

ICAR: Interaction and Classification by Rubine

SOFTWARE EXPLANATION

User ID: 504426

Software name: ICAR

Author: Jérôme Salut, University Toulouse III

PURPOSE

This work uses the algorithm of Dean Rubine [1] for specifying gestures and give an concrete implementation. The design of this implementation is oriented to re-usability in any well-known operating systems.

MOTIVATION

ICAR is a free java implementation of Rubine algorithm. The main motivation concerns rapid prototyping of post-WIMP applications by Master Degree students.

RESULT

The software is proposed as an AWT java component in a JAR file and can be used with standard Java JDK without any supplementary installation.

Tools necessary to install and use

- Java JDK 1.3 or superior (<http://www.java.sun.com>)
- Ivy-java 1.6 library or superior (<http://www.tls.cena.fr/products/ivy>)

TECHNICAL SUMMARY

ICAR consists for the designer, in an AWT component.

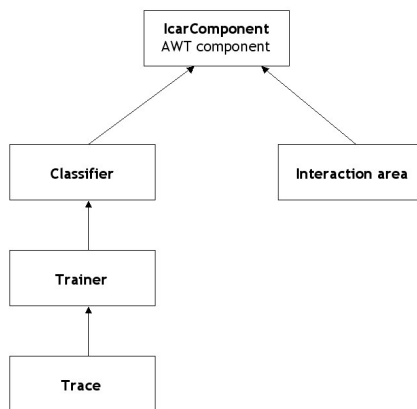


Figure 1: ICAR architecture

Actually, this component is divided in two parts (cf. Figure 1):

- one for trace acquisition. Each gesture is represented here by several points $P_i (X, Y, t)$
- and the second for classification. These two parts are connected by event communication.

INSTALLATION MANUAL

System requirements

ICAR may be used on all java sun virtual machine (JVM)

Hardware requirements

We don't recommend minimum requirements for running ICAR component.

Disk storage: less than 1.5 Mo

Installation

Successfully installing and running ICAR requires these basic steps:

1. Install a Sun JRE or JDK for your operating system
2. Download an ivy JAR on the appropriate website
3. Uncompress the file icar-1.0.zip to one directory (you can use winzip on Windows or unzip on unix systems)

HOW TO USE ICAR?

To use ICAR, you must add icar-1.0.jar to your CLASSPATH.

For example, type:

```
csh: % setenv CLASSPATH ${CLASSPATH}:/path/icar-1.0.jar
```

```
bash: % export CLASSPATH=${CLASSPATH}:/path/icar-1.0.jar
```

```
DOS: > set CLASSPATH %CLASSPATH%;C:\path\icar-1.0.jar
```

For compilation, type:

```
% javac -classpath /path/icar-1.0.jar My-class.java
```

For execution, type:

```
% java -classpath /path/icar-1.0.jar My-class
```

Note: for recompilation, simply type make on Unix systems.

SCREENSHOTS

Here are several screenshots of proposed applications issued from ICAR system.

ICAR administration console allows designers to build new or modify dictionaries (cf. Figure 2).

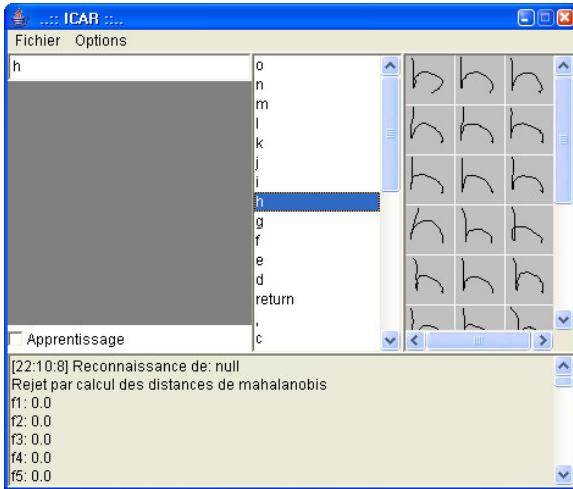


Figure 2: ICAR administration console

The proposed ICAR client implementation consists only in a frame where the ICAR AWT component is inserted. The Text area is inserted to control recognized gesture (cf. Figure 3).

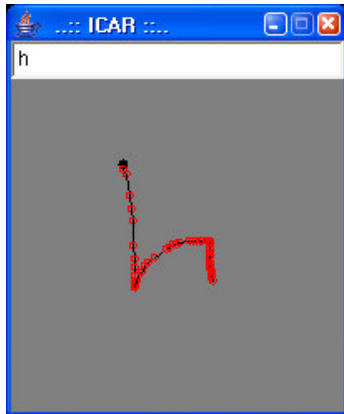


Figure 3: ICAR client

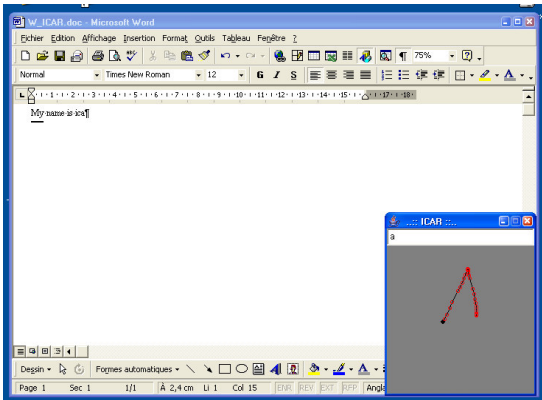


Figure 4: ICAR client connected to Microsoft Word

ICAR can be connected to third party software. An example is provided which use the ivy middleware [2] (cf. Figure 4.).

DOCUMENTATION

Documentation is provided in the ICAR archive in directory javadoc. To build documentation, simply type make doc on Unix systems.

EXAMPLE

To use ICAR in own implementation, insert IcarComponent component in a frame et just wait for events!

```
...
// component creation
IcarComponent zoneIcar;

zoneIcar = new IcarComponent("dictionary_name.dat", 200, 200);

// Mode Training mode desactivated
zoneIcar.setTrainingMode(false);

// event listener
zoneIcar.addIcarRapportAnalyseListener(this);

// add to main frame
myFrame.add(zoneIcar);

// implement "IcarRapportAnalyseListener" interface

public void processIcarRapportAnalyseEvent(IcarRapportAnalyseEvent event) {
    if (event.getMotif() != null) {
        System.out.println(
            event.getMotif());
    }
}
...

```

ICAR produces on standard output recognized gestures relatively to the current dictionary.

It is possible to load ou save different dictionaries.

REFERENCES

1. Dean Rubine, "specifying Gestures by Example", in Computer Graphics, volume 25, Number 4, July 1991.
2. Buisson M., Bustico A., Chatty, S., Colin F-R., Jestin Y., Maury S., Mertz Ch., Truillet Ph., ivy : un bus logiciel au service du développement de prototypes de systèmes interactifs, Proceedings of the 14th French-speaking conference on HCI, Poitiers, pp. 223-226.