Construction adaptative de concepts par structuration d'entités de traitement d'images

Adaptative Concept Building by Image Processing Entity Structuration

Yazid Abchiche, Patrice Dalle et Yohann Magnien

Institut de Recherche en Informatique de Toulouse

IRIT - Université Paul Sabatier

118, route de Narbonne

31062 Toulouse Cedex 4

France.

Yazid.Abchiche@irit.fr, Patrice.Dalle@irit.fr, Yohann.Magnien@irit.fr

Résumé :

Cette recherche se situe dans le cadre de la conception interactive et incrémentale d'applications de traitement et d'analyse d'images. Dans notre approche, un système spécialisé sera le résultat d'un dialogue entre un concepteur et un Système Multi-Agents (SMA) adaptatif[1]*.*
À partir d'interactions avec un concepteur désirant le spécialiser pour résoudre un problème particulier (par exemple identifier et localiser un objet), le système fait évoluer sa configuration de manière à privilégier les liens entre les opérateurs contribuant à la résolution.
Il est réalisé en encapsulant chaque opérateur par un agent dont le comportement est essentiellement coopératif. Les agents exploitent un formalisme original [10] pour représenter les informations qu'ils traitent et qu'ils produisent. La configuration des opérateurs résulte donc de l'auto-organisation du collectif d'agents qui transforment l'information et attachent aux entités produites une description formelle de l'effet de ces transformations. La compétence des agents de traitement d'images est représentée via les descripteurs de propriétés de traitement d'images qu'ils peuvent manipuler. Leur évolution résulte des interactions entre agents et des réactions de l'utilisateur aux résultats obtenus.

Mots-clés :

Traitement d'images, conception d'application, construction de concepts, modélisation du traitement d'images, système multi-agent, interaction et autonomie.

Abstract:

This work takes place in the field of interactive and incremental design of image processing and analyzing applications. In our approach, a specialized system will be the result of a dialog between a designer and an adaptative Multi-Agent System (MAS).
The interactions with a designer who wants to specialize the system in order to solve a specific problem (for example, to localize and identify an object), will lead the MAS evolution so that the stress is put on the links between the operators contributing to the solution.
It is achieved by encapsulating each operator in an agent whose behavior is essentially cooperative. The agents use an original formalism [10] to represent the information they process and produce. The operator configuration is the result of the agent self-organization, which transforms the data and attaches a formal description of those transformation effects to the produced entities. The image processing agent skills are represented through image processing property descriptors that the agents can handle. The agent society evolution is the result of the interactions among the agents and of the user reactions about the produced results.

Keywords:

Image processing, application design, concept building, image processing modeling, multi-agent system, interaction and autonomy.

1 Introduction

Nous présentons une nouvelle démarche de conception d'application de Traitement d'images (TI).
Une application est le résultat de la configuration d'un système adaptatif qui se spécialise pour répondre au problème particulier posé par le concepteur. Cette adaptation résulte d'un dialogue entre le concepteur, qui connaît les concepts du domaine et cherche à en fournir une description visuelle, et le système, qui élabore des concepts artificiels en structurant les données, en les décrivant, en les présentant au concepteur et en analysant ses réactions.
Le concepteur considère que l'ensemble du système est configuré lorsqu'il se comportera comme s'il attachait le même sens que lui aux entités présentes dans l'image.
Pour parvenir à un tel fonctionnement, le système doit être doté d'un comportement autonome lui permettant de construire ses propres concepts à partir des données. Il doit donc en avoir une représentation et savoir la manipuler. La

[1] Cette recherche est menée en partenariat avec l'équipe SMAC (Systèmes Multi-Agents Coopératifs) de l'IRIT, dirigée par P. Glize.

représentation (que nous décrivons au §3.3) doit recouvrir deux notions :

- le concept lui-même, c'est-à-dire les caractéristiques communes des entités instances de ce concept,
- le mécanisme permettant de construire des instances du concept. Il ne doit pas être exprimé par un graphe d'opérateurs, mais faire appel à un formalisme plus abstrait et plus générique.

D'autre part le système doit pouvoir décrire et présenter les concepts à l'utilisateur. La représentation des concepts devra être basée sur la notion d'indices visuels, ce qui permet d'établir un langage visuel commun avec l'utilisateur et de construire une interface lui présentant ces concepts et recueillant ses réactions.

Après avoir situé cette approche par rapport aux différentes méthodes de conception d'application de TI et montré l'intérêt des SMA pour concevoir des systèmes adaptatifs (*cf.* §2), nous décrivons l'architecture générale du système proposé (*cf.* §3).
Elle possède deux niveaux interconnectés :

- Un système de base (*cf.* §3.1) qui met en œuvre les opérateurs de TI pour analyser et structurer les données images. Il a des capacités d'auto-organisation.

- Un système secondaire (*cf.* §3.2) qui construit du "sens" (artificiel) sur les productions du système de base. Il permet de représenter l'organisation du système de base et de la faire évoluer en injectant des tendances sous la forme de descriptions de concepts à rechercher. Il a donc un comportement autonome pour élaborer ces concepts, mais il a aussi une forte connexion avec l'utilisateur pour les associer à des notions du domaine d'application.

La communication entre les deux composantes du système et entre celui-ci et l'utilisateur exploite un formalisme basé sur une modélisation des entités et des opérateurs de traitement d'images (*cf. §*3.3).
Nous décrivons ensuite le principe de fonctionnement du système (*cf. §*4) et nous précisons enfin les options d'implémentation que nous avons retenues (*cf. §*5).
La première version de ce système a été réalisée en simulant le système secondaire par des artefacts [8]. Quant aux modalités d'interactions avec l'utilisateur, elles ont été présentées dans [19] et sont utilisées dans le cadre d'un système de construction d'ontologies de traitement d'images [20].

2 État de l'art

Différentes approches ont été utilisées ou sont en cours d'étude pour construire une application spécialisée.

- L'espace des problèmes peut être représenté par un graphe de toutes les liaisons possibles entre les entités que peut générer le TI [17]. Construire une application revient à trouver un chemin dans ce graphe. L'inconvénient d'une telle approche est son caractère figé et prévisible.

- Les systèmes de planification, qu'ils soient restreints à des domaines particuliers comme MVP [5] ou génériques comme BORG [6] cherchent à dépasser ces limitations en possédant une autonomie de conception. Les approches sont généralement fonctionnelles. Elles décomposent le TI en niveaux d'abstraction (tâches, fonctionnalités, méthodes) et exploitent une expertise en résolution de problèmes de TI. La difficulté sera alors d'acquérir et de représenter cette expertise. En outre, l'utilisateur devra savoir exprimer sa requête de manière suffisamment complète pour que l'expertise du système puisse s'appliquer.

- Enfin, des approches mettent l'accent sur le caractère adaptatif et opportuniste du système [18]. Le système est conçu comme un environnement de pilotage de programme. Cependant, la phase de spécialisation de ce système à un domaine d'application particulier reste encore le plus souvent à la charge du concepteur.

- Nos précédents travaux [4] [10] [9] ont cherché à apporter une réponse à la planification de graphes d'opérateurs et à leur validation, en introduisant une modélisation des données et des traitements d'images. Celle-ci est basée sur une représentation des transformations d'information effectuées par chaque opérateur. Elle permet de propager l'effet de ces transformations dans le système et de guider ainsi son évolution. Si elle facilite l'expression du problème, il reste néanmoins nécessaire de le formuler complètement.

Les systèmes présentés ci-dessus sont souvent implémentés par des objets et des règles ou par des systèmes à base de tableau noir. Cependant les agents sont de plus en plus utilisés en analyse d'image. [3] par exemple, implémente sous forme d'agents un système de segmentation faisant coopérer contours et régions et que l'utilisateur exploite en situant les agents initiaux et en contrôlant leur évolution. [15] définit un cadre méthodologique de segmentation d'images basé sur la coopération de méthodes complémentaires et utilisant des agents spécialisés.
Dans le domaine de la vision, [13] met l'accent sur les protocoles d'interaction entre agents et sur la modélisation de leur communication. L'utilisation des agents pour représenter les connaissances est maintenant courante en vision par ordinateur. Cependant, il n'y a pas une définition unique de la notion d'agent. Pour chaque application, un modèle d'agent est défini. [7] relève plusieurs avantages de l'approche multi-agent pour le traitement d'images :

- facilité de construction et de mise à jour,
- possibilité d'exploitation d'une architecture parallèle,
- capacité de focalisation,
- résolution de problèmes hétérogènes et fiabilité.

Les SMA se sont en effet révélés adaptés pour prendre en compte les conditions incertaines fréquemment rencontrées en TI et sont à la base de différents travaux sur la vision ou l'interprétation [2] [14]. À ces qualités, nous ajouterons la capacité de produire des comportements émergents à partir des interactions entre les agents.
Nous avons proposé [8] une première implémentation d'un système multi-agent à un seul niveau, les agents encapsulant chaque opérateur de TI et échangeant leurs données sur le principe de coopération. Cependant ces agents ne possédaient pas de connaissances sur le TI et il fallait donc savoir formuler complètement l'objectif, le système s'arrêtant lorsque cet objectif était atteint, ce qui lui donnait un comportement assez combinatoire.
L'introduction d'une représentation du TI (*cf. §*3.3) nous permet de lui donner un comportement beaucoup plus adaptatif. Elle permet aussi de définir un second niveau orientant le système de base en lui décrivant des objectifs et analysant l'évolution de sa configuration.

3 Présentation du système

Le système est décomposable en 2 sous-systèmes : le système de base et le système secondaire.

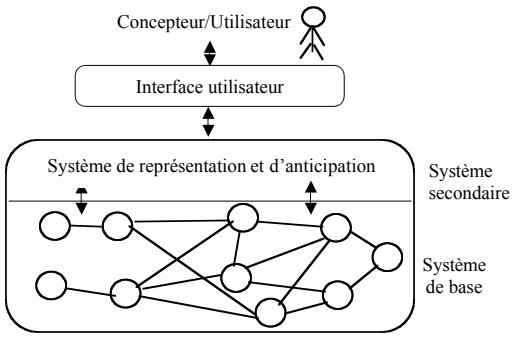


Figure I *:* schéma général du système de TI.

Il communique d'une part avec son environnement pour acquérir des images et d'autre part avec l'utilisateur, concepteur d'application en phase de configuration et utilisateur final en phase d'exploitation.

3.1 Système de base

Le rôle du système de base est d'enchaîner, de combiner et de contrôler des opérateurs. Chaque opérateur remplit une fonction de traitement d'images, de structuration de données ou de construction d'entités à différents niveaux de complexité. Ce système n'a aucune notion du sens des entités qu'il construit. Il est réalisé sous la forme d'un système multi-agent (*cf. §*5). Avant d'être spécialisé, il présente *a priori* une forte combinatoire. Ses caractéristiques sont les suivantes :

- Granularité faible : l'unité de base est la fonction de traitement d'images, qui peut être très élémentaire (c'est-à-dire qu'elle met en jeu un seul descripteur). Chaque agent encapsule un opérateur de base de traitement ou d'analyse d'images (ex: seuillage, extraction de primitive, segmentation, mesure, etc.).

- *Configurable* : possibilité de construire des macro-opérateurs. En effet, à l'achèvement de la conception, nous aurons des forces de liens (croyances) plus importantes pour certaines accointances.

- *Fortement connecté* : chaque opérateur est connecté à tous les agents à qui il peut fournir des informations et à tous ceux qui peuvent lui fournir des informations.

- *Comportement autonome* : le système cherche à réaliser une fonction globale de structuration des données et d'agrégation des fonctions. L'autonomie des agents permet de leur donner un comportement adaptatif et donc leur permet de s'adapter aux différentes conditions opératoires.

3.2 Système secondaire

Son rôle est double. D'une part, il construit des concepts artificiels : il décrit formellement une abstraction des entités produites par le système de base. Cette description constitue une représentation des concepts dont les entités sont des instances. De plus il dispose de mécanismes de simplification, d'abstraction, de combinaison permettant la production de nouveaux concepts à atteindre, ce qui orientera l'évolution du système de base.
D'autre part, il propose et confronte ses concepts artificiels à ceux de l'utilisateur et il modifie ses objectifs en fonction des réactions de celui-ci.

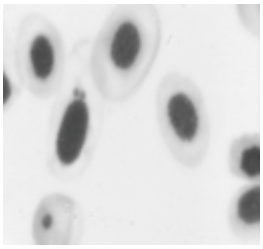
3.3 Langage de description de concepts

Les objectifs du système de base sont donnés en termes de concepts ; ils indiquent donc des entités à construire, résultats de l'action d'un ensemble d'opérateurs à configurer. On a donc besoin d'un langage qui rende compte des entités obtenues et qui permette de mesurer une « distance » avec l'entité souhaitée. D'autre part, le système secondaire décrit et analyse l'évolution d'organisation du système de base. Le langage doit donc aussi rendre compte des actions effectuées pour produire l'entité courante.

Le formalisme proposé par P. Dejean [10] à la suite des travaux de modélisation du TI menés par J.-M. Inglebert [16] semble satisfaire ces deux contraintes. Il fournit une description orientée concepts et entités du TI. Il dispose d'un Langage de Description des Concepts (LDC) [12] basé sur la notion d'indices visuels, liés aux mesures effectuées par les opérateurs sur les données. D'autre part, un modèle des données de TI en cinq champs (trame, cache, index, signal, codage) permet d'implémenter un Langage de Description des Données (LDD) [10] décrivant les cinq champs à partir du formalisme LDC.

Exemple d'expression du langage :

Image : cellules.



noyaux_ou_micro-noyaux : les taches sombres
(((nivg 0 113)))

cytoplasmes : les taches sombres ou grises (inclut les noyaux_ou_micro-noyaux)
(((nivg 0 226)))

cellules : les cytoplasmes qui contiennent des noyaux_ou_micro-noyaux (en fait, qui intersectent, pour prévoir le cas des noyaux qui touchent le bord du cytoplasme)
((inter ((vois4)(cytoplasmes)) (noyaux_ou_micro-noyaux)))

micro-noyaux : petits noyaux_ou_micro-noyaux
(((srf (mindom srf) 100) ((vois4)(noyaux_ou_micro-noyaux))))

cellules_avec_micro-noyaux : cellules contenant un ou des micro-noyaux
((inter ((vois4)(cellules)) (micro-noyaux)))

On voit que cette expression «*cellules_avec_micro-noyaux* », obtenue par application d'un graphe d'opérateurs, décrit bien un résultat, les caractéristiques formelles d'une entité ; elle permet de retrouver le graphe d'opérateurs qui l'a produite, par planification [9] ; elle rend donc compte de son organisation, mais elle en est bien distincte.

On peut agir sur cette expression pour :

- *La généraliser* : on dispose de quatre moyens différents : on peut étendre un domaine, changer un descripteur par un descripteur plus général, supprimer un descripteur ou bien combiner (disjonction) l'expression avec celle d'un autre concept.
- *La spécialiser* : opération inverse de la généralisation.
- *La combiner* avec une autre.

Ces possibilités d'actions sur les expressions du langage permettent les mécanismes de description, d'abstraction, de simplification, de combinaison que doit avoir le système secondaire. De plus, les descripteurs du langage rendent compte des actions effectuées pour obtenir l'entité courante. Le formalisme permet donc bien de relier les deux systèmes afin de mettre en œuvre les mécanismes de descriptions et d'anticipation du système secondaire.

4 Fonctionnement du système

Pour résoudre un problème posé par l'utilisateur, le système fait évoluer sa configuration de manière à constituer un graphe d'opérateurs capable d'extraire l'information demandée dans l'image. Cette évolution est réalisée de manière incrémentale via des cycles d'interactions et d'exécutions au cours desquels l'utilisateur précisera son objectif en fonction des résultats présentés par le système.

Pour initialiser le système de base, des agents (en général d'identification) reçoivent une requête du système secondaire qu'ils propagent dans tout le système (en fonction de leurs accointances), en la reformulant, jusqu'à ce que d'autres agents aient assez d'information pour déclencher un traitement. Les données se répandent dans tout le réseau, jusqu'à ce que les agents qui ont reçu la requête initiale aient les informations suffisantes pour y répondre de manière satisfaisante.

En fonction des évaluations réalisées par les agents d'interprétation, une information d'utilité est propagée en retour ; elle permet de modifier les poids des liens (degrés de croyance, *cf. §*4.1.2) entre agents de manière à ce qu'à terme, l'opérateur le plus pertinent soit exécuté, avec le paramétrage adapté.

Le système secondaire produit des concepts artificiels qu'il présente à l'utilisateur via l'interface. Lors de la phase de spécialisation, le concepteur réagit pour faire apprendre au système les concepts du domaine d'application. Pour cela, il dispose des moyens d'interaction de l'interface pour guider le système vers la production des concepts souhaités et pour nommer les concepts artificiels produits ayant du sens dans le domaine .
Lors de la phase d'exploitation par l'utilisateur final, le système présente les concepts appris ayant du sens dans le domaine.
Les deux sous-systèmes s'influencent mutuellement : lorsque le système de base produit de nouvelles entités, leur description influence le système secondaire ; de même, lorsque le système secondaire produit de nouveaux concepts, ils sont traduits en de nouvelles orientations pour faire évoluer le système de base.

4.1 Fonctionnement du système de base

Le principe de collaboration entre les agents est la coopération : la force des accointances est modifiée suivant la possibilité d'exploitation des résultats de l'action d'un agent par les agents récepteurs (principe d'utilité). Pour cela, chaque agent cherche à détecter les situations non-coopératives [21] qui sont réduites dans notre cas au *conflit*, à l'*inutilité* et à la *concurrence*. Elles sont décrites dans le module de résolution de cas de non-coopération (*cf. §*4.1.1).

4.1.1 Modèle d'agent

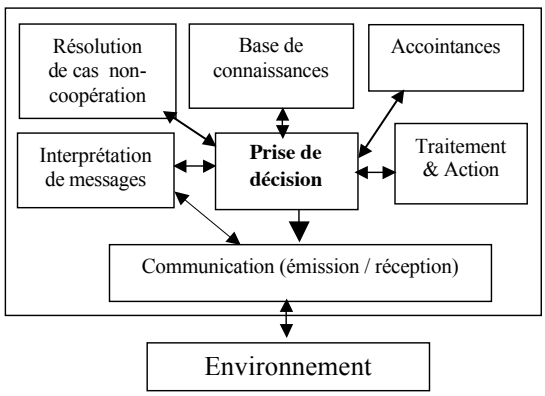


Figure II *:* agent et son environnement.

Tous les agents du système de base disposent des modules de la figure ci-dessus.

- **Base de connaissances**

Elle représente la croyance en soi (connaissances relatives à l'opérateur encapsulé par l'agent, ses entrées, ses sorties, ses transformations). Les services proposés par l'agent sont donc :

- L'effet de l'opérateur de TI qu'il encapsule, sur la donnée,
- La description des concepts produits sous la forme d'expressions du LDC : à partir de l'expression décrivant la donnée d'entrée, l'agent produit une nouvelle expression intégrant les transformations effectuées par son opérateur.

Ces services et son expertise dans le domaine, lui permettent de répondre à une requête d'un autre agent. Une valeur est attribuée initialement à chaque possibilité, que l'agent augmente ou diminue en fonction de ses interactions avec les autres agents. Elle évolue avec son expérience dans le temps.

La phase d'apprentissage lui permet d'ajuster les paramètres liés à l'opérateur de TI qu'il encapsule et de donner une configuration type pour le domaine d'expertise.

Les effets des interactions avec l'expert du domaine, transmis par le système secondaire, permettent de préciser la configuration type afin de faire émerger des concepts plus précis du domaine.

- **Module de communication**

Il gère l'émission et la réception des messages entre l'agent et son environnement. Il contient les procédures d'envoi et de réception de messages.
Le langage d'interaction entre les agents est une forme simplifiée de celui proposé par Y. Demazeau *et al*. [13] :

<Interaction>::=<Communication><message>

<Communication>::=<id><from><to><mode>

<Message>::=<type><priority><contents>

- **Module d'interprétation des messages**

Il vérifie l'absence d'ambiguïtés dans le message (possibilité de satisfaction s'il s'agit d'une requête, etc.) et transmet ses conclusions au module de prise de décision.

- **Module de prise de décision**

C'est le noyau de l'agent. Ce module prend une décision en fonction de ce qui lui sera retourné par le module d'interprétation de messages. (Ex : lancer le traitement en cas de non ambiguïté et de possibilité de satisfaction de la requête de l'émetteur ou reporter le traitement dans le cas contraire).

- Il détecte les situations de non-coopération qu'il transmet au module de *résolution de cas de non-coopération*.

- Il met à jour les croyances : la tâche de conception consiste à faire évoluer, pour chaque agent, sa base de connaissances (ajustement des paramètres de l'opérateur associé) et les liens d'échanges avec les autres agents pour obtenir une organisation satisfaisante c'est-à-dire la configuration d'un graphe d'opérateurs dont l'exécution produit un résultat satisfaisant l'objectif. Cette évolution se traduit par la variation du degré de croyance (*cf.* §4.1.2) en fonction des retours de l'utilisateur qui réagit aux résultats présentés et d'interactions internes, entre agents.

- Il sélectionne les agents fournisseurs : l'agent fait appel à tous les agents qui peuvent produire la donnée dont il a besoin en entrée, sélectionne une des données proposées, d'après ses croyances.

- Enfin, ce module met à jour les accointances (en entrée et en sortie) dans le cas de création d'un nouvel agent (ou de changement des croyances) en fonction des messages qui lui sont transmis par le module d'*interprétation de messages*, suite à son interaction avec les autres agents.

- **Module de résolution des cas de non-coopération**

Ce module assure la résolution des problèmes de non-coopération:

- *Inutilité* : production d'une donnée qui, finalement, n'est pas utilisée par les autres agents. L'inutilité est détectée lorsque les liens (degrés de croyance) associés aux accointances de sortie (agent client) sont très faibles. Elle est prise en compte par le mécanisme d'ajustement des croyances.

- *Concurrence* : situation où deux ou plusieurs agents répondent à la même requête par des résultats identiques. L'agent sélectionne d'après les croyances et non d'après la qualité des données proposées. Il n'y a donc pas de concurrence à son niveau. L'agent informe les agents émetteurs de résultats identiques qu'ils sont en concurrence pour qu'ils puissent y remédier. C'est le mécanisme de rétro-propagation des évaluations de résultat qui, en faisant évoluer les croyances, permettra la concurrence entre agents.

- *Conflits* : accès de plusieurs agents à une ressource commune (modification d'une image commune), production de résultats contradictoires. Les accès à une même ressource (une donnée) par plusieurs agents visant à la transformer sont résolus par la création d'une donnée transformée propre à chaque agent.

- **Module de traitement**

Il contient, d'une part, les mécanismes de lancement de l'opérateur de traitement d'images contenu dans une bibliothèque de TI et, d'autre part, les mécanismes de contrôle d'exécution de cet opérateur. Il est déclenché par le module de *prise de décision*. Les résultats ainsi produits seront retournés à ce dernier. Celui-ci se chargera de générer la représentation de la transformation effectuée, dans le langage de description des données, puis de transmettre ces résultats au module de l'interface de communication qui, à son tour, les transmettra aux agents concernés, en fonction des entités produites par l'un et exploitées par l'autre.

- **Accointances**

Chaque agent connaît explicitement les autres agents avec qui il peut être en relation directe.
Une accointances est composée de six champs :

<id> : identifiant (ou signature) de l'agent qui permet de récupérer sont instance,
<inputs> : paramètres en entrée de cet agent qui fournissent la description de la donnée d'entrée et des paramètres de l'opérateur associé,
<outputs> : paramètres en sortie de cet agent qui fournissent la description de la donnée de sortie,
<belief> : degré de croyance attribué à cet agent qui varie en fonction des différentes interactions,
<type> : type de l'agent (cognitif, réactif) qui détermine le mode d'interaction,
<services> : services rendus par cet agent qui décrivent la transformation liée à l'opérateur.

La description des données d'entrée et de sortie ainsi que celles des services rendus par l'agent, est exprimée dans le formalisme décrit dans [10] et présenté au §3.3 pour modéliser le traitement d'images.

4.1.2 Fonctionnement

Une première maquette du *système de base* fondée sur les croyances a été proposée et développée [1] sur ce principe pour évaluer l'évolution des croyances en fonction des différentes interactions entres les agents du système. Dans un premier temps, nous avons simulé l'expression des requêtes de l'utilisateur et l'arrêt du système, par un artefact. Sur un problème expérimental, le système se comporte comme prévu : les opérateurs se configurent pour produire le résultat recherché et les croyances évoluent en fonction de la satisfaction ou non des requêtes.

Principe d'ajustement des croyances

Lorsque deux agents X et Z sont connectés, X étant fournisseur de Z, X possède une croyance C_{xz} dans sa capacité à répondre aux requêtes de Z. Inversement Z possède une croyance C_{zx} qui qualifie sa confiance dans les capacités de X. Soit une configuration où Z a deux fournisseurs X et Y.

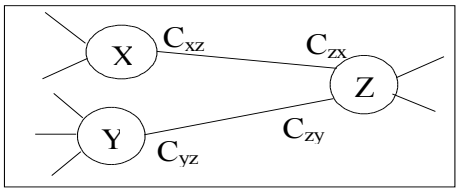


Figure III : interaction basée sur le degré de croyance.

Lorsque Z émet une requête, les agents concernés, ici X et Y, proposent leurs résultats. En fonction des croyances C_{zx} et C_{zy} et d'une pondération par un tirage aléatoire, l'agent Z sélectionne le résultat qui lui paraît le plus fiable. En fonction de ce choix, il met à jour ses croyances C_{zx} et C_{zy}. D'autre part, il communique ce choix à X et Y qui à leur tour réévaluent C_{xz} et C_{yz}. Si ultérieurement, ces données transmises par X ou Y s'avèrent non pertinentes, cette information sera propagée en retour, ce qui entraînera un deuxième ajustement des croyances.

Cependant, le comportement du système de base reste d'une part, combinatoire par manque de retours de l'environnement, et d'autre part, non évolutif (pas de modification du comportement des agents, à part les croyances, et pas de création de nouveaux agents).

Dans l'organisation du système présenté, nous proposons des solutions, en cours d'évaluation, pour introduire un nouveau comportement via le rôle du système secondaire.

4.1.3 Analyse et évaluation

Pour se comporter de manière non combinatoire, le système doit recevoir, de son environnement, des perturbations ou des contraintes qui vont orienter son évolution. Ces réactions doivent introduire dans le système deux types d'information : l'un portant sur les spécificités du traitement d'images et l'autre sur les connaissances du domaine d'application. Le premier type sera introduit par une modélisation du traitement d'images via un formalisme (*cf. §*5.1) de description syntaxique des transformations réalisées par les opérateurs. Le second résultera des interactions faites par l'utilisateur à la suite de l'interprétation qu'il fait des résultats intermédiaires et finaux.

Le système de base fait évoluer sa configuration c'est-à-dire la force des liens entre les agents, en suivant deux tendances :

- *Une tendance interne :* chaque agent cherche à être utile. Il évalue en permanence son niveau de coopération avec les agents auxquels il est lié et fait évoluer un degré de croyance pour éviter les situations de non-coopération. Globalement, cette tendance pousse le système à structurer et à simplifier (au sens de la schématisation) les données images, et à agglomérer les fonctions en modules plus complexes.

- *Une tendance externe :* qui dirige le système vers des objectifs à atteindre sous la forme d'instances de

concepts à produire. Ce système est orienté par le système secondaire pour découvrir des structures interprétables dans le domaine d'application.

4.1.4 Adaptation et évolution

Afin donner au système un comportement évolutif, on introduit un mécanisme de diffusion.

Le principe est le suivant : lorsqu'un agent reçoit une requête, il cherche à son niveau s'il dispose de tous les éléments pour satisfaire cette requête. Si ce n'est pas le cas, il émet à son tour des requêtes pour obtenir les entrées qui lui manquent. Lorsqu'il a pu déclencher le traitement, il répond à l'agent émetteur de la requête mais aussi à tous les autres agents qui pourraient exploiter l'information (ensemble des accointances en sortie).

Ainsi, un agent peut recevoir quatre types de messages :

- *Insertion* : prise en compte de l'arrivée d'un nouvel agent dans le système.
- *Requête* : demande de service.
- *Réponse à une requête* : réception d'une information demandée.
- *Information* : réception d'une donnée non sollicitée. L'agent détermine s'il peut déclencher son traitement. S'il ne peut pas, il émet des requêtes à son tour pour obtenir les entrées manquantes. Un agent qui reçoit une donnée en entrée doit informer l'agent fournisseur de l'élection de sa donnée ou non. Cela permettra à ce dernier de mesurer son degré d'utilité et de croyance. Cela se traduit par un message que nous appelons *commentaire de service fourni* dont les valeurs possibles pour le cas d'information sur le résultat reçu sont :

  - *élu*, son résultat à été utilisé par un autre agent,
  - *non-élu*, son résultat n'est pas pris en considération,
  - *élu-rejeté*, son résultat est pris en considération puis a été critiqué après des retours.

Ce mécanisme de diffusion permet d'activer des agents par l'évolution des données et non seulement par la résolution d'un objectif. On peut ainsi exploiter de manière beaucoup plus complète les possibilités du SMA et le faire évoluer en conséquence. Cette multiplication des enchaînement possibles sera également exploitée pour favoriser des créations de nouveaux opérateurs par agglomération (composition ou fusion).

4.2 Fonctionnement du système secondaire

Ce système est à l'étude. Nous présentons ici le fonctionnement du système secondaire dont nous établissons les spécifications.
Initialement, il dispose au moins de concepts basiques et très généraux de TI (ex : objet clair, rempli, forme allongée, etc.) applicables dans tous les domaines. Il fonctionne parallèlement au système de base qu'il décrit et il cherche des points stables dans les descriptions des entités produites. On peut considérer que ces stabilités détectées et leur description constituent le sens (artificiel) que le système construit en analysant l'image. Il leur correspond des macro-opérateurs capables de détecter dans l'image les concepts associés. Ceux-ci sont communiqués à l'interface pour être confrontés avec ceux de l'utilisateur. La recherche de ces points stables fait appel à des mesures portant sur les expressions des entités produites dans le système de base.
Le système secondaire manipule alors les concepts produits, par un mécanisme interne utilisant des opérations de composition, de spécialisation et de généralisation. Il génère ainsi de nouveaux concepts. Ce mécanisme interne cherche à maximiser la ressemblance entre les concepts appris et les concepts trouvés dans les productions du système de base. Enfin, il exploite les retours de l'utilisateur communiqués par l'interface pour donner des orientations au système de base.

5 Réalisation

5.1 Formalisme de TI

Une première implémentation a été réalisée dans le cadre d'un planifieur, ExTI[2] [9] et a montré l'utilité du formalisme dans la conception d'un système basé sur les indices visuels.
Dans sa forme actuelle, le formalisme a été implémenté par des schémas XML qui offrent une forte structuration du document et une grande souplesse d'évolution. En effet, nous avons utilisé Castor, un outil Java qui permet de générer une hiérarchie de classes fidèle aux schémas XML correspondants. Nous avons donc modélisé les différents schémas XML nécessaires à l'implémentation du formalisme, à savoir :

- LDC,
- LDD,
- Transformation,
- Types de données utilisés par les opérateurs,
- Opérateurs.

Exemple de schéma :

[2] ExTI (**Ex**pert en **T**raitement d'**I**mages).

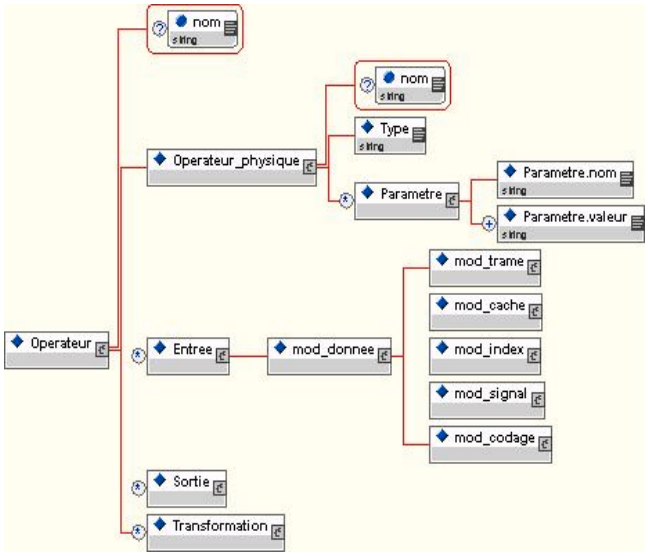


Figure IV : Schéma XML d'un opérateur.

Les opérateurs sont alors modélisés par une instance de ce schéma, c'est-à-dire un fichier XML. Cela dote le système d'une certaine souplesse d'utilisation et d'évolution. L'architecture du système de base n'est pas dépendante d'une bibliothèque d'opérateurs particulière, ni d'un type de données particulier et on peut donc personnaliser la famille d'opérateurs utilisée à volonté en fonction des besoins. Cela est visible sur le schéma avec le sous-champ « Type » du champ « Operateur_physique » et le sous-champ « mod_donnee » des champs « entree » et « sortie ». Les types de données manipulées par l'opérateur sont modélisés par des restrictions sur les cinq champs du modèle de données [11].

5.2 Système de base

Le système de base est implémenté par un système multi-agent où chaque agent encapsule un opérateur de la bibliothèque[3] de TI, fournie sous forme de fichiers exécutables, écrite en C et C++ et compilée. Chaque agent est un *thread* Java et il est composé de l'ensemble des modules décrits dans le modèle d'agents du §4.1.1.

Les agents sont initialisés à partir d'un fichier XML contenant la description d'un opérateur. Les champs des accointances (*cf.* §4.1.1) sont remplis avec les objets Java correspondant au fichier XML de l'opérateur ; la liste des agents atteignables en entrée ou en sortie est créée d'une manière dynamique en déterminant la compatibilité des types d'entrée et de sortie de l'agent avec ceux déjà présents dans le système. En effet, quand on ajoute un agent, il envoie un message de demande d'insertion (de type *INSERT*) à tous les agents du système ; à la réception de ce dernier, chaque agent examine la compatibilité de : (a) ses entrées avec les sorties du demandeur d'insertion ; (b) ses sorties avec les entrées de l'agent émetteur du message *INSERT*. S'il y a compatibilité, il sera inséré comme accointances en entrée dans le cas (a) et comme accointances en sortie dans le cas (b). Dans les deux cas d'insertion, l'agent émetteur du message *INSERT* sera informé de son insertion pour qu'à son tour il insère l'agent qui lui a répondu.

5.3 Système secondaire

Le système secondaire étant encore au stade des spécifications, nous simulons son fonctionnement par des artefacts de description et de génération. Pour permettre l'évolution du système, nous rendons les descriptions de concepts accessibles à l'utilisateur. Dans la version actuelle, celui-ci doit donc être capable d'interpréter et de modifier les expressions formelles des concepts.

5.4 Interface

L'interface fait l'objet d'une recherche séparée (en cours). Elle permet à l'utilisateur de parler naturellement des données (entités : pixel, région, amas de régions, etc.) et de leurs contenus [19]. Elle exploite le langage de description des concepts pour les présenter et les manipuler à l'aide d'indices visuels (forme, orientation, aspect, etc.). Elle permet de manipuler les données de façon interactive et de définir des concepts incrémentalement.

6 Conclusion et perspectives

La conception d'une application spécialisée de TI consiste à configurer le graphe des opérateurs adéquats. Cette configuration résulte de l'auto-organisation d'un système composé d'agents représentant les opérateurs de TI et ayant essentiellement une attitude sociale coopérative. Ce système s'organise et se stabilise en fonction des réactions provenant de son environnement. Nous avons réalisé une maquette du système de base et nous l'avons évaluée sur un jeu d'opérateurs. Nous avons simulé les retours de l'utilisateur par des artefacts. D'autre part, nous augmentons la compétence des agents, en les dotant de la capacité de manipuler formellement les descriptions symboliques des entités produites par ces traitements. Nous définissons alors un système secondaire,

[3] Actuellement le système utilise une bibliothèque propre et la bibliothèque Pandore disponible sur : http://www.greyc.ismra.fr/~regis/Pandore/

fonctionnant en miroir du système de base, pour élaborer ses propres concepts. Dans cette approche, on ne construit pas une application en demandant à un système de résoudre un problème complètement spécifié. Au contraire, l'application résultera de l'adaptation d'un système dialoguant avec l'utilisateur pour mettre ses propres concepts en adéquation avec ceux du domaine étudié.

Références

[1] Y. Abchiche, Conception d'une plate-forme pour la configuration d'un graphe d'opérateurs de traitement d'images par système multi-agent, DEA Informatique de l'Image et du Langage, UPS Toulouse, France, 1999.

[2] O. Boissier, Y. Demazeau, MAVI: A Multi-Agent System for Visual Integration, IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems, Las Vegas, Nevada, USA, pp. 731-738, October 2-5, 1994.

[3] A. Boucher, Une approche décentralisée et adaptative de la gestion d'informations en vision, Thèse de Doctorat en Informatique, Université Joseph Fourier, France, 18 janvier 1999.

[4] O. Capdevielle, P. Dalle, Image processing chain construction by interactive goal specification, First IEEE ICIP, Austin, Texas (USA), pp. 816-820, November 1994.

[5] S. Chien, H. Mortensen, Automatic image processing for scientific data analysis of a large image database, IEEE Trans on PAMI, vol 18, no 8, pp. 854-859, 1996.

[6] R. Clouard, A. Elmoataz, C. Porquet, M. Revenu, Borg: a knowledge-based system for automatic generation of image processing programs, IEEE Trans on PAMI vol 21, no 2, pp. 128-144, February 1999.

[7] D. Crevier, R. Lepage, Knowledge-Based Image Understanding System: A Survey, Proceeding of Vision and Image Understanding, vol. 67, No. 2, August, pp. 161-185, Article no IV960520. Academic Press 1997.

[8] P. Dalle, Y. Abchiche, Agents de Médiation pour la Conception de Systèmes de Traitement d'images, Colloque ALCAA (Agents Logiciels, Coopération, Apprentissage & Activité humaine), Biarritz, France, pp. 08-20, 6-8 octobre 2000.

[9] P. Dalle, P. Dejean, Planification en Traitement d'images : Approche basée sur les données, 11ème RFIA Clermont-Ferrand, France, pp. 75-84, tome II, janvier 1998.

[10] P. Dejean, P. Dalle, Image analysis operators as concept constructors, SwSIAI'96, San Antonio, Texas, USA, pp. 66-70, April 1996.

[11] P. Dejean, P. Dalle, Modèle symbolique de la donnée de traitement d'images, 10ème RFIA, pp. 746-755, Rennes, France, janvier 1996.

[12] P. Dejean, P. Dalle Un langage de description de concepts pour la formulation d'objectifs d'analyse, ORASIS, Pôle Vision du GDR-PRC CHM, Clermont-Ferrand, France, pp. 219-224, 20-24 mai 1996.

[13] Y. Demazeau, O. Boissier, J.L. Koning, Interaction protocols in distributed artificial intelligence and their use to control robot vision system, Proceedings of the 6th International Conference on Artificial Intelligence Control of Robots, Smolenice, Slovaquia, World Publishing Company inc, Singapore, pp. 17-30, September 1994.

[14] T. Drummond, T. Caelli, Task-Specific Object Recognition and Scene Understanding, Computer Vision and Image Understanding, 80, pp. 315-348, 2000.

[15] L. Germond, M. Dojat, C. Taylor and C. Garbay, A Cooperative Framework for Segmentation of MRI brain scans Artificial Intelligence in Medicine 20: pp. 77-94, 2000.

[16] J-M. Inglebert, Planification automatique de chaînes de traitement d'images, Thèse de Doctorat en Informatique, Université Paul Sabatier, Toulouse, France, 1992.

[17] T. Matsuyama, V. Hwang, SIGMA: a framework for image understanding, Proceedings of the 9th IJCAI, pp. 908-915 1985.

[18] S. Moisan, R. Vincent, M. Thonnat, Program supervision: from knowledge modeling to dedicated engines, Rapport INRIA no 3324, 1997.

[19] A. Nouvel, P. Dalle, Description des entités du traitement d'images pour la conception interactive d'applications, ORASIS, Cahors, France, pp. 349-358, 5-8 juin 2001.

[20] A. Nouvel, P. Dalle, Une approche interactive de définition d'ontologie image, 13ème RFIA, France, janvier 2002.

[21] Équipe SMAC, Une théorie de la coopération pour l'auto-organisation des systèmes artificiels, Rapport no 97-58-R, IRIT-UPS, Toulouse, France, décembre 1997.