Assisting Designers in Developing Interactive Business Oriented Applications

Jean Vanderdonckt
Université catholique de Louvain, Institut d'Administration et de Gestion
Place des Doyens, 1 - B-1348 Louvain-la-Neuve, Belgium
Phone: +32-(0)10-47 85 25 - Fax: +32-(0)10-47 83 24
E-mail: vanderdonckt@qant.ucl.ac.be, vanderdoncktj@acm.org
URL: http://www.qant.ucl.ac.be/membres/jv/jv.html

1 Introduction

Up to some recent years, many software tools, techniques and methods have been developed to systematically produce parts or whole of a user interface for a target interactive application. For example, software tools that automatically generate a working user interface have been demonstrated feasible and operational in the domain of interactive business oriented applications. These days, there is a shift of focus (Vanderdonckt 1996) from automated generation of user interfaces (where the process is blindly driven without human intervention) to computer-aided design of user interface (where the task analyst, the designer, the developer, the evaluator are more active in the production process by deciding options, by driving the process, and by being assisted by software tools and guided by methods in this process).

This paper will primarily focus on assisting designers in a particular task involved in the global development life cycle: the presentation design. A visual, argumented, manipulable and computer-aided technique is presented to assist designers in cooperating with the system to build a first sketch of the presentation. This technique has been implemented in the SEGUIA tool (Système Expert Générant une « User Interface » de manière Assistée).

2 The SEGUIA Presentation Generator

From a **descriptive** viewpoint, SEGUIA consists in a software that automatically produces the presentation of a Windows-based user interface of a business ori-

ented application (e.g., data management, database systems, office automation). This user interface is composed of a main application window with child windows and dialog boxes. The main application window comes with a menu bar and pull-down menus and the other compound objects are filled with traditional interaction objects (e.g., edit box, list box, combination box). Since the produced user interface is workable, it can be shown, executed and tested by anyone. SEGUIA is the software that supports the presentation design in the TRIDENT methodology to develop highly interactive business oriented applications (Bodart *et al.* 1995).

From a **technical** viewpoint, it is an expert system built on top of the AION/DS expert system shell marketed by Trinzic Software (Trinzic 1993). This expert system shell initiates an inference engine capable of processing rules contained in a knowledge base in a forward or backward chaining approach. Here, the forward chaining is exploited to process ergonomic rules, sometimes called guidelines, (Smith and Mosier 1986) to progressively produce a final user interface.

From a **methodological** viewpoint, SEGUIA follows a model-based approach (Puerta, 1997; Puerta, 1999) to derive a presentation from two models: a data model augmented with meta-information (practically, an entity-relationship model detailed with additional properties for attributes) and an activity-chaining graph model (practically, a model which specifies how the data flow is regulated among semantic functions belonging to the semantic core of an application).

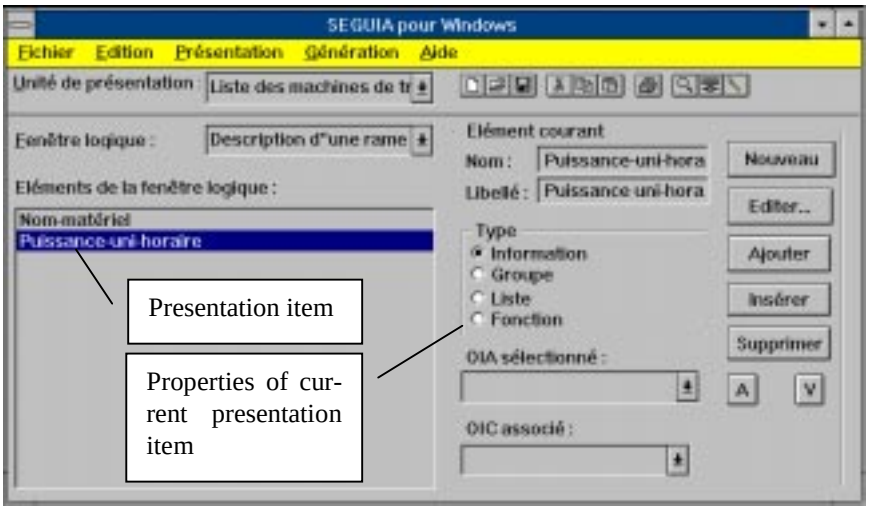

Figure 1. The SEGUIA environment showing properties of a presentation item.

From a **procedural** viewpoint, the attributes of entities and relationships, along with their meta-information (fig. 1), are first considered to select appropriate Abstract Interaction Objects (AIOs). Since AION/DS allows the dynamic creation of objects, each selected AIO is mapped onto a physical Concrete Interaction Object (CIO) of the Windows environment. Then, the activity chaining graph and other properties are analyzed to derive the positioning, the size and the arrangement of individual CIOs into a compound CIO (e.g., a dialog box or a child window). Two placement algorithms have been developed for this purpose. The processing of each algorithm produces the physical value of previously created CIO, such as location, height and length, color, font, selection properties, default values and so on (fig. 2).

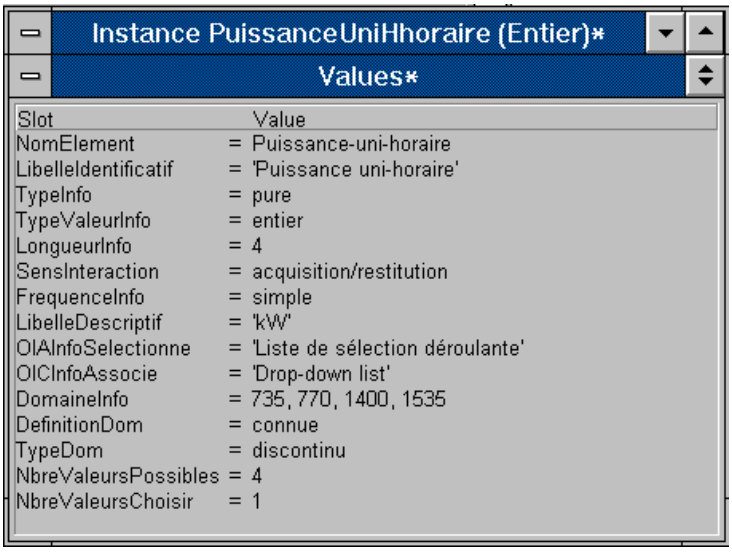

Figure 2. Generated properties for a presentation item with AIO and CIO.

Finally, this dynamically created user interface can be edited in a direct manipulation graphical editor to edit any property of generated CIOs. After that, the final results are converted and stored as Windows resource files to be used in a Windows-based development environment (e.g., Borland Delphi or Microsoft Basic V5.0).

3 Automatic versus Computer-Aided Design

The generation process can be executed in two ways:

1. in an automated way: the designer is supposed to provide the complete information required by the starting models (i.e., the entity-relationship model

and the activity-chaining graph) and to launch the generation within SEGUIA. This process is therefore considered as *single phased* (the designer launches the generation by selecting the appropriate menu item in SEGUIA) and *blind* (the designers does not see any intermediate results, only the final presentation can be finally seen);

2. in a computer-aided way: the designer is supposed to progressively input and refine the information required by the starting models so that the software can take advantage of information as it is provided by the designer. The software then interacts with the designer to progressively produces an intermediate solution for both selection of AIOs and positioning of CIOs. This process is therefore considered (fig. 3) as *three phased* (an automatic generation of a starting proposal from existing information, a phase of progressive refinement where the designer is presented with several options to decide from, and a phase considering the task context by exploiting domain properties), *visual* (the current working presentation is generated on the fly and presented during generation to the designer), and *justified* (each design option is accompanied with some argumentation explaining what the advantages and the inconveniences are on the usability). This last justification is based on the guidelines that just served to generate the ongoing presentation (Vanderdonckt 1999).

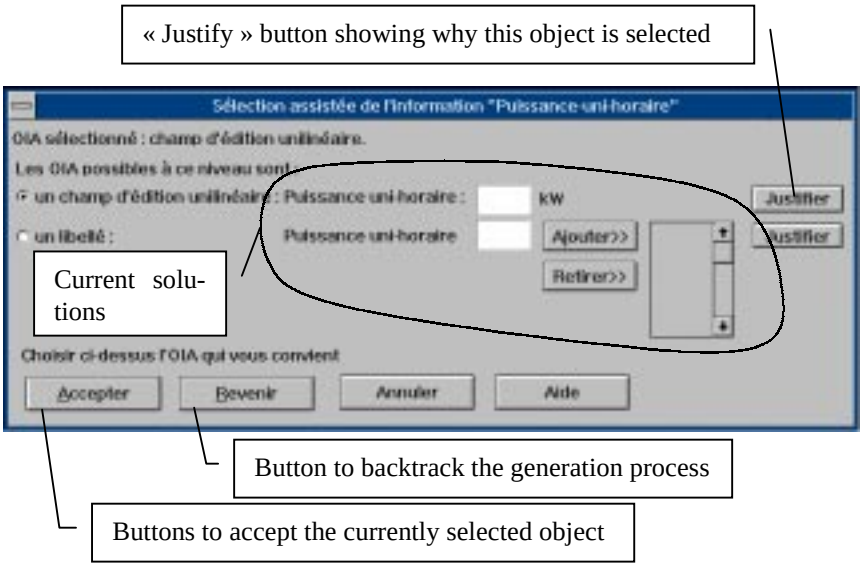

Figure 3. Computer-aided selection of an Abstract Interaction Object.

4 Conclusion

The big win of SEGUIA is that any change in the underlying models can be automatically reflected into a similar change in the generated presentation. Having the underlying models fully specified, an automated generation requires an average time of 15 minutes, mostly due to the manual operations. A re-generation of a previously generated presentation takes at most 10 minutes. The proposed approach can therefore reduce the cost of producing a first presentation, but also cuts the time every time the models are evolving. Of course, the final presentation should be interpreted as a first sketch to be further adapted to user's needs. If time and resources are insufficient, this first proposal is a workable user interface that can be incorporated in a final Windows application.

It seems that designers are appreciating automated generation and computer-aided design of such a presentation unequally. When resources are tight, the "blind" process is launched; when time constraints allow some flexibility or when the designer wants to be more involved in the process, the computer-aided process is preferred. This conclusion seems to confirm the trend according to which fully automated generation of user interface tend to disappear and to be replaced by a designer-controllable process, which is more desirable (Vanderdonckt 1996).

5 References

Bodart, F., Hennebert, A.-M., Leheureux, J.-M., Provot, I., Vanderdonckt, J. & Zucchinetti, G. (1995). Key activities for a development Methodology of Interactive Applications. In Benyon, D. & Palanque, Ph. (Eds): *Critical Issues in User Interface Systems Engineering*, pp. 109-134. Berlin: Springer-Verlag.

Smith, S.L. & Mosier, J.N. (1986). *Design guidelines for the user interface software*. Technical Report ESD-TR-86-278 (NTIS No. AD A177198). Hanscom Air Force Base (Massachusetts): U.S.A.F. Electronic Systems Division.

Puerta, A.R. (July-August 1997). A Model-Based Interface Development Environment. *IEEE Software*, 14(4), , pp. 41-47.

Puerta, A.R., Cheng, E., Ou, T., & Min, J. (1999), *MOBILE: User-Centered Interface Building*. In Proc. of ACM Conference on Human Aspects in Computing Systems (CHI'99, Pittsburgh, May 1999), New York: ACM Press.

Trinzic Corp. (1993). *AION Development System (ADS) V6.04*, General Reference. Palo Alto: Trinzic Corporation.

Vanderdonckt, J. (Ed.) (1996). *Computer-Aided Design of User Interfaces, Proc. of the 2nd Int. Workshop on Computer-Aided Design of User Interfaces* (CADUI'96, Namur, June 5-7, 1996). Namur: Presses Universitaires de Namur.

Vanderdonckt, J. (1999). Development Milestones towards a Tool for Working with Guidelines. *Interacting with Computers*, 11 (4), to appear.