Bringing Interaction Specifications to HCI Design Patterns

Maíra Greco de Paula
Departamento de Informática
R. Marquês de São Vicente, 225
Gávea – Rio de Janeiro – RJ
Brazil – 22453-900
+55 21 3114-1500 ext 3323
mgreco@inf.puc-rio.br

Simone Diniz Junqueira Barbosa
Departamento de Informática
R. Marquês de São Vicente, 225
Gávea – Rio de Janeiro – RJ
Brazil – 22453-900
+55 21 3114-1500 ext 4353
simone@inf.puc-rio.br

INTRODUCTION

Architectural patterns were originally defined in a narrative style, enriched by relevant contextual information [Alexander 1977]. They described the motivation and rationale from the point of view of the persons who would use the "architectural product" described by the pattern. Software design patterns, on the other hand, are defined in a more structured format, and the motivation is typically described from the point of view of system components [Gamma et al. 1995]. HCI design patterns have tried to bring together the advantages of a more structured description, which eases reading and scanning, and the richness of contextual information [van Welie 2001]. However, they fall short of directly supporting design, in that they provide information about good design solutions, but not a tool to model the resulting application using them.

One of the major advantages of software design patterns as defined in [Gamma et al. 1995] is that each pattern includes one or more diagrams that may be put together into a blueprint of the application, i.e., a software specification. In this paper, we propose to follow a similar approach and use a notation to represent HCI patterns that may be used to *specify* user-system interaction.

REPRESENTATION LANGUAGE REQUIREMENTS FOR INTERACTIVE APPLICATIONS

Some HCI patterns include diagrams that look like storyboards, representing the concrete user interface without irrelevant details. However, many interesting patterns are related to *interaction paths* that span beyond individual interface elements. For these patterns, we need a representation language at a higher level of abstraction, and which focuses on the possible interactions users may have with the system. Differently from the diagrams in software design patterns, the interactions should be represented from the user's point-of-view, i.e., without unnecessary details about the system's internal functionality. Moreover, the language should support the representation of patterns that aren't specific to a single user interface style or environment, so the pattern will not be unnecessarily limited.

We propose the use of a representation language for interaction specification, which can be used in HCI design patterns. A partial interactive design solution would fill the "diagram" slot proposed by Alexander and his colleagues.

Some application-specific interactions cannot be captured in patterns. An interaction representation language should be able to organize and compose both specific and pattern-derived interaction specifications into an application "blueprint". This will help designers not only build applications faster, but also developing a more coherent whole.

As suggested by [Granlund et al. 2001], it may also be interesting to create task patterns. However, such patterns are usually domain-specific, and do not point directly to interactive solutions.

A SEMIOTIC ENGINEERING VIEW OF INTERACTION

Semiotic Engineering is a theory of HCI which views the interface as a designer-to-users message, representing the designer's solution to what he believes are the users' problems, needs, and preferences [de Souza 1993; de Souza, forthcoming]. In this message, he is telling users, directly or indirectly, what he had in mind when he conceived the application: "This is who I think you are, what I think you want to do, how and why; what I think you need and prefer; the system I have built for you and how you can or should use it.". We believe that, if the designers' role as communicators is supported more extensively, users should be better able to understand and use the designed artifact. So, we need a set of tools that empower the designer, supporting his reflection about the interactive solution being conceived.

In this perspective, interaction design is concerned with building a coherent and cohesive message, in such a way that it strives to maximize the chances that the message will be interpreted by users as meant by the designer. In other words, interaction design may be viewed as conversation design. This kind of conversation is unique, because the designer is no longer present when it occurs (during interaction). Instead, he builds into the application interface a communicative agent, called the *designer's deputy*. This agent may appear explicitly, as an interface agent, or may be implicitly expressed by the user interface, appearing in the form of labels, messages, choice of widgets and colors, and so on. It is thus the designer's responsibility to build into his deputy the spectrum of conversations it will be able to carry out with users.

A POSSIBLE REPRESENTATION LANGUAGE FOR INTERACTIVE APPLICATIONS

In previous work, we have presented MoLIC (Modelling Language for Interaction as Conversation), a language based on Semiotic Engineering for representing interaction [Barbosa et al. 2002]. We briefly summarize here the basics of this language.

In MoLIC, interaction is represented by conversation stages named scenes, in which it is the users' turn to "talk". Some of their utterances may cause the conversation to transition to another scene, and some of them trigger or require a response from the system. System processes are represented by black-boxes, to emphasize that users will only get to know what is happening inside the system via the deputy's utterances. Figure 1 illustrates the interaction in a common search task.

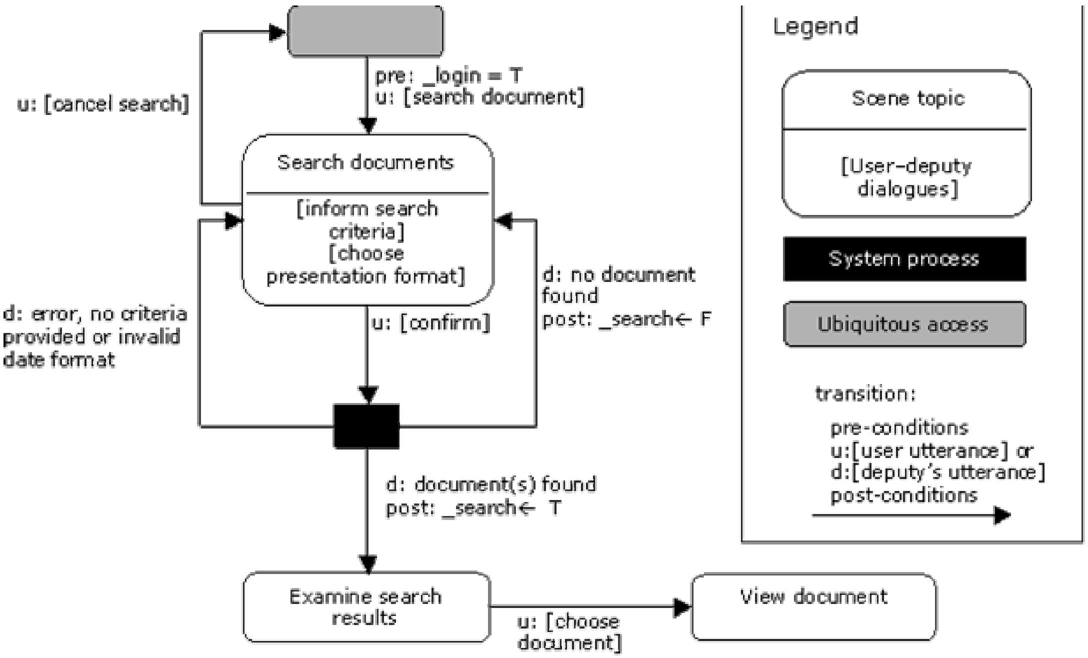

Figure 1. Sample diagram using MoLIC.

MoLIC helps designers acquire a global view of the application they are conceiving, from a user's standpoint. The diagrammatic version of MoLIC does not include user interface details that belong to any specific scene; this important issue has been addressed by other HCI-related studies, such as storyboarding, which is already used in certain HCI pattern languages.

CONCLUDING REMARKS

HCI patterns should attempt to bring together the advantages of both architectural and software design patterns: they should include not only rich, contextual information and design instructions, but also actual HCI design specifications that, together with application-specific specifications, compose the entire application's HCI design.

The design solutions represented graphically in current HCI patterns are mostly based on storyboards. We have argued for the need of a language in a higher level of abstraction that is able to represent interaction (and not just pieces of the user interface). We propose to use MoLIC as an interaction modelling language for diagrammatically representing higher-level HCI design patterns.

We are currently evaluating the use of MoLIC for representing HCI design patterns. Another interesting issue is to assess the usefulness of MoLIC and HCI design patterns for evaluating existing products.

REFERENCES

Alexander, C. The Timeless Way of Building. Oxford University Press. New York, NY. 1979.

Barbosa, S.D.J., de Souza, C.S., de Paula, M.G., Silveira, M.S. Modelo de Interação como Ponte entre o Modelo de Tarefas e a Especificação da Interface (in Portuguese). Proceedings of V Simpósio Brasileiro sobre Fatores Humanos em Sistemas Computacionais, IHC 2002. Fortaleza, CE. 2002.

de Souza, C.S. The Semiotic Engineering of User Interface Languages. Internacional Journal of Man-Machine Studies, 39, 753-773. 1993.

de Souza, C.S. The Semiotic Engineering of Human-Computer Interaction. Forthcoming.

Gamma, E., Helm, R., Johnson, R., Vlissides, J. Design Patterns. Addison-Wesley. 1995.

Granlund, A., Lafrenière, D., Carr, D.A. A Pattern-Supported Approach to the User Interface Design Process. Proceedings of HCI International 2001. New Orleans, USA. 2001.

Van Welie, M. Patterns for Designers? Position Paper in CHI 2002 Workshop on "Patterns in Practice". Available online at http://www.welie.com/patterns/chi2002-workshop/index.html