Promoting a Separation of Concerns via Closely-Related Interaction and Presentation Models

Bruno Santana da Silva, Otávio A Martins Netto, Simone Diniz Junqueira Barbosa
Departamento de Informática, PUC-Rio
R. Marquês de São Vicente, 225 / 4º andar RDC
Gávea, Rio de Janeiro, RJ
Brasil, 22451-900
{brunosantana, onetto, simone}@inf.puc-rio.br

ABSTRACT

A common goal of HCI design processes is to provide an organized set of procedures and representations to develop high-quality interactive systems. However, we find large gaps between representations in existing approaches, and some representations do not have a clear scope or focus. This paper proposes to use a set of closely-related notations to address different sets of design issues in successive stages of design, aiming to achieve a clear separation of concerns. In particular, it focuses on the transition from an interaction model to an abstract presentation model.

Keywords

abstract presentation model, interaction model, interaction design, separation of concerns

INTRODUCTION

Many design processes have been proposed in the literature for defining the activities of analysis, design, evaluation and implementation of user interfaces [6, 8]. The common goal of such processes is to provide the designer with an organized set of procedures and representations to develop high-quality interactive systems. The quality of use is determined by attaining the users' goals, meeting their expectations, desires, values, skills, and capabilities. Although design processes share a common high-level goal, each process is composed of a different selection and organization of design activities, where each activity is supported by different representations.

One of the problems with existing approaches is due to large gaps between representations. Such gaps require many heterogeneous decision-making processes to move from one representation to other, i.e. decisions that involve different sets of concerns. This violates the purpose of models itself: to provide focus and abstraction, addressing a reduced set of concerns and thus making the problem space more manageable [9].

In this paper, we propose to use a set of closely-related notations to address different sets of design issues in successive design stages. The differences in the notation used at each stage reflect the scope and level of detail necessary to deal with different sets of concerns. However, care was taken to minimize the notational differences, so as not to hinder learning and use. We have explored this idea in various case studies in different interaction styles and application domains. Our preliminary findings reveal that the intended separation of concerns can in fact be achieved by following the proposed approach.

The following section briefly describes some models and representations used in a typical human-computer interaction (HCI) design life cycle and argues that existing representations should be complemented in order to better segment HCI design issues. In the next two sections, the paper describes how the gap between task models and interface design may be bridged by a family of closely related HCI design notations. The fourth section describes a notation to represent the user interface presentation at a low level of detail, and the following section illustrates a case study in which the proposed solution is applied. The paper concludes with some remarks on the perceived benefits of this unified approach and points to future research directions.

REPRESENTATIONS AND MODELS IN AN HCI DESIGN PROCESS

Within human-computer interaction (HCI), the star life cycle [8] was proposed as an iterative user interaction design process centered on usability evaluation. Figure 1 presents the star life cycle, annotated with representations that are usually employed at each stage.

The shaded rounded rectangles in Figure 1 represent the stages in the life cycle, and which can be performed in any order, always followed by an evaluation. Designers usually start from the analyses stage and proceed clockwise. The

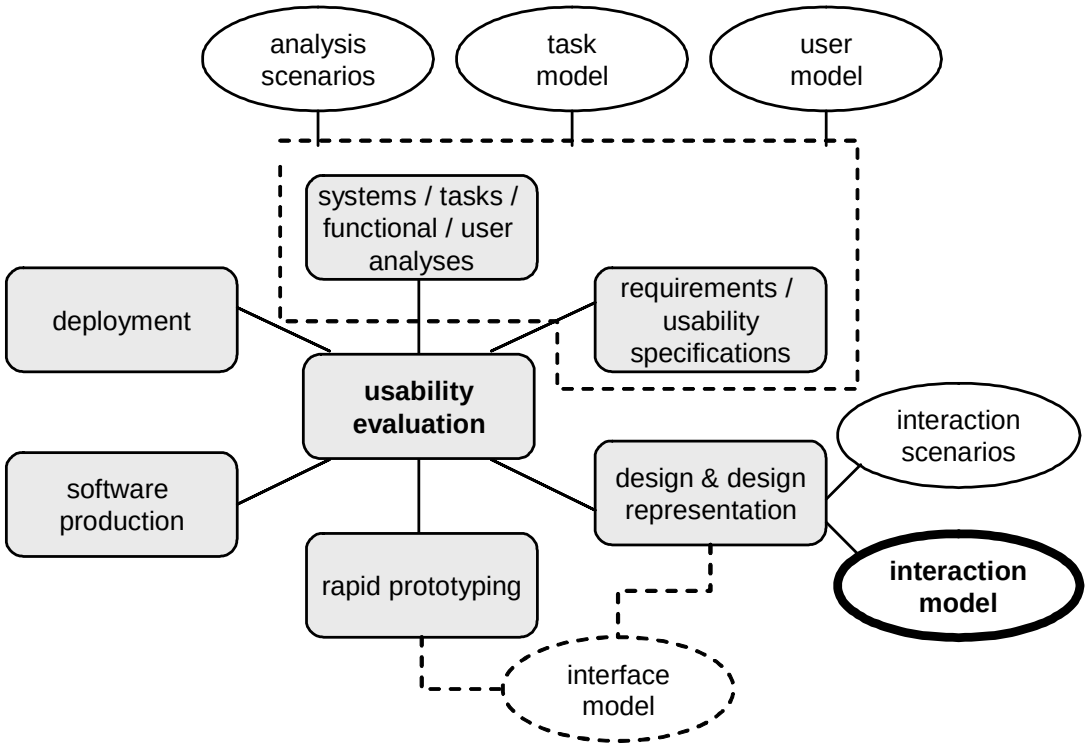


Figure 1. The star life cycle for user interaction development, adapted from [8] and including some representations and models commonly used at each stage.

ellipses in the figure indicate models and representations that are usually employed in each stage of the process.

Scenarios are typically the first representation used in an interaction design process. Scenarios may be defined as narratives that "project a concrete description of activity that the user engages in when performing a specific task, a description sufficiently detailed so that design implications can be inferred and reasoned about" [2, p.4]. Scenarios describe real data, users, and processes. They may include users' roles and motivation, their organizational setting and context, goals and associated tasks, and the users' interpretive processes. The main goal of using scenarios in the analyses is to identify the user tasks and tasks sequences that will be supported by the system. Scenarios may be complemented by task models, which organize and structure the tasks that users will need to perform and detail the steps to achieve the corresponding user goals. The way in which tasks will be designed and executed must be adequate to the user profiles captured during the analyses stages and represented in the user model.

Dialogue or interaction models are used at the design stage, with the goal to define and specify user-system actions and dialogues that will take place during interaction. In other words, they describe how the interaction should occur. As such, the interaction model represents the user-system communication processes, and not necessarily (at this stage) the detailed and concrete user interface elements of the final product.

In the rapid prototyping stage, interface models may be used to specify the user interface elements and their spatial (or temporal) arrangement. Abstract user interface models have been increasingly used as a basis for user interface generation for multiple devices and platforms.

It is important for a development process to promote smooth transitions between the design and development stages, in which the results and artifacts produced in one stage serve as a resource to the next. Looking into existing representations, we believe that there is still a gap between (abstract) design representations such as task models and user interface representations or prototypes. Addressing this gap is an important challenge for HCI researchers. It is precisely during the transition between these representations that many design decisions are made which will determine the final product's quality of use. Moreover, if left unrepresented, decisions will not be captured and will thus hinder maintenance, evolution and reuse efforts.

In order to bridge this gap, it may be necessary to introduce some intermediate representation(s), each one addressing a narrower range of issues and thus reducing the problem and solution spaces at each design stage. The problem in introducing new representations is the potential burden on designers, who may need to learn completely different notations and frequently switch from one to another during

design. To overcome this problem, one solution is to devise a family of models whose notations are closely related and yet allow designers to address few concerns at a time. By ensuring models have a specific scope and level of detail [9], the problem space is reduced and fewer issues need to be considered at each step. In addition, it is desirable that we have a set of transformation rules or heuristics that help us move from one model to the next.

Previous attempts have been made at bridging this gap by introducing new representations. One of the representations devised to achieve this goal is MoLIC [1], an interaction modeling language that explores the interaction-as-conversation metaphor. After elaborating scenarios and/or modeling tasks to understand the structure and context of the users' activities, and organizing the problem space, MoLIC models are built to explore the solution space, where designers represent how these activities will occur during interaction. We have investigated the use of MoLIC in several design projects, and found that the set of concerns we needed to deal with when moving from MoLIC to the user interface design was still broader and more heterogeneous than we had hoped. In an attempt to segment the problem space and to focus design activities at different levels of scope and detail, this paper proposes to unfold MoLIC in a family of closely-related design models, which do not require designers to learn multiple unrelated notations as they progress from conceiving the interactive solution to defining its abstract presentation.

MOLIC INTERACTION MODELS

MoLIC stands for "Modeling Language for Interaction as Conversation" [1]. It is an *epistemic tool*, in that its major goal is to support designers in reflecting about the interactive solution being conceived, while it's being conceived [13]. MoLIC is grounded on the Semiotic Engineering theory of HCI, which characterizes the user interface as a meta-communication artifact, through which designers communicate with users about how users may or should interact with the system [5]. The designer-to-user messages are conveyed by the *designer's deputy*, which is the spokesman for the designer at the user interface and with whom users will interact while using the application. MoLIC views interaction as a conversation between the user and the designers' deputy. In this perspective, MoLIC focuses on conversational content and structure, allowing to represent interactive solutions in a technology-independent notation. It is important to note that different technological environments will often encourage designers to produce different interactive solutions. However, having a common notation promotes documentation and reuse of interactive solutions across environments, making it easier to think about how differences between environments cause different design choices. MoLIC does not represent concrete user interface elements such as widgets and screen layout, nor the full *expression* of the user interface. Instead, it represents the *content* of the user-system interaction that will surface at the user interface, defining which communicative exchanges may/should take place (see [7] for an account of Hjelmslev's content-expression dichotomy).

MoLIC has been used in two stages during HCI design. At first, designers produce a high-level conceptual model, in which the *essence* of the interactive behavior is defined, by representing all the possible communication exchanges during interaction. At this stage, designers focus on user-system turn-taking and the definition of conversation topics, subtopics and dialogues. Designers are encouraged to plan for *conversation repair*, i.e., to define how *communicative breakdowns* should be avoided or dealt with. Figure 2a presents a small piece of a conceptual interaction model of a typical internet browser for the user goal "find in this webpage". In MoLIC interaction models, a scene (rounded rectangle) represents a user-deputy conversation about a certain topic. This conversation may comprise one or more dialogues and each one is composed of one or more user/deputy utterances organized in conversational pairs.

At this stage, MoLIC supports the reflection of multidisciplinary design team members on the conceptual design, promoting a balanced participation of professionals that hold various perspectives on the envisaged solution [11].In the next stage, the conceptual model is detailed to include the signs involved in each communicative exchange (in each dialogue). Throughout this text, the term *sign* is used to denote any element that belongs to the problem domain or is introduced by the application, and that will be interpreted by users. Users will attempt to associate meaning to the signs as related to their goals or tasks. Although the signs' concrete expressions aren't represented in MoLIC, their content is characterized, and some constraints are defined that will later support designers in selecting the appropriate expression for each sign. For instance, a sign may be characterized as holding one of a fixed set of possible values, which encourages the selection of one of the following widgets: set of radio buttons, single-select listbox, drop-down list, or similar. If the number of possible values is known, the sign characterization is made more precise and thus the set of candidate expressions (user interface elements) for the sign may be reduced. For instance, if there are only three possible values and there is enough display space, a set of radio buttons would seem a better candidate than the other widgets. As with any iterative process, it is natural to switch back and forth between the conceptual and detailed interaction models. Figure 2b presents the detailed interaction model of Figure 2a.

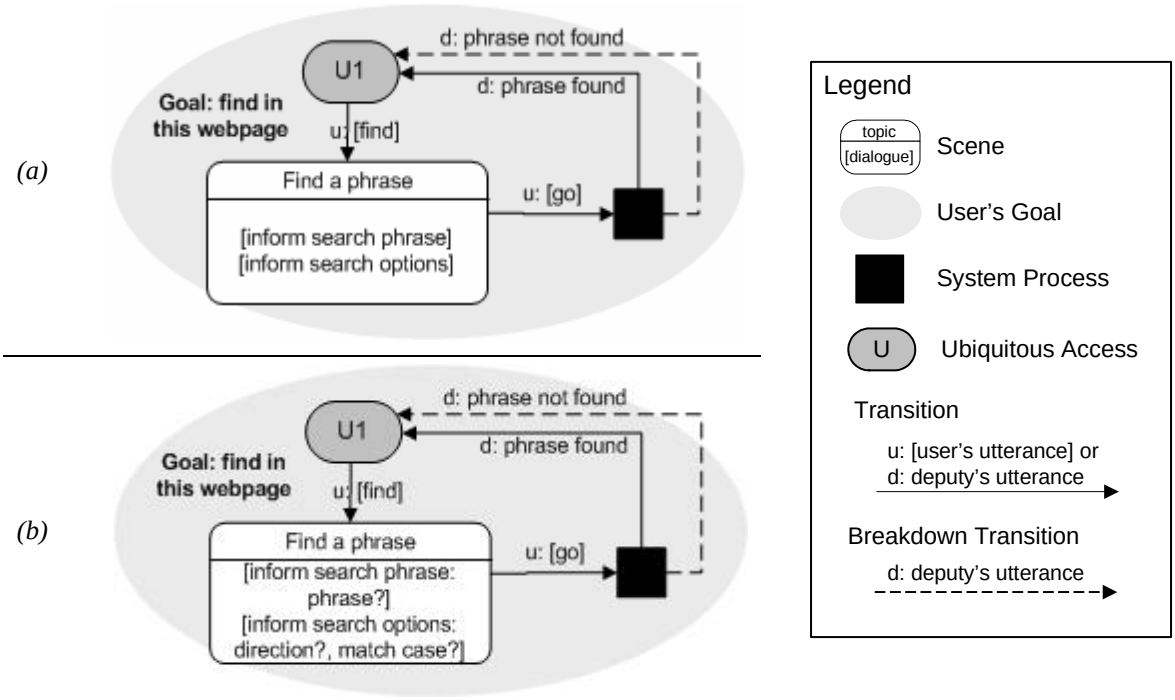


Figure 2. MoLIC conceptual (*a*) and detailed (*b*) interaction models for the user goal "find in this webpage".

FROM INTERACTION TO PRESENTATION

Originally, the detailed MoLIC representation was thought of as an appropriate resource for designers to define some of the concrete presentation concerns, such as widget selection and grouping (a step towards defining the spatial layout). However, our case studies have shown that specific platform considerations may require that the topic-subtopic dialogue structure be restructured in a different configuration of presentation units – forms, web pages, windows and the like – than the earlier presentation-independent, content-driven groupings, before proceeding to the more concrete presentation concerns. We believe that dealing separately with these structural concerns makes it easier for designers to reflect on commonalities and distinctions across platforms, even when based on a single conceptual solution.

For instance, a scene may be mapped onto one or more display units, such as webpages or windows, depending on the expected display size and resolution. Conversely, designers may decide to join two or more scenes in a single display unit, taking into account additional criteria, such as the users' preferences or need for efficient interaction. In our example, the interaction model (presented in Figure 2a and b) to "find in this webpage" may be mapped onto a single window (as in Mozilla Firefox ®) or in different windows (as in Microsoft Internet Explorer ®) as shown in Figure 3.

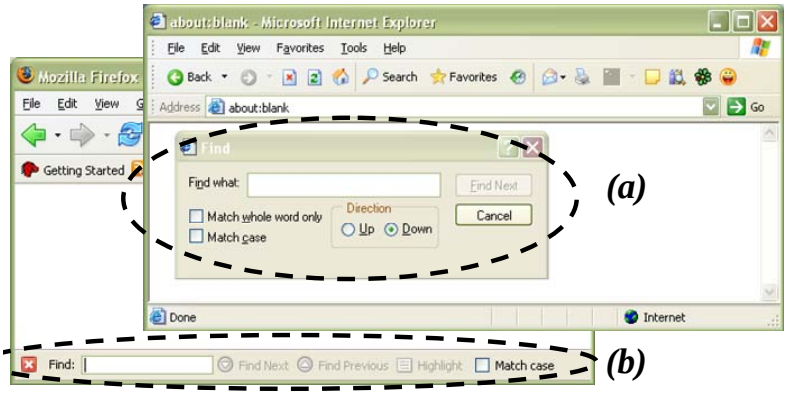


Figure 3. Some possible interfaces to achieve user goal "find in this webpage": Microsoft Internet Explorer® (*a*) and Mozilla FireFox® (*b*).

This way, we propose a clear separation of concerns at each design step, corresponding to different representations, as indicated in Table 1.

Existing presentation models

Various user interface models and representations have been proposed at different levels of abstraction. Constantine [4] proposes "canonical abstract prototypes" to define the characteristics of the user interface elements (without yet choosing platform-specific widgets) and their spatial arrangement within a screen. The notation was conceived to be mnemonic and the symbols may be articulated to represent a large number of abstract widgets. However, it is not clear how canonical abstract prototypes are derived from the previous representations, which makes traceability between representations a challenge. The detailed graphical nature of the representation may hinder the designers' understanding of the overall structure of each screen and where each screen fits in the whole set of possible interaction paths.

Some abstract user interface representations have been defined with the goal of generating the concrete user interface [12, 15]. They are usually derived from task models [3], and aim to automatically generate user interfaces for different platforms and devices, without the designer's interference. Paternò and co-authors allow designers to customize the user interface generation process [10], but it is not clear how well designers understand the consequences of this customization and whether they can foresee undesirable side effects. A number of XML-based user interface description languages (UIDLs) have been reviewed in [14]. UIDLs are usually targeted at automated processes, either for exchanging designs between development environments or for generating user interfaces. However, their textual nature may make it harder for designers to reflect on presentation solutions of a visual nature, and also their understanding of the overall context in which each element is placed, both spatially and temporally (with respect to the sequences defined by the designed interaction paths).

The focus of our work is to support human designers in their design activities, and not to formally specify or generate user interfaces. Instead of computer application tools, we focus on *epistemic tools*. As such, we believe that the intermediate step of presentation mapping presented in Table 1 provides structure and context to better support designers in defining the detailed presentation model.

However, we believe that the representation of the presentation structure should not introduce an entirely novel notation. Instead, it would be better to reuse the notations used in previous design steps as much as possible, extending them when necessary. In the next

Table 1. Separation of concerns promoted by different representations.

Scope	Representation	User interface/ interaction concerns
Interaction Design	conceptual interaction model	▪ user-system turn-taking ▪ system-initiated repair ▪ user-initiated repair ▪ patterns of interaction ▪ alternative and shortcut interaction paths
	detailed interaction model	▪ signs content and constraints ▪ breakdown prevention and repair tags (associated to scenes, system processes, dialogues or signs)
Interface Design	abstract presentation model	▪ presentation (scene-to-screen/form/web page) mapping ▪ dialogue and sign groupings and ordering
	detailed presentation model	▪ spatial layout ▪ widget selection ▪ platform-specific design details (e.g. the use of color, graphic elements, animation, sound, and so on)

section, we introduce an extension to the original MoLIC diagrams to play the role of this abstract presentation model.

MOLIC PRESENTATION DIAGRAM

The goal of the presentation diagram is to smooth the transition between the MoLIC interaction diagram and existing user interface representations, such as storyboards and UIDL specifications, as well as to provide the context in which each individual user interface element will be defined. It is important to introduce as few elements as possible to the MoLIC notation, so that designers who use MoLIC will easily and rapidly learn to produce this diagram.

The conversational context is provided by the presentation diagram through the representation of a blueprint that relates different presentation units connected by transitions, in a way analogous to the transition utterances in the interaction model. The presentation diagram also allows designers to represent how the interaction with a user interface element influences another. Moreover, the presentation diagram represents the actual topic-subtopic-dialog-sign structure, and may include some preliminary suggestions for spatial arrangement of these elements, when designing visual interfaces.

The Notation

Table 2 presents the notation elements used in MoLIC presentation diagrams. Only the first two elements – presentation unit and breakdown message issued by the designer's deputy – are entirely novel. They are meant as grouping elements that take into account the platform characteristics, in addition to the content segmentation already represented in the interaction model. For instance, when the target platform is a small-screen device such as a PDA or a cell phone, the dialogues in a scene in the interaction model may be mapped onto multiple presentation units in the presentation model. Conversely, if designers have segmented the conversation content in various scenes in the interaction model, they may now decide, for a larger-screen display, to join some of the original scenes in a single presentation unit, in the presentation model.

The remaining elements are very similar to the ones used in MoLIC interaction diagrams, to facilitate learning, use, and traceability. The graphic element that represents a scene now gains another dimension: its size should indicate the expected size of the corresponding element in the concrete user interface. It may be possible to also consider its relative position in the presentation unit as an indication for its relative position at the concrete user interface, but this is left for future studies.

Table 2. Notation elements used in MoLIC presentation diagrams.

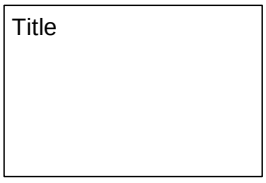
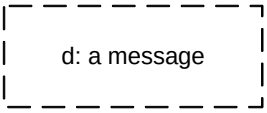
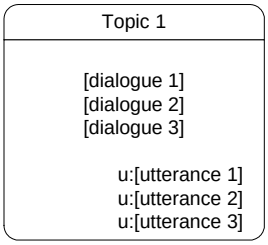
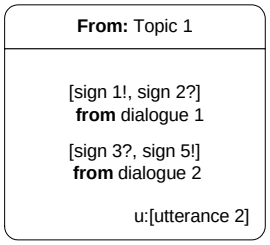
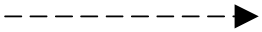
Graphic notation	Name and Usage
	**Presentation unit (screen, form, web page, conversation period, and the like)** Within a presentation unit the designers should represent which scenes, dialogues and signs will be presented. Each presentation unit should have an identifying title.
	**Breakdown message issued by the designer's deputy** This message may be embedded within a presentation unit or define a standalone presentation unit.
	**Scene** In addition to the dialogues represented in the interaction model, a scene in the presentation model encompasses the user transition utterances, represented by u:[utterance]. The relative size and position of the scene within a presentation unit may hint at envisaged characteristics of spatial layout to be ratified or revised at later stages.

Table 2 (cont'd)

Graphic notation	Name and Usage
	**Partially mapped scene or dialogue** When a scene is mapped onto more than one presentation unit, the source scene and the selected dialogues in the presentation unit must be represented. When a dialog is mapped onto more than one presentation unit, the source scene and dialogue, as well as the selected signs, must be represented. The keyword "from" indicates the partial nature of the mapping.
	Transition utterance When arriving at a presentation unit, it means a transition or navigation. When arriving at a scene, dialogue or sign, it means a change (of value) in the target element.
	**Breakdown transition utterance** Breakdown transitions arrive at a standalone presentation unit or at a breakdown message embedded within a scene.

One may note the introduction of a "partially mapped scene or dialogue". This element is an extension of a scene used to provide traceability information, i.e., information that allows designers to identify the corresponding element in the interaction model.

Interaction-to-Presentation Transformations

In a MoLIC interaction diagram, the structuring of dialogues and scenes is determined mainly by the criterion of topic coherence. The main goal there is to support the designer's reflection and understanding about the content and structure of the conversations that may take place between the user and the designer's deputy. When it comes to presentation, it is necessary to also take into account the devices' characteristics. This is still a reflective stage, because there are usually various possible mappings from a *conversation structure* (scene / dialog / signs) onto the interface structure (presentation units: screen, web pages, forms, etc.) where that conversation will occur. For instance, it is not possible to automatically infer whether a scene should be mapped onto one or more presentation units. This kind of decision must be consciously made by the designer when defining the abstract user interface.

The simplest transformation from an interaction model to a user interface model consists of mapping each scene to an individual presentation unit. However, this kind of transformation is not always suitable. It is important to encourage the designer's reflection on the interactive solution throughout the process.

In our case studies, a few transformations have been identified, and may be used as guidance in the construction of presentation diagrams. They may also help designers to reflect on how to map potentially ambiguous, inconsistent or incomplete interaction solutions onto a presentation diagram. Each transformation is described below.

Scene-to-presentation unit mapping. This transformation is a 1-to-1 mapping, where a scene composes a presentation unit. If the designer strongly considers the platform and device characteristics while building the interaction model, chances are that this is the only transformation necessary.

Mapping of the breakdown messages. Since breakdown utterances have already been defined in the interaction model, this transformation consists in defining where the corresponding breakdown message will occur in the user interface: within a presentation unit or scene, or as a standalone presentation unit, such as a dialog box.

Multiple-scenes-to-single-presentation unit mapping. In some cases, designers may decide to group one or more scenes (corresponding to different conversation topics) in a single presentation unit. This is usually the case when efficiency is an important usability goal and the topics are closely related and thus may be grouped under a single "supertopic" (Figure 4a).

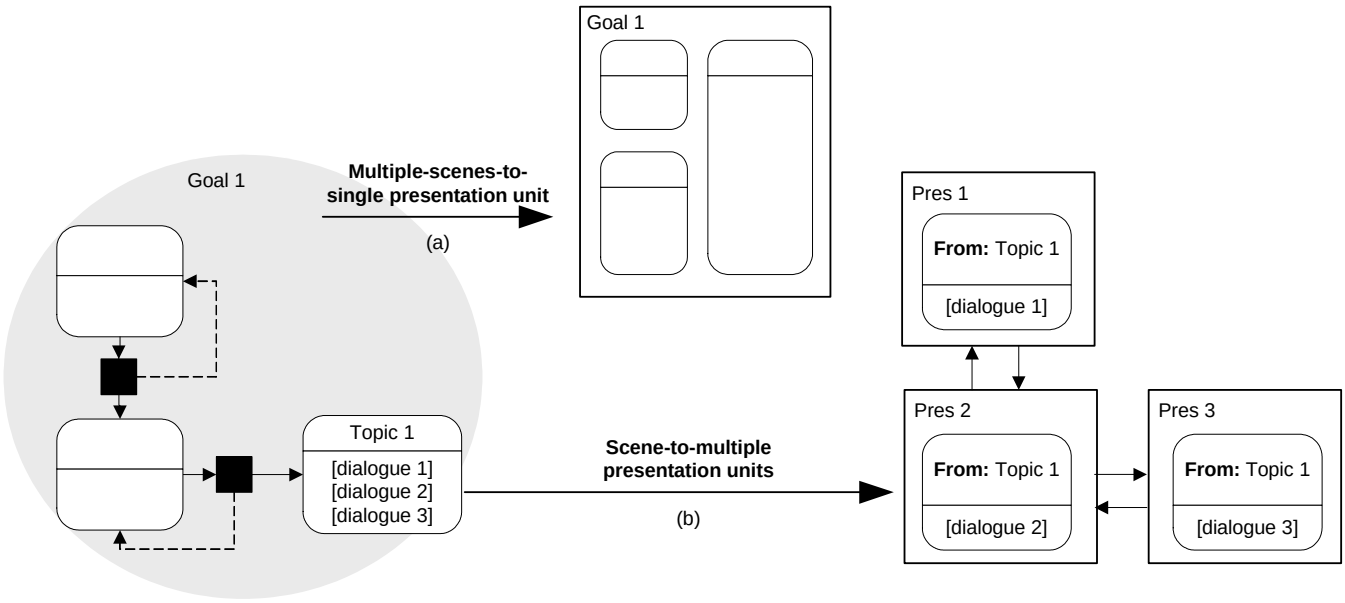


Figure 4. Examples of multiple-scenes-to-presentation unit (a) and scene-to-multiple presentation units (b) mappings.

Scene-to-multiple presentation units mapping. For complex scenes, or scenes whose dialogues may occupy a large portion of the presentation unit, designers may choose to segment (at the dialog level) a scene in the interaction model into two or more connected scenes in the presentation diagram (Figure 4b).

Dialog-to-multiple presentation units mapping. This case is similar to the previous one, but instead of separating dialogues across scenes, the mapping will separate sets of signs within dialogues into different presentation scenes. A common usage of this mapping is found in voice user interfaces. Instead of asking for more than one information in a single dialogue, each sign would be mapped onto its own presentation scene.

The notation presented here builds upon the MoLIC interaction diagrammatic notation and reuses the diagram elements as much as possible. The similarity between the interaction and presentation notations and the set of usual transformations make it easy to understand and build the presentation diagram from the interaction model. We believe that the transformations support the designers' reflection on the presentation of the conceived solution, helping the designer to understand the consequences of the design choices being made. The narrow scope of the decisions reduces the cognitive load on designers, who are able to analyze and discuss about high-level interface presentation issues before specifying or implementing the concrete user interface, thus achieving the desired degree of separation of concerns.

CASE STUDY

A number of case studies were developed in the course of this work, using MoLIC to develop the interaction model and the presentation notation proposed here to create the corresponding abstract presentation model. The main goals of the case studies were to investigate the limitations of interaction modeling with MoLIC, and whether the abstract presentation model proposed here promotes the reflection of the desired set of concerns, while avoiding concrete user interface decisions.

In this section, we present the interaction and presentation models of a hotel reservation system associated to a conference website. The system may be accessed by conference attendees, who inform the date period of their stay and then search for a hotel that has avaliable rooms during that period. Users may access the hotels location map, select a hotel to make a reservation, and inform the billing information. After providing all the necessary information, users are asked to confirm the reservation to make it final. An interaction model for this system is presented in Figure 5. It was built taking into account the topics and subtopics for the user-system conversation so that the user may achieve his goals. It didn't take into account, however, presentation aspects of web applications.

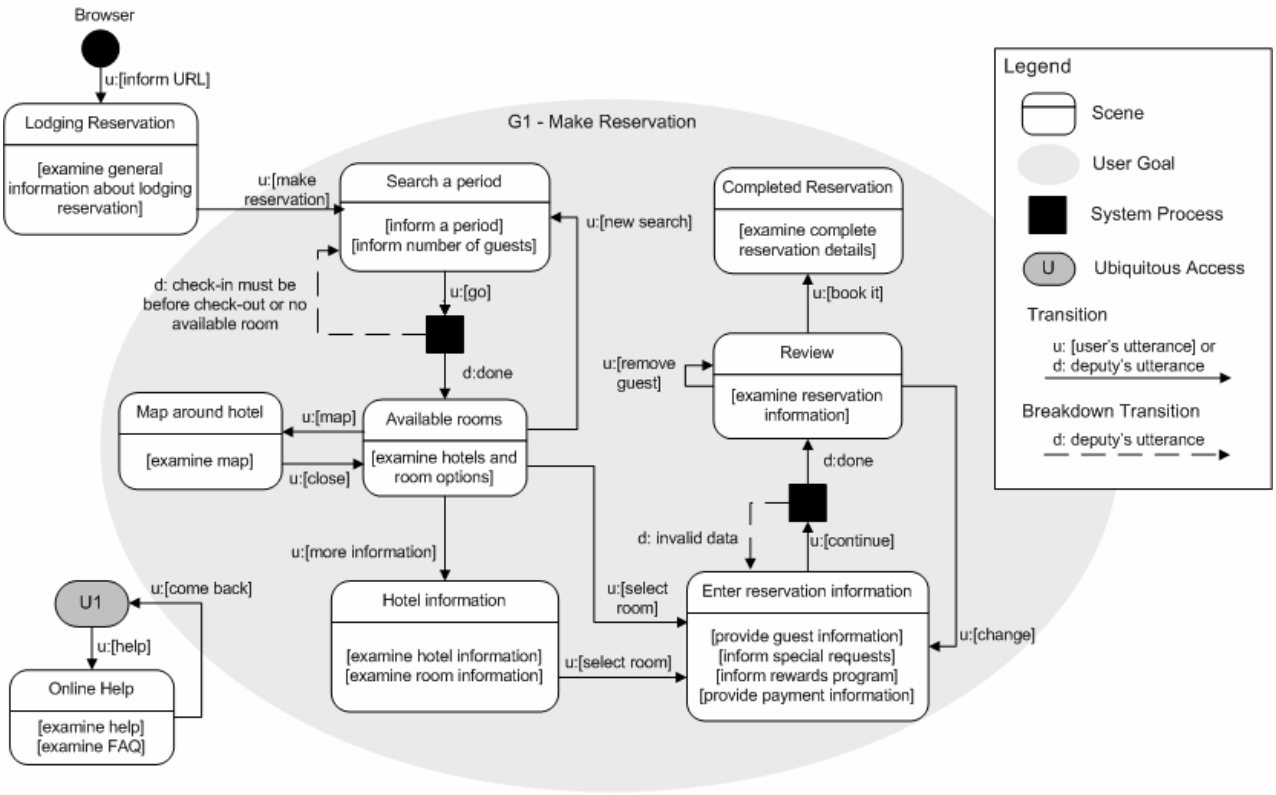


Figure 5. Interaction diagram for a hotel reservation system.

From the interaction diagram, some decisions were made regarding presentation, originating the presentation model shown in Figure 6. The mapping transformations were as follows:

(1) Scene-to-presentation unit mapping: The scene "Map around hotel" was directly mapped onto a presentation unit. This is the simplest mapping, when the topic segmentation designed into the interaction model remains valid when considering the presentation aspects of the corresponding scenes. It was kept as a separate presentation unit to help visualize the map using a large display space.

(2) Mapping of the breakdown messages: The breakdown message "d:invalid data" was mapped onto both a standalone presentation unit and as an embedded breakdown message within the "Enter reservation information" scene. The standalone presentation unit means that the designer wants to call the user's attention to the problems that occurred during the previous interactions, and he'll achieve this by presenting the breakdown in a separate webpage. The embedded breakdown message was also chosen to give users detailed information about the specific problems in the context where they can be corrected, usually accompanied by instructions on how to repair them.

(3) Multiple-scenes-to-single-presentation unit mapping: To favor performance, the scenes "Lodging Reservation" and "Search a period" were joined in a single presentation unit. This kind of decision is often made when designing web applications, where each transition may cause a costly delay in going back and forth the web server.

(4) Scene-to-multiple presentation units mapping: The scene "Search a period" was mapped onto two presentation units, characterizing a case of scene-to-multiple presentation units mapping. The designer decided to do this to increase the flexibility of the interaction, so that the user may make changes to the desired period of time whenever he wishes to do so.

(5) Dialog-to-multiple presentation units mapping: When deciding to include calendar controls to help users select the check-in and check-out dates, the designer realized that such controls for date selection would take up a lot of room in the page. Based on this realization, he further decided to map the dialog [inform a period] in the scene "Search a period" onto an additional presentation unit.

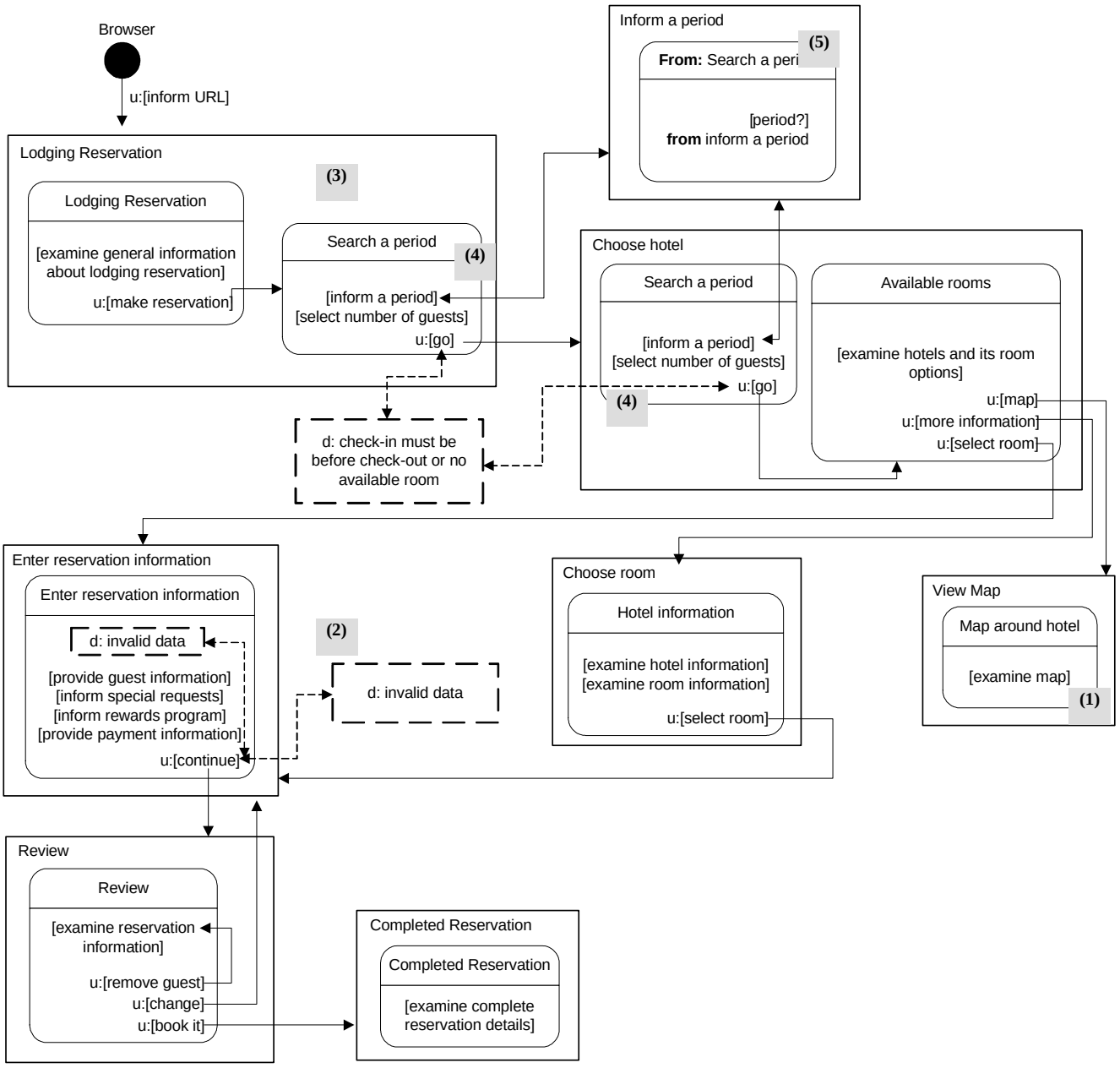


Figure 6. Presentation diagram for the hotel reservation system, with the interaction-to-presentation mappings indicated by the numbers in parentheses.

Alternative design choices

Let us reconsider item (4) of the described case study, "Scene-to-multiple presentation units mapping". The reason for duplicating the "Search a period" scene was to increase the flexibility of interaction, by letting the user achieve the corresponding goal whenever he wanted to. The designer could realize, at this stage, that including this scene in the "Choose hotel" presentation unit would sacrifice the legibility and aesthetic quality of this presentation unit. He might then decide to only present the dates and not provide a direct way of changing them at that moment. Figure 7 illustrates how the presentation model would look like in this case.

This example gives an indication that the proposed presentation model, taking the original MoLIC diagram as a resource, makes it possible for designers to reflect on presentation questions and reason about alternative design decisions.

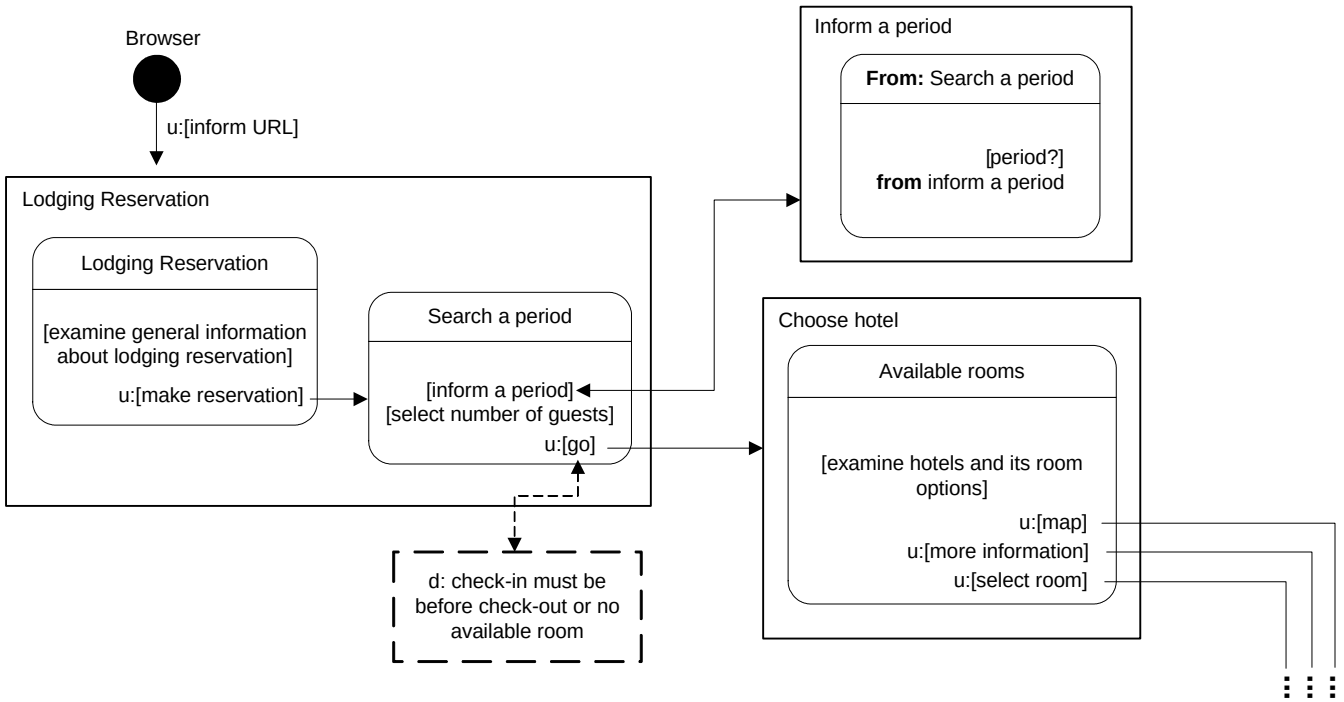


Figure 7. Alternative presentation model when privileging legibility over flexibility.

CONCLUDING REMARKS

This work proposes to use a family of representations to promote separation of concerns and smooth the transition between models and representations in HCI design. The ultimate goal is to promote a more focused reflection on a clear set of design issues at each stage of the design process. This paper described a presentation model that complements an interaction model by detailing how presentation units such as screens, windows, forms and web pages are internally structured, and how users will move from one unit to another during interaction. We have presented a few mappings that illustrate the decisions that designers make when planning the user interface based on the interaction model, but the set of possible mappings is not meant to be complete or universal.

Our preliminary findings are encouraging, but a number of issues remain to be investigated. We are currently working on a set of heuristics to better support this part of the design process and to help capture and document the underlying decisions. In particular, we need to conduct further studies to explore in which ways the chosen separation of concerns promotes reflection, and to assess if the chosen set of concerns is adequately captured in the presentation diagram. We want to investigate what kinds of presentation behaviors should be included in the representation, such as dealing with popup and modal windows, for instance. We are already exploring the concept of stereotypes to define repeating elements such as menus or navigation bars, and the definition of conditional slots to hold variable instructions or error messages. In addition, we are building a case base to compare the solutions represented in each model across domains, to identify patterns in the mappings between the models and to explore the ways in which the presentation model may determine the elements in the concrete user interface design.

Finally, we are conducting model based evaluation studies that focus on the application of formative evaluation techniques throughout the HCI design life cycle. We are investigating to what extent existing evaluation methods can be adapted to allow us to identify communication breakdowns into interaction and interface modeling using MoLIC.

ACKNOWLEDGMENTS

Bruno Santana da Silva thanks CAPES for the support to his Masters program. Otávio A Martins Netto thanks Faculdade de Minas (FAMINAS – Muriaé MG) for the financial support to his work. Simone Diniz Junqueira Barbosa thanks CNPq for her individual research grant. All authors thank their colleagues at the Semiotic Engineering Research Group for discussions leading to this work.

REFERENCES

1. Barbosa, S. D. J.; Paula, M. G. “Designing and Evaluating Interaction as Conversation: a Modeling Language based on Semiotic Engineering” In J.Jorge;

N. J. Nunes; J. Falcão e Cunha (eds.) Interactive Systems Design, Specification, and Verification – DSV-IS 2003, Funchal, Madeira Island, Portugal, June 2003, LNCS 2844 2003. pp. 16–33.

2. Carroll, J. M. (ed) *Scenario-based design: envisioning work and technology in system development*, New York, Wiley. 1995.
3. Clerckx, T. and Coninx, K. Integrating Task Models in Automatic User Interface Generation. *Technical Report TR-LUC-EDM-0302*, EDM/LUC Diepenbeek, Belgium, 2003.
4. Constantine, L. L. Canonical Abstract Prototypes for Abstract Visual and Interaction Design. *Reprint of keynote delivered at DSV-IS*, Madeira, Portugal, 4 June 2003.
5. de Souza, C. S. *The Semiotic Engineering of Human-Computer Interaction*. The MIT Press, 2005.
6. Dix, A.; Finlay, J.; Abowd, G.; and Beale, R.. *Human–Computer Interaction, 2nd edition*. Prentice Hall, Europe. 1998.
7. Eco, U. *A Theory of Semiotics*. Bloomington. Indiana University Press. 1979.
8. Hix, D.; Hartson, H. *Developing User Interfaces: Ensuring Usability Through Product and Process***.** John Wiley and Sons, 1993.
9. Hoover, S.; Rinderle, J. Models and Abstractions in Design. *Design Studies*. Volume 12, Number 4, October, 1991.
10. Paternò, F.; Santoro, C. A unified method for designing interactive systems adaptable to mobile and stationary platforms. *Interacting with Computers* 15 (2003) 349–366
11. Paula, M. G.; Silva, B. S.; Barbosa, S. D. J. Using an Interaction Model as a Resource for Communication in Design. *Procedings of CHI 2005, Extended Abstracts Volume*. April 2005. 4p.
12. Puerta, A.R. Issues in Automatic Generation of User Interfaces in Model-Based Systems. *Proceedings of Computer-Aided Design of User Interfaces, CADUI 1996*, ed. by Jean Vanderdonckt. Presses Universitaires de Namur, Namur, Belgium, 1996, pp. 323-325.
13. Schön, D. *The Reflective Practitioner: How Professionals Think in Action*. New York: Basic Books, 1983.
14. Souchon, N. and Vanderdonckt, J. A Review of XML-Compliant User Interface Description Languages. *Proceedings of the X Workshop on Design, Specification, and Verification of Interactive Systems, DSV-IS 2003*. Madeira, 4-6 June 2003.
15. Vanderdonckt, J. Automatic Generation of a User Interface for Highly Interactive Business-Oriented Applications, *Readings in Intelligent User Interfaces*, M.T. Maybury & W. Wahlster (eds.), Morgan Kaufmann, San Francisco, 1998, pp. 516-520.