properties (such as accessibility, user experience, privacy, resilience, etc.) that, beyond usability, should be taken into account during the design and development to fulfill particular requirements for interactive systems. The rest of the chapter describes our contributions in terms of usability evaluations methods and the agenda for future work.

5.1 Overview of methods for evaluating interactive systems

The HCI community has been prolific in the development of usability evaluation methods (UEMs)[20] to support the continuous assessment during the development process of interactive systems. During the Cost Action 294-MAUSE[21] we have analyzed and compared a comprehensive set of usability evaluation methods (Law, Scapin, Cockton, Springett, Stary, Winckler, 2009) [129]. In the MAUSE project we have found a considerable variability among usability evaluation methods. It was interesting to notice that some methods are more suitable than others to be employed at certain phases of development process (for example inspection methods can be employed even with low-fidelity prototypes in early design phase whilst model-based analysis will required a more complete describe of the interactive system). Moreover, the outputs provided by different methods can also vary considerably, for example: by providing a simple metric about usability, by giving a list of usability problems to be fixed in the interactive system, or by proposing a set of design guidelines (or recommendations) that are aimed at systematize new knowledge about how to enhance the interaction with the systems.

Usability evaluation methods can be presented in many different ways but in the present work we follow the classification originally published in (Bernhaupt, Navarre, Palanque and Winckler, 2007) [29] as illustrated by Figure 18. The list of evaluation methods presented hereafter is far of being exhaustive and does not take into account adaptations of methods to cope with the idiosyncrasies of non-traditional environments (such as the living room and mobile applications) which is the subject of the work done by Bernhaupt (2009) [27]. However assume that the description of methods presented here is fair enough to support the discussions about the contributions of usability evaluation methods to the engineering of interactive systems.

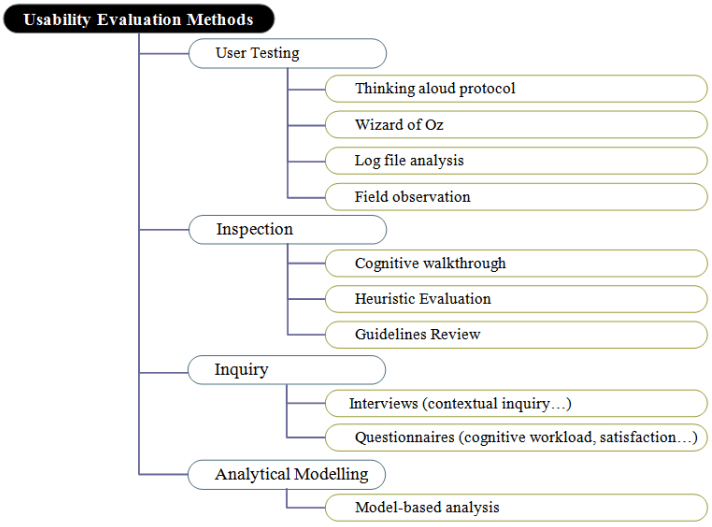


Figure 18. Overview of evaluation methods.

[20] Besides usability, some usability evaluation methods (UEMs) can be used to assess other user interface proprieties such as UX. Along this chapter the term usability evaluation methods is used to refer to any method allowing to assess interactive system. The mention to the evaluation of specific properties will be done explicitly whereas the distinction is necessary.

[21] Cost Action 294 MAUSE, which stands for *Towards the MAturation of Information Technology USability Evaluation*, was a cooperation project from 2005-2009 with the mission of bringing more science to bear on the development of usability evaluation methods (UEMs). Further information at: http://www.cost294.org

5.1.1 User testing

The methods classified as user testing imply that user activity is observed and recorded while users are performing their tasks. The observation can be based on non-intrusive methods such as log file analysis or based on direct observations of users performing tasks that can be predefined and run in controlled settings (ex. thinking aloud protocol, Wizard of Oz) or have users performing their usual tasks in their working environment (ex. field observation).

In many cases, user testing seems to be a preferred strategy for evaluation interactive systems because it allows the investigation of how users adopt and interact with technology providing valuable information about the general usability and the user experiences [204]. Indeed, direct or indirect observation of users provides valuable information about how users use and perceive the system in a given moment of time. As a snapshot, such observations should be repeated several times along the development process to understand user's needs and requirements evolve over time due to training, influence of social and environmental aspects. Therefore, user testing are time consuming, costly and demanding in terms of logistics, which often lead to restrain the number of participants and the number of scenarios to be tested in a single evaluation [156]. Thus, users testing methods might lead to the lack of representativeness (if the sample of participants is too small or not representative of the target population of potential users) and completeness (if only a subset of scenarios is assessed).

The automation procured by methods based on the analysis of log files make the collection of user traces easier; however, such indirect observations of users trace alone are often hard to analyze an interpret in terms of usability problems mainly because it is very difficult to know who are the users, what are the tasks they are trying to perform and how they perceive the systems [111]. Nevertheless, some of these shortcomings with file log analysis can be relieved by combining user traces collection with online questionnaires [252]. In addition to that, some recent techniques supporting the creation of users portray by exploiting Big Data and social networks [175] suggest that log file analysis might play a more important role in the development of new methods for observing users, at least on the Web.

5.1.2 Inspection

Evaluation based on inspection methods assumes that human-factors experts rely on ergonomic knowledge provided by guideline recommendations or on their own experience to identify usability problems while inspecting the user interface. Inspection methods are performed by evaluators without any direct user involvement [162]. These methods can be employed with full-fledged prototypes but also produce good results with low-fidelity prototypes which represents an asset for identifying usability problems in early phases of the development process. Among the methods in this category are worthy of mention the heuristic evaluation [156], cognitive walkthrough [162] and guideline reviews [18].

Whilst some approach such as cognitive walkthrough are focused on the systematic inspection of the user interface through the execution of tasks by evaluators that try to assess its understandability and ease of learning, other inspection methods rely on knowledge and expertise of evaluators who can be guided by heuristics (in the case of heuristic evaluation) or guidelines (as in ergonomic review).

Heuristics and guidelines are a way to alleviate the lack of expertise by prompting evaluators to look at the user to certain features that can be interpreted as usability problems. However, even experts experience difficulties in applying guidelines, at least in the format in which they are conflict with one another because there is a wide gap between the recommendation (ex. "make the site consistent") and its applications [111]. In order to overcome these limitations of manual inspection, much effort has been devoted in the development of tools supporting the collection and the organization of guidelines [233], guiding the inspection [36] and supporting automated guidelines inspection [207][265]. One of the advantages of employing automated tools for guidelines inspection is that no extensive knowledge on ergonomics is required since most the technical details are embedded into the tools. However, due to the inner nature of guidelines, it seems clear that not all guidelines can be fully automated and some of them still require interpretation and manual inspection. For Web applications, the automation limit of full inspection has been estimated to ~18% with some partial support to 49% of guidelines [268]. Nonetheless, the lack of experts to perform usability evaluations and the high costs of other evaluation methods (e.g. user testing) make automated inspection of guidelines a suitable method for supporting frequent evaluations of Web applications [250].

5.1.3 Inquiry

Inquiry methods encompass methods based on interviews and questionnaires that are omnipresent in the HCI research as a mean to collect implicit knowledge that can hardly be obtained without asking users. For example, contextual inquiry is specifically designed to uncover implicit knowledge about work processes and often require a workplace visit [128]. The analysis of interviews performed using contextual inquiry can be intimidating and often required skilled person with social science training and some of the constrains for the applicability of this method are similar to user testing.

Whilst questionnaires can be used in a large sense to collect information about users, it is often particular interests for the research in the engineering of interactive those specialized questionnaires that have been developed to assess particular properties such as usability (e.g. SUS [37]), cognitive workload (e.g. Nasa TLX [91]), factors for adoption of the system (e.g. UTAUT [235]), user satisfaction (e.g. QUIS), UX (e.g. AttrakDiff [100] [101]), etc. In some cases the results are given in the form of numeric scores (e.g. SUS) or visual graphs (e.g. AttrakDiff) that can be used as indicators of the overall quality of interactive systems. Such indicators can be easily communicated with the development team but they don't explain what the usability problems are, which parts of the systems are affected by the problems and how to solve fix them. For this reason, questionnaires are often used in combination with user testing techniques. It is also worthy of mention that the use of specialized questionnaires imply that users had used the system, or at least a prototype, before answering the questions; so contrary to interviews, questionnaires are less suitable for early phases of the development process.

5.1.4 Analytical modelling

Evaluation methods based on analytical modelling exploit models used to specify the behavior of interactive systems as a means to predict flaws or defects that could compromise the usability of final application. For these purpose two main categories of models are worthy of being investigated: task models (that are used to analyzed user tasks with the system) and systems models (that are used to specify the behavior of interactive systems).

In previous work [246] we have identified four suitable strategies for analytical modelling of interactive systems: i) static verification of models for detecting problems such as model inconsistency, data dependency, etc., ii) dynamic verification which implies the simulation and/or execution of models for supporting automated walkthrough inspections, iii) verification of system properties that test the compatibility of models against predefined behavioral rules expressed as temporal logic, and iv) synergistic assessment of system models and tasks models to determine the feasibility of user tasks with the system specifications.

Model-based evaluations are of particular interest for the engineering of interactive systems because evaluation outputs can be directly associated to parts of interactive system (even if represented by model constructs), which helps developers to locate and fix defects found during assessments. Moreover, model-based evaluations can be in a large extent automated thus allowing systematic inspection of the models that will be used to build the systems. But before getting the benefits, some challenges should be overcome. One of the main challenges is find suitable modeling notation with the expressiveness power to describe the system behavior and with the corresponding semantics of constructs for allowing the identification of usability problems [265]. In previous work we have investigated these challenges for the assessment of Web applications. The expressiveness power for describing the navigation behavior of Web applications was met with the development of the SWC notation [255]. The required semantic for assessing SWC models was found by mapping concepts present in ergonomic guidelines (ex. "page") to the SWC constructs (ex. "basic state"). Therefore, guidelines like "*Each page must have a link to it*" can be translated into "*Each state must have a transition pointing to it*", which is a formal statement that can be verified under SWC models [265].

It is interesting to notice that model-based evaluations are performed over specifications, not over the final implementation. So that even if usability problems are detected by analytical modelling and fixed on the models, there is no guarantee that the final application will be free of usability problems at run time. For example, models describing the navigation over Web applications often lack of contents that are only available at runtime; thus, the automatic generation of content might include new links that can modify the usability predicted by evaluations performed over navigation models [246]. Methods that rely only on specifications of the system might lack of sufficient information to determine the metrics that better predict the occurrence of usability problems at run time. For example, users might have many strategies for navigating Web application to accomplish their goals. If we don't understand the user mental model for accomplishing tasks it is difficult to check if the paths specified over navigation models are indicators of usability problems or not. Nonetheless, goal-driven evaluations over model is made possible if we combine synergistically scenarios extracted from task models and navigation models, as we have demonstrated in [257].

Indeed, the outputs provided by models-based evaluation methods lack of empirical information from real users. For that reason more recent works propose methods that combine analytical modelling techniques with other user testing methods. For example, Paterno, Piruzza and Santoro (2006) [181] propose to combine task specification based on ConcurTaskTree (CTT) with multiple data sources (e.g. eye-tracking data, video records) in order to better understand the user interaction and the task models used to support the development process of multimodal user interfaces. Following the same attempt, in [29] we have combined model-based evaluation with user testing and cognitive walkthrough to investigate low-level interaction issues such as fusion/fission of events and temporal behavior that make multimodal user interfaces difficult to evaluate. Finally, model-based evaluations imply the availability of appropriated tool support. Moreover, model-based evaluations are often considered costly, notably because the construction of models is demanding and time consuming. These limitations restrain the use of model-based evaluation to application domains were the costs and the risks associated to usability problems are high enough to justify their use (e.g. safety and critical systems). These constraints can be balanced by the fact that

model-based evaluations can be performed in earlier phases of the development process thus reducing costs of fixing problems in fully implemented applications and reducing the risk of potential hazards caused by human error, for example [195].

5.2 Assessing multiple properties of interactive systems

Whilst usability has been prominent in early HCI research other properties such as accessibility [237] and user experience (UX) [101] are nowadays also at the core of the research agenda in the field by extending the scope of the assessment of users' needs with respect to the systems. For example, whilst accessibility addresses the needs of impaired users to accomplish their tasks with the system [36], UX goes beyond the pragmatic aspect of usability by taking into account dimensions such as emotion, aesthetics or visual appearance, identification, stimulation, meaning/value or even fun, enjoyment, pleasure or flow [28][101].

According to the interactive system at concern many other user interface properties might become relevant and/or major factors for user's adoption of interactive systems. For example, in a study concerning e-voting systems [248] we have examined the relationship of user interfaces properties including usability, user acceptance and trust and other related concepts like security, reliability and privacy which have an direct impact on the design of the user interface. It is worthy of mention that whilst some properties such as usability (i.e. for casting votes) and privacy (i.e. no link can be made between a vote and a voter) were expressed as users concerns for the adoption of the system, others requirements such as accuracy (i.e. no vote can be altered, deleted or invalidated) and robustness (i.e. any number of parties or authorities cannot disrupt or influence the election and final tally) which are specific from the application domain of e-voting [48] cannot be directly attributed to users' needs.

The inclusion of new properties as requirements for the development of interactive systems call for suitable evaluation methods to cope with specific attributes and inner dimensions of such as properties. During the TwinTide project[22] we have investigated in which extensions usability evaluation methods could be adapted and/or extended to assess diverse user interface properties in a variety of different domains in which HCI design and evaluation plays an important role [130]. Despite the main categories of methods presented in Figure 18 remain relevant, many variations on method exist. For example, whereas usability emphasizes effectiveness and efficiency [105] UX includes hedonic dimensions in addition to the pragmatic ones [101], and is thus subjective attributes that can cannot be evaluated with stopwatches or logging. This is the reason why new inquiry methods based on probing [30] and questionnaires [100] have been developed to assess UX.

Dealing with multiple requirements expressed by users and/or issued form the application domain is a recurrent aspect of the development of interactive systems. And yet, the integration of multiple user interface properties presents many challenges. When two or more user interface properties are involved they might appear sometimes competing or in conflict (when the analysis of individual properties favors different design options). To illustrate this we recall a case study [146] used in in a previous work to analyze the design alternatives for granting users with access to data over a simple Web site. If only usability and accessibility are at concern, the obvious design solution would be to provide direct access to data over the web site (as this simplify the user' tasks and do not impose any constraint to impaired users). However, if privacy issues are also taken into account, the implementation of a login becomes mandatory and, ultimately, this design option will reduce the system usability (as logging into the systems takes times, recalling password is cognitive demanding and error-prone). It is interesting to notice that a simple login will not ensure completely the security of the Web site as it doesn't prevent spam attacks from bots. By adding a visual CATPCHA[23] to the Web site the property security is enhanced but usability is downgraded (as extra user effort is required to complete the login) and accessibility is totally compromised for blind users. In order to accommodate blind users an audio/visual CAPTCHA can be added, but this will not make the application more secure or usable. It would be possible to envisage more complex mechanisms for making the application secure but it not certain that users would still be able to use that application.

In conclusion, while in a user-centered design process we often try to highlight those properties that favor end-users, we should not forget that the underlying technology and idiosyncrasies of application domains have a huge influence of the development process of interactive systems. The solution to be adopted requires a deep analysis of all tradeoffs involved which only implies the level of priority given to properties but also the analysis of extra costs imposed by design alternatives that might downgrade the importance of other properties to levels that are bellow of acceptable by the users.

---

[22] Cost Action IC0904 TwinTide: which stands for *Towards the Integration of transectorial IT design and Evaluation*, was a cooperation project from 2010-2013 aimed at harmonizing research and practice on design and evaluation methodologies for computing artefacts, across sectors and disciplines. Further information at: http://twintide.org

[23] CAPTCHA (acronym for : completely automated public Turing test to tell computers and humans apart) is a program that protects websites against bots by generating and grading tests that humans can pass but current computer programs cannot.

5.3 Contributions to the assessment of interactive systems

My early research carrier started with the investigation of usability evaluation methods for the Web. During my master studies (1997-1990) I have proposed an evaluation method that combined log file analysis and online inquiry, which was aimed at better understanding the activity of remote users [252]. Through the years, I have examined diverse usability evaluation methods not only for Web applications but also for assessing a variety of application including e-voting systems [248], mobile applications [11], multimodal ground-segment systems [29][172], information visualization techniques [81][251]. In due course, I came to realize that, in most evaluations, there was a gap between the descriptions of problems (that could be found with existing usability evaluation methods) and the parts of the system that should be fixed. For example, by running user testing it is possible to determine the user threshold for performing a task with the system; however the results do not tell which part of the system should be tuned to increase user performance. In such cases, the solution to be found relied on the development of new design options for the systems that should pass a new round of evaluation to determine if the user performance lies in an acceptable range. Whilst iterations are cornerstone in user-centered development process, each cycle introduces costs with no guarantee that the problem will be fixed. The costs are even higher when the problems are found in late phases of the development process. In the attempt to overcome this gap, I have oriented my research towards the investigation of model-based approaches for supporting the assessment of interactive systems in early phases of development process. Hereafter I propose a summary three contributions in the matter. The first one concerns a method for automating the inspection of guidelines over navigation models. Then we discuss usability evaluation and co-execution of models. Finally we present our attempts at combining data collected through direct observation of individual as task model.

5.3.1 Model-based inspection of guidelines

In the context of development of Web application, evaluations are not only important for identifying problems before deploying the application but also for checking that no usability flaws are introduced whilst updating Web site's contents [114]. With the advent of specialized tools in the last decade, automated inspection of guidelines become the prominent evaluation method for assessing Web applications [36]. The underlying process for automating guideline inspection consist in interpreting user interface guidelines (such as "*Verify that links connect to existing pages*", source: Borges et al, 1996; "*Every page should have a link up to your home page*", source: Nielsen, 1993) and translate them into algorithms and software code for inspecting the HTML/CSS code [111]. It is worthy of mention that usability and accessibility become critical requirements due to the hard concurrency between Web sites (e.g. electronic commerce applications) and legal commitment for quality of information delivery (e.g. Accessibility responsibility of content published on the Web) [262].

In previous work [258] we have examined a variety of evaluation methods that would be suitable for assessing Web applications along the whole development process and we have suggested in [250][257] that guidelines inspection could be performed over navigation models used to build Web applications. The automation of guidelines inspection over artifacts requires the interpretation of guidelines with respect to the constructs available. In order to provide a formal and non-ambiguous description of concepts embedded into guidelines, we have proposed an Ontology that was stablished after analyzing a corpus of 208 guidelines issues from the W3C/WAI Content Accessibility Guidelines [237] and the set of guidelines compiled during the EvalWeb project [209] (which covered a comprehensive number of guidelines sources). In a first step, the Ontology was used for mapping guidelines to artifacts; for example, in a graph-based navigation model, an arc is interpreted as a "link". Then the guidelines are rewritten in a declarative form [267] using a XML syntax that can be processed by a tool based on a rule engines (i.e. Drools[24]).

This contribution to the automation of guideline inspection are at the core of the PhD thesis of Joseph Xiong (from 2004-2008) which was funded under a CIFRE[25] agreement in partnership with the company Génigraph [265]. The approach proposed in Joseph Xiong's PhD thesis explores mechanism for inspecting artifacts available at each phase of the development process. Such mechanisms have been fully integrated into the tool e-Citiz[26] developed by Génigraph so that applications generated with e-Citiz are inspected before deployment which guarantees they are free of technical accessibility flaws. Moreover, we have demonstrated that by exploiting artifacts such as navigation models an important number of guidelines could be taken into account in early phases of the development and in particular during phases of the development process where the participation of users is not straightforward (i.e. specification phases) [268][265]. From a scientific point of view, these studies trace the route for investigating questions about the expressiveness power and the semantic of models to support usability evaluation over artifacts used in early phases of the development process.

---

[24] http://drools.org/

[25] CIFRE : "*Conventions industrielles de formation par la recherche",* is a French program for funding PhD thesis that are developed in partnership with the industry. Further information at : http://www.anrt.asso.fr/fr/espace_cifre/accueil.jsp

[26] e-Citiz is a model-driven framework for developing administrative procedures that are deployed under the Web platform. Further information about e-Citiz and Génigraph at: http://www.e-citiz.fr/

5.3.2 Usability evaluation based on the simulation and co-execution of models

During early evaluation phases of development designers have to check if abstract modelling will behave as expected. In our work we contributed to methods that involve simulation and co-execution of models as a mean to support usability evaluation in early phases of the development process. The simplest strategy is to use simulations as a kind of automated walkthrough for inspecting a model [246]. Nonetheless, we have also investigated the co-execution of interweaved models that can be required for developing interactive systems.

In previous work we have demonstrated how navigation models and task models could be synergistically exploited to improve the development process not only of Web applications [251][246] but also interactive systems in general [13]. By automatically simulating scenarios over navigation modelling it is possible to verify the system's conformance with the user requirements specified through task models. The co-execution of task and navigation models extends the potential of model simulations by cross-checking the compatibility between different models used for specifying the interactive systems [156].

Indeed, the engineering of interactive systems can involve the production of various models such as task model, user model, domain model, dialog model, training model … that should not be considered independently as they usually co-evolve and represent different views of the same world. When formal description techniques are used, the process of verification and modification of models is iterative and iteration is conditioned by the result of formal verification. This allows proofs to be made on the system model in addition to classical empirical testing once the system has been implemented. Modelling systems in a formal way helps to deal with issues such as complexity, helps to avoid the need for a human observer to check the models and to write code. It allows reasoning about the models via verification and validation and also to meet three basic requirements notably: reliability (generic and specific properties), efficiency (performance of the system, the user and the two together) and finally to address usability issues (by means of tasks models for instance to assess effectiveness). The expression and verification of properties has been previously studied for formal notations in the field of interactive systems [64].

The complexity of usability evaluation over multiple models was addressed in a previous work [13] where we have proposed an approach for assessing properties of interactive systems by exploiting task and system models. Such as an approach relies on the inspecting of task and system models described using formal notations, as illustrated in a) compatibility of task models and system modelsb) compatibility of tasks, system and interruption models

Figure 7 (extracted from Barboni, Ladry, Navarre, Palanque & Winckler, 2010) [13]. The integration of task and system models at tool level implies the identification of basic bricks from both notations and existence of appropriated tool support. For that we have followed the recommendations stated in [156] for having tool level integration divided into two parts: the first is the editing of the correspondence between the two models while the second consists in a co-simulation of these models.

So far we have mainly addressed the compatibility between models via the co-execution of task and system models [13]. For that, we have proposed an approach that is illustrated at a) compatibility of task models and system modelsb) compatibility of tasks, system and interruption models

Figure 7 (see page 12) [172]. We suggest that approach can be extended in the future for exploring the co-execution of other models such as user models, training, requirement, and low fidelity prototypes.

In another work that illustrates the utility of co-model execution for assessing interactive systems, we have introduced a model-based analytical technique for investigating potential disruptive effects of interruptions on task performance in a multitasking environment [172]. We assume that a system A is called more interruption-tolerant than a system B if, for the same rate and type of occurrence of interruptions, the number of goals reached by a given user is higher than with the system B [246]. By employing stochastic methods and formal description techniques describing the interruptions behavior, user's tasks and the system behavior of the interaction techniques at concern, we are able to determine the number of effective user actions a user manages to perform during a fixed period of time under a varying number of interrupts occurring randomly during that period. The simulation of models using injected values allowed the identification of the threshold for interruptions for every interaction technique. That number of effective user actions was then used an indicator of the resilience of an interaction technique to external interrupts. For example, in our case study concerning of an application manipulating icons in a desktop environment, we have found that an interaction technique named Drag 'n Drop is better suited for low interrupt rates while another interaction technique called Speak 'n Drop takes the lead as the number of interrupts increases [246].

5.3.3 Triangulating task models, user scenarios and UX dimensions

Usability evaluation methods based on user testing are quite efficient for tracking a variety of usability problems and providing complimentary views about the knowledge ones might have about users and their needs. However, such information is often available through different formats which difficult the analysis and makes the mapping hard to connect problems to the parts of the interactive system at a concern. In order to overcome this gap, we have

start to investigate methods for triangulating information about users collected using empirical methods (ex. user testing and interviews) and modeling artifacts (ex. task models) that could be used to build the application.

In a series of empirical studies, we have collected data through user testing [11] and semi-structured interviews [244] that provide overwhelming evidence for corroborating the notion that scenarios reported by users contain information that could be classified according user experience (UX) dimensions. For example, when users' stories express a feeling of relief after performing a task (e.g. "…*"...under the influence of anger, there is a chance that I miss to provide the data that better describes the problem. So they [the system] should use a text field to require users to think a little and calm down...*") we can classify as the expression of the UX dimension "*emotion*". It is worthy of notice that scenarios often contain actions (e.g. "…*provide the data that better describes the problem* …") that might be mapped to task models describing the user activity. Based on the association of UX dimensions with tasks via the interpolation of user scenarios, it was possible to extrapolate the results in a single task model as shown in [244]. The mapping among methods was possible because it is easy to identify the concept of tasks in scenarios reported by users and the tasks behind the system.

By combining these methods, it was possible to provide a clear representation of the tasks and to point out wehre the UX dimensions emerge in existing applications, as illustrated by Figure 19. This aspect of our research is expected to help designers to understand which tasks are worthy of more attention in order to produce the expected UX result. Conversely, designers can focus on specific UX dimensions and look up the tasks that users are more likely to perceive as desired effects. It is worth noting that instead of using a specific application, we investigate a generic tasks model from which several scenarios could be extracted and then analyzed.

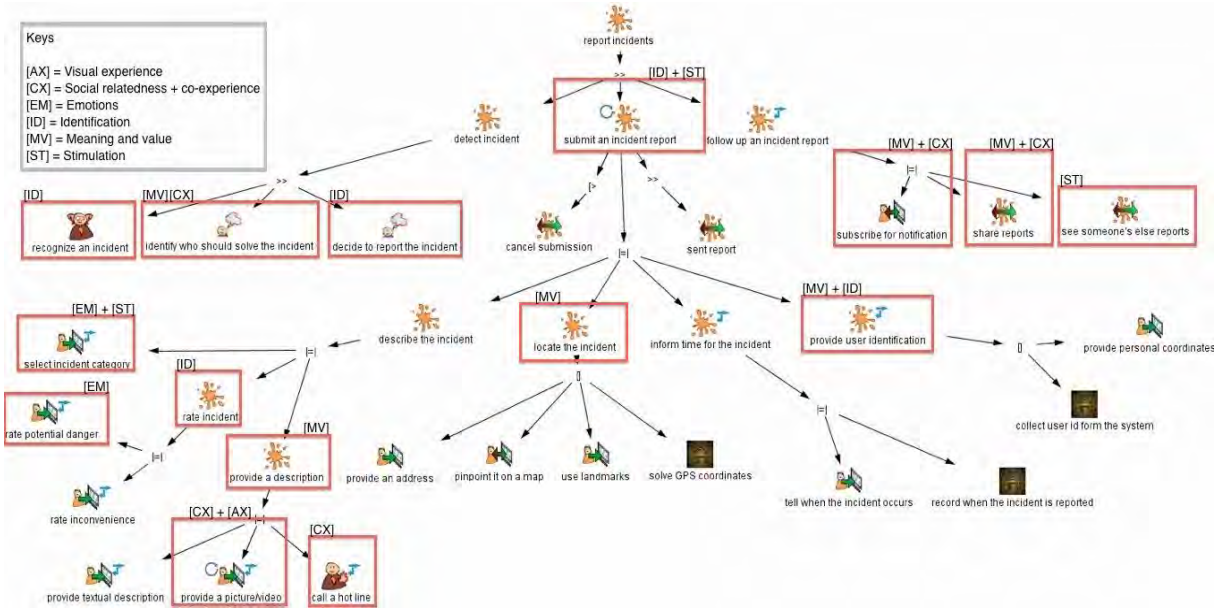


Figure 19. Articulation of task models, user scenarios and UX dimensions in the context of an application for reporting incident in urban context of use (Winckler, Bach, Bernhaupt, 2013).

5.4 Challenges for assessing Web applications

When Tim Berners-Lee presented his mockup for the Web in 1989 at the CERN lab, unconsciously he was giving a reason for everyone around the world to use a computer: information supply in large scale and connectivity as never made before. In the last years, the Web has grown exponentially (considering new users and server providers) but, unfortunately, it does not really mean easy-to-use interfaces. In fact, even experienced users are more and more struggling to find the information over the Web due bad design.

One of the main challenges for the evaluation of Web based applications are related to the diversity of users and diversity of platforms they use (from desktop to mobile phones). The problem is that Web presents some special features such as, fast content updating, diversity of technologies to build the interface, cross-platform issues and low budgets for Web projects, etc., that make difficult to apply the traditional usability evaluation methods. In spite of those features, traditional usability methods (such as Heuristic Evaluation and Users Testing) had been used with some changes and new methods based on remote evaluation have been proposed. The most recent developments in usability evaluation methods for the Web are concerning with:

- Low cost methods to make the evaluation cheap enough to be applied even for small Web project.
- Remote evaluation allows to reach users or evaluators outside of usability labs and in real work conditions taking advantage of distributed nature of Web resources.

- Automated or semi-automated tools can make easier and faster the evaluation for designers not experts in human-computer interaction. Tools also can help usability experts collect and analyze special data from user's interaction.

Other requirements are not so obvious but not less important are: a) Real users' involvement in websites projects in a UCD fashion; b) Data from real user interaction; and c) Subjective information about the interface. Nevertheless all effort made still now, the current methods and tools alone not ensure usability in Web projects.

Evaluations based on models have been scarcely developed for Web but have been proved their efficacy to help software engineering to build complex and efficient software. Even though create the interface using models could increase the project complexity, we suggests that is the best approach to integrate user interface requirements in data intensive Web sites. The advantages of models for design are clear. But it has some important advantages for the evaluation too: automatic verification and correction could be made over the code without design interference; some guidelines could be automatically verified and others, which could not be verified could be informed to users.

5.5 Research agenda

There is no universal method for assessing interactive systems. However, the currently available methods serve for a specific purpose and if chosen wisely it is possible to get good results with them, although for better results it always advisable to combine different methods along the development process… The development of evaluation methods is a long-standing topic in the research agenda on HCI and there is still plenty of room for improvements in particular with respect to the development of new methods for dealing with the idiosyncrasies of non-standard interactive systems. However, as far as the engineering of interactive systems are at a concern, we believe that some important issues must be addressed, as follows:

- To develop mechanisms for articulating evaluation outputs with design artefacts. As discussed in section 5.3.3, one of biggest drawbacks with the way evaluation of interactive systems are performed is the gap between evaluation outputs (and in particular those that refer to qualitative data) and the artifacts used to build applications. Our preliminary results [11][244] suggest that we could envisage a better articulation between UX dimensions, scenarios and task models. Further studies are required to investigate whether or not such articulation is possible with other properties and artefacts and what would be the possible outcomes.
- To investigate methods and approaches that is able to support a fair comparing multiple conflicting properties that might play a role in the development of interactive systems (such as usability, accessibility, UX, dependability, security, trust, etc. as discussed in section 5.2). We have already started to work in this topic by identifying conflicting properties that might affect the selection of guidelines used in inspections [146] but much more remain to be done in order to provide a clear analysis of the impact of multiple properties on design of interactive systems. Among the interesting question in the matter include: When to perform the assessment of multiple properties? Should they be evaluated at the same time or at different phases of the development process when the appropriate data for assessing them are available? Which indicators to use to compare properties? How to report conflicts between properties? How to document the tradeoffs and the design solutions for solving conflicts?
- To improve model-based approaches for supporting the evaluation of interactive system thought the development process in earlier phases of the development process. Based in our previous work we can demonstrate the contributions of model-based approach for the assessment of interactive systems. However, most of the existing solutions operate at the phases of specification when a task model or a system model is available. We suggest that, provided the appropriate underlying model-based specifications for low-fidelity prototypes, it would be possible to use analytical evaluation methods in earlier phases of the development process. This axis of the research is going to be explored in the PhD thesis of Thiago ROCHA who joined the ICS in October 2014 with a scholarship form the CSF program[27].
- To investigate in a long run the evolution of the impact of properties of interactive systems. It is quite well know that user perception about the system evolve overtime even after the system is deployed. We suggest that some theories about information diffusion [198] are worthy of being investigated in the context of usability evaluation methods. We believe that that by collecting and aligning all snapshots provided by evaluations in a long run could provide new insights about the impact of properties and design decisions for engineering interactive systems.

[27] CSF: "*Ciência sem Fronteira",* aka Science without Borders, is scholarship program funded by the Brazilian federal government which seeks at strengthen initiatives of science and technology through international mobility of undergraduate and graduate students and researchers. Further information at http://www.cienciasemfronteiras.gov.br/

6 Conclusions and future work

This work has been an initial attempt for trying to analyses and synthetize the problems related to the engineering of interactive systems across domains. In particular, we have investigate the foundations for organizing the research around four main pillars that include the development process, modeling languages, users and evaluation methods. A short term research agenda has been described in the end of each previous chapters. Hereafter we present additional research items to be accomplished in a medium and long run.

6.1 Development a framework to assess methods for engineering interactive systems

In a long run we want to reference chart modeling languages, practices of development process, tools and methods used to engineering interactive systems. For that we need a framework and a classification would help developers and research to identify the method and tools that best suits to their needs. This is a daunting task but we assume that the preliminary steps have been achieved in the present work.

However, rather than a personal classification of techniques, we expect that the community would contribute to the development of such as a catalogue with their knowledge and experience about tools methods and techniques. For that we are planning to develop collaborative tools where research and industrialist could report their own experiences with methods and tools.

I order to achieve this goal, in a medium run we plan to create a task force inside the research community through a Cost Action. Based on our experience in two COST Actions, MAUSE - Towards the Maturation of IT Usability Evaluation (http://www.cost.eu/COST_Actions/ict/294) and TWINTIDE (http://www.twintide.org/) we consider that kind of networking project is the best strategy for supporting the research activities. Moreover, we think that this kind of collaborative activity would be of interest of the members of the IFIP working groups 13.2 (Methodology for User-Centred System Design) and IFIP Working Group 13.4/2.7 (User Interface Engineering).

6.2 Adaptation and extension of models and processes

As discussed along this work, models and processes are central pieces for engineering interactive system. Nonetheless, there is plenty of room for extending and adapting existing models and processes to describe the knowledge required to understand users and the behavior of the interactive systems to be build. This is a never ending activity that purse not only my personal work but the work of all people that are interested in the field of engineering of interactive systems. Nonetheless, as far models are a concern, we plan to pursue our research around the extension of tasks models. Giving the place that task models play in connecting the knowledge about the users with the design of interactive systems, we assume task models are the starting point for improving the knowledge in the field. For the process, we plan to purse the investigation the extensibility of micro-process that could be used with model-based approach to inform the development team how to solve problems with the design of interactive systems.

6.3 Transferability of methods, models and processes

In our research in the fields of Web and Safety Critical Systems we have found a set of modeling methods, micro development processes and evaluation methods that we consider that could be used in other application domains. We have validated these research tools in some projects in the domains of Safety Critical Systems and Web applications development. We suggest that these research tools can also be useful in other domains. However, to prove our hypothesis beyond the domains that we already know, we have to deal with two issues, the replication of studies and identification of opportunities for transferability of methods, models and processes. For the replication, we are planning to follow two main strategies:

- To replicate the methods in our going and in future research projects. This would be the easiest track because we can have the control of the methods and research tools employed. Although the replication would be limited to the application domain for which we have identified suitable sources of funding, we consider a very feasible strategy;
- An alternative solution to the self-replication is to get replication studies from the community. So that people interested in the tools could replicate themselves the studies and report the way they have adapted the research tools. Whilst user studies that involve experimentation are hard to replicate given the variability of factors affecting empirical studies, we suggest that the replication of models and processes could be more easily achieved because we can ground these studies using models. Moreover, in order to get a fair comparison between the research tools we need a reference framework, which has been already proposed in the section 6.1.

For the identification of opportunities for transferability we are planning to explore other application domains in future projects, so that we can extend our knowledge based of application domains.

6.4 Investigate the convergence towards Web technology

Convergence of technology and use of interactive system are blurring the boundaries of research topics that were often associated to a single discipline. Among the new forms of interactive systems that appeared in the last decades, it is noteworthy the increasing importance of Web applications. Currently, many legacy systems are being migrated to the Web and many new information systems are only deployed over the Web. A wide range of new, complex applications is emerging because of the popularity and the ubiquity of the Web itself. Social and mobile Web applications are reshaping the role of users from information consumers to information providers [8] which creates a shift in the relationship between users and interactive systems. We can also observe a phenomenon of convergence/cross-pollination concerning use of Web technologies for developing interactive system, using the Web platform as a means to deploy products and services and or a mean to get in touch with target users. Indeed, the emergence of the Web technology change a lot the way how we develop software nowadays. For that we plan to investigate in which extension the use of Web technology will affect methods, process, and tools for developing interactive systems in general.

References

[1] Abiteboul, S. “Querying Semi-structured data,” in International Conference on Data Base Theory (ICDT), pp. 1 – 18, Delphi, Greece, 1997. See http://dbpubs.stanford.edu:8090/pub/1996-19.

[2] Adomavicius, A., Tuzhilin, A. 2001. Using Data Mining Methods to Build Customer Profiles. Computer 34, 2 (February 2001), 74-82. DOI=10.1109/2.901170 http://dx.doi.org/10.1109/2.901170

[3] Allaire, J. Macromedia Flash MX—A next-generation rich client. White paper, March 2002. Available at: http://download.macromedia.com/pub/flash/whitepapers/richclient.pdf

[4] Aquino, N., Vanderdonckt, J., Panach, J. I., Pastor, O. (2011) Conceptual Modelling of Interaction. In: Handbook of Conceptual Modeling: Theory, Practice, and Research Challenges. Embley, D. W., Thalheim, B. (Eds.). Chapter 1. Springer 2011, XIX, pages 3-15. ISBN 978-3-642-15865-0.

[5] Arellano, C., Díaz, O., Iturrioz, J. Crowdsourced Web Augmentation: A Security Model. In Proc. of WISE 2010. Springer LNCS 6488, pages 294-307.

[6] Albertos Marco, F., Gallud, J., Penichet, V., Winckler. M., A Model-Based Approach for Supporting Offline Interaction with Web Sites Resilient to Interruptions. In Proc. of Current Trends in Web Engineering - ICWE 2013 International Workshops, Aalborg, Denmark, June 25th 2013, Jesper Kjeldskov, Michael Sheng (Eds.), Springer, LNCS 8295, pages 156-171.

[7] Albuquerque, D., Cafeo, B. B. P., Garcia, A. F., Barbosa, S. D. J., Abrahão, S., Ribeiro, A. Quantifying usability of domain-specific languages: An empirical study on software maintenance. Journal of Systems and Software 101: 245-259 (2015)

[8] Anhai Doan, Raghu Ramakrishnan, and Alon Y. Halevy. 2011. Crowdsourcing systems on the World-Wide Web. Commun. ACM 54, 4 (April 2011), 86-96. DOI=10.1145/1924421.1924442

[9] Ardissono, L., Goy, A., Petrone, G., Segnan, M., Torasso, P.: Ubiquitous User Assistance in a Tourist Information Server. In: De Bra, P., Brusilovsky, P., Conejo, R. (eds.) AH2002. LNCS, vol. 2347, pp. 14–23. Springer, Heidelberg (2002).

[10] Ardito, C.; Lanzilotti, R.; Buono, P.; Piccinno, A. A tool to Support Usability Inspection. In Proceedings of ACM Advanced Visual Interfaces (AVI’2006), Venice, Italy, May 23-26, 2006.

[11] Astels, D. Test-Driven Development: A Practical Guide. Prentice Hall. 2003. 592 pages.

[12] Bach, C., Bernhaupt, R., Stein D'Agostini, C., Winckler, M. Mobile Applications for Incident Reporting Systems in Urban Contexts: lessons learned from an empirical study. In Proc. of European Conference on Cognitive Ergonomics (ECCE 2013), Toulouse, France, August 26-28, 2013, pages 29-38, ACM DL.

[13] Barboni, E., Ladry, J., Navarre, D., Palanque, P., Winckler, M. Beyond Modelling: An Integrated Environment Supporting Co-Execution of Tasks and Systems Models. In proceedings of ACM Symposium on Engineering Interactive Systems (EICS'2010), June 19-23, 2010, Berlin, Germany. ACM DL, Sheridan. Pages 165-174.

[14] Barboni, E., Martinie, C., Navarre, D., Palanque, P., Winckler, M. (2013) Bridging the gap between a behavioural formal description technique and a user interface description language: Enhancing ICO with a graphical user interface markup language. In Journal of Science of Computer Programming. Elsevier. Sci. Comput. Program. 86: 3-29.

[15] Barnard, P.J.: Interacting Cognitive Subsystems: A Psycholinguistic Approach to Short-term Memory. Chapter 6 in: Ellis, A. (ed.) Progress in the Psychology of Language Vol. 2., Lawrence Erlbaum, 197–258 (1985).

[16] Baron, M., Lucquiaud, V., Autard, D., Scapin, D. K-MADe : un environement pour le noyau du modèle de description de l'activité, 18ème Conférence Francophone sur l'Interaction Homme-Machine (IHM'2006), edited by ACM Press, Montréal, 2006, pp. 287-288

[17] Bass, L., Pellegrino, R., Reed, S., Seacord, R., Sheppard, R., Szezur, M. R. The Arch model: Seeheim revisited. In: User Interface Developer’s workshop version 1.0, (1991)

[18] Bass, L., Clements, P., Kazman, R. 1998. Software Architecture in Practice. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

[19] Bastide, R., Palanque, P. A Visual and Formal Glue Between Application and Interaction. In: Journal of Visual Language and Computing, Volume 10, Issue 5, October 1999, pages 481–507.

[20] Bastien, C., Scapin, D.L. Evaluating a user interface with ergonomic criteria. Int. J. Hum. Comput. Interaction 7(2): 105-121 (1995)

[21] Beaudouin-Lafon, M., Mackay, W. 2002. Prototyping tools and techniques. In The human-computer interaction handbook, Julie A. Jacko and Andrew Sears (Eds.). L. Erlbaum Associates Inc., Hillsdale, NJ, USA 1006-1031.

[22] Beck, K. (1999). Extreme Programming Explained. Palo Alto, CA: Addison-Wesley.

[23] Beck, Kent et al. (2001). Manifesto for Agile Software Development. Agile Alliance. Retrieved 14 June 2010. Available at: http://www.agilemanifesto.org/

[24] Beck, K. Test Driven Development: By Example. Pearson Education. 2003. 240 pages.

[25] Beizer, B., "Software testing techniques (2nd ed.)," Van Nostrand Reinhold Co., 1990.

[26] Berners-Lee, T., Cailliau, R., Luotonen, A., Nielsen, H. F., and Secret, A. 1994. The World-Wide Web. Commun. ACM 37, 8 (August 1994), 76-82. DOI=10.1145/179606.179671 http://doi.acm.org/10.1145/179606.179671

[27] Bernhaupt, R. User Experience Evaluation in Games and Entertainment, Springer Verlag, 2010.

[28] Bernhaupt, R. Usability and User Experience Evaluation in Non-Traditional Environments. Habilitation à Diriger des Recherche (HDR), Université Paul Sabatier. 2009.

[29] Bernhaupt, B., Navarre, D., Palanque, P., Winckler, M. Model-Based Evaluation: A New Way to Support Usability Evaluation of Multimodal Interactive Applications. In Maturing Usability: Quality in Software, Interaction and Quality. Springer (Human-Computer Interaction Series), October 2007, Law E., Thora Hvannberg E., Cockton G. & Vanderdonckt J. (Eds.). pp. 96-122.

[30] Bernhaupt, R., Weiss, A., Obrist, M., Tscheligi, M. 2007. Playful probing: making probing more fun. In Proceedings of the 11th IFIP TC 13 international conference on Human-computer interaction (INTERACT'07). Springer-Verlag, Berlin, Heidelberg, 606-619.

[31] Berry, M. W., Browne, M. (2005) Understanding Search Engines: Mathematical Modeling and Text Retrieval. Second Edition. SIAM press. DOI: http://dx.doi.org/10.1137/1.9780898718164

[32] Bézivin, J. From Object Composition to Model Transformation with the MDA. In Proceedings of TOOLS'2001. IEEE Press Tools#39, pp. 350–354 . (August 2001).

[33] Bias, R.G., Mayhew, D.J. (Eds.). (1994). Cost-Justifying Usability. San Francisco: Morgan Kaufmann.

[34] Boehm, B. 1986. A spiral model of software development and enhancement. SIGSOFT Softw. Eng. Notes 11, 4 (August 1986), 14-24. DOI=http://dx.doi.org/10.1145/12944.12948

[35] Boring, R.L., Gertman, D.I., Joe, J.C., Marble, J.L. Human reliability analysis in the US Nuclear power industry: A comparison of atomistic and holistic methods. Proceedings of the 49th Annual Meeting of the Human Factors and Ergonomics Society, pp. 1815- 1819.

[36] Brajnik, G. Automatic web usability evaluation: what needs to be done? In Proceedings of 6th Human Factors and the Web Conference, Austin, Texas, June 2000.

[37] Brooke, J. (1996) "SUS: a 'quick and dirty' usability scale". In P. W. Jordan, B. Thomas, B. A. Weerdmeester & A. L. McClelland (eds.) Usability Evaluation in Industry. London: Taylor and Francis.

[38] Brun, P., Beaudouin-Lafon, M. A taxonomy and evaluation of formalisms for the specification of interactive systems. In Proceedings of the HCI'95 conference on People and computers X (HCI '95), M. A. R. Kirby, A. J. Dix, and J. E. Finlay (Eds.). Cambridge University Press, New York, NY, USA, 197-212.

[39] Brusilovsky, P. Adaptive hypermedia. User Modeling and User-Adapted Interaction, 11(1-2):87–110,March 2001.

[40] Cajander, A., Larusdottir, M., Gulliksen, J., Existing but Not Explicit - The User Perspective in Scrum Projects in Practice. In Proceedings of the Human-Computer Interaction – INTERACT 2013. Springer LNCS 8119. pages 762-779, 2013.

[41] Calvary, G., Serna, A., Scapin, D., Winckler, M. (2011) Advanced User Interfaces for e-Government Applications. In: Practical Studies in e-Government (chapter 12). Assar, S., Boughzala, I., Boydens, I. (eds.) Springer. 1st Edition., 2011, X, 240 p. ISBN: 978-1-4419-7532-4.

[42] Calvary, G., Coutaz J., Thevenin, D., Limbourg, Q., Bouillon, L., Vanderdonckt, J. A. Unifying Reference Framework for Multi-Target User Interfaces, Interacting With Computers, Vol. 15/3, pp 289-308, 2003.

[43] Campos, J. C. and M. Harrison, Formally verifying interactive systems: a review, in: Design, Specification and Verification of Interactive Systems '97 (1997), pp. 109–124.

[44] Card, S. K., Moran, T. P., & Newell, A. (1983). The psychology of human-computer interaction. Hillsdale, NJ: Lawrence Erlbaum Associates.

[45] Carr, D. A. 1994. Specification of interface interaction objects. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '94). ACM, New York, NY, USA, 372-378. DOI=10.1145/191666.191793 http://doi.acm.org/10.1145/191666.191793

[46] Carroll, J. M. (2003) Making Use Scenario-Based Design of Human-Computer Interactions. MIT Press, 382 pages.

[47] Cava, R., Freitas, M. D. S., Barboni, E., Palanque, P., Winckler, M. Inside-In Search: an alternative for performing ancillary search tasks on the Web. Latin-American Conference on the Web (LA-WEB 2014). Ouro Preto, Brazil, October 20-23 2014. IEEE CPS.

[48] Céret, E., Dupuy-Chessa, S., Calvary, G., Front, A., Rieu, D. A taxonomy of design methods process models. Information & Software Technology 55(5): 795-821 (2013).

[49] Cetinkaya, O., Cetinkaya, D. (2007) "Verification and Validation Issues in Electronic Voting". The Electronic Journal of e-Government, Vol. 5, No. 2, pp. 117-126.

[50] Chen, P. The Entity-Relationship Model - Toward a Unified View of Data. ACM Trans. Database Syst., 1(1):9–36, 1976.

[51] Cockburn, A., and Jones, S., Which way now? Analysing and easing inadequacies in WWW navigation. International Journal of HumanComputer Studies pp. 105-129, 1996.

[52] Constantine, S. (ed.) (2000) User Interfaces for All: Concepts, Methods and Tools. Mahwah, NJ: Lawrence Erlbaum Associates (ISBN 0-8058-2967-9, 760 pages).

[53] Cooper, A. (1999). The Inmates are Running the Asylum, SAMS: Macmillan Computer Publishing. 228 pages.

[54] Constantine, L., L., Lockwood, L. A. D. (2001). Structure and style in use cases for user interface design. In Object modeling and user interface design, Mark Van Harmelen (Ed.). Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA 245-279.

[55] Cretu, L.G., Dumitriu, F. Model-Driven Engineering of Information Systems: Principles, Techniques, and Practice. CRC Press, September, 2014. 350 pages.

[56] Cugola, G., Ghezzi, C. Software processes: a retrospective and a path to the future. Software Process: Improvement and Practice 4(3): 101-123 (1998).

[57] Coutaz, J., Calvary, G. (2012) HCI and Software Engineering for user Interface Plasticity. Chapter 12 in The Human-Computer Handbook – Fundamentals, Evolving Technologies, and Emerging Applications. Jacko, J. (ed). CRC Press Taylor and Francis Group. pages. 1195-1220. 2012.

[58] Coutaz, J. 2010. User interface plasticity: model driven engineering to the limit! In Proceedings of the 2nd ACM SIGCHI symposium on Engineering interactive computing systems (EICS '10). ACM, New York, NY, USA, 1-8. DOI=10.1145/1822018.1822019 http://doi.acm.org/10.1145/1822018.1822019

[59] Dey, A. 2001. Understanding and Using Context. Personal Ubiquitous Comput. 5, 1 (January 2001), 4-7. DOI=10.1007/s007790170019 http://dx.doi.org/10.1007/s007790170019

[60] Diaper, D. & Stanton, N. A. (eds.) The Handbook of Task Analysis for Human-Computer Interaction. Lawrence Erlbaum Associates, 2004. 650 pages.

[61] Diaz, O., Arellano, C., Iturrioz, J. Layman tuning of websites: facing change resilience. In.: Proc. of WWW2008 Conference, (Beijing, 2008), 127-1128.

[62] Díaz, O., Arellano, C., Aldalur, I., Medina, H., Firmenich, S. End-User Browser-Side Modification of Web Pages. In Proceedings of WISE (Thessalonoki, Greece), pp. 293-307, 2014.

[63] Dix, A., Finlay, J. E., Abowd, G., Beale, R. 2003. Human-Computer Interaction (3rd Edition). Prentice-Hall, Inc., Upper Saddle River, NJ, USA. 834 pages.

[64] Dix, A. J., 1991 Formal Methods for Interactive Systems. s.l: Academic Press. 0-12-218315-0.

[65] Dix, A. Formal Methods: An Introduction to and Overview of the Use of Formal Methods within HCI. (Chapter 2) In perspectives on HCI: Diverse Approaches. Monk, A., Gilbert, N. (eds). London, Academic Press. 1995. pp. 9-43.

[66] Doherty, G., Harrison, M. Integrating Joint Behaviour and Dialogue Description, presented at the 5th International Eurographics Workshop on Design, Specification and Verification of Interactive Systems (DSV-IS '98). Abingdon, U.K. June 2-5 1998. Springer, Eurographics series.

[67] Dragicevic, P., Fekete, J-D. 2004. Support for input adaptability in the ICON toolkit. In Proceedings of the 6th International conference on Multimodal interfaces (ICMI '04). ACM, New York, NY, USA, 212-219. DOI=10.1145/1027933.1027969 http://doi.acm.org/10.1145/1027933.1027969

[68] Engels, G., Heckel, R., Taentzer, G., Ehrig, H. A Combined Reference Model- and View-Based Approach to System Specification. International Journal of Software Engineering and Knowledge Engineering. 11/1997; 7(4). DOI: 10.1142/S0218194097000266

[69] European Union (2012). Article 29 Data Protection Working Party-Opinion 04/2012 on Cookie Consent Exemption. Available at: http://ec.europa.eu/justice/data-protection/article-29/documentation/opinion-recommendation/files/2012/wp194_en.pdf

[70] Fahssi, R., Martinie, C., Palanque, P. Enhanced Task Modelling for Systematic Identification and Explicit Representation of Human Errors. In Proceedings of the 15th IFIP TC 13 International Conference (INTERACT 2015), Bamberg, Germany, September 14–18, 2015. Springer-Verlag LNCS 9299. pp. 192-212.

[71] Firmenich, S. *Un enfoque para soporte de tareas mediante adaptacion en el cliente Web*. PhD thesis presented on April 11th 2013. Universidad de La Plata, La Plata, Argentina. Available at: http://www.irit.fr/~Marco.Winckler/firmenich-these2013.pdf

[72] Firmenich, D., Firmenich, S., Rossi, G., Winckler, M., Distante, D. User Interface Adaptation Using Web Augmentation Techniques: Towards a Negotiated Approach. In Proceedings od the 15th International Conference, ICWE 2015, Rotterdam, The Netherlands, June 23-26, 2015. Springer Lecture Notes in Computer Science 9114, ISBN 978-3-319-19889-7. pp.: 147-164

[73] Firmenich, S., Winckler, M., Rossi, G., Gordillo, S. (2011) A Crowdsourced Approach for Concern-Sensitive Integration of Information across the Web. Journal of Web Engineering (JWE). Rinton Press., Vol.10 No.4, December 2011, pages: 289-315.

[74] Firmenich, S., Winckler, M., Rossi, G., Gordillo, S. A Framework for Concern-Sensitive, Client-Side Adaptation. In Proceedings of International Conference on Web Engineering (ICWE 2011), Paphos, Cyprus, June 20-24, 2011. Springer, LNCS 6757, pages 198-213. Also available at: http://dx.doi.org/10.1007/978-3-642-22233-7_14 (best paper award)

[75] Firmenich, S., Rossi, G., Winckler, M., Palanque, P. An Approach for Supporting Distributed User Interface Orchestration over the Web. In. International Journal of Human-Computer Studies. ISSN 1071-5819. Vol. 72(1): 53-76 (2014). At: http://dx.doi.org/10.1016/j.ijhcs.2013.08.014

[76] Firmenich, S., Rossi, G., Winckler, M. A Domain Specific Language for Orchestrating User Tasks whilst Navigation Web sites. In Proceedings of ICWE 2013, Aalborg, Denmark, July 8-12, 2013. Springer LNCS Vol. 7977, p. 224-232.

[77] Fitts, P. M. (1954). The information capacity of the human motor system in controlling the amplitude of movement. Journal of Experimental Psychology 47 (6): 381–391

[78] Fleming, J. Web navigation: designing the user experience. O'Reilly

[79] Forbrig, P., Martinie, C., Palanque, P., Winckler, M., Fahssi, R. Rapid Task-Models Development Using Sub-models, Sub-routines and Generic Components. In Proceedings of the 5th International Conference on Human-Centered Software Engineering (HCSE 204), Sauer, S. et al (eds.) September 16-18 2014, Paderbon, Germany. Springer LNCS 8742, pp. 144–163, 2014.

[80] Forsberg, K., Mooz, H. The Relationship of System Engineering to the Project Cycle. In Proceedings of the First Annual Symposium of National Council on System Engineering, October 1991: 57–65.

[81] Freitas, C. M. D. S., Luzzardi, P., Cava, R. A., Winckler, M., Pimenta, M. S., Nedel, L. Evaluating Usability of Information VisualizationTechniques. In Proceedings of the V Simpósio sobre Fatores Humanos em Sistemas Computacionais – IHC'2002, Fortaleza, Brazil, 7-10 October 2002.

[82] Fuggetta A., Di Nitto, E. 2014. Software process. In Proceedings of the on Future of Software Engineering (FOSE 2014). ACM, New York, NY, USA, 1-12. DOI=http://dx.doi.org/10.1145/2593882.2593883

[83] Garrett, J. J. The Elements of User Experience: User-Centered Design for the Web. Peachpit Press (October 21, 2002). ISBN-10: 0735712026. 208 pages.

[84] Ghiani, G., Paternò, F., Santoro, C., Spano, L. A Set of Languages for Context-Aware Adaptation. CASFE 2012. CEUR Workshop Proceedings 970, CEUR-WS.org 2012

[85] Göransson B., Gulliksen J. & Boivie I. (2003) The Usability Design Process - Integrating User-Centred Systems Design in the Software Development Process. In Software Process: Improvement and Practice (SPIP), vol. 8, issue 2, Wiley & Sons.

[86] Green, M. Design Notations and User Interface Management Systems. In: User Interface Management Systems Eurographic Seminars 1985, pp 89-107. Springer.

[87] Guerrero, J., Vanderdonckt, J., Gonzalez, J.L., Winckler, M. Modeling User Interfaces to Workflow Information Systems. In 4th International Conference on Autonomic and Autonomous Systems (ICAS'08), Gosier, Guadalupe, March 16-21 2008. pp. 55-60, IEEE International. ISSN/ISBN: 0769530931.

[88] Guerrero, J., Vanderdonckt, J., González, J.M., Winckler, M. Towards a Library of Workflow User Interface Patterns. International Workshop on the Design, Verification and Specification of Interactive Systems (DSVIS'2008). Kingston, Ontario, Canada, July 16-18 2008. Springer LNCS 5136, pp. 96-101.

[89] Gulliksen, J., Göransson, B., Boivie, I., Persson, J., Blomkvist, S. & Cajander, Å. (2003) Key Principles for User Centred Systems Design. International Journal Behaviour & Information Technology, Vol. 22, No. 6. pp.397-409.

[90] Hackos, J. T., Redish, J. C. 1998. *User and Task Analysis for Interface Design*. John Wiley & Sons, Inc., New York, NY, USA. 518 pages.

[91] Halasz, F., Schwartz, M. The Dexter Hypertext Reference Model. Communications of the ACM vol. 37, no. 2, pp. 30-39, 1994.

[92] Hamon, A., Palanque, P., Cronel, M., André, R., Barboni, E., Navarre, D. 2014. Formal modelling of dynamic instantiation of input devices and interaction techniques: application to multi-touch interactions. In Proceedings of the 2014 ACM SIGCHI symposium on Engineering interactive computing systems (EICS'14). ACM, New York, NY, USA, 173-178. DOI=10.1145/2607023.2610286 http://doi.acm.org/10.1145/2607023.2610286

[93] Hamon, A., Palanque, P., André, R., Barboni, E., Cronel, M., & Navarre, D. Multi-Touch Interactions for Control and Display in Interactive Cockpits. HCI'Aero 2014, ACM DL.

[94] Han, J., Haihong, E., Guan Le, Jian Du. Survey on NoSQL database. 6th International Conference on Pervasive Computing and Applications (ICPCA), 2011. Port Elisabeth, IEEE. DOI: 10.1109/ICPCA.2011.6106531

[95] Han, H., Tokuda, T. Towards flexible and lightweight integration of web applications by end-user programming. IJWIS 6(4): 359-373 (2010).

[96] Harel, D. Statecharts: a visual formalism for complex systems. Science of computer Programming, vol. 8; 1987.

[97] Harrison, M., Thimbleby, H. (eds). Formal Methods in Human-Computer Interaction. Cambridge Press. 1990. 344 pages. ISBN-10: 052137202X.

[98] Hart, S. G. & Staveland, L. E. (1988) Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research. In P. A. Hancock and N. Meshkati (Eds.) Human Mental Workload. Amsterdam: North Holland Press.

[99] Harper, B. D. & Norman, K. L. (1993). Improving User Satisfaction: The Questionnaire for User Interaction Satisfaction Version 5.5. Proceedings of the 1st Annual Mid-Atlantic Human Factors Conference, (pp. 224-228), Virginia Beach, VA.

[100] Hassenzahl, M. (2007). AttrakDiff(tm). Available at: http://www.attrakdiff.de.

[101] Hassenzahl, M. 2004. The interplay of beauty, goodness, and usability in interactive products. Human-Computer Interaction 19, 4 (Dec. 2004), 319-349.

[102] Hartson HR, Hix D. Toward empirically derived methodologies and tools for human-computer interface development. Int J Man Mach Stud 1989;31(4):477-494.

[103] Herbsleb, J. D., Moitra, D. Global software development. IEEE Software, vol. 18, no. 2, pp. 16–20, 2001.

[104] Hix, D. and Hartson, H. R., Developing User Interfaces: Ensuring Usability through Product & Process, New York: John Wiley and Sons, Inc. 1993.

[105] Hoste, L., Dumas, B., & Signer, B. (2011). Mudra: a unified multimodal interaction framework. ICMI '11 (pp. 97-104). ACM.

[106] International Atomic Energy Agency, Experience in the use of Systematic Approach to Training (SAT) for Nuclear Power Plant Personnel, Technical Report, IAEA-TECDOC-1057.

[107] International Electrotechnical Commission. IEC 61511: Functional safety - Safety instrumented systems for the process industry sector. Beuth Verlag GMbH, Berlin, 2003.

[108] ISO 9241-11 (1998) Ergonomic requirements for office work with visual display terminals (VDTs) Part 11: Guidance on Usability. This standard has been revised by the ISO 9241-171:2008 Ergonomics of human-system interaction -- Part 171: Guidance on software accessibility.

[109] ISO 13407 (1999). Human-centred design process for interactive systems. This standard has been revised by: ISO 9241-210:2010

[110] ISO 8879 (1986). Information processing - Text and office systems - Standard Generalized Markup Language (SGML).

[111] Ivory, M. Y. Automated Web Site Evaluation: Researchers' and Practitioners' Perspectives. Kluwer Academic Publishers. Dordrecht, The Netherlands. 2003. 200p.

[112] Jackson, M. Software Requirements and Specifications: A Lexicon of Practice, Principles and Prejudices. Addison-Wesley Professional; 1 edition (September 2, 1995) 228 pages.

[113] Jacobson, I., Christerson, M., Jonsson, P., and Övergaard, G. (1992) Object-Oriented Software Engineering: A Use Case Driven Approach. Reading, MA: Addison-Wesley.

[114] Javaux, D., Colard, M.-I., Vanderdonckt, J. Visual Display Design: a Comparison of Two Methodologies, in Proc. of 1st Int. Conf. on Applied Ergonomics ICAE'96 (Istanbul, 21-24 May 1996), A.F. Özok & G. Salvendy (eds.), Istanbul - West Lafayette, 1996, pp. 662-667.

[115] John, B. E., Kieras, D. E. The GOMS Family of User Interface Analysis Techniques: Comparison and Contrast. ACM Trans. Computer-Human Interaction 3(4): 320-351 (1996).

[116] Johnson, P. Human computer interaction: psychology, task analysis and software engineering, Berkshire, UK: McGRAW-HILL Book Company Europe, 1992.

[117] Johnson, C.: On the over emphasis of human error as a cause of aviation accidents: systemic failures and human error in US NTSB and Canadian TSB aviation reports 1996–2003, Ergonomics (2006).

[118] Johnson, C. The Natural History of Bugs: Using Formal Methods to Analyse Software Related Failures in Space Missions. In proceeedings of Formal Methods, International Symposium of Formal Methods Europe (FM 2005), Newcastle, UK, July 18-22, 2005. Springer LNCS 3582, pages 9-25.

[119] Kleppe, A. G., Warmer, J., Bast, W. 2003. MDA Explained: The Model Driven Architecture: Practice and Promise. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

[120] Kieffer, S., Coyette, A., Vanderdonckt, J. 2010. User interface design by sketching: a complexity analysis of widget representations. In Proceedings of the 2nd ACM SIGCHI symposium on Engineering interactive computing systems (EICS '10). ACM, New York, NY, USA, 57-66.

[121] Kobsa, A. 2001. Generic User Modeling Systems. User Modeling and User-Adapted Interaction 11, 1-2 (March 2001), 49-63. DOI=10.1023/A:1011187500863 http://dx.doi.org/10.1023/A:1011187500863

[122] Kruchten, P. The Rational Unified Process: an introduction. (2nd edition) Addison-Wesley, Boston. 2004.

[123] Krulwich, B. Intelligent User Profiling. Using Large-Scale Demographic Data. AI Magazine Volume 18 Number 2 (1997).

[124] Kuniavsky, M., Raghavan, S. Guidelines are a tool: building a design knowledge management system for programmers. In *Proceedings of the 2005 conference on Designing for User eXperience* (DUX '05). AIGA: American Institute of Graphic Arts, New York, NY, USA.

[125] Lam, S.K., Riedl, J. The past, present, and future of Wikipedia. Computer, Vol. 44 , N. 3, March 2011, pages 87 – 90.

[126] Lazar, J. (Editor) Universal Usability: Designing Computer Interfaces for Diverse User Populations. Wiley, April 2007. 628 pages.

[127] Lazar, J. Public policy and HCI in the U.S. context. Interactions 18(3): 36-40 (2011).

[128] Lazar, J., Feng, J. H., Hochheiser, H. 2010. Research Methods in Human-Computer Interaction. Wiley Publishing. 448 pages.

[129] Law, E., Scapin, D., Cockton, G., Springett, M., Stary, C. & Winckler, M. (2009) Maturation of Usability Evaluation Methods: Retrospect and Prospect, Final Project Report, ISBN: 978-2-917490-06-8.

[130] Law, E., Cockton, G., Stary, C., Neubauer, M., Hvannberg, E., Vermeeren, A. (2012) COST IC0904 TwinTide Working Groups Final Report. Copenhagen, Denmark - October 15th, 2012. 150 pages.

[131] Limbourg, Q., Pribeanu, C., Vanderdonckt, J: Towards Uniformed Task Models in a Model-Based Approach. DSV-IS 2001: 164-182.

[132] Limbourg, Q., Vanderdonckt, J., Michotte, B., Bouillon, L., López-Jaquero, V. USIXML: A Language Supporting Multi-path Development of User Interfaces. EHCI/DS-VIS 2004: 200-220.

[133] Lu, S., Paris, C., Vander Linder, K. Towards the Automatic Construction of Task Models from Object-Oriented Diagrams. In Proceedings of 7th Working Conference on Engineering Human-Computer Interaction. September 14-18, 1998, Heraklion, Crete, Greece. Kluwer Academic Publishers. Pp. 169-190.

[134] Luyten, K., Clerckx, T., Coninx, K., Vanderdonck, J. Derivation of a Dialog Model from a Task Model by Activity Chain Extraction. In proc. of DSV-IS 2003, LNCS 2844, pp. 203–217, 2003. Springer-Verlag Berlin Heidelberg 2003.

[135] Maggi, F.M., Mooij, A.J., van der Aalst, W.M.P. User-Guided Discovery of Declarative Process Models. In N. Chawla, I. King, and A. Sperduti, editors, IEEE Symposium on Computational Intelligence and Data Mining (CIDM 2011), pages 192-199, Paris, France, April 2011. IEEE.

[136] Martin, J. (1991). Rapid Application Development. Macmillan. ISBN 0-02-376775-8.

[137] Martinie, C. (2011) A Synergistic Models-Based Approach to Develop Usable, Reliable and Operable Interactive Critical Systems. PhD thesis presented on December 25th 2011. Université Paul Sabatier. Available at: http://www.irit.fr/~Marco.Winckler/martinie-these2011.pdf

[138] Martinie, C., Barboni, E., Navarre, D., Palanque, P., Fahssi, R., Poupart, E., Cubero-Castan, E. Multi-models-based engineering of collaborative systems: application to collision avoidance operations for spacecraft. In Proceedings of the 2014 ACM SIGCHI symposium on Engineering interactive computing systems (EICS '14). ACM, New York, NY, USA, 85-94. DOI=10.1145/2607023.2607031 http://doi.acm.org/10.1145/2607023.2607031

[139] Martinie, C., Navarre, D., Winckler, M. A formal approach supporting effective and efficient training program for improving operators' reliability (regular paper). Dans : Safety and Reliability for managing Risk (ESREL), Rhodes Grece, September 05-09, 2010, Taylor & Francis Group, p. 234-243, 2010.

[140] Martinie, C., Palanque, P., Barboni, E., Winckler, M., Ragosta, M., Pasquini, A., Lanzi, P. Formal Tasks and Systems Models as a Tool for Specifying and Assessing Automation Designs. International Conference on Application and Theory of Automation in Command and Control Systems 2011. IRIT Press, pp. 50-59.

[141] Martinie, C., Palanque, P., Winckler, M., Navarre, D., Poupart, E. Model-Based Training: An Approach Supporting Operability of Critical Interactive Systems: Application to Satellite Ground Segments. In proceedings of ACM SIGCHI conference Engineering Interactive Computing Systems (EICS 2011), Pisa, Italy, June 13-16 2011. ACM DL, Sheridan Press, pages 53-62.

[142] Martinie, C., Palanque, P., Navarre, D., Barboni, E .A Development Process for Usable Large Scale Interactive Critical Systems: Application to Satellite Ground Segments. Human-Centered Software Engineering 2012 (HCSE). LNCS7623. p:112-134. Springer.

[143] Martinie, C., Palanque, P., Ragosta, M., Fahssi, R. 2013. Extending procedural task models by systematic explicit integration of objects, knowledge and information. In Proceedings of the 31st European Conference on Cognitive Ergonomics (ECCE'13). August 26-28, 2013, Toulouse, France. ACM, New York, NY, USA, Article 23, 10 pages. DOI=10.1145/2501907.2501954 http://doi.acm.org/10.1145/2501907.2501954.

[144] Martinie, C., Palanque, P., Winckler, M. Structuring and Composition Mechanism to Address Scalability Issues in Task Models. IFIP TC13 Conference on Human-Computer Interaction (INTERACT 2011), September 05-09 2011, Lisbon, Portugal, Springer LNCS 6949, pages 593-611.

[145] Matera, M.; Costabile, M. F.; Garzotto, F.; Paolini, P. SUE inspection: an effective method for systematic usability evaluation of hypermedia. IEEE Transactions on System, Man and Cybernetics. v. 32 (1), January 2002.

[146] Masip, L., Martinie, C., Winckler, M., Palanque, P., Granollers, T. Oliva, M. A Design Process for Exhibiting Design Choices and Trade-offs in (potentially) Conflicting User Interface Guidelines. In proceedings of the 4th Human-Centered Software Engineering (HCSE 2012), Toulouse, France, October 29-31, 2012. Springer, LNCS 7623, p. 53-71.

[147] Mbaki, E., Vanderdonckt, J., Winckler, M. Multi-level Dialog Modelling in Highly Interactive Web Interfaces. In 7th Int. Workshop on Web-Oriented Software Technologies (IWWOST'2008). New York, USA. July 14, 2008. In conjunction with ICWE'2008. ISBN 978-80-227-2899-7. pages 44-49.

[148] McDermid, J., & Ripken, K. (1983). Life cycle support in the Ada environment. ACM SIGAda Ada Letters, III(1).

[149] Meiselwitz, G., Wentz, B., Lazar, J. Universal Usability: Past, Present, and Future. Foundations and Trends in Human-Computer Interaction 3(4): 213-333 (2010).

[150] Meyer, B. Seven Principles of Software Testing. Computer, vol. 41, no. 8, pp. 99–101, Aug. 2008, doi:10.1109/MC.2008.306

[151] Mori, G., Paternò, F., Santoro, C. Design and Development of Multidevice User Interfaces through Multiple Logical Descriptions. IEEE Trans. Software Eng. 30(8): 507-520 (2004)

[152] Morville, P., Rosenfeld, L. Information Architecture for the World Wide Web: Designing Large-Scale Web Sites, 3rd Edition Paperback – December 7, 2006

[153] Mulder, S., Yaar, I. (2006). The User Is Always Right: A Practical Guide to Creating and Using Personas for the Web. New Riderds, 312 pages.

[154] Muller, P-A, Studer, P., Fondement, F., Bézivin, J. Platform independent Web application modeling and development with Netsilon. Software and System Modeling 4(4): 424-442 (2005).

[155] Murugesan, S. and Deshpande, Y. (2001) Web Engineering: managing diversity and complexity of web application development. Springer, 385 pages.

[156] Navarre, D.; Palanque, P.; Bastide, R.; Paternó, F., and Santoro, C. A tool suite for integrating task and system models through scenarios. In DSV-IS'2001 (Glasgow, Scotland, June 13-15, 2001). LNCS 2220. Springer; 2001

[157] Navarre, D., Palanque, P., Bastide, R., Schyn, A., Winckler, M., Nedel, L., Freitas, C. M. D. S. A Formal Description of Multimodal Interaction Techniques for Immersive Virtual Reality Applications. INTERACT 2005 proceedings. Springer 2005 Lecture Notes in Computer Science ISBN 3-540-28943-7 pages 170-183.

[158] Navarre, D., Palanque, P., Ladry, J-F., Barboni, E. (2009) ICOs: A model-based user interface description technique dedicated to interactive systems addressing usability, reliability and scalability. ACM Trans. Computer-Human Interaction. 16, 4, Article 18 (November 2009), 56 pages.

[159] Navarre, D., Palanque, P., Winckler, M. (2009) Task Models and System Models as a Bridge between HCI and Sofware Engineering. In: Human-Centered Software Engineering Software Engineering Models, Patterns and Architectures for HCI. (chapter 17). Seffah, A., Vanderdonckt, J., Desmarais, M. (eds.) Springer (Human-Computer Interaction Series), June 2009, pp. 357-385.

[160] Ng, A., Lepinski, J., Wigdor, D., Sanders, S., & Dietz, P. (2012). Designing for low-latency direct-touch input. 25th ACM UIST conference pp. 453-464. ACM.

[161] Nielsen, J. Usability Engineering. Morgan Kaufmann, 1993, 362 pages.

[162] Nielsen, J. Mack, R. Usability Inspection Methods. Wiley, 1994. 448 p.

[163] Norman, D. A., Draper, S. W. (eds.) (1986): User Centered System Design: New Perspectives on Human-Computer Interaction. Hillsdale, NJ, Lawrence Erlbaum Associates.

[164] Norman, D. The Psychology Of Everyday Things. (1988). Basic Books; 1 edition (June 13, 1988). ISBN 978-0-465-06710-7

[165] Novak, J. (2008) Game Development Essentials: An introduction. Delmar Cengage Learning. ISBN-10: 1111307652 512 pages.

[166] Object Management Group, Inc. (OMG) Unified Modeling Language (UML) Resource Page. Available at http://www.uml.org/

[167] Object Management Group, Inc. (OMG) IFML: The Interaction Flow Modeling Language. Available at http://www.ifml.org/

[168] Palanque, P., Barboni, E., Martinie, C., Navarre, D., Winckler, M. A Tool Supported Model-based Approach for Engineering Usability Evaluation of Interaction Techniques. In proceedings of ACM SIGCHI conference Engineering Interactive Computing Systems (EICS 2011), Pisa, Italy, June 13-16 2011. ACM DL, Sheridan Press, pages 21-30.

[169] Palanque, P., Bastide, R., Winckler, M. Automatic Generation of Interactive Systems: Why A Task Model is not Enough. 10th International Conference on Human-Computer Interaction – HCI International'2003, Heraklion, Greece, June 2003.

[170] Palanque, P., Bernhaupt, R., Winckler, M. Usability Evaluation: Commonalities and Discrepancies in Games and Safety Critical Systems. In Proc. of the 1st European Workshop on HCI Design and Evaluation: The influence of domain. Limassol, Cyprus - April 8th, 2011. Eds. G. Christou, P. Zaphiris, & E. Law. IRIT Press, Toulouse, France. ISBN: 978-2-917490-13-6 EAN: 9782917490136. pp. 37-44.

[171] Palanque, P., Farenc, C., and Bastide, R. 1999. Embedding ergonomic rules as generic requirements in the development process of interactive software. In Proceedings of IFIP TC13 Seventh International Conference on Human-Computer Interaction (Edinburgh, Scotland, August 1999). Amsterdam, The Netherlands: IOS Press.

[172] Palanque, P., Paterno, F. (Eds.). Formal Approaches to Computing and Information Technology. In the Series: Formal Approaches to Computing and Information Technology (FACIT). 1998. Springer. 376 pages. ISBN-10: 3540761586.

[173] Palanque, P., Winckler, M., Ladry, J.F., ter Beek, M. H., Faconti, G. P., Massink, M. A Formal Approach Supporting the Comparative Predictive Assessment of the Interruption-Tolerance of Interactive Systems In Proceedings of the ACM Symposium on Engineering Interactive Computing Systems (EICS 2009), Pittsburgh, USA, ACM, New York. pp. 211-220.

[174] Palanque, P., Winckler, M., Martinie, C. (2011) A Formal Model-Based Approach for Designing Interruptions-Tolerant Advanced User Interfaces. In : Model-Driven Development of Advanced User Interfaces . Hussmann, H., Meixner, G. & Zuehlke, D. (Eds.). Springer, Studies in Computational Intelligence Series, Vol. 1061. pp. 141-170, 304p., ISBN: 978-3-642-14561-2.

[175] Panigrahy, R., Najork, M., Xie, Y. 2012. How user behavior is related to social affinity. In Proceedings of the fifth ACM international conference on Web search and data mining (WSDM '12). ACM, New York, NY, USA, 713-722. DOI=10.1145/2124295.2124379 http://doi.acm.org/10.1145/2124295.2124379

[176] Parnas, D. L. On the use of transition diagrams in the design of a user interface for an interactive computer system. In Proceedings of the 1969 24th national conference (ACM '69). ACM, New York, NY, USA, 379-385. DOI=10.1145/800195.805945 http://doi.acm.org/10.1145/800195.805945.

[177] Parnas, D. L., Clements, P. C. 1986. A rational design process: How and why to fake it. IEEE Transactions on Software Engieering. 12, 2 (February 1986), 251-257.

[178] Fabio Paternò: Towards a UML for Interactive Systems. In Proceedings of EHCI 2001, Springer LNCS 2254, pp. 7-18

[179] Paternò, F., Mancini, C. and Meniconi, S. ConcurTaskTrees: A Diagrammatic Notation for Specifying Task Models. In Proceeding of Interact'97. Chapman & Hall (1997), 362-369.

[180] Paternò, F., Mori, G., Galiberti, R. 2001. CTTE: an environment for analysis and development of task models of cooperative applications. In *CHI '01 Extended Abstracts on Human Factors in Computing Systems* (CHI EA '01). ACM, New York, NY, USA, 21-22. DOI=10.1145/634067.634084 http://doi.acm.org/10.1145/634067.634084

[181] Paternò, F., Piruzza, A., Santoro, C. Remote Web Usability Evaluation Exploiting Multimodal Information On User Behavior. CADUI 2006: 287-298.

[182] Pariser, E. (2012) The Filter Bubble: How the New Personalized Web Is Changing What We Read and How We Think. Penguin Books, 304 pages.

[183] Perseil, I., Pautet, L. A Co-Modeling Methodology Designed for RT Architecture Models Integration. In 12th International Conference on Engineering of Complex Computer Systems, ICECCS, pages 371–376, Auckland, New Zealand, July 2007. IEEE Computer Society.

[184] Pierret, C., Delouis, I. & Scapin, D. L. (1989). Un outil d'acquisition et de representation des tâches orienté-objets (Rapport de recherche N° 1063), INRIA.

[185] Pinheiro da Silva, P., Paton, N. W. User Interface Modeling in UMLi. IEEE Software 20(4): 62-69 (2003)

[186] Pontico, F. (2008) Une méthode de conception basée sur des patrons d'interface pour les applications d'e-Gouvernement. PhD thesis presented on July 11th 2008. Université Paul Sabatier. Available at: http://www.irit.fr/~Marco.Winckler/pontico-these2008.pdf

[187] Pontico, F. "A pattern-based user interface design method for e-Government applications". University of Toulouse, France. Presented on July 11th 2008.

[188] Pontico, F., Farenc, C., Winckler, M. Model-based support for specifying eService eGovernment applications. In Proceedings of the 5th international conference on Task models and diagrams for users interface design (TAMODIA'06). Springer LNCS 4385, Berlin, Heidelberg, 54-67.

[189] Pontico, F., Winckler, M., Limbourg, Q. Towards a Universal Catalogue of User Interface Patterns for eGovernment Web sites. Sixth International EGOV Conference 2007, Regensburg (Germany), September 3 - 7, 2007.

[190] Pontico, F., Winckler, M., Limbourg, M. Organizing user interface patterns for e-Government applications. In Proceedings of the Engineering Interactive Systems (EIS), Gulliksen, J., Borup, M., Harning, Palanque, P., Veer, G., Wesson, J. (Eds.). Lecture Notes In Computer Science, Vol. 4940. Springer-Verlag, Berlin, Heidelberg 601-619. DOI=10.1007/978-3-540-92698-6_36.

[191] Radio Technical Commission for Aeronautics (RTCA) and European Organisation for Civil Aviation Equipment (EUROCAE). (2012) DO-178C, Software Considerations in Airborne Systems and Equipment Certification

[192] Raggett, D., Lam, J., and Alexander, I. 1996. HTML 3: Electronic Publishing on the World Wide Web. Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA.

[193] Raggett, D. 2009. The Web of Things: Extending the Web into the Real World. In Proceedings of the 36th Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM '10). Springer-Verlag, Berlin, Heidelberg, 96-107. DOI=10.1007/978-3-642-11266-9_8

[194] Rasmussen, J. 1987. Skills, rules, and knowledge; signals, signs, and symbols, and other distinctions in human performance models. In System design for human interaction, A. P. Sage (Ed.). IEEE Press, Piscataway, NJ, USA 291-300.

[195] Razali, R. Understanding the Efficacy of Graphical Formal Methods-Empirical Assessments. World Applied Sciences Journal 31 (5): 889-903, 2014. ISSN 1818-4952. DOI: 10.5829/idosi.wasj.2014.31.05.1976

[196] Reason, J. (1990) Human Error. Cambridge University Press, 320 pages.

[197] Ricca, F. and P. Tonella, Analysis and testing of web applications, in: ICSE '01: Proceedings of the 23rd International Conference on Software Engineering (2001), pp. 25–34.

[198] Robles Luna, E., Garrigos, I., Grigera, J., Winckler, M. Capture and Evolution of Web requirements using WebSpec. In proceedings on International Conference on Web Engineering (ICWE'2010), July 5 - 9, 2010 in Vienna, Austria.

[199] Rodrigues da Silva, A. Model-driven engineering: A survey supported by the unified conceptual model. In Computer Languages, Systems & Structures. Volume 43, October 2015, Pages 139–155.

[200] Rogers, E. M. (2003) Diffusion of Innovations. Editor S & S International. 576 pages.

[201] Rosson, M. B., Carroll, J. M. 2001. *Usability Engineering: Scenario-Based Development of Human-Computer Interaction*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[202] Rouse, W.B. Models of human problem solving: Detection, diagnosis and compensation for system failures. Preprint for Proceedings of IFAC Conference on Analysis, Design and Evaluation of Man-Machine Systems. Baden-Baden, FRG., September, 1981.

[203] Royce, W. W. Managing the development of large software systems: concepts and techniques. In Proceedings of the 9th international conference on Software Engineering (ICSE '87). IEEE Computer Society Press, Los Alamitos, CA, USA, 328-338. Reprinted from proceedings of IEEE WESCON, August 1970, pages 1-9.

[204] Rubin, J. Handbook of Usability Testing: How to Plan, Design and Conduct Effective Tests. John Wiley & Sons. New York. 1994. 330 p.

[205] Sahila, A. G., Latha, P. Privacy Preserving and Fully Anonymous Protocols for Profile Matching in Mobile Social Networks. International Journal of Computer Science and Mobile Applications (IJCSMA), Vol.2 Issue. 2, February, 2014, pg. 106-110 ISSN: 2321-8363.

[206] Salminen, D., Tallberg, A. Human errors in fatal ad serious occupational accidents in Finland. Ergonomics Vol. 39 N. 7 (1996à, pp. 980-988.

[207] Sangiorgi, U. B., Vanderdonckt, J., GAMBIT: Addressing multi-platform collaborative sketching with html5. EICS 2012: 257-262.

[208] Scapin, D., Leulier, C., Vanderdonckt, J., Bastien, C., Farenc, C., Palanque, P., Bastide, R. Towards automated testing of web usability guidelines. 6th Internaitonal Conference on human factors & the web; Austin, Texas, USA. 2000.

[209] Scapin, D. L., Bastien, C., Garrigues, S., Leulier, C., Conception ergonomique d'interfaces web: démarche et outil logiciel de guidage et de support, INRIA Technical Report of EvalWeb project, INRIA-Université de Toulouse 1-UCL, Rocquencourt-Toulouse-Louvain, December 1999.

[210] Scapin, D. L., Marie-Dessoude, P., Winckler, M., Detraux, C. Personal Information Systems: User Views and Information Categorization. In Proc. of the International Conference on Advances in Human-oriented and Personalized Mechanisms, Technologies, and Services (CENTRIC'2011), Barcelona, October 28-29, 2011, IARIA, pp.40-47. ISBN: 978-1-61208-167-0.

[211] Schaefer, R. A Survey on Transformation Tools for Model Based User Interface Development. In: Human-Computer Interaction. Interaction Design and Usability. Springer LNCS 4550, 2007, pp 1178-1187.

[212] Schiaffino, A., Amandi, A. 2009. Intelligent user profiling. In Artificial intelligence, Max Bramer (Ed.). Lecture Notes In Computer Science, Vol. 5640. Springer-Verlag, Berlin, Heidelberg 193-216.

[213] Schmidt, D. C. Model-Driven Engineering. IEEE Computer, Vol. 39, N. 2, 2006, Page(s):25 - 31

[214] Scholtz, J., Laskowski, Sh. Developing Usability Tools and Techniques for Designing and Testing Web Sites. In Proceedings of the 4th Conference on Human Factors and the Web. Basking Ridge, USA, June 1998.

[215] Schwaber, K., Beedle, M. (2001) Agile Software Development with Scrum (Series in Agile Software Development). Pearson 1st Edition. 158 pages. ISBN-10: 0130676349.

[216] Seffah, A., Vanderdonckt, J., Desmarais, M. C. (Eds.). Human-Centered Software Engineering Software Engineering Models, Patterns and Architectures for HCI. Springer-Verlag London. 2009. 397 pages. DOI: 10.1007/978-1-84800-907-3

[217] Seffah, A. Gulliksen, J., Desmarais, M. C. (Eds.) Human-Centered Software Engineering: Integrating Usability in the Development Process. Springer Netherlands. 2005. 391 pages. DOI: 10.1007/1-4020-4113-6.

[218] Sharma, R., Pavlovi, V. I., Huang, T. S. Toward Multimodal Human–Computer Interface. Proceedings of the IEEE, Vol. 86, N. 5, May 1998, pages 853-869.

[219] Sinnig, D., Chalin, P., Khendek, F. (2007). Common semantics for use cases and task models. In Proceedings of the 6th international conference on Integrated formal methods (IFM'07), Jim Davies and Jeremy Gibbons (Eds.). Springer-Verlag, Berlin, Heidelberg, 579-598.

[220] Stanciulescu, A., Limbourg, Q., Vanderdonckt, J., Michotte, B., Montero, F. 2005. A transformational approach for multimodal web user interfaces based on UsiXML. In *Proceedings of the 7th international conference on Multimodal interfaces* (ICMI '05). ACM, New York, NY, USA, 259-266. DOI=10.1145/1088463.1088508 http://doi.acm.org/10.1145/1088463.1088508

[221] State Chart XML (SCXML): State Machine Notation for Control Abstraction. W3C Working Draft 21 February 2007, available at: http://www.w3.org/TR/scxml/

[222] Stermsek, G., Strembeck, M., Neumann, G. A User Profile Derivation Approach based on Log-File Analysis. IKE, page 258-264. CSREA Press, (2007).

[223] Storey, N. Safety critical computer systems. Addison-Wesley (1996), 472 pages. ISBN-10: 0201427877.

[224] Sutcliffe, A. Symbiosis and synergy? Scenarios, task analysis and reuse of HCI knowledge. Interacting with Computers 15 (2003) 245–263.

[225] SysML Forum. Available at: http://sysmlforum.com/sysml-specifications/

[226] Tamburri, D., Lago, P., Van Vliet, H. Organizational social structures for software engineering. ACM computing Surveys, Vol. 46(1), 2014.

[227] Teo, L., John, B. E. John, and Blackmon, M. H. (2012). CogTool-Explorer: A model of goal-directed user exploration that considers information layout. Proceedings of CHI 2012 (Austin, TX, May 5-10, 2012). ACM, New York.

[228] ter Beek, M. H., Faconti, G. P., Massink, M., Palanque, P., Winckler, M. Resilience of Interaction Techniques to Interrupts: A Formal Model-based Approach In Proceedings of the 12th IFIP TC 13

Conference on Human-Computer Interaction (INTERACT 2009). T. Gross et al. (Eds.), Part I, LNCS 5726, pp. 494–509, Uppsala, Sweden, 24-28 August 2009.

[229] U.S. Department of Defense Training Document (1975). Pamphlet 350-30. August, 1975.

[230] Van den Bergh, J., Coninx, K. From Task to Dialog Model in the UML. In proc. of Task, Models and DIAgrams (TAMODIA 2007), volume 4849 of Lecture Notes in Computer Science, page 98-111. Springer, (2007).

[231] Van Duyne, D. K., Landay, J. A., and Hong, J. A., 2002. The Design of Sites: Patterns, Principles, and Processes for Crafting a Customer-Centered Web Experience. Reading, MA: Addison-Wesley, 2002.

[232] Vanderdonckt, J.; Farenc, C. Tools for Working with Guidelines: Annual Meeting of the Special Interest Group (TFWWG'2000). Springer publishers, London, UK. 2001.

[233] Vanderdonckt, J. Development milestones towards a tool for working with guidelines. Interacting with Computers 12(2): 81-118 (1999).

[234] van Duyne, D. K.; Landay, J. A.; Hong, J. I. The Design of Sites: Patterns, Principles and Processes for Crafting a Costumer-centered Web experience. Addison-Wesley, Boston, USA. 762 pages.

[235] Venkatesh, V., Morris, M.G., Davis, G. B., Davis, F. D. (2003), "User acceptance of information technology: toward a unified view", MIS Quarterly 27 (2003) (3), pp. 425–478.

[236] Vogt, T. Difficulties in Using Style Guides for Designing User Interfaces. Tools for Working with Guidelines. J. Vanderdonckt, C. Farenc (Eds.), Springer-Verlag, London, pp. 197-208.(2001).

[237] W3C (2008). Web Content Accessibility Guidelines 2.0. W3C Candidate Recommendation April 2008. Available at http://www.w3.org/TR/WCAG20/

[238] W3C (1998). Extensible Markup Language (XML) 1.0. Available at: http://www.w3.org/XML/

[239] W3C (2014). HTML 5. W3C Recommendation 28 October 2014. At: http://www.w3.org/TR/html/

[240] Wang, W-L., Hemminger, T. L., and Tang, M. H. 2007. A user-oriented reliability modelling approach for web systems. Int. J. Web Eng. Technol. 3, 3 (January 2007), 288-306. DOI=10.1504/IJWET.2007.012058 http://dx.doi.org/10.1504/IJWET.2007.012058

[241] Weiser, M. The world is not a desktop. Interactions, ACM, USA; January 1994; pp. 7-8.

[242] Whitehead, J. 2007. Collaboration in Software Engineering: A Roadmap. In 2007 Future of Software Engineering (FOSE '07). IEEE Computer Society, Washington, DC, USA, 214-225. DOI=http://dx.doi.org/10.1109/FOSE.2007.4

[243] Winckler, M. StateWebCharts: a Formal Notation for Navigation Modelling of Web Applications. (PhD thesis) Université Toulouse 1, Toulouse, France, April 2nd, 2004.

[244] Winckler, M. (2010). *L'Administration Électronique*: The French Approach to E-Government. Interactions, Vol. XVII, n. 6, November+December 2010, p. 52-55.

[245] Winckler, M., Bach, C., Bernhaupt, R. Identifying User Experience Dimensions for Mobile Incident Reporting in Urban Contexts. In IEEE Transactions on Professional Communication. Volume 56, Issue 2, June 2013, pp. 97-119. [doi: http://dx.doi.org/10.1109/TPC.2013.2257212]

[246] Winckler, M., Barboni, E., Farenc, C., Palanque, P. SWCEditor: a Model-Based Tool for Interactive Modelling of Web Navigation. International Conference on Computer-Aided Design of User Interface - CADUI'2004, Funchal, Portugal, 13-16 January 2004.

[247] Winckler, M., Barboni, E., Palanque, P., Farenc, C. (2006). What Kind of Verification of Formal Navigation Modelling for Reliable and Usable Web Applications? Electronic Notes Theoretical Computer Science 157(2). pages: 207-211.

[248] Winckler, M., Bernhaupt, R., Palanque, P., Lundin, D., Leach, K., Ryan, P., Alberdi, E., Strigini, L. Assessing the Usability of Open Verifiable e-Voting Systems. In Proceedings of the 1st International Conference on eGovernment and eGovernnace (ICEGOV). 12-13 March 2009, Ankara, Turkey. Kaplan, A., Balci, A., Aktan, C. C., Dalbay, O. (Eds.). Turksat. Vol. 1. pages 281- 296. ISBN 975-6339-00-0.

[249] Winckler, M., Bernhaupt, R., Pontico, F. (2010) Challenges for the Development of User Interface Pattern Languages: a case study on the e-Government domain. IADIS International Journal on WWW/Internet. Vol. 8, N. 2. Pages 59-84.

[250] Winckler, M., Farenc, C., Palanque, P. Automatic Evaluation for the Web: How Improve Navigation Guidelines. Workshop on Automated Testing in ACM Conference on Computer-Human Interaction – CHI'2002, Minneapolis, USA, April 21-22, 2002.

[251] Winckler, M., Freitas, C. M. D. S., Palanque, P. Tasks and Scenario-based Evaluation of Information Visualization Techniques. In Proceedings of the 3rd International Workshop on Task Models and Diagrams for User Interface Design - TAMODIA 2004, Prague, Czech Republic, 15-16 November 2004. ACM, New York, NY, USA, pp. 165-172. DOI=10.1145/1045446.1045475

[252] Winckler, M., Freitas, C. M. D. S., Lima, J. D. De. 2000. Usability remote evaluation for WWW. In CHI '00 Extended Abstracts on Human Factors in Computing Systems (CHI EA '00). ACM, New York, NY, USA, 131-132. DOI=10.1145/633292.633367

[253] Winckler, M., Gaits, V., Vo, D-B., Firmenich, S., Rossi, G. An approach and tool support for assisting users to fill-in web forms with personal information. In Proceedings of the 29th ACM international conference on Design of communication (SIGDOC'11). ACM, New York, NY, USA, 195-202. DOI=10.1145/2038476.2038515 http://doi.acm.org/10.1145/2038476.2038515.

[254] Winckler, M., Palanque, P. (2012) Models as Representations for Supporting the Development of e-Procedures. In: Usability in Government Systems: User Experience Design for Citizens and Public Servants. Buie, E. & Murray, D. (eds.). Morgan Kaufmann, 2012. pp. 301-315.

[255] Winckler, M.; Palanque, P. StateWebCharts: a Formal Description Technique Dedicated to Navigation Modelling of Web Applications. International Workshop on Design, Specification and Verification of Interactive Systems - DSVIS'2003, Funchal, Portugal, June 2003. (Lecture Notes)

[256] Winckler, M.; Palanque, P.; Farenc, C.; Pimenta, M. Who does what with whom in Web Development? 2nd International Workshop on Task Models and Diagrams for User Interface Design - TAMODIA'2003, Héraklion, Grèce, June 2003.

[257] Winckler, M., Palanque, P., Farenc, C., Pimenta, M. Task-Based Assessment of Web Navigation Design. First International Workshop in Task Models and Diagrams for User Interface Design - TAMODIA'2002, Bucarest, Romania, 18-19 July 2002.

[258] Winckler, M., Pimenta, M., Palanque, P., Farenc, C. Usability Evaluation Methods: What is still missing for the Web? In proc. of the 8th International Conference on HCI International, New Orleans, USA, 5-10 August 2001.

[259] Winckler, M., Pontico, F. A Model-Driven Architecture for Logging Navigation. In Proceedings of the International Workshop on Logging Traces of Web Activity: The Mechanics of Data Collection. Co-located with 15th International World Wide Web Conference (WWW2006). Edinburgh, Scotland, Tuesday, May 23-27, 2006.

[260] Winckler, M., Vanderdonckt, J. Towards a User-Centered Design of Web Applications based on a Task Model. In Proceedings of 5th International Workshop on Web-Oriented Software Technologies (IWWOST'2005). Porto, Portugal, 12th-13th June 2005.

[261] Winckler, M., Vanderdonckt, J., Trindade, F., Stanciulescu, A. Cascading Dialog Modeling with UsiXML. International Workshop on the Design, Verification and Specification of Interactive Systems (DSVIS'2008). Kingston, Ontario, Canada, July 16-18 2008. Springer LNCS 5136. pp. 121-135.

[262] Winckler, M., Xiong, J., Noirhomme-Fraiture, M. Accessibility Legislation and Codes of Practice: an Accessibility Study of Web Sites of French and Belgium Local Administrations. In: proceedings of the 1st International Workshop on Design & Evaluation of e-Government Applications and Services (DEGAS'07) Rio de Janeiro, Brazil, September 11th, 2007. CEUR Workshop Proceedings, ISSN 1613-0073, online at http://ceur-ws.org/Vol-285.

[263] Woerndl, W. Authorization of User Profile Access in Identity Management. ICWI, page 309-316. IADIS, (2004).

[264] Wroblewski, L. Which One: Responsive Design, Device Experiences, or RESS? Available at: http://www.lukew.com/ff/entry.asp?1509

[265] Xiong, J. (2008) Une méthode d'inspection automatique de recommandations ergonomiques tout au long du processus de conception des applications Web. PhD thesis presented on September 29th 2008. Université Paul Sabatier. Available at http://www.irit.fr/~Marco.Winckler/xiong-these2008.pdf

[266] Xiong, J.; Diouf, M.; Farenc, C.; Winckler, M. Automatic Guidelines Inspection: From Web site Specification to Deployment. In proceedings of 6th International Conference on Computer-Aided Design of User Interfaces CADUI'2006. Bucharest, Romania, June 5-8, 2006.

[267] Xiong, J., Farenc, C., Winckler, M. An Ontology-Based Approach for Dealing with Web Guidelines. 2nd Int. Workshop on Web Usability and Accessibility (IWWUA), In conjuntion with the WISE' 2008. Auckland, New Zealand Sept. 1-4, 2008. Springer LNCS 5176, 132-141 pp.

[268] Xiong, J., Winckler, M. (2008) An investigation of tool support for accessibility assessment throughout the development process of Web sites. Journal of Web Engineering (JWE) (special issue about Web Usability and Accessibility). Rinton Press. Vol.7 No.4. pages: 281-298

Curriculum Vitae

Marco WINCKLER

April 2016

Interactive Critical Systems (ICS) - Institute of Research in Informatics of Toulouse (IRIT)
University Paul Sabatier (Toulouse 3), 118 route de Narbonne, 31062 Toulouse CEDEX 9, France
Phone: +33 (0)5.61.55.63.59 / E-mail: winckler@irit.fr / Web: http://www.irit.fr/~Marco.Winckler/

Identification

Full Name: Marco Antonio ALBA WINCKLER

Nationality: Brazilian

Education

(2004) **PhD thesis**: "StateWebCharts: a Formal Notation for Navigation Modelling of Web Applications". *Université Toulouse 1*, Toulouse, France. Supervisors: Philippe Palanque and Christelle Farenc. Jury: Jean Vanderdockt (Examiner), Dominique Scapin (Examiner), Oscar Pastor (Examiner), Joëlle Coutaz (President of the Jury), Rémi Bastide.

(1999) **Master thesis:** "A method for remote usability evaluation of Web sites". *Universidade Federal do Rio Grande do Sul* (UFRGS), Porto Alegre (RS), Brazil. Supervisors: José Valdeni de Lima and Carla Maria Dal Sasso Freitas.

(1996) **Undergraduated** in Computer Science, *Universidade de Passo Fundo*, Brazil.

Mobility

(2006-2007) **Post-doc** internship from September 2006 to February 2007 at Belgian Laboratory of Computer-Human Interaction (BCHI), *Université catholique de Louvain*, Louvain-la-Neuve, Belgium. Supervisor: Jean Vanderdonckt.

Current position

(Since 2005) **Assistant Professor** (*Maître de Conference*) at the Department of Informatics of the *Université Paul Sabatier* (Toulouse 3), Toulouse, France. From 2005 to 2009 attached to the department SERECOM of the *IUT de Tarbes*. From 2009 onwards, attached to the FSI (Faculty of Sciences and Engineering/*Faculté des Sciences et Ingénierie*).

(Since 2000) Member of the Institute of Research in Informatics of Toulouse (IRIT, UMR 5505), **ICS team**.

Other professional activities

(2014-2016) **Chair** of the IFIP Working Group 13.2 (Methodologies for User-Centered Systems Design).

(2013-2016) **Secretary** for the IFIP TC 13 (Human-Computer Interaction).

(2013-2016) **Executive Members** of the European Association of Cognitive Ergonomics (EACE).

(2011-2015) **Council member** (elected) of the *Faculté des Sciences et Ingénierie*, *Université Paul Sabatier*.

(2011-2014) **Expert member** for publication for the IFIP TC 13 (Human-Computer Interaction).

(2010-2014) **Vice-chair** of the IFIP Working Group 13.2 (Methodologies for User-Centered Systems Design).

(2010-2013) **Member** of the ACM SIGCHI Specialized Conferences Committee.

(2010-2013) **Member** of the ACM SIGCHI International Public Policy Committee.

(2008-2011) **Steering Committee Member** of the council on Human-Computer Interaction (CEIHC) of the Brazilian Computing Society (SBC).

Other scientific affiliations

(Since 2010) **ISWE** (International Society for Web Engineering).

(Since 2010) **Member** of the IFIP Working Group 13.4/2.7 (User Interface Engineering).

(Since 2008) **ACM SIGCHI** (Association for Computing Machinery, Special Interest Group on CHI).

(Since 2005) **Member** of the IFIP Working Group 13.2 (Methodologies for User-Centered Systems Design).

(Since 2001) **AFIHM** (*Association Francophone d'Interaction Homme-Machine*)

(Since 1998) **SBC** (Brazilian Computing Society/*Sociedade Brasileira da Computação*).

Awards

(2012) **Outstanding service award**. Granted by the International Federation for Information Processing (IFIP) on behalf of the Technical Committee TC13 on Human-Computer Interaction during the Amsterdam (The Netherlands) assembly, September 2012.

(2011) **Best paper award** of the International Conference on Web Engineering (ICWE 2011), Paphos, Cyprus, June 20-24th, 2011.

Invited keynotes

(2013) "Tasks Model Composition: beyond data, representing user activities", keynote at the 5th International Workshop on Lightweight Composition on the Web (**ComposableWeb 2013**), July 8th 2013, Aalborg, Denmark.

(2012) "Beyond an Application Domain: Web Engineering as a Research Perspective for Human-Computer Interaction", keynote at the 13th edition of the International Conference on Human-Computer Interaction (**Interacción 2012**), October 3-5th 2013, Elche, Spain.

Student supervision

Co-supervision of PhD thesis

(2015- …) **HAK, Jean Luc**. "An approach for supporting the traceability of design decisions and the evolution of prototypes during the development processes of interactive systems". Started in September 2015 at University of Toulouse, France.

(2014- …) **ROCHA SILVA, Thiago**. "Definition of a model-based approach for the specification of requirements and test validation of Web applications". Started in September 2014 at University of Toulouse, France.

(2013- …) **DE LIMA COSTA, Thaíse Kelly**. "A Web Portal for the Diffusion of Serious Games for Health Care Training". Started in March 2013 at Universidade Federal da Paraíba (UFPA), Brazil. Co-supervision with Liliane dos Santos Machado and Ana Maria Gondim Valença. PhD Internship in University of Toulouse, France.

(2013- …) **ALDALUR, Iñigo**. "An approach for supporting End-User Development (EUD) though Web augmentation techniques". Started in September 2013 at Universidad del Pais Vasco (UPV/EHU), Spain. Co-supervision with Oscar Diaz.

(2012- …) **ALBERTOS MARCO, Félix**. "Supporting Offline Interaction with Web Sites Resilient to Interruptions Applied to E-learning Environments". Started in September 2012 at *Universidad de Castilla-La Mancha*, Albacete, Spain. Co-supervision with Victor Penichet & José Gallud.

(2010-…) **CAVA, Ricardo.** "Integrating Graph Visualization and Multidimensional Data Visualization Techniques". Started in Mars 2010 at *Universidade Federal do Rio Grande do Sul* (UFRGS), Porto Alegre, Brazil. Co-supervision with Carla M. D. S. Freitas.

(2009-2013) **FIRMENICH, Sergio**. "Client-side adaptation of Web Applications". *Universidad Nationale de La Plata* – UNLP, La Plata, Argentina. Presented on April 11th 2013. Co-supervision (50%) with Gustavo Rossi and Silvia Gordillo.

(2009-2011) **MARTINIE, Célia**. "A Synergistic Models-Based Approach to Develop Usable, Reliable and Operable Interactive Critical Systems". University of Toulouse, France. Presented on December 5th 2011. Co-supervision (50%) with Philippe Palanque.

(2004-2008) **XIONG, Joseph**. "An evaluation method for automatic inspection of ergonomic guidelines throughout the development process of Web". University of Toulouse, France. Presented on Septembre 29th 2008. Co-supervised (50%) with Christelle Farenc.

(2004-2008) **PONTICO, Florence.** "A pattern-based user interface design method for e-Government applications". University of Toulouse, France. Presented on July 11th 2008. Co-supervised (50%) with Regina Bernhaupt.

Participation in PhD evaluation committees

(2015) **BASTOS E MARQUES DA SILVA, Carlos Eduardo**. *The MAP-I Doctoral Program of the Universities of Minho, Aveiro and Porto, Braga, Portugal, September 25th 2015.*

(2014) **DE SOSA, Josune**. *University of the Basque Country*. San Sebastian, Spain, September 11th 2014.

(2013) **GASPARINI, Isabela**. *Universidade Federal do Rio Grande do Sul (UFRGS)*, Porto Alegre, Brazil, December 20th 2013.

MASIP, Llúcia. *Universidad de Lleida*, Lleida, Spain, December 13th 2013.

(2012) **MBAKI LUZAYISU, Efrem**. *Université catholique de Louvain*, Louvain-la-Neuve, Belgium. October 9th 2012.

(2010) **GUERRERO GARCIA, Josefina**. *Université catholique de Louvain*, Louvain-la-Neuve, Belgium. June 2nd, 2010.

(2010) **PANACH NAVARRETE, Jose Ignacio**. *Universidad Politécnica de Valencia*, Valencia, Spain. May 24th, 2010.

(2009) **GANNEAU, Vincent**. *Université Joseph Fourier*, Grenoble, France. January 20th, 2009.

(2008) **SOTTET, Jean-Sébastien.** *Université Joseph Fourier*, Grenoble, France. October 10th, 2008.

Organization of scientific events

As program co-chair/co-organizer

(2017) IFIP TC13 INTERACT Conference, Mumbai, India, Stockholm, Sweden, August 29th-31th, 2016.

(2016) 6th International Working Conference on Human-Centred Software Engineering and 8th International Working Conference on Human Error, Safety, and System Development (HCSE+HESSD 2016), Stockholm, Sweden, August 29th-31th, 2016.

(2015) 5th International Conference on Application and Theory of Automation in Command and Control Systems, Toulouse, France — September 30 — October 2, 2015

(2014) International Conference on Web Engineering (ICWE 2014), July 1-4, 2014, Toulouse, France.

International Conference on Human-Centered on Software Engineering (HCSE'2014), September 16-18, 2014, Paderborn, Germany

(2013) International workshop on Engineering Mobile Web Applications (Emotions'2013). July 10th 2013, in conjunction with ICWE 2013, Aalborg, Denmark.

International workshop on Public Policies and HCI. April 28th 2013, in conjunction with ACM SIGCHI Conference (CHI 2013), Paris, France.

(2012) International Conference on Human-Centered on Software Engineering (HCSE'2012), October 29-31, 2012, Toulouse, France.

(2011) IFIP TC 13.2 Workshop on Software and Usability Engineering Cross-Pollination (PUX'2011). September 6th 2011, in conjunction with INTERACT'2011, Lisbon, Portugal.

(2009) International Workshop on Design & Evaluation of e-Government Applications and Services (DEGAS'2009). August 24th 2009, in conjunction with INTERACT 2009, Uppsala, Sweden.

8th International Workshop on Web-Oriented Software Technologies (IWWOST'2009). June 23th 2009, in conjunction with ICWE'2009, San Sebastian, Spain.

(2008) Brazilian Symposium on Human-Computer Interaction (IHC'2008). October, 21-24 2008, Porto Alegre, Brazil.

International Workshop on Web-Oriented Software Technologies (IWWOST'2008). July 14th 2008, in conjunction with ICWE'2008, New York, USA.

(2007) International Workshop on Task Models and Diagrams for User Interface Design (TAMODIA'2007), November 7-9, 2007, Toulouse, France.

International Workshop on Web-Oriented Software Technologies (IWWOST'2007). July 18th 2007, in conjunction with ICWE'2007, Como, Italy.

International Workshop on Design & Evaluation of e-Government Applications and Services (DEGAS'2007). September 10th 2007, in conjunction with INTERACT'2007, Rio de Janeiro, Brazil.

As technical track co-chair

(2016) ACM Engineering Interactive Systems (EICS 2016), Brussels, Belgium (PhD Symposium).

International Conference on Web Engineering (ICWE'2016), Lugano, Switzerland (PhD Symposium).

(2015) ACM Engineering Interactive Systems (EICS 2015), Duisburg, Germany (Late Breaking Results).

International Conference on Web Engineering (ICWE'2015), Lugano, Switzerland (HCI track).

(2014) ACM Engineering Interactive Systems (EICS 2014), Rome, Italy (tutorials chair).

(2013) ACM Engineering Interactive Systems (EICS 2011), Pisa, Italy (workshop chair).

International Conference on Web Engineering (ICWE'2013), Aalborg, Denmark (PhD track).

(2012) Brazilian Symposium on Human-Computer Interaction (IHC'2010), Brazil (industrial papers).

(2011) ACM Engineering Interactive Systems (EICS 2011), Pisa, Italy (tutorials).

(2010) Brazilian Symposium on Human-Computer Interaction (IHC'2010), Brazil (tutorials).

(2006) Brazilian Symposium on Human-Computer Interaction, IHC'2006, Natal, Brazil (short papers).

As organization, publication, or publicity chair/co-chair

(2015) IFIP TC13 INTERACT'2015, Bamberg, Germany (publication).

(2013) IFIP TC13 INTERACT'2013, Cape Town, South Africa (publication).

International Conference on Application and Theory of Automation in Command and Control Systems (ATACCS'2013), Naples, Italy (publicity).

(2012) International Conference Fun & Games'2012, Toulouse, France (organization and publication).

International Conference on Application and Theory of Automation in Command and Control Systems (ATACCS'2012), London, United Kingdom (publicity and publication).

(2011) IFIP TC13 INTERACT'2011, Lisbon, Portugal (publication).

International Conference on Application and Theory of Automation in Command and Control Systems (ATACCS'2011), Barcelona, Spain (publicity and publication).

(2010) French Conference on Human-Computer Interaction (IHM'2010), Luxembourg (publication).

ACM Symposium on Engineering Interactive Computing Systems (EICS'2010), Berlin, Germany (publicity).

(2009) IFIP TC13 INTERACT'2009, Uppsala, Sweden (publication chair).

ACM Symposium on Engineering Interactive Computing Systems (EICS'2009), Pittsburgh, United States of America (publicity).

(2007) FIP TC13 INTERACT'2007, Rio de Janeiro, Brazil (publication).

Reviewing activities

Member of conferences program committee

(2016) ACM Engineering Interactive Systems (EICS 2016), Brussels, Belgium (AC)

AVI 2016 - International Working Conference on Advanced Visual Interfaces, Bari, Italy.

Brazilian Symposium on Human-Computer Interaction (IHC'2016), São Paulo, Brazil.

IHM 2016 - 28ème édition, la conférence francophone sur l'Interaction Homme-Machine vient en Suisse, à Fribourg.

INDIN (2016 IEEE International Conference on Industrial Informatics), Poitiers, France.

International Conference Multimedia-Interaction-Design-Innovation (MIDI'2016), Gdansk, Poland.

International Conference on Web Engineering (ICWE'2016), Lugano, Switzerland.

ManComp 2016 - 1st Workshop on Managed Complexity, Prague, Czech Republic.

ROCHI'2016, the International Conference on Human-Computer Interaction, Iasi, Romenia.

Spanish Conference on Human-Computer Interaction (INTERACCION'2016), Salamanca, Spain.

XVIII Brazilian Symposium on Virtual and Augmented Reality (SVR'2016), Gramado, Brazil.

XXIV Workshop sobre Educação em Computação

(2015) ACM Engineering Interactive Systems (EICS'2015), Duisburg, Germany.

ATACCS 2015 – 5th International Conference on Application and Theory of Automation in Command and Control Systems, Toulouse, France.

Brazilian Symposium on Human Factors in Computing Systems (IHC'2015), Salvador, Brazil.

HuFo 2015 International Workshop on Human Factors in Software Development Processes, Bozen-Bolzano, Italy.

IFIP TC13 INTERACT'2015, Bamberg, Germany.

International Conference Multimedia-Interaction-Design-Innovation (MIDI'2015), Warsaw, Poland.

International Conference on Web Engineering (ICWE'2015), Rotterdam, Netherlands.

International Workshop on Emerging Web Application Design (EWAD2015), Stockholm, Sweden.

International Workshop on Human Factors in Software Development Processes, Bozen-Bolzano, Italy.

MHMC 2015 - 1st International Workshop on Multimodal Interaction in Industrial Human-Machine Communication, Luxemburg.

S-BPM ONE 2015. 7th International Conference on Subject-Oriented Business Process Management. University of Erlangen-Nuremberg, Germany.

Spanish Conference on Human-Computer Interaction (INTERACCION'2015), Vilanova i la Geltrú, Cataluña.

WEI 2015 - XXIII Workshop sobre Educação em Computação

XVII Brazilian Symposium on Virtual and Augmented Reality (SVR'2015), Rio e Janeiro, Brazil.

(2014) ACM Symposium on Applied Computing, track on User Interface Generation, Gyeongju, Korea.

ACM AVI 2014 - International Working Conference on Advanced Visual Interfaces, Como, Italy.

ACM Engineering Interactive Systems (EICS'2014), Rome, Italy.

Brazilian Symposium on Human Factors in Computing Systems (IHC'2014), Foz do Inguaçu, Brazil.

International Conference on Cloud Computing and Services Science (CLOSER'2014), Barcelona, Spain.

International Conference on Information Systems and Design of Communication (ISDOC'2014), Lisbon, Portugal.

International Conference Multimedia-Interaction-Design-Innovation (MIDI'2014), Warsaw, Poland.

International Conference on Web Engineering (ICWE'2014), Toulouse, France.

ICWE Workshop on Crowdsourced Web Engineering (CroWE), Zurich, Switzerland

Spanish Conference on Human-Computer Interaction (INTERACCION'2014), Puerto de la Cruz, Tenerife, España.

International Conference on Web Information Systems and Technologies (WebIST'2014), Barcelona, Spain.

Simpósio Brasileiro de Informática da Educação - 25° SBIE.

V WEIHC - Workshop sobre Ensino de IHC

Workshop sobre questões Culturais em IHC

World Wide Web Conference (WWW'2014), Seoul, Korea.

XVI Brazilian Symposium on Virtual and Augmented Reality (SVR'2014), Salvador, Brazil.

(2013) ACM SIGCHI Conference (CHI'2013), Paris, France.

ACM Engineering Interactive Systems (EICS'2013), London, UK.

ACM Special Interest Group on Design of Communication (SIGDOC'2013), Greenville, USA.

Brazilian Symposium on Human Factors in Computing Systems (IHC'2013), Manaus, Brazil.

British HCI conference (BHCI'2013), West London, United Kingdom.

IEEE International Workshop on Communicating Business Process and Software Models (CPSM'2013), Eindhoven, the Netherlands.

IFIP TC13 Conference on Human-Computer Interaction (INTERACT'2013), Cape Town.

International Conference on Cloud Computing and Services Science (CLOSER'2013), Barcelona, Spain.

International Conference on Entertainment Computing (ICEC'2013), São Paulo, Brazil.

International Conference on Information Systems and Design of Communication (ISDOC'2013), Lisbon, Portugal.

International Conference Multimedia-Interaction-Design-Innovation (MIDI'2013), Gdansk, Poland.

International Conference on Web Engineering (ICWE'2013), Aalborg, Denmark.

International Workshop on Model-Driven and Agile Engineering for the Web (MDWE'2013), Aalborg, Denmark.

International Conference on Web Information Systems and Technologies (WebIST'2013).

International Workshop on Quality in Web Engineering (QWE'13), Aalborg, Denmark.

Spanish Conference on Human-Computer Interaction (INTERACCION'2013), Madrid, Spain.

World Wide Web Conference (WWW'2013), Rio de Janeiro, Brazil.

XV Brazilian Symposium on Virtual and Augmented Reality (SVR'2013), Cuiabá, Brazil.

(2012) • International Joint Conference on Ambient Intelligence (AmI'2012) • Int. Workshop on Quality in Web Engineering (QWE'2012) • 1st Int. Workshop on Crowdsourced Web Engineering • 8thInternational Workshop on Model-Driven Web Engineering (MDWE 2012) • British HCI conference (HCI 2012) • WWW/Interenet (CIAWI'2012) • XIII Interacción Persona-Ordenador (Interacción'2012) • 14th IEEE International Symposium on Web Systems Evolution (WSE'2012) • 11th International Conference on Entertainment Computing (ICEC'2012) • Workshop Open Source and Design of Communication OSDOC2012 • ACM SIGCHI Conference on Human Factors in Computing Systems (Work in Progress track) • Fun and Games (FnG 2012) (local organization and publication chair) • ACM Engineering Interactive Systems (EICS2012) • ACM International Conference on Design of Communication (SIGDOC 2012) • Brazilian Symposium on Human Factors in Computing Systems (IHC'2012) (industrial papers co-chair) • International Conference on Web Engineering (ICWE'2012) • HCSE'2012 - 4th Conference on Human Centred Software Engineering (general co-chair) • WEBIST'2012 - Int. Conference on Web Information Systems and Technologies • WWW'2012 - World Wide Web Conference • 1st International Conference on Application and Theory of Automation in Command and Control Systems (ATACCS 2012) (Publicity & Publication chair) • 2nd International Conference on Cloud Computing and Services Science (CLOSER 2012) • Intelligent User Interfaces (IUI'2012) • 13th Ibero-American Conference on Artificial Intelligence (IBERAMIA'2012)

(2011) • IFIP TC13 Conference on Human-Computer Interaction INTERACT'2011 • International Conference on Web Engineering ICWE'2011 • ACM Engineering Interactive Systems (EICS2011) • International Conference on Entertainment Computing (ICEC 2011) • International Workshop on Supportive User Interfaces (SUI 2011) in conjunction with EICS 2011 • ACM International Conference on Design of Communication (SIGDOC 2011) • European Workshop on HCI Design and Evaluation (EHCIDE'2011), Limassol, Cyprus • International Workshop on DATA Visualization and Integration in Enterprises and on the Web (DATAVIEW'2011) • Brazilian Symposium on Human Factors in Computing Systems and Latin American Conference on Human-Computer Interaction (IHC+CLIHC'2011) • International Workshop on Model-Driven Web Engineering (MDWE 2011) in conjunction with ICWE2011 • International Workshop on Quality in Web Engineering (QWE'2011) in conjunction with ICWE2011 • International Workshop on The Web and Requirements Engineering (WeRE'2011) in conjunction with ICWE2011.

(2010) • Workshop Open Source and Design of Communication (OSDOC'2010) • DATAVIEW'10 International Workshop on DATA Visualization and Integration in Enterprises and on the Web • ACM International Conference on Design of Communication (SIGDOC'2010) • ACM Symposium on Applied Computing SAC'2010 (HCI track) • ACM Symposium on Engineering Interactive Systems EICS 2010 • Conference on Human-Centred Software Engineering (HCSE 2010) • International Conference on Web Engineering ICWE'2010 • IFIP Conference on Human-Centred Software Engineering (HCSE 2010) • 22nd French Conference on Human-Computer Interaction - IHM'2010 • ACM International Conference on Design of Communication (SIGCDOC 2010) • 9th Brazilian Symposium on Human-Computer Interaction, IHC'2010 • International Workshop on The Web and Requirements Engineering (WeRE'10) in conjunction with IEEE's Requirements Engineering Conference (RE) • International Workshop on on Quality in Web Engineering (QWE'2010) in conjunction Int. Conference on Web Engineering (ICWE) • International NordiCHI 2010 • 6th International Workshop on Model-Driven Web Engineering (MDWE 2010) in conjunction Int. Conference on Web Engineering (ICWE).

(2009) • ACM Symposium on Engineering Interactive Systems EICS 2009 • Latin-American Conference on Human-Computer Interaction, CLIHC 2009 • 9th International Conference on Web Engineering, ICWE 2009 • 3rd International Workshop on Web Usability and Accessibility, IWWUA'2009 • 5th Model-Driven Web Engineering Workshop, MDWE 2009, in conjunction with ICWE'2009 • 8th International workshop on TAsk MOdels and DIAgrams, TAMODIA 2009 • International Conference Interfaces and Human Computer Interaction, IADIS 2009.

(2008-2004) • ICWE 2008 • CADUI 2008 • DSVIS'2008 • EIS'2008 joint with HCSE 2008 and TAMODIA 2008 • IWWUA'2008 • LA-WEB'2008 • CIAWI'2007 • DEGAS'2007 • EIS'2007 • IADIS'2007 • IFIP TC13 INTERACT'2007 • IWWOST'2007 • IWWUA'2007 • TAMODIA'2007 • IHC'2006 • TAMODIA'2006 • IADIS'2005 • INTERACT'2005 • TAMODIA'2005 • IHC'2004 • TAMODIA'2004.

Reviewer for journals

Behaviour & Information Technology - BIT (Taylor & Francis)
Entertainment Computing (Elsevier)
Innovations in Systems and Software Engineering – ISSE (Springer)
Interacting with Computers - IwC (Elsevier)
International Journal of Human-Computer Interaction (Taylor & Francis)
International Journal of Human-Computer Studies - IJHCS (Elsevier)
International Journal of Web Engineering and Technology - IJWET (InderScience Publishers)
Journal of the Brazilian Computer Society – JBCS (Springer)
Journal of Multimodal User Interfaces - JMUI (Springer)
Journal of Symbolic Computation - JSC (Elsevier)
Journal of Systems and Software (Elsevier)
Journal of the Brazilian Computer Society (JBCS) Springer
Journal of Universal Computer Science - JUCS (TUGraz)
Journal of Web Engineering - JWE (Rinton Press)
Revista de Iniciação Científica (REIC - SBC)
Revue Ingénierie des systèmes d'information (ISI)
Science of Computer Programming – SCP (Elsevier)
Software Quality Journal - SQJ (Springer)
Software Testing, Verification and Reliability (Wiley & Sons)
Transactions on Computer Human Interaction – TOCHI (ACM)
Transactions on the Web – TWEB (ACM)
World Wide Web Journal – WWWJ (Springer)

Publications

Journals (11)

(2016) Lazar, J., Abascal, A., Barbosa, S., Barksdale, J., Friedman, B., Grossklags, J., Gulliksen, J., Johnson, J., McEwan, T., Martinez-Normand, L., Michalk, W., Tsai, J., VanDerVeer, G., vonAxelson, H., Walldius, A., Whitney. G., Winckler, M., Wulf, V., Churchill, E., Cranor, L., Davis, J., Hedge, A., Hochheiser, H., Hourcade, J-P., Lewis, C., Nathan, L., Paterno, F., Reid, B., Quesenbery, W., Selker, T., and Wentz, B. (2015). Human-Computer Interaction and International Public Policymaking: A Framework for Understanding and Taking Future Actions. Foundations and Trends in Human-Computer Interaction 9 (2), 69-149.

(2014) Firmenich, S., Rossi, G., Winckler, M., Palanque, P. An Approach for Supporting Distributed User Interface Orchestration over the Web. In. International Journal of Human-Computer Studies. ISSN 1071-5819. Vol. 72(1): 53-76 (2014). At: http://dx.doi.org/10.1016/j.ijhcs.2013.08.014

Normand, L. M., Paternò, F., Winckler, M. Public policies and multilingualism in HCI. Interactions 21(3): 70-73 (2014)

Barboni, E., Martinie, C., Navarre, D., Palanque, P., Winckler, M. (2013) Bridging the gap between a behavioural formal description technique and a user interface description language: Enhancing ICO with a graphical user interface markup language. In Journal of Science of Computer Programming. Elsevier. Sci. Comput. Program. 86: 3-29.

(2013) Winckler, M., Bach, C., Bernhaupt, R. Identifying User Experience Dimensions for Mobile Incident Reporting in Urban Contexts. In IEEE Transactions on Professional Communication. Volume 56 , Issue 2, June 2013, pp. 97-119. [doi: http://dx.doi.org/10.1109/TPC.2013.2257212]

(2012) Lazar, J., Prates, R., von Axelson, H., Winckler, M., Wulf, V., Abascal, J., Davis, J., Evers, V., Gulliksen, J., Jorge, J., McEwan, T., Paterno, F., Persson, H. (2012) HCI public policy activities in 2012. Interactions (New York, N.Y.). , v.19, p.78-81. [doi:10.1145/2168931.2168947]

(2011) Firmenich, S., Winckler, M., Rossi, G., Gordillo, S. (2011) A Crowdsourced Approach for Concern-Sensitive Integration of Information across the Web. Journal of Web Engineering (JWE). Rinton Press., Vol.10 No.4, December 2011, pages: 289-315.

(2010) Winckler, M. (2010). *L'Administration Électronique*: The French Approach to E-Government. Interactions, Vol. XVII, n. 6, November+December 2010, p. 52-55.

Winckler, M., Bernhaupt, R., Pontico, F. (2010) Challenges for the Development of User Interface Pattern Languages: a case study on the e-Government domain. IADIS International Journal on WWW/Internet. Vol. 8, N. 2. Pages 59-84.

(2008) Xiong, J., Winckler, M. (2008) An investigation of tool support for accessibility assessment throughout the development process of Web sites. Journal of Web Engineering (JWE) (special issue about Web Usability and Accessibility). Rinton Press. Vol.7 No.4. pages: 281-298

(2006) Winckler, M., Barboni, E., Palanque, P., Farenc, C. (2006). What Kind of Verification of Formal Navigation Modelling for Reliable and Usable Web Applications? Electronic Notes Theoretical Computer Science 157(2). pages: 207-211

Book chapters (8)

(2015) Martinie, C., Palanque, P., Winckler, M. Designing and Assessing Interactive Systems Using Task Models. p:29-58. Brazilian Computing Society.

(2012) Winckler, M., Palanque, P. (2012) Models as Representations for Supporting the Development of e-Procedures. In: Usability in Government Systems: User Experience Design for Citizens and Public Servants. Buie, E. & Murray, D. (eds.). Morgan Kaufmann, 2012. pp. 301-315.

(2011) Palanque, P., Winckler, M., Martinie, C. (2011) A Formal Model-Based Approach for Designing Interruptions-Tolerant Advanced User Interfaces. In : Model-Driven Development of Advanced User Interfaces. Hussmann, H., Meixner, G. & Zuehlke, D. (Eds.). Springer, Studies in Computational Intelligence Series, Vol. 1061. pp. 141-170, 304p., ISBN: 978-3-642-14561-2.

Calvary, G., Serna, A., Scapin, D., Winckler, M. (2011) Advanced User Interfaces for e-Government Applications. In: Practical Studies in e-Government (chapter 12). Assar, S.,

Boughzala, I., Boydens, I. (eds.) Springer. 1st Edition., 2011, X, 240 p. ISBN: 978-1-4419-7532-4

(2010) Navarre, D., Palanque, P., Martinie, C., Winckler, M., Steere, S. (2010) Formal Description Techniques for Human-Machine Interfaces. In: The Handbook of Human-Machine Interaction. Boy, G. (editor). Ashgate Publishing, 2010, pp. 235-266.

(2009) Navarre, D., Palanque, P., Winckler, M. (2009) Task Models and System Models as a Bridge between HCI and Sofware Engineering. In: Human-Centered Software Engineering Software Engineering Models, Patterns and Architectures for HCI. (chapter 17). Seffah, A., Vanderdonckt, J., Desmarais, M. (eds.) Springer (Human-Computer Interaction Series), June 2009, pp. 357-385.

(2007) Bernhaupt, B., Navarre, D., Palanque, P., Winckler, M. Model-Based Evaluation: A New Way to Support Usability Evaluation of Multimodal Interactive Applications. In Maturing Usability: Quality in Software, Interaction and Quality. Springer (Human-Computer Interaction Series), October 2007, Law E., Thora Hvannberg E., Cockton G. & Vanderdonckt J. (Eds.). pp. 96-122.

(2002) Winckler, M., Pimenta, M. (2002) *Usability Evaluation of Web sites/Avaliação de Usabilidade de Sites Web*. In: Nedel, L. (editor) X Escola de Informática da SBC-Sul (ERI2002), Caxias do Sul, Criciúma, Cascavel, Brazil. 2002. pp. 85-137. (in Portuguese).

Full research papers in international conferences with peer reviewing (44)

(2016) Bosetti, G., Firmenich, S., Rossi, G., Winckler, M. Web Objects Ambient: an integrated platform supporting new kinds of Personal Web experiences. (to appear) ICWE 2016.

(2015) Firmenich, D., Firmenich, S., Winckler, M., Rossi, G., Distante, D. User Interface Adaptation Using Web Augmentation Techniques: Towards a Negotiated Approach. International Conference on Web Engineering 2015 (ICWE). LNCSvol:9114. p:147-164. Springer.

Winckler, M., Freitas, C. M. D. S., Palanque, P., Cava, R., Barboni, E. Usability aspects of the inside-in approach for ancillary search tasks on the web. IFIP TC13 Conference on Human-Computer Interaction 2015 (INTERACT). LNCSvol:9297. p:207-226. Springer.

(2014) Forbrig, P., Martinie, C., Palanque, P., Winckler, M., Fahssi, R. Rapid Task-Models Development Using Sub-models, Sub-routines and Generic Components. HCSE 2014: 144-163

Cava, R. A., Freitas, C. M. D. S., Barboni, E., Palanque, P., Winckler, P. Inside-In Search: An Alternative for Performing Ancillary Search Tasks on the Web. LA-WEB 2014: 91-99

(2013) Bach, C., Bernhaupt, R., Stein D'Agostini, C., Winckler, M. Mobile Applications for Incident Reporting Systems in Urban Contexts: lessons learned from an empirical study. In Proc. of ECCE 2013, Toulouse, France, August 26-28, 2013, ACM DL.

Galindo, M., Martinie, C., Palanque, P., Winckler, M., Forbrig, P. Tuning an HCI Curriculum for Master Students to Address Interactive Critical Systems Aspects. In Proceeding of HCI (1) 2013, Las Vegas USA, July 21-26 2013, Springer, LNCS 8004, p. 51-60

Winckler, M., Bach, C., Bernhaupt, R. Characterizing Incidents Reporting Systems across Applications Domains. In Proceeding of HCI (1) 2013, Las Vegas USA, July 21-26 2013, Springer, LNCS 8004, p. 521-530

(2012) Firmenich, S., Gaits, V., Gordillo, S., Rossi, G., Winckler, M. Supporting Users Tasks with Personal Information Management and Web Forms Augmentation. In Proc. of the International Conference on Web Engineering (ICWE 2012), Berlin, Germany, July 23-27, 2012. Springer, LNCS 7387, p. 268-282.

Masip, L., Martinie, C., Winckler, M., Palanque, P., Granollers, T. Oliva, M. A Design Process for Exhibiting Design Choices and Trade-offs in (potentially) Conflicting User Interface Guidelines. In proceedings of the 4th Human-Centered Software Engineering (HCSE 2012), Toulouse, France, October 29-31, 2012. Springer, LNCS 7623, p. 53-71.

(2011) Winckler, M., Gaits, V., Vo, D-B., Firmenich, S., Rossi, G. An approach and tool support for assisting users to fill-in web forms with personal information. In Proceedings of the 29th ACM international conference on Design of communication (SIGDOC '11). ACM, New York, NY, USA, 195-202. DOI=10.1145/2038476.2038515 http://doi.acm.org/10.1145/2038476.2038515.

Scapin, D. L., Marie-Dessoude, P., Winckler, M., Detraux, C. Personal Information Systems: User Views and Information Categorization. In Proc. of CENTRIC'2011, Barcelone, October 28-29, 2011, IARIA, pp.40-47. ISBN: 978-1-61208-167-0.

Firmenich, S., Winckler, M., Rossi, G., Gordillo, S. A Framework for Concern-Sensitive, Client-Side Adaptation. In Proceedings of International Conference on Web Engineering (ICWE 2011), Paphos, Cyprus, June 20-24, 2011. Springer, LNCS 6757, pages 198-213. Also available at: http://dx.doi.org/10.1007/978-3-642-22233-7_14 (**best paper award**)

Martinie, C., Palanque, P., Winckler, M. Structuring and Composition Mechanism to Address Scalability Issues in Task Models. IFIP TC13 Conference on Human-Computer Interaction (INTERACT 2011), September 05-09 2011, Lisboa, Portugal, Springer LNCS 6949, pp. 593-611.

Martinie, C., Palanque, P., Barboni, E., Winckler, M., Ragosta, M., Pasquini, A., Lanzi, P. Formal Tasks and Systems Models as a Tool for Specifying and Assessing Automation Designs. In: 1st International Conference on Application and Theory of Automation in Command and Control Systems. (ATACCS2011) May 26-27, 2011, Barcelona, Spain. pp.46-50.

Palanque, P., Barboni, E., Martinie, C., Navarre, D., Winckler, M. A Tool Supported Model-based Approach for Engineering Usability Evaluation of Interaction Techniques. In proceedings of ACM SIGCHI conference Engineering Interactive Computing Systems (EICS 2011), Pisa, Italy, June 13-16 2011. ACM DL, Sheridan Press, pages 21-30.

Martinie, C., Palanque, P., Winckler, M., Navarre, D., Poupart, E. Model-Based Training: An Approach Supporting Operability of Critical Interactive Systems: Application to Satellite Ground Segments. In proceedings of ACM SIGCHI conference Engineering Interactive Computing Systems (EICS 2011), Pisa, Italy, June 13-16 2011. ACM DL, Sheridan Press, pages 53-62.

(2010) Robles Luna, E., Garrigos, I., Grigera, J., Winckler, M. Capture and Evolution of Web requirements using WebSpec. In proceedings on International Conference on Web Engineering (ICWE'2010), July 5 - 9, 2010 in Vienna, Austria.

Martinie, C., Navarre, D., Winckler, M. A formal approach supporting effective and efficient training program for improving operators' reliability (regular paper). Dans : Safety and Reliability for managing Risk (ESREL), Rhodes Grece, September 05-09, 2010, Taylor & Francis Group, p. 234-243, 2010.

Martinie, C., Palanque, P., Winckler, M., Conversy, S. DREAMER: a Design Rationale Environment for Argumentation, Modeling and Engineering Requirements. In proceedings of the 28th ACM International Conference on Design of Communication (SIGDOC'2010), September 26-29, 2010, São Carlos, Brazil. ACM Press. pp. 73-80. ISBN: 978-1-4503-0403-0.

Barboni, E., Ladry, J., Navarre, D., Palanque, P., Winckler, M. Beyond Modelling: An Integrated Environment Supporting Co-Execution of Tasks and Systems Models. In proceedings of ACM Symposium on Engineering Interactive Systems (EICS'2010), June 19-23, 2010, Berlin, Germany. ACM DL, Sheridan. Pages 165-174.

(2009) Palanque, P., Winckler, M., Ladry, J.F., ter Beek, M. H., Faconti, G. P., Massink, M. A Formal Approach Supporting the Comparative Predictive Assessment of the Interruption-Tolerance of Interactive Systems In Proceedings of the ACM Symposium on Engineering Interactive Computing Systems (EICS 2009), Pittsburgh, USA, ACM, New York. pp. 211-220.

ter Beek, M. H., Faconti, G. P., Massink, M., Palanque, P., Winckler, M. Resilience of Interaction Techniques to Interrupts: A Formal Model-based Approach In Proceedings of the 12th IFIP TC 13 Conference on Human-Computer Interaction (INTERACT 2009). T. Gross et al. (Eds.), Part I, LNCS 5726, pp. 494–509, Uppsala, Sweden, 24-28 August 2009.

Bernhaupt, R., Winkler, M., Pontico, F. User Interface Patterns: a field study evaluation. In Proc. eedings of IADIS International Conference Interfaces and Human Computer Interaction 2009, Algarve, Portugal, 20 - 22 June. ISBN: 978-072-8924-80-5.

Alberdi, E., Strigini, L., Leach, K., Ryan, P., Palanque, P., Winckler, M. Gaining assurance in a voter-verifiable voting system. In Proceedings of the The Second International Conference on Dependability (DEPEND 2009), IEEE Computer Society, June 18-23, 2009 Athens, Greece. pp. 99-104. DOI 10.1109/DEPEND.2009.2.

Winckler, M., Bernhaupt, R., Palanque, P., Lundin, D., Leach, K., Ryan, P., Alberdi, E., Strigini, L. Assessing the Usability of Open Verifiable e-Voting Systems. In Proc. of the 1st Int. Conference on eGovernment and eGovernnace (ICEGOV). 12-13 March 2009, Ankara, Turkey. Kaplan, A., Balci, A., Aktan, C. C., Dalbay, O. (Eds.). Turksat. Vol. 1. pages 281- 296. ISBN 975-6339-00-0.

(2008) Winckler, M., Vanderdonckt, J., Trindade, F., Stanciulescu, A. Cascading Dialog Modeling with UsiXML. International Workshop on the Design, Verification and Specification of Interactive Systems (DSVIS'2008). Kingston, Ontario, Canada, July 16-18 2008. Springer LNCS 5136. pp. 121-135

Guerrero, J., Vanderdonckt, J., Gonzalez, J.L., Winckler, M. Modeling User Interfaces to Workflow Information Systems. In 4th International Conference on Autonomic and Autonomous Systems (ICAS'08), Gosier, Guadalupe, March 16-21 2008. pp. 55-60, IEEE International. ISSN/ISBN: 0769530931.

(2007) Pontico, F., Winckler, M., Limbourg, Q. Towards a Universal Catalogue of User Interface Patterns for eGovernment Web sites. Sixth International EGOV Conference 2007, Regensburg (Germany), September 3 - 7, 2007.

Bernhaupt, R., Palanque, P., Winckler, M., Navarre, D. Usability Study of Multi-Modal Interfaces using Eye-Tracking. Proceedings of INTERACT 2007, Rio, Brazil, September 2007, Springer LNCS 4663, part 2, pp. 412-424.

Pontico, F., Winckler, M., Limbourg, M. Organizing user interface patterns for e-Government applications. In Proceedings of the Engineering Interactive Systems (EIS), Gulliksen, J., Borup, M., Harning, Palanque, P., Veer, G., Wesson, J. (Eds.). Lecture Notes In Computer Science, Vol. 4940. Springer-Verlag, Berlin, Heidelberg 601-619. DOI=10.1007/978-3-540-92698-6_36.

(2006) Pontico, F., Farenc, C., Winckler, M. Model-based support for specifying eService eGovernment applications. In Proceedings of the 5th international conference on Task models and diagrams for users interface design (TAMODIA'06), Karin Coninx, Kris Luyten, and Kevin A. Schneider (Eds.). Springer LNCS 4385, Berlin, Heidelberg, 54-67. ISBN: 978-3-540-70815-5.

Palanque, P., Bernhaupt, R., Navarre, D., Ould, M., Winckler, M. Supporting Usability Evaluation of Multimodal Man-Machine Interfaces for Space Ground Segment Applications Using Petri net Based Formal Specification. In Proceedings of the 9th International Conference on Space Operations, 2006, Rome, 2006.

Xiong, J., Diouf, M., Farenc, C., Winckler, M. Automatic Guidelines Inspection: From Web site Specification to Deployment. In Proceedings of the 6th International Conference on Computer-Aided Design of User Interfaces CADUI´2006. Calvary, G., Prebeanu,C., Santucci, G., vanderdonckt, J. (eds.) Bucharest, Romania, June 5-8, 2006. Springer, Dordrecht, The Netherlands. pp. 273-286. ISBN: 978-1-4020-5819-6.

(2005) Navarre, N., Palanque, P., Schyn, A., Winckler, M., Nedel, L., Freitas, C. A Formal Description of Multimodal Interaction Techniques for Immersive Virtual Reality Applications. In Proceedings of TC13-IFIP Conference INTERACT 2005, Rome, Italy (12th-16th September 2005), Springer LNCS Volume 3585, 2005, pp 170-183.

(2004) Winckler, M., Freitas, C. M. D. S., Palanque, P. Tasks and Scenario-based Evaluation of Information Visualization Techniques. In Proceedings of the 3rd International Workshop on Task Models and Diagrams for User Interface Design - TAMODIA 2004, Prague, Czech Republic, 15-16 November 2004. ACM, New York, NY, USA, pp. 165-172. DOI=10.1145/1045446.1045475

Winckler, M., Barboni, E., Farenc, C., Palanque, P. SWCEditor: a Model-Based Tool for Interactive Modelling of Web Navigation. International Conference on Computer-Aided Design of User Interface - CADUI'2004, Funchal, Portugal, 13-16 January 2004.

(2003) Winckler, M., Palanque, P., Farenc, C., Pimenta, M. Who does what with whom in Web Development? 2nd International Workshop on Task Models and Diagrams for User Interface Design - TAMODIA'2003, Heraklion, Greece, June 2003.

Palanque, P., Bastide, R., Winckler, M. Automatic Generation of Interactive Systems: Why A Task Model is not Enough. 10th International Conference on Human-Computer Interaction – HCI International'2003, Heraklion, Greece, June 2003.

Winckler, M., Palanque, P. StateWebCharts: a formal description technique dedicated to navigation modelling of Web applications. In Proceedings of the International Workshop on Design, Specification and Verification of Interactive Systems - DSVIS'2003, Funchal, Portugal, June 2003, Springer LNCS Volume 2844, 2003, pp 61-76. Also available at: http://dx.doi.org/10.1007/978-3-540-39929-2_5

Freitas, C., Cava, R., Winckler, M., Palanque, P. Synergistic Use of Visualisation Technique and Web Navigation Model for Information Space Exploration. 10th International Conference on Human-Computer Interaction – HCI International'2003, Heraklion, Greece, June 2003.

(2002) Winckler, M., Palanque, P., Farenc, C., Pimenta, M. Task-Based Assessment of Web Navigation Design. First International Workshop in Task Models and Diagrams for User Interface Design - TAMODIA'2002, Bucarest, Romania, 18-19 July 2002.

(2001) Winckler, M., Pimenta, M., Palanque, P., Farenc, C. Usability Evaluation Methods: What is still missing for the Web? 8th International Conference on HCI International, New Orleans, USA, 5-10 August 2001.

Farenc, C., Palanque, P., Bastien, C., Scapin, D., Winckler, M. Towards a General Guidance and Support Tool for Usability Optimization. International Conference on Universal Access in Human-Computer – UAHCI'2001, New Orleans, USA, 5-10 August 2001.

Short papers in conferences with peer reviewing (11)

(2013) Firmenich, S., Rossi, G., Winckler, M. A Domain Specific Language for Orchestrating User Tasks whilst Navigation Web sites. In Proceedings of ICWE 2013, Aalborg, Denmark, July 8-12, 2013. Springer LNCS Vol. 7977, p. 224-232.

(2011) Firmenich, S., Rossi, G., Gordillo, S., Winckler, M., A flexible architecture for Client-Side Adaptation. In extended proc. of International Conference on Web Engineering (ICWE 2011), Paphos, Cyprus, June 20-24 2011, Springer, LNCS Volume 7059, 2012, pp 327-331.

Firmenich, S., Winckler, M., Rossi, G. A Tool Support for Web Applications Adaptation using Navigation History. In Proc. of IFIP TC13 Conference on Human-Computer Interaction (INTERACT 2011), Lisbon, Portugal, September 5-9 2011. Springer, LNCS 6948, pages 337-344.

(2010) Firmenich, S., Gordillo, S., Rossi, G., Winckler, M. Client-Side Adaptation: An Approach Based in Reutilization Using Transversal Models. In Proceedings of the International Conference on Web Engineering (ICWE 2010), Vienna, Austria, July 5-7, 2010, Springer LNCS Vol. 6385, pp. 566-570.

(2009) Bernhaupt, R., Winckler, M., Pontico, F. Are User Interface Pattern Languages Usable? A Report from the Trenches. In Proceedings of the 12th IFIP TC 13 Conference on Human-Computer Interaction (INTERACT 2009). T. Gross et al. (Eds.): INTERACT 2009, Part II, LNCS 5727, pp. 542–545, 2009.

(2008) Guerrero, J., Vanderdonckt, J., González, J.M., Winckler, M. Towards a Library of Workflow User Interface Patterns. International Workshop on the Design, Verification and Specification of Interactive Systems (DSVIS'2008). Kingston, Ontario, Canada, July 16-18 2008. Springer LNCS 5136, pp. 96-101.

Palanque, P., Winckler, M., Bernhaupt, R., Alberdi, E., Strigini, L., Ryan, P. AROVE-v: Assessing the resilience of open verifiable E-voting systems. 7th European Dependable Computing Conference (EDCC 2008). Kaunas, Lithuania. May 7-9 2008. (fast abstract).

(2004) Xiong, J., Farenc, C., Winckler, M. *Vérification de règles ergonomiques sur un modèle de navigation des application Web*. In proceedings of the 16th French-speaking conference on human computer-interaction, Namur, Belgium, September 2004. (short paper in French)

Pontico, F., Farenc, C., Winckler, M. *Une architecture de dialogue basée sur un modèle pour les applications Web*. In proceedings of the 16th French-speaking conference on human computer-interaction, Namur, Belgium, September 2004. (short paper in French)

(2001) Winckler, M., Farenc, C., Palanque, P. Bastide, R. Designing Navigation for Web Interfaces. In Proceedings of the 15th *Conférence Annuel de l'AFIHM / HCI* Group - IHM-HCI'2001, Lille, France, 10-14 septembre 2001.

(1999) Winckler, M. A. A., Freitas, C. M. D. S., Lima, J. D. de. Remote Usability Testing: a Case Study. Annual conference of the Computer-Human Interaction Special Interest Group of the Human Factors and Ergonomics Society of Australia - OZCHI99', Wagga-Wagga, Australia, 28-30 November 1999.

Research papers at workshops with peer reviewing (28)

(2015) Firmenich, S., Bosetti, G., Rossi, G., Winckler, M. Flexible distribution of existing Web interfaces: an architecture involving developers and end-users. (to appear) DUI'2016 Workshop.

(2014) Silva, T., Hak, J. L., Winckler, M. A Review of Milestones in the History of GUI Prototyping Tools. Workshop on User Experience and User-Centered Development Processes 2015 (IFIP WG 13.2). p:1-12. University of Bamberg Press.

Winckler, M., Barboni, E., Carrere, C. Semantics of States and Transitions in statecharts-based markup languages: a comparative study between SWC and SCXML. Workshop on Engineering Interactive Computer Systems with SCXML 2014. p:28-32. Technical University of Darmstadt.

(2013) Forbrig, P., Zaki, M., Palanque, P., Winckler, M. Supportive User Interfaces and Task Migratability in Smart Environments. In 3rd Workshop on Distributed User Interfaces (DUI 2013), London, UK, June 24th 2013. ISBN-13: 978-84-616-4792-7, p. 42-45.

Albertos Marco, F., Penichet, V., Gallud, J., Winckler, M. Making Distributed User Interfaces Interruption-Resistant: A Model-Based Approach. In 3rd Workshop on Distributed User Interfaces (DUI 2013), London, UK, June 24th 2013. ISBN-13: 978-84-616-4792-7, p. 18-21.

Albertos Marco, F., Gallud, J., Penichet, V., Winckler. M., A Model-Based Approach for Supporting Offline Interaction with Web Sites Resilient to Interruptions. In Proc. of Current Trends in Web Engineering - ICWE 2013 International Workshops, Aalborg, Denmark, June 25th 2013, Jesper Kjeldskov, Michael Sheng (Eds.), Springer, LNCS 8295, pages 156-171.

(2012) Stein D'Agostini, C., Winckler, M. A Model-Based Approach for Supporting Aspect-Oriented Development of Personal Information Management Systems. In proceedings of the Workshop on Model-Driven and Agile Engineering for the Web held on July 25th 2012 in conjunction with ICWE 2012, Berlin, Germany. M. Grossniklaus , M. Wimmer (Eds.), Springer, LNCS Volume 7703, 2012, pp 26-40.

Winckler, M., Bach, C., Bernhaupt, R. Identifying User eXperiencing factors along the development process: A case study. In proceedings of the International Workshop on the Interplay between User Experience Evaluation and System Development (I-UxSED 2012) held on October 15th in conjunction with NORDICHI 2012, Copenhagen, Denmark. Law, E., Abrahão, S., Vermeeren, A.P.O.S., Hvannberg, E. T. (Eds.). Audio Visual Services - University of Leicester, pp. 37-42, october 2012. At: http://www.irit.fr/recherches/ICS/projects/twintide/upload/539.pdf.

(2011) Barboni, E., Martinie, C., Navarre, D., Palanque, P., Winckler, M. UsiXML Concrete Behaviour with a Formal Description Technique for Interactive Systems. In Proc. of the International Workshop on User Interface Description Languages (UIDL'2011), Lisbon, Portugal, September 5th 2011, in conjunction with the IFIP TC13 INTERACT conference.

Mbaki, E., Vanderndonckt, J., Winckler, M. Model-Driven Engineering of Dialogues for Multi-platform Graphical User Interfaces. In Proc. of the International Workshop on User Interface Description Languages (UIDL'2011), Lisbon, Portugal, September 5th 2011, in conjunction with the IFIP TC13 INTERACT conference.

Bach, C., Bernhaupt, R., Winckler, M. Mobile Incident Reporting in Urban Contexts: Towards the Identification of Emerging User Interface Patterns. In Proc. of the 5th IFIP's Working Group 13.2 Workshop on Software and Usability Engineering Cross-Pollination: Patterns, Usability and User Experience (PUX) to be held in Lisbon, Portugal, September 5th 2011, in conjunction with the IFIP TC13 INTERACT conference.

Palanque, P., Bernhaupt, R., Winckler, M. Usability Evaluation: Commonalities and Discrepancies in Games and Safety Critical Systems . In Proc. of 1stEuropean Workshop on HCI Design and Evaluation: The influence of domain. Limassol, Cyprus - April 8th, 2011. IRIT Press, Toulouse, France. ISBN: 978-2-917490-13-6 EAN: 9782917490136. pages 37-44.

(2010) Martinie, C., Ladry, J., Navarre, D., Palanque, P., Winckler, M. Embedding Requirements in Design Rationale to Deal Explicitely with User eXperience and Usability in an "intensive" Model-Based Development Approach. 5th International Workshop on Model Driven Development of Advanced User Interfaces (MDDAUI 2010) at ACM SIGCHI 2010, April 10, 2010, Atlanta, USA.

Winckler, M., Palanque, P. *Desafios para Programas de Mestrado em Interação Humano-Computador (IHC): a experiência do M2IHM*. In Proc. of the *Simpósio de Fatores Humanos em Sistemas Computacionais (IHC 2010)*, Belo Horizonte, October 05-08,2010, Vol. II, Sociedade Brasileira de Computação, ISSN 2178-7700, p. 71-74 (in Portuguese).

(2009) Winckler, M., Scapin, D., Pontico, F., Calvary, G., Serna, A. Profiling User Requirements for Multi-Target e-Government Applications: a case study In Proc. International Workshop on Design & Evaluation of e-Government Applications and Services (DEGAS 2009), Uppsala, Sweden, 24/08/2009, Vol. 492, CEUR Workshop Proceedings, p. 9-16, August 24, 2009. Available at http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-492/

(2008) González Calleros, J.M., Stanciulescu, A.,Vanderdonckt, J.,Delacre, J.P.,Winckler, M.,A Comparative Analysis of Tranformation Engines for User Interface Development. Proc. of the 4th International Workshop on Model-Driven Web Engineering (MDWE 2008), pp. 16-30. Toulouse, France, September 2008. CEUR Workshop Proceedings, ISSN 1613-0073. Also available at: http://ceur-ws.org/Vol-389

Xiong, J., Farenc, C., Winckler, M. An Ontology-Based Approach for Dealing with Web Guidelines. 2nd Int. Workshop on Web Usability and Accessibility (IWWUA), In conjuntion with the WISE' 2008. Auckland, New Zealand Sept. 1-4, 2008. Springer LNCS 5176, 132-141 pp.

Mbaki, E., Vanderdonckt, J., Winckler, M. Multi-level Dialog Modelling in Highly Interactive Web Interfaces. In 7th Int. Workshop on Web-Oriented Software Technologies (IWWOST'2008). New York, USA. July 14, 2008. In conjunction with ICWE'2008. ISBN 978-80-227-2899-7.

(2007) Winckler, M., Xiong, J., Noirhomme-Fraiture, M. Accessibility Legislation and Codes of Practice: an Accessibility Study of Web Sites of French and Belgium Local Administrations. 1st Int. Workshop on Design & Evaluation of e-Government Applications and Services (DEGAS'07). Rio de Janeiro, Brazil, Sept. 11th 2007. CEUR Workshop Proceedings, ISSN 1613-0073, online at http://ceur-ws.org/Vol-285.

Xiong, J., Farenc, C., Winckler, M. Analyzing Tool Support for Inspecting Accessibility Guidelines during the Development Process of Web Sites. In Proceedings of 1st Int. Workshop on Web Usability and Accessibility (IWWUA), in conjuntion with the 8th Int. Conference on Web Information Systems Engineering (WISE' 2007). Nancy, France, December 3-7, 2007. Springer LNCS 4832. pages 470-480.

(2006) Winckler, M., Xiong, J., Farenc, C. Extending Automated Guidelines Inspection with Ontology for User Interface of Web applications. In Proceedings of the 2nd International Workshop on Automated Specification and Verification of Web Systems Cyprus, November 19, 2006.

Pontico, F., Winckler, M. Citizens, Stakeholders and Designers: modeling for user diversity. In Proceedings of the International workshop on User involvement and representation in e-Government projects, in conjunction with NordiCHI'2006, Olso, Norway, October 15th 2006. Folstaf, A., Artman, H. Krogstie, J. (eds.) SINTEF report A314, pp. 59-64. ISBN: 82-14-04040-X.

Winckler, M., Pontico, F. A Model-Driven Architecture for Logging Navigation. In Proceedings of the International Workshop on Logging Traces of Web Activity: The Mechanics of Data Collection. Co-located with 15th International World Wide Web Conference (WWW2006). Edinburgh, Scotland, Tuesday, May 23-27, 2006.

(2005) Winckler, M., Vanderdonckt, J. Towards a User-Centered Design of Web Applications based on a Task Model. In Proceedings of 5th International Workshop on Web-Oriented Software Technologies (IWWOST'2005). Porto, Portugal, 12th-13th June 2005.

Navarre, D., Palanque, P., Winckler, M. What User Interface Tools are needed for Safety Critical Interactive Systems? ACM CHI 2005 workshop on the Future of UI Design Tools! Portland, USA, 3rd-4th April 2005

Winckler, M., Barboni, E., Farenc, C., Palanque, P. What Kind of Verification of Formal Navigation Modelling for Reliable and Usable Web Applications? In Proceedings of the 1st International Workshop on Automated Specification and Verification of Web Sites (WWV'2005), Valencia, Spain, 14-15 mars 2005. p. 33-36.

(2002) Winckler, M., Farenc, C., Palanque, P. Automatic Evaluation for the Web: How Improve Navigation Guidelines. In Proceedings of the International Workshop on Automated Testing in ACM Conference on Computer-Human Interaction - CHI'2002, Minneapolis, United States of America, 21-22 April 2002.

(2001) Freitas, C.M.D.S., Winckler, M.A., Luzzardi, P.R.G., Nedel, L.P. and Pimenta, M. Usability issues in information visualization applications. In Proceedings of the International Workshop on Data Usability, Wagening, Hollande, 19-20 November 2001.

Full research papers in national conferences with peer reviewing (15)

(2013) Stein D'Agostini, C., Cava, R., Dorneles, C., Firmenich, S., Freitas, C., Palanque, P., Winckler, M. *Proposta de um Framework para Visualização de Dados Agregados por Similaridade para Auxiliar Consultas durante a Navegação na Web*. In Proc. Simpósio Brasileiro de Fatores Humanos em Sistemas Computacionais (IHC 2013), Manaus, Brazil, October 08-10, 2013, Sociedade Brasileira de Computação (SBC), 2013. (in Portuguese)

(2012) Martinie, C., Palanque, P., Navarre, D., Barboni, E., Winckler, M. *Un processus de développement outillé pour l'exploitation systématique des bénéfices offerts par une conception des systèmes interactifs centrée tâches*. IHM 2012, Biarritz, France, October 17-19th 2012. (in French).

(2011) Bim, S. A., Silveira, M., Prates, R., Winckler, M. *Ensino de IHC – Atualizando as Discussões sobre a Experiência Brasileira*. In Proceedings of the *XIX Workshop sobre Educação em Computação* – WEI'2011, Natal, Brazil, 19-22 July 2011. (in Portuguese)

(2010) Ladry, J.F., Palanque, P., Navarre, D., Barboni, E., Winckler, M. *Une approche à base de modèles pour l'ingénierie logicielle de techniques d'interaction.*. In. Proceedings of the *Conférence Francophone sur l'Interaction Homme-Machine* - IHM 2010, Luxembourg, September 20-23, 2010, ACM, New York, NY, USA, 81-88. DOI=10.1145/1941007.1941019. (in French).

(2009) Palanque, P., Ladry, J-F., Barboni, E., Navarre, D., Winckler, M. *Une Approche Formelle pour l'Evaluation de la Tolérance aux Interruptions des Système Interactifs*. In Proceedings of *the Conférence Francophone sur l'Interaction Homme-Machine* (IHM 2009), Grenoble, August 13-16, 2009, ACM DL, p. 141-150. (in French)

(2005) Winckler, M., Farenc, C., Barboni, E., Pontico, F. *Modélisation orientée tâche de la navigation d'un application Web: catalogue des thèses de l'AFIHM*. In Proceedings of the 17th Conférence Francophone sur l'Interaction Homme-Machine, Toulouse, France, October 27-30, 2005, ACM, p. 91-99. (in French)

(2002) Freitas, C. M. D. S., Luzzardi, P., Cava, R. A., Winckler, M., Pimenta, M. S., Nedel, L. Evaluating Usability of Information VisualizationTechniques. In Proceedings of the V Simpósio sobre Fatores Humanos em Sistemas Computacionais – IHC'2002, Fortaleza, Brazil, 7-10 October 2002. (in English)

Winckler, M., Farenc, C., Palanque, P. *Une démarche structurée pour la conception et l'évaluation d'applications Web par l'exploitation synergique des modèles de tâche et de navigation*. In Proceedings of the 14th Conférence Francophone sur l'Interaction Homme-Machine - IHM2002, Poitiers, France, November 26-29, 2002. (in French)

(2001) Winckler, M., Farenc, C., Palanque, P., Pimenta, M. *Avaliação da Navegação de Interfaces Web a partir de Modelos*. IV Simpósio sobre Fatores Humanos em Sistemas Computacionais – IHC'2001, Florianópolis, Brazil, 14-19 October 2001. (in Portuguese)

(2000) Borges, C. M., Winckler, M., Basso, K. *Considerações sobre o Uso de Cores em Interfaces WWW*. III Workshop sobre Fatores Humanos em Sistemas Computacionais. Gramado – RS,Brazil, 18-20 October 2000. (in Portuguese)

Winckler, M., Nemetz, F., Lima, J. V. de. *Interação entre aprendiz e computador: métodos para desenvolvimento e avaliação de interfaces.* In: Tarouco, Liane (Org.). Tecnologia Digital na Educação. p.7-33. Porto Alegre, Brazil. 2000. (in Portuguese)

Vicari, R. M., Fritsch, E. F., Kruger, S. E., Flores, L. V., Winckler, M. A. A. *Tutores inteligentes: uma aplicação teórica e prática*. In: Tarouco, Liane (Org.). Tecnologia Digital na Educação. Porto Alegre, Brazil. 2000. (in Portuguese)

(1998) Winckler, M., Nemetz, F., Lima, J. V. de. *Estudo de caso da aplicação do método de avaliação heurística em um projeto multidisciplinar.* I Workshop sobre Fatores Humanos em Sistemas Computacionais IHC'98, Maringá, Brazil, pp. 1-9, 12-13 October 1998. (in Portuguese)

Winckler, M., Lima, J. V. de, Freitas, C. M. D.S. *Proposta de uma Metodologia Interativa de Identificação de Perfis de Usuários e Problemas de Usabilidade na WWW*. Taller International de Software Educativo - TISE'98, Santiago, Chile, 02-04 December 1998. (in Portuguese)

(1997) Winckler, M., Padilha, E. G., Lima, J. V. de. O Uso de uma Interface WWW para o ConVer: uma Ferramenta de Apoio ao Ensino da Conjugação Verbal da Língua Portuguesa. Taller Internacional de Software Educativo - TISE'97, Santiago, Chile, 1-3 December 1997. (in Portuguese)

Posters and position papers (14)

(2016) Rocha Silva, T., Winckler, M. Towards Automated Requirements Checking throughout Development Processes of Interactive Systems. In 2nd Workshop on Continuous Requirements Engineering (CRE'16). March 14, 2016 · Gothenburg, Sweden.

(2015) Martinie, M., Palanque, P., Winckler, M., Bernhaupt, R. AUTOM AT ICS: Research activities on Automation. International Conference on Application and Theory of Automation in Command and Control Systems 2015 (ATACCS).

(2013) Lazar, J., Barbosa, S. J.D., Gulliksen, J., McEwan, T., Martínez, L., Palanque, P., Prates, R., Tsai, J., Winckler, M., Wulf, V. Engaging the human-computer interaction community with public policymaking internationally. CHI Extended Abstracts 2013: 3279-3282.

(2012) Winckler, M., Bernhaupt, R., Bach, C., Gatellier, B. Challenges for the Gamification of Incident Reporting Systems. In extended proceedings of the 4th International Conference on Fun and Games, Toulouse, France, September 4-6, 2012, IRIT Press, pp. 24-26. ISBN: 978-2-917490-21.

(2011) Maciel, C., Furtado, E., Winckler, M., Selbach, M. S., Prates, R. O. Overview of the Brazilian Computer Society's Council for Human-Computer Interaction (CEIHC). In Proc. of the IFIP TC13 INTERACT'2011, Lisbon, Portugal, September 5-9, 2011. Springer, LNCS 6949, p. 679-680.

(2009) Vo, D-B, Winckler, M. *PIAFF: un outil d'aide à la saisie d'informations*. Dans : Interaction Homme-Machine (IHM 2009), Grenoble, August 13-16th 2009, ACM DL, p. 355-358.

(2006) Taneva, S., Palanque, P., Basnyat, S., Winckler, M., Law, E. Clinical Application Design: Task Modeling with Failure in Mind 11th World Congress on Internet in Medicine. MedNet 2006. Poster Presentation. Toronto, Canada, October 15-18 2006.

Taneva, S., Palanque, P., Basnyat, S., Winckler, M., Law, E. Analysis of Communication Breakdowns for eHealth Systems Design 11Th World Congress on Internet in Medicine. Toronto, Canada, October 13-20 2006.

(2002) Freitas, C. M. D. S., Luzzardi, P. R. G., Cava, R. A., Pimenta, M., Winckler, M., Nedel, L. P. Usability issues in the evaluation of information visualization techniques. In. Advanced Visual Interfaces - AVI'2002, Trente, Italie, May 22-24, 2002.

(2000) Winckler, M., Freitas, C. M. D. S., Lima, J. D. De. 2000. Usability remote evaluation for WWW. In CHI '00 Extended Abstracts on Human Factors in Computing Systems (CHI EA '00). ACM, New York, NY, USA, 131-132. DOI=10.1145/633292.633367

(1999) Gralewaski, D., Winckler, M., Indrusiak, L. S., Reis, R. A. Jale - Java Layout Editor. Seminário Interno da Microeletrônica – SIM'99, July 9-10, 1999, Pelotas - RS, Brazil.

(1998) Padilha , E. G., Winckler, M., Viccari, R.V. ConVer: a Conjugação Verbal. Rede Ibero-Americana em Informática na Educação - RIBIE'98, Brasília - DF, Brazil, 20-23 October 1998.

(1997) Winckler, M., Padilha, E. G., Lima, J. V. de. Experiência do Uso de Uma Ferramenta WWW para Ensino Interativo da Língua Portuguesa. Simpósio Brasileiro de Informática na Educação - SBIE'97, São José dos Campos - SP, Brazil, 18-20 November 1997.
Nemetz, F., Winckler M., de Lima, J. V. Evaluating Evaluation Methods for Hypermedia Applications. World Conference on Educational Multimedia, Hypermedia & Telecommunication - ED-MEDIA & ED-TELECOM 97, Calgary – Canada, 14-19 June 1997.

Editorial work

Edited conference proceedings (19)

(2015) Proceedings of the 15th IFIP TC13 International Conference Human-Computer Interaction (INTERACT 2015). Bamberg, Germany, September 14-18, 2015. Abascal, J., Barbosa, S., Fetter, M., Gross, T., Palanque, P., Winckler, M. (Eds.). Four volumes Springer. LNCS 9296 (Part I), LNCS 9297 (Part II), LNC 9298 (Part III) and LNCS 9299 (Part IV).

Casteleyn, S., Rossi, G., Winckler, M. (eds) Engineering the Web for Users, Developers and the Crowd. Journal of Web Engineering vol:5&614. Rinton Press.

Proceedings of the 5th International Conference on Application and Theory of Automation in Command and Control Systems (ATACCS 2015). Michael Feary, Thomas Feuerle, Cristina

Gonzalez Rechea, Francisco Javier Saez, Chris Johnson, Célia Martinie, Philippe Palanque, Alberto Pasquini, Pim van Leeuwen, and Marco Winckler (Eds.). September 30th - October 2nd, 2015, Toulouse, France. ACM, New York, NY, USA. ACM ISBN: 978-1-4503-3562-1. 167 pages. Available at: http://dl.acm.org/citation.cfm?id=2899361

(2013) Proceedings of the 14th IFIP TC13 International Conference Human-Computer Interaction (INTERACT 2013). Cape Town, South Africa, September 8-12, 2013. Kotzé, P., Marsden, G., Lindgaard, G., Wesson, J., Winckler, M. (Eds.). Four volumes Springer. LNCS 8117 (Part I), LNCS 8118 (Part II), LNC 8119 (Part III) and LNCS 8120 (Part IV). Available at SpringerLink: http://www.springer.com/computer/hci/book/978-3-642-40482-5

Proceedings of the 3rd International Conference on Application and Theory of Automation in Command and Control Systems (ATACCS'2013). May 28-30, 2013, Naples, Italy. Brat, G., Garcia, E., Moccia, A., Palanque, P., Pasquini, A., Saez, F., Winckler, M. (eds.). IRIT PRESS. 207 pages. ISBN: 978-2-917490-24-2. Also available at the ACM Digital Library: http://dl.acm.org/citation.cfm?id=2494493

(2012) Proceedings of the 4th International Conference on Human-Centered Software Engineering (HCSE'2012) Toulouse, France, 29-31 October 2012. Winckler, M., Forbrig, P., Bernhaupt, R. (eds.) Springer LNCS 7623. Available at http://dx.doi.org/10.1007/978-3-642-34347-6.

Proceedings of the 2nd International Conference on Application and Theory of Automation in Command and Control Systems. London (ATACCS'2012), UK, May 29-31, 2012. Garcia, E., Johnson, C., Ochieng, W. O., Palanque, P., Saez, F., Vilaplana, M. A., Winckler, M. (eds.) IRIT PRESS. 237 pages. ISBN: 978-2-917490-20-4. Also available at the ACM Digital Library: http://dl.acm.org/citation.cfm?id=2325676

Proceedings of the 4th International Conference on Fun and Games. Toulouse, France, 4-6 September 2012. Bernhaupt, R., Isbister, K., Mueller, F., Winckler, M. (eds.). ACM, ICPS ACM. ISBN: 978-1-4503-1570-8. 136 pages. Also available at the ACM Digital Library: http://dl.acm.org/citation.cfm?id=2367616

(2011) Proceedings of the 13th IFIP TC 13 International Conference Human-Computer Interaction (INTERACT 2011). Lisbon, Portugal, September 5-9, 2011, Campos, P., Graham, N., Jorge, J., Nunes, N., Palanque, P., Winckler, M. (eds.). Four volumes Springer LNCS 6946 (Part I), LNCS 6947 (Part II), LNC 6948 (Part III) and LNCS 6949 (Part IV). Available at SpringerLink: http://www.springer.com/computer/hci/book/978-3-642-23773-7

Proceedings of the 1st International Conference on Application and Theory of Automation in Command and Control Systems (ATACCS'2011). May 26-27, 2011. Barcelona, Spain. Editors: Eduardo Garcia, Arnab Majumdar, Philippe Palanque, Alberto Pasquini, Francisco Javier Saez and Marco Winckler. 188 pages. IRIT Press. ISBN: 978-2-917490-14-3. EAN: 9782917490143. Also available at ACM Digital Library: http://dl.acm.org/citation.cfm?id=2248467

(2010) Extended Proceedings on *Tópicos em Sistemas Colaborativos, Interativos, Multimídia, Web e Banco de Dados*. Pereira, A.C.M., Pappa, G.L., Winckler, M., Gomes, R.L. (eds.). Belo Horizonte, Brazil, October 5-8 2010. Brazilian Computing Society/Sociedade Brasileira de Computação (SBC). 276 p. ISBN: 857669249-2. (in Portuguese) Also available online at: http://www.irit.fr/~Marco.Winckler/SWIB2010-minicursos.pdf

Proceedings of the 22nd *Conférence Francophone sur l'Interaction Homme-Machine, IHM 2010*. David, B., Noirhomme-Fraiture, M., Tricot, A., Koenig, V., Vagner, A., Winckler, M. (Eds.) Luxembourg, LU, 20-24 September, 2010. ACM SIGCHI Series. ISBN: 978-3-642-11749-7. 248 pages. (in French). Available at ACM DL: http://portal.acm.org/citation.cfm?id=1941007

(2009) Proceedings of the 7th IFIP WG 13.5 Working Conference Human Error, Safety and Systems Development, HESSD 2009. Palanque, P., Vanderdonckt, J., Winckler, M. (Eds.) Brussels, Belgium, September 23-25, 2009. Springer LNCS 5962. ISBN: 978-3-642-11749-7. Also available online at SpringerLink: http://www.springer.com/computer/hci/book/978-3-642-11749-7

Proceedings of the 12th IFIP TC13 INTERACT'2009. Gross, T., Gulliksen, J., Kotzé, P., Oestreicher, L., Palanque, P., Prates, R., Winckler, M. (eds.) Uppsala, Sweden, Uppsala, Sweden, 24-28 August 2009. Two volumes Springer LNCS 5726 (Part I) and LNCS 5727 (Part II). Also availabel at SpringerLink : http://www.springer.com/computer/hci/book/978-3-642-03654-5

Proceedings of the COST294-MAUSE Closing Conference Proceedings : Maturation of Usability Evaluation Methods: Retrospect and Prospect (Final Reports of COST294-MAUSE Working

Groups). Law, E., Scapin, D., Cockton, G., Springett, M., Stary, C., Winckler, M. (Eds.) Brindisi, Italy, March 19-21, 2009. IRIT Press, Toulouse, France. ISBN :978-2-917490-06-8. 188 pages. Also available online at: http://www.irit.fr/~Marco.Winckler/cost294-proceedings-complet.pdf

(2008) Proceedings of the 8th Brazilian Symposium on Human Factors on Computing Systems. Filgueiras, L., Winckler, M. (Eds.). ACM International Conference Proceeding Series, Vol. 378. Available at ACM Digital Library at http://dl.acm.org/citation.cfm?id=1497470. 357 pages. 2008. ISBN: 978-85-7669-203-4.

(2007) Proceedings of the International Workshop on TASk MOdels and DIAgrams (TAMODIA'2007) Winckler, M., Johnson, H., Palanque, P. (Eds.) Toulouse, France, 7-9 November 2007. Springer LNCS 4849. ISBN 978-3-540-77221-7. Also available online at SpringerLink at http://link.springer.com/book/10.1007/978-3-540-70816-2/

(2006) Proceedings og the 6th Brazilian Symposium on Human Factors on Computing Systems (IHC 2006) – extended proceedings. Prates, R. O., de Assis, A. S. F. R., Anacleto, J. C., Winckler, M. A. A., Betiol, A. H., Filgueiras, L. V. L., Silveira, M. S., da Silva, E. J. (Eds.). Brazilain Computing Society/Sociedade Brasileira da Computação (SBC). 2006. ISBN:85-7669-099-3. Also available online at: http://www.irit.fr/~Marco.Winckler/IHC2006_AnaisEstendidos-final.pdf

(2004) Proceedings of 3rd International Workshop on TAsk MOdels and DIAgrams for user interface design (TAMODIA2004). Palanque, P., Slavik, P. and Winckler M. (Eds.) November 15-16 2004, Prague, Czech Republic. ACM Digital Library, ISBN: 1-59593-000-0, Softcover. Also available online at: http://www.irit.fr/~Marco.Winckler/TAMODIA2004-Proceedings-FINAL.pdf

Edited electronic workshop proceedings (3)

(2012) Bernhaupt, R., Isbister, K., Mueller, F., Winckler, M. (eds.) Extended proceedings of the 4thInternational Conference on Fun and Games. Toulouse, France, 4-6 September 2012, IRIT Press, ISBN: 978-2-917490-21-1. 142 pages. Also available online at: http://www.irit.fr/~Marco.Winckler/FnG2012-EXTENDED-proceedings.pdf

(2011) Peter Forbrig, Regina Bernhaupt, Marco Winckler & Janet Wesson (eds.) 5th Workshop on Software and Usability Engineering Cross-Pollination: Patterns, Usability and User Experience (PUX 2011). Held in conjunction with IFIP TC13 INTERACT'2011, Lisboa, Portugal, September 6th 2011. Available online at: http://www.irit.fr/~Marco.Winckler/PUX2011-proceedings.pdf

Bim, S. A., Silveira, M. S., Prates, R. O., Winckler, M. (eds.) Workshop sobre Ensino de Interação Humano-Computador (WEIHC). Em conjunto com o Simpósio de Fatores Humanos em Sistemas Computacionais e a Conferência Latino-Americana de Interação Humano-Computador - IHC-CLIHC 2011. Porto de Galinhas, Pernambuco, 25 de outubro 2011, SBC Press, available online at: http://www.irit.fr/recherches/ICS/events/conferences/weihc/WEIHC2011-proceedings.pdf

(2009) Marco Winckler, Monique Noirhome-Fraiture, Dominique Scapin, Gaelle Calvary & Audrey Serna Proceedings of the 2nd International Workshop on Design & Evaluation of e-Government Applications and Services (DEGAS'09), held in conjunction with IFIP TC13 INTERACT'2009. Uppsala, Sweden, August 24th, 2009. CEUR Workshop Proceedings, Vol. 492, ISSN 1613-0073, available online at: http://ceur-ws.org/Vol-492/.

(2007) Marco Winckler & Monique Noirhome-Fraiture (Eds.) Proceedings of the 1st International Workshop on Design & Evaluation of e-Government Applications and Services (DEGAS'07), held in conjunction with IFIP TC13 INTERACT'2007. Rio de Janeiro, Brazil, September 11th, 2007. CEUR Workshop Proceedings, Vol. 285, ISSN 1613-0073, available online at: http://ceur-ws.org/Vol-285.

Oscar Mayora, Jean Vanderdonckt & Marco Winckler (Eds.) IFIP TC13 INTERACT'2007 extended Proceedings (Workshops). Rio de Janeiro, Brazil, September 10th-14th, 2007. Available online at: http://www.irit.fr/~Marco.Winckler/interact2007-extended-proceedings-final.pdf

Research and cooperation projects

Ongoing projects

(2016-2018) **Web Augmentation Methods for Adapting Web sites for supporting Opportunistic User Requirements**

Summary: The main goal of this project is to investigate Web augmentation methods, techniques and tools for supporting the adaptation of Web sites to cope with opportunistic user's requirements (i.e. associated to infrequent and short lasting tasks). We aim at exploring Web augmentation methods and tools for empowering end-users to make volatile adaptations of Web content that don't require any modification of Web applications at the server-side. The investigation of Web augmentation methods has several implications of both theoretical and practical importance for development of adaptive software in the field of Software Engineering and Requirements Engineering. Moreover, such study also contributes to the research on innovative technologies for Web Engineering and to the research in the field of Human-Computer Interaction, where user-centered design methods are a cornerstone. The specific goals of the project include: investigate users requirements for adapting Web applications; to investigate the user experience with Web augmentation tools, methods and functions; and to investigate the use of Web augmentation tools by end-users.

Partners: ICS-IRIT, University Paul Sabatier (Toulouse, France), *Laboratorio de Investigación y Formación en Informática Avanzada (LIFIA) – Facultad de Informática – Universidad Nacional de La Plata* (La Plata, Argentina), *Departamento de Ingeniería Electrónica e Informática - Universidad Católica "Nuestra Señora de la Asunción"* (Asunción, Paraguay).

IRIT participants: Philippe Palanque, Regina Bernhaupt, Célia Martinie, Jean Luk Hak & Marco Winckler (coordinator).

Funding: STIC AmSud.

Duration: 24 months / January 2016 – December 2017.

Former projects

(2012-2015) **VIDAS: Visualization and Interaction with Data Aggregated by Similarity functions**

Summary: This project is motivated by the huge amount of information available on the Web nowadays, where resolving ambiguities becomes a daily task for most users. In the last decade, several specialized tools have been created upon similarity functions that, given a keyword and a context, determine the degree of similarity (or probability) that information in a dataset corresponds to the user's query. Quite often such tools are meant for experts and require training and knowledge on the application domain to be used. The project VIDAS is dedicated to the study of methods and tools allowing end-users, not only domain experts, to visualize and then interact with uncertain data and/or data aggregated by similarity functions. This project implies a joint research on the fields of Human-Computer Interaction, Data Bases and Information Visualization.

Partners: ICS-IRIT, University Paul Sabatier (Toulouse, France), *Universidade Federal do Rio Grande do Sul* – UFRGS (Porto Alegre, Brazil) and *Universidade Federal de Santa Catarina – UFSC* (Florianópolis, Brazil).

IRIT participants: Philippe Palanque, Regina Bernhaupt & Marco Winckler (coordinator).

Funding: CAPES/COFECUB N. 735-12. ICS-IRIT budget: 40 K€.

Duration: 48 months / January 2012 – December 2015.

(2010-2014) **TWINTIDE: Towards the Integration of Transectorial IT Design and Evaluation**

Summary: TwinTide is a COST Action which aims at providing harmonization and leadership on design and evaluation (D&E) methods among HCI researchers and professionals working across diverse sectors and application domains. The underlying motivation for this project is to bring together a broad experience on D&E methods which ultimately will enable the comparison of methods, their applications, and potential for transferability of both established and novel D&E approaches. The project is organized around collaborative activities held by working groups in open workshops that are aimed at facilitating the production of a generic D&E method selection and application framework and scientific publications reaching the wider research community. This action also provides systematic training and networking opportunities such as short term scientific missions (STSMs) and training schools.

Partners: Representative from 24 UE countries (Austria, Belgium, Cyprus, Denmark, Iceland, Ireland, Italy, Finland, France, Germany, Greece, Latvia, Macedonia, Netherlands, Norway, Poland, Portugal, Romania, Serbia, Slovenia, Spain, Sweden, Switzerland, Turkey and UK).

IRIT participants: Philippe Palanque & Marco Winckler (dissemination coordinators).

Funding: European Science Foundation (ESF) – COST Action 0904.

Duration: 48 months / January 2010 – January 2014.

(2010-2013) **PIMI: Personal Information Management through Internet**.

Summary: The project PIMI is concerned by the definition of an environment for deploying Personal Information Management System (PIMS) and the subsequent integration of personal data with third-party applications over the Web platform. The ultimate goal of PIMI is to provide users with easy access and management facilities for interacting with personal data (ex. addresses, bank account information, identification documents, etc.) that are often required whilst accessing diverse types of Web applications (ex. e-commerce Web sites, administrative and governmental Web sites, etc.). Among the challenges, this project focuses the deployment of a cross-platform application allowing users to access their personal data everywhere. Moreover, we are concerned by mechanisms providing users with full control during the operations of data sharing between their PIMS and third-party Web applications. Other scientific questions treated by the partners of PIMI include user perception of trust on personal data, quality of services, service composition, and the flexible organization of personal information spaces over the Web platform.

Partners: GENIGRAPH, INRIA Rocquencourt, IRIT, IT (Institut Telecom), LRI, Montimage, and IUT de Tarbes.

IRIT participants: Marco Winckler & Andreas Herzig.

Funding: National (French) Project ANR (French National Agency for the Research). Budget ICS-IRIT: 135K€ / overall project budget: 1100 K€.

Duration: 36 months / January 2010 – December 2013.

(2010-2015) **HALA!: Higher Automation Levels in Automation!**

Summary: HALA! Network is a Research Network established within the framework of SESAR WP-E to spearhead long term and innovative research in Air Traffic Management (ATM) in pursuit of the SESAR 2020 vision and beyond. HALA! is the acronym for "Higher Automation Level in ATM" which is also the focus of the network. The privileged application domain is ATM but some aspects of automation in other kinds of application (e.g. games, web, etc) will be also explored in the context of the conference ATACCS (Application and Theory of Automation in Command and Control Systems – http://www.ataccs.org/) which is organized as part of the HALA! project.

Partners: Universidad Politécnica de Madrid (UPM), Kungliga Tekniska Högskolan – KTH Stockholm, CRIDA, Boeing Research & Technology Europe S.L, DEEP BLUE, EADS Deutschland, GMVIS SKYSOFT S.A., The Foundation Stichting Nationaal Lucht- en Ruimtevaartlaboratorium (NLR), PILDo LABS, TU Dresden, Université Paul Sabatier – Toulouse III (UPS), Imperial College of Science, TU BRAUNSCHWEIG.

IRIT participants: Philippe Palanque (resp.), Marco Winckler, David Navarre & Célia Martinie.

Funding: EUROCONTROL Research Network. Budget ICS-IRIT (for 2011 only): 10K€ / Overall project budget (for 2011 only): 200 K€.

Duration: 36 months / Sept. 2010 – August 2013.

(2010-2013) **Ubiloop: User Interfaces for Reporting Critical Incidents in Urban Contexts of Use**

Summary: The project Ubiloop was focused on the development of innovative methods for designing and building incident reporting system allowing citizens to report to local administrations problems they might found in their neighborhood (ex. broken street lamps, graffiti, etc.). In our working scenario, citizens should be able to use diverse kind of devices (mainly mobile phones) to inform to local administrations the occurrence of incidents that need immediate care. The goals associated to this project imply activities of research and development covering in the following areas: engineering of interactive systems, human-interaction in mobile contexts, social aspects related to incident reporting and risk management. The neighborhood association "Croix de Pierre" of Toulouse contributed to this project consortium acting as end users for the scenarios proposed.

Partners: ICS-IRIT, Institute of Political Studies (IEP) of Toulouse, Génigraph.

IRIT participants: Marco Winckler (scientific coordinator) & Regina Bernhaup.

Funding: *Fonds Européen de Développement Régional* (FEDER), Région Midi-Pyrénnées. Budget ICS-IRIT: 100K€ / Overall project budget: 400 K€.

Duration: 24 months / December 2010 – April 2013.

(2008-2011) **MyCitizSpace: Personalize Citizens Space / *Espace Citoyen Personnalisable***

Summary: This project was focused on development of methods and tools for building Web applications supporting administrative procedures (the so-called e-procedures) such as online tax declaration, VISA applications, request of passport and ID cards, etc. Such Web applications are aimed to simplify users' tasks and to improve the communication between citizens and administrations by replacing papers forms by electronic Web forms. One of the outcomes of this project was to propose a model-based approach that supports the description of complex workflows underlying the expected behavior for administrative e-procedures. Such models are then used to control the user navigation among the different pages of the Web application, so that it was possible to prove by construction that the final application behaved as specified in the models. In this project we also investigated methods for supporting universal access to e-procedures by the means of multi-platform user interfaces.

Partners: INRIA Rocquencourt, LIG-University Joseph Fourrier, Silcor, Génigraph, University Paul Sabatier - IRIT.

IRIT participants: Marco Winckler (scientific coordinator), Philippe Palanque & Christelle Farenc.

Funding: National (French) Project ANR (French National Agency for the Research). Budget ICS-IRIT of 60 K €/ overall project budget of 650 K€.

Duration: 36 months / January 2008 – July 2011.

(2009-2010) **MEDIAWeb: MEDIA-Web: Methods for Evaluation and Design of rich Internet Applications (RIAs)**

Summary: The main focus of the MEDIAWeb project is the investigation of models, notations and tools to assist designers during the development of Rich Internet Applications (RIAs). This is an international cooperation project that only funded travel expenses of researchers (usually 2 travels per year). Despite the fact this project does not fund the research it supported the activities of co-supervision of the PhD student, to name Sergio Firmenich who defended his thesis on April 2013.

Partners: ICS-IRIT (Toulouse, France), LIFIA (La Plata, Argentina)

IRIT participants: Marco Winckler (coordinator), Philippe Palanque & Christelle Farenc

Funding: SECYT / CNRS-INRIA. ICS-IRIT budget: 5 K€.

Duration: 24 months / January 2009 – December 2010

(2006-2008) **ResIST: Resilience for IST** (Network of Excellence framework VI)

Summary: ReSIST was a Network of Excellence that addressed the strategic objective "Towards a global dependability and security framework" of the European Union Work Program, and responds to the stated "need for resilience, self-healing, dynamic content and volatile environments". It integrates leading researchers active in the multidisciplinary domains of Dependability, Security, and Human Factors, in order that Europe will have a well-focused coherent set of research activities aimed at ensuring that future "ubiquitous computing systems", the immense systems of ever-evolving networks of computers and mobile devices which are needed to support and provide Ambient Intelligence (AmI), have the necessary resilience and survivability, despite any residual development and physical faults, interaction mistakes, or malicious attacks and disruptions.

Partners: 18 partners including the coordination by the LAAS-CNRS.

IRIT participants: Philippe Palanque (coordinator), Marco Winckler, Jean-François Ladry (PhD student), Eric Barboni (Post Doc) & Sandra Steere (Post Doc)

Funding: EU (IST), FP6

Duration: 36 months, January 2006 – December 2008.

(2007-2008) **AROVE-v: Assessing the resilience of open verifiable E-voting systems**

Summary: AROVE-v investigated some practical, case study based learning about building

dependability cases for systems of these characteristics. In particular, the domain chosen for this mini-project is "E-voting". In this context, the goal of AROVE-v is to identify necessary components of a case supporting the use of an E-voting system from the viewpoint of resilience, and recommend methods for establishing the necessary evidence. This project was funded by the EU ReSIST.

Partners: City University (London), Newcastle, IRIT-University Paul Sabatier

IRIT participants: Marco Winckler (coordinator), Philippe Palanque & Regina Bernhaupt

Funding: ReSIST network of Excellence FP6

Duration: 12 months.

(2007-2008) **FAREUS : Formal Analysis of Evolving Resilient Usable Systems**

Partners: ISTI-CNR (Italy), IRIT, Newcastle University

IRIT participants: Marco Winckler (coordinator), Philippe Palanque & Jean-François Ladry

Funding: ReSIST network of Excellence FP6

Duration: 12 months.

Summary: The proposed research is based on the vision of a future of pervasive networked devices, able to interface with the environment to support context-aware services; diverse interaction techniques through dynamic reconfiguration of devices made available to the users depending on context situations; large-scale, dynamic, disconnected networks providing a huge variety of services.

(2005-2007) **COST 294 ACTION MAUSE : Towards the MAturation of USability Evaluation**

Summary: The ultimate goal of MAUSE was to bring more science to bear on Usability Evaluation Methods (UEM) development, evaluation, and comparison, aiming for results that can be transferred to industry and educators, thus leading to increased competitiveness of European industry and benefit to the public. The main objective of the COST 294 is threefold: a) to deepen the understanding about the inherent strengths and weaknesses of individual Usability Evaluation Methods (UEMs); b) to identify reliable and valid methods to compare different UEMs in terms of their effectiveness, efficiency as well as scope of applicability; and c) to develop efficacious strategies for extracting useful information from the results of UEMs to improve the system tested. This was cooperation project that involved patterns from 19 European countries. In addition to the usual scientific contributions expected from every member in the consortium, the ICS-IRIT team has played the role of disseminations leaders for this project.

Partners: representatives of 19 EU countries.

IRIT participants: Philippe Palanque & Marco Winckler (dissemination coordinators)

Funding: European Science Foundation (ESF) – COST Action 294

Duration: 36 months

(2005-2007) **WebAUDIT : Assisted User interface Design of Interactive *Téléprocedures* on the Web**

Summary: This project was concerned by models and techniques supporting the development of electronic administrative procedures (called *téléprocedures* in French) with usability. The goal of this project was twofold: i) to develop models, methods and techniques to support the specification of *téléprocedures* - our aim was to describe a model-based development process which can ensure the non-ambiguous description of requirements driving to safe and feasible implementations; and ii) to develop methods and tools to help end-users to dealt with usable *téléprocedures*. This aspect must covers the most basic user activities, such as filling-in the forms, as well to guide users throughout the process helping them to monitor the process and to follow-up the inner workflow process. The development of such as a guiding method is also concerned by people with disabilities. This project supported the and financed the PhD thesis of Joseph Xiong (2004-2008) via a funding MNRT CIFRE and also contributed to funding of the PhD thesis of Florence Pontico (2004-2008).

Partners: Génigraph, University Paul Sabatier - IRIT.

IRIT participants: Marco Winckler (scientific coordinator) and Christelle Farenc

Funding: Industrial Cooperation, MNRT (CIFRE). ICS-Budget: 100 K€

Duration: 36 months / January 2005 – December 2007

(2002-2006) **SPIDERWEB: Specification and Prototyping for user Interface Design, Engineering and Re-engineering for the Web**

Summary: SPIDER was an international project whose main aim was to support the cooperation among researchers from IRIT (Toulouse, France) and the *Instituto de Informática*-UFRGS (Porto Alegre, Brazil) on the investigation of models, tools and theories about Computer Human Interaction and Web. The main goal of this project was to produce a framework for supporting the development of web applications with usability. In the core of this framework were methods to specify web design and describe user interface. This project investigate methods from many fields including: formal methods for HCI, web visualization, task modeling, usability evaluation, guidelines for web design.

Partners: IRIT-University Paul Sabatier (Toulouse, France) and *Universidade Federal do Rio Grande do Sul* – UFRGS (Porto Alegre, Brazil)

IRIT participants: Philippe Palanque, Christelle Farenc & Marco Winckler (coordinator)

Funding: CAPES/COFECUB (project de coopération franco-brésilien). ICS-IRIT budget: 40 K€

Duration: 48 months / February 2002 – December 2006

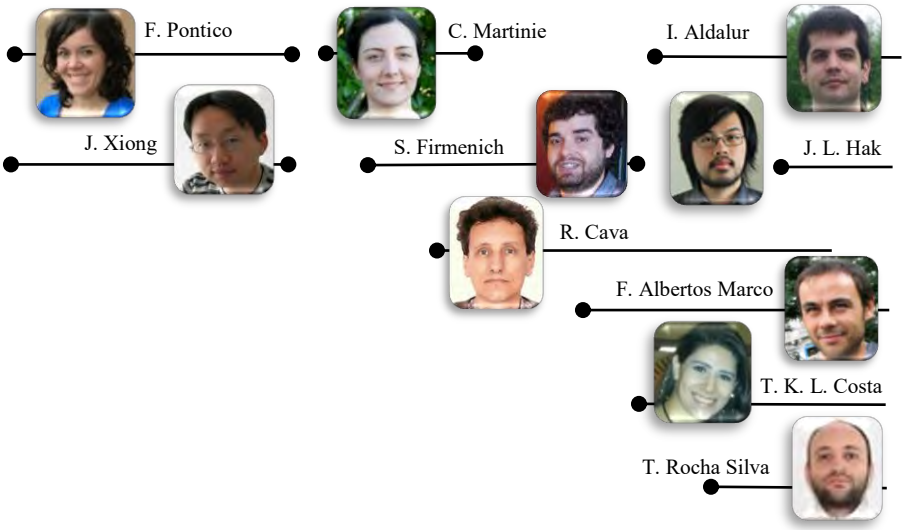
Timeline

2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017

Positions

PhD student	A.T.E.R. Université Toulouse 1	Assistant Professor / *Maître de Conférences* *Dept. SERECOM* *IUT de Tarbes*	Assistant Professor / *Maître de Conférences* *FSI-UPS*

PhD Students



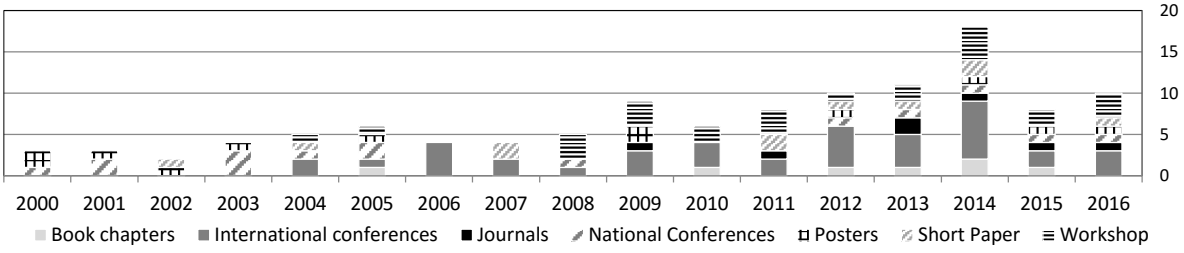
Projects

Internatioal cooperation	SPIDERWeb Capes/Cofecub		MEDIAWeb SECYT/CNRS	VIDAS Capes/Cofecub	WAMAW-OUR Stic AmeSUD
Cost Action		MAUSE		TwinTide	
Industrial Cooperation		WebAUDIT			
ANR		MyCitizSpace		PIMI	
EU		EU NoE ReSIST		FEDER Ubiloop	
		EUROCONTROL HALA !			

Conferences organization

TAMODIA 2007 TP Chair
IHC 2008 TP Chair
PUX 2011 Organiser
ICWE 2014 General chair
INTERACT 2017 TPC chair
IWWOST 2009 Organiser
IWWOST 2007 Organiser
IWWOST 2008 Organiser
HCSE 2012 TP Chair
HCSE 2014 Organiser
ATACCS 2015 Local Organiser
DEGAS 2007 Organiser
DEGAS 2009 Organiser
Public Polices at CHI 2013, Organiser
HCSE+HESSD 2016 PC chair

Publications



PART III – Selected Publications

Firmenich, S., Rossi, G., Winckler, M., Palanque, P. An Approach for Supporting Distributed User Interface Orchestration over the Web. In. International Journal of Human-Computer Studies. ISSN 1071-5819. Vol. 72(1): 53-76 (2014). At: http://dx.doi.org/10.1016/j.ijhcs.2013.08.014

Winckler, M., Bach, C., Bernhaupt, R. Identifying User Experience Dimensions for Mobile Incident Reporting in Urban Contexts. In IEEE Transactions on Professional Communication. Volume 56, Issue 2, June 2013, pp. 97-119. [doi: http://dx.doi.org/10.1109/TPC.2013.2257212]

Palanque, P., Barboni, E., Martinie, C., Navarre, D., Winckler, M. A model-based approach for supporting engineering usability evaluation of interaction techniques. In Proceedings of the 3rd ACM SIGCHI symposium on Engineering interactive computing systems (EICS '11). ACM, New York, NY, USA, 21-30. DOI=http://dx.doi.org/10.1145/1996461.1996490

Martinie, C., Palanque, P., Winckler, M. Structuring and Composition Mechanism to Address Scalability Issues in Task Models. IFIP TC13 Conference on Human-Computer Interaction (INTERACT 2011), September 05-09 2011, Lisbon, Portugal, Springer LNCS 6949, pages 593-611.

Winckler, M., Palanque, P. StateWebCharts: a formal description technique dedicated to navigation modelling of Web applications. In Proceedings of the International Workshop on Design, Specification and Verification of Interactive Systems - DSVIS'2003, Funchal, Portugal, June 2003, Springer LNCS Volume 2844, 2003, pp 61-76.

Int. J. Human-Computer Studies 72 (2014) 53–76

Contents lists available at ScienceDirect

Int. J. Human-Computer Studies

journal homepage: www.elsevier.com/locate/ijhcs

An approach for supporting distributed user interface orchestration over the Web

Sergio Firmenich [a], Gustavo Rossi [a], Marco Winckler [b,*], Philippe Palanque [b]

[a] Universidad Nacional de la Plata and CONICET, Argentina
[b] ICS-IRIT, Université Paul Sabatier, France

ARTICLE INFO

Article history:
Received 11 September 2012
Received in revised form
19 July 2013
Accepted 28 August 2013
Available online 18 September 2013

Keywords:
Distributed user interfaces
Task and process modeling
Web application
Web augmentation
Collaborative Web tasks

ABSTRACT

Currently, a lot of the tasks engaged by users over the Web involve dealing with multiple Web sites. Moreover, whilst Web navigation was considered as a lonely activity in the past, a large proportion of users are nowadays engaged in collaborative activities over the Web. In this paper we argue that these two aspects of collaboration and tasks spanning over multiple Web sites call for a level of coordination that require Distributed User Interfaces (DUI). In this context, DUIs would play a major role by helping multiple users to coordinate their activities whilst working collaboratively to complete tasks at different Web sites. For that, we propose in this paper an approach to create distributed user interfaces featuring procedures that are aimed to orchestrate user tasks over multiple Web sites. Our approach supports flexible process modeling by allowing users to combine manual tasks and automated tasks from a repertoire of patterns of tasks performed over the Web. In our approach, whilst manual tasks can be regarded as simple instructions that tell users how to perform a task over a Web site, automated tasks correspond to tools built under the concept of *Web augmentation* (as it *augments* the repertoire of tasks users can perform over the Web) called *Web augmenters*. Both manual and automated tasks are usually supported by specific DOM elements available in different Web sites. Thus, by combining tasks and DOM elements distributed in diverse Web sites our approach supports the creation of procedures that allows seamless users interaction with diverse Web site. Moreover, such an approach is aimed at supporting the collaboration between users sharing procedures. The approach is duly illustrated by a case study describing a collaborative trip planning over the Web.

1. Introduction

Currently, many tasks users engage over the Web involve dealing with different Web sites; for example, planning a simple trip would require the visit of a first Web site for booking a hotel, a second one for booking flights and many more for finding interesting sightseeing places at the destination… Despite the fact that users would consider such Web navigation as being part of the same task (i.e. planning a trip) most Web sites will run independently with little support to the actual users' concern (Firmenich et al., 2010). Moreover, although Web navigation was regarded in the past as a solitary activity, a large proportion of users are nowadays engaged in collaborative activities (Morris, 2008); for example sharing with colleagues the results of a search for cheap hotels, explaining to friends how to book a seat next to yours in a flight, outsourcing tasks such as asking the community to suggest nice sightseeing places at the destination… For a motivating example, in Fig. 1 we illustrate a scenario for collaborative trip planning to attend a conference. For accomplishing this common goal, two users need to gather general information about a conference such as the conference dates and location, buy flights, book hotel, etc. Each user can perform the required tasks individually. However, if users want to travel together, some coordination and communication will be required for booking the same flights, hotels etc. Moreover, users might decide to share the work, for example one user can book for both participants. In the scenario presented by Fig. 1 it is possible to notice the many Web sites that will be visited by users. As we shall see, despite the fact that there is dependency between the information provided, Web sites are not integrated. While building service-based software such as mashups can be a solution for combining data and information from different providers, many times this approach might have limitations as they can hardly integrate all possible opportunistic tasks that users might have in mind; for example, checking the traffic situation on the way to the airport, booking the preference users' restaurant at the destination, etc.

We argue that there is a huge set of processes and tasks (such as planning a trip collaboratively) performed nowadays over the

* Corresponding author. Tel.: +33 5 61 55 62 58.
E-mail address: winckler@irit.fr (M. Winckler).

1071-5819/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.ijhcs.2013.08.014

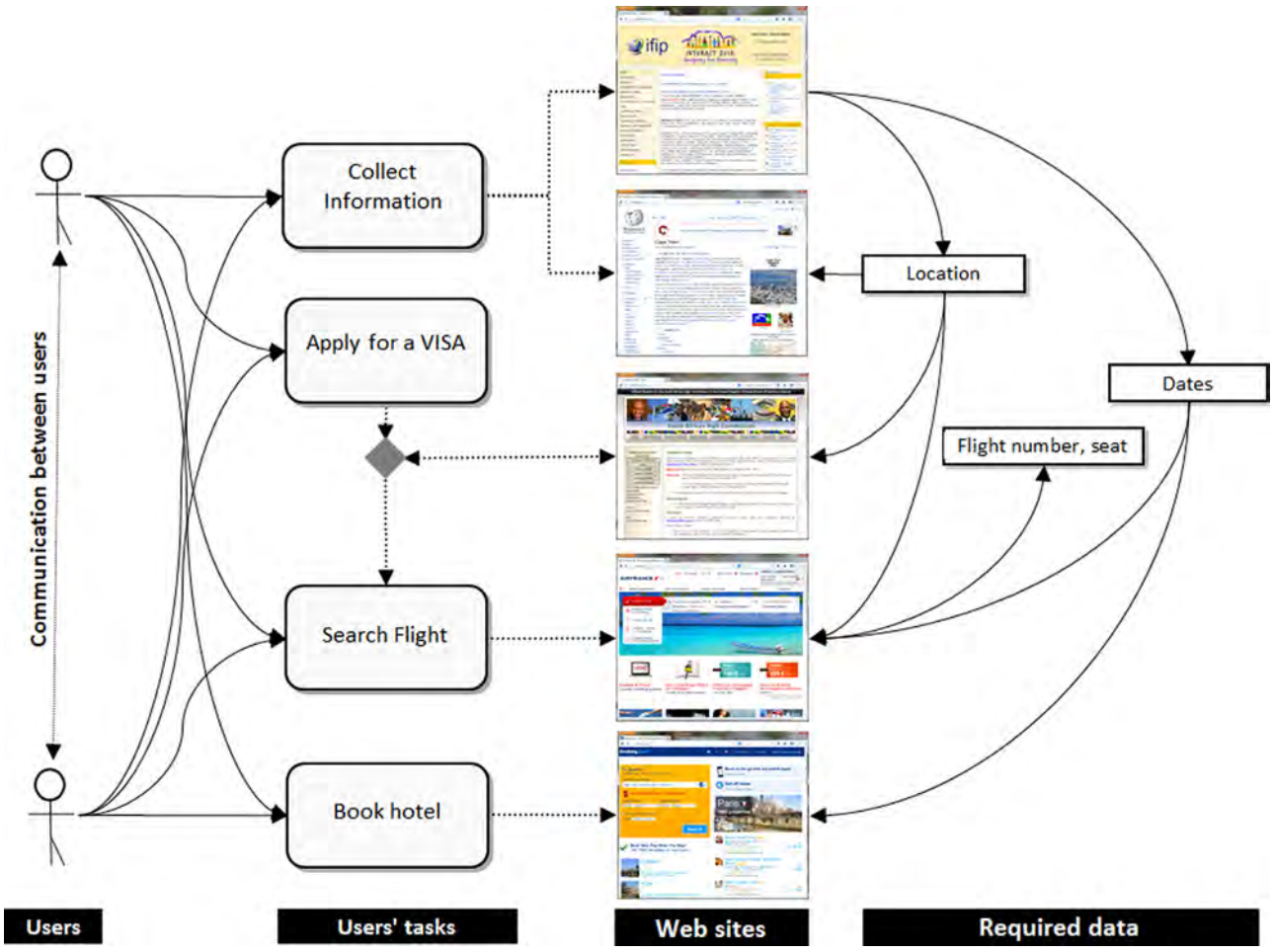


Fig. 1. Distributed user interaction for planning a trip including information sharing between users.

Web which requires a level of coordination that can only be achieved by Distributed User Interfaces (DUI). Distributed User Interfaces (DUI) has been recently defined as a "*user interface whose components are distributed across one or more of the dimensions input, output, platform, space and time*" (Gallud et al., 2011). The example shown by Fig. 1 highlights two important aspects of user activity over the Web that appeal for distributed user interfaces: (i) users tasks are distributed at multiple applications, i.e. Web sites, that are not directly connected to each other; (ii) users working collaboratively share information to accomplish a common goal (i.e. travel together).

In this paper we propose an approach to building distributed user interfaces that aim at ensuring a smoother user interaction whilst users are performing their tasks across multiple Web sites. Our approach combines individual user tasks to create procedures that seek to orchestrate user tasks over multiple Web sites. Such an approach is aimed at: (i) allowing flexible tasks modeling so that users can create ad-hoc processes for describing how to accomplish tasks in different Web sites; (ii) helping users to share such processes with friends and colleagues who might benefit of the guidance provided by prior task planning; (iii) support a seamlessly integration between data available over Web pages and the actual tasks performed by users across different Web sites; (iv) support the automation of user tasks by the means of Web augmentation tools that are aimed at helping users in their tasks; (v) mediate user interactions with the Web site via the execution of the so-called Web augmentation tools.

Our approach is built upon the concept of Web augmentation (Bouvin, 1999; Brusilovsky, 2007; Brusilovsky et al., 2007) that defines strategies for implementing tools that can extend the set of elementary tasks users can do whilst navigating the Web. For that, we have developed a dedicated framework called Context Sensitive Navigation (CSN) (Firmenich et al., 2010) which implements a set of Web augmentation tools called *augmenters* (Firmenich et al., 2011). These augmenters are the basic building blocks for extracting information and DOM elements from diverse Web sites for creating distributed user interfaces. In Section 2 we revise the main concepts that are necessary to understand how our proposal is related to the state of the art on DUIs. Section 3 describes the CSN framework and individual augmenters. Later on in Section 4 we properly present an overview of our approach for building DUIs by composing individual augments and a Domain Specific Language (DSL) that formalizes the composition of procedures made of the assembly of individual tasks (Section 4.2). The section about the approach ends with the description of the corresponding tool support (Section 4.3). Section 4.4 presents a case study of a DUI for collaborative trip planning using the tool support that demonstrates the feasibility of our approach. Section 6 reports the preliminary results of a usability testing with 11 participants. In Section 2.4 we discuss the contributions of our approach to the research in distribute user interfaces (DUI). Finally, Section 7 presents conclusion, lessons learned and future work.

2. Related work

The field of Distributed User Interfaces is broad and spans over the development of Web applications. For the sake of conciseness

we will concentrate on those research works involving Web applications which are close to our intent.

2.1. Collaborative work on the Web

There are many tasks that users can perform collaboratively in small groups (i.e., colleagues, family, or friends) such as researching group projects or reports, arranging joint travel, or planning shared entertainment opportunities. With the advent of Web 2.0 paradigm, several Web applications implement tools that allow users to interact and collaborate with each other in a social media (Morris, 2008). The collaboration using Web 2.0 technology is often focused a single Web site where the context of the task they allowed to do is defined by the Web site (e.g. edit a document, annotate page, upload images, etc.); a popular Web 2.0 tool is Google Docs (docs.google.com) where multiple users can edit collaboratively a single document.

However, more recent tools extended the concept of collaboration for supporting distributed user interfaces over multiple devices. As discussed by Paul and Morris (2009), most of these tools support awareness features (e.g., sharing of group members' query histories, browsing histories, and/or comments on results) and division of work features (e.g., chat systems, the ability to manually divide search results or URLs among group members, and/or algorithmic techniques for modifying group members' search results based on others' actions).

There is strong evidence that people frequently share the results of their searching of the Web (Melchior et al.; Morris, 2008; Schmid et al., 2012). In the last years much attention has been focused on collaborative Web search (Rädle et al., 2012). Several tools, such as CoSearch (Amershi and Morris, 2008), SearchTogether (Morris and Horvitz, 2007), CoSense (Paul and Morris, 2009) and Twister-Search (Rädle et al., 2012), support collaborative activities of searching over the Web. CoSearch (Amershi and Morris, 2008) features a specialized browser that implements a queue of queries performed by the group and users can associate notes to individual Web pages. Users can share the work by downloading distinct subsets of search results to their mobile phones. SearchTogether (Morris and Horvitz, 2007) is a prototype that enables remote users to synchronously or asynchronously collaborate when searching the Web by implementation mechanisms for awareness (people can see the activities of other participants), division of labor (search can be split among individuals), and persistence (results are available for later use). SearchTogether is so far a tool for supporting the investigation of sensemaking (which means how people value or give a meaning to information) in collaborative search. In order to investigate more precisely how sensemaking activities occur in Web navigation, Amershi and Morris (2008) have specifically designed the tool CoSense. CoSense supports collaborative searching as active discussion among participants around the results found by the group.

2.2. Web augmentation and data integration tools

After having found useful information over the Web, it is very likely that information will be used to accomplish other tasks. Indeed, quite often the contents users need to accomplish their goals are scattered around different Web sites. In more recent years, several tools have been developed to support the integration of Web contents into a seamlessly user interface. By doing so, the interface produced is “augmented” with respect to what users can do at the original Web sites. We can identify two coarse-grained approaches for developing such Web augmentation tools: (i) mashing up contents or services in a new application and (ii) augmenting the original application, generally by running adaptation scripts on the client side. Mashups (Yu et al., 2008; Wong and Hong, 2007) are an interesting alternative for final users to combine existing resources and services in a new specialized application. Most of the approaches aim at allowing users without programming skills to produce these new applications. Visual and intuitive tools such as (Daniel et al., 2009; Yahoo Pipes!, 2012) simplify the development of these applications. Some of the approaches are focused on UI integration, but others – such as Yahoo Pipes! (Yahoo Pipes!, 2012) – are focused on data integration. One of the main problems about mashups is that they, usually, are based on Web services. The fact is that most Web applications do not provide Web services to access their functionality or information. In order to solve this problem, Han and Tokuda (2008) propose an approach to integrate contents of third party applications by describing and extracting these contents at the client-side and then, using these contents later by generating virtual Web services that allow accessing them.

As we said before, the goal of mashups tools is to integrate UI or data in order to generate a new application. Clearly, this means that Web applications are taken off their original versions, which could not be desirable for many users; moreover Web applications are really dynamic about functionalities or contents, for example, news, special offers, etc. In contraposition, several tools for Web augmentation exist which aim to modify the preferred Web sites of users, while maintaining intrinsic features of these. Usually, these tools are implemented as browser's plug-ins that allow users to either install or generate scripts for modifying the Web pages they visit; for example GreaseMonkey (Diaz et al., 2010; GreaseMonkey, 2012; Pilgrim, 2005) and Scriptish (Scriptish, 2012). Nowadays, the most popular tool is GreaseMonkey, a browser plug-in for client-side scripting. Basically GreaseMonkey executes scripts (JavaScript files installed and created by users) when a target Web page is loaded. Then the Web page DOM is modified before it is shown to the user. Most of the GreaseMonkey scripts adapt Web pages in a static way, i.e. that the adaptation goals do not contemplate which is the user task or concern. Although passing information among Web applications with GreaseMonkey scripts is not so easy to achieve, this is not the common goal of GreaseMonkey scripts.

However, there are other approaches for supporting users' task in several ways. For example, MozillaUbiquity (MozillaUbiquity, 2012) – another Firefox plug-in – tries to improve user experience by empowering his browser with commands for dispatching operations related with his task. With MozillaUbiquity users execute commands (developed by the community) for specific operations; while some of them are more task-related (for instance, to publish some text from the Web site he is navigating in a specific social network) others can be addressed for adapting the current Web page in some way (for example, highlighting some text). MozillaUbiquity could be used even for building mashups. These commands are executed under user demand, and adaptations are not made automatically. Although MozillaUbiquity allows users to develop and execute commands for making shorter the distance between two distinct Web Applications, moving information (and performing adaptations with this information) from one of them to another is not fully exploited. A similar tool is Operator (Operator, 2012). Operator is another Firefox plug-in that performs this kind of operation by basing the available ones in correspondence to the Microformats found in the Web page the user is visiting. By finding a specific Microformat, Operator can use it for consuming Web services from Flickr, Google Calendar or del.icio.us, in order to support users with a specific task, for example, to add some event in Google Calendar based on some Microformat about dates or events found in any Web page.

Some tools have been specifically designed to automate repetitive users' tasks over the Web; for instance, CoScripter

(Bogart et al., 2008) (another Firefox Plug-in developed by IBM) allows users to record their interactions – Web pages visited, DOM events like clicks over DOM elements, etc. – and then, they can repeat the process automatically later. Although most of the scenario recorded is rigid, the approach is a bit flexible because it allows users to repeat the same scenario (Web sites, interactions, etc.) while changing the information entered in form inputs when the scenario was recorded. In this way, CoScripter is not useful when users need to change slightly the process, for example by changing one of the involved Web applications. This is not a minor issue since it is normal that users utilize diverse Web applications for the same goal (a clear example is booking a hotel room). While CoScripter is more of a tool for supporting repetitive business processes, we think that users' tasks cannot be only supported by filling forms but that there may exist several kinds of DOMs interventions for improving users' experience. CoScripter is not the only "high level API" for supporting processes on the Web. For example, ChickenFoot (Bolin et al., 2005) is a tool for programming users scripts and it extends JavaScript by defining new commands like "click()", "enter()", "find()", etc., which make easier the development of Web automation scripts. Although ChickenFoot has a powerful and expressive language, it hardly contemplates fine changes in the adaptation process and it prevents reuse of scripts as these are really DOM-dependent. The tool Koala (Little et al., 2007) proposes a more natural scripting language; for example, the expression "click('search button')" in ChickenFoot would be equivalent to "Click 'search button'" in Koala. Selenium (Selenium, 2012) is another tool based on recording users interactions. Although it was originally for performing UI tests, it is not limited to that and, in the same way as ChickenFoot and CoScripter, it could be helpful for performing repetitive tasks.

Although these tools are really useful and we share the philosophy behind them (in terms of putting the power on users hands, trying to make as easy as possible the end-user development, and trying to improve user experience), we think that the use of the Web is not as rigid as these approaches need. All these tools are based on registering all events (at least most of them) that occur in the Browser in order to be able to repeat the same sequence of events later. Anyway, behind each event that occurs in the browser there is a user intend, concern or a task in mind. This semantic weight behind each event is not taken into account, and for us, it is important to do it in order to understand in a better way which is the real user concern. In this way, we think that these tools are more for automatic-use Web applications rather than adapting them in order to add new contents (or adapt these), functionalities (or adapt these), which could be useful in the context of users tasks.

Moreover, we believe that it is necessary to go a step further in those aspects which imply an integration of information among several Web pages. In Firmenich et al. (2011) we showed how to use the actual user concern (expressed in his navigational history) as an additional parameter to adapt the target application. The main constraint in our previous work was that the development of the artifacts could be made only by experienced JavaScript programmers. This is not only counterproductive for creating new scenarios, but this mechanism is not good to analyze real behavior of users on the Web.

2.3. Task and process modeling

Task analysis is widely recognized as one fundamental way to focus on the specific user needs and to improve the general understanding of how users may interact with a user interface to accomplish a given interactive goal (Diaper and Stanton, 2004). In the field of HCI many task model notations have been proposed for capturing in models the various elements of user activity in interactive systems Card et al. (1983).

In the last decades, several tasks notations have been developed as means to describe work carried out by users whilst interacting with a system (Paternò et al., 1997). Despite the fact that various specific task notations exist, they are mainly structured around two concepts: task decomposition (often represented as a hierarchy) and task flow (for showing the order in which tasks are executed) (Limbourg et al., 2001). When adequately combined, these concepts can provide an exhaustive and complete representation of large quantity of information in a single model. However, as discussed by Paternò and Zini (2004), when applied to real-life systems, tasks notations end up in very large, hard-to-manage models thus making task modeling a time-consuming and sometimes painful activity.

More recent task model notations such as HAMSTERS (Martinie et al., 2011) integrate structuration mechanisms such as modularity in the tasks representation, reuse of sub-tasks elements, data flow and communication features between task elements. These features allows more flexibility in the organization of task-model diagrams and allow us to envisage the use of tasks models for describing complex interactive systems build upon the composition of individual tasks and their distribution into the user interface (Luyten and Coninx, 2005).

However, the experience with task-model for building DUIs is mainly driven but the construction of an application whose interface is going to be distributed among different devices and platforms (Luyten and Coninx, 2005; Manca and Paterno, 2011; Melchior et al.). In order to integrate tasks that can be performed in multiple Web sites, Daniel, Soi and Casati (Daniel et al., 2009) propose the concept of user interface orchestration that is aimed to describe how distributed user interfaces can be coordinated around a single business process. These works are promising and yet they demonstrate the value of tasks models for helping to build complex distributed user interfaces.

2.4. Positioning the contribution with respect to the DUI paradigm

Systems which support the management of complex tasks and of a huge amount of information usually require Distributed User Interfaces (DUIs) for allowing multiple users to perform their tasks in a (possibly) concurrent way. Several definitions of this kind of interfaces co-exist and present interestingly complementary viewpoints Vanderdonckt (2010).

In Lin et al. (2009), a UI distribution is defined as "*the repartition of one or many elements from one or many user interfaces in order to support one or many users to carry out one or many tasks on one or many domains in one or many contexts of use, each context of use consisting of users, platforms, and environments*". This definition focusses on the notion of repartition of UI elements in several locations. Another definition proposed by Elmqvist (2011) identifies five dimensions for the distribution of UI components: input, output, platform, space and time. This definition emphasizes the fact that there is no need to distribute on all these dimensions to be eligible for DUI paradigm. Demeure et al. (2008) also propose a reference framework (called 4C) to analyze DUIs, which is composed of four concepts: computation, coordination, communication and configuration. It builds on top of the design space identified by Salber based on Production, Coordination and Communication for groupware (Laurillau and Nigay, 2002).

In the contribution presented in this paper, we propose a tool-supported architecture allowing information exchange and dynamic tasks allocation between multiple distributed users involved in a common task. This contribution heavily relies on the exploitation of innovative Web technologies (as demonstrated in the case study presented in Section 4.4). However, the proposed

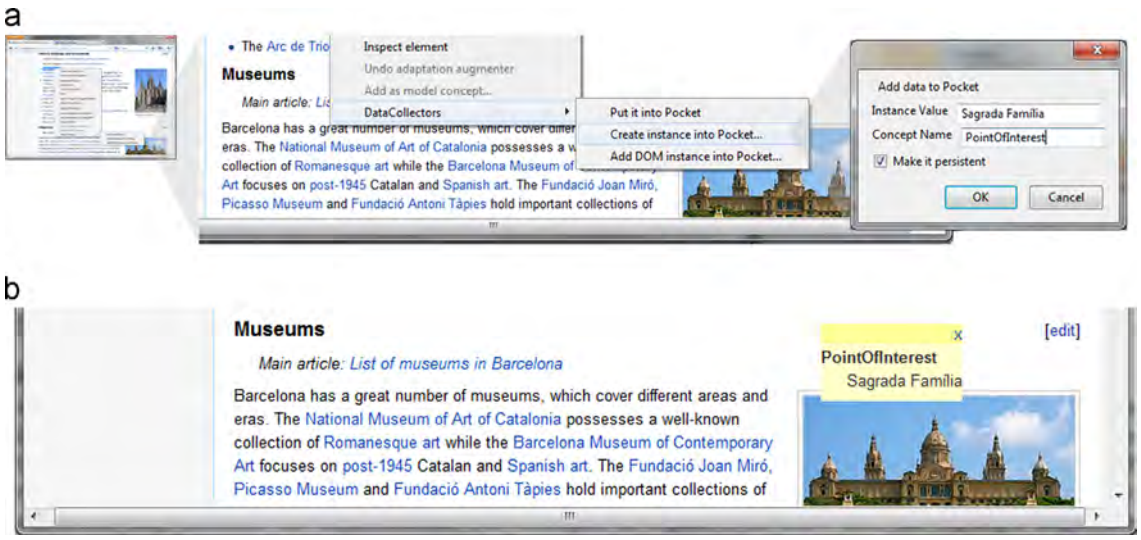


Fig. 2. Illustration of the use of an augmenter pocket. (a) Menu in the augmenter tool for collecting data whilst navigating a Web page. (b) Items collected with the augmenter pocket as they appear on Web pages.

framework could be transposed to any platform supporting multi-user interactions. Indeed, the construction of a procedure (that will be then allocated to a remote collaborator) is based on the blending in one single entity of several user interfaces components extracted from multiple Web sites. The same could be performed in a similar way between multiple interactive applications.

According to Elmqvist's definition of DUIs (Elmqvist, 2011) we can position our work as follows:

- Input: user input is distributed amongst the various users exploiting multiple Web sites.
- Output: in our approach, there is no specific contribution on output beyond the fact that output is tightened to input (which is distributed as explained above).
- Platform: even though the contribution does not address explicitly the distribution over various platforms, there is a high potential of distribution due to the close connection of the contribution with Web technologies.
- Space: the choice of Web technologies was explicitly made to support distribution of user and of their tasks in space. Indeed, the contribution directly targets at supporting multiple remote users involved in collaborative activities as demonstrated in the case study.
- Time: the contribution here mainly focusses on asynchronous collaborations as augmenters are meant to be "sent" to remote users who will exploit them at their will. Synchronous mechanisms could be added to the framework to provide communication support for performing the tasks but this has not been presented here.

3. Overview of an Web augmentation

Our approach for building distributed user interfaces is based on the CSN framework (Firmenich et al., 2010) and it relies on the existence of Web augmentation tools that can support users' tasks that use contents from multiple Web sites. Hereafter we show how to use Web augmentation tools to support users' tasks whilst the navigation over the Web. We introduce Web augmentation tools and the CSN framework. Later, we describe how such Web augmentation tools built with our framework can be combined to feature distributed user interfaces that can support processes performed by multiple users.

3.1. Web augmentation tools and the CSN framework

The concept of Web augmentation, as defined by Bouvin (1999), is used to refer to a set of tools that extend tasks users can usually perform on the Web. Several strategies for implementing Web augmentation tools exist (Diaz et al., 2010; Firmenich et al., 2010; GreaseMonkey, 2012; Pilgrim, 2005; Scriptish, 2012). In our previous work (Firmenich et al., 2010) we have presented our own framework called CSN that support the development of such Web augmentation tools called *augmenters*. Augmenters feature generic functions that can be executed on client-side (i.e. on the Web browser) and perform adaptation of DOM elements. Tasks supported by augmenters might include *automatic filling in forms*, *highlighting text*…

Fig. 2 illustrates the execution of an *augmenter* called *pocket*. This *augmenter* was implemented under the metaphor of a real pocket; thus users can collect data into the pocket and then carry on such data whilst navigating the Web. Data can be simple text values (such as "*Sagrada Familia*", "*Barcelona*") or text values with semantic meanings (such as like "*PointOfInterest*" or "*City*"). Fig. 2a shows how the augmenter is activated from a contextual menu over a Web page to create an instance of the concept *PointOfInterest* for the previous selected text "*Sagrada Família*". Once collected, the augmenter modifies the Web page in the client by creating a new DOM element featuring an electronic post-it to display the information collected by the user, as shown in Fig. 2b.

Fig. 3 illustrates another augmenter called *highlight* that can be activated directly from the information previously collected by the users using the augmenter *pocket*. In Fig. 3a, we show how the *Highlight* augmenter is triggered by right-click the target Pocket element, which opens the contextual menu with all the augmenters available. Fig. 3b shows the Web page already augmented, after executing the augmenter. Note that the *PointOfInterest* was collected from Wikipedia and is available on other Web sites. Since the augmenter was executed with the concept *PointOfInterest*, all its instances were contemplated for being highlighted. The augmenter can be executed only with one instance, for example "*La Rambla*".

Currently, a large set of augmenters are deployed with the CSN framework. These augmenters are grouped in Table 1 according to the following categories: data collectors (which at aimed to help users to collect information from Web pages), filling forms (which are focusing on users tasks involving Web forms), select

Fig. 3. Illustration of the use of an augmenter highlight. (a) Triggering the augmenter highlight. (b) Highlight effect on Web page.

data (in the Web page by editing CSS properties), navigation (which adds elements to Web pages for helping users to improve navigation) and accessibility (which concerns edits on the Web page that are aimed to improve the accessibility of the Web page).

It is interesting to notice that augmenters can be used in combination for supporting sequences of tasks. Fig. 4 shows how the augmenters *ConceptInstanceCollector* and *IconifiedLink* can be executed in the sequence in a row to create navigation from Web pages to the Google Maps site using data previously collected.

Augmenters can be used to support users or to automate tasks. For example, the augmenter *UntypedDataCollector*, which is a particular implementation of the *augmenter pocket* without semantic meaning for the values collected, acts as a memo or electronic post-it that follows users through his navigation on different Web sites. This augmenter helps users to remember important information whilst navigating the Web. Another example is the *CopyDataIntoInput* that can be used to automatically fill in form fields with data previously collected by any augmenter in the category *data collector*. These two examples clearly show that these augmenters are not only extending what users can do on Web pages (e.g. create electronic post-its) but that user performance can be improved by automating some of the user actions (i.e. automatically fill-in forms). It is also interesting to notice that *augmenters* provide users with an alternative for performing the tasks. For example, users can type the information previously collected into a form field or trigger the *augmenter CopyDataIntoInput* to perform the same action. At some extension, such augmenters allow distributed control over tasks that users perform on Web sites.

So far, we have shown examples where only plain text is collected. However, this is not the only kind of information that may be *moved* from one application to another. Sometimes is convenient to show several parts of the user interface from different Web sites at one time, such as what Mashups or Web Integration approaches do. In this work, users can manage DOM elements with the same flexibility as plain text, i.e.: collect DOM elements into the *Pocket* under some semantic label, and then performing adaptations based on this semantic label using augmenters. As a first example, in Fig. 5a we show how a user can collect a specific DOM element from Booking.com and then insert it in other Web site. This could be made by executing the corresponding augmenter *UseForReplace* that replaces the element which is selected by the user with the collected DOM element into the Pocket.

Fig. 5b shows the execution of this augmenter in the Airfrance.com Web site. The result of the adaptation is shown in Fig. 5c. Note that although the augmenter can be executed from the concept *HotelInfo*, and in this way the DOM element inserted may vary if a different DOM element was collected under this semantic tag.

The example from Fig. 5 shows how to apply concern-sensitive integration of UI components. Here, the user has finished the task of booking a hotel room, and when he was going to buy flight tickets, informations about the hotel (name, address, both check-in and check-out dates, etc.) were available in other Web page. This kind of integration allows users to interact with parts of the UI from an external Web site in the current one. For example, if the collected DOM element contains links or forms, then the user can interact with this element when inserted in other existing Web page.

3.2. Users of the CSN framework

The CSN framework is supported by the means of a Firefox plugin. A small set of basic Web augmenters are deployed with the installation of the plugin including the before mentioned *highlight* and *pocket*. However, it is possible to create and install additional augmenters into the plugin. Therefore, there are two kinds of users of the CSN framework:

(i) End-users who take advantage of existing augmenters deployed with the CSN plugin; these users benefit of these augmenters during navigating by "collecting" information to be used when adapting the user interface.
(ii) Developers who create and distribute new augmenters that can be played in the Web browser. Despite the fact that a large set of augmenters already exist, creative developers can use the framework to implement new augmenters at they will. With this structure we can put the power on the crowd, allowing any Web developer to create his own augmenter. Moreover, new augmenters can be shared by other users of the tools implemented with the CSN framework. This crowdsourcing development has been proved successful in other communities, for example with those using GreaseMonkey scripts (GreaseMonkey, 2012).

3.3. Creating and describing a simple augmenter in the CSN framework

This section shows how new *augmenters* can be created and instantiated into the CSN Framework. Overall, there is no need to

Table 1
Set of augmenters currently deployed with the CSN framework.

Augmenter	Category	Description	Execution mode
UntypedDataCollector	Data collector	Quickly add some data into the pocket without semantic weight	User/system
ConceptInstanceCollector	Data collector	Add conceptualized data into the pocket, for example, for the concept "*City*"" collect the text "Barcelona"	User/system
DOM Element Collector	Data collector	Collect a specific DOM element into the Pocket	User/system
Remote Collector	Data collector	Collect data from a remote Web application (not the current Web site)	System
RegularExpressionBasedCollector	Data collector	Add data into the pocket based in a regular expression, for example, for collecting e-mail addresses.	System
MicroformatBasedDataCollector	Data collector	Add data into the pocket based in Microformats annotations.	User/system
CopyDataIntoInput	Filling forms	Copy a value form the Pocket to the select input	User/system
AnnotateInputWithMicroformat	Filling forms	Annotate the Web forms inputs with Microformats	User/system
MicroformatFormFiller	Filling forms	Based in the Microformats found in the current Web page	User/system
Highlight	Select data	Highlight selected data from pocket in the current Web pages	User/system
Remove	Select data	Delete selected element from the current Web page	User/system
WikiLinkConvertion	Navigation	Convert selected data into links to the corresponding Wikipedia article	User/system
IconifiedLink	Navigation	Add links to Google Maps and Flickr	User/system
PopUpMessage	Navigation	Shows a Popup Message, under certain circumstances, where is suggested to the user some links for being followed.	System
NewNavigationMenu	Navigation	Add new navigational menu	System
ConvertPocketIntoMenu	Navigation	Transform each Pocket element into a Link relevant for the current application or user concern	System
DOM Inject	Integration	Inject the selected element of the Pocket (only DOM Instance elements) as a child of the selected element	User/system
Replace DOM element	Integration	Replace an element of the current Web page for the selected element of the Pocket (only DOM Instance elements)	User/system
Add floating DOM element	Integration	Insert as a floating UI component the selected element of the Pocket (only DOM Instance elements)	User/system
ReplaceImageWithAltText	Accessibility	Replace all Images an put in place the alt text	System
IncreaseTextSize	Accessibility	Increase the size of the target text	System

create new augmenters to follow our approach for building distribute user interface. However, the example below illustrates the potential of the framework to create new behavior and user interfaces that can be useful for building DUIs for the Web.

Generally speaking the CSN framework provides a template that can integrate a file containing a JavaScript functions in which a new object has to be defined inheriting from *AbstractAugmenter*, as illustrated by Fig. 6. In order to create an *augmenter* the following elements should be described: (i) Metadata information used to identify the augmenters; (ii) a method *getAugmenterInstance* which is used to get augmenter instances; (iii) a class definition; in the example the class *HighlightingAugmenter* is the new augmenter object. It is also necessary to provide a value to the label instance variable; this value is used to create the menu showing to users that the augmenter is available for use; and (iv) an extension point of heritance: the last line from the example shows how the new augmenter is a subclass of *AbstractAdapter* (the extension point for augmenters).

By following the simple template above, developers can create a huge set of augmenters to feature adaptations on the DOM of the Web pages where the augmenter is triggered. The use of the template ensures that the JavaScript functions are compatible with other augmenters in the framework. Thus, a community of developers can create augmenters and share them with all other developers and end-users of the CSN plugin.

4. Building DUIs with Web augmenters

In this section we present how we have extended the CSN framework to create distribute user interfaces.

4.1. The approach in a nutshell

The overall idea behind our approach is that it supports collaboration between users that accomplish complex processes through distributed interactions over many Web sites. Such processes, called procedures here, are built by assembling basic building blocks provided by the CSN framework, each augmenter supporting a single user tasks. Different tasks in a procedure can be performed in different Web sites but one augmenter is instantiated to support the user interaction with a specific Web site. The user interface of each augmenter features elements that are extracted from the user interface of the Web site instantiated by the augmenter. The corresponding user interfaces for the whole procedure is thus a composition of elements that are originally distributed across different Web sites.

Not all user tasks are (yet) supported by a specific augmenter; so that the user interface might include instructions for the user to perform certain tasks manually using the functions already supported by the Web browser. For that purpose we use the definitions made in previous works (Byrne et al., 1999; Heath, 2010) that describe what users can do on a Web browser. These tasks, that we call here "*primitive tasks*" include actions such as "*go to a Web page*", "*locate information*", "*configure the Web browser*", "*fill in a form*", etc. The primitive tasks used in our approach are heavily inspired by the work of Byrne et Ali (Byrne et al., 1999). However we assume that, according to new features implemented by most recent Web browser, the set of primitive tasks could be extended in the future. Fig. 7 provides a view at glance of the approach.

The composition of the user interface is done by creating a sequence of individual tasks, which is formalized by a Domain Specific Language (DSL) and stored as a XML file. A tool called procedure player is able to parse that XML file and reconstruct the procedures on the Web browser. It includes functions for supporting the collaboration between users. As we shall see, the approach includes three phases, as follows:

- Definition of *Primitive* and *augmentation* tasks: it concerns the inclusion of augmenters in the CSN framework. This phase required a highly skilled Web developer who is able to program augmenters. Whilst this task is demanding, the work should be done once and it will benefit many more users. The development of new augmenters is out of the scope of this paper. Nonetheless, the

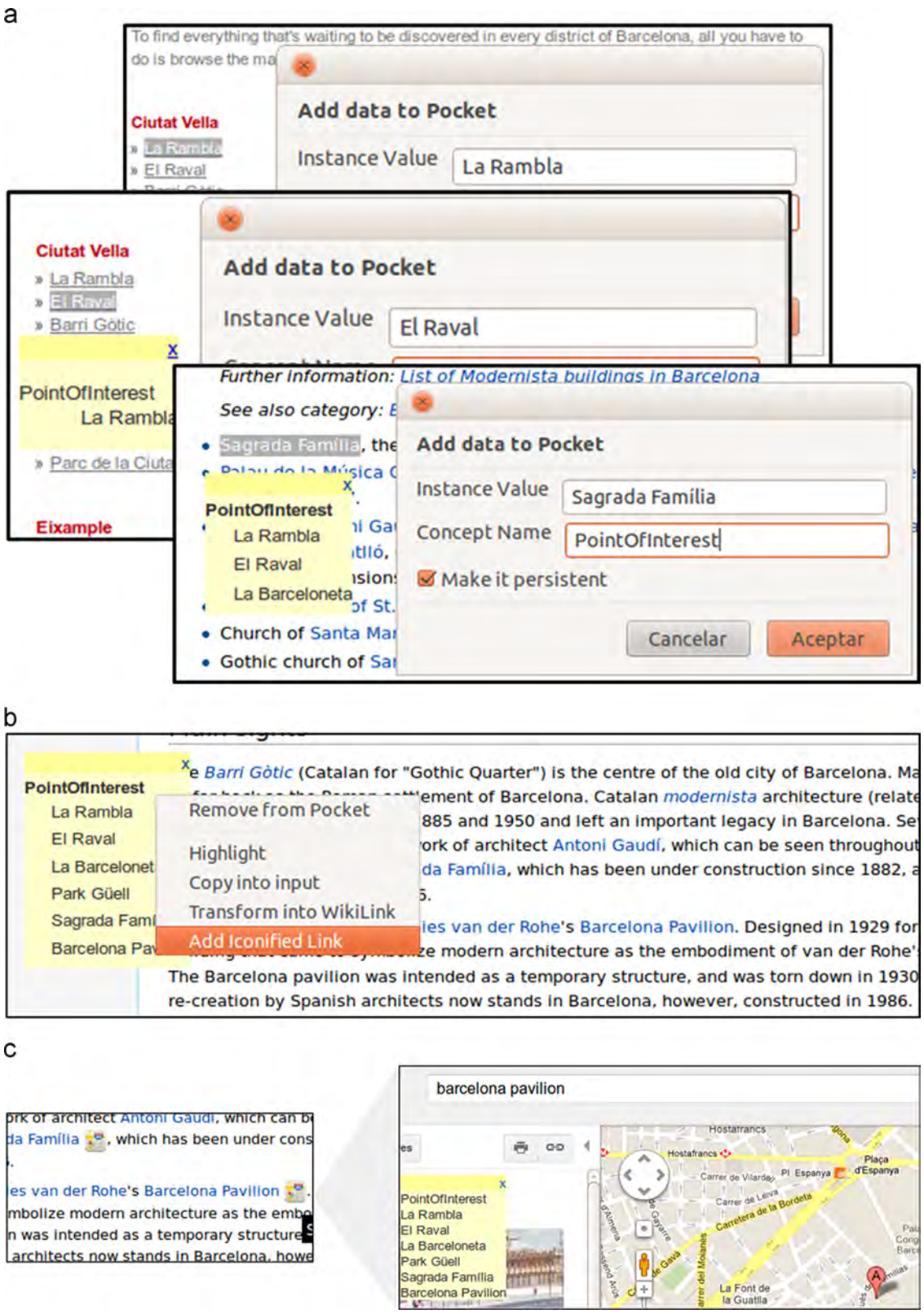


Fig. 4. Combined use of augmenters ConceptInstanceCollector and IconifiedLink. (a) Collecting 'Points of interest'. (b) Applying the augmenter IconifiedLink over collected 'Points of Interest'. (c) Result of the augmenters IconifiedLink over the previously collected point of the interest. At the left, a Web page augmented with the icons that feature a link to the Google Map Web site. At the right, the corresponding navigation which integrates the previously collected 'Points of interest' as an entry point for Google Maps.

approach is delivered with a large set of tasks, so that users do not need to create new *augmenters* to create their procedures.

- *Composition* refers to the definition of procedures by compositing tasks available in the *repository*. Users have to select a task from the repository to create a sequence of tasks. This sequence is stored in the *factory* by the means of a DSL describing all tasks in the procedure; this step is repeated until the composition is finished.

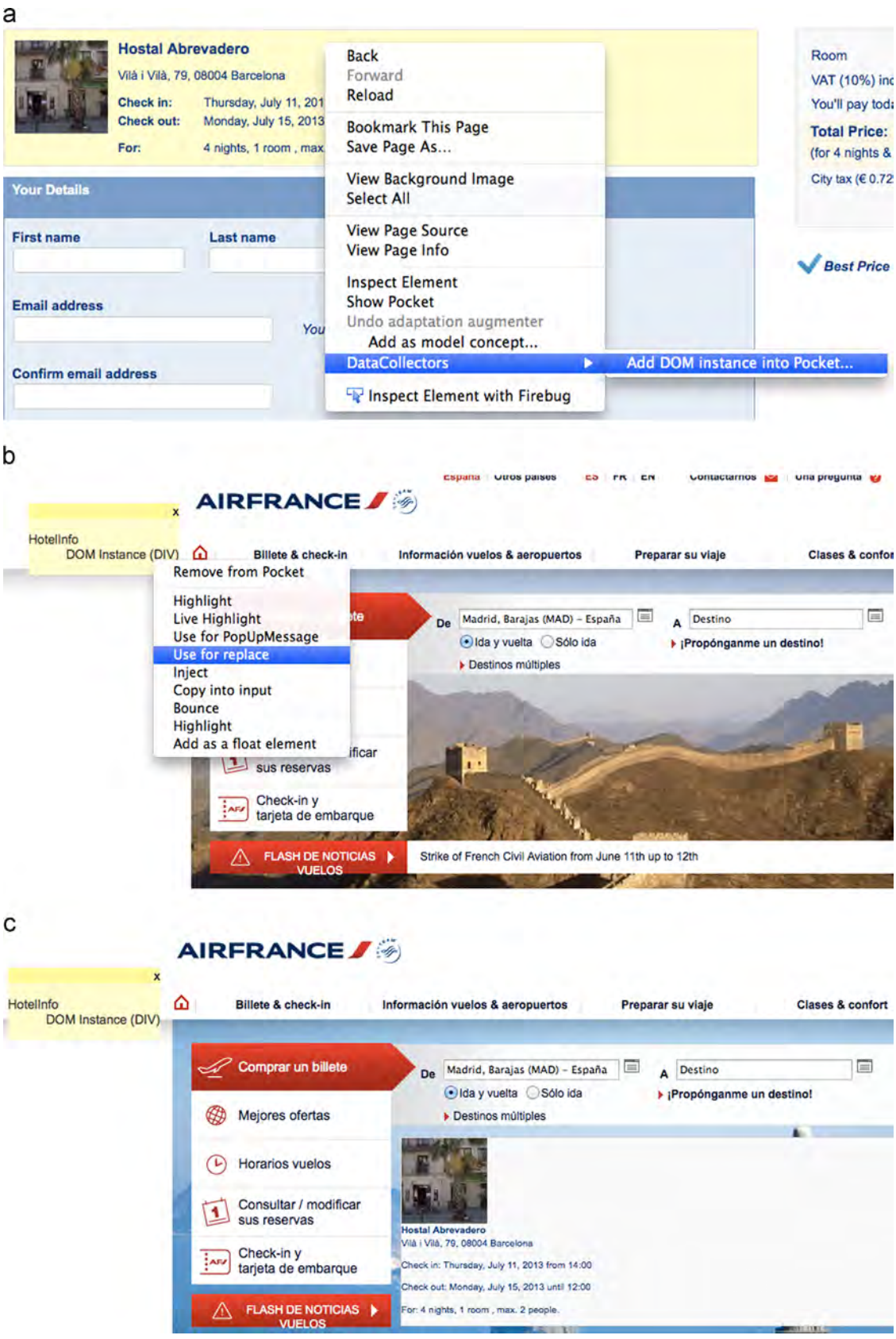


Fig. 5. Collecting DOM elements with DOMElementCollector augmenter and using the element in other Web page. (a) Collecting DOM element from Booking.com. (b) Execution of UseForReplace augmenter. (c) Result of applying Use For Replace augmenter.

```
// augmenter identification
var metadata = {"name":"My Augmenter Name", "author":"Author Name", "description": "This augmenter ........"};
// getter of augmenter instances
function getAugmenterInstance(){
   return new HighlightingAugmenter ();
};
// class definition
function HighlightingAugmenter(){
   this.label = "Highlight";
};
// point of inheritance in the CSN framework
HighlightingAugmenter.prototype = new AbstractAdapter();
```

Fig. 6. JavaScript template for building augmenters accordingly to the CSN framework.

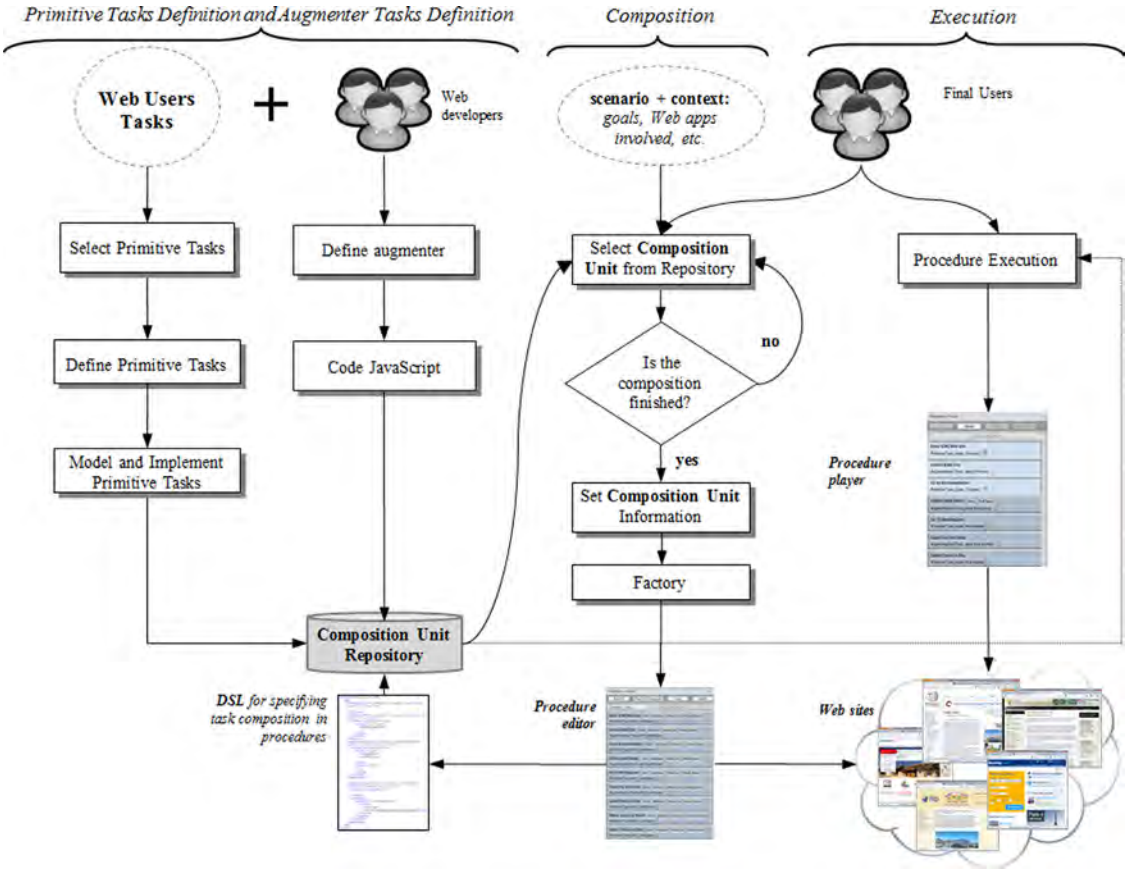


Fig. 7. Overview of the process leading to the creation of DUIs using task composition and Web augmenters.

- *Execution*: this phase features a player that is concerned by the execution of the procedure previously encoded by the DSL.

As shown in the figure above, procedures provide an alternative for the interaction with Web sites by automating users' tasks but they do prevent users from interacting directly with Web. Thus the user input is shared (or distributed) between the procedure player and the Web sites.

It is important to say that procedure player integrate functions that also support the collaboration between users, according to the policies for task synchronization described by the means of a Domain Specific Language (DSL).

4.2. A DSL for describing composition of users tasks on Web applications

In this section we provide a view at glance about the DSL that we have developed to specify the procedures and tasks in our approach. Task composition is defined according with the DSL metamodel shown in Fig. 8. This metamodel defines those elements contemplated by the DSL and their relations.

Basically, a procedure defined with our DSL is realized as an XML file containing a list of tasks. For each task additional properties can be added including *preconditions*, *postconditions* and *attributes*. The DSL elements and the corresponding syntax include the following:

- **Primitive tasks**: the tag `<primitivetask/>` describes these tasks. The current list of primitive tasks supported is based on Byrne et al. (1999).
- **Augmenters**: the tag `<augmentationtask/>` describes these elements. The list of augmenters depends on what is currently installed on the users' browser.
- **Composed tasks**: using the tag `<composedtask/>`, composed tasks are used to group a set of subtasks (either primitives or augmenters) in a single block.

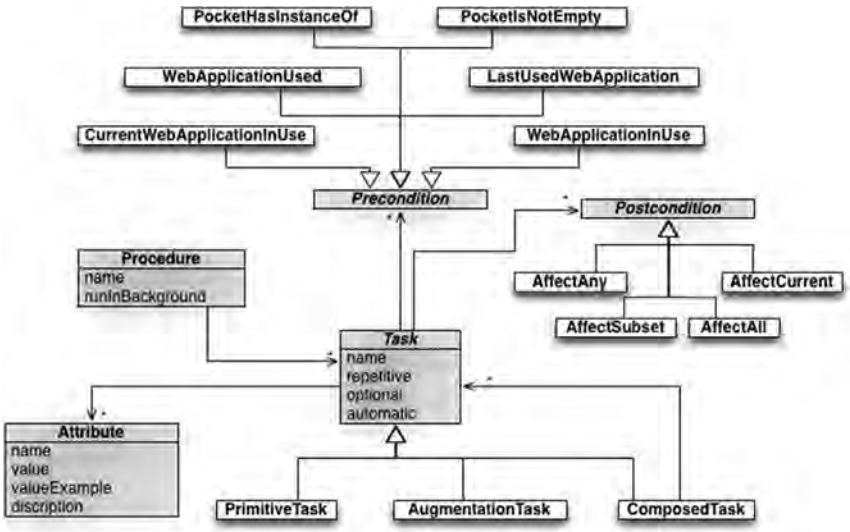


Fig. 8. The DSL metamodel.

- **Preconditions**: preconditions are used to decide whether (or not) tasks will be executed or not according to the information is available. These are described in the DSL with the tag `<precondition/>` which can be a:
  - **Precondition about collected data**: for conditioning the execution of a task according to the collected data. For example the precondition *PocketHasInstanceOf*() allows specifying that a task will be executed if the Pocket has an instance of a particular concept, for example, a *PointOfInterest* instance.
  - **Precondition about navigational history**: for conditioning the execution of a task according to the Web applications used. For example, *WebApplicationUsed* checks whether a particular Web site (defined by a parameter: a URL) ever appears in the browser navigational history.
- **Post-conditions**: post-conditions are specified to determine the effect of executing a particular task, and the tag `<post-condition/>` is used for this purpose. As shown by Fig. 8, there are four possible post-conditions. For example, *AffectCurrent* is used to specify that the execution will modify the current Web site.
- **Repetition, optional and automatic properties**: both Primitive and Augmentation tasks have three intrinsic properties. A *repetition* property specifies if the task may be executed more than once. The *optional* property allows skipping the execution of the current task. If the property *automatic* is *true*, then the player automatically triggers the task. In this case, all the attributes needed by the tasks need to have an associated value. Defaults values, *false* in the three cases, will be used if a task definition does not provide a value. In addition to these properties, each task can be set as ***synchronized***.
- **Attributes**: refer to data required to accomplish tasks. Ex.: the task *Provide URL* needs an attribute URL. Attributes, with their name and values, must be defined for each task in their specifications.
- **DUI component**: this property, only valid for Augmentation tasks, refers to DOM elements that may be extracted from Web sites and then associated with the task. The DUI component may be a specific DOM Element for static ones, and an xPath value in order to get the DOM element dynamically when the tasks is executed.

4.3. Tool support

We have developed tools as a proof of concept for our approach. Our tools help to edit, execute and synchronize the execution of procedures performed by multiple users on multiple Web sites. The tool presented hereafter is an extension of the previously described CSN framework, which allows developers to create and install new augmenters. The tools are delivered with a set of basic augmenters plus a set of primitive tasks that can be used in the composition of procedures. Fig. 9 shows the general architecture of our tools which relies on two main components:

- A task repository is a cloud service hosted in a Web server where users store tasks, augmenters, procedures and history information about the execution of procedures; and,
- A browser plugin available to the client that supports both the edition mode (allowing composing new sequences of tasks) and the execution mode (play task sequences).

In Fig. 9 we show how the user 1 can use the tool (i.e. task edition mode) in order to create a new procedure by composing basic tasks. Once the procedure is created, he can play and share his creation with other users. Once a procedure is created, the user can upload it into the repository. Other users can then download the procedure from the task repository and execute it in their Web browser using the plugin (i.e. task execution mode). It is noteworthy that the edition and execution of procedures are not supported simultaneously by our tools; indeed, a user cannot modify a procedure that is being executed by another user.

The synchronization between users, which is an important aspect for supporting collaboration, is implemented and delivered as part of the architecture of the tool. In the current implementation the user who creates and publishes a procedure will be notified when another user executes it. Usually, the synchronization between users will always occur at the end of the execution of a procedure. Nonetheless, tasks defined as ***synchronized*** will be updated in the task repository immediately after their execution in the client. Users can at any moment explore the task repository and check the current state of execution of the procedures they have published there.

The task repository has several responsibilities, including:

- Managing user profiles: users profiles are needed since several preferences and partners (other users with which usually collaborate in procedures executions) may be specified.
- Storing augmenters: this is a repository from where users can download and install new augmenters.
- Storing procedure: this is a procedure repository from where users can get procedures defined by other users.

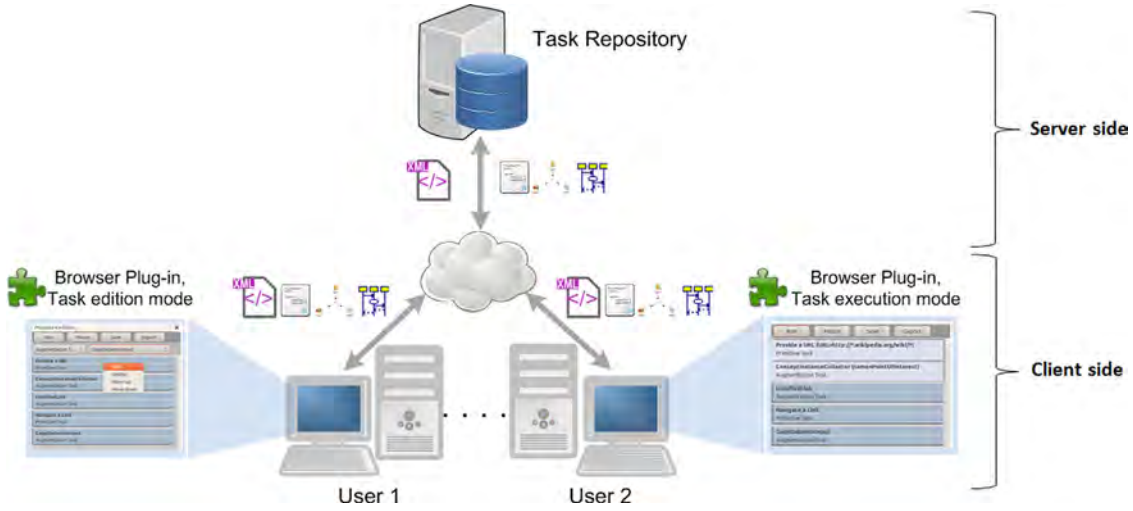


Fig. 9. Overall architecture of the approach describing the composition of tasks in the DSL.

- Storing procedure execution states: when users are collaborating in order to facilitate the procedure execution of each other, they can upload the states of their own execution; then, other users may take advantage of tasks which have been already executed with concrete data.
- Allowing synchronization between users: this is achieved by allowing users to share their procedures execution states with specific partners.

The client-side plugin allows users to:

- Edit and create new procedures by combining tasks (edition mode): this implies to modify the composition of tasks (both augmentation and primitive tasks) and the information these tasks are expecting to be executed.
- Execute a specific procedure (execution mode): by selecting one of the procedures locally installed, users may run it and then facilitate their tasks.
- Share both new procedures and information about execution of these: in the middle of an execution, we allow users to share the state of the procedure execution. Then, users may share with their partners the information used for the procedure.

The same client-side tool edits and executes procedures. Fig. 10 shows some screenshots of the tool in the edition mode. Each of the three screenshots (a, b and c) show at the top the main menu the buttons "run", "save" and "export", that correspond to commands related to the whole procedure being edited. Just above the main menu, Fig. 10a shows a vertical menu that is used to select a task that will be used in the composition of a procedure. The options: *Primitive Task*, which can be deployed in a second menu as shown by Fig. 10b; and *Augmentation Task*, as shown by Fig. 10c. These menus are built dynamically with the tasks locally available. If Augmentation tasks installed locally are not enough to perform some task, the user can install new augmenters. Once a particular task is selected; it is included in a list that appears just below these menus as shown by Fig. 11.

Using the tool in the edition mode, users can initialize tasks for working with a specific Web site. For example, Fig. 11a shows how the user edits the attribute "*Provide a URL*" task by accessing the task's options from the contextual menu. In this case, the task "*Provide a URL*" only has one attribute to be specified: the URL. Note that in Fig. 11b the URL entered is a regular expression (notice the "*"). This option is just for giving flexibility to the whole procedure. With this value for the attribute, the task would be considered completed once the user visits a Web site whose URL matches with this regular expression. The execution mode is illustrated by Fig. 11c. The tool shows five tasks, in which the first two (i.e. "*Provide a URL*" and "*ConceptInstanceCollector*") were already performed; this is made visible by the different colors used. In the case of "*Provide a URL*", the value has been changed to a specific Wikipedia article about Barcelona.

It is also possible to group sequences of tasks as shown in Fig. 12. For that, users must select two or more tasks, and then an option "Group tasks" on the contextual menu will appear. Similarly, if tasks are grouped, an option will allow ungrouping them. It is also possible to change the order of tasks in the list by selecting the options "*Move up*" or "*Move down*".

Once a procedure has been completed, users might share it with friends and colleagues. This task can be done by selecting the button "*Export*" as shown in Fig. 13. In edition mode (see Fig. 13a) the user can decide to: upload the procedure into the repository, send it via email to other users, or both. When the procedure is uploaded into the repository, it becomes available in a Web server for all users using the plugin. The option 'share with' is used when a procedure is intended to be shared with a few users. Fig. 13b shows the export window for the execution mode. In this mode, share the procedure means to allow other users to see with which information the user has executed the procedure. Since the user could have used confidential data for performing the tasks, he can choose from the list those tasks for which the attributes values must not be shared. This is made by selecting the tasks with private information from the list.

Finally, in Fig. 14 we show the main options to manage procedures. On the one hand, the main menu allows users to create a new procedure. On the other hand, by choosing "Manage procedures" option, the user can manage the procedures already installed in his browser. In Fig. 14, we show the only procedure installed, named "Conference trip planning". By selecting one of the procedures listed, the user can execute it, edit it, or see the previous executions made by himself and those executions shared by his partners.

4.4. Distributing UI components inside of procedure tool

In this section we show how both the approach and the tools support DUI by combining the use of some augmenters and some properties contemplated in the DSL. For that we propose two scenarios concerning (i) the inclusion of DOM elements into

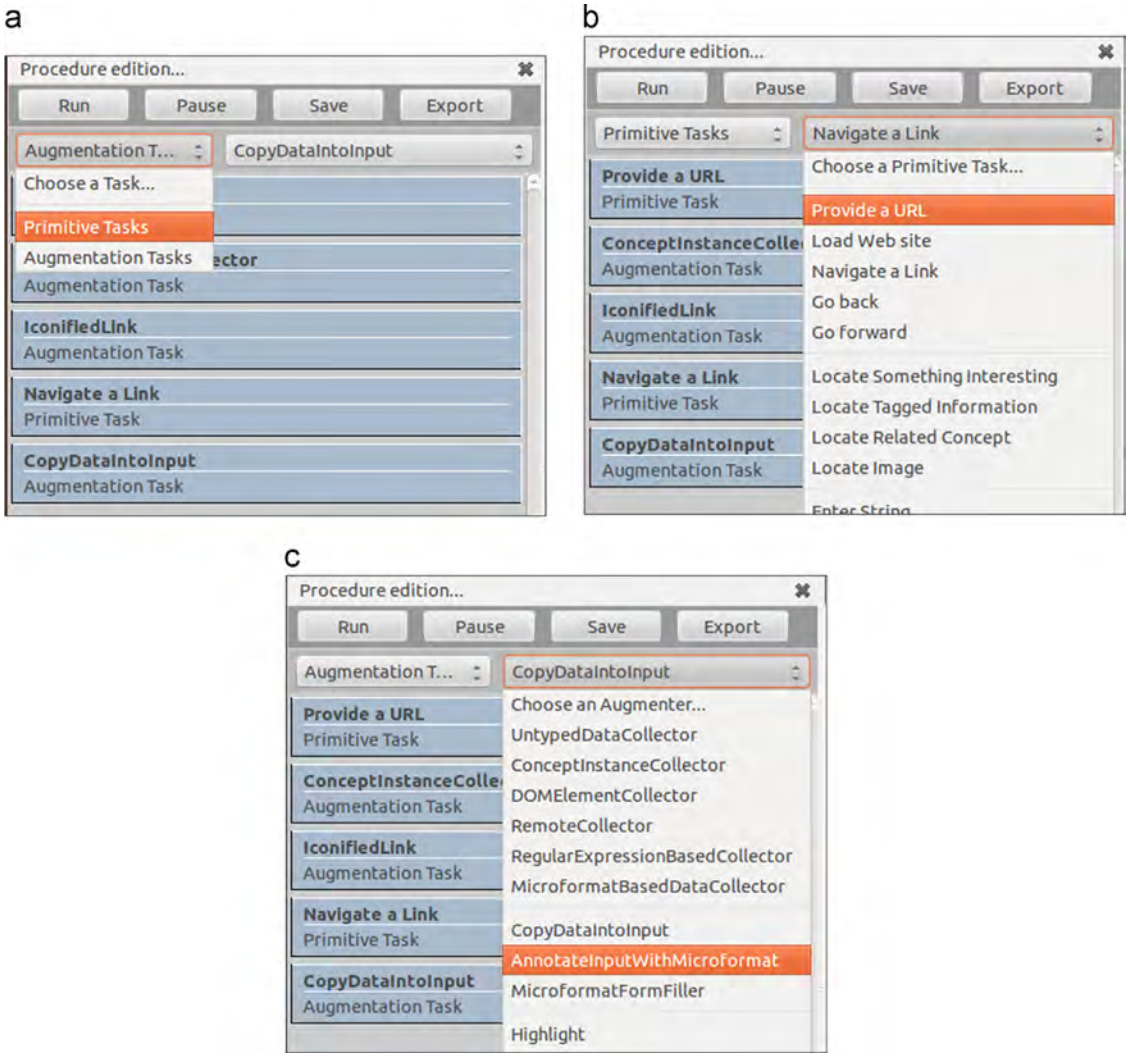


Fig. 10. Edition mode allowing task selection (either primitive tasks or augmenters). (a) Menu of tasks (primitive/augmenter). (b) Selection of primitive tasks. (c) Selection of an existing augmenter.

procedures and their corresponding interactors issued from the original Web site; and (ii) user collaboration via interaction with DUIs components embedded into procedures.

4.4.1. *Embedding DOM elements into procedures*

Users can use the procedure tool to collect graphical components of Web sites that are encoded as DOM elements and embed them into procedures. This operation is done from the window task edition as shown in Fig. 15. From there, users can select the option "*Select DOM Element*" and then explore the Web page to pick the corresponding DOM element. DOM elements are automatically detected by the tool and shown in highlight (in yellow) when the user moves the mouse over then. The selection of the DOM element is made by clicking on a highlighted area; this action makes the corresponding DOM elements embedded into the procedure tool as show by Fig. 15.

By collecting DOM elements, users are not only collecting information but they are also capturing the inner behavior necessary for interacting to the original Web site. Indeed, collected DOM elements might contain anchors, links and other interactors which users can use in the distribution of the user interface. Such as interactors becomes available when users share their procedures with other users. These elements are not only static copies DOM elements. To illustrate this, we provide in Fig. 16 a small scenario: let us assume a user wants to share for a while his Gmail mailbox with a friend without providing his login and password information. For that, the user creates a procedure embedding the form fields for logging into Gmail as illustrated by Fig. 16a. At first the user creates a task and associates the corresponding DOM element that contains all the form fields for performing a login into Gmail. The embedded DOM elements are shown in Fig. 16b. Then, the user types his login information on the procedure as shown in Fig. 16c. In the sequence, the user records the procedure and sends it his friend.

Fig. 17 illustrates the user's friend view of the procedure in the player mode. As we shall see, the procedure contains the DOM elements (i.e. the login) required to access a Gmail account in Spanish as this was the version of the Web site used for creating the procedure. Indeed, the first user created the procedure under an English version of Firefox running on Mac OS (Fig. 16) whilst his friend plays the procedure under an English version of Firefox running on Window 7 (Fig. 17). Assuming that the other user can read Spanish, the only thing he needs to do to access his friend's account is to click on the button "Acceder" (i.e. submit in Spanish) which is embedded into the procedure as part of the login form

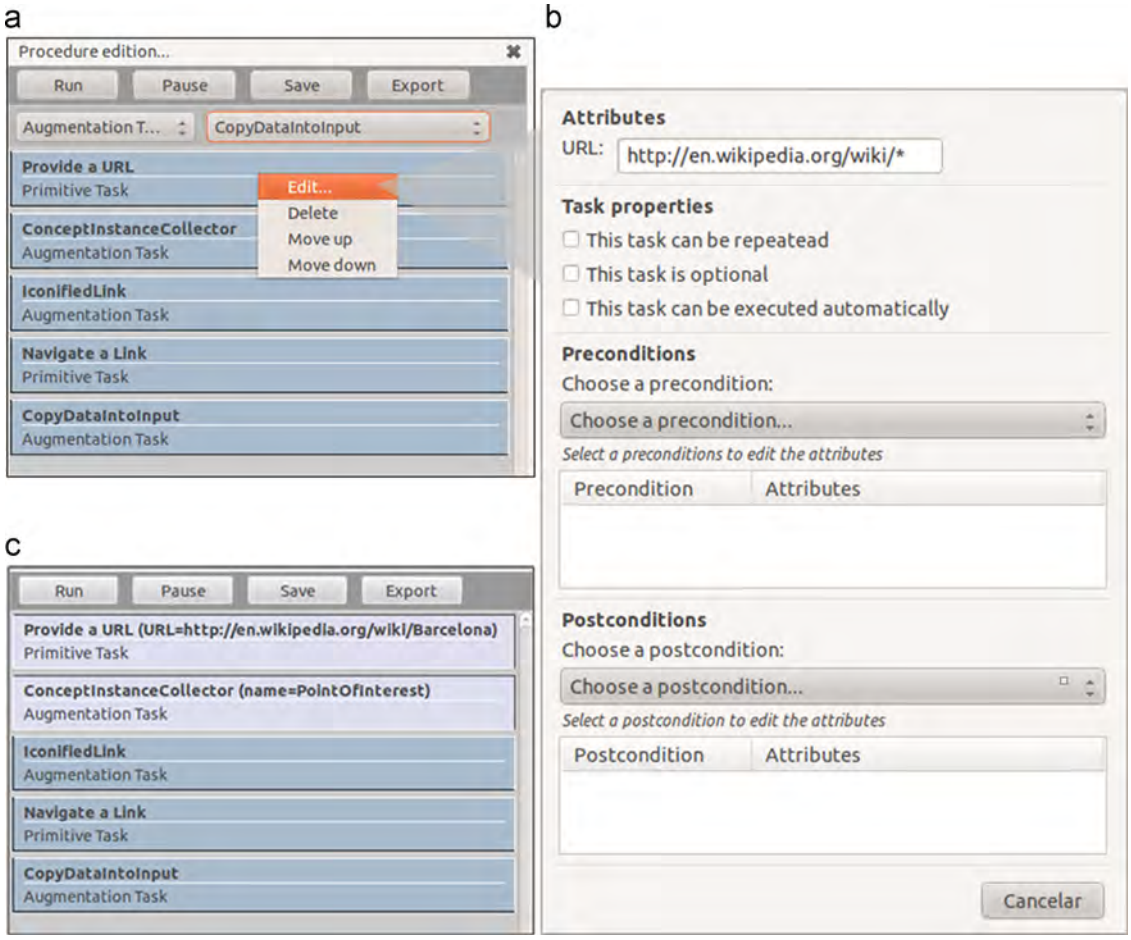


Fig. 11. Edition mode showing how to define properties of tasks (a and b) and execution mode (c). (a) Edition mode. (b) Edit task attributes. (c) List of tasks in the execution mode. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

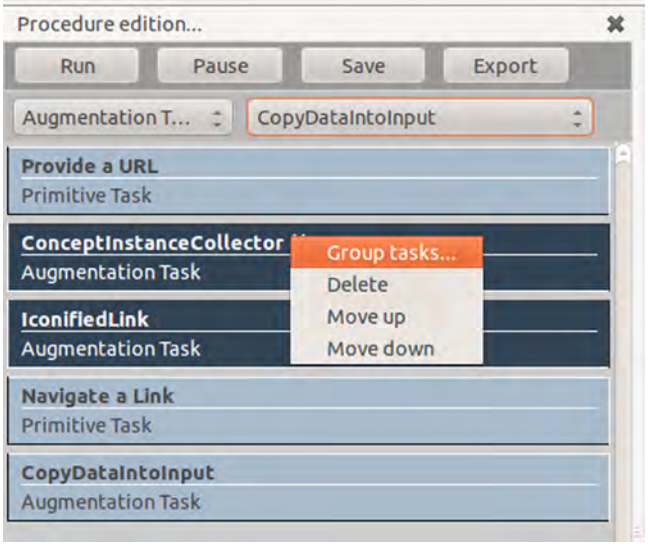


Fig. 12. Grouping tasks.

shown in Fig. 17; this action will grant the access to Gmail (as shown in the right side of Fig. 17) without knowing his friend's password. This scenario might look artificial at first sight but it illustrates how powerful our approach is with respect to the distribution of the user interface.

5. Case study

This section illustrates our approach by the means of a case study concerning a trip planning to attend a conference. Our main concern here is to demonstrate the creation and execution of complex procedures that combine both manual and automated tasks. For that, let us assume that two friends named Peter and John are planning to go together to a scientific conference. To accomplish this goal they should visit the conference Web site for getting general information (such as location, dates, etc.), collect points of interest at the destination, book flights and hotels. The first step is to define a sequence of minimal actions that they should do to accomplish this goal. Then, they can share informations during the process. Both Peter and John have previously installed the plugin as described in Section 4.3.

Peter decided to create a procedure as it is shown at the left-side of the browser window in Fig. 18. The procedure is built incrementally while Peter is navigating the different Web sites required to accomplish the task. As he collects data during his navigation, it becomes recorded as part of the procedure itself. For example, the first tasks in the list is "*Provide a URL*" where we can read the parameter "URL=http://www.interact2013.org" indicating the Web site visited by the user. We can also read that "*Provide a URL*" is a primitive task, which means that users should inform themselves which Web site they want to make part of the procedure. Because users might have many browser windows opened at the same time to perform different tasks, it is important to let users inform whether or not a specific Web site been visualized should be part of the procedure he has in mind. That

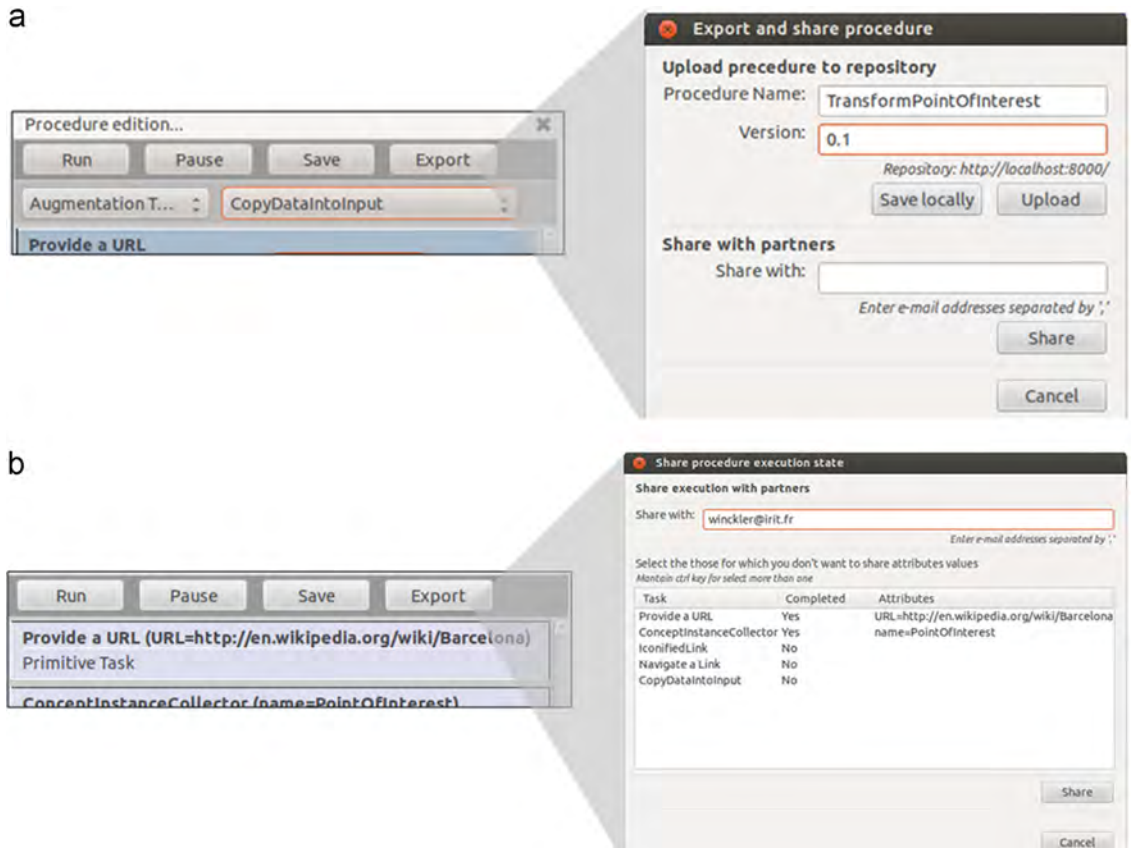


Fig. 13. Using functions *upload* and *sharing* in the tool. (a) Edition mode. (b) Execution mode.

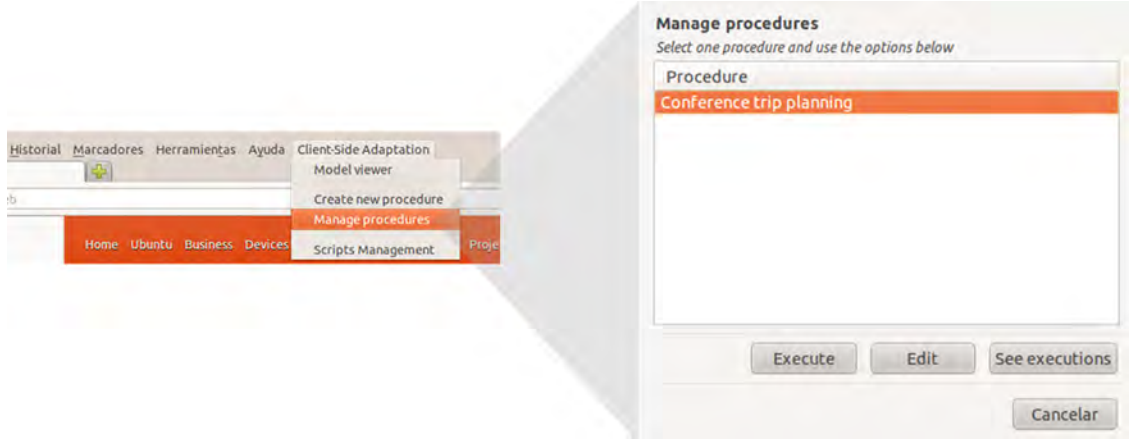


Fig. 14. Main menu and procedure manage window.

is why this task is set as *primitive* and therefore performed manually by the user.

As we shall see in Fig. 18, the second, third and fourth elements in the list (lines 2–4) concern the augmenter *ConceptInstanceCollector* that was used three times to collect data from the Web site; respectively the *destination*, *dateFrom* and *dateTo*. For that, the only thing Peter had to do is to point at information on the Web page and trigger the augmenter, so that a copy operation is done. The data became part of the procedure (see the parameters *name* and *value* next to the augmenter definition). Moreover, these data become available when Peter is visiting the Web site for booking his flight at expedia.com (line nine). As that data is recorded as part of the procedure Peter can reuse it when filling in the form for booking the flight; moreover, Peter can configure the augmenter *CopyDataIntoInput* to automatically fill in the form with the appropriate data as described in line 10 of Fig. 18. The data used for a procedure remain available during the entire execution. Fig. 19 shows the manual tasks (i.e. primitive tasks) that are defined to indicate what users have to do for the payment. We assume that Peter has represented these tasks to provide contextual help for accomplish the whole tasks. Indeed, in some case manual tasks do not need to be explicitly represented.

The whole procedure created by Peter is detailed in Table 2. Notice that tasks have been grouped and that for the sake of clarity the Web sites where tasks are performed have been included. The numbers in the first column indicate the order of execution of tasks. Preconditions and post-conditions have not been specified for the case study (Figs. 20 and 21).

So far we have shown how Peter has created the planning trip procedure. But then, he decides to share it with his friend John. For

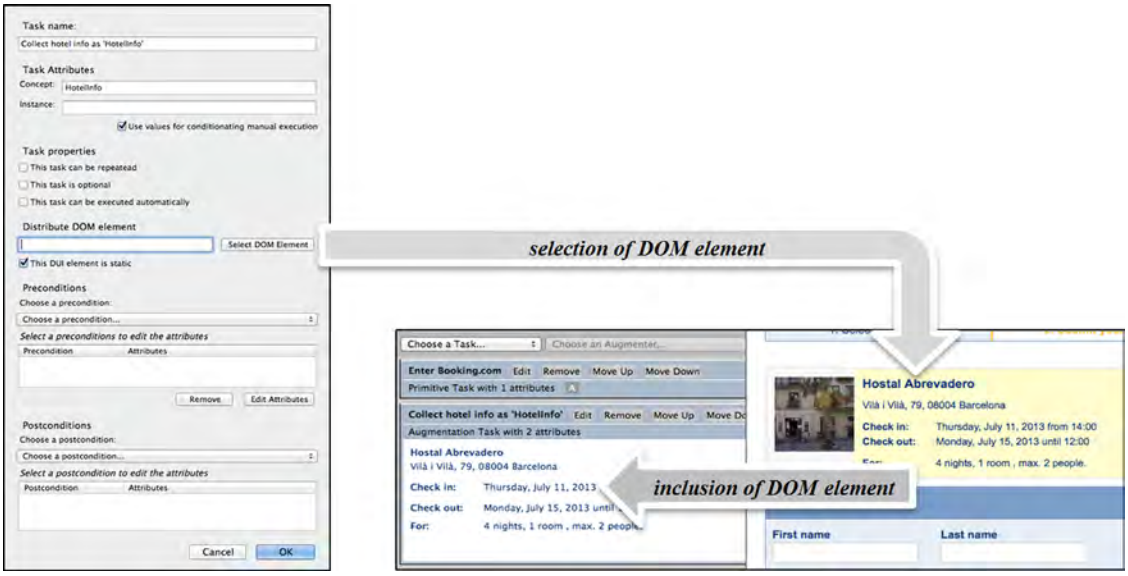


Fig. 15. DOM selection from Web site and later integration inside of procedure tool. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

that Peter triggers the option “export” in the main menu, execution mode (see Fig. 13b) and then a dialog menu appears with the list of tasks in that procedure. It is noteworthy that the procedure contains all data used by Peter, including his credit card information that he does not want to share with John (Fig. 20). So that Peter cleans the values associated to these tasks whilst sending the procedure to John.

John will be notified by email that a new procedure is made available by his friend John. By clicking in the URL provided in the email he can download that procedure that will appear in his browser as shown in Fig. 21. Notice that John sees the procedure in the execution mode of the tool. Most of the tasks contain attributes with data previously provided by Peter, which makes John's tasks easier to perform, except by data associated to tasks that Peter decided do not share with John.

When John finishes his procedure he can share this information with Peter. The procedure for sharing information is similar to that of sharing a procedure. The data is sent to the task repository that will notify Peter that John has being executing the procedure. By selecting the combo box as shown in Fig. 22, Peter can see the values he has used (Fig. 22a) and those provided by his friend John (Fig. 22b).

Note that, for example, the execution perform by John is not finished (note the two visible tasks in the list, which appear as not completed). Besides that, comparing both executions, we can see that John has used different information in the tasks; for instance, he has collected different *Point of Interest*.

6. Evaluation of the tools

The tools presented above are fully operational and they can be used to coordinate distributed user interactions over the Web. In order to highlight the contributions of these tools, we present in this section a preliminary evaluation.

The evaluation concerns the procedure player. We assume that user performance could be an important factor for adopting the tools proposed in this paper. To investigate this, we have performed a user testing experiment that was aimed at investigating whether users perceive our tool as usable and whether they were able to perform tasks faster using our tools than navigating individual Web sites. The user testing basically consisted of observing users' performance and comments whilst users accomplish tasks (Han and Tokuda, 2008). The study was run in a controlled room equipped with a PC running Unix and a Web browser Firefox 19 which integrates the procedure tool and a full set of augmenters. A chronometer was used to record user performance but sessions were not video-taped.

At first participants were asked to fill in a pre-questionnaire in order to collect demographic data and their experience with Web applications. Then, we have explained to the participants the goals of the study and they were asked to think aloud during the testing session. Participants were also introduced to our tools and get a short explanation about how it works. In order to make sure that all participants would receive the same information the instructions were provided by a 6 min video demonstrating the use of the procedure player.

Then we have asked participants to envisage a professional travel to the conference INTERACT with a colleague, named John, who is keen to share his trip plan with him. The travelers had to book a hotel in the city where the conference takes place. We assume that John had found the cheapest hotel offers at the Web site booking.com and that he was keen to share this hint. However, the participants had to make the arrangements for the trip alone. This general scenario encompasses the main tasks:

- Task 1 (T1): To get general information about the conference and the cheaper hotel.
- Task 2 (T2): To search at booking.com for a room in the hotel during conferences dates (check-in: first day of the conference; check-out: last day of the conference).
- Task 3 (T3): To book and pay for the room.

In order to compare the value added by our approach, we have asked participants to perform this trip planning with and without our tools. Whilst not using our tools, the trip planning featured a printed version of an email that contained all John's trip arrangements; then the participant had to visit the Web sites indicated in the email to perform the tasks. Whilst using our tools, the scenario included the trip planning made by John using the procedure editor; in this case John's email contained a link from where it was possible to download the procedure from the task repository. Thus, to execute the procedure the participant should have to perform the following additional task:

- Task 0: To import the procedure.

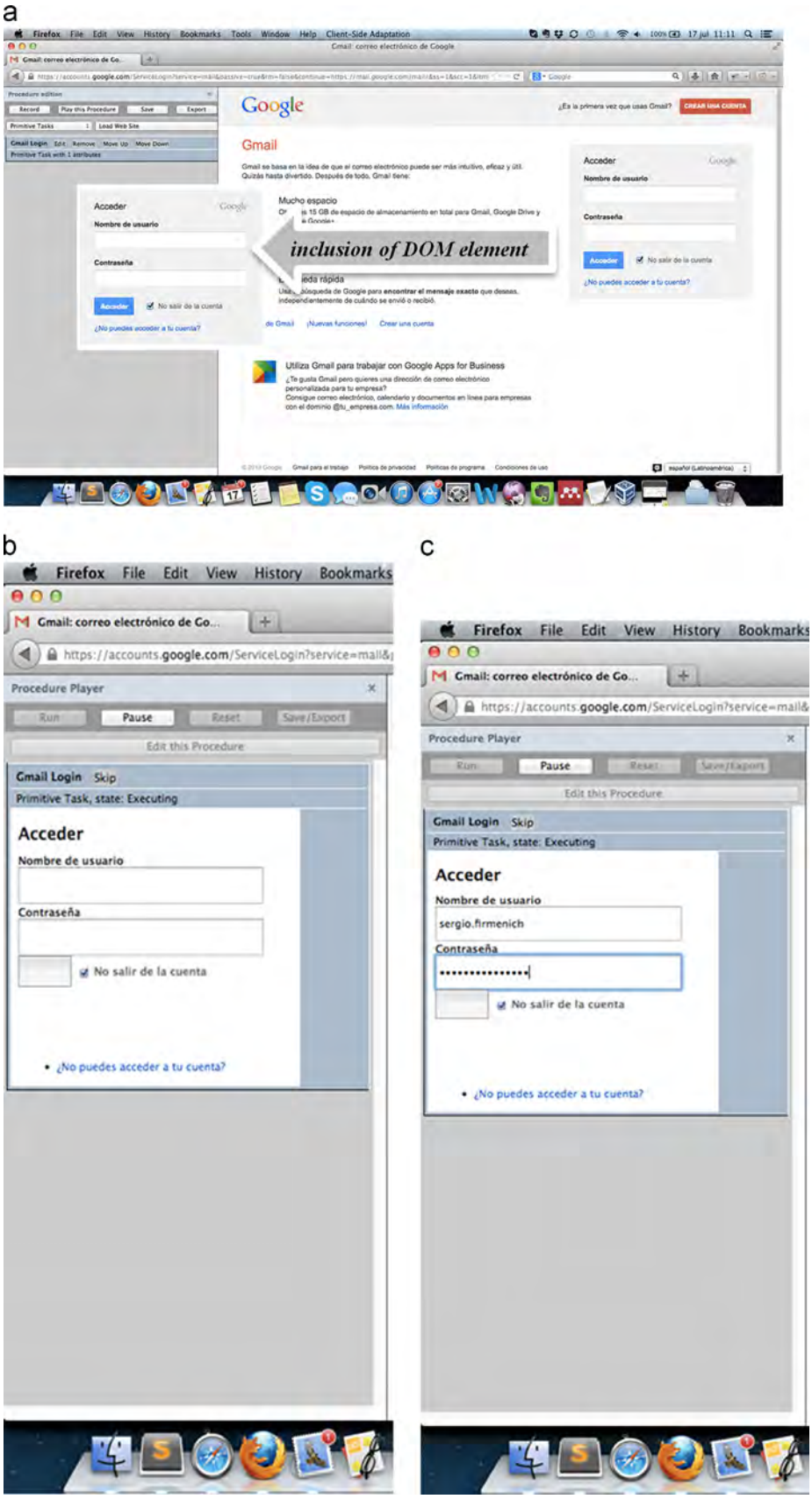


Fig. 16. Creation of a procedure embedding DOM element login using the procedure plugin in English version of Firefox for Mac OS, with Gmail in Spanish. (a) Inclusion of DOM element: login form. (b) Embedded DOM. (c) Customizing DOM.

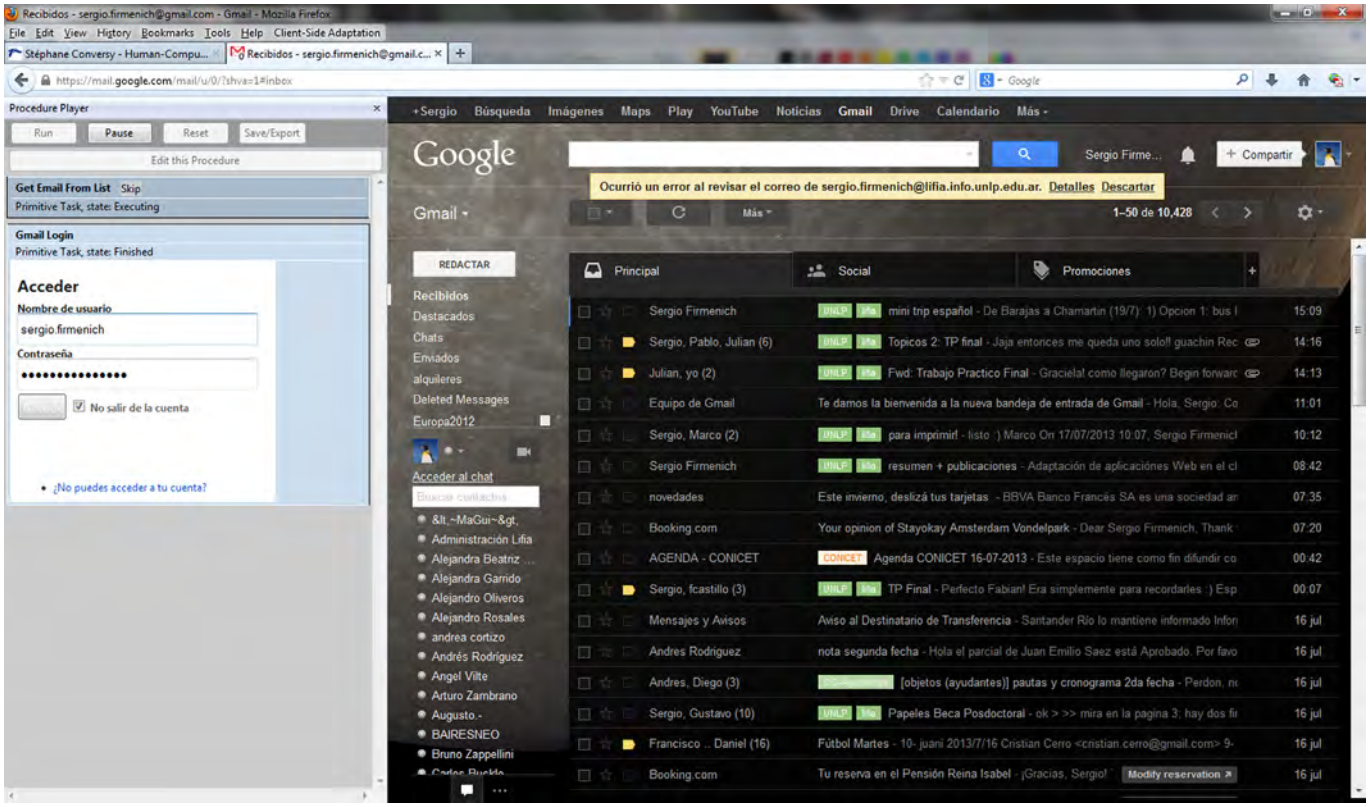


Fig. 17. Running the procedure plugin under an English version of Firefox for Windows 7.

Fig. 18. A procedure for planning a trip as it is visualized when navigating the Web. At the left-side, the plugin with the list of tasks (including both primitive and augmenters). At the right-site the current Web site.

After performing every task, participants were asked to assess from 1 to 5 the difficulty to perform the task (1 from very easy to 5 very difficult). In order to measure efficiency we took the time consumed by each user task. Effectiveness was measured in terms of the number of tasks users could finish.

At first they were asked to perform the tasks without the tools. Then they were allowed to perform the tasks using the procedure player. After accomplishing the tasks with and without the procedure player, participants were asked to fill in a form used to determine the satisfaction degree, including: what they liked in the tools, what they did not like, whether they experienced difficulties in activating the tools, which task were difficult to perform, whether (or not) they were aware of the actions using the tools. Finally, a System Usability Scale questionnaire (i.e. SUS (Brooke, 1996)) was used to determine tools usability.

For this study about the usability of our tools, eleven participants (9 males and 3 females, aged from 20 to 33 years old,

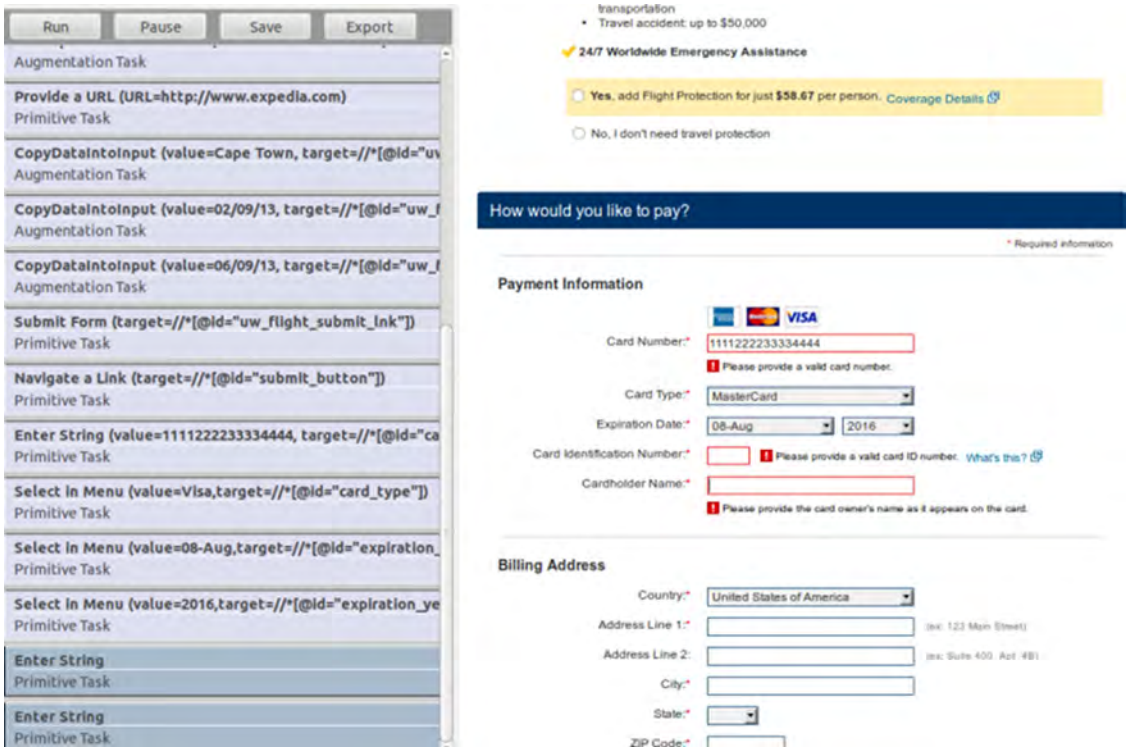


Fig. 19. Tasks related to payment in a procedure.

Table 2
Peter's procedure for planning a trip to attend a conference.

User tasks	Web site	Task (*primitive/augmenter*)	Attributes
1. Visit conference Web site	–	Primitive task: *provideURL*	url
2. Collect data about the conference	www.interact2013.org	*ConceptInstanceCollector*	conference place
		ConceptInstanceCollector	destination
		ConceptInstanceCollector	dateFrom
		ConceptInstanceCollector	dateTo
3. Visit site for getting information about destination	–	Primitive task: *provideURL*	url
4. Collect points of interest at destination	www.interact2013.org	∗*ConceptInstanceCollector*	point of interest
5. Visit site for booking a flight	–	Primitive task: *provideURL*	url
6. Search for flights	www.expedia.com	*CopyDataIntoInput*	destination
		CopyDataIntoInput	dateFrom
		CopyDataIntoInput	dateTo
		Primitive task: *submitForm*	
7. Go to payment page	www.expedia.com	Primitive task: *clickButton*	
8. Pay for flights booking	www.expedia.com	Primitive task: *enterString*	card number
		Primitive task: *selectMenu*	card type
		Primitive task: *selectMenu* Primitive task: *selectMenu*	expiration month
		Primitive task: *enterString*	expiration year
		Primitive task: *submitForm*	card holder
9. Collect data about flight booking	www.expedia.com	*ConceptInstanceCollector*	flights number
		ConceptInstanceCollector	flight set
10. Visit site for booking hotel	–	Primitive task: *provideURL*	url
11. Search hotels	www.booking.com	*CopyDataIntoInput*	destination
		CopyDataIntoInput	dateFrom
		CopyDataIntoInput	dateTo
		Primitive task: *submitForm*	
12. Pay for hotel booking	www.booking.com	Primitive task: *enterString*	card number
		Primitive task: *selectMenu*	card type
		Primitive task: *selectMenu* Primitive task: *selectMenu*	expiration month
		Primitive task: *enterString*	expiration year
		Primitive task: *submitForm*	card holder
13. Collect data about hotel		*ConceptInstanceCollector*	link of hotel's Web page

Legend: –: task can be performed from any Web site; ∗: task can be performed one or more times.

Average=27.64, SD=3.79) were recruited at the University of La Plata, Argentina. Among the participants we counted two PhD, 7 PhD students; the rest of them were undergraduate students. All participants were experienced Web users (i.e. >5 years using the Web) who browse the Web as a part of their daily activities. The pool of users was selected by convenience and it is not representative of the population of Web users. However, each user performed the manual task and used the

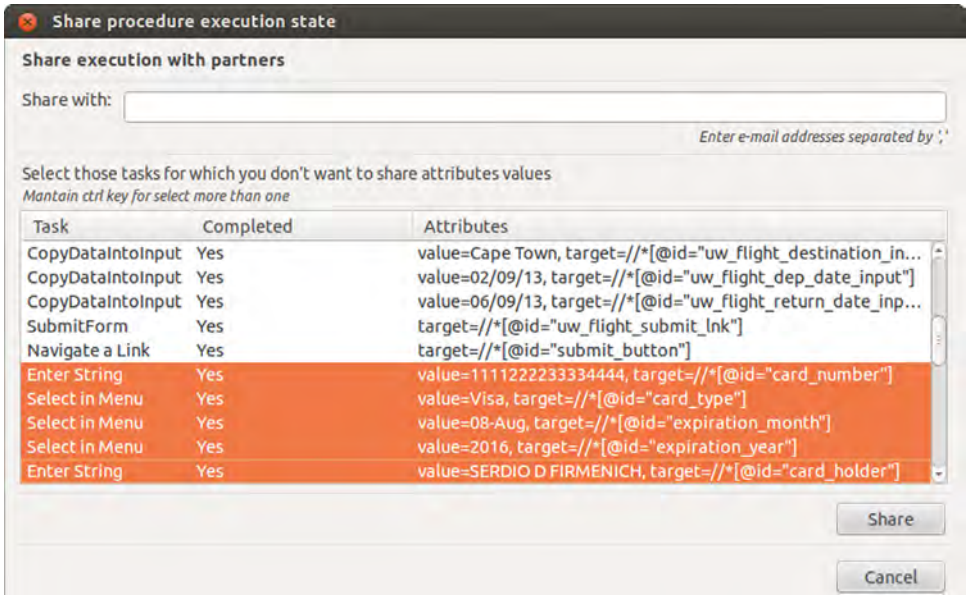


Fig. 20. Sharing a procedure with other users.

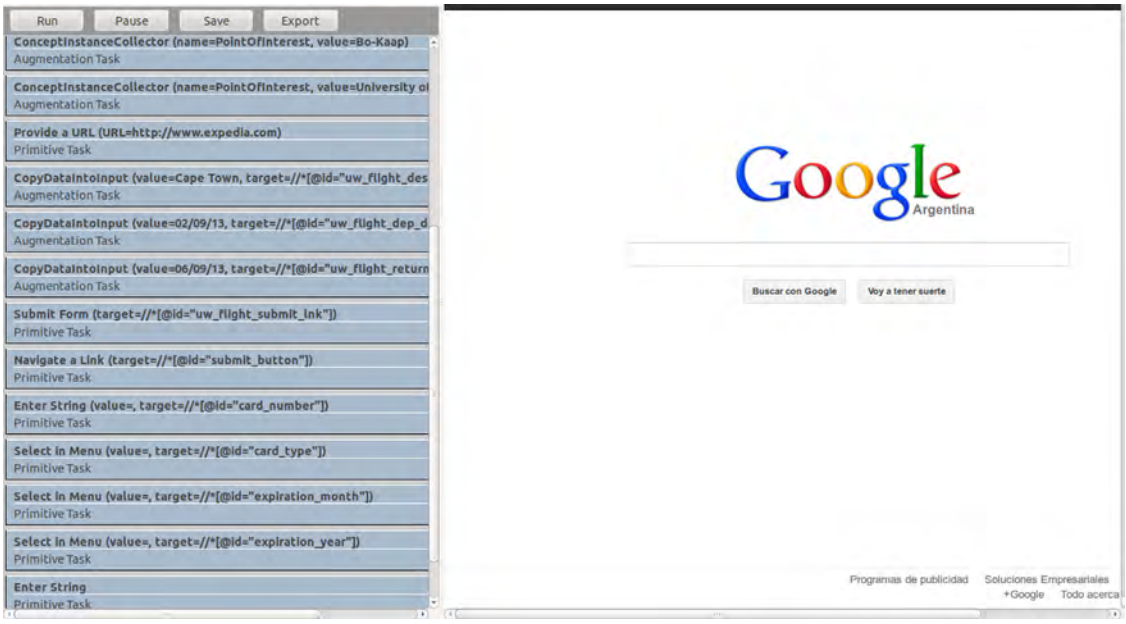


Fig. 21. Visualization of the procedure in the execution mode.

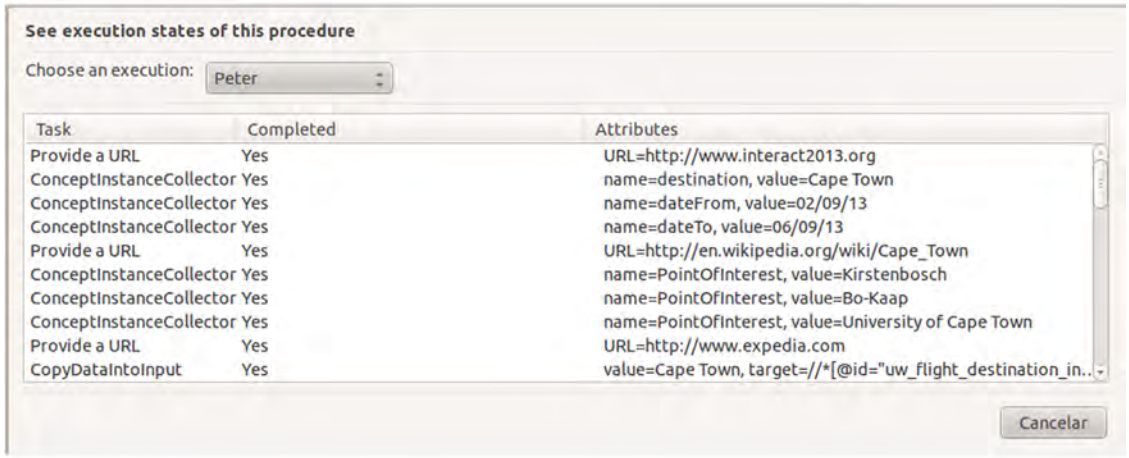
procedure player, so it is still possible to compare the relative user performance.

All participants in the study completed both manual tasks and used the procedure player. Overall the users judged the tasks with the procedure player easy to follow. In a scale from 1 to 5 (1 is very easy, 5 very difficult), the average score for tasks were: Task 0=2.6, Task 1=1.4, Task 2=1.6, and Task 3=1.18. As indicate in Fig. 23 the task performance (except of the task 0) was improved using our tools.

All participants reported positive comments about the tools. The overall the results provided by SUS were also positive with an average SUS score of 75.2. However, three participants given score below 70 points to the SUS (i.e. 50, 55 and 67.5) and a single user gave 100 points to the procedure paper. Moreover, participants provided several suggestions for improving the usability of the tool such as providing explicitly mark for informing when the set of tasks has been finished. In general participants think that the tools help to navigate among the diverse site Web and they appreciate the automation of tasks. Despite the fact that the user input was distributed between the procedure player and the Web sites, none of the participants raised comments about the fact that these two ways of interacting with Web sites would be conflicting. However, some of them ($N=5$) mentioned they would be appreciate it if the tools could automate the tasks without asking users to follow the list.

These results are rather preliminary. They do not take into account a detailed analysis of the synchronization of tasks between the users. Moreover, it only covers the execution of procedures. Yet it shows some evidence of use for the tools and an initial positive feedback. Indeed, the study shows that users can very easily perform collaborative tasks in distributed Web sites. Users indeed appreciated the support for planning tasks in an integrated manner. User performance was greatly improved using the approach. However, further studies will be required to investigate the usability of the tool with a larger population.

a



b

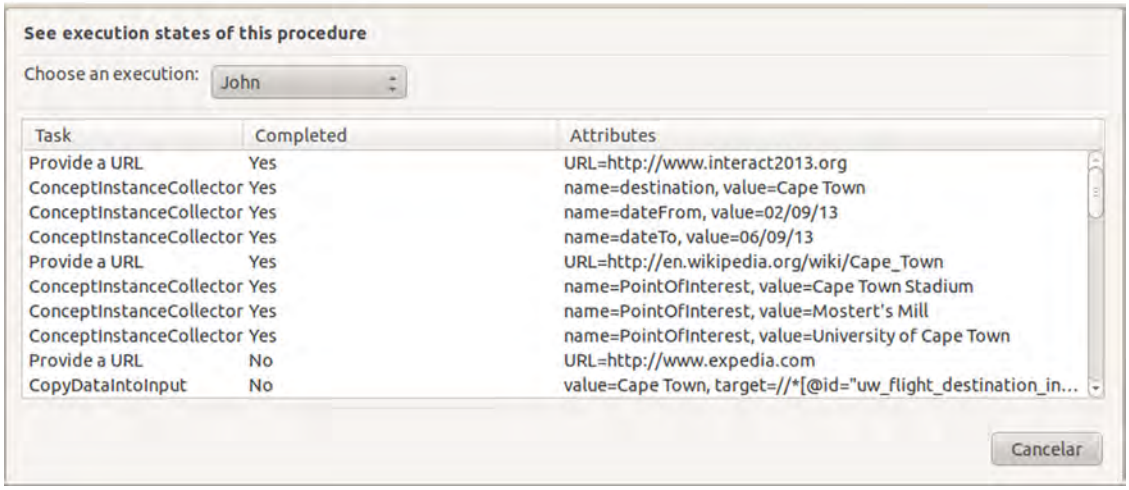


Fig. 22. Visualization of values used in procedure. By selecting the user in combo box, Peter can compare his values with the values used John. (a) Values used by Peter to accomplish the procedure. (b) Values used by John to accomplish the procedure.

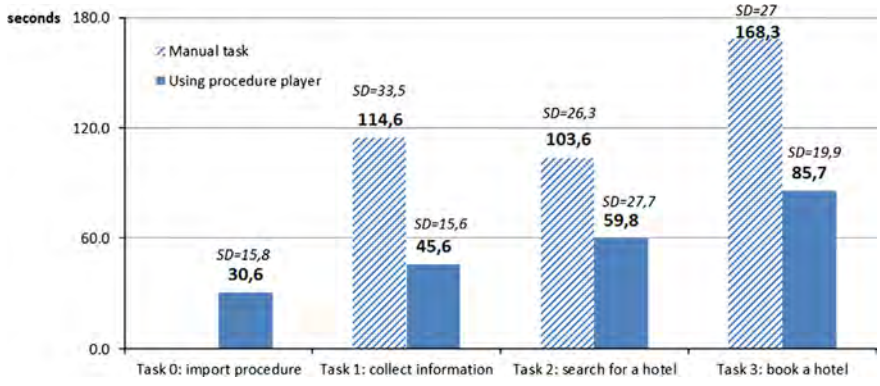


Fig. 23. Average user performance for accomplishing tasks manually and with the procedure player.

7. Conclusions, lessons learned and future work

In this paper we have presented a novel approach for building Distributed User Interfaces (DUIs) aimed at helping users to collaborate whilst navigating multiple Web sites. These DUIs feature a list of tasks that are constructed by assembling building blocks from a repertoire of manual Web tasks (called primitive tasks here) and a set of components called Web augmenters that automate user tasks. By assembling these tasks, users can create their own procedure. The composition mechanism can be described by a dedicated DSL and supported by a tool featuring a plugin. The distributed aspect of this interface is duly discussed in Section 7.

The key aspect of this contribution is the fact that our approach is able to include small pieces of software components, called Web augmenters to support tasks users performing over the Web. By assembling these elements in a list, we argue that we can create a dedicated procedure featuring a distribute user interface that helps users to accomplish a given goal. It is important to notice that, to reach some goals, users still have to act on the Web browser. For that, the approach allows the inclusion of primitive tasks which are used in the composition to provide a kind of contextual help, so that users will know what to do whilst navigating a particular Web site. The DSL that is delivered as part of our approach allows formalizing the task composition. It is noteworthy that this is only possible because the repertoire of tasks users can perform are limited by the number of actions users can do on a Web browser. The elementary user tasks of a Web browser are simple and very well known in the literature (Byrne et al., 1999; Heath, 2010). Web augmenters can automate some of these tasks and perform adaptations on Web pages to be visualized in the Web browser. These new components let users perform additional tasks that go beyond of traditional Web users. Nonetheless, these are extensions that should be coded by developers. The impact of the developer community in developing scripts is known in the GreaseMonkey community (GreaseMonkey, 2012) and motivates this kind of research.

Some of the aspects of procedure modeling and execution discussed in this paper might resemble some techniques for end-user programming (Lieberman et al., 2006; Lin et al., 2009). In the past, experiments with mashups tools for end-user programming (Lin et al., 2009) have demonstrated that some visual tools such as CoScript (Bogart et al., 2008) could be used to create scripts integrating data from many Web sites. Nonetheless, we consider that a direct comparison would be unfair as the skill required to compose procedures with Web augmenters is more close to what we expect from Web developers than from ordinary Web users.

The current implementation is limited to the information exchange between users and the tasks they have performed. In the section case study, we have shown how two users can report to each other the values they have used to accomplish a trip planning. However, it would be possible to envisage a more fine-grained collaboration on which users could distribute the tasks to be performed in a procedure. We are currently working on our tool to increase the level of interaction between tasks performed by users. One of the aspects that we are investigating is to merge procedures developed by two (or more) users, where each procedure contains tasks that are performed by individuals in order to reach a high level goal that is represented by the combination of several procedures.

Given the fact that this approach is very new, the tools have only been used by a limited number of users that were specially recruited to an early trial. Nonetheless they seem enough to prove the overall concept. Future work will include user testing of the tools with a larger population. Such empirical study should investigate in detail the usability of the tool in creation mode and the synchronization aspects of tasks between users. Moreover, we are planning to investigate how users perceive the introduction of new Web augmenters into the plugin.

In parallel we are working on the development of new augmenters that could be supporting a higher level of automation of user tasks. In the near future we are planning to make the tools accessible for the public so that we can start investigating the usability and the user experience with our tools. We are also planning to integrate more function in our tools for supporting a better communication between the users during the execution of distributed procedure. In a long term run we want to investigate the use of procedure descriptions as a support to the analysis of the user activity over multiple Web sites; by doing so we hope that we could optimize the processes used by users to accomplish their tasks on multiple Web sites.

Rather than a definitive solution to the problem of integrating data among multiple Web sites and supporting users collaboration whilst navigating Web application, this work proposes new challenges for the development of Distributed User Interfaces over the Web. For example, how to model and describe user tasks that can be scattered in multiple Web sites? How to help users perform these tasks efficiently? How to automate user tasks over the Web? How to build interfaces that help users to share information about their activity over the Web with their friends and colleagues? How many pieces of the user interface of Web sites can be extracted and then rearranged to feature a new sequence of tasks? How to deal with Web technology to provide better support to users' tasks on multiple Web sites? Certainly much work remains to be done but the results that we obtained with our approach are promising for investigating these questions.

Annex. XSD Specification of the DSL

```
< ?xml version = “1.0”003F >
< xs:schema xmlns:xs = “http://www.w3.org/2001/
 XMLSchema”
    ttargetNamespace = “http://www.lifia.info.unlp.edu.ar”
    xmlns = “http://www.lifia.info.unlp.edu.ar” >
  < xs:element name = “procedure” >
    < xs:complexType >
    < xs:sequence minOccurs = “1” maxOccurs = “1” >
      < !– SYNCHRONIZATION!– >
        < xs:element name = “configuration” >
        < xs:complexType >
          < xs:element name = “server” type = “xs:string”/ >
          < xs:element name = “executionId” type = “xs:
string”/ >
          < /xs:complexType >
        < /xs:element >
        < !– TASK DEFINITION!– >
        < xs:element name = “tasks” >
        < xs:complexType >
          < xs:sequence minOccurs = “1”
maxOccurs = “unbounded” >
              < xs:element name = “tasks” >
              < xs:complexType >
                < xs:all minOccurs = “1” >
                  < xs:element name = “primitiveTask”
minOccurs = “0” maxOccurs = “*”/ >
                    < xs:complexType >
                      < xs:group ref = “taskDefinition”/ >
                      < /xs:complexType >
                  < /xs:element >
                  < xs:element ref = “augmentationtask”
minOccurs = “0” maxOccurs = “*”/ >
                    < xs:complexType >
                      < xs:group ref = “taskDefinition”/ >
                      < /xs:complexType >
                  < /xs:element >
                  < /xs:all >
                < /xs:complexType >
              < /xs:element >
            < /xs:sequence >
          < /xs:complexType >
        < /xs:element >
      < /xs:sequence >
    < /xs:complexType >
  < /xs:element >
```

```
<xs:group name="taskDefinition">
  <xs:sequence>
    <!- PROPERTIES!->
    <xs:attribute name="id" type="xs:string"/>
    <xs:attribute name="repetitive" type="xs:string"/>
    <xs:attribute name="optional" type="xs:string"/>
    <xs:attribute name="automatic" type="xs:string"/>
    <xs:attribute name="synchronize" type="xs:string"/>

    <!- ATTRIBUTES!->
    <xs:element name="attributes" minOccurs="0"/>
      <xs:complexType>
        <xs:all minOccurs="0" maxOccurs="*">
          <xs:element name="attribute">
            <xs:complexType>
              <xs:attribute name="id" type="xs:string"/>
              <xs:sequence>
                <xs:element name="name" type="xs:
string"/>
                <xs:element name="type" type="xs:
string"/>
                <xs:element name="value" type="xs:
string"/>
                <xs:element name="valueExample"
type="xs:string"/>
              </xs:sequence>
            </xs:complexType>
          </xs:element>
        </xs:all>
      </xs:complexType>
    </xs:element>
    <!- PRECONDITIONS!->
    <xs:element name="preconditions" minOccurs="0"/>
      <xs:complexType>
        <xs:all minOccurs="0" maxOccurs="*">
          <xs:element name="precondition">
            <xs:complexType>
              <xs:element name="type" type="xs:
string">
                <xs:simpleType>
                  <xs:restriction base="xs:string">
                    <xs:enumeration
value="LastUsedWebApplicationIs"/>
                    <xs:enumeration
value="WebApplicationInUse"/>
                    <xs:enumeration
value="WebApplicationUsed"/>
                    <xs:enumeration
value="PocketHasInstanceOf"/>
                    <xs:enumeration
value="PocketIsNotEmpty"/>
                  </xs:restriction>
                </xs:simpleType>
              </xs:element>
              <xs:sequence minOccurs="1"
maxOccurs="*">
                <xs:element name="name" type="xs:
string"/>
                <xs:element name="type" type="xs:
string"/>
                <xs:element name="value" type="xs:
string"/>
              </xs:sequence>
            </xs:complexType>
          </xs:element>
        </xs:all>
      </xs:complexType>
    </xs:element>
    <!- POSTCONDITIONS!->
    <xs:element name="postconditions" minOccurs="0"/>
      <xs:complexType>
        <xs:all minOccurs="0" maxOccurs="*">
          <xs:element name="postcondition">
            <xs:complexType>
              <xs:element name="type" type="xs:
string">
                <xs:simpleType>
                  <xs:restriction base="xs:string">
                    <xs:enumeration
value="AffectCurrent"/>
                    <xs:enumeration value="AffectAny"/>
                    <xs:enumeration
value="AffectSubset"/>
                    <xs:enumeration value="AffectAll"/>
                  </xs:restriction>
                </xs:simpleType>
              </xs:element>
              <xs:sequence minOccurs="1"
maxOccurs="*">
                <xs:element name="name" type="xs:
string"/>
                <xs:element name="type" type="xs:
string"/>
                <xs:element name="value" type="xs:
string"/>
              </xs:sequence>
            </xs:complexType>
          </xs:element>
        </xs:all>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
</xs:group>
</xs:schema>
```

References

Amershi, S., Morris, M.R., 2008. CoSearch: a system for co-located collaborative web search. In: Proceedings of the Twenty-Sixth Annual SIGCHI Conference on Human Factors in Computing Systems (CHI '08). ACM, New York, NY, USA, pp. 1647–1656. doi:10.1145/1357054.1357311. ⟨http://doi.acm.org/10.1145/1357054.1357311⟩.

Bogart, C., Burnett, M., Cypher, A., Scaffidi, C., 2008. End-user programming in the wild: a field study of CoScripter scripts. In: Proceeding of EEE Symposium on Visual Languages and Human-Centric Computing. Germany, pp. 39–46.

Bolin Michael, Webber Matthew, Rha Philip, Wilson Tom, C. Miller Robert, 2005. Automation and customization of rendered web pages. In: Proceedings of the Eighteenth annual ACM Symposium on User Interface Software and Technology (UIST '05). ACM, New York, NY, USA, pp. 163–172. http://dx.doi.org/10.1145/1095034.1095062.

Bouvin, N.O., 1999. Unifying strategies for web augmentation. In: Proceedings of the 10th ACM Conference on Hypertext and Hypermedia.

Brooke, J., 1996. SUS: A 'Quick and Dirty' Usability Scale. In: Usability Evaluation in Industry. Taylor and Francis, London.

Brusilovsky, P., 2007. Adaptive navigation support. The adaptive web: Peter Brusilovsky, Alfred Kobsa, and Wolfgang Nejdl (Eds.). Lecture Notes In Computer Science,vol. 4321, Springer-Verlag, Berlin, Heidelberg, pp. 263–290.

Brusilovsky P., Kobsa A., Nejdl W., 2007. The Adaptive Web: Methods and Strategies of Web Personalization, vol. 4321, Springer-Verlag LNCS, Berlin, Heidelberg.

Byrne, M.D., John, B., Wehrle, N., Crow, D., 1999. The tangled Web we wove: a taskonomy of WWW use. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: the CHI is the Limit (CHI '99). ACM, New York, NY, USA, pp. 544–551. doi:10.1145/302979.303154. ⟨http://doi.acm.org/10.1145/302979.303154⟩.

Card, S., Moran, T., Newell, A., 1983. The Psychology of Human–Computer Interaction. Lawrence Erlbaum Associates, Hillsdale, NJ. (448 pp).

Daniel, F., Casati, F., Soi, S., Fox, J., Zancarli, D., Shan, M., 2009. Hosted universal integration on the web: the mashArt platform. In: Proceedings of ICSOC/ServiceWave. Stockholm, pp. 647–648.
Demeure, A., Sottet, J.S., Calvary, G., Coutaz, J., Ganneau, V., Vanderdonckt, J., 2008. The 4C reference model for distributed user interfaces. In: Proceedings of the International Conference on Autonomic and Autonomous Systems, IEEE Explore, Piscataway, pp. 61–69.
Diaper, D., Stanton, N.A. (Eds.), 2004. The Handbook of Task Analysis for Human–Computer Interaction. Lawrence Erlbaum Associates, Mahwah, New Jersey, USA.
Diaz, O., Arellano, C., Iturrioz, J., 2010. Interfaces for scripting: making greasemonkey scripts resilient to website upgrades. In: Proceeding of ICWE2010. Springer, Vienna, pp. 233–247.
Elmqvist, N., 2011. Distributed user interfaces: state of the art. In: Gallud, J.A., et al. (Eds.), Distributed User Interfaces: Designing Interfaces for the Distributed Ecosystem, Human–Computer Interaction Series. Springer-Verlag, Berlin, Heidelberg, pp. 2011–2012.
Firmenich, S., Rossi, G., Urbieta, M., Gordillo, S., Challiol, C., Nanard, J., Nanard, M., Araujo, J., 2010. Engineering concern-sensitive navigation structures. Concepts, tools and examples. Journal of Web Engineering 9 (2), 157–185.
Firmenich, S., Winckler, M., Rossi, G., Gordillo, S., 2011. A crowdsourced approach for concern-sensitive integration of information across the web. Journal of Web Engineering (JWE) 10 (4), 289–315.
Gallud, J.A., Tesoriero, R., Penichet, V.R.M. (Eds.), 2011. Distributed User Interfaces Designing Interfaces for the Distributed Ecosystem. Human-Computer Interaction Series. Springer, Berlin, Heidelberg.
GreaseMonkey, 2012. ⟨http://www.greasespot.net/⟩ (last accessed 13.07.12).
Han, H., Tokuda, T., 2008. A method for integration of web applications based on information extraction. In: Proceedings of ICWE. Springer, New York, pp. 189–195.
Laurillau, Y., Nigay, L., 2002. Clover architecture for groupware. In: Proceedings of the 2002 ACM Conference on Computer Supported Cooperative Work (CSCW '02). ACM, New York, NY, USA, pp. 236–245.
Little, G., Lau, T., Cypher, A., Lin, J., Haber, E., Kandogan, E., 2007. Koala: capture, share, automate, personalize business processes on the web. In: Proceedings of the Conference on Human Factors in Computing Systems, CHI. ACM, San Jose, California, April 2007, pp. 943–946.
Limbourg, Q., Pribeanu, C., Vanderdonckt, J., 2001. Towards uniformed task models in a model-based approach. In: Johnson, Chris (Ed.), Proccedings of the 8th Workshop on Interactive Systems: Design, Specification, and Verification (DSV-IS '01). Springer-Verlag, London, UK, pp. 164–182.
Luyten, K., Coninx, K., 2005. Distributed user interface elements to support smart interaction spaces. IEEE Symposium on Multimedia. Irvine, California, USA, December 12-14, 2005.
Lieberman, Paternò, Wulf, (Eds.), 2006. End User Development. Series: Human–Computer Interaction Series, vol. 9, XVI, Springer, 492 Berlin, Heidelberg.
Lin, J., Wong, J., Nichols, J., Cypher, A., Lau, T.A., 2009. End-user programming of mashups with vegemite. In: Proceedings of the 14th International Conference on Intelligent User Interfaces (IUI '09). ACM, New York, NY, USA, pp. 97–106.
Manca, M., Paterno, F., 2011. Distributed user interfaces with Maria. In: Gallud, J.A., Tesoriero, R., Penichet, V.R.M. (Eds.), Distributed User Interfaces Designing Interfaces for the Distributed Ecosystem. Human-Computer Interaction Series. Springer, Berlin, Heidelberg, pp. 33–40.
Martinie, C., Palanque, P., Winckler, M., 2011. Structuring and composition mechanisms to address scalability issues in task models. In: Proceedings of INTERACT (3). Springer LNCS, pp. 589–609.
Melchior, J., Vanderdonckt, J., Van Roy, P., 2011. A model-based approach for distributed user interfaces. In: Proceedings of the 3rd ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS '11). ACM, New York, NY, USA, pp. 11–20. http://dx.doi.org/10.1145/1996461.1996488.
Morris, M.R., Horvitz, E., 2007. Search together: an interface for collaborative web search. In: Proceedings of the UIST '07, October 7–10, 2007, ACM Press, Newport, Rhode Island, USA, New York, USA. pp. 3–12.
Morris, M.R., 2008. A survey of collaborative web search practices. In: Proceedings of the 26th SIGCHI Conference on Human Factors in Computing Systems (CHI '08). ACM Press, pp. 1657–1660. doi:10.1145/1357054.1357312. ⟨http://doi.acm.org/10.1145/1357054.1357312⟩.
MozillaUbiquity, 2012. At: ⟨http://mozillalabs.com/ubiquity/⟩ (last accessed 13.07.12).
Operator, 2012. ⟨https://addons.mozilla.org/en-US/firefox/addon/operator/⟩ (last accessed 13.07.12).
Paternò, F., Mancini, C., Meniconi, S., 1997. Concur task trees: a diagrammatic notation for specifying task models. In: Proceedings of Interact'97. Chapman & Hall, Sydney, Australia, pp. 362–369.
Paternò, F., Zini, E., 2004. Applying information visualization techniques to visual representations of task models. In: Proceedings of TAMODIA '04. ACM, New York, NY, USA, pp. 105–111.
Paul, S.A., Morris, M.R., 2009. CoSense: enhancing sensemaking for collaborative web search. In: Proceedings of the 27th International Conference on Human Factors in Computing Systems (CHI '09). ACM, New York, NY, USA, pp. 1771–1780. http://dx.doi.org/10.1145/1518701.1518974.
Pilgrim, M., 2005. Greasemonkey Hacks – Tips and Tools for Remixing the Web with Firefox. O'Reilly.
Rädle, R., Jetter, H.-C., Reiterer, H., 2012. TwisterSeacrh: a distributed user interfac fro collaborative Web search. In: Proceedings of the 2nd Workshop on Distrubuted User Interfaces (in conjunction with CHI 2012). Austin, Texas, USA, May 5th 2012.
Selenium, 2012. Available at: ⟨http://jroller.com/selenium/⟩ (last accessed 14.07.12).
Schmid, O., Masson, A.L., Hirsbrunner, B., 2012. Collaborative web browsing: multiple users, multiple pages, concurrent access, one display. In: Proceedings of the fourth ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS '12). ACM Press, New York, USA, pp. 141–150.doi:10.1145/2305484.2305508. < ⟨http://doi.acm.org/10.1145/2305484.2305508 > .
Scriptish, 2012. ⟨http://scriptish.org/⟩ (last accessed 14.07.12).
Tom, Heath, 2010. A Taskonomy for the Semantic Web. Semant. web 1, 1,2 (April 2010), pp. 75–81.
Vanderdonckt, J., 2010. Distributed user interfaces: how to distribute user interface elements across users, platforms, and environments. In: Garrido, J.L., Paterno, F., Panach, J., Benghazi, K., Aquino, N. (Eds.)Proceedings of XIth Congreso Internacional de Interacción Persona-Ordenador Interacción 2010 (Valencia, 7–10 September 2010). AIPO, Valencia, 2010, pp. 3–14, Keynote address.
Wong, J., Hong, J.I., 2007. Making mashups with marmite: towards end-user programming for the web. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '07); ACM, New York, NY, USA, pp. 1435–1444.
Yu, J., Benatallah, B., Casati, F., Daniel, F., 2008. Understanding Mashup Development. IEEE Internet Computing 12, 44–52.
Yahoo Pipes!, 2012. ⟨http://pipes.yahoo.com/⟩ (last accessed 13.07.12).

Research Article

Identifying User Experience Dimensions for Mobile Incident Reporting in Urban Contexts

—MARCO WINCKLER, CEDRIC BACH, AND REGINA BERNHAUPT

***Abstract*—Research problem:** *Despite the increasing interests raised by incident reporting systems, it is still unclear what dimensions of user experience (UX) and other contextual factors should be taken into account for the various stages of declaring an incident using mobile-phone applications.* **Research questions:** *How do citizens perceive and describe urban incidents? What UX dimensions are important for reporting an incident with a mobile-phone application? What other (contextual) factors are important from the users' point of view when declaring incidents? Which of the UX dimensions and contextual factors are important when in the various phases during an incident declaration?* **Literature review:** *Overall, there is a lack of empirical research in the domain of incident reporting. In general, the UX dimensions—visual and aesthetic experience, emotion, stimulation, identification, meaning and value, and social relatedness/coexperience—are important when designing interactive systems. It also shows that incidents are related to the citizen's perception of the environment.* **Methodology:** *A triangulated method approach combining interviews, a survey of existing systems, and a model-based task analysis were applied. This allows us to present a generic task model for incident reporting with a detailed description of UX dimensions affected in the various subtasks.* **Results and conclusions:** *Our findings point out the effect of UX dimensions in the task engaged by users when reporting urban citizens. The overall UX is directly influenced by the perceived level of severity, inconvenience and involvement, the personal context, and the technological mobile context. We have found that while several UX dimensions are highly relevant, they are not equally distributed along the several subtasks that citizens engage when reporting incidents. This study shows that semistructured requirement interviews can provide information about UX dimensions and it highlights the importance of the identification of UX dimensions in early phases of the development process.*

Index Terms*—e-government, incident reporting, mobile services, service quality, smart phone, user experience (UX).*

INTRODUCTION

Incident reporting is a very well-known technique in safety-critical domains, such as air-traffic management [1] and health [2], [3], where specialized users are trained to provide detailed information about accidents or a deviation from current policies. In these domains, reporting an incident is part of the work routine. An incident report often features a document that focuses on objective facts rather than personal opinions. Reporting incidents is considered an important mean for monitoring the quality of the environment and enables authorities to promote safety and improve the technical systems (either in terms of design and/or working procedures).

In recent years, several governments have started to make use of information and communication technology to allow citizens to report urban incidents in their neighborhood (such as a broken street lamp or a street water leak) to the local administration. In this context, citizens can use incident reporting tools as self-service applications [18], allowing the citizens to produce and consume services electronically without direct contact with the local administration. These applications are part of a variety of initiatives for promoting active participation of citizens in the actions of the government through the use of information and communication technology (e-government) [4], and mobile and wireless technology (m-government) [5], [6]. The state of Virginia (US) was a pioneer in deploying m-government applications, such as weather forecast, election monitoring and tourism information [7]. New applications, such as BlueTooth [8], use geolocation functions (GPS) embedded into cell phones to deliver personalized information to citizens (such as emergency phone calls and events taking place in the city). Although most m-government applications concern an urban environment (such as traffic jams, parking availability in town, and WiFi access), applications have been used even in rural areas [9].

Currently, many m-government services publish information through citizens' cell phones, but few services allow citizens to interact with the administration. For example, Fixmystreet allows users to report incidents in the cities that adopted

Manuscript received June 13, 2012; revised February 15, 2013; accepted March 25, 2013. Date of current version May 20, 2013.
The authors are with the ICS-IRIT, Université Paul Sabatier, Toulouse Cedex 31062, France (email: winckler@irit.fr).
Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

IEEE 10.1109/TPC.2013.2257212

this system [11]; however, the user interface provided is only available on web platforms and does not take into account the specifics of mobile technology.

Mobile technology offers many opportunities for m-government to obtain citizens' feedback about their environment. The latest generation of mobile devices includes touch interaction, global positioning systems (GPS), and camera, so that mobile phones (smart phones) provide users/citizens with the means to report incidents by specifying, for example, the location (such as selecting the location on a map), sending a precise location identification (such as using GPS), or simply providing a proof of the incident (such as taking a photo). However, mobile technology also imposes some constraints [10]. Due to the small screen size and low resolution, m-government solutions have to avoid the display of the same quantity of information compared to a standard computer. The same holds true for interaction resources (such as data inputs) which are also restricted.

The acceptance of m-government services is directly related to the ability of the application to address the users' needs [12], [13]. The issues of interface design are critical in the development of interactive systems, and usability of applications should be a central objective of conception [14]. A high-quality interface allows users to achieve their purpose (such as notification of an incident) in an efficient and satisfying way. Moreover, users might become dissatisfied and/or upset if they fail to achieve a goal. Thus, beyond usability, the overall (positive) user experience (UX) is important to make a service successful in terms of user takeup and frequency of usage. UX is a concept that goes beyond the pragmatic aspect of usability by taking into account dimensions such as emotion, aesthetics or visual appearance, identification, stimulation, meaning/value or even fun, enjoyment, pleasure, or flow [15].

For quite a while, it became clear that users do not only expect to receive information from the government, but citizens also expect to inform the government of their specific needs or experiences [16]. The willingness (and need) of direct and onsite citizen involvement is often highlighted during natural disasters and massive accidents [17]. Despite the fact that active use of the service is a key success factor, little is known about incident reporting systems by citizens in the field of m-government. So far, there have been no prior studies investigating what UX dimension should be taken into account when designing self-service systems for reporting incidents in urban contexts using mobile technology. Even less is known about how the UX with such systems might affect citizens' opinion about the quality of the service provided by the government.

The main goal of this work is *to investigate which UX dimensions contribute to the overall user experience in the domain of incident reporting with mobile phone applications.* This research was conducted within the project FEDER Ubiloop. Our working scenario is illustrated by Fig. 1 that presents how citizens might use diverse types of devices (mainly mobile phones) to report incidents such as potholes, missing road signs, graffiti, broken furniture in parks, and hornets. Ubiloop is proposed as a self-service system for mediating the communication between citizens and the administration. It presumes that incidents reported by citizens will prompt the city administration to those problems that are perceived as affecting their quality of life in the city. On one hand, the citizens are empowered with a system that will help them to autonomously perform an incident report, thus reducing bureaucracy. On the other hand, the city administration can have access to data provided by citizens, thus improving the detection of problems that would be difficult to identify otherwise.

In this context, the present study addresses the following research questions:

RQ1. How do citizens perceive and describe urban incidents as part of their perception of the quality of the environment?
RQ2. What dimensions of UX are important for reporting an incident with a mobile-phone application?
RQ3. What other (contextual) factors are important from a user's point of view when declaring incidents?
RQ4. Which of the UX dimensions and contextual factors is important ***when*** in the various phases during an incident declaration?

Since the context of incident reporting in urban contexts has sparsely been investigated from a human-computer interaction (HCI) or psychology perspective, it was important to first understand the context in which the mobile incident reporting application will be used and, second, to identify the relevant UX dimensions. In the next section, we present the state of the art on UX in general. Then, we present the methodological approach we have

employed during our research, which encompassed the triangulation of three methods: two types of interviews with end users (semistructured requirement interview; scenario-based interview), a model-based task analysis, and a survey of existing systems. The section on result is structured following the typical task of declaring an incident: first, we present results on how people perceive the environment and what they perceive as incidents (semistructured requirement interview), then, the UX dimensions were identified based on the second interview (scenario-based interviews). To validate the findings, a model-based task analysis is used, identifying all possible tasks for an incident report. An ideal incident reporting system would support all of these tasks. This model-based task analysis was then compared with the results of a survey on incident reporting systems, showing to what extent current systems do support the general task. This general task model was then triangulated with the data from the interviews, identifying different UX dimensions for the various subtasks in the general model-based task description. This paper concludes with a discussion and future research section.

Literature Review

The main contribution of our work is on what user experience dimensions are important for incident reporting with mobile-phone applications. We are not aware of any work identifying UX dimensions for incident reporting; thus, a general overview on UX and the selection of UX dimensions for reporting incidents via mobile-phone applications is presented. The goal of this section is to revise the literature about UX dimensions and discuss in which extensions they are relevant to the design of incident reporting systems. Therefore, this section starts with an overview of the theoretical orientation, how literature was selected, followed by the definition of UX dimensions and the influence of the context.

Theoretical Orientation Incident reporting systems are related to several applications domains, including safety-critical systems, mobile technologies, and e-government. Despite the fact that vast literature in these domains exists, there is very little information about which UX dimensions should be taken into account when designing incident reporting systems. As far as incident reports systems are concerned, we assume their usage will be influenced by 1) the general UX when interacting with the mobile phone (as a product), but also by all experiences when interacting with the specific services and the service characteristics in 2) the specific usage context.

Selection of Literature for the Review We have started investigating publications in the field of HCI by looking for results of empirical studies with users using incident reporting systems and UX dimensions that should be taken into account when designing such applications. For that purpose, we have browsed digital libraries, such as ACM DL, Springer Link, and IEEE Xplore using the keywords user experience (UX), incident reporting systems, and mobile applications; selected articles should comply with (at least) one of the following criteria: to provide a definition on the topics covered by this paper, to illustrate existing applications in the domain, to identify problems that remain to be solved in the domain, and reporting empirical studies with citizens. We have found very few studies addressing specifically incident reporting systems. Conversely, we have selected many references that describe the importance of UX dimensions in the design of interactive systems. We also have selected references that describe contextual factors related to the application domain.

UX Dimensions An overall positive UX can be the key to the longer term acceptance and usage of a system [19]. UX can be defined as to go beyond usability, focusing on cognitive, sociocognitive, and affective aspects of the UX in their interaction with artifacts [20]. UX is commonly understood as being subjective, dynamic, and context dependent [21].

It is still controversial if UX is measurable [22]. This work assumes that it is possible to measure a set of dimensions that contribute to the overall UX. What is important to note is that limited usability must not lead to a bad UX, but on the contrary, can lead to a positive UX, while good usability does not necessarily lead to a positive overall UX. This has been shown, for example, in the area of interactive TV [23].

The literature in HCI describes a broad variety of dimensions that are associated with UX. To understand the most important UX dimensions, we did a literature review on all available publications on UX starting in the early 1990s, with a focus on journals and publications in the field of HCI. Based on an assembly of 247 articles, we identified a set of UX dimensions that are central for interactive systems. Table I shows the six most commonly described UX dimensions in HCI literature. Due

Fig. 1. Overview of incident reporting: users report incidents like potholes, tagging, obstacles, or broken street lamps to the local government using mobile-phone applications.

to space constraints, the table only indicates a selection of the most important references.

The dimension *visual and aesthetic experience* refers to the pleasure that people gain from sensory perceptions, how beautiful something is perceived [24]. It includes beauty [25] and refers to classic aesthetics characteristics like clarity and symmetry [25], [26]. Overall, it is about how aesthetically pleasing and sensually satisfying an interaction is [27]. It has been shown that attractiveness and aesthetics do have a significant influence on the perceived usability of a system [28], [29].

Emotion has been identified as a key dimension of UX [25]. For Desmet and Hekkert [30], the emotional experience is one of the three main dimensions contributing to product experience, including feelings and emotions elicited. Alben [28] addresses the dimension *emotion* as an outcome for user interaction. Mahlke and Thüring [31] state that affect and emotion are considered as important parts of the UX with interactive systems before, during, and after interacting with the system.

Hassenzahl [25] describes *stimulation* as a hedonic attribute of a product, which can lead to new impressions, opportunities and insights. Sheldon et al. [32] state the need for pleasurable stimulation, focusing more on joyful aspects of the interaction. Hedonic experiences were subsumed by Karapanos et al. [33] under the term "innovativeness" to describe the ability of a product to excite the user through its novelty.

For Hassenzahl [34], the *identification* dimension addresses the human need to express one's self through objects. This self-presentational function of products is entirely social; as individuals want to be seen in specific ways by relevant others. Thus, using or owning a specific product is a way to reach a desired self-presentation. Identification can be seen as self-expression through an object to communicate identity. Jääskö and Mattelmäki [35] define user personality as part of user experience in sociocultural contexts, including the self-image, attitudes, values, lifestyle, and previous experiences.

Meaning and value refer to "*Ideo pleasure*" [36], indicating values the product can satisfy. This means that products are sometimes chosen because they reflect or represent values that are important to the person. Desmet and Hekkert [24] refer to two aspects of meaning: the experience of meaning and the meaning attached to a product.

The construct of *social relatedness/coexperience* as a UX dimension is addressed by Gaver and Martin [37] under the term of intimacy which is used to refer to nonverbal, inexplicit forms of communication. Jordan [36] describes the construct of sociopleasure as something that deals with interaction with others. Products that facilitate communication as well as those that serve as conversation pieces contribute to sociopleasure.

Mobile service UX has been defined as the combination of dimensions of service experience and UX [38]. Service experience is affected by dimensions, such as *perishability*, *intangibility*, and the *self-service nature* of the services itself [39]. There are a variety of elements that should be taken into account for any type of mobile-based services, including the coherence of the service integration, social navigation and interaction, the ability to dynamically change services, the intangibility of the service, and the availability of multiple interaction styles [40].

In this paper, we investigate how these UX dimensions are reported by users during semidirected requirements interviews and in which extensions they are related to tasks that users engage in when reporting incidents.

Influence of the Context Before designing and developing an incident reporting system, it is important to understand how people perceive and act on the environment. Currently, the main mechanisms on how individuals act in their environment are poorly understood in the field of environmental psychology [41] and unfamiliar to the HCI domain. Nonetheless, it seems important to know: (1) how individuals perceive their environment, (2) how they discover incidents, and (3) how they transfer this knowledge to self-service systems. Two concepts are important to understand how people perceive their environment: place identity and amenity. Place identity [42] refers to the cognitive aspects related to the perception of the environment, including one's attitudes, feelings, ideas, memories, personal values, and preferences toward the whole range and all types of physical settings. These aspects of place identity allow people to understand the environment they live in and their overall experience. In this way, one can consider place identity as a structure of the self-identity, which means situated and self-centered. Thus, the same physical environment can be perceived differently by individuals. For example, a handrail can be perceived as an aid for elderly people and as an object to play with for kids.

The concept of amenity refers to the ability of spaces to evoke emotional responses, such as attractiveness and desirability. Amenity refers to any benefits of a property, especially those that affect attractiveness or value of places. Amenities include facilities, such as restaurants, parks, swimming pools, theaters, children's playgrounds, and bicycles paths. Amenities also include pleasant architecture, nearby activities, good schools, or a low crime rate, all of which add to the desirability of place and property. The concept of amenity explains how environmental qualities can have an impact on the hedonic and social perception of environment.

The identification or perception of an incident is related to a mental contradiction between an expected state of the environment (influenced by the place identity of a person and the amenities given in that environment) and the real state of this environment. When this contradiction is too high, people feel the need to report this contradiction or correct it.

For Moles and Rohmer [43], the main role of the urban environment is to act as a mediator between individuals and the society. Such mediators exist on different levels ranging from a macro to a micro level. At the macro level, the role of the urban environment includes building public transportation or the global management of the city. Individuals typically do not have a lot of influence on the macro level. On the micro level, the urban environment refers to events and objects that individuals interact with in their daily actions (like taking a bus or enjoying a park). Thus, the role of a designer of any incident reporting system would be to improve the communication between the individual (and his or her daily experiences on the micro level of the urban environment) and the local administration or government (on the macro level of the urban environment).

METHODOLOGY

The goal of our research is to understand what UX dimensions contribute to the overall UX in the domain of incident reporting with mobile-phone

applications. The following sections explain how we attempted to reach this goal. It first explains our choice of a research methodology. After explaining the choice of a research methodology, we separately describe the participants, processes, and data-analysis techniques for each of the methods we used.

Choice of Research Methodology To answer the research questions, we have been triangulating and applying three methods: (1) interviews, (2) a survey of existing systems, and (3) a model-based task analysis. The methods were applied in the following three steps:

Step 1) To understand how citizens perceive incidents (RQ1), we have been performing a semistructured interview, identifying requirements for mobile applications for incident declaration.

Step 2) To understand what UX dimensions are important (RQ2) and what other dimensions can be influencing the design and development (RQ3), we:
- (a) identified scenarios of typical incidents in the semistructured requirement interviews;
- (b) we completed and extended the scenarios and the scenario description by comparing them with the results of a survey on existing systems;
- (c) we then conducted the scenario-based interviews.

Step 3) To identify what UX dimensions are important during an incident declaration (RQ4):
- (a) a model-based task analysis was performed (based on the survey of existing systems)
- (b) the UX dimensions and (contextual) factors were associated with the subtasks in the model-based task description.

Semistructured Requirement Interview In order to understand users' requirements for incident reporting systems, we have conducted a semistructured requirement interview.

Participants All participants were French native speakers. Their education level ranged from high school to obtaining a Ph.D. They lived in France in the Toulouse metropolitan area for at least two years and up to 40 years ($M = 13$, $SD = 10$).

TABLE I
SUMMARY OF LITERATURE REVIEW OF UX DIMENSIONS

UX Dimensions	References												
	Hekkert, 2006 [24]	Hassenzahl, 2003 [25]	Hassenzahl & Tractinsky 2006 [27]	Lavie&Tractinsky, 2004 [26]	Alben, 1996 [28]	Quin & Tran, 2010 [29]	Desmet & Hekkert, 2007 [30]	Mahlke & Thüring, 2007 [31]	Sheldon et al., 2001 [32]	Karapanos et al., 2010 [33]	Hassenzahl, 2008 [34]	Jordan, 2000 [36]	Jääskö and Mattelmäki, 2003 [35]	Gaver & Martin, 2000 [37]
Visual and aesthetic experience	■	■		■	■	■	■					■	■	
Emotion	■		■		■		■	■	■					■
Stimulation	■	■							■	■		■	■	
Identification		■					■		■		■	■	■	■
Meaning and value							■		■			■	■	
Social relatedness/co-experience											■	■		■

Note: ■ means UX dimension reported by the respective authors.

For the semistructured requirement interview, we had nine participants (labeled in the following P1 to P9) six males and three females ($M = 40$ years old, $SD = 15$). Participants were selected from a neighborhood association that had been previously identified as a pilot test population for the Ubiloop project. Our goal when recruiting participants, who are already actively engage in local communities, was to have participants who would be very likely to report incidents and who would be ready to actively act when detecting an incident. In order to gain better insight on necessary requirements for smart-phone applications to report incidents, we selected participants that were knowledgeable in terms of smart-phone usage. All participants owned at least one smart phone (and up to three mobile phones). Phones were used to make phone calls ($n = 9$), send short text messages ($n = 8$), receive and send email ($n = 5$), access the internet via the phone ($n = 6$), make photos ($n = 8$) or videos ($n = 3$), and use the GPS ($n = 5$).

How Data Were Collected. At first, participants were informed about the goal of the interview: to explore the utility of smart-phone applications for reporting changes or degradations in the quality of the environment. Then, they were prompted to report about:

(a) How they perceive places and their environment; such as to tell dimensions they consider important for the quality of their environment (either their neighborhood or working place).

Negative experiences in terms of environmental quality; such as to tell about events they have in their neighborhood or working place.
Personal involvement with problems; such as to identify who they think should be in charge of solving problems in their neighborhood: themselves (personal level) or the local government (societal level).
Preferred system design; such as to tell how they would like to report degradations of the environment (such as incidents) and what kind of technology should be used (for example, web service on PC or smart phones).

(b) UX dimensions they think are important; such as to name elements that are important for a good experience or a good quality of service.

Identification of Scenarios To analyze the actual support provided by existing applications, we conducted an analysis of existing services for incident reporting in urban contexts. This survey focused on the front office (such as reporter tools) and not on the back office (such as officer tools). Applications for incident reporting were first identified from the set of tools ranked by web search engines (such as google.com). Then, only those that were available for remote testing were selected for further analysis. Fifteen applications were selected covering international and national incident reporting services (covering the US, Canada, UK, the Netherlands, Australia, New Zealand, Norway, South Korea, Spain, and India) and eight city services (New York, Vienna, Copenhagen, Lisbon, and four French towns: Paris, Pouancé, Merignac, and Athis-Mons). Most applications are accessible from the website of city administrations.

These 23 applications included different technological platforms resulting in eight services that are available only on the web (such as those that can be used only on standard computers), three services that are optimized for webmobile (such as those that can be used on smart phones and standard computers), ten applications that are delivered through the web but also provided as an embedded smart-phone application, and two services that are available as embedded smart-phone apps only. In a second step, we excluded all web applications and focused only on those that are available on mobile platforms.

The remaining 12 applications were then compared accordingly to their task support for reporting incidents as described in Table II.

Despite the fact that these applications address the same problem of reporting incidents in an urban context using mobile technology, the scenarios implemented are different. For example, while in some systems the identification of the user is mandatory, others accept anonymous reports; in some systems, users can chose how to provide the location (such as an address or GPS coordinate). Other solutions only offer a simple text field for entering an address.

As far as the main task "detect incident" is of concern, four applications provide some support to help users recognize what type of problem would be an incident and the other four applications provide a mechanism for helping users identify how to solve the incident. None of the tools surveyed explicitly motivates users to report incidents.

The task "submit an incident" is supported by different scenarios but there is some consensus on subtasks such as "provide a description" using either text or a picture and "to locate the incident" by pinpointing it on a map. Other subtasks, such as select incident category, record when the incident is reported and provided personal coordinates (available in 11 of the 12 applications analyzed). To rate an incident in terms of severity is only possible in two applications and only one considers potential danger. While textual descriptions and pictures are common, none of the applications considers an alternative service, allowing the user to call a hotline, which would be effective for blind users for instance.

The use of interactive maps where users can pinpoint the incident is supported by all applications analyzed. GPS coordinates are supported by nine applications, four of them allow users to provide an address; another three allow users to use landmarks to locate an incident.

Most of the applications ($n = 11$) automatically collect the data/time when an incident is reported. Only one lets users provide the time when the incident occurred. All applications provide either support for users to share reports and/or to see reports from other users, but only four applications allow users to subscribe for the outcome or result associated with the incident they have reported. The presence of specific features does not necessarily add value to the system; in some cases, it might be the opposite, as the absence of unnecessary features might create a feeling of efficiency.

Based on this analysis and the results of the requirements interviews, we came up with seven

scenarios that cover all identified aspects and subtasks of an incident declaration. They were used as input in the scenario-based interviews and are presented in the following section.

Conducting Scenario-based Interviews For the scenario-based interview, we invited nine participants (labeled P10 to P18 in results section), six males and three females ($M = 27$ years old; $SD = 6$). These participants represented a younger population compared to the participants from the first interview. Participants had a broad knowledge on various forms of information and communication technologies, using mobile phones and internet frequently. All nine participants stated they use their smart phones for calls and sending text messages, eight use it for mail and accessing internet via the smart phone, seven use it for making photos/videos and seven use the GPS function. However none of the 18 participants had used an application to report incidents using a smart phone. All participants gave written consent for participating in this study and our institution research ethics committee deemed the research "exempt."

For this interview, participants were asked to consider the seven scenarios identified earlier. Participants were introduced to each scenario and then asked to explain how they would envisage reporting the incident using their smart phones. The scenarios were chosen to represent the most common incidents in the area of Toulouse, represented incidents identified by users in the requirement interviews, and should reflect the most frequent types of incidents supported by existing systems. Moreover, each scenario was designed to highlight a specific context of use. The incidents explored in the scenarios include:

(1) Broken street lamp. This incident was chosen to explore situations that would be difficult to illustrate with a picture. Broken street lamps are often noticed during the night which makes photos almost impossible as many smart phones do not have a flash nor do they cover long distances. The scenario provides some geographic information to prompt if participants would use photos when reporting the incident.
(2) Pothole. The pothole incident was designed to investigate users' personal involvement. It describes people riding a bicycle over a pothole and then feeling backpain afterwards. This scenario is aimed to explore emotions and social behavior triggered by (negative) emotional experiences with incidents.
(3) Missing road sign. The scenario of a missing road sign takes into account possible limitations for using a smart phone to report an incident when people are in movement, for example, driving a car. This scenario explores time/place aspects of incident reporting.
(4) Bulky waste. In Toulouse, waste removal is performed by two different services: garbage trucks regularly collect any waste that fits into the standard waste containers; however to remove bulky waste, people need to call the local administration for booking a larger truck; otherwise, the waste will remain in place, causing a nuisance. This scenario explores how (active) usage of services can prevent incidents, what knowledge people have about local administration procedures (such as whom to call), and people's previous experiences with local administration.
(5) Hornet's nest. This scenario depicts a hornet nest close to a playground with some hornets flying around people. It is aimed to explore the influence of perceived danger on the incident reporting.
(6) Tag/graffiti. In this scenario the participant is told to be going to an appointment when he notices some fresh graffiti next to his car; participants promptly report this incident even if they are in a rush. This scenario is aimed to explore the perception of the level of nuisance and priority, need for immediate action, and feeling of duty toward society.
(7) Broken bench in a park. This scenario explores difficulties for locating precise incidents. It also prompts people to get involved with (*a priori*) minor incidents.

All sessions were recorded and then transcribed by a French native speaker. The transcriptions were analyzed accordingly to the Grounded Theory Approach [44], [45]. A corpus of 92,240 words was analyzed and coded accordingly to 11 classes/codes with 1125 segments of text. Every segment of text was interpreted accordingly to the context given in the scenario; for example, when users expressed a feeling of relief after reporting an incident, the segment was coded in the class emotions. In order to reduce the impact of subjective interpretation, we used the set definitions presented in the state of the art. Moreover, the coding was cross-checked by two independent reviewers with strong backgrounds (Ph.D.) in HCI. The 11 classes and codes refers to the six UX dimensions (including

TABLE II
TASKS SUPPORTED BY 12 EXISTING APPLICATIONS FOR REPORTING INCIDENTS IN URBAN CONTEXT

Tasks for reporting an incident				Existing mobile applications											
				SeeClickFix	City sourced	Fixcity	Fixmystreet.com	Fitthathole	Fixmystreet.nz	Neatstreets	Verbeterdeburt	FiksGataMi	Buitenbeter	Fixmystreet.kr	Na Minha Rua
Detect incident	Recognize incident			■	■	■							■		
	Identify who should solve the incident							■			■	■	■		
	Decide to report incident														
Submit an incident	Describe the incident	Select incident category		■	■		■	■	■	■	■	■	■	■	■
		Rate the incident	Potential danger		■			■							
			Inconvenience												
		Provide a description	Text	■	■	■	■	■	■	■	■	■	■	■	■
			Picture	■	■	■	■	■	■	■	■	■	■	■	■
			Call hotline												
	Locate the incident	Provide address			■	■		■							■
		Pinpoint on a map		■	■	■	■	■	■	■	■	■	■	■	■
		Use landmarks			■	■		■							
		GPS coordinates		■	■		■	■	■	■		■	■	■	
	Inform time of the incident	Tell when incident occurs						■							
		Record when the incident is reported		■	■	■	■	■	■	■	■	■	■	■	
	Provide user identification	Collect user ID from the system		■	■	■					■			■	
		Provide personal coordinates		■	■	■	■	■	■	■	■	■		■	■
Follow up the incident	Subscribe for notification			■	■		■					■			
	Share reports			■	■	■		■	■	■	■	■	■	■	■
	See someone else reports			■	■			■	■	■	■	■	■	■	■

Note: ■ means task is supported.

visual and aesthetic experience, emotions, stimulation, identification, meaning and value, and social relatedness/coexperience) plus contextual dimensions including user motivation (to report the incident), severity, (level of) inconvenience, diversity of technical platforms, and communication style. The coding was supported by the MaxQDA 10 software [48].

Model-Based Task Analysis Typical user tasks for reporting incidents were analyzed and described using a model-based notation [47]. Task analysis is widely recognized as one fundamental way to focus on specific user needs and to improve the general understanding of how users may interact with a user interface to accomplish a given goal when using an interactive system [46]. A task model is a generalization of alternative solutions for achieving a goal—in our case, to report an incident. Each alternative solution is specifically addressed by a scenario. By modeling the tasks of reporting an incident, it is possible to have an abstraction of contextual alternatives, which is required for determining optional/mandatory tasks, inner dependencies between tasks, as well as pre and postconditions associated with task execution.

RESULTS

This section reports the results of the study. It integrates the results of the different parts of the study to provide cohesive answers to each research question. The section concludes by presenting a task model emerging from these results.

Results for Research Question 1 How do citizens perceive and describe urban incidents as part of their perception of the quality of the environment?

We found out that the starting point for any incident report relies on user's skills to detect the incident, which can be refined as being able (a) to identify an event that could be perceived as a problem or nuisance, (b) to detect an event that could prevent the occurrence of a likely problem, and (c) to envisage something worth reporting that could improve the quality of the environment and/or its management. For example: Participant 11 (in the future, we refer to participants as PXX, where P indicates participant and XX indicates their identification number) commented on the scenario of the pothole (a) "... this happened to me. I was driving with my bike on a pothole and it really damaged both of the wheels. At this occasion I really wanted to report the incident." P16: (b) "... I would be willing to spend more time explaining how to prevent that incident than would do for reporting just a minor incident [afterwards]." P7: (c)

TABLE III
SYNTHESIS OF INTERVIEWS FOCUSING ON UX DIMENSIONS THEIR CORRESPONDENCE WITH TASKS AND IMPLICATIONS FOR THE DESIGN OF INCIDENT REPORTING SYSTEMS

UX Dimensions	Users reporting the dimension in requirement interviews (population *N* = 9)	Users reporting the dimension in scenario-based interviews (population *N* = 9)	Total of segments (requirement interviews + scenario-based interview)	User tasks	Implications for design of incident reporting systems
Visual and aesthetic experience [AX]	6	6	21 (4,2%)	-	All users express their preferences to applications for incident reporting, mainly for aesthetics reasons. So, design options should be generally supported by a smart-phone applications dedicated to incident reporting rather than a web application running on mobile phone's browser. However, users wish to have a website on a larger PC display to see a map of all reports in their neighborhood.
				[Provide a picture / video]	Users consider good pictures as more valuable and significant for incident reporting. So, the service should provide guidance to take pictures with a good visual quality (e.g., through photography tutorial).
Emotions [EM]	9	6	61 (12,1%)	-	Declare incident helps users to improve the quality of their environment. This contributes to an overall positive emotional state.
				[Select incident category]	Some types of incident would generate negative emotional responses. For example "dangerous" animals as hornets, rats, or snakes are a source of phobia. The system should help users to regain their calm by reporting the incident and by providing safety instructions.
				[Rate potential danger]	Rate the perceived danger of an incident is a mean to express a negative emotion, especially when users had been involved in the incident (see case study of the pothole). It's helpful to regain calm and rationality. So, the design option should provide sliders to rate incidents.
				-	Some users declare that they would be proud to help local administration to improve the environment quality. Design options would be to support collective IR as events / games (i.e., organized by local administration) in order to improve the emotional involvement with the service.
Stimulation [ST]	9	2	29 (5,7%)	[Submit an incident report]	Smart-phone applications that provide an easy and fast way to submit a list of incidents also stimulate users to submit incident. Design options should globally favor efficient applications integration in the smart-phones' operational system.
				[Select incident category]	Incident categories are a good mean to prompt users to recognize then declare different types of incidents. Nevertheless, incident categories should be short (i.e., 5 items through 3 levels) to avoid short-terms memory workload.
				[See someone's' else reports]	Look at other reports is a good mean to share different users' point of views and also to recognize problems encountered by different citizens. Users prefer to make this activity at home on a website. Design options should provide interactive maps, including filters, available on website.
Identification [ID]	9	9	150 (29,6)%	[Decide to report the incident]	Decision to report is a consequence of both user's personality (e.g., citizens' duty) and the mean to do it. In terms of design implications, the users should have a mean to instantly report an incident when they perceive it. For now, this mean should be his/her smart phone including a dedicated app.
Meaning and values [MV]	9	9	110 (21,7)%	[Identify who should solve the incident]	The users expect to know the effective means, benevolence, and aims of the e-services and generally how the service can work. This is important because it's a precondition (for users) to use and be confident with the system. In terms of design solutions, this should lead to a video and/or a webpage describing the service intentions, means, policies, and workflows.
				[Provide a description]	Users involved in incident declaration would like to provide valuable reports. So, according to the incident type, users would like to provide (at least) the mandatory data. A design solution should be to provide a kind of template and/or a tutorial explaining how to provide efficient incident description. For this point, the guidance of users is important, especially for incidents that are potentially dangerous.
				[Locate the incident]	Location of incident is a mandatory issue for IR. So users would like to provide a good location of the incident. Most of them suggest using cardinal coordinates (i.e., with the use of GPS). However, some users would also like to keep control on the transfer of these coordinates. Indeed, some users identify cardinal coordinates as private data and would like to transfer them only after an explicit action. So in terms of design solution, the user should allow (or avoid) automatic location, for example in a "preferences" menu of the app.
				[Provide user ID]	Users consider their identification as a credibility cue of their reports. They also consider identification as a mean to prevent the service from fake reports. Nevertheless, some users expect to have an option of automatic identification, in the purpose of effectiveness, but another part of users would like to transfer their identification after an explicit action. So, as location, a design option should propose an option between automatic and explicit identification (e.g., in a preferences menu).
				[Subscribe for notification]	Notifications are a cue of the local administration capability to manage/solve IR. Depending on the incident type and the level of user involvement with the
					incident, users would like to be informed of IR evolution. A design option, during the IR procedure, should propose to users to freely subscribe for notifications.
				[Share reports]	Users consider the different signs of the service activities as cues of value and credibility of the service. In terms of design option, a mean to share reports with local admin /citizens would be a mean to appreciate the overall activity of local admin in order to solve incidents.
Social relatedness Co-experience [CX]	9	9	135 (26,7)%	[Identify who should solve the incident]	Users will share information only if they estimate that information would be taken into account by at least one service. So, in order to prompt users to report incidents, it is important to indicate which service is in charge to take into account which type of incident (e.g., in the notifications, on the website describing the service workflow).
				[Provide a picture/ video]	Pictures or videos are a good mean for users to explain to someone else the incident and its context. The service should help users to take an efficient picture of an incident, for example through a tutorial.
				[Call a hot line]	Some users would share their experience of an incident directly by phone, especially with dangerous ones (e.g., Hornet nest). Furthermore, a phone call is a direct mean to ensure users that the local administration shares the same comprehension of the incident. To do this, the app should provide a mean to call a hotline, according to the incident type.
				[Subscribe for notification]	Users don't expect automatic responses. The service should provide a reformulation of incident report by human agents.
				[Share reports]	Sharing information is important to build and keep a community around incident reporting activities (i.e., between administration and citizens). Design options should provide interactive maps with filters to show/locate the different incidents. Subscription to RSS flow would be also a solution to share reports with other citizens. These kinds of activities should be supported mainly by dedicated websites at home.

"With this e-service you could make a request to add a pedestrian crossing [due to the] more and more people who want to cross here to reach the new shops."

The detection of an incident is based on tangible characteristics identified in the environment and how an individual interprets them in the respective location. The perception of an individual of the nature of an incident appears to have an impact on its level of involvement in the reporting process, it also influences the time and number of operations a user is willing to spend and to perform an incident report:

- Interviewer: "According to the severity, you would allow yourself more time for reporting the incident?"
- P15: "Yes that's it!"

Participants differentiate incidents with different degrees of severity ranging from a minor incident to dangerous incidents. The report of a minor incident will generally be driven by the perception that it is citizen duty. In this case, people want to spend very little time, with only some actions to be performed on the smart phone and an interaction time span of less than a minute as, for example:

> P16: "As [this is] an incident of little importance, I want something fast, a few steps. It's just a matter of service to the city, to be a good citizen."

Conversely, participants would be willing to spend more time for potentially dangerous incidents, as stated below:

> P18: "Because for all minor incidents that is OK to be vague. But there is a need to be accurate [in the case of a dangerous incident] even if it takes more time."

The level of inconvenience is characterized by the troubling nature of the incident either from an organizational point of view or in terms of moral or material values. Inconvenient incident may damage equipment or disturb the peace, as illustrated below:

> P16: "The tag generally does not shock me but some content of tags may be disturbing and inappropriate. In this case it should be possible to associate a level of inconvenience to the incident report."

The level of involvement in the incident: Individuals involved in an incident often want to report it and, in such cases, they would devote more time to make a precise report:

- P15: "Here I can take 2 to 3 minutes to write this kind of incident report. It bothers me much less (as reporting a minor incident). Because it affects me directly."
- P16: "If the tag is on my house then it is clear that I will make the report with everything necessary. It all depends on how I am personally involved [with the tag]."

Results for Research Question 2 What dimensions of UX are important for reporting an incident with a mobile-phone application?

To investigate which dimensions of UX are important for reporting an incident with a mobile-phone application (RQ2), the transcripts of the two interviews were analyzed following the Grounded Theory Approach classifying users' statements. The two interviews provided evidence for identifying the following UX dimensions: visual and aesthetic experience, emotion, stimulation, identification, meaning and value, and social relatedness/coexperience. Table III shows how many users mentioned the respective UX dimensions during the interviews and the main findings. This shows the number of participants who provided segments related to the UX dimensions, respectively, during the requirement interviews and during the scenario-based interview. The last column in Table IV shows a synthesis of implications for the user interface design of incident reporting systems. Hereafter, we provide excerpts of participants' comments that illustrate how these UX dimensions are related to incident reporting systems.

In general, visual and aesthetics experience was considered by participants to be less important than other UX dimensions. Nonetheless, interviews point out two interesting aspects: first, the visual quality of a smart-phone application should be better than the visual quality of websites displayed on a smart phone. The second aspect is related to the quality of the pictures taken with smart phones. People want to provide a good and clear picture of the incident and perceive that aspect as important to establish a trustful relationship with the local government. This aspect creates a link between visual and aesthetic experience and the overall trust on the e-government service, as mentioned by:

> P15: "If the photo is good, they [the local administration] will see the problem. . . "

The interviews identified positive and negative emotions that are related to how people perceive places and their environment (place identity) and to the various levels of the domestic environment (micro/macro level). Emotions were also judged as important to design for, since the application can be a mean to overcome negative experiences, and the reporting of an incident affects users not only in terms of positive emotions (joy), but also influences long-term perceptions (pride). Thus, three sources of emotion have been identified: emotions related to the quality of user environment, negative emotions associated with the occurrence of incidents, and (positive) emotions that can be attributed to the use of the system.

Some participants expressed their pleasure to be in a "high quality" environment; as incident reporting helps to improve the quality of the environment. It also contributes to an overall positive emotional state. For example, some participants think that the application could allow them to experience positive emotions of pride and enthusiasm, especially from having had the opportunity to contribute to the improvement of the environmental quality of the city, as mentioned:

> P18: "... I would be very happy to do that [to report of a broken bench]. So the national pride of Toulouse is increased."

Negative emotions are reported from previous experiences especially if an incident directly involves the user (such as a bad experience with a pothole while riding a bicycle). Negative emotions were also related to the degree of influence participants perceive on ability to influence the macro level of the domestic environment, like the perception of overpopulation due to a large number of new buildings in the area, or the increasing level of noise due to heavy traffic. There are some positive emotions can be attributed to the use of a system, in particular, when the system helps users to overcome a negative experience. For example, participants mentioned that the application could help them overcome the (negative) emotional perception and, eventually rationalize the experience, if they are allowed to express themselves via the incident reporting system. Nevertheless, these emotions can be influenced by the users' ability to use the application, as quoted below:

> P13: "...under the influence of anger, there is a chance that I miss to report the required data and that as a result the reporting [an incident] is not considered. So they [the system] should use a text field to require users to think a little and calm down..."

Negative emotions also result from fear that an incident report might lead to a reprisal. In the example below, P3 was afraid to take a picture of graffiti leading to the identification of its maker who would felt accused by the incident declaration and then decide to take revenge on the declarant.

Interviewer: "So the problem is to take pictures, so if you make a picture you are afraid that there will be a kind of retaliation?"

> P3: "Yes, I got this..., this kind of feeling. Yes."

The next UX dimension explored is stimulation, which refers to the ability to stimulate users to use the application, for example by recommending the use of specific services. Participants often mentioned that if they were allowed to see incident reports provided by other citizens, they would feel stimulated to look for similar problems in their neighborhood, especially if these incidents involve ideas for improving the quality of life in their neighborhood, for example:

- P4: "... I even find it difficult to imagine that [the incident] unless someone talks to me about it. Perhaps the application could prompt us to look at some incidents or perhaps we could see what others have reported and [to incidents that] I am not sensitive to [perceive them]..."
- P7: "Besides going to report your [own] idea, you could ask if there are other ideas [proposed by other]... [that are] close to your home."

Being able to report incidents with a smart phone can be an incentive to be an active member of the (local) community and, thus, start a relationship with the local administration.

> P2: "... Having this application [such as an incident reporting system] may give the consciousness of a kind of mission, of vigilance. So one can say that one would not miss any incident, this may encourage people to go out for a walk, instead of staying at home... "

It is noteworthy that this dimension is also related to the perception of vigilance that can involve the security in the neighborhood, which can be considered beyond the scope of incident reporting systems addressed in this work.

The interviews showed that identification is related to three concepts: the identity and personality of the individual, how people identify themselves with

a place, and the identification with (and attachment to) the smart phone.

Identification is important in all phases of the incident reporting including: people's identification with a place (place identity) supports the diagnosis of the incident (sensitivity to the types of incidents), people's willingness to report an incident (personal values, attachment to places), and identification with the means available for reporting incidents (such as identification with the smart phone).

The identification (identity/personality) therefore concerns all personal values of the user. But identification is also related to the user's interest and ideas, the willingness to act, and to perform citizen duties, for example:

> P3: "... Well, maybe my perceptions are a bit unusual compared to others, but I see lots of things to report ... It's in my nature, I am open, and so I'm reporting back information [to the local authorities]. That's it".

Place identity is central to the willingness to report an incident as expressed in:

> P14: "Well if it's a bench on which I used to sit with my family every Saturday afternoon... then yes ... it will make it [the intention to report an incident] stronger. But if I just passed by and I never use it, well I do not even know if I would see that it is broken find out."

The level of identification with the smart phone is a positive promoter for incident reporting, for example:

> P11: "Usually I cannot forget the appointment with the bulky waste, because I note everything on the agenda of my smart phone that I have always with me."

The value of the incident reporting is influenced by the perception that users have about the utility of their incident reports. The value of incident reporting systems can be reduced if is misused to denunciate someone or to transfer the work from an administrative agent to citizens. For example:

- P1: "Well, it must be of good citizenship anyway. This is the civic duty, it is not denunciation. And the service must works in this spirit".
- P16: "Yes then it does not have the exact location of the pothole, but... it is agent's duties to be careful to locate it [the incident] it in the field. Otherwise I will feel be doing the agents' job, which completely devaluate the service."

The dimension of meaning and value is also directly influenced by the perceived efficiency of the local administration/government. If an incident is reported but never solved, participants are told they would be keen to abandon the application, as stated below:

> P12: "On this type of incident I would like information from the back-end service. How they tackle the problem? Are they going to fix it? And at least, if they have understood it [the incident report]? Otherwise it will give the impression that it is useless to make reports and then I'll stop making it."

In general terms, participants think incident reporting systems as worthy in three situations: (a) to provide reliable evidence of existing incidents; (b) to provide personal identification, as evidence of the individual commitment; and (c) to rely on users reporting the same incident. For example:

- P14: (a) "For this incident I want to take a photo as a proof. In this way they can trust me."
- P4: (b) "If we do not identify ourselves, everyone will begin to send anything and everything. Because there are always idiots who play around and misuse applications. So the service loses its value if invaded by spam."
- P17: (c) "I see an interest in knowing that other people reported the incident, like that according the type of incident, I will make an additional incident report to give more importance to the incident, to be sure the incident will be considered by the service."

What became evident in the interviews is that participants did not perceive the incident report as part of their duties; but they felt it more like an act of sharing information. It is like a tweet (twitter message) that helps them to get in touch with the local administration.

In this sense, from the perspective of users, we have to consider the m-government service of incident reporting as a special type of social network. This is clear in the example below where a participant identifies the system UBILOOP as that social network:

- P11: "I take a picture of the broken bench. Then I press the "Share" button. In the smartphone a bunch of social networks is shown where I can put the photo. So there I simply diffuse the photo on the community network UBILOOP."

TABLE IV
THREE SCENARIOS EXTRACTED FROM THE TASK MODELS FOR REPORTING AN INCIDENT

Scenario 1: Minimalist report	Scenario 2: Fair report with feedback	Scenario 3: Detailed report with pictures and GPS coordinates
detect incident recognize an incident identify who should solve the incident decide to report the incident submit an incident report describe the incident provide textual description locate the incident use landmarks send report	detect incident recognize an incident identify who should solve the incident decide to report the incident submit an incident report describe the incident select incident category rate incident rate potential danger rate inconvenience provide a description provide textual description locate the incident provide an address provide user identification provide personal coordinates inform time for the incident record when the incident is reported send report follow up an incident report subscribe for notification	detect incident recognize an incident identify who should solve the incident decide to report the incident submit an incident report describe the incident select incident category rate incident rate potential danger rate inconvenience provide a description provide textual description provide a picture provide a picture provide a picture locate the incident solve GPS coordinates inform time for incident tell when the incident occurs send report

- P13: "I do not care that my report is on track, it's secondary. I want a human being answer me, so I can make sure he understood my problem and that it will be repaired or not. It should not be something automatic; it has to be people who respond."

The participants also express the need for sharing information with other citizens. For example:

> P11: "I guess this will be more or less a community network. So I would probably not be alone in reporting the incident."

Results for Research Question 3 What other (contextual) factors are important from a user's point of view when declaring incidents?

The first dimension explored is communication style, which mainly refers to types of synchronous communication (such as via phone call) versus asynchronous communication (such as a text message). In general, the user preference is directed toward an incident report via asynchronous communication. However, the choice of asynchronous communication is not exclusive; some participants redirect their preferences to asynchronous communication (such as making a phone call) to report a dangerous incident or to prevent one (such as removing bulky or garden waste).

> P13: "... for this incident, I would like something synchronous.... One solution would be that I send a photo with the GPS and that would allow me to exchange directly on the phone with an agent who could see what I mean by looking at the photo with the incident".

In terms of feedback from the local administration/authorities, participants referred to a notification by email, short text messages (SMS), or a history function within the application, showing which incidents have been reported, and which of these has been successfully removed.

The next dimension explored, diversity of technical platforms, refers to the idiosyncrasies of interaction techniques and the platforms (such as web, Android, or iPhone). Participants referred to the technical possibilities provided by embedded technology into smart phones when describing scenarios, for example:

- P12: "... [for this incident I would use] an audio message rather than text, it would suit me better";
- P19: "... it is sufficient that I activate a vocal command to the GPS system, then it records my position and makes me a memo to report the incident later when I am not driving the car";

The use of incident reporting systems and an internet application accessible from home were mentioned several times, for example:

> P1: "... on the Internet at home we could see the diversity of types of incidents reported; this could allow me to think of problems I never thought of before".

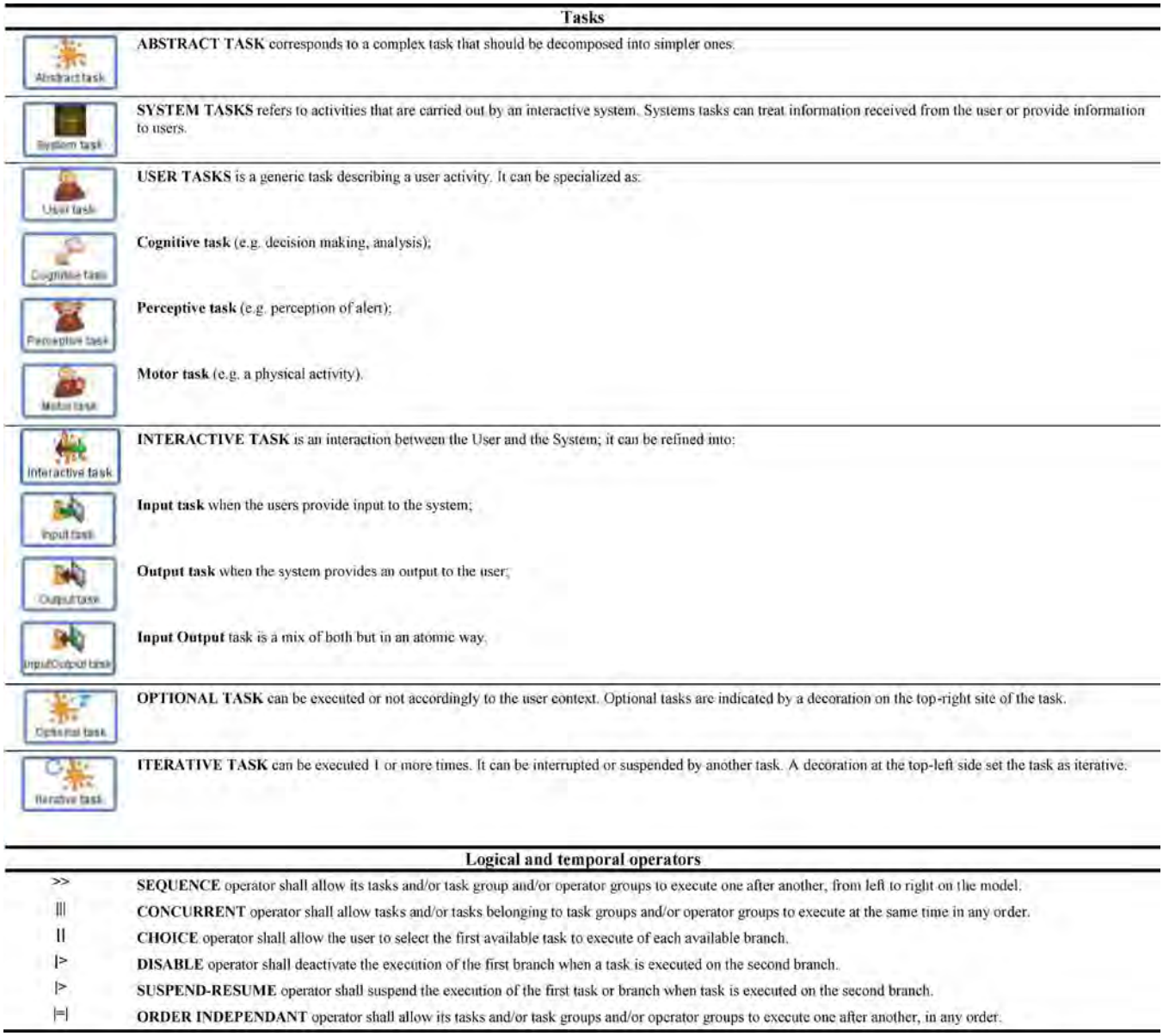


Fig. 2. Tasks and operators in HAMSTER notation.

Most frequently, participants mentioned an application for the smart phone, the ability to send photos of an incident, to send GPS or Cartesian coordinates, to record noise or record voice messages, to use vocal commands, to write text, annotate an image (drawing tool), to access an electronic calendar, to make a video, to select from an interface that simply provides check boxes and form-fill in or text fields, to send an email, to indicate incidents on an interactive map, or to use a personal information system.

Results for Research Question 4 Which of the UX dimensions and contextual factors is important **when** in the various phases during an incident declaration?

This subsection provides the responses.

To answer this research question, we first need a general task model that is able to describe all of the subtasks that would be possible and necessary to declare an incident. The following section describes the model-based task analysis that was performed to generate such a general task model.

In this subsection, we present a generic task model for incident reporting systems. For that purpose, we employ a task model notation called Human-Centered Assessment and Modeling to

Support Task Engineering for Resilient Systems (HAMSTERS) [47], which extends model-based approaches, such as CTTE [49], to provide more powerful structuring mechanisms. (See [50] for further details.) The main goal of this model-based task analysis is to describe all possible scenarios, leading users to successfully report an incident to the local administration.

Task models using HAMSTERS are created by identifying the main goal for every user or system activity. Each goal will feature a task in the model that is depicted according to the actors involved in the task execution (such as the user, the system, or both at a time). Complex goals, represented by abstract tasks, are then decomposed in a hierarchy of subtasks. The next step consists in connecting tasks using logical and temporal operators for expressing dependence between task execution (for example, sequence, choice, order independence...). The operators can be used to simulate the execution of tasks; each sequence of execution is then considered a valid scenario to the task model. By using these basic components of the HAMSTER notation shown in Fig. 2, it is possible to create a generic model of user activity with a system.

Task models were created as a generalization of the scenarios previously identified in the study. By analyzing users' scenarios and existing applications, we have found a pattern that encompasses three main tasks: (1) to detect the incident, (2) to submit an incident report, and (3) to follow up on an incident report. Fig. 3 illustrates the hierarchical organization of these tasks using the HAMSTER notation. The operator $\gg$ indicates that these tasks should be performed in sequence. The execution of the tasks starts with "detect incident," continues with "submit an incident report," and finishes with "follow up an incident report." The task "submit an incident" is set as an iterative (symbol on the left-hand side) to indicate that users might revise reports many times before submitting them. The follow up of an incident report is set as optional (see right-hand side symbol) since not all citizens will be interested in the outcomes of an incident report.

The tasks presented in Fig. 3 are complex so they are depicted by abstract tasks. This model has been in the sequence extended as shown in Fig. 4 to accommodate all of the possible variations according to the identified scenarios. For the task "detect incident," we found out that it encompasses the subtasks "recognize an incident" and "identify who should solve an incident and decide to

Fig. 3. Main tasks for reporting an incident.

report the incident". The subtask "submit an incident report" encompasses several subtasks for describing an incident report. Generally speaking, the information requested in the identification of the incident includes a description, a location, the time associated with the occurrence of the incident, and the identification of the person reporting the incident. Not all of this information is mandatory for identifying an incident; however, the models indicates that at least the description and the location of the incident should be provided by the user. This means that subtasks "describe an incident" and "locate the incident" are set as mandatory while the subtasks "tell when the incident occurs" and "provide user identification" are described as optional ones. Since the task "submit an incident report" is defined as iterative, all subtasks in this hierarchy can be edited and revised by the user until the subtask "send report is performed." Users can cancel the submission at any time (which is indicated by the operator disabling: $[>$).

Users can describe an incident in several ways: for example, they can inform the incident category (such as a broken lamp or pothole), rate incident severity, or provide a description for it. Alternatives for such descriptions include sending a text, picture or video, and to call a hot line. The operator $|=|$ indicates that these activities can be done in any order.

The location of an incident is mandatory; otherwise, it would be very difficult to put the means in place to fix it. However, according to the context, the location can be informed as an address, a position on a map, a relative reference to a landmark (for example, in front of the Eiffel tower), or precise GPS coordinates.

A report can be completed by adding optional information about the time and the user. In some situations, users are able to report the time for the incident, which implies the user task "tell when the incident occurred" and the system task "record

when the incident is reported". The subtask "report time for the incident" is optional because it is very likely that incidents occur without any witnesses so that the exact time of an incident is unknown.

Users might be requested to provide personal identification either by identifying themselves or allowing the system to use personal coordinates already known by the system (for example, a cell-phone number). The level of identification of users can vary considerably from a system to another (for example, from anonymous to providing the user's name, personal address, cell-phone number, and user id). Precise user identification might help the city administration in many ways, for example, to prevent spam and false reporting and to contact the citizen when further information is needed. However, we shall notice that this is a requirement for the administration, not for the users. Indeed, incidents description might remain accurate and valid even if reported anonymously. For all of these reasons, user identification is described as optional.

After submitting a report, some users might want to follow up on an incident report. To allow this, the users should subscribe for notification; otherwise, the current legislation will not authorize the city administration to notify the user directly when the problem will be fixed. It is worth noting that the subscription for notification might also engage users in communication with the city administration. Some users might want to share reports using a social network or just be interested in seeing reports sent by others users. Of course, not all users will follow up on an incident report so closely, so this and all subsequent subtasks are described as optional.

The task model presented in Fig. 4 provides a comprehensive view of tasks related to incident reporting; however, it does not impose any particular design for the system. Indeed, many sequences of tasks performed on that task model lead to different suitable scenarios to reach the same goal. Using the simulator embedded into the HAMSTER editor, it is possible to extract all scenarios supported by a task model. Table IV illustrates only three possible scenarios extracted from the task model for incident reporting.

Scenario 1, presented by Table IV, will require a very minimalist system featuring a few text fields where users can provide textual description; locate the incident using landmarks and a button to send the report. Scenario 2 requires more information from users and, as a consequence, a more complex user interface, as users should select incident category, rate the incident (both danger and inconvenience), provide a textual description of the incident, provide an address, and provide personal coordinates which will ultimately be required when the users ask to subscribe for notification. Scenario 3 will ultimately require a far more complex user interface not only to accommodate the tasks described in scenario 2 but also to integrate tasks, allowing users to provide pictures of the incident and allowing the system to solve GPS coordinates that will be automatically added as part of the incident report.

The task model is considered valid as we can accommodate all of the scenarios identified during interviews and/or supported by the systems assessed in the survey.

Developing a Task Model From the Results The mapping between the task model and existing systems has been shown in Table II.

Based on the association of UX dimensions with tasks via the interpolation of user scenarios, it was possible to extrapolate the results in a single task model as shown in Fig. 5. In order to illustrate how Fig. 5 should be read, let us consider a simple scenario: "*A citizen sees a broken bench in a park and then decides to make an incident report. His motivation for reporting is to take actively part in the community. As the incident is of some importance to him, there is a variety of detailed information given in the report, including a photo and geo-localization data. Our citizen also wants that the incident report is also available for other users but he prefers to stay anonymous when using the application*". The task model presented by Fig. 5 supports this scenario. It is worth noting that this model has been decorated with rectangles that represent different UX dimensions (such as [AX] for visual experience, [ID] for identification). These decorations aim at highlighting where, during the task executions, UX dimensions were found to be important by interviewing participants. The importance is derived from the frequency of UX dimensions in the user's scenarios as shown in Table III. In order to illustrate how the task model presented by Fig. 5 should be read, we provide hereafter an extended scenario including UX dimensions:

> "*I am passing by at this park every Sunday and this bench has not been repaired for weeks* **[ID]**. *It is time now to report that, so it will get fixed. It is not really a problem or unsafe, but the bench is simply not usable in the current*

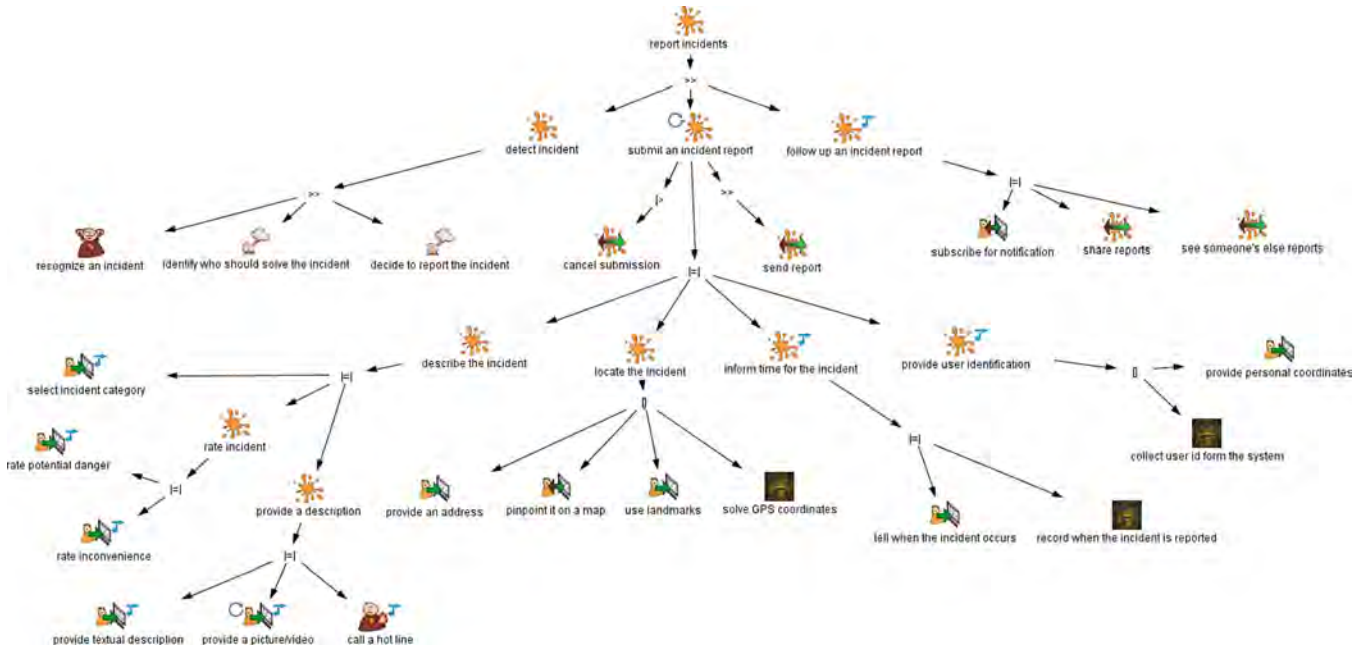


Fig. 4. Generic task model for reporting an incident.

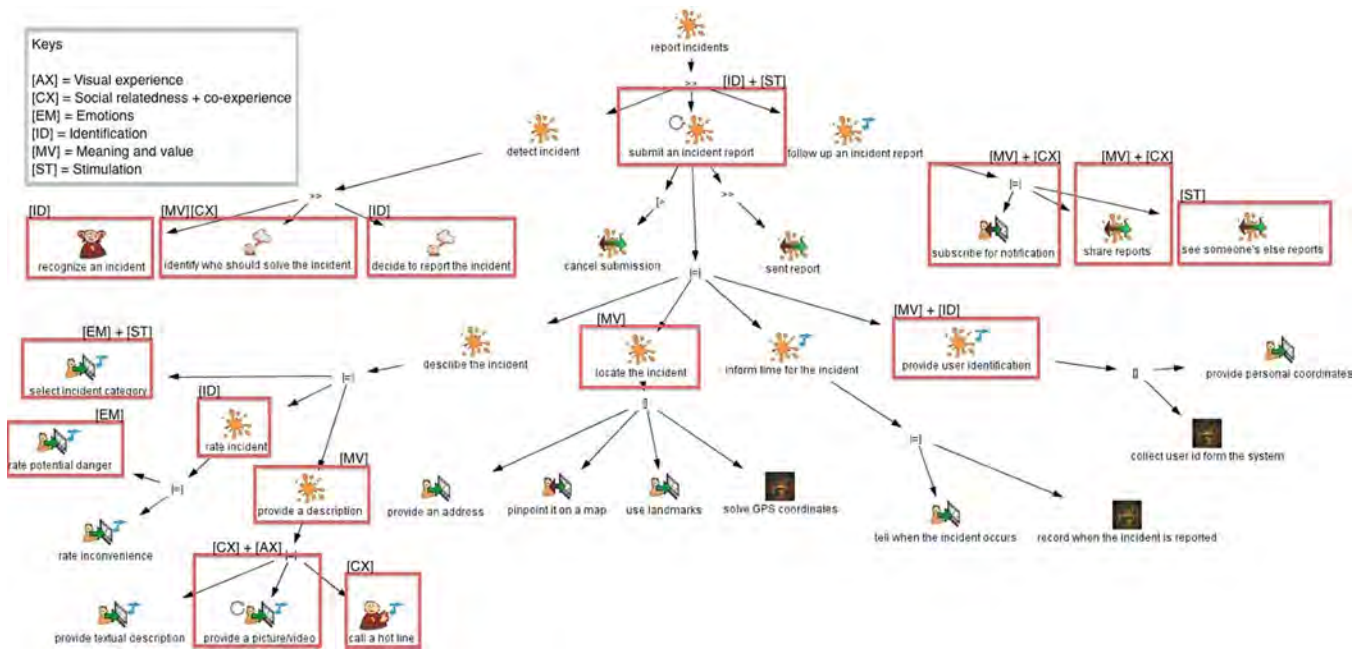


Fig. 5. Generic task and most important UX dimensions for each subtask.

state **[MV]**. *[:detect/recognize the incident:]. It seems important now to make sure that the appropriate person is informed about that bench* **[CX]**, *I think I should use the application to report the incident, because I want to be a good citizen* **[ID]**. *I think it is a good idea to send them a photo so they can see that the bench is really broken and that the wood has to be replaced. And when they see the photo they see that it is really there and so they will not need my contact information to have a proof that the broken bench really exists.* **[MV]** *[:describe the incident:]. (and so on)*".

This example shows how user tasks are interrelated to the UX dimensions. The various UX dimensions

do apply to the subtasks to a varying degree. We just refer to the most important UX dimensions in the diagram. There is one subtask that is not related to any UX dimension because it is optional and it is considered as an automatic task (by the system), such as [: inform time for the incident:]. All UX dimensions have been associated with the subtasks. It is interesting to notice that some tasks (such as [provide a picture/video]) can be influenced by more than one UX dimension (such as visual experience and aesthetic, and social relatedness coexperience) while other tasks can be considered neutral with respect to UX dimensions; which means that, even if necessary for the system, these tasks do not raise any particular UX. One possible implication for this association is that if designers want to reinforce or create a specific UX dimension, they might work on the tasks that might have an impact on users.

Table III provides a summary of the results collected by combining task models and interviews. Since our main focus was to analyze UX dimensions, the other segments of the corpus are not included in the table. The fourth column shows the total number of segments (in both interviews). About 45% of the interviews corpus talks about the overall UX dimensions (506 segments/1125). The fifth column shows the mapping of UX dimensions and user tasks.

It is interesting to notice that the UX dimensions *identification* (29.6%), *mean and value* (21.7%), and *social relatedness/coexperience* (26.7%) were frequent in all interviews. Except for *emotions* and *stimulation*, we could not find any difference between the numbers of participants reporting segments allowing the identification of *UX dimensions*. In the case of *emotions*, it was referred by all participants of the requirement interviews, while only 2/3 ($N = 6$ out 9) of participants of the scenario-based interview mention this dimension. The case of *stimulation* is more contrasting as only two participants mention this dimension during the scenario-based interview while all participants mention it during the requirement interview. This difference can be explained by the counting method since we only consider new scenarios provided by the participants themselves; the fact that the scenario-based interview prompted participants to focus on specific scenarios might have prevented them to talk spontaneously about new scenarios.

Conclusions, Limitations, and Suggestions For Future Research

This section concludes this paper. It aims at summarizing the main conclusions of our study, identifies its limitations, and discusses the implications for future research.

Conclusions This paper provided two main kinds of contributions that are worth discussing: the first one refers to the knowledge that can be obtained in terms of UX dimensions affecting self-services for reporting incidents; the second refers to the methodological aspects involving the triangulation of methods, which might provide some insight into associate UX dimensions and user interface design. Hereafter, we present our conclusions followed by limitations and future research.

A significant result from the present study is to point out the effect of UX dimensions in the task engaged by users. The semidirected requirement interviews showed several social implications for the task of reporting incidents in an urban context. These social implications can be translated by several UX dimensions, such as emotions (that motivate users to report an incident), user identification (tells which particular incidents users are willing to spend some time for writing a report), and visual experience (how aesthetics affect user perception of the system) that might influence the act of reporting an incident. However, the results show that the importance of UX dimensions is not equally distributed along the several subtasks that citizens engage in when reporting incidents. By using a model-based task analysis, it was possible to remove ambiguities present in the discourse of participants and then to formalize users' requirements. Moreover, model-based task analysis provided an accurate description of user tasks. As described in [47] and [50], tasks models not only improve the understanding of user tasks but they also can be used to assess whether an incident reporting system was effectively implemented to support the specified set of user tasks.

The second element for discussion is on the choice of methods. To identify UX dimensions that are important in the area of incident reporting, a triangulated method approach was chosen by combining a model-based task analysis, a survey of existing systems, and a set of requirement interviews. Model-based task analysis was chosen to provide a common ground for comparing incident reporting systems worldwide. The task model was also demonstrated as useful to anchor the findings expressed by users during semidirected

requirement interviews in terms of: users' scenarios that correspond to the general task model, and UX dimensions that are always reported in connection with tasks. Task models were thus used as a kind of "lingua franca," enabling us to identify a set of UX dimensions and their relation to (sub) tasks of incident reporting.

By combining these methods, it was possible to provide a clear representation of the tasks and to point out the lack of support for existing applications. This aspect of the present research will certainly help designers understand which tasks are worthy of more attention in order to produce the expected UX result. Conversely, designers can focus on specific UX dimensions and look up the tasks that users are more likely to perceive as desired effects. It is worth noting that instead of using a specific application, we investigate a generic tasks model from which several scenarios could be extracted and then analyzed. This step is extremely important for the future development of new incident reporting systems. We suggest that an approach for task analysis is extremely helpful to cover all design options to achieve a given goal.

The knowledge-obtained user requirements from incident reporting systems can be directly employed in the design of future applications. On one hand, this can be read as a set of recommendations for designing incident reporting systems. (See Table III.) On the other hand, this work has identified how UX dimensions affect tasks for incident reporting systems. So if governmental agencies want to provide high-quality incident reporting systems, they should concentrate their efforts on the design of applications that communicate positive UX dimensions. However, further investigation is necessary to determine whether (or not) users' perception of UX dimensions can influence the design of such systems in other countries.

As far as the use of methods is concerned, the proposed triangulation of methods might provide new insights for interpreting results related to overall UX and how to plot them into tasks models, which are aimed to support design activities. The mapping among methods was possible because it is easy to identify the concept of tasks in scenarios reported by users and to identify tasks behind the functions provided by existing systems. Despite this being a first attempt, we assume that approach can be reused in other studies related to UX and user interface design.

The investigation of incident reporting systems in the e-government domain is quite new. Despite the fact that many applications exist, we could not find any detailed analysis about the user tasks for declaring incidents. The lack of detailed analysis of user tasks can explain, at least partly, problems such as late adoption and definite rejection of applications. The discussion presented on UX presented in this paper might be considered useful for pointing out where to look at for overall user experience with e/m-government applications. We expect that these results could contribute to further research in the field and contribute to build a more general understanding about how UX dimensions affect users of e/m-government applications.

Limitations Our results are based on the citizens' point of view and, thus, only provide insights on UX dimensions that users felt were important. Indeed, the study does not taken into account the point of view of stakeholders who might have different criteria for assessing the importance of tasks. Another limitation is that the study was held in early phases of the development process. The UX dimensions identified are thus associated with requirements that are derived from citizens' expectations and previous experiences, as reported during the interviews. An additional investigation should be conducted to establish a correlation between UX dimensions identified in early phases of the development process and those dimensions that can be observed after system deployment and usage of applications. Another limitation to the interpretation of UX dimensions is from the transcripts of the interviews. Indeed, the coding of UX dimensions was based on expert reviews that are aimed to interpret the findings according to a predefined set of dimensions. Nevertheless, two measures have been taken to reduce the impact of subjectivity of interpretations: first of all, clear definitions of UX dimensions were defined before starting the coding, thus providing a scope for interpreting the segments; then, the coding was revised by independent experts, who conducted cross-checking, so that the coding provided makes sense.

Some of the results might provide insights about how users perceive their environment, their willingness to report incidents to authorities, and how UX dimensions are related to user tasks, which might influence the design of tools for improving the communications with administration. However, the interpretation of results is limited to the context of e/m-government applications. Moreover, it is limited by the cultural context in France where the

study was run. It is noteworthy that even if the identification of UX dimensions is valid for incident reporting system, in general, the detailed results are very specific to the e-government domain since we have to focus on citizens and the quality of their environment. Further studies will be required to investigate whether the results might remain valid for other kinds of applications.

Suggestions for Future Research Further studies will be required to take into account requirements raised by administrative agencies and deputies. Our next steps within the project thus include:

- The design and implementation of incident reporting system for mobile phones. Currently, we are designing an application based on a user-centered design approach that integrates the UX dimensions identified in this paper. Our aim is to deploy our application in several mobile platforms so that we can pursue our research toward the investigation of how specific features of mobile platform might affect the UX of systems for reporting incidents.
- A user testing experiment with end users is going to organize to assess the usability and the UX of the prototypes. Our main goal is to compare the results of actual use of running prototypes with the UX dimensions identified in the interviews.
- The availability of these prototypes also opens the perspective for establishing longitudinal studies on the evolution of the UX. After having tested the initial prototypes and having fixed the major usability and UX flaws that might be introduced during the development process, we aim to deploy the applications to a larger population to collect data about their experiences with the systems. Such a strategy allows the comparison of UX dimensions expressed as user requirements (the present study), user experience in the initial use, and user experience over 6 months using the system.

The Ubiloop project has focused on incident reporting in an urban context of use. However, as discussed in the introduction of this paper, incident reporting systems are relevant to a broader range of application domains. Despite the fact that we focus on incident reporting as an e-government service, some of our results can also be extended (in particular, when looking at geolocalization and temporal issues) to other domains, such as air-traffic management and health. Over the long term, we would like to investigate in which extension the UX dimensions identified in the present study remain relevant if the application domain changes. The investigation of influences of the domain can be held because the overall tasks remain the same. This issue is beyond of the scope defined for the Ubiloop project but the tasks and the inner context can provide an initial framework to address such scientific questions.

ACKNOWLEDGMENTS

This work is part of the Ubiloop project partly funded by the European Union. Europe is moving in France Midi-Pyrenees with the European Regional Development Fund (ERDF).

REFERENCES

[1] C. W. Johnson, *Failure in Safety-Critical Systems: A Handbook of Accident and Incident Reporting*. Glasgow, UK: University of Glasgow Press, 2003.
[2] J. T. Reason, "Human error: Models and management," *Brit. Med. J.*, 2000.
[3] M. Kaufmann, S. Staender, G. von Below, H. Brunnerc, L. Portenier, and D. Scheidegger, "Déclaration anonyme informatisée d'incidents critiques: Une contribution à la sécurité des patients," *Bull. Médecins Suisses*, vol. 84, 2003.
[4] M. Winckler, R. Bernhaupt, and F. Pontico, "Challenges for the development of user interface pattern languages: A case study on the e-government domain," *IADIS Int.J. WWW/Internet*, vol. 8, no. 2, pp. 59–84, 2010.
[5] T. El Kiki and E. Lawrence, "Mobile user satisfaction and usage analysis model of mgovernment services," presented at the 2nd Eur. Conf. Mobile Gov., Brighton, UK, 2006.
[6] G. Song, "Transcending e-Government: A case of mobile government in Beijing," presented at the 1st Eur. Conf. Mobile Gov., Sussex, UK, Jul. 2005.
[7] J. Moon, IBM Center for the Business of Government. (2004). From e-Government? Emerging practices in the use of M-Technology by state governments. [Online]. Available: http://www.businessofgovernment.org/report/e-government-m-government-emerging-practices-use-mobile-technology-state-governments
[8] F. Carcillo, L. Marcellin, and A. Tringale, "BlueTo: A location-based service for m-government solutions," in *Proc. EURO. mGOV.*, 2006, pp. 51–60.
[9] M. Ntaliani, C. Costopoulou, N. Manouselis, and S. Karetsos, "M-government services for rural SMEs," *Int. J. Electron. Secur. Digit. Forens.*, vol. 2, no. 4, pp. 407–423, 2009.

[10] G. Misuraca, "Futuring e-government: Governance and policy implications for designing ICT-enabled knowledge society," presented at the ICEGOV, Bogota, Colombia, 2009.
[11] mySociety Ltd., Fixmystreet. (2012, Jan. 30). [Online]. Available: http://www.fixmystreet.com
[12] P. Rossel, M. Finger, and G. Misuraca, "Mobile e-government options: Between technology-driven and user centric," *Electron. J. e-Gov.*, vol. 4, no. 2, pp. 79–86, 2006.
[13] I. Kushchu and H. Kuscu, "From e-government to m-government: Facing the inevitable," presented at the Eur. Conf. E-Government, Dublin, Ireland, 2003.
[14] M. Winckler, D. L. Scapin, F. Pontico, G. Calvary, and A. Serna, "Profiling user requirements for multi-target e-government applications: A case study," in *Proc. Int. Workshop Design Eval. e-Government Appl. Services*, Uppsala, Sweden, Aug. 8, 2009, vol. 492, pp. 9–16.
[15] R. Bernhaupt, *User Experience Evaluation in Games and Entertainment.* Berlin, Germany: Springer-Verlag, 2010.
[16] J. O'Looney, IBM Center for the Business of Government. (2003). Using technology to increase citizen participation in government: The use of models and simulation. [Online]. Available: http://www.businessofgovernment.org/report/using-technology-increase-citizen-participation-government-use-models-and-simulation
[17] D. P. Moynihan, IBM Center for the Business of Government. (2007). From forest fires to Hurricane Katrina: Case studies of incident command systems. [Online]. Available: http://www.businessofgovernment.org/report/forest-fires-hurricane-katrina-case-studies-incident-command-systems
[18] M. L. Meuter, A. L. Ostrom, M. J. Bitner, and R. Roundtree, "The influence of technology anxiety on consumer use and experiences with welf-service technologies," *J. Bus. Res.*, vol. 56, pp. 899–906, 2003.
[19] M. von Wilamowitz-Moellendorff, M. Hassenzahl, and A. Platz, "Dynamics of user experience: How the perceived quality of mobile phones changes over time," in *Proc. User Experience—Towards a Unified View, Workshop at the 4th Nordic Conf. Human-Comput. Interact.*, 2006, pp. 74–78.
[20] E. Law and P. Van Schaik, "Modelling user experience—An agenda for research and practice," *Interact. Comput.*, vol. 22, no. 5, pp. 313–322, 2010.
[21] E. Law, V. Roto, M. Hassenzahl, A. Vermeeren, and J. Kort, "Understanding, scoping and defining user experience: A survey approach," presented at the CHI, Boston, MA, 2009.
[22] E. Law, "The measurability and predictability of user experience," in *Proc. EICS*, New York, 2011, pp. 1–10.
[23] M. Pirker, R. Bernhaupt, and T. Mirlacher, "Investigating usability and user experience as possible entry barriers for touch interaction in the living room," in *Proc. Euroitv*, 2010, pp. 145–154.
[24] P. Hekkert, "Design aesthetics: Principles of pleasure in product design," *Psychol. Sci.*, vol. 48, no. 2, pp. 157–172, 2006.
[25] M. Hassenzahl, "The thing and I: Understanding the relationship between user and product," in *Funology: From Usability to Enjoyment*, M. Blythe, C. Overbeeke, A. F. Monk, and P. C. Wright, Eds. Dordrecht, the Netherlands: Kluwer, 2003, pp. 31–42.
[26] T. Lavie and N. Tractinsky, "Assessing dimensions of perceived visual aesthetics of web sites," *Int. J. Human-Comput. Studies*, vol. 60, no. 3, pp. 269–298, 2004.
[27] M. Hassenzahl and N. Tractinsky, "User experience—A research agenda," *Behav. Inf. Technol.*, vol. 25, no. 2, pp. 91–97, 2006.
[28] L. Alben, "Quality of experience: Defining the criteria for effective interaction design," *Interactions*, vol. 3, pp. 11–15, 1996.
[29] J. M. Quinn and T. Q. Tran, "Attractive phones don't have to work better: Independent effects of attractiveness, effectiveness, and efficiency on perceived usability," in *Proc. CHI*, New York, 2010, pp. 353–362.
[30] P. M. A. Desmet and P. Hekkert, "Framework of product experience," *Int. J. Design*, vol. 1, no. 1, pp. 57–66, 2007.
[31] S. Mahlke and M. Thüring, "Studying antecedents of emotional experiences in interactive contexts," in *Proc. CHI* , New York, 2007, pp. 915–918.
[32] K. M. Sheldon, A. J. Elliot, Y. Kim, and T. Kasser, "What is satisfying about satisfying events? Testing 10 candidate psychological needs," *J. Personal. Social Psychol.*, vol. 80, no. 2, pp. 325–339, 2001.
[33] E. Karapanos, J. Zimmerman, J. Forlizzi, and J.-B. Martens, "Measuring the dynamics of remembered experience over time," *Interact. Comput.*, vol. 22, no. 5, pp. 328–335, 2010.
[34] M. Hassenzahl, "Aesthetics in interactive products: Correlates and consequences of beauty," in *Product Exper.* New York: Elsevier, 2008, ch. 11, pp. 287–302.
[35] V. Jääskö and T. Mattelmäki, "Observing and probing," in *Proc. DPPI* , New York, 2003, pp. 126–131.
[36] P. Jordan, *Designing Pleasurable Products: An Introduction to the New Human Factors.* London, UK: Taylor & Francis, 2000.
[37] B. Gaver and H. Martin, "Alternatives: Exploring information appliances through conceptual design proposals," in *Proc. CHI* , New York, 2000, pp. 209–216.
[38] K. Väänänen-Vainio-Mattila, V. Roto, and M. Hassenzahl, "Now let's do it in practice: User experience evaluation methods in product development," in *Proc. Extended Abstracts Human Factors Comput. Syst.*, 2008, pp. 3961–3964.
[39] K. Väänänen-Vainio-Mattila, "Towards a life cycle framework of mobile service user experience," presented at the 2nd MIUX Workshop at MobileHCI, Amsterdam, the Netherlands, 2008.

[40] K. Väänänen-Vainio-Mattila, H. Väätäjä, and T. Vainio, P. Saariluoma and H. Isomäki, Eds., "Opportunities and chalengers of designing the service user eXperiense (SUX) in Web 2.0," in *Future Interaction Design II.* New York: Springer, 2008.
[41] W. H. Ittelson, H. M. Procshansky, L. G. Rivlin, and G. H. Winkel*, An Introduction to Environmental Psychology*. New York: : Holt, Rinehart, and Winston, 1974.
[42] H. Proshansky, A. Fabian, and R. Kaminoff, "Place identity: Physical world socialization of the self," *J. Environ. Psychol.*, pp. 57–83, 1983.
[43] A. Moles and E. Rohmer*, Psychologie de l'espace*. Strasbourg, France: Casterman, 1978.
[44] J. Lazar, J. H. Feng, and H. Hochheiser*, Research Methods in Human-Computer Interaction*. Hoboken, NJ: Wiley, 2010.
[45] B. G. Glaser and A. L. Strauss, *The Discovery of Grounded Theory: Strategies for Qualitative Research*. Chicago, IL: Adline, 1967.
[46] D. Diaper and N. A. Stanton, Eds.*, The Handbook of Task Analysis for Human-Computer Interaction*. Mahwah, NJ: Erlbaum, 2004, p. 650.
[47] E. Barboni, J. Ladry, D. Navarre, P. Palanque, and M. Winckler, "Beyond modelling: An integrated environment supporting co-execution of tasks and systems models," in *Proc. 2nd ACM SIGCHI Symp. Eng. Interact. Comput. Syst.*, New York, Jun. 2010, pp. 165–174.
[48] QDA Software Ltd., MaxQDA, 1995. [Online]. Available: http://www.maxqda.com
[49] M. Giulio, F. Paternò, and C. Santoro, "CTTE: Support for developing and analyzing task models for interactive system design," *IEEE Trans. Software Eng.*, vol. 28, no. 8, pp. 797–813, Aug. 2002.
[50] C. Martinie, P. A. Palanque, and M. Winckler, "Structuring and composition mechanisms to address scalability issues in task models," in *Proc. INTERACT*, 2011, vol. 3, pp. 589–609.

Marco Winckler received the Ph.D. degree in Informatics from the University of Toulouse 1 Capitole, France, in 2004. Currently, he is Assistant Professor in Computer Sciences, Université Paul Sabatier, Toulouse, France, and a member of the Interactive Critical System (ICS) research team of the Institute of Research in Informatics of Toulouse (IRIT). His research focuses on models, methods, techniques and tools to support the development of reliable, usable, and effective interactive systems. He serves as Vice-Chair for the IFIP working group 13.2 on Methodologies for User-Centered Systems Design and he is expert member of the IFIP TC 13 on Human–Computer Interaction.

Cedric Bach received the Ph.D. degree from the institute INRIA Rocquencourt, France, in 2004. Currently, he is a full-time researcher at the Ergonomics and Human Factors Department, Bertin Technologies, Toulouse, France. As a Human Factors specialist, he daily analyzes models, and designs and evaluates different types of human–machine interaction. For 10 years, his academic research has focused on the usability engineering in different domains, such as aeronautics, space, mobile technologies, and virtual environments.

Regina Bernhaupt received the Ph.D. degree in applied informatics at the University of Salzburg. She is invited Professor at the University Paul Sabatier/ Institute of Research in Informatics of Toulouse (IRIT). In addition, she is Director of User Experience Research at ruwido research (www.ruwido.com). She is an active member of the community on human-computer interaction, having organized conferences and several workshops. Her work focuses on how to evaluate usability and user experience in various contexts, especially for entertainment-oriented products and services. She recently edited a book on evaluating user experience in games and is the author of several international publications.

A Model-based Approach for Supporting Engineering Usability Evaluation of Interaction Techniques

Philippe Palanque, Eric Barboni, Célia Martinie, David Navarre & Marco Winckler
Institute of Research in Informatics of Toulouse
University Paul Sabatier
118 route de Narbonne, 31062 Toulouse CEDEX 9, France
{palanque, barboni, martinie, navarre, winckler}@irit.fr

ABSTRACT
This paper offers a contribution for engineering interaction techniques by proposing a model-based approach for supporting usability evaluation. This approach combines different techniques including *formal analysis* of models, *simulation* and, in particular, *analysis of log* data in a model-based environment. This approach is integrated in a process and is supported by a model-based CASE tool for modeling, simulation and evaluation of interactive systems. A case study illustrates the approach and operation of the tool. The results demonstrate that the log data at model level can be used not only to identify usability problems but also to identify where to operate changes to these models in order to fix usability problems. Finally we show how the analysis of log data allows the designer to easily shape up the interaction technique (as the results of log analysis are presented at the same abstraction level of models). Such as an approach offers an alternative to user testing that are very difficult to configure and to interpret especially when advanced interaction techniques are concerned.

Keywords
Interactive systems modeling, interaction techniques, performance evaluation, model-based usability evaluation.

ACM Classification Keywords
H5.m. Information interfaces and presentation (e.g., HCI).

General Terms
Verification.

INTRODUCTION
The development of interaction techniques to improve users' performance and satisfaction whilst using an interactive system has been a keystone for the research in the Human Computer Interaction field. Since the seminal work of Fitts [14] many refinements of the Fitts' law have been proposed to predict the difficulty for performing interactive tasks on a system, such as McKenzie extending Fitts' law for two-dimensional task [25] or the steering law from Accot & Zhai for constrained movement in trajectory-based tasks [1]. These laws can be applied at design time, indeed, to fine-tune the graphical representation of objects (such as size and shape) or to promote the development of new interaction techniques (e.g. multiple click, Drag and Drop…). In this context, innovation quite often occurs by extending existing interaction techniques. For example, Direct Manipulation enhanced interaction techniques such as Drag and Pop and Drag and Pick from Baudisch et al. [7] extend the Drag and Drop interaction technique by moving closer the possible reactive icons as soon as a drag event is produced. Boomerang of Kobayashi and Igarashi [20] proposes another extension to Drag and Drop interaction by defining mechanisms to suspend and resume a Drag and Drop interaction (after an interruption has occurred for instance). Another extension to standard interaction techniques is the work called "semantic pointing" of Blanch et al. [9] which proposes an adaptation of the control-display ratio based on the interactivity of the object below the mouse cursor. Other extensions have also been proposed taking into account nonstandard input and output devices such MAGIC pointing (which stands for Manual and Gaze Input Cascaded) [36] or Collomb & Hascoët [12] who adapt interaction technique such as Drag and Pop or propose new ones as Drag and Throw and Push and Throw [16] on wall-size display.

When proposing such enhanced interaction techniques, researchers typically build a prototype implementing it and then proceed to comparative assessments of the interaction technique with respect to existing ones. Prototypes are then assessments by the mean of user testing. Any necessary adjustment to improve performance or to fix usability problems are eventually done directly on the prototype. This design process raises the following main issues:

1. **Definition of the precise behavior of the interaction technique**: usually the interaction technique is only described in an informal way using text, screen captures or video (see Drag and Pop page [20]). Such informal descriptions leave a lot of information underspecified such as, for instance in the case of Drag and Pop, the size of the icons, the distance between the reactive icons and the one currently selected, … With more conventional interaction technique (such as Drag and Drop) the same issues remain leaving critical information not precisely defined.

EICS'11, June 13–16, 2011, Pisa, Italy.

2. **Ensuring the correctness of the interaction technique**: interaction technique testing is usually performed in an informal way by means of a sequence of tests. As the behavior of the interaction technique depends on the state of the interactive system and the behavior of the user, tests coverage remains very limited providing no mean to ensure that it will be robust and not fail [35]. Such concerns is well known in the field of interactive systems engineering and one of the only ways to address that problem is by using verification techniques on a formal model.
3. **Tuning the interaction technique to get optimal performance**: one clear example of fine-tuning mean for interaction techniques is the acceleration of the mouse cursor according to the speed movement of the mouse on the table. An efficient acceleration improves significantly selection time but usually remains hidden from the interaction technique description [18]. Each time a new interaction technique is defined, identification of the tunable element and how they have to be tuned are critical. Without an abstract and precise description of the interaction technique, such tuning is made at the code level on objects that are necessarily at a much lower level than the ones describing the interaction technique.
4. **Recording information during the usability tests:** when testing interaction techniques it is very difficult to record pertinent information which is usually at a low level of abstraction i.e. mostly at the level of user interface events [24]. Indeed, most of these events are directly produced by the input devices and thus makes it impossible to understand what happens when several input devices are used in a multimodal way as in a combined click on a bimanual interaction technique.
5. **Interpreting usability tests:** results to improve (both tuning and modification) the interaction technique from usability tests are difficult to interpret as these results usually remain at the event level and thus again are far away from the interaction technique states and behavior. Decisions for improvements are thus hard to make and implementations cumbersome and error prone thus leading to more usability tests than what would have been needed to define and tune the interaction technique.
6. **Reusing and refining interaction techniques**: makes it possible for other researchers and/or companies to reuse what has been proposed and potentially improve it. Building on previous works and knowledge developed in the field is a complex task and requires special means to promote it.
7. **Comparative assessment of interaction techniques**: if only final implementation is available; it is very difficult to determine which usability problems come from the design or from miscoding the interaction technique. Moreover, comparative assessment will require the execution of the same scenarios with all interaction technique being compared. For example, if interruption-tolerant systems are a concern, interruptible scenarios should occur with the same frequency and duration in all user testing sessions.

In order to address these problems, this paper proposes a model-based approach to support parts of the engineering process of interaction techniques. Next section introduces our approach that investigates several techniques including *formal analysis* of models, *simulation* and *analysis of log* data in a model-based environment. In particular we demonstrate how model-based approaches can facilitate the tasks of usability evaluation involving users. The subsequent section illustrates the approach with a case study for the modeling and evaluation a multimodal interaction techniques using two mice in a configuration similar to that embedded into the next generation of aircraft cockpits. Then, we discuss of the advantages and limitations of our approach in particularly with respect to previous works in the field.

APPROACH

The approach proposed in the section is based on models that are used as an input for a series of steps aiming to assess the specification of the interaction technique prior to usability testing with users.

Basic assumptions about the approach

The approach relies on the following assumptions:

- **Use of formal description technique**: the formalism used for describing interaction techniques has a strong impact on the kind of assessments that can be performed [29]. As we shall see, the precise assessment and fine-tuning of interaction techniques requires a complete and unambiguous description of the states of the interaction technique can be in as well as the events triggering state changes and the occurrence of fusion/fission of events.
- **Models must support prototyping:** this means that models run synchronously with user interface prototypes. Model execution is also a main requirement for checking usability properties such as liveness [19]. Moreover, models should be co-executable (i.e. changes in one model trigger changes in all related models in the framework) with any other models or software components use to build the system. Models execution will ensure that system models can be used as a prototype of the expected final application, which means that the prototype behaves exactly as modeled. This requirement implies the prototype is driven by the execution of models and for that there is an appropriate tool support.
- **Evolvability of models:** the approach aims at supporting the fine-tuning of the interaction techniques, which means that models should evolve according to the design of interaction technique until the expected performance fir the system has been reached.

Appropriate models and formalism are essential, but these are just part of the solution. On one hand, we need models to provide developers with a precise description of the system behavior. Based on that, designers can build the system as planned and evaluators can decide which are the most suitable and interesting scenarios to test with users (given that it will be extremely expensive to test all possible scenarios with users only). On the other hand, there must be a bridge between the specifications and the actual application offered to the users. As shown later, the results of *user testing* are directly associated to models instead of the implementation. This is made possible because *models* and *executable prototypes* remain consistent and compatible during the *implementation* phase. For that reason, the approach is heavily dependent of the fact that the formalism is executable and on the tool support which executes them.

So far only ICO [26] and Petshop [4] could support all the aspects described above. The ICO formalism has been demonstrated efficient for describing several techniques including 3D [27] and multimodal [22][30]. ICO models are executable and fully supported by the CASE tool PetShop which has been shown effective for prototyping interactive techniques [28].

Our approach employs methods such as formal analysis to ensure some basic system properties demonstrating the reliability of the interaction technique. It also integrates formal modeling and user testing by exploiting model-based log data. The use of model-based log analysis is very different from "classical" log as in [17] where the information collected is directly related to user's actions on the interface using the input devices (e.g. clicks, move …) thus far away from higher abstraction elements such as models or code.

The approach in a Nutshell

The general iterative approach for the assessment of interaction techniques is illustrated by Figure 1 (only colored parts are addressed by this paper). It is assumed that early steps such as requirements analysis, design and low-fidelity prototyping have been already carried out previously so that, the process starts by modeling activities.

Different models must be provided including: *task models* (for describing user tasks), *behavioral models* (using the ICO description technique for describing both the general behavior of the interaction technique and the treatment of low level events from input devices). From task models we *extract test scenarios* that are used as input both for the *execution* and the usability evaluation of the prototype (based on the execution of the ICO models). The prototype then follows a usability evaluation plan based on user testing, which produces logs data that can be classical *logs such as video, questionnaires… as well as model-based logs* (that are related to the ICO models).These logs thus support the assessment of the models of the interaction technique, and allow iterating while performance of these techniques does not meet the corresponding requirements. This is represented on Figure 1 by the box decision with its output towards *Interaction Technique OK* or towards *System and Task modeling* if there is still a need for tunings). The main idea is that usability problems discovered with the usability test can now point out these problems on the system models themselves, making it easier for designer both to locate where changes have to be made and additionally how to make them.

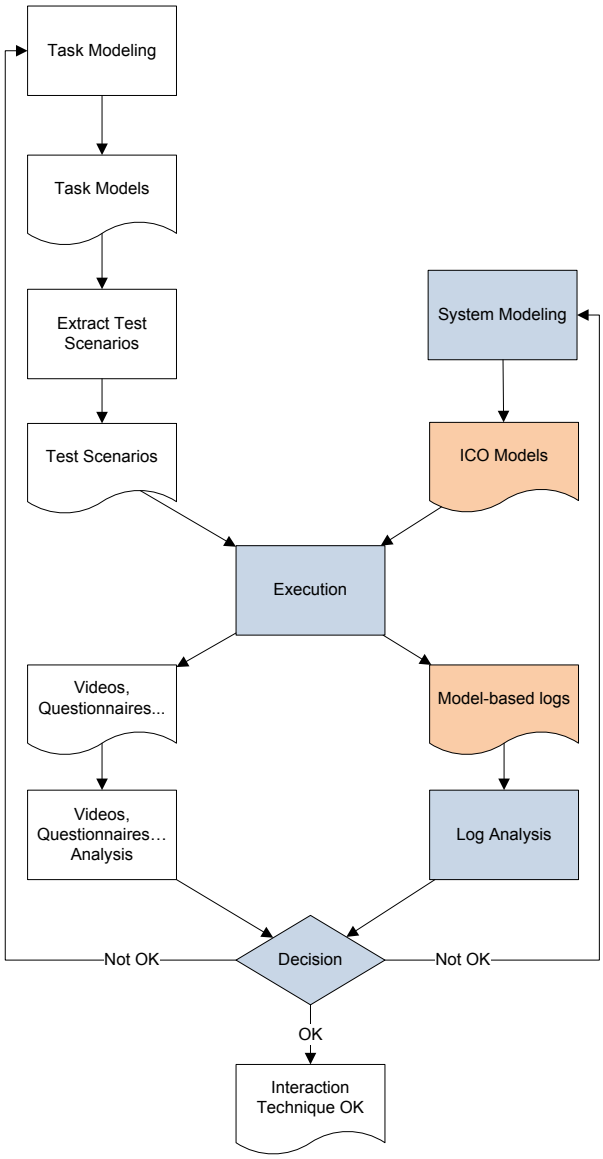


Figure 1. Development process for engineering interaction techniques.

The approach is quite generic but heavily dependent of a tool support and formalism able to support the main assumptions drew about models. In the present case, the approach exploits the ICO notation for describing the entire behavior of the interaction techniques and relies on the tool support Petshop for executing ICO models. Petshop not only support the modeling and execution of ICO models but also ensure the connection of models with the prototype. In previous work we have described how to use

the ICO formalism to several techniques including 3D [27] and multimodal [30] interaction techniques. The interested reader can find in [26] a detailed description of the formalism. Iterative prototyping of applications using the ICO formalism have also been fully described in previous work (see [28]) with the help of the tool support PetShop [5]. The rest of the paper details the different steps of the approach. Particular importance is given to model-based evaluation methods which include *formal analysis*, *simulation*, *performance analysis*, and *log data analysis* provided by user testing. These evaluation techniques are better described along the case study which is presented in the next section.

CASE STUDY

The objective of the case study is to present the various phases of the approach on a simple but realistic application which reproduces a classic desktop environment as shown by Figure 2. In this application a set of icons are presented in a window on a grid. The icons can be moved to different locations and deleted once they reach the trash bin icon. The user's goal is to remove all the icons on the user interface.

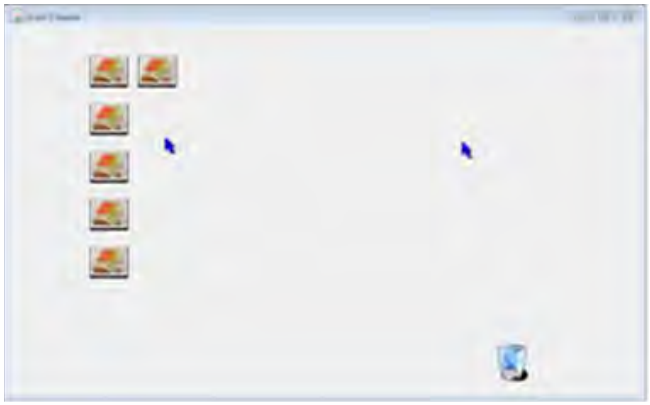


Figure 2. User interface of the case study.

Several interaction techniques could support this goal. However, to illustrate our approach we have investigated an interaction technique based on a *multimodal click* using two mice. To remove an icon from the desktop using the *multimodal click* a user must click on the icon and then on the trash bin. It can also be performed by two users. Indeed, the *multimodal click* is implemented in modern civil aircraft cockpits to allow the interaction of pilot and co-pilot.

Formal specification

Figure 3 and Figure 4 present the ICO models that specify the behavior of the *multimodal click* interaction technique. The Interactive Cooperative Objects (ICO) formalism is a formal description technique designed to the specification, modeling and implementation of interactive systems. It uses concepts borrowed from the object-oriented approach to describe the structural or static aspects of systems, and uses high-level Petri nets to describe their dynamics or behavioral aspects. In addition to these, specific components of the ICO formalism deal with presentation [6] and with command aspects of the interactive systems.

Figure 3 describes the behavior at the dialog level while Figure 4 describes the behavior of the associated mouse transducer. It is interesting to notice that the behavior of the interactive system there is a single model for two mice. In fact, both mice behave exactly the same. At Figure 3 the place *Frame* contains a token that is the reference to the presentation frame which is the user interface of the case study. Place *Trash* contains a token that is the reference to the trash element of the application. Each time a combined click (simultaneous click on both mice) is performed, an event *combinedClick* is received and it fires the *combinedClick* transition: the reference to the icon the target pointer of first mouse is on is stored in *obj1* and the reference to the icon the target pointer of second mouse is on is stored in *obj2*.

Once this transition is fired and consumed, a token is deposited in place *testIcon*. From this place, one transition only can be fired at a time; it will depend on which condition is fulfilled. If the target pointer of the first mouse is on a file icon (*obj1* is an instance of the *Icon* class) and that the target pointer of the second mouse is on the trash icon (*obj2* is equal to *trash*) or conversely, the transition *fileIconAndTrashSelected* will be fired and the file icon will be deleted. In all other cases, the transition *fileIconAndTrashNotSelected* will be fired, but no action will be triggered. In both cases, the token that is present in place *testIcon* will be consumed.

Figure 3. ICO model *combined click* specifying the interaction technique *multimodal click*.

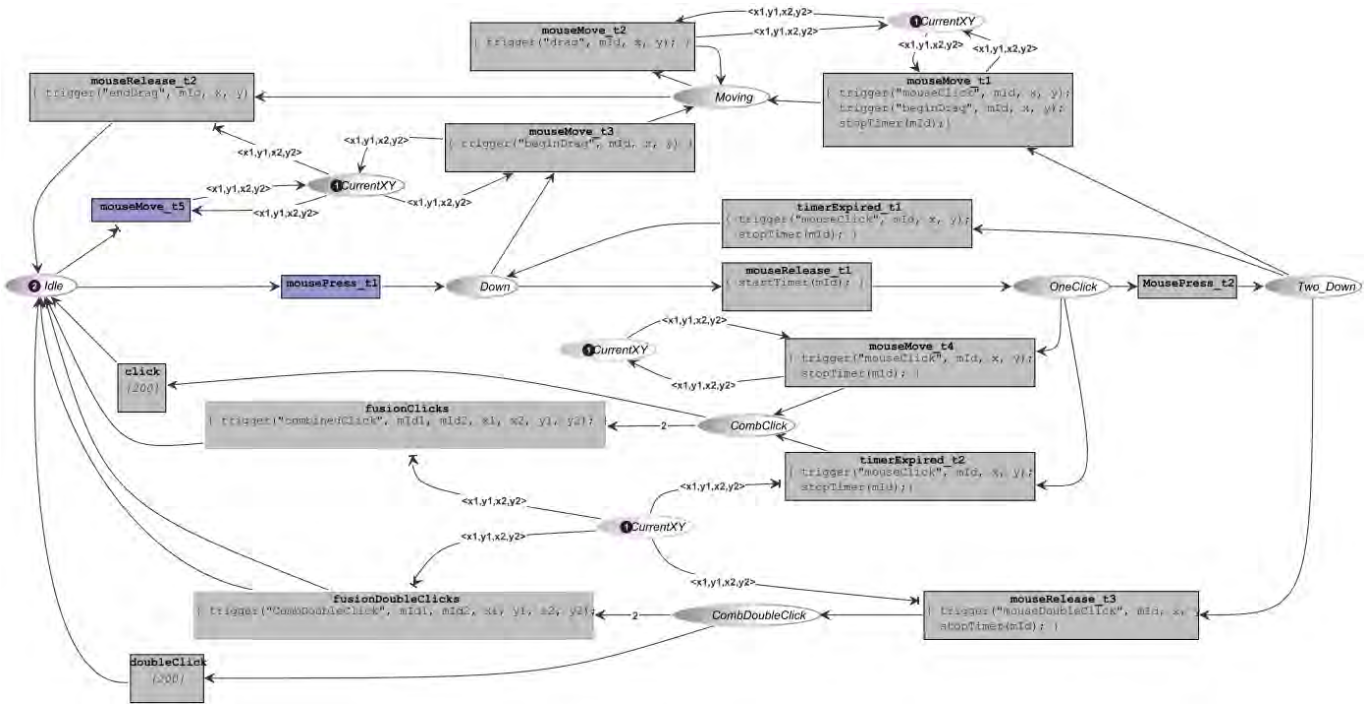


Figure 4. ICO model of the transducer controlling mouse events.

Figure 4 specifies the behavior of the physical interaction between the user and the system. This transducer takes mouse low level events as inputs such as *mouseMove* or *mousePress* and triggers them as higher level events such as *combinedClick* that will be handled by the interaction technique part (Figure 3). Place *currentXY* contains a token that keeps track of the mice target pointers' coordinates. For readability purpose, this place has been demultiplied into virtual places (functionality of the Petshop tool), which are clones of the initial one and are several views of this single place. It avoids overcrowding the figure. Each time a *mouseMove* event is caught (*mouseMove_t1*, *mouseMove_t2*, *mouseMove_t3*, *mouseMove_t4, mouseMove_t5*), the current coordinates are updated with the (x, y) deltas that have been done by the target pointers. In the initial state (2 tokens in place *Idle*, one for each mouse), the *mousePress_t1* and *mouseMove_t5* transitions are fireable (users can move or press their mouse). Once a *mousePress* event is caught, the *mousePress_t1* transition is fired and the corresponding mouse token is put into the place *Down*. If a mouseRelease event occurs, the transition mouseRelease_t1 transition will be fired and a timer armed in case of a future double click. The token is put into the *OneClick* place. If a *mouseMove* event occurs, the *mouseMove_t4* transition will be fired. If the timer previously set expires, the transition *timerExpired_t2* will be fired. In both cases, the timer is stopped, a *mouseClick* event is triggered and the token put into the place *CombClick*.

- If all the previously described sequence of events happens for both mice, two tokens are in the *CombClick* place, and the transition *fusionClick* can be fired. It can be fired only in this case because the weight of the corresponding arc is 2. This *fusionClick* transition, when fired, triggers a *combinedClick* event (which will be caught by the interaction technique model, Figure 3) and then deposits the 2 tokens back to the place *Idle*.
- In other cases for the place *CombClick*, if one token arrives in and that no other token arrives in within 200 milliseconds, the *click* transition will be fired and the token will go back to the place *Idle*.

The same principle is applied for the combined double click, which is not detailed here as not used in our case study.

Implementation in the PetShop environment

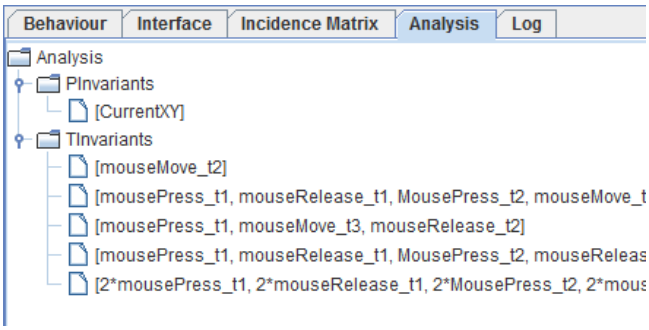
The described models are then connected to the user interface application software by means of activation functions and rendering functions. The activation function associates each event from the presentation part with the event handler to be triggered and the associated rendering method for representing the activation or the deactivation. For example in Figure 3, when a "combinedClick" event is received, the corresponding event handler which is the transition *combinedClick* is activated and a corresponding service (software in the transition's body) is executed.

The rendering function maintains the consistency between the internal state of the system (place, events) and its external appearance (UI rendering methods) by reflecting system states changes. When a token enters in one place, corresponding rendering events are going to be triggered. In this case study, there is currently no one, but we could for example add a rendering event associated to the place *testIcon* in Figure 3 that would make the mouse pointers blinking before entering into the appropriate transition.

Formal Analysis

Thanks to the CASE tool Petshop, it is possible to verify properties of the ICO model. Figure 5a and Figure 5b represent the invariant analysis of an ICO models presented in Figure 3 and Figure 4, respectively.

a) Analysis ICO model *combinedClick* (see Figure 3).



b) Analysis for ICO model *transducer* (see Figure 4)

Figure 5. Invariant Analysis.

The *PInvariants* sub trees in Figure 5a gather place invariants for the models. We can verify that places *Trash*, *Frame* and *CurrentXY* will always contain a token. This means that whatever state the model is in, and whatever events are produced, these places will never lose their resources. The *TInvariants* subtrees gather transition invariants for the models. This set of transition invariants for this interaction technique demonstrates liveness of the transitions handling the mouse events. For example, we can ensure that the user will always be able to move the pointing device from a position to another as the *mouseMove_t2* and *mouseMove_t5* are part of the invariants. According to the result of the analysis process, it can be decided to modify the model. For example, we can decide that the trash bin could not accept more than a certain number of icons. In [29] the interested reader can find a detailed explanation on how to perform such verification in the field of interactive systems.

Models simulation

Thanks to the PetShop environment [4], ICO models can be executed and simulated. This means that events produced by the users while interacting with the input devices will be received by the models that will evolve accordingly. So that the model drives the execution of the interactive systems and that (at the same time) user's actions trigger evolution on the model. Simulation and model-based checking of system specifications have been used to predict usability problems such as unreachable states of the systems or conflicting on events required for fusion. In [8] we provide a more complete account on scenario-based simulations.

Performance analysis

Performance analysis can be done in different ways accordingly to the information provided by models. For example, according to Fitts' law this size of the reactive icon is one of the parameter used for computing the difficulty index of the activity. Variation on this size will thus have an impact on the performance (time to select the Trash) but also the number of errors (an error would consist of moving the icon to be deleted on top of the icon of the Trash and then moving outside). Given that models can provide information such as target screen size, size of icons, etc, predictive information based on Fitt's law can be applied. Interruptions models can also combined with dialog models to assess the system tolerance and performance under different frequencies and duration of interruptions as presented in [31, 33]. If the values provided by the interaction technique are inside the range of values expected for performance, then we can proceed with the step of usability testing with users.

Log analysis of user testing

User testing is a very well-known evaluation technique based on the observation of users performing tasks with a system. It can be used to assess the usability evaluation of interactive systems with respect to its *effectiveness* (i.e. users can perform their tasks), *efficiency* (i.e. tasks can be performed without any unnecessary delay) and *user satisfaction.* Whilst *user satisfaction* is measured with specific questionnaires and interviews, *efficiency* and *effectiveness* are assessed through the execution of controlled tasks. Users can be extremely fast so that it is often difficult to observe all events raised by them. The protocol for conducting user testing is out of the scope of this paper. Our goal is just to demonstrate how user actions during usability testing can be collected and traced back to models. For that purpose, the Petshop tool was enriched with a log tool for assisting evaluators by recording user interactions into a log file. As models are connected to the prototype it is possible to play user interactions directly on models as they were recorded into log files.

In Petshop, each basic Petri net event like transition firing token movements (token added to a place, token removed from a place) is recorded in a single row. Only one log file is generated even if more than one model is involved. Log data are stored in XML files that can be transformed and exported to spreadsheet tools for statistically analysis.

Log files are structured as follows: the first field corresponds to the name of the model concerned by records. The second field represents the type of the Petri net node (i.e. *place* or *transition*). The third field corresponds to the given *node name*. The forth field represents the *action performed* associated to a node; for a transition it can be *fire a transition* whilst for a place the

actions are related to the token movement such as *tokenAdded* or *tokenRemoved*. The fifth field represents the *time elapsed* from the start of the application in milliseconds. The last two fields (named *data 1*and *data 2*) represent additional data related to the node type. For a place the field *data1* would represents the number of tokens *added* or *removed* at the same time, the field *data2* would point out to the objects embed by the token. For a transition, the field *data2* is only used to note the substitutions used for firing a transition.

We have asked a user to *delete as fast as possible all icons on our desktop applications using the multimodal click*. The execution of this task generated a huge series of events triggered on the Petshop environment (more than 4000 lines). Due to space reasons, Figure 6 presents a short excerpt of that user testing session. This excerpt will be used hereafter to illustrate different usability problems found with the interaction technique *multimodal click*.

It is noteworthy that the model-based log (presented in Figure 6) records all the change which occurs in the model described by Figure 3 and Figure 4 during the prototype executing; this include firing of each transitions, removal and addition of a token in a place (with the complete print of all the values describing the content of the tokens). It is possible to know which model is in charge of the event by looking at the field *class*.

Two transitions are important to handle the correctness of the interaction technique *multimodal click*. If the transition *fileIconAndTrashSelected* is fired it means that the user complete the deletion of an icon using the interaction technique. If the transition *fileIconAndTrashNotSelected* is fired the user failed to delete the icon but manage to do a proper *multimodal click*. The Figure 6 presents three successful combined clicks (lines 1425-1428, 1435-1438, 2897-2900) and one miss (lines 1699-1702).

In our scenario, the user completes his task after he has removed all icons. We can then calculate the time for completion of the task and the number of missed combined click by hands or with spreadsheet formulas. At this point, log data help to count the number of successful and failed tasks as well as the user performance. However, log data are also useful to understand complex sequences of events that may have led to unexpected user fails whilst trying to delete an icon. For that purpose we should take a look on the events recorded at the transducer level.

	class	type	name	action	time	data1	data2
605	mouse_transducer	transition	mousePress_t1	fire	8000		1*{evt:{mId=>1}}
			...				
609	mouse_transducer	transition	mouseRelease_t1	fire	9900		1*{evt:{mId=>1}}
			...				
613	mouse_transducer	transition	timerExpired	fire	10500		1*{evt:{mId=>1}}
			...				
617	mouse_transducer	transition	click	fire	10700		1*{evt:{mId=>1}}
			...				
650	mouse_transducer	transition	click	fire	10950		1*{evt:{mId=>2}}
			...				
752	mouse_transducer	transition	mousePress_t1	fire	12000		1*{evt:{mId=>1}}
			...				
758	mouse_transducer	transition	mousePress_t1	fire	12010		1*{evt:{mId=>2}}
			...				
761	mouse_transducer	transition	mouseRelease_t1	fire	12020		1*{evt:{mId=>2}}
			...				
774	mouse_transducer	transition	timerExpired	fire	12220		1*{evt:{mId=>2}}
			...				
781	mouse_transducer	transition	click	fire	12420		1*{evt:{mId=>2}}
			...				
794	mouse_transducer	transition	mouseRelease_t1	fire	13010		1*{evt:{mId=>1}}
			...				
1425	CombinedClick_Delete_File	transition	combinedClick	fire	20300		1*{evt:{x1=>40,x2=>400,y1=>40,y2=>400},presentationFrame=>javax....}
1426	CombinedClick_Delete_File	place	testIcon	tokenAdded	20300	1	Icon1,Trash
1427	CombinedClick_Delete_File	place	Frame	tokenRemoved	20300		
1428	CombinedClick_Delete_File	transition	fileIconAndTrashSelected	fire	20300		1*{Icon2,Trash}
			...				
1435	CombinedClick_Delete_File	transition	combinedClick	fire	22100		1*{evt:{x1=>40,x2=>400,y1=>80,y2=>400},presentationFrame=>javax....}
1436	CombinedClick_Delete_File	place	testIcon	tokenAdded	22100	1	Icon2,Trash
1437	CombinedClick_Delete_File	place	Frame	tokenRemoved	22100		
1438	CombinedClick_Delete_File	transition	fileIconAndTrashSelected	fire	22100		1*{Icon1,Trash}
			...				
1699	CombinedClick_Delete_File	transition	combinedClick	fire	23450		1*{evt:{x1=>290,x2=>400,y1=>40,y2=>400},presentationFrame=>javax....}
1700	CombinedClick_Delete_File	place	testIcon	tokenAdded	23450	1	null,Trash
1701	CombinedClick_Delete_File	place	Frame	tokenRemoved	23450		
1702	CombinedClick_Delete_File	transition	fileIconAndTrashNotSelecte	fire	23450		1*{null,Trash}
			...				
2897	CombinedClick_Delete_File	transition	combinedClick	fire	40340		1*{evt:{x1=>120,x2=>400,y1=>120,y2=>400},presentationFrame=>javax....}
2898	CombinedClick_Delete_File	place	testIcon	tokenAdded	40340	1	Icon6,Trash
2899	CombinedClick_Delete_File	place	Frame	tokenRemoved	40340		
2900	CombinedClick_Delete_File	transition	fileIconAndTrashSelected	fire	40340		1*{Icon6,Trash}

Figure 6. Excerpt of a Model-based log provided by PetShop.

In the model *mouse_transducer* (see Figure 4) the transition *click* is fired only when the two clicks are not combined. The substitution contains the ID of a mouse. According to the log the delay between click on mouse 1 (line 617) and click on mouse 2 (line 640) is 250ms. In fact, one of the main assumptions about this interaction technique is that a combined click is counted only if the user clicks on the two mice in an interval of less than 200ms. This makes possible to count number of failed combined click. Based on such as information, the evaluator can decide the means delay of failure for combined click and propose a better value for the trigger.

From the log data produce by our user whilst trying to perform his task, we could extract fine-grained information such as the time between the pressing the first mouse's button with the cursor on an and second mouse click on the trash bin. This time give us the total time for complete a task. The sequence of events described by the lines 752, 758, 761, 774, 781 and 794 are particularly revealing of a usability problem that would be very difficult to found without a model-based log environment. Indeed, it shows that the user failed a combined click when s/he pressed the two mice at the same time but forgot to release the button of first mouse (see *mid at* column *date* 2 in Figure 6) leading to a click after 200ms trigger on the second mouse.

EVOLVABILITY OF MODELS

According to the performance evaluation obtained with the log analysis, some fine tuning can be applied in the model to increase the performance of the interaction technique. This fine tuning can either be done during or before the simulation as in PetShop models can be modified while executed. The data collected during assessment steps, and in particular log data, can then be exploited to assess the performance of the interaction technique. If the performance does not fit the expectations, the log data can be used as input to modify or tune the model that will be simulated again.

As we have seen in our case study, the time of 200ms between two clicks with the two mice was too short in at least one situation. If this scenario is confirmed with more users, it would be reasonable to change the values used in the time interval. Such modification or fine-tuning of the interaction technique is made much easier as the information in the log is already related to the structure of the model.

The parameter used within the case study (time between two clicks) is just an example of what is possible to do, but the approach is generic enough to support the analysis of any parameter while the models explicitly highlight such parameters.

DISCUSSION AND RELATED WORK

Model-based approaches for supporting usability evaluation are not a new idea *per se*. In The HCI community many researchers have described user interface elements by means of models. The interested reader can find a structured state of art of model-based user interface in [26] where the different modeling techniques are categorized by criteria such as: Language (Petri nets-based, state-based, flow-based code-based and constraints-based.), Interaction Coverage (Low Level, Multimodality, Tangible, Fusion, Widget, Rendering and Dialog), Scalability (Toy Example, Case Study or Real Size application), Tool support and Expressiveness (Data Description, State Representation, Event Representation), Time (Quantitative or Qualitative), Concurrent Behavior & Dynamic Instantiation.

Beyond this descriptive aspect, models can also be used to support the evaluation of the user interface for properties (such as liveness and safety) or even for usability. Usability evaluation is a major concern in HCI as it is one of the very few means to assess/ensure matching between a system under design and the users' needs. It is interesting to point, however, previous work combining Model-Based approach and usability evaluation as:

- Paternò et al. [32] which develops MultiDevice RemUsine for remote evaluation of mobile interfaces;
- EvaHelper [3], a framework to help the developer (to perform evaluation on mobile applications) made up of EvaLogger to simplify and EvaWriter to process the log to extract from the log useful data about usability;
- ReModEl [11] (Remote MODel-based Evaluation) composed of a server (containing task structure and task states), a client (for the tester) and another client interface for the usability expert.

In addition, research work following the same philosophy can also be found for the Web domain such as remote automatic evaluation of Web applications based on a combination of web browser monitoring and task models or application, for example AWUSA [34]. Usability evaluation can also be found for more generic (other than mobile and web) purposes as in [13] coupling a workbench for automated usability evaluation called MeMo (Mental Models) and a model based framework for UI generation called MASP (Muli-Access Service Platform). This is similar in [21] with automatic usability evaluation. Lastly, [17] proposes the integration of Analysis of Usability Information from User Interface Events within a compositional model.

However, the scope of these contributions greatly varies accordingly to the formalism used for specifying the application and the tool support available. For example, approaches based on task models cannot capture fine-grained interactions (such as fusion of mouse events in multimodal interaction techniques) that are a basic requirement for evaluation user performance with interaction techniques. Moreover, these previous works propose model-based method to solve a particular problem with the user interface at a time.

Another trick problem is the fact that one should prove that the prototypes behave exactly as specified by models. For example, how to prove that a user will not loose of sight the

cursor even if he places the pointer away from screen area? Recently Gimblett and Thimbleby [15] have proposed an approach which consists in the discovery of models able to specify interaction techniques. However, discovery models do not driven the execution of the application. As a consequence, errors related to specific instance of data might occur during the execution of the application and the will never appear in models. Models can be gradually amended but behavior discovery in models is as random as discovery of software flaws with user testing, in the sense that user testing is limited to the scenarios proposed to users; some serious problems can be hidden in rare and difficult to sought possible scenarios.

CONCLUSION AND FUTURE WORK

In this paper we have presented an approach to support testing and evaluation of (multimodal) interaction techniques. The main feature of this approach is that testing and evaluation are based on models. This approach is targeted to novel and complex interaction techniques (including multimodal ones) which are much harder to assess using traditional user testing evaluation methods [23].

More precisely, we have presented ICO and Petshop as a formal approach to support a bi-directional relationship between models and executable prototypes, and to ease user testing by supporting a model-based log analysis. We have demonstrated in the section case study that some problems are very difficult to observe without appropriate tools. Using executable ICO models we were able to discover fine-grained sequences of events that provide evaluators with an accurate description of user actions. These tools not only are useful to observe the events triggered by users but they can also support the activities of identifying the part of the models related to the problem. Additionally, sequences of actions can be then (re)played on the model allowing a multiple inspection of error-prone scenarios.

At first sight, the development process proposed in this work might look familiar to some readers as it combines several existing techniques for supporting the evaluation of interactive systems. The approach is deliberately synergistic as it integrates our previous work on model-based evaluation techniques. It is the combinations of different techniques, rather than a single one that provides a solution for all the claims made in the introduction.

We are currently applying this research to the field of interactive cockpits where standards such as ARINC 661 [2] specification provide recommendations for WIMP interfaces. However, as the captain and first officer are interacting with the KCCU device (Keyboard Cursor Control Unit) the issue of multimodal interaction technique is raised bringing directly to the field of safety critical system problems usually investigated in general public interactive systems.

REFERENCES

1. Accot, J. and Zhai, S. 1997. Beyond Fitts' law: models for trajectory-based HCI tasks. In Proceedings of CHI'97 (Atlanta, USA). ACM, pp. 295-302.
2. ARINC 661 Cockpit Display System Interfaces to User Systems. ARINC Specification 661. Airlines Electronic Engineering Committee 2002
3. Balagtas-Fernandez F., Hußmann H., A Methodology and Framework to Simplify Usability Analysis of Mobile Applications. In Proc. of ASE'2009, p. 520-524.
4. Barboni, E., Ladry, J-F, Navarre, D., Palanque, P., Winckler, M. *Beyond Modelling: An Integrated Environment Supporting Co-Execution of Tasks and Systems Models.* In proceedings of ACM Symposium on Engineering Interactive Systems (EICS'2010), June 19-23, 2010, Berlin, Germany. Pages: 165-174
5. Bastide, R., Navarre, D., and Palanque, P. 2002. A model-based tool for interactive prototyping of highly interactive applications. In Proceedings of CHI '02, pp. 516-517.
6. Bastide R., Palanque P., Le Duc H., and Munoz J. (1998). Integrating Rendering Specifications into a Formalism for the Design of Interactive Systems. Proc. of the 5th Eurographics workshop on Design, Specification and Verification of Interactive systems DSV-IS'98. Springer Verlag.
7. Baudisch, P., Cutrell, E., Robbins, D., Czerwinski, M., Tand-ler, P. Bederson, B., and Zierlinger, A. Drag-and-Pop and Drag-and-Pick: Techniques for Accessing Remote Screen Content on Touch- and Pen-operated Systems. In Proc. of Interact 2003, pp. 57-64.
8. Bernhaupt, R., Navarre, D., Palanque, P., Winckler, M. Model-Based Evaluation: A New Way to Support Usability Evaluation of Multimodal Interactive Applications. In Maturing Usability: Quality in Software, Interaction and Quality (chapter 5). Law E., Thora Hvannberg E., Cockton G. & Vanderdonckt J. (Eds.). Springer Verlag (HCI Series), 2007, p. 96-122.
9. Blanch R., Guiard Y. and Beaudouin-Lafon M. Semantic Pointing: Improving Target Acquisition with Control-Display Ratio Adaptation. In Proceedings of CHI 2004, (Vienna, Austria, April 2004), pp. 519-526.
10. Bowman, Doug A., Hodges, Larry F. Formalizing the Design, Evaluation, and Application of Interaction Techniques for Immersive Virtual Environments. J. Vis. Lang. Comput. 10(1): 37-53 (1999).
11. Buchholz G, Engel J, Märtin C, Propp S. Model-based usability evaluation - evaluation of tool support. HCII 2007, (Beijing, China) pp. 1043-52. Springer-Verlag.
12. Collomb, M. and Hascoët, M. 2008. Extending drag-and-drop to new interactive environments: A multi-display, multi-instrument and multi-user approach. Interact. Comput. 20, 6 (Dec. 2008), pp. 562-573.

13. Feuerstack, S., Blumendorf, M., Kern, M., Kruppa, M., Quade, M., Runge, M., and Albayrak, S. 2008. Automated Usability Evaluation during Model-Based Interactive System Development. In Proceedings of TAMODIA'08 (Pisa, Italy). LNCS, vol. 5247. Springer-Verlag, pp. 134-141.
14. Fitts, P. M. 1954. The information capacity of the human motor system in controlling the amplitude of move-ment. Journal of Experimental Psychology, 47, pp. 381-391.
15. Gimblett, A., Thimbleby, H. User interface model discovery: towards a generic approach. In Proc. of the 2nd ACM SIGCHI symposium on Engineering interactive computing systems (EICS '10). ACM, New York, NY, USA, 145-154.
16. Hascoët, M. 2003. Throwing models for large displays. In: HCI2003, Designing for society, vol. 2. British HCI Group. pp. 73-77.
17. Hilbert, D. M. and Redmiles, D. F. 2000. Extracting usability information from user interface events. ACM Comput. Surv. 32, 4 (Dec. 2000), pp. 384-421.
18. Hinckley, K., Cutrell, E., Bathiche, S., and Muss, T. 2002. Quantitative analysis of scrolling techniques. In Proceedings of CHI'02 (Minneapolis, USA), pp. 65-72.
19. Jiao, L., Cheung, T-Y, Lu, W. Characterizing Liveness of Petri Nets in Terms of Siphon. ICATPN 2002, LNCS 2360, pp. 203-216, Springer-Verlag.
20. Kobayashi, M. and Igarashi, T. 2007. Boomerang: suspendable drag-and-drop interactions based on a throw-and-catch metaphor. In Proc. of the 20th Annual ACM Symposium on User interface Software and Technology (Newport, Rhode Island, USA, October 07 - 10, 2007). UIST '07. ACM, New York, NY, 187-190.
21. Kristoffersen, S. 2009. A Preliminary Experiment of Checking Usability Principles with Formal Methods. In Proc. of ACHI'09. IEEE Computer Society, 261-270.
22. Ladry J-F., Navarre D., Palanque P. Formal Description Techniques to Support the Design, Construction and Evaluation of Fusion Engines for SURE (Safe Usable, Reliable and Evolvable) Multimodal Interfaces. In: ICMI-MLMI 2009, Cambridge, Massachusetts, USA, ACM, p. 135-142, 2009.
23. Lalanne D., Nigay L., Palanque P., Robinson P., Vanderdonckt J., and Ladry J-F. 2009. Fusion engines for multimodal input: a survey. In Proceedings of the 2009 international conference on Multimodal interfaces (ICMI-MLMI '09). ACM, New York, USA, 153-160.
24. Mackay, W.E. Ethics, lies and videotape, in *Proceedings of CHI '95* (Denver CO, May 1995), ACM Press, 138-145.
25. MacKenzie, I. S. and Buxton, W. 1992. Extending Fitts' law to two-dimensional tasks. In Proceedings of CHI'92 (Monterey, United States), ACM, pp. 219-226
26. Navarre, D., Palanque, P., Ladry, J., and Barboni, E. 2009. ICOs: A model-based user interface description technique dedicated to interactive systems addressing usability, reliability and scalability. ACM TOCHI 16,4 (Nov. 2009), pp. 1-56.
27. Navarre, D.; Palanque, P.; Schyn, A.; Winckler, M.; Nedel, L.; Freitas, C. M. D. S. A Formal Description of Multimodal Interaction Techniques for Immersive Virtual Reality Applications. In Proceedings of TC13-IFIP Conference INTERACT 2005, Rome, Italy (12th-16th September 2005). pp. 170-183.
28. Navarre D. Palanque P. Bastide R, and Sy, O. Structuring interactive systems specifications for executability and prototypability. In Proceedings of DSV-IS'2000. LNCS. n° 1946, Springer Verlag, pp. 145-161.
29. Palanque P. & Bastide R. (1995). Verification of an Interactive Software by analysis of its formal specification. Proc. of the IFIP TC 13 Interact'95, pp. 191-197.
30. Palanque, P., Bernhaupt, R., Navarre, D., Ould, M., Winckler, M. Supporting Usability Evaluation of Multimodal Man-Machine Interfaces for Space Ground Segment Applications Using Petri net Based Formal Specification. Ninth International Conference on Space Operations, Rome, Italy, June 18-22, 2006.
31. Palanque, P., Winckler, M., Ladry, J-F., ter Beek, M., Faconti, G., Massink, M. A Formal Approach Supporting the Comparative Predictive Assessment of the Interruption-Tolerance of Interactive Systems In Proceedings of the ACM Symposium on Engineering Interactive Computing Systems (EICS 2009) Pittsburgh, USA. pp. 211-220. ACM Press.
32. Paternò, F., Russino, A., Santoro, C. (2007) Remote evaluation of Mobile Applications. In Proceedings of TAMODIA 2007, Toulouse, France, LNCS Vol. 4849.
33. ter Beek, M., Faconti, G., Massink, M., Palanque, P., Winckler, M. Resilience of Interaction Techniques to Interrupts: A Formal Model-based Approach 12th IFIP TC 13 Conference on Human-Computer Interaction (INTERACT 2009) Uppsala, Sweeden, Springer Verlag, LNCS n° 5727, p. 494-509.
34. Tiedtke, T., Märtin, C., Gerth, N. Awusa, A tool for automated website usability analysis. In: PreProceedings of the 9th Int. Workshop DSV-IS 2002, Rostock, Germany, pp. 251-266 (2002).
35. Yuan X., Cohen M. & Memon A. GUI Interaction Testing: Incorporating Event Context", IEEE Transactions on Software Engineering, 2010, IEEE Computer Society.
36. Zhai, S., Morimoto, C. Ihde, S. (1999) Manual And Gaze Input Cascaded (MAGIC) Pointing. In Proceeding of CHI'99, pp. 246-253.

Structuring and Composition Mechanisms to Address Scalability Issues in Task Models

Célia Martinie, Philippe Palanque, and Marco Winckler

Institute of Research in Informatics of Toulouse (IRIT), University Paul Sabatier, France
{martinie, palanque, winckler}@irit.fr

Abstract. Along tasks analysis and modeling history it has been demonstrated by experience that task modeling activities become cumbersome when performed on large, real-life systems. However, one of the main goals of task models is to provide designers with a structured and complete description of the users tasks especially when these user tasks are numerous and/or complex. Several authors proposed to handle that problem by providing tools aiming at supporting both construction and understanding (usually via simulation) of models. One of the most popular examples is CTTE environment which is dedicated to the engineering of CTT task models. The paper shows how to extend notations for task description with two kinds of mechanisms: composition and refinement/abstraction. Refinement/abstraction mechanisms make it possible to decompose a task model into several models and to interconnect them. Composition mechanisms make it possible to define communication means between task models. The paper proposes a precise definition of these mechanisms, their integration into a notation for describing task models and demonstrates that altogether, these two structuring mechanisms support the effective exploitation of task models for large scale application. The use of the mechanisms is presented on a real-life case study from the space domain describing operators' tasks to monitor a satellite and manage failures.

1 Introduction

In the field of Human-Computer Interaction (HCI), the user centered paradigm [22] has reached a popularity level where the question about it, is no longer whether it is valid or not but rather how it should be embedded in the development process of interactive applications. Many notations, processes and tools have been proposed for gathering information about the users either in formal (via formal requirements as in [17] or formal task models [27]) or informal ways (via brainstorming [7, 9] or prototyping [31]). One of the main advantages put forward by notations is that they make it possible to handle real-size applications and, if provided with a formal semantics, make it possible to reason about the models built with the notations[1] and assess the presence or absence of properties.

While exploiting a notation for building a model of the real world, usually, two main activities are carried out in order to tackle the problem of size of the resulting model:

Abstraction: details from the real world will be omitted and abstracted away if they are not of interest for the use that will be made of the model. For instance, using a finite state automata (FSA) [14] the continuous evolution of the world (such as in a flow of liquid

[1] In the rest of the paper we make the following distinction between a notation and a model: a notation provides a mean for representing information from the real world. The resulting use of the notation is called model. If a notation is defined formally it is called a formalism.

while emptying a bottle) will be discretized in a set of states (for instance three different states: Full, Emptying and Empty) representing an abstraction of the infinite number of states of the real world.

Filtering: the notation plays the role of a filter capturing the information it is able to capture and letting through what cannot be captured. Taking again the example of a FSA, information about the time elapsed for emptying the bottle cannot be represented and thus will not appear in the model.

Of course, notations can be (and have been) extended in order to be able to represent more information than initially planned. FSA have been extended in [38] to handle data (such as for instance the size of the bottle) or even time [30]. However, the most widely used notations stay away from universality and embed strong abstraction and filtering. Examples of such notations are UML class diagram [29] or entity relationship diagrams [5] only capturing data-related information or basic Petri nets [23] or FSA [14] only capturing behavior-related information. Indeed, despite these mechanisms a notation capturing too much information would end up in oversized and unmanageable models. To represent a larger part of the world, several models have to be built using several notations. The complexity then lays in defining processes and tools making it possible to ensure conformance and consistency of the various models corresponding to different views of the same world.

In the field of HCI many notations have been proposed for capturing in models the various elements of socio-technical systems. In the last decades, several tasks notations have been developed as means to describe work carried out by users whilst interacting with a system [8]. Despite the fact that various specific task notations exist, they are mainly structured around two concepts: task decomposition (often represented as a hierarchy) and task flow (for showing the order in which tasks are executed) [16]. When adequately combined, these concepts can provide an exhaustive and complete representation of large quantity of information in a single model. However, as discussed by Paterno & Zini [28], when applied to real-life systems, tasks notations end up in very large, hard-to-manage models thus making task modeling a time-consuming and sometimes painful activity.

This paper introduces some generic structuring mechanisms to tasks notations for describing task models in order to overcome the problems identified above and to make it possible to represent users' activities in large socio-technical systems. Section 2 discusses the structuring mechanisms offered by existing tasks notations. It also includes an abstract presentation of the proposed mechanisms. Section 3 introduces informally the real-life case study from the satellite ground segments domain focusing on controllers' tasks to monitor a satellite and manage failures. It also presents an initial approach for modeling controllers' tasks with CTT [26] and highlights the need for structuring mechanisms. Section 4 gives an overview of HAMSTERS (introduced in [1]) and details how the notation has been extended to integrate the new mechanisms. Section 5 details how the various elements of the notation are used to model the entire case study. Section 6 is dedicated to the analysis in terms of efficiency of the proposed mechanisms for structuring task models. Section 7 concludes the paper and presents research directions for future work.

2 Structuring Issues for Tasks Notations

Task models are useful when designing real-size systems as they aim at representing a large quantity of information related to user goals and to the activities to be carried out in order to reach these goals.

Notation-based structuring: Whilst some task notations such as UAN [13] and GOMS [4] are mainly textual, CTT [26, 24], MAD [33] or AMBOSS [1] provide graphical representations, which favor legibility and understanding of complex problems [6]. Every task notation proposes hierarchical task decomposition. This hierarchical representation of tasks is grounded in psychology [32] reflecting in the models the way people structure their activities. The decomposition of tasks in subtasks enables the creation of several levels in the tree hierarchy of tasks; this has been extensively used for supporting the mechanisms of abstraction and refinement in task models.

Not all notations used for describing activities are hierarchical. Indeed, notations used in the workflow domain (such as YAWL [36] or BPMN [36]) are graphs but hierarchical representations are more prominent in task model notations as this enforces abstraction/refinement mechanisms [36]. In hierarchical representations, the order of execution of tasks is given by the navigation through the hierarchy of tasks and the temporal operators between sibling tasks (i.e. tasks in the same level of the hierarchy). By combining hierarchy and temporal operators, it is possible to have different tasks models that exhibit equivalent behavior [10].

Another important issue for structuring task models refers to the relationship between tasks and goals. In the Task Knowledge Structure (TKS) [15] for instance, a specific substructure corresponds to each goal. CTT [26, 24], MAD [33], and GTA [35] provide a more flexible organization of tasks so that a goal can be reached in different ways. Van Welie, van der Veer and Eliëns [35] argue that the higher the tasks in the hierarchy the closer they are of organizational goals, whilst low-level tasks are more likely to represent individual goals. Even if the distinction between individual's and organization's goals is sometimes difficult to settle, it has in the end an impact on how models are structured.

Structuring mechanisms can sometimes be found in unexpected places. For example, CTT [26] includes the operator *iterative tasks* (symbol next to a task*) so that repetitive tasks are represented only once even though they may occur many times. That operator is mainly used for describing iterative behavioral aspect of the task but additionally makes it possible to significantly reduce the size of a task model. CTT offers another structuring mechanism via the notion of collaborative tasks modeled using the operator *connecting tasks* (symbol↔). In a nutshell, this notion embeds in models the fact that some tasks require the involvement of several persons (or roles) to be performed. The task of each person is modeled in a task model and the collaboration (i.e. order dependences between the tasks of the operators) is represented in another additional model. While necessary to represent collaborative activities, this notion and its related operator results in a role-based structuring of task models splitting a bigger model in several smaller models. CTTE is the unique tool currently available supporting *collaborative tasks*. However, only qualitative temporal relationships can be represented as no information (i.e. data flow) can be exchanged between the models.

An important issue that must be considered when structuring task models is its potential for reuse [3]. In fact, some tasks (such as login into systems for instance) remain structurally similar in different applications. This feature has been introduced in notations like CTT [26] so that some generic tasks can be used as building blocks that can be cut and pasted in models. However, a modification of one of these copies is not reflected to the other ones. Concerned by the reusability of tasks models, Gaffar et al. [12] have investigated structuring mechanisms around the notion of patterns to be used in task models. They propose a method and a tool to model generic tasks patterns as building blocks that can be instantiated and customized when modeling real-life socio-technical systems. One of the advantages of task patterns is the fact they provide more flexibility for reuse as they correspond to a generic problem.

Lastly, Sining et al. [34] introduce the notion of modular task models at a more generic level than the recursive tasks offered in CTT. Such modules would be structured in a task diagram describing in an exhaustive way their relationship. This concept is very similar to some one proposal in this paper. However, its implementation by means of task diagram adds unnecessary complexity (a new notation) and does not support the exchange of information between the modular models as proposed in the current paper.

Tool-based structuring: Some problems associated to the management of large and complex tasks models, can be overcome with the help of tool support. Currently, several of the tasks notations presented above are accompanied with an edition (and sometimes a simulation) environment. For instance, EUTERPE [35] supports the Groupware Task Analysis (GTA) notation, K-MADe supports the *Méthode Analytique de Description* (MAD) notation [33] and CTTE [25] supports CTT notation. These environments exploit the fact that task models are naturally represented as a hierarchy of sub-tasks, to implement abstraction/refinement mechanisms by supporting actions such as pruning/expanding sub-parts of the trees. More recently [28], CTTE tool has been enriched with visualization techniques (fisheye view and semantic zooming) to better support creation and management of CTT tasks models. Moreover, through collaborative tasks (previously described), CTTE is the only environment currently available supporting the decomposition of tasks in several communicating diagrams (even though the original goal was the representation of collaboration and not to support models structuring).

Abstraction/refinement and composition mechanisms: This section has illustrated the mechanisms currently available for structuring task models. However, when it comes to large systems these mechanisms and tools are insufficient (see CTT model in Fig. 2). We propose two new mechanisms to handle more efficiently this complexity. The first one is based on refinement/abstraction principle and makes it possible to decompose a task model in several models and to interconnect them. A large task model can thus be decomposed into several sub-models. These sub-models can then be reused (as a "copy") in various places of the same model and even in other models. Each time one of these parts is modified, the modification is reflected in all the other "copies". The Composition mechanism makes it possible to define communication mechanisms between task models. This task model structuring mechanisms is similar to procedures calls in programming languages and parameterization is possible via input and output parameters. In order to keep the same semantics as for the single model, communication protocols have also been introduced.

3 Case Study: Ground Segment Operations for PICARD satellite

In order to illustrate scalability problems of task models, in this section we present a real-life system belonging to a ground segment application that is currently used to monitor and to control the Picard satellite[2]. The task models presented in this section exploit standard structuring mechanisms such as those currently available in the CTT notation (and presented above). These models will then be revised in section 4 to include the structuring mechanisms of abstraction/refinement and composition that we have designed to overcome the limitations that were faced due to the large number of tasks and activities.

[2] The Picard satellite was launched by CNES in June 2010 and is dedicated to solar observation.

3.1 Informal description of case study

The case study belongs to the space domain and more precisely to the command and control interactive systems of satellites. **Fig. 1** presents a schematic view of a space system as defined in the European Standard ECSS-E-70 [11]. Such space systems are made of two parts: the *space segment* (including the spacecraft) and the *ground segment* which is made up of the ground station segment (antennas for communication with the space segment) and the mission control system. Our focus is the *operation control system* (bottom-left icon in **Fig. 1**) which belongs to the mission control system. This system is in charge of maintaining the spacecraft in operation and is thus heavily related to the spacecraft structure and functioning. Next paragraphs describe the context, goals, roles, tasks and system functions of the case study that have to be represented in the task models. We only provide here the excerpt necessary to discuss the various models and the way they can be decomposed and structured. Indeed, tasks and operations of a mission control system are much more numerous than what is presented here but this excerpt is representative of the real system in terms of complexity and of operators' everyday activities.

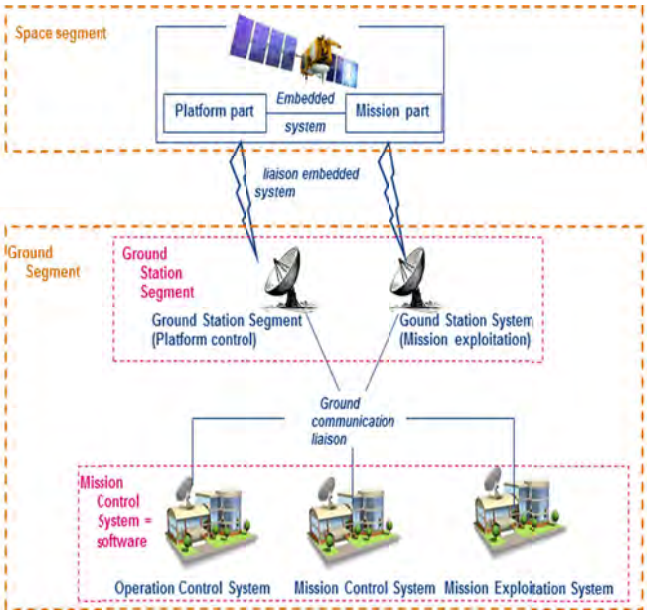


Fig. 1. The satellite application domain in a nutshell

3.1.1 Ground Segment Operations: the context

The Operation Control System consists in a room containing several computers and various hardware equipment dedicated to: 1) receive information from the satellite via the ground station segment, 2) organize and record this information and 3) send commands to the satellite (space segment). In this room, controllers (usually called operators) are in charge of monitoring the state of the space segment platform part by examining periodically several parameters such as current voltage of the power generators, current satellite mode, and status of the communication link. The main goal of operators is to keep the satellite in a nominal mode, so that the mission (for instance observation of the sun by taking pictures) can be performed. If a severe incident occurs, the satellite can automatically switch to a

mode called “survival mode” where all functions are disabled and the mission is stopped (i.e. data gathering is stopped and already collected data might be lost). In case of the occurrence of an adverse event, the team has to avoid that the satellite switches to this mode, except if it is to prevent a satellite loss. When a controller detects an adverse event (usually a failure) and understands the issue, he/she has to apply a *procedure*, selected from a list of referenced *procedures*, to recover from that failure. If the failure is more complicated to understand, he/she has to inform the entire team (one or more satellite engineer and experienced controllers can collaborate) to solve the issue and select the adequate *procedure*. If such procedure is not available they might need to design a new one that in turn has to be entered in the ground segment and sent to the space segment). The operator often collaborate with other controllers or with dedicated engineers such as, for instance, Radio Frequency engineers when special operations on the communication link are required.

3.1.2 Task analysis of a satellite controller’s activities

Controllers are in charge of two main activities: observing periodically (i.e. **monitoring**) the vital parameters of the satellite and performing **maintenance operations** when a failure occurs. Depending on the satellite between a couple of thousands and tens of thousands parameters have to be monitored. The more frequent and relevant monitoring activities include observing: *satellite mode*, T*elemetry* (measures coming from the satellite), *Sun array drivers statuses*, *error parameters for the platform*, *error parameters for the mission*, *power voltage* (energy for the satellite), *ground station communication status*, and *on board computer main parameters*.

The number of *procedures* for maintenance operations goes beyond the hundred. Due to page limits, we only present a selection of three sub-routines concerning failure cases that are critical for the mission. However, these sub-routines allow us to exhibit all the problems that we faced and that were related to the structuring of task models. The sub-routines are:

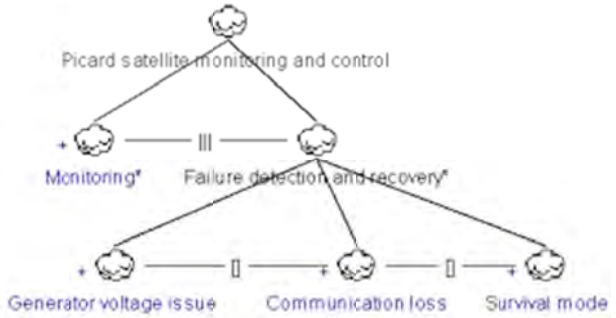
- **Recovering from a power voltage issue**: a wrong voltage parameter value is detected and the controller has to reset the satellite flight software.
- **Re-establishing the communication link**: the ground operation control system may not be receiving Telemetry from the satellite. The first activity of the controller is also to reset the flight software in this case. The other activities are not presented.
- **Investigating why an automatic switch to survival mode occurred**: Most satellites (including Picard) are not always all the time in ‘visibility’ of a ground station (only geostationary ones are). For such satellites the parameters are updated and the telecommands (TCs) sent when the satellite is visible for one of the ground station. Meanwhile, it evolves in an autonomous mode (self-triggering On Board Control Procedure (OBCP) if needed). During a non-visibility period if vital parameters’ values go beyond or below a given threshold, the satellite flight software (SW) switches itself to survival mode. In the next visibility period, controllers will have to understand what happened, find a solution and then send TCs to set the satellite back to its nominal mode.

Furthermore, after each failure detection and recovery sub-routine, the controller has to record the failure that happened in a dedicated application.

Lastly, the case study exhibits operators’ activities related to the management of the communication link between the Ground Segment and the Space Segment which also has to be monitored and possibly repaired.

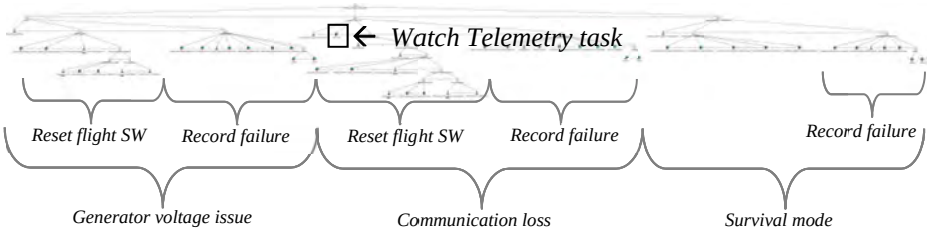
3.2 Task model using CTT

Our first attempt of modeling operator's tasks into sub-goals ended up with unreadable models due to their size. We present here one of the original models using CTT notation [27] (and CTTe tool [25]) as this notation is widely used in the HCI community. The point is not here to read the model but to show first that real-life tasks can be modeled and then that the resulting model can be too large to be handled in an efficient way. The tasks have been decomposed into several models, all presented in Fig. 2. Fig. 2a provides a high level view on the two main operators' tasks namely *Monitoring* and *Failure detection and recovery*. In this model we have used CTTe abstraction mechanism (represented in the model by the symbol + corresponding to abstracting away further details about these tasks). These two tasks are detailed in Fig. 2b and Fig. 2c, respectively. The subtask *Watch Telemetry* is highlighted to show where it appears in both models. This is a typical example of multiple appearance of the same task (which could be an entire sub-tree) in various models. Modifying information about it (name, duration, precondition, temporal operators …) is really cumbersome as these modifications have to be repeated manually. In addition, it is not possible to express in CTT the fact that these two sub-trees represent the same activity. Fig. 2c provides the decomposition for the tasks *Failure detection and recovery* including subtasks *Generator voltage issue*, *Communication loss* and *Survival mode*. It is important to note that these three tasks only represent a limited subset of the hundreds of procedures and related tasks needed to control the satellite, but despite that, the model is already large and complex. The model in Fig. 2c contains 74 tasks, for 3 types of failure (described in section 3.1.2), and then 3 types of procedures to set the satellite back to its nominal mode. For each type of failure, a special task tree is performed by the controller, who then additionally records the failure in a dedicated application. This is the reason why the "Failure detection and recovery" sub-goal has several parts in common (both "*Reset flight software*" which appears twice and "*Record failure*" which appears three times). By lack of structuring mechanisms the corresponding tasks are duplicated. Furthermore, most of the tasks in the operation of recording a failure are also duplicated, even though there are small differences related to the fact that they have to be customized according to the type of failure.



a) Top-view of controllers' tasks, including *Monitoring* and *Failure detection and recovery*

b) Decomposition of sub-task *Monitoring* highlighting the atomic task *Watch Telemetry* that also appear in other models (see figure below).



c) Decomposition of task *Failure detection and recovery* highlighting the reoccurrence of tasks *Reset flight SW* and *Record failure* into the hierarchy of subtask *Generator voltage issue*, *Communication loss* and *Survival mode*.

Fig. 2. Models of operators' task using CTT notation.

3.3 Issues raised by the legibility and scalability of the tasks models of the case study

The modeling of the case study using standard structuring mechanisms resulted in a complex and cumbersome activity.

One of the problems is that some tasks (in particular monitoring tasks) appeared several times (without specific temporal constraints) in the same model or even in several procedures. To model this kind of behavior it is required to duplicate monitoring tasks in various places of the task models.

Additionally; currently available structuring mechanisms do not support accurate representation of data flow. To describe complex conditions involving data and/or objects associated to tasks, the designer must combine hierarchies and operators between siblings' tasks in order to represent the expected behavior. As a consequence, the available solution leads to the multiplication of tasks in the model, thus making them illegible as demonstrated in Fig. 2. It is important to note that we have only modeled 3 types of failures, but at operation time in the Command and Control Centre, the number of possible failures is much higher. This demonstrated the fact that tasks notations have to be extended[3] in order to handle such problems if they are to be used for representing operators' activities in real-life complex systems. It is important to note that this aspect is critical for task modeling. Indeed, task modeling is precisely meant to be used for large scale systems. Indeed, small scale systems usually do not require the use of task models for understanding users' activities that are usually rather simple to describe.

4 HAMSTERS and its Extensions for Structuring Models

In order to demonstrate the use of the proposed structuring mechanisms for task models, we used a tool-supported notation called HAMSTERS (Human-centered Assessment and Modeling to Support Task Engineering for Resilient Systems). The rational for the choice on HAMSTERS, is that we have the possibility to extend both the notation and the tool (as the tool has been developed by us). However, as stated in section 2 we believe the mechanisms are generic and thus could be integrated in many other notations and tools.

4.1 Short introduction to HAMSTERS

HAMSTERS notation is inspired by existing notations, especially CTT [26] and has been designed to remain compatible with CTT (from the point of view of people building the

[3] it would have been possible to decompose the main model into sub-models within CTTe tool [25], but simulation would have been impossible

models) as models are hierarchical and graphically represented featuring operators between the tasks. However, HAMSTERS includes extensions such as pre-conditions associated to task executions, data flow across task models, more detailed interactive tasks… HAMSTERS models can be edited and simulated in a dedicated environment which also provides a dedicated API for observing editing and simulation events making it possible to connect task models to system models [2] and [19] based on the Interactive Cooperative Objects formalism [20] and [21]. The notation presented hereafter provides additional extensions that are required for structuring models. Table 1 illustrates HAMSERTS' constructs, including:

- Abstract task is a task that involves sub-tasks of different types.
- System task is a task performed only by the system.
- User task is a generic task describing a user activity. It can be specialized as a Motor task (e.g. a physical activity), a Cognitive task (e.g. decision making, analysis), or Perceptive task (e.g. perception of alert).
- Interactive task represents an interaction between the User and the System; it can be refined into Input task when the users provide input to the system, Output task when the system provides an output to the user and InputOutput task which is a mix of both but performed in an atomic way.

Task type	Icons in HAMSTERS task model
Abstract Task	
System Task	
User Tasks	
Interactive Tasks	

Table 1. Tasks types in HAMSTERS

As in CTT, each particular task of the model can be either iterative, optional or both (see Fig. 3). *Iterative* property of tasks means that a task that can be executed 1 or several times; it can be interrupted or suspended by another task. It should not be a subtask of an ENABLE operator but of an INTERRUPT or SUSPEND_RESUME operator. *Optional* tasks do not require to be executed for the goal to be reached.

Fig. 3. Optional task Iterative task and of a task both iterative and optional

Additionally, (as in CTT again) temporal relationship between tasks is represented by means of operators as described in [26]. In HAMSTERS, the notion of objects can represent both information and knowledge needed for performing a task. This information might be modified when the task is performed. HAMSTERS offers two types of relationships between tasks: the first one describes how tasks are related to other tasks (using the temporal operators as presented above) and the second one represents the information flow between tasks (as illustrated in Fig. 4 where the PIN entered in the first task is conveyed to the next task by means of input and output ports). According to the discussion in the introduction about tasks notations it could be argued that adding

information to behavioral models such as task models will degrade legibility for instance. However, as such information has an impact on the availability of tasks and the action they perform they must be exhibited in order to avoid under-specified tasks models.

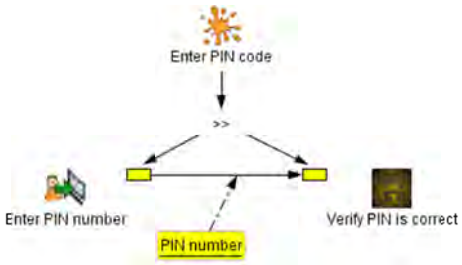


Fig. 4. Input (right-hand side of a task) and output (left-hand side of a task) flow in HAMSTERS

4.2 Structuring mechanisms in HAMSTERS notation

In order to address the scalability problems introduced in section 3, (as stated in section 2) we have identified 2 mechanisms: composition and abstraction/refinement. This section details how these mechanisms are integrated in HAMSTERS: composition is by means of sub-routines and the abstraction/refinement by means of sub-models. More precisely, we have added four types of sub-routines (see Table 2), parameters handling for sub-routines (see Table 3) and copy of tasks (or tasks sub-trees) (Table 4).

4.2.1. Composition mechanism: sub-routines

A sub-routine is a group of activities that a user performs several times possibly in different contexts which might exhibit different types of information flows. A sub-routine contains:

- The name of the sub-routine;
- The icon of an "Abstract" task type (as the sub-routine consists of a group of tasks that can belong to different types);
- Specialized input and output ports attached both to the left and to the right of the icon. The graphical symbol of these specialized ports can be filled (if they handle parameters) or not (if they don't). These ports are mechanisms for representing required parameters to and/or from sub-routines, thus providing explicit representation of data flow during task execution.

Structuring mechanisms	Related icon in HAMSTERS
Sub-routine with no input value and no output value	Reset Flight SW
Sub-routine with at least one input value and no output value	Record failure
Sub-routine with no input value and at least one output value	Find root cause of the switch
Sub routine with at least one input value and at least one output value	Operate communication frequency

Table 2. Graphical representation of sub-routines in HAMSTERS

The sub-routine is then modeled in a dedicated model where the root task is the icon of the sub-routine. As stated above, the behavior of the sub-routine can be different according to the value of the parameters (see Table 3). In such a case the precondition in the sub-routine is represented by the symbol . In the main task model, the same symbol can be used by a task for changing the behavior of the model according to the value of the output parameter of the sub-routine. The value of the output parameter (if any) has to be assigned inside the sub-routine. This assignment is represented by the symbol . Such representation makes it very easy to identify in the sub-routine both the number of output parameters and where in the model those parameters are modified.

New artefact	Icon in HAMSTERS
Condition on a parameter	
Assignment of a value to an output parameter (of a sub-routine)	

Table 3. Input and output parameters management

4.2.2. Abstraction/refinement mechanism: sub-models

Table 4 presents the task icon used to indicate that the task depicted by this icon is a copy of another task in the model. There can be several copies for a given task and all these copies correspond to the same sub-model. A modification in the sub-model (including its name) is thus replicated in all the sub-models. Table 4 only represents the copy icon for the "Output" task type but one "Copy" kind of icon is available for all the tasks types (see Table 1).

Abstraction/refinement	Icon in HAMSTERS
Sub-model (copy) of an existing Output task	Watch Telemetry

Table 4. Illustration of sub-models in HAMSTERS (only for an output task)

5. Case study modeling using the proposed structuring mechanisms

Fig. 5 shows the main task model for the main operator goal which is the monitoring and control of the Picard satellite. This main goal can be subdivided into 2 sub-routines: Satellite monitoring (highlighted by the black disc n°1) and Failure detection and recovery (highlighted by the black disc n°2). These 2 sub-routines are 2 sub-goals that can be achieved concurrently (||| temporal operator) and their detailed structures are presented respectively in Fig. 6 a) and Fig. 6 b).

Fig. 5. PICARD satellite monitoring and control main task model with HAMSTERS

5.1 Sub-routine with empty input and output parameters

In Fig. 6 a), the sub-routine corresponding to the sub-goal "Satellite monitoring" (part 1 of Fig. 5) is decomposed into 8 output tasks that are corresponding to the monitoring activities detailed in section 3.1.2. Fig. 6 b) details the "Failure Detection and Recovery" sub-routine (part 2 of Fig. 5) and is decomposed into a choice ([] temporal operator) of 3 sub-goals, which are depending on the type of failure discovered by the controller.

The first failure type corresponds to the power voltage issue, and is detected by the controller if he/she understands (cognitive task) that the power voltage parameter is wrong. This is done by observing or just having observed (while monitoring, Fig. 6 a)) this voltage parameter. His/her activities will then consist in resetting the satellite flight software and then recording this failure in the dedicated application. Second failure type (Fig. 6 b)) is a communication link loss and is detected by the controller when he/she understands that Telemetry from the satellite is not available anymore. His/her activities will then consist in resetting the transmission link. Third and fourth failure types (last task at the second level of Fig. 6 b)) detail the activities that are performed by the controller when he/she detects that the satellite has automatically switched to survival mode.

The second and third failure types may require the controller to perform a "Reset flight SW" *procedure*, which has been modeled (disc 4 on Fig. 6 b) as a sub-routine with empty input and output parameters. This sub-routine is very useful as it avoids duplicating an entire sub-task model. The sub-routine "Reset flight SW procedure" is detailed in Fig. 7 b).

5.2 Sub-routine with at least one input parameter and empty output parameter

The "Record failure" sub-routine (disc 3 on Fig. 6 b detailed in Fig. 7 a) is performed by the controller each time a failure has been detected and aims at recording information in a dedicated desktop application about problems encountered in operations. An input value is required, because one or more of the group of tasks that are composing the sub-routine need information to be executed. As shown on Fig. 7 a), the record of a failure can slightly differ from one failure to another, and the input parameter of the sub-routine will also help in modeling the differences into sub-task trees (using the condition parameter introduced in Table 3).

5.3 Sub-routine with empty input parameter and at least one output parameter

In case of an automatic switch of the satellite to the survival mode (third failure type on Fig. 6 b), the controller will perform a *procedure* to find out the root cause of this issue. It is modeled by the "Find root cause of the switch" sub-routine, and this sub-routine provides an output object that will be used by the next sub-routine "Record failure (out)" to record the failure type that has been found.

5.4 Sub-routine with both input and output parameters

In case of a "Reset flight SW" sub-routine (disc 4 on Fig. 6 b detailed in Fig. 7 a), the controller has to prepare the RF communication link. The "Operate Com frequency" sub-routine requires an input and provides an output. It indicates that the information produced while preparing the RF is then used by the controller to reach his/her goal.

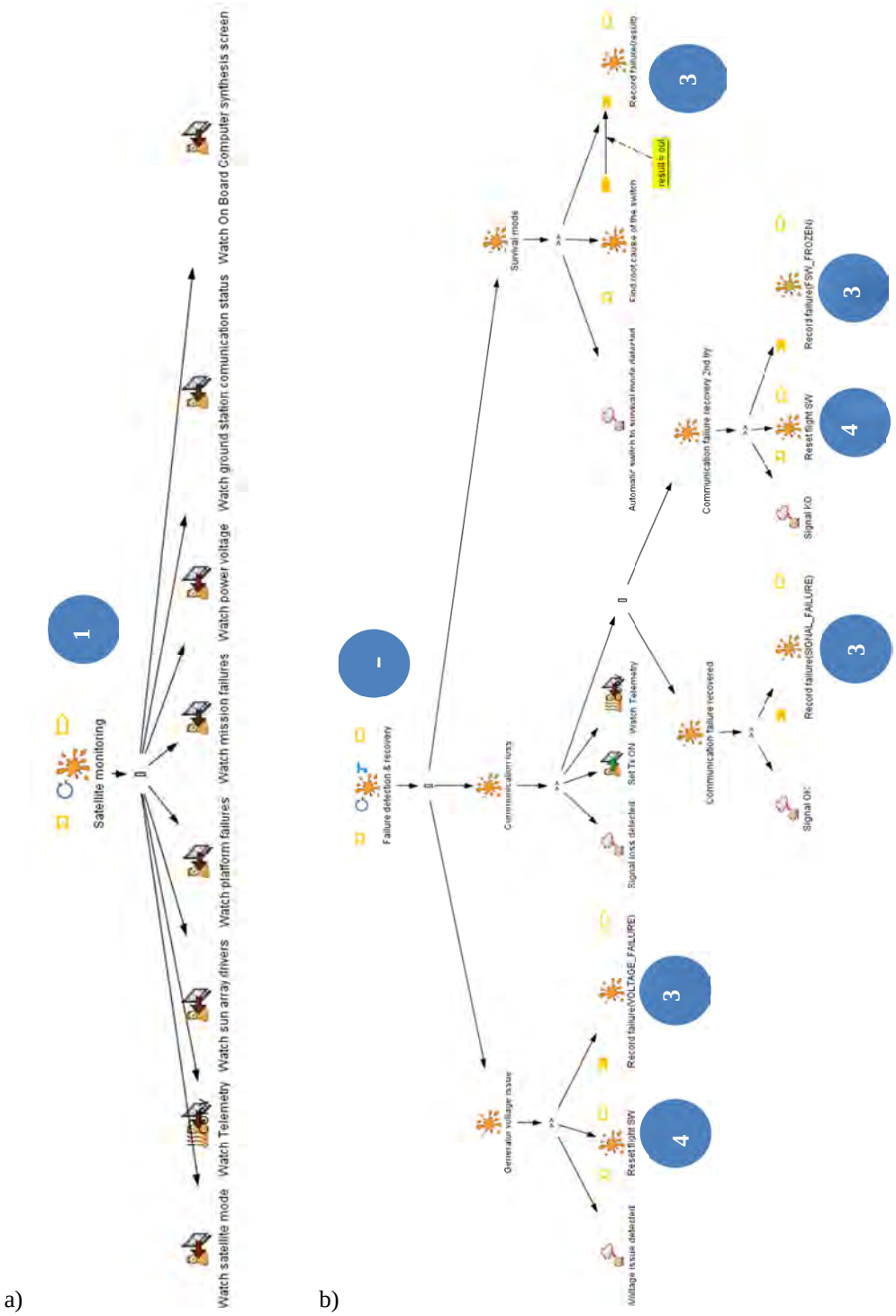
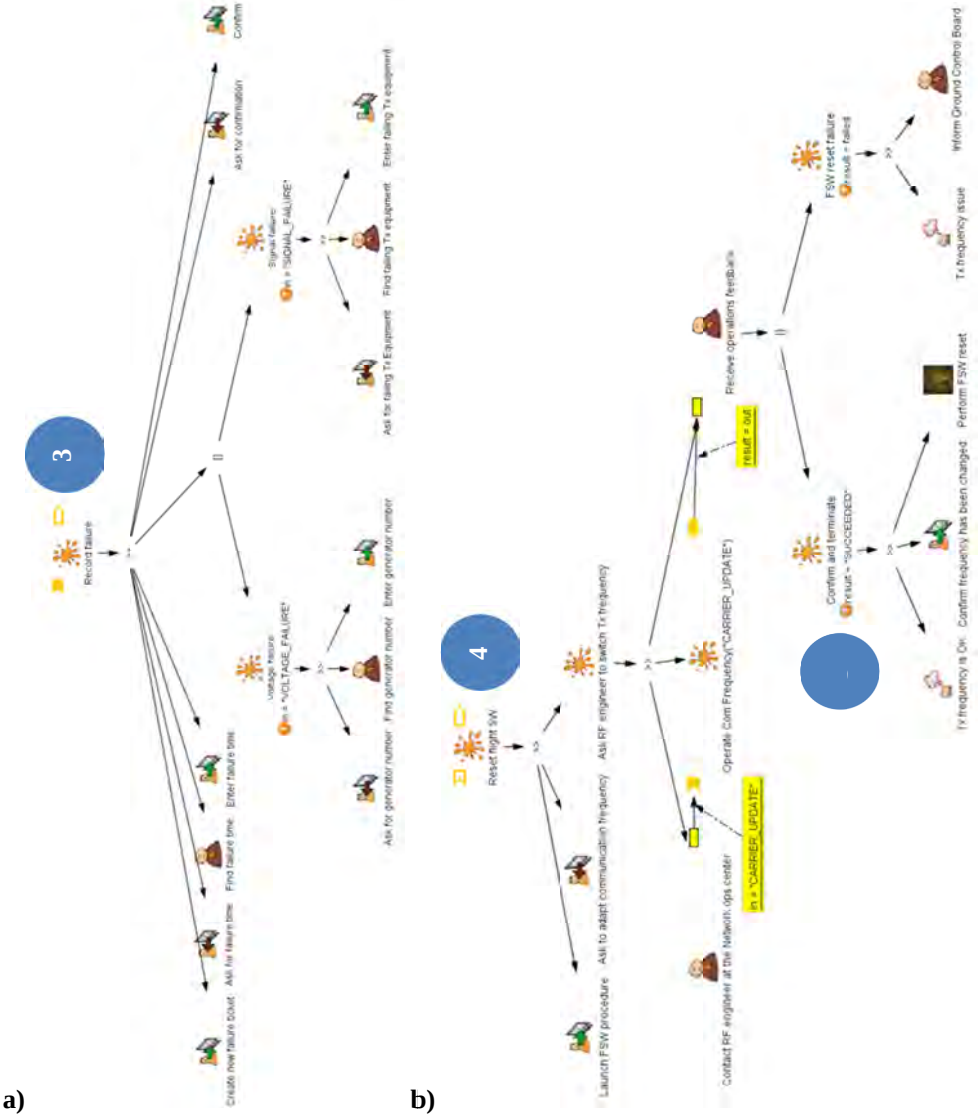


Fig. 6. Tasks of satellite Monitoring (a) and, failure detection and recovery (b).

5.5. Condition on a parameter

Fig. 7 a) shows how the conditions on the input object can be used to customize the Record failure sub-routine. Indeed, in that model, there is a common part of the model dedicated to the recording of the failures but also specific parts which vary according to the failure types. In this model, if the failure type is a voltage issue ("VOLTAGE_FAILURE" is the input value of the sub-routine), the controller is asked to enter the power generator number in the application; if the failure type is a signal failure ("SIGNAL_FAILURE" is the input value of the sub-routine), the controller is asked to enter the failing transmission equipment. In both cases, failure time information and confirmation are required. In Fig. 7 a) this alternative is represented by the choice operator and the fact that each choice uses a precondition on the failure type in=SIGNAL_FAILURE for instance.



a) b)

Fig. 7. a) Record failure sub-routine b) and Reset flight SW (software) sub-routine

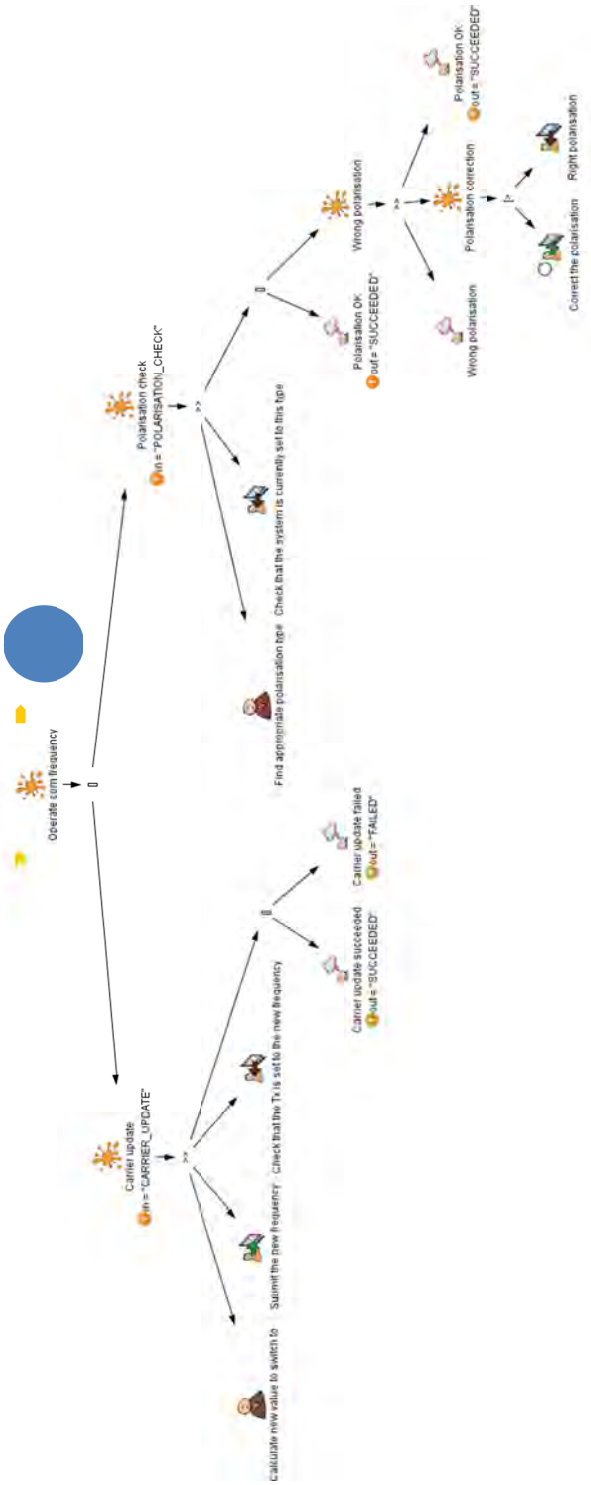


Fig. 8. Operate com (communication) frequency sub-routine

5.6 Assignment of a value to an output parameter

Fig. 8 details the "Operate Com Frequency" sub-routine. It gives an example on how the assignment of a value to an output parameter (exclamation mark icon) is used to support storing of history information about what happened when the sub-routine has been performed. In Fig. 8 the assignment out=SUCCEEDED records in the out parameter the outcome of the sub-routine which can, in turn, be used as condition in the main task model. Besides, in each sub-task group, the output parameter is assigned which guarantees that the sub-routine always delivers an updated output parameter. The "out" parameter is assigned the value "SUCCEEDED" when the operation on the communication frequency succeeded, and it is assigned the value "FAILED" when the operation on the communication frequency failed.

5.7 Sub-model of an existing task (copy)

On Fig. 6 b), in the second failure case "Communication loss", the controller has to send a command to set the transmission to ON (input task which corresponds to resetting the transmission) and look at the Telemetry status. This activity of observing the Telemetry status is already described in the Fig. 6 a) sub-routine as it is part of the "Monitoring" job of the controller. The artifact "Copy" of an existing task is here used to maintain consistency between the models and avoiding duplication (copy of the "Watch at Telemetry" task of the Monitoring task model described in Fig. 6 a)). This allows the reader of the model to understand that this user's activity is already referenced in another part of the task model. It is important to note that this sub-model structuring can be applied to any node in the task model. If the node is not a leave task, then the entire sub-tree is managed as a copy i.e. modification in one is reflected in all the copies. This issue has been raised in section 3.2 when the CTT model has been presented.

6. Analysis of the Case Study

6.1 Quantitative analysis

Table 4 presents statistics on the number of model elements needed to describe the controller's activities, using both CTT and HAMSTERS notations (together with the structuring mechanisms proposed). It is important to note that part of the difference between the two representations lies for some aspects in the structuring mechanisms and for other aspects in the notations themselves. However, adding those mechanisms to CTT or to any other notation dedicated to task models would exhibit similar advantages.

The first line of the table (in italics) highlights the fact that the number of models is much higher using the mechanisms than without using them. Then, for each model, the number of nodes, arcs and operators is presented. This is due to the sub-routine extension that promotes splitting a model into several parts. The sub-model extension has no impact on the number of models.

The total number of tasks required to model the controller's activities is decreased by about 20% using HAMSTERS notation together with the structuring mechanisms. There is a bigger difference (66% reduction) as far as both the operators and the arcs are concerned.

This is due to the fact that operators are considered as part of the node in HAMSTERS and don't need to be duplicated if the temporal operator between several tasks in a row is the same (as this is the case in CTT notation). The light grey lines in the lower part of the table correspond only to models appearing in the HAMSTERS modeling. They have no counterpart in CTT.

	Notations	
Number of elements used for the case study according to the notations:	**CTT**	**HAMSTERS**
Number of models	*4*	*6*
Tasks for representing the main model	6	3
Operators for representing the main model	3	1
Arcs for representing the main model	11	3
Tasks for representing the "Monitoring" task model	9	9
Operators for representing the "Monitoring" task model	7	1
Arcs for representing the "Monitoring" task model	22	9
Tasks for representing "Failure detection and recovery"	74	20
Operators for representing "Failure detection and recovery"	52	7
Arcs for representing "Failure detection and recovery"	175	27
Tasks for "Operate communication frequency"	19	17
Operators for "Operate communication frequency"	11	7
Arcs for "Operate communication frequency"	40	23
Tasks for representing the "Record failure" task model	-	15
Operators for representing the "Record failure" task model	-	4
Arcs for representing the "Record failure" task model	-	18
Tasks for representing the "Reset flight SW" task model	-	14
Operators for representing the "Reset flight SW" task model	-	5
Arcs for representing the "Reset flight SW" task model	-	20
Total number of tasks	**102**	**78**
Total number of operators	**73**	**25**
Total number of arcs	**248**	**100**

Table 5. Quantitative comparison of the two modeling techniques

6.2 Qualitative analysis

Qualitative assessment is a more complex task which would require very detailed usability analysis of both the use of the structuring mechanisms and the idiosyncrasies of each notation. This is premature to the work presented here as it would require the distribution of HAMSTERS at a large scale (as this has been done for a long time with CTT) and then the setting up of an ethnographic study. However, even though such experiments were conducted we would face the same difficulties as those encountered in the late 80's when the software engineering community was struggling to assess whether Object-Oriented programming was "better or not" with respect to "structured programming" [37]. Indeed, there are so many parameters involved such as training, maturity of tools, application domain, background of the developers … that a definite result is out of reach. Another related element is the user interface and interaction technique embedded in the tool supporting the notation. Indeed, even CTTE proposes the CTTVis extensions providing zooming and enhanced interaction techniques for task models edition. Such aspects have a he impact on the adoption and performance of user while building task models.
However, the current version of HAMSTERS tool is publicly available at http://www.irit.fr/ICS/hamsters/ and we plan to gather feedback from the future users.

7. Discussion and Conclusions

This paper has presented two new structuring mechanisms for notations dedicated to task modeling. We have shown how these mechanisms are different from the mechanisms currently available in the various existing notations. These mechanisms have been applied on a large case study from the space domain. We have used CTT notation as a reference point to support the reader who might be familiar with this notation and to compare quantitatively the size of the resulting models.

While this paper has focused on the mechanisms and the notation themselves this work belong to a long research program aiming at exploiting in a systematic way models in the design and validation of large (potentially safety critical) interactive systems. Indeed, these mechanisms would be extremely useful for addressing the issue of user errors in tasks models (in order, for instance, to identify scenarios exhibiting user errors) as in [24] or to support training of operators as in [18]. In the context of safety critical systems such elements are of primary importance as the resilience of the entire socio-technical system (including operators, procedures, training and computing system) has to be assessed prior to deployment.

References

1. Amboss,http://wwwcs.uni-paderborn.de/cs/ag-szwillus/lehre/ws05_06/PG/PGAMBOSS.
2. Barboni E., Ladry J-F., Navarre D., Palanque P. & Winckler M. 2010. Beyond modeling: an integrated environment supporting co-execution of tasks and systems models. Proc. of the 2nd ACM SIGCHI symp. on Engineering interactive computing systems (EICS '10), 165-174.
3. Breedvelt, I., Paterno F., Sereriins, C.: Reusable structures in task models. In Harrison, M.D., Torres, J.C., eds.: DSVIS'97, Springer-verlag (1997) 251–265.
4. Card, S., Moran, T., Newell, A. The Psychology of Human-Computer Interaction, Lawrence Erlbaum, Hillsdale, 1983.
5. Chen, P., The Entity-Relationship Model - Toward a Unified View of Data, ACM Transactions on Database Systems, Vol. 1, No. 1, 1976, pp. 9-36.
6. Chattratichart, J., Kuljis, J. (2002) Exploring the effect of control-flow and traversal direction on VPL usability for novices. Journal of Visual Languages and Computing. v13. 471-500.
7. Dennis, A., & Valacich, J., Computer Brainstorms: More Heads are Better than One. Journal of Applied Psychology 78, 4 (1993), 531--537.
8. Diaper, D. & Stanton, N. A. (eds.) The Handbook of Task Analysis for Human-Computer Interaction. Lawrence Erlbaum Associates, 2004. 650 pages.
9. Diehl, M., & Stroebe, W. Productivity Loss in Brainstorming Groups: Towards a Solution of a Riddle, Journal of Personality and Social Psychology 53, 3 (1987), 497-- 509.
10. Dittmar, A. 2000. More precise descriptions of temporal relations within task models. Design, Specification, and Verification of Interactive Systems (DSV-IS'00). Springer-Verlag, 151-168.
11. European Cooperation for Space Standardization, Space Engineering, Ground Systems and Operations, ECSS-E-70C, 31 July 2008.
12. Gaffar, A., Sinnig, D., Seffah, A., Forbrig, P. 2004. Modeling patterns for task models. In Proceedings of the 3rd annual conference on Task models and diagrams (TAMODIA '04). ACM, New York, NY, USA, 99-104.
13. Hartson, R.H. and Gray, P.D. Temporal aspects of tasks in the User Action Notation in Human Computer Interaction. Lawrence Erlbaum Associates, 1992. Vol. 7, pp. 1-45.
14. Hopcroft J. & Ullman J. Introduction To Automata Theory, Languages, And Computation, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, 1990.

15. Johnson, H. & Johnson, P. (1991) Task Knowledge Structures: Psychological basis and integration into system design. Acta Psychologica, 78 pp 3-26.
16. Limbourg, Q., Pribeanu, C., Vanderdonckt, J. 2001. Towards Uniformed Task Models in a Model-Based Approach. 8th Workshop on Interactive Systems: Design, Specification, and Verification- (DSV-IS '01), Chris Johnson (Ed.). Springer-Verlag, London, UK, 164-182;
17. Mavin A. & Maiden N.A.M., (2003), 'Determining Socio-Technical Systems Requirements: Experiences with Generating and Walking Through Scenarios', Proc. 11th Int. Conf. on Requirements Engineering, IEEE Computer Society Press, 213-222.
18. Martinie C., Palanque P., Navarre D., Winckler M. & Poupart E. Model-Based Training: An Approach Supporting Operability of Critical Interactive Systems: Application to Satellite Ground Segments, Proceedings of ACM SIGCHI EICS 2011, Pisa, ACM Press.
19. Navarre D., Palanque P., Bastide R., Paternó F. & Santoro C. A tool suite for integrating task and system models through scenarios. In 8th Eurographics workshop on Design, Specification and Verification of Interactive Systems, DSV-IS'2001; LNCS no. 2220. Springer Verlag 2001.
20. Navarre, D., Palanque, P., Ladry, J., and Barboni, E. 2009. ICOs: A model-based user interface description technique dedicated to interactive systems addressing usability, reliability and scalability. ACM TOCHI 16,4 (Nov. 2009), pp. 1-56.
21. Navarre D., Palanque P. & Bastide R. (2003). A Tool-Supported Design Framework for Safety Critical Interactive Systems in Interacting with computers, Elsevier, Vol. 15/3, pp 309-328.
22. Norman, D. & Drapper, S. (Eds.). (1986) User Centred System Design. L. Erlbaum, U.S., ISBN-10: 0898597811
23. Petri, C. A. "Kommunikation Mit Automaten." Technical University Darmstadt (1962).
24. Palanque, P., Basnyat, S.: Task Patterns For Taking Into Account In An Efficient and Systematic Way Both Standard And Erroneous User Behaviours. IFIP 13.5 Working Conf. on Human Error, Safety and Systems Development (HESSD), Kluwer Academic Publisher (2004), pp. 109–130.
25. Paternò F. (2001). CTTE: An Environment for Analysis and Development of Task Models of Cooperative Applications. ACM CHI 01 (Seattle, 2001) Extended Abstracts, ACM Press.
26. Paternò F. (2003) ConcurTaskTrees: An Engineered Notation for Task Model. The Handbook of Task Analysis for Human-Computer Interaction. pp. 483-503. Lawrence Erlbaum Associates.
27. Paternò, F., Mancini, C. and Meniconi, S. ConcurTaskTrees: A Diagrammatic Notation for Specifying Task Models. In: Proc. of Interact'97. Chapman & Hall (1997), 362-369.
28. Paternò, F., Zini, E. 2004. Applying information visualization techniques to visual representations of task models. Proc. of TAMODIA '04. ACM, New York, NY, USA, 105-111.
29. Rational Software Corporation. UML Notation Guide. 1.1 ed.1997.
30. Rajeev Alur. 1999. Timed Automata. In Proceedings of the 11th International Conference on Computer Aided Verification (CAV '99), Springer-Verlag, London, UK, 8-22.
31. Rettig M. "Prototyping for Tiny Fingers." Communication of the ACM 37, no. 4 (1994) 21-27
32. Sebillotte, S. Hierarchical planning as method for task analysis: the example of office task analysis. Behaviour & Information Technology, 1362-3001, Vol.7, n°3, 1988, Pages 275-293.
33. Scapin, D. L. K-MADe. In: COST294-MAUSE 3rd International Workshop, Review, Report and Refine Usability Evaluation Methods (R3 UEMs), Athens (March 5, 2007).
34. Sinnig, D., Wurdel, M., Forbrig, P., Chalin, P., Khendek, F.: Practical Extensions for Task Models. TAMODIA 2007. LNCS, vol. 4849, pp. 42–55. Springer, Heidelberg (2007)
35. van Welie, M., van der Veer, G. C., Eliëns, A. Euterpe - Tool support for analyzing cooperative environments. Ninth European Conf. on Cognitive Ergonomics ,pp. 25-30, August 24-26, 1998.
36. van Welie, M., van der Veer, G; C., Eliëns, A. An Ontology for Task World Models: In: 5th International Eurographics Workshop on Design Specification and Verification of Interactive Systems DSV-IS98 ,pp. 57-70, 3-5 June 1998, Abingdon, UK.
37. Wieringa R. A survey of structured and object-oriented software specification methods and techniques. ACM Comput. Surv. 30, 4 (December 1998), 459-527.
38. Wood W.A. Transition network grammars for natural language analysis. Communications of the ACM 13, 10 (October 1970), 591-606

StateWebCharts: a Formal Description Technique Dedicated to Navigation Modelling of Web Applications

Marco Winckler & Philippe Palanque

LIIHS – IRIT, University of Toulouse III
118, route de Narbonne
31062 Toulouse Cedex 4 France
{winckler, palanque}@irit.fr

Abstract. This paper presents StateWebCharts (SWC), a formal description technique based on statecharts for describing navigation on web applications. This notation extends the classical statecharts notation by adding more necessary concepts such as an appropriate semantics for states and transitions in a Web context, including notions like dialog initiative control and client and server activities. As well as statecharts do, this formal description technique features a graphical representation thus making it easier to use for web designers and formal enough to allow to rigorously reason about properties of navigation models. In order to show the applicability of the notation, we show, in the paper, its use on two real-size web applications.

1. Introduction

"Web applications[1]" is a widely-used and fuzzy term for web sites including informational-centric sites, e-commerce sites, portal sites, etc. Despite the apparent facility to create web pages (HTML pages) the successful development of large web applications is a complex activity that requires appropriate methods and tools [13]. This inherent complexity is not only due to the huge number of pages that must be managed or the diversity of technologies employed (JavaScript, Java, Active-X, etc) but also to dynamic aspects such as on-the-fly page generation. In addition, web applications require regular maintenance in order to update pages content, to follow a particular business workflow, to include new features for supporting new task and/or users, and so on.

To deal with such complex development of web applications, modelling support is essential to provide an abstract view of the application. Modelling can help designers during design phases by defining formally the requirements, providing multi-level of details as well as providing support for testing prior implementation. Support from modelling can also be obtained in later phases via, for instance, support for verification prior to implementation [1].

[1] Some authors [3, 6] call *web applications* only data intensive application which presents dynamic content generation and the term *websites* is applied only for applications based on static content. This distinction is not relevant in this paper and thus, these terms are used as synonyms here.

Statecharts [8, 10] and statecharts-like notations have already been widely used for modelling various aspects of web applications. For instance, they have been previously used for navigation modelling of hypertext/hypermedia applications [14, 18, 20], web applications [12] and even WIMP interfaces [10]. This previous work shows some limitations in the expressive power of statecharts for handling specific and critical aspects of web applications' modelling. For instance, it is not possible, using statecharts, to represent who (the user or the system) is at the source of an event to be received by the application. Even previous works that focused on navigation modelling for web applications, such as [14], do not clearly explain how statecharts can be effectively used to model other web applications features such as dynamic content generation.

For that reason, we have extended statecharts to the StateWebCharts notation (SWC) that provides dedicated constructs for modelling specificities of states and transitions in web applications. Our aim is to provide a visual notation easy-to-apply for web designers and formal enough to be subject of automatic verification, thus supporting designer's activity throughout the design process. Most elements included in SWC notation aim at providing explicit feedback about the interaction between users and the system.

As for now, SWC is mainly used to describe the navigation between documents rather than interaction between objects. We distinguish navigation (communication links between information units) from interaction (e.g. manipulation of interface widgets such as scrollbars, and windows interactors). SWC is powerful enough for handling these two aspects, but such concerns are beyond the scope of this paper.

This paper aims at presenting SWC notation in detail and at showing how this notation can be used for modelling navigation in Web applications. Next section (section 2) presents a formal definition of statecharts. This formal definition is used as a basis for the introduction of the extensions at the core of SWC (section 3). Section 4 presents an exhaustive list of the various key elements of web navigation and for each of these elements how SWC can be used for modelling them. In section 5, this paper brings a detailed discussion about the related work on navigation modelling including statecharts-like notation as well as other approaches for navigation modelling. Conclusion and future work are presented in section 6.

2. The Statecharts Notation

Statecharts [8, 9] is a visual formalism that extends state diagrams for modelling complex/reactive systems. Statecharts can be defined as a set of the *states, transitions, events, conditions* and *variables* and their inter-relations. There are numerous extensions to Statecharts to support different modelling needs and with different semantics [9]. Hereafter we introduce the basics of statecharts that are relevant for this paper.

A formal definition of statecharts (also called state machine) is [8]:

S is defined as the set of states;

$\rho : S \rightarrow 2^S$, is the map function that associates each state to its sub-states, where **ρ(s)=0** means *s* is a *basic state* with no children inside;

$\psi : S \rightarrow$ **{AND / XOR}** is the function that defines whether $s \in S$ is a *composed AND / XOR* state or not

H is the set of *history symbols*

$\gamma : H \rightarrow S$ is the function that match *history symbols* to *states*[2]

$\delta : S \rightarrow 2^{S \cup H}$ is the *default function* that defines the initial states in *S*

Φ is the set of *final state symbols*

V is the set of variables

C is the set of *conditions*

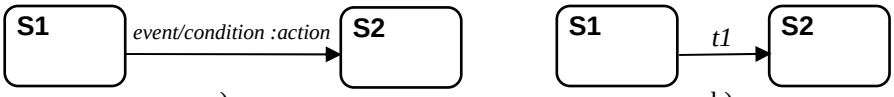
E is the set of *events*

A is the set of *actions*, where each action is a term of a language $\mathbf{£_a}$, which defines the allowed operations in a SWC machine

L = E x C x A is the set of *labels* on transitions

$T \subset 2^S \times L \times 2^{S \cup H \cup \Phi}$, is the set of transitions represented by a source state (2^S), a label (L) and a target state ($2^{S \cup H \cup \Phi}$)

States are graphically represented by rounded rectangles and transitions are represented by unidirectional arrows going out from a source state to a target state (see Figure 1). Transitions are usually represented by arrows that are labelled by the expression *event/condition:action* (see figure 1a). Optionally, the label could be just a generic identification (*t1*, as in Figure 1b). Guard conditions and actions are optional. If the guard condition is not given it is assumed to be *true*. When actions are not given, control is given to the arrival state. Parameters can be passed-on from a state to another. Only events are explicitly required on transitions.



a) b)

Figure 1. Graphical representation of states and transitions

By opposition with state diagrams that are "flat" and inherently sequential in nature, statecharts propose three basic structuring mechanisms: hierarchy of states, orthogonality and broadcasting communication. The first two ones, that are critical for web applications navigation modelling, are presented more in detail hereafter.

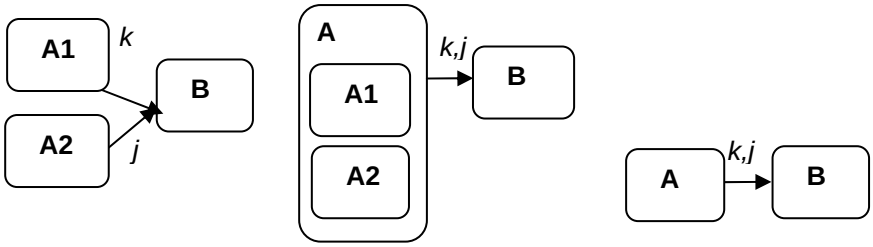
2.1. Hierarchy

The hierarchy is represented by composition of nested states (XOR-states) thus allowing an efficient use of transition arrows. XOR-states can have exclusively one active sub-state at a time. Figures 2a, 2b and 2c are equivalent representations

- On Figure 2b, states *A1* and *A2* are nested into the XOR-state *A*. All transitions going out from a composite state are propagated to sub-states;

[2] The difference between *History states* types (*shallow* and *deep* history) and *end-states* is not relevant for this paper.

- Figure 2c hides details for the XOR-state *A*. This is a useful abstraction mechanism in statecharts.



a) b) c)

Figure 2. Hierarchy of states in statecharts with XOR-states.

2.2. Orthogonality

Orthogonality is the decomposition of composite states into concurrent regions representing independent modules in a system. Each concurrent region in an AND-state is delimited by a dashed row. Figure 3 shows 3 concurrent states: D, C and A. Like a XOR-state, each concurrent region can have at most one active state at a time.

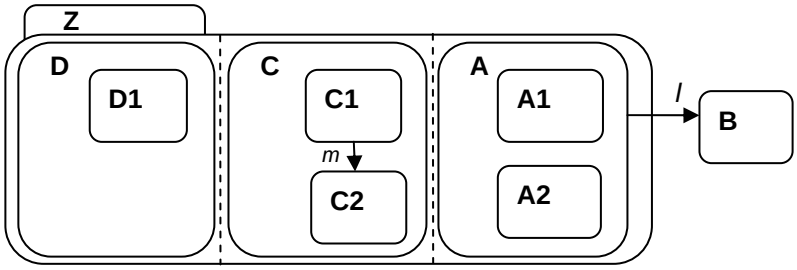


Figure 3. Concurrent states in statecharts with AND-states.

Like a state diagram, a statecharts model starts in an initial state represented by an arrow with a black circle at its starting end (see figure 4). It is also possible to define the initial state in a XOR-state, as shown by figure 4a (state A1). In figure 4a, the execution starts by state *B*. If transition *p* is activated, the system enters the state A1 (the initial state in the composite state *A*). Figure 4b uses a history state, which is represented by an *H* inside a circle. The history state sets the active state to the most recent state in the set (*A1* or *A2*).

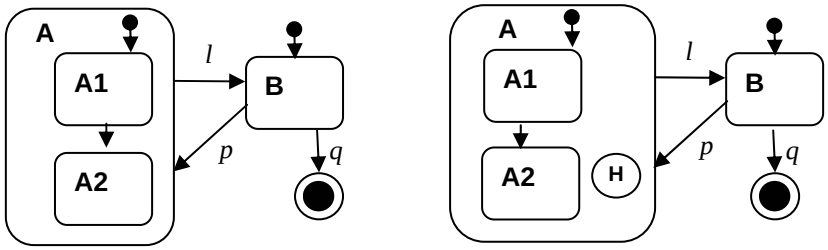


Figure 4. Initial states, history states and final states in statecharts.

Transition *q* in both Figure 4a and Figure 4b is going from state *B* to a *final state* which is represented by a black dot inside a circle. *Final states* mean that the execution is completed. When the enclosing state is on the top state, then it means that the entire state machine has completed.

The operational semantics for statecharts is given by a sequence of steps. At each step the state machine evaluates a single transition and may assume a new state configuration (the set of currently active states). When an event happens, the system transmits it to the transition associated to the triggered event. Then, the corresponding guard condition is evaluated, and if it is true the statemachine sets the target state as active.

An optional activity can be added to the transition label, indicating which activity will take place when the transition happens. The triggered activity can in turn be received by the system as another event to trigger other transitions creating compounding transitions. The broadcasting mechanism in statecharts is represented by events that are associated to more than one transition. In that case, when an event happens, all transitions associated to the triggered event are evaluated and executed if the guarding conditions are true. In classical statecharts, activities and events are considered to be instantaneous (they take no time to perform).

3. StateWebCharts Formal Description

In SWC, states are abstractions of containers for objects (graphic or executable objects). For web applications such containers are usually HTML pages. States in SWC are represented according their function in the modelling. In a similar way, a SWC transition explicitly represents the agent activating it. The basis of SWC modelling is a state machine, as described in previous section, plus the following elements:

P is the set of containers storing information content that will be exhibited as a whole in a web application (generally web pages). **P** is defined by the set (**o**, **k**) where **o** is the set of objects (text, sound, image, etc) it contains and **k** is the set of anchors in the set. The set **P** include an empty container.

$\mathbf{\Omega : S \rightarrow P}$, is the map function that associates each state to a container

$\mathbf{M : k \rightarrow E}$, is map function that associates a anchor to an event in the state machine

$\mathbf{\Sigma : S \rightarrow AC}$ is the mapping function that associates a state to its activities

AC is the set of optional actions associated to a state. $\mathbf{AC=AC_{entry} \cup AC_{do} \cup AC_{exit}}$, where $\forall$ ac $\in$ A, AC_{entry} is the action executed when entering a state, AC_{do} is the main action executed by it, AC_{exit} is the action executed before the state is left.

Y = {static/transient/dynamic/external} is the set of sub-types for a basic state

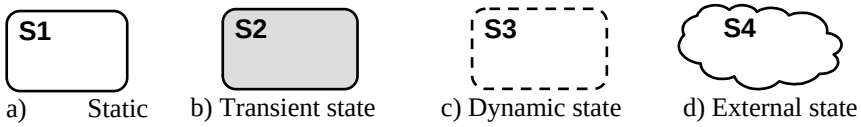
$\mathbf{\varpi : S \rightarrow Y}$ is the function that maps a sub-type to a basic state in the state machine $\forall$ s $\in$ S **.** ρ(s) = 0, $\varpi \neq \varnothing$, $\exists$ y $\in$ Y, ϖ (s)=y

W = {user/system/completion} is the set of events sub-types where each event type indicates an agent triggering the event in the system

$\mathbf{E = W_{user} \cup W_{system} \cup W_{completion}}$ is the redefined set of event in a system

A container *P* is considered as a compound document according to W3C DOM[3] definition, which may contain objects (text, images, etc) as well as other documents.

By the function $\varpi : S \rightarrow Y$ we make each basic state $s \in S$ assume an appropriate sub-type *static*, *transient*, *dynamic* or *external*. Each sub-type describes a special function performed by the state in the SWC state machine. Figure 4 shows the graphic representation of these sub-types.



a) Static b) Transient state c) Dynamic state d) External state

Figure 5. Basic state sub-types in SWC notation.

Static states (figure 5a) are the most basic structures to represent information in SWC. A *static state* refers to a container with a static set of objects; once in a static state the same set of objects is always presented. However, the objects it contains are not necessarily static by themselves; they could have dynamic behaviour as we usually find, for example, in applets, JavaScript or animated images. Static is the default type for *States*.

Transient states (figure 5b) describe a non-deterministic behaviour in the state machine. *Transient* states are not part of the original statecharts notation, but they are needed when a single transition cannot determine the next state of the state machine (see figure 8 for an example). The formal definition for a transient state says that only *completion* or *system* events are accepted as outgoing transitions. Frequently they refer to server-side parts of web applications, such as CGI[4] and Java Servelts programs. *Transient states* only process instructions and they do not have a visual representation towards users.

Dynamic states (figure 5c) represent content that is dynamically generated at runtime. Usually they are the result of a *transient state* processing. The associated

[3] http://www.w3.org/DOM/

[4] CGI - *Common Gateway Interface*

container of a *dynamic state* is empty. The semantics for this state is that in the modelling phase designers are not able to determine which content (transitions and objects) will be made available at run time. However, designers can include static objects and transitions inside dynamic states; in such case transitions are represented, but designer must keep in mind that missing transitions might appear at run time and change the navigation behaviour.

External states (figure 5d) represent information that is accessible through relationships (transitions) but are not part of the current design. For example, consider two states A and B. While creating a transition from A to B, the content of B is not accessible and cannot be modified. Thus B is considered external to the current design. Usually they represent connections to external sites. *External states* avoid representing transitions with no target state, however all activities (whatever it is *entry, do, exit*) in *external states* are *null*.

Events are classified in SWC notation according to the agent triggering them: **user** (e.g. a mouse click), **system** (e.g. a method invocation that affects the activity in a state) or **completion** (e.g. execute the next activity). A *completion* event is a fictional event that is associated to transitions without triggers, e.g. change the system state after a timestamp. Fictional *completion* events allow us to give the same representation for all transitions in SWC machines. This classification of event sources is propagated to the representation of transitions. Transitions whose event is triggered by a user are graphically drawn as continuous arrows (figure 6a.) while transitions triggered by *system* or *completion* events are drawn as dashed arrows (figure 6b).

a) b)

Figure 6. Graphical representation of SWC transitions.

Even though figure 6 only shows transitions ids, we can promptly identify who owns the control on the activation of a transition, whether the system (transition *t2*) or a user (transition *t1*). In order to be able to use the SWC models to perform usability evaluation, the fact that a transition is related to a user event or not is critical. Thus, SWC puts in a single graphic representation those transitions (*completion* and *system* transitions) that are not triggered by the users. If explicit representation is required for distinguishing between completion and system events, a full label for transition (as presented by figure 1a) such as *t2=completion/true:action1*.

4. Web Navigation Modelling with SWC

Web applications have some similarities with hypermedia and hypertext systems such as the occurrence of multimedia content, linking information units, etc., but many other features are specific to the web environment such as the mandatory use of browser, client/server architecture, and so on. This section describes the most

important features related to navigation design for web applications and their corresponding representation with SWC notation, when it is applicable.

4.1. Browser effects

Web applications can only be accessed through dedicated client applications called web browsers. Browsers interpret every single page sent back by the web server before to display it according proprietary (browser vendors') directives for technology, client-side system platform (e.g. PC Windows, Palm, etc) and additional preferences for display set by users. In addition, from a user interface perspective, the browser itself proposes functions (e.g., cut, copy, save) that could compete with the ones proposed by the application. Recent works [7, 17] have analysed the non-uniform implementation of functions such as history mechanisms (back button and history list, for instance) of web browsers. The worst is that most users rely on such mechanisms to navigation because web applications provide poor navigation [5].

As several browsers with different capabilities are available, it is almost impossible for the designer of the web application to know precisely the software environment of the user. Moreover, it impossible to predict when users will make use of back of the application, so, it is not an advisable strategy to represent browser controls (such as *back button* and other history mechanisms) as part of application design. Such controls are considered as interaction mechanisms such as scroll bars and windows selection that are not represented by SWC notation.

4.2. Link types support

When analysing how pages are related to each other on web applications we can observe three different types of links: a) internal-pages links, b) inter-pages links and c) external links. *Internal-pages* links related different parts of a same web page, which can be very helpful for long documents. These links present the same semantic behaviour of scroll bars on windows browser, so at first sight we can consider irrelevant to include the specification of such elements into navigation design. If required, *internal-links* can be easily represented with SWC by decomposing the page in a composite state and create links between the sub-parts of the document as presented by figure 7a. As we can see in figure 7a is a *spaghetti*-link interconnection between all sections of a same document.

Inter-page links is the most classical example; it means a simple connection of two pages belonging to the same web site. We can see in figure 7b how two pages can be connected by an *inter-page* link with passage of a parameter that indicates the subsection in the document to be displayed. Figure 7b is a preferable representation for the same problem described above (*internal-pages*) because it increases the legibility of diagrams.

External links are links connecting the web application with foreign web sites or non-relevant parts of the web application for the current design. Even though its name makes a reference to a link, this concept is treated as a state because it is not possible

to represent targetless transitions in SWC, even though the transition makes references to an external site.

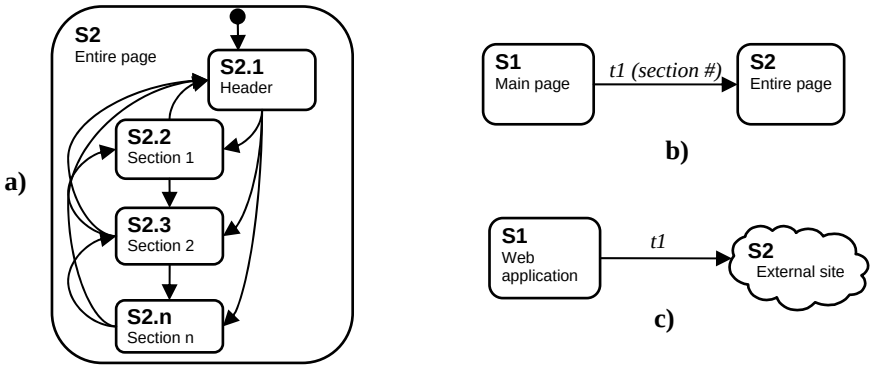


Figure 7. Links types: a) internal-page, b) inter-page and c) external.

4.3. System-driven navigation (the use of transient states)

In many cases, the combination of event plus condition determines the next state. However, it not true for all cases. Consider the case of user authentication in figure 8. In this example, the event *press button* (to send user name and password) in transition *t1* does not count to determine whether user will get access to the system or not. But it is the result processing of the transient state *S2*. Notice transitions *t2* and *t3* going out from *S2* presenting system events.

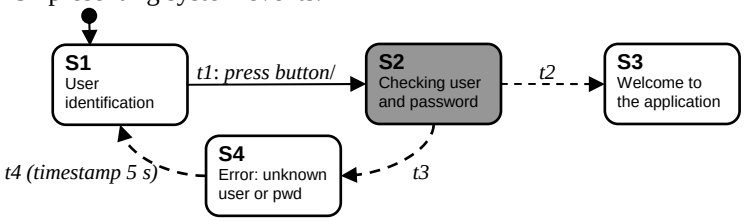


Figure 8. Example of a simple user authentication .

In most cases, the user will send additional information filling in forms or following parameterized links to a server-side application (represented by a transient state) that will execute some processing and then send back user the appropriate answer.

4.4. Dynamic content generation

A particular feature of web application is the dynamic generation of pages on an application by server-side applications. Dynamic pages does not exist on the web server and that is why the function $\mathbf{\Omega : S \rightarrow p}$ (see section 3) maps an empty set to dynamic states. Dynamic states represent such unpredictable content for the page but it does not exclude the possibility to represent required transitions for the design.

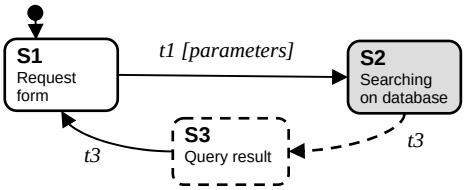


Figure 9. Query search.

Figure 9 shows a classical example of dynamic content generation as result of a database query. Notice that the dynamic state *S3* has a user transition that allows user return to the request form (state *SI*). It important to note that, at run time, the page resulting by the database searching can include links that are not represented in the modelling and may alter the navigation on the web application.

4.5. Frames

Frames are elements that split the browser's windows into two or more concurrent visual areas were each region can load and display a different document. *Frames* were introduced as a standard in HTML 4.0[5]. Links in a frame region can alter the exhibition of documents in another frame region. *Frames* are modelled in SWC with AND-states where each orthogonal region represents an individual frame, as shown in figure 10. When entering the state *A*, two concurrent regions are activated *A'* and *A''* which pass the control to their initial states *B* and *C*, respectively. When the transition *t2* is fired, the configuration in region *A''* changes arbitrarily to states *D*, the region *A'* maintains its configuration.

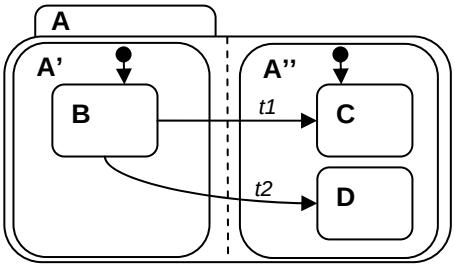


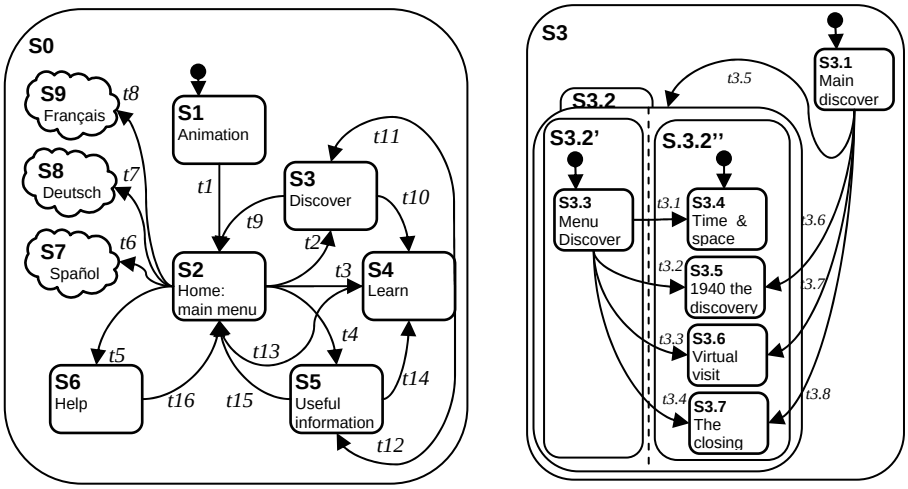
Figure 10. Concurrent visual area representations (frames).

4.6. Modularization

The number of pages on most web applications increases very quickly and the representation of documents and links became a problem in flat-notation such as automata [4]. In addition, large projects must be cut in small part and splat among the member of development team. The modularization is also required to deal with the complexity during the development. SWC takes benefits from the multi-level

[5] http://www.w3.org/TR/1998/REC-html40-19980424/

hierarchy from classical statecharts to better manage large web applications. Figure 11 presents a partial modelling for the web site *The Cave of Lascaux*[6].



a) Global view b) *S3* details

Figure 11. Hierarchical view for the Web Site '*The Cave of Lascaux*'

Figure 11a present the global view for the application, which contains 9 states, some of them are composite whose details are not represented in higher level. For example, *S3* is a composite state whose details are shown in figure 11b. In such approach, composite states represent classes of pages which share the same structure. Sub-states inherit relationships from their parents. For example, in Figure 1a, the transitions (*t9 a*nd *t10*) going from state *S3-Discover* to states *S2-Home-main menu* and *S4-Learn*, respectively, are shared by all *S3* sub-states (*S3.1, S3.2, S3.2', S3.2'', S3.3, S3.4, S3.5, S3.6* and *S3.7*). The states at the left are instances of classes of pages that have their own navigation. For reasons of space only state *S3* is detailed in this modelling, even though the S3 some of its sub-states (*S3.4, S3.5, S3.6* and *S3.7*) are at their turn suitable to be decomposed in modules.

4.7. Dialog control modelling

Modelling dialog control means to identify who (system or user) causes events changing the interface. As before mentioned in section 3, SWC explicitly represents system interaction (by *system* and *completion* events) and user control (by *user events*). A typical example of system event is timed transitions used to redirect Web pages. In the figure 12, users start at the state *S1*, which contains two associated transitions: *e2* and *e3*. The transition *e2* represents a system event that, once activate, will change system state to *S2* five seconds [5s] after users have been entered in *S1*. Users can also cause a transition by selecting a link associated to user event *e3*.

[6] http://www.culture.fr/culture/arcnat/lascaux/en/

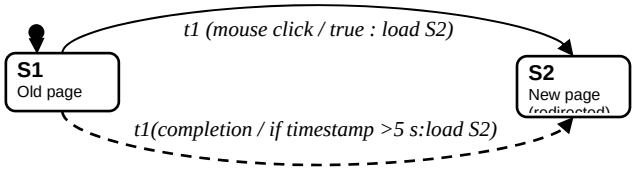


Figure 12. User x System dialog control.

4.8. Client-side and server-side execution

On its origins Web applications were built over a client/server architecture where the server-side is responsible for all processing leaving to the client-side (the web browser) just the display of the content information. The advent of new technologies such as JavaScript, Java and Active-X, for example, put on the client more interactive than just display functions. We define *client-side execution* as any processing changing the state of the application without communication with a web server. *Server-side execution*, at its turn, is defined as any instruction processed on a web server following a client's request.

Transient states and *system/completion* transitions in SWC are suitable to describe executable states and system initiative on web applications but they say nothing about where (on the client- or server-sides) it occurs. SWC do not impose a particular architecture for the design and a modelling can be quite easily implemented using as thin-client architecture (no processing in the client-side) or a robust-client (full client-side functionality). However, at this time we have not included a description about how to model objects in a container, so we could consider that *transient* states are always on the server-side.

5. Discussion and Related Work

Research work in navigation modelling has a long history in hypertext and multimedia domain [16, 20 and 22]. Web applications are directly originated from this research field and much of the web technology related to construction of web pages find its main contributions in hypertext and hypermedia research work.

State-based notations such as Petri nets [16] and Statecharts [14, 18, 20] have been explored to represented navigation for hypertext systems. However, when trying to represent web applications they do not model dynamic content generation, web link-types support (external states, for examples), client and server-side execution, and other aspects related to web domain. Besides, some of them [16, 20] do not make explicit the separation between interaction and navigation aspects in the models while this is a critical aspect for web application.

More recent work devoted to web applications, propose efficient solutions to describe navigation and architecture in a single representation, as it has been done by Connallen [3] with UML stereotypes and Fraternalli with WebML [2]. These approaches mainly target data-intensive applications and propose even prototyping environments to increase productivity. The main inconvenient is that navigation is

described at a very coarse grain (for instance navigation between classes of documents) and it almost impossible to represent detailed navigation on instances of these classes or documents. The same problem appears in KOCH [11]. Other approaches such as UML stereotypes as in [3] and WebML [2] may reduce refrain creativity at design time as they impose the underlying technology and as they do not provide efficient abstraction views of the application under developement.

Other studies such as those presented in [4 and 12] take into account all the navigation aspects of web applications (that have been presented in section 4). They are able to represent dynamic content generation and provide efficient support to link-types. However, they do not allow for explicating who (between the user and the system) is triggering events.

Table 1 presents a summary of several notation dedicated to navigation modelling coming from different domains such as Wimp interfaces, hypertext/multimedia systems and web applications. We compare how these notations are able to deal with web design concerns such as those described in section 4. Each aspect is rated according to the following values:

- **N** (no) means that the notation does not support the modelling of such aspect or if it is possible, no information is available on how to cope with it;
- **C** (cumbersome) i.e. the notation provides some support for modelling this aspect but some limitations exist;
- **P** (primitive) the aspect is fully supported and fully documented in the approach (it can be seen as a primitive of the notation).

Table 1. Comparative study of several notations for modelling navigation.

	Methods/Notations											
	WIMP interfaces		*Hypertext/Multimedia systems*				*Web applications*					
Web Design Features	Horrocks' statecharts [10]	UMLi [21]	Petri nets [16]	Zheng&Pong's statecharts [20]	HBMS [14, 18]	OOHDM [15]	Automata [4]	UML class diagrams [11]	UML stereotypes [3]	WebML [2]	Leung's statecharts [12]	StateWebCharts - SWC
Interaction Modelling	**P**	**P**	**P**	**P**	**P**	**P**	**P**	**N**	**N**	**N**	**N**	**C**
Navigation modelling	**C**	**C**	**P**	**P**	**P**	**P**	**P**	**P**	**P**	**P**	**P**	**P**
(Web) Link-types support	**N**	**C**	**C**	**C**	**C**	**C**	**P**	**P**	**P**	**N**	**P**	**P**
System-driven navigation	**N**	**C**	**N**	**N**	**N**	**N**	**N**	**N**	**P**	**P**	**P**	**P**
Dynamic content generation	**N**	**N**	**N**	**N**	**N**	**P**	**C**	**P**	**P**	**P**	**P**	**P**
Frames	**N**	**N**	**N**	**P**	**P**	**C**	**P**	**N**	**N**	**N**	**P**	**P**
Modularization	**P**	**P**	**C**	**P**	**P**	**P**	**C**	**P**	**P**	**P**	**P**	**P**
Dialog control modelling	**N**	**N**	**N**	**N**	**N**	**N**	**N**	**N**	**N**	**N**	**N**	**P**
Client-side execution	**N**	**N**	**N**	**N**	**N**	**N**	**N**	**N**	**P**	**P**	**N**	**C**
Server-side execution	**N**	**N**	**N**	**N**	**N**	**N**	**N**	**N**	**P**	**P**	**N**	**C**

Legend: N no information is provided, **C** cumbersome, **P** primitive.

6. Conclusions and Future work

In this paper we have presented a statechart-based formalism, StateWebCharts (SWC), which is able to deal with navigation design of web applications. SWC is a technological-independent notation whose main intention is to enable designer to model all the specific features required for modelling navigation of web applications.

One of the contributions of the SWC notation proposed here is that it makes explicit in the models the points where users interact with the application with respect to those where the system drives and controls the navigation. Moreover, all elements in SWC have a clear semantic with a corresponding visual representation, which is supposed to increase the legibility of the models. SWC supports *client-side execution* and *server-side execution* with some limitations as explained in section 4.8. However, this is an intended limitation as solving this problem (for instance by including architectural information within the notation) would bind models to implementation/architectural concerns too early on the design process. In the same way, SWC is not the best solution for representing interaction on objects inside states. Here again, the focus of SWC is more on the early design phases where low level interaction modelling is premature. Besides, several notations deal very efficiently with these aspects and our goal is more to integrate SWC with such approaches rather than making it suitable for all purposes.

Relationships between SWC models and other models that has to be built during the development process of web applications has already been studied and can be found in [23]. For instance this paper presents how conformance between task models and SWC can be checked. This is another advantage of using formal description techniques for navigation modelling.

As for future work, we intend to use SWC model as a key component of the evaluation phase. Indeed, this phase is really critical for web application development as they are by nature hard to test and evaluate. The idea is to exploit the models to pilot and drive (possibly remote) evaluations by providing users with structural information about navigation and continuously monitoring coverage of the tests.

Acknowledgments

This work has been partially supported by Capes/Cofecub SpiderWeb project. First author is also sponsored by CNPq (Brazilian Council for Research and Development).

References

1. Campos, J. C., Harrison, M. D. (1997) Formally Verifying Interactive Systems: A Review. In Harrison, M.D.&Torres, J.C.(eds.), DSVIS'97, 109-124 pp, Springer.
2. Ceri, S.; Fraternali, P. & Bongio, A. Language (WebML): a modelling language for designing Web sites. In Proc. 9th WWW Conference, Amsterdam, May 2000.
3. Connallen, J. Building Web Applications with UML. Addison-Wesley, 1999.

4. Dimuro, G. P.; Costa, A. C. R. Towards an automata-based navigation model for the specification of web sites. In…: 5th Workshop on Formal Methods, Gramado, 2002. Electronic Notes in Theoretical Computer Science, Amsterdam, 2002.
5. Fleming, J. Web Navigation: Designing the User Experience. O'Reilly. 1998.
6. Fraternali, P. Tools and approaches for developing data-intensive Web applications: a Survey. ACM Computing Surveys, 31(3), 227-263p. 1999.
7. Greenberg, S. and Cockburn, A. Getting back to back: Alternate behaviors for a web browser's back button. In Proceedings: 5th Annual Human Factors and the Web Conference, Maryland, USA, 1999.
8. Harel, D. Statecharts: a visual formalism for computer system. Science of Computer Programming, 8, N. 3:231-271 p., 1987.
9. Harel, D.; Naamad, A. The STATEMATE semantics of statecharts. ACM Trans. Software Engineering Methodology, vol. 5, 4 (Oct. 1996), 293–333 pp.
10. Horrocks, I. Constructing the User Interface with Statecharts. Addison-Wesley, Harlow. 1999, 253 p.
11. Koch, N.; Kraus, A. The expressive Power of UML-based Web Engineering. In 2nd International Workshop on Web-oriented Software Technology (IWWOST02). D. Schwabe, O. Pastor, G. Rossi, and L. Olsina (eds.), June 2002.
12. Leung, K., Hui, L., Yiu, S., Tang, R. Modelling Web Navigation by StateCharts. In Proceedings: 24th Inter. Comp. Software and Applications Conf., 2000, Electronic Edition (IEEE Computer Society DL).
13. Murugesan, S.; Deshpande, Y. (2001). Web Engineering: Managing Diversity and Complexity of Web Applications Development. Berlin: Springer.
14. Oliveira, M.C.F. de; Turine, M. A. S.; Masiero, P.C. A Statechart-Based Model for Modeling Hypermedia Applications. ACM TOIS. April 2001.
15. Schwabe, D.; Esmeraldo, L.; Rossi, G. & Lyardet, F. (2001) Engineering Web Applications for Reuse. IEEE Multimedia, 8(1), 20-31.
16. Stotts, P. D.; Furuta, R. Petri-net-based hypertext: document structure with browsing semantics. ACM Trans. on Inf. Syst. 7, 1 (Jan. 1989), Pages 3 - 29.
17. Tauscher, T and Greenberg, S. How people revisit web pages: Empirical findings and implications for the design of history systems', International Journal of Human Computer Studies 47(1), 97–138. 1997.
18. Turine, M. A. S.; Oliveira, M. C. F.; Masieiro, P. C. A navigation-oriented hypertext model based on statecharts. In Proceeding… 8th ACM Hypertext Conf. April, 1997, Southampton United Kingdom. Pages 102 - 111.
19. Winckler, M.; Farenc, C.; Palanque, P. & Bastide, R. Designing Navigation for Web Interfaces. IHM-HCI2001 Proceedings, Lille France, September 2001.
20. Zheng, Y.; Pong, M. C. 1992. Using statecharts to model hypertext. In Proceedings of the ACM Conference Pankaj K. Gargypertxt (ECHT'92, Milan, Italy). ACM Press, New York, NY, 242-250.
21. Silva, P. P. da, Paton, N. W. UMLi: The Unified Modelling Language for Interactive Applications. In 3rd International Conference on the Unified Modeling Language UML'2000. LNCS V.1939, 117-132 p., Springer, Oct. 2000.
22. Halasz, F., Schwartz, M. The Dexter hypertext reference model, Communications of the ACM, v.37 n.2, p.30-39, Feb. 1994
23. Winckler, M.; Palanque, P.; Farenc, C.; Pimenta, M. Task-Based Assessment of Web Navigation Design. In Proceedings: ACM TAMODIA'02, Bucharest, 2002.