Graphs of Models for Exploring Design Spaces in the engineering of Human Computer Interaction

Alexandre Demeure, Dimitri Masson, Gaëlle Calvary

Graphs of models for exploring design spaces in the engineering of Human Computer Interaction

Alexandre Demeure, Dimitri Masson
Laboratory of Informatics of Grenoble
655 Avenue de l'Europe
38330 Montbonnot-Saint-Martin, France
+33 (0)4 76 51 48 54
firstname.lastname@inrialpes.fr

Gaelle Calvary
Laboratory of Informatics of Grenoble
385, rue de la Bibliothèque - B.P. 53 - 38041
Grenoble Cedex 9, France
+33 (0)4 76 51 48 54
gaelle.calvary@imag.fr

ABSTRACT
Model Driven Engineering (MDE) has focused on the latest stages of the design process so far and as a result has missed the opportunity to foster creativity in the early phases. Our research aims at stretching MDE all over the design process including the creative phases so that to go beyond the well-known 'fast-food UIs' limit of MDE. We propose to consider sketches and prototypes as models. This paper claims for storing these models in a graph so that to both inspire designers and support adaptation at runtime.

Keywords
Model based User Interfaces, graph of models, design spaces, creativity.

ACM Classification Keywords
H.5.2 [**Information interfaces and presentation**]: User interfaces – *prototyping.*

INTRODUCTION
Early phases of User Interfaces (UI) design require the production of numerous propositions so that to result in a successful design [1, 11]. Those propositions are usually explored through sketches and prototypes that quickly materialize designers' ideas as a support for discussion, selection and validation. Whilst those early phases are crucial for good design, we observe that currently Model Driven Engineering (MDE) sustains the latest stages of design only (i.e., when the code of the concrete UI is produced). This can be explained by the historical grounding of MDE that comes from software engineering. Those approaches aim at proposing optimal solutions for a given problem in a particular context (e.g. SUPPLE [5]) but not at sustaining human creativity. As a result, MDE seems to be pushed at its limits [2]: advanced UIs or aesthetic UIs seem to be out of range.

We believe that the relative disappointment with regard to MDE is due to this lack of support of early phases. In this paper, we propose to consider sketches and prototypes as models to support the exploration of numerous ideas. We store these models in a graph that makes explicit the relationships between models. This graph and the related exploring tools are currently work-in-progress.

RELATED WORKS
Buxton [1] and Tohidi [11] elicit sketching and prototyping as key for creative designs whatever the domain is. Buxton [1] stresses that the value of sketches does not lie in the produced artifact itself (the drawing) but in its ability to trigger the desired and appropriate behaviors, conversations and interactions. Indeed, sketches are a vehicle, not a target: designers do not draw sketches to depict ideas that are well consolidated in their mind. Rather, they draw sketches to try out vague and uncertain ideas. When seeing the sketches, designers can spot problems they may not have anticipated. Even more, they can see new features and relations among elements that they have drawn. Some of them were not intended in the original sketches. These unintended discoveries promote new ideas and refine current ones.

Tools exist to help designers to sketch and prototype UIs. A simple yet quiet efficient example is a pen coupled with a sheet of paper. However, paper based sketches are not really appropriate to describe interaction. In some cases, this shortcoming can simply be overcome by using animated GIF. More generally, electronic tools such as SILK [6] or DENIM [8] have been developed to enable designers to quickly specify the interaction directly from sketches. Other tools such as SketchiXML [3] enable the designers to sketch a UI that is then interpreted as a set of UsiXML widgets. However, the set of widgets is not extensible (i.e. a brand new widget can not be added), which is a strong limitation for creativity.

SEMAIS'11, Feb 13 2011, Palo Alto, CA, USA

Demeure [4] explored semantic graphs for storing and reusing UI components both at design time and runtime. Masson [9] investigated genetic algorithms as a support for exploring possible UIs for a given task by assembling UI components that correspond to the (sub)tasks and tasks operators. However, in both cases, the components were formally described. Thus sketches and prototypes were not taken into account which dramatically limits the design space exploration.

STRUCTURE OF THE GRAPH OF MODELS

As in [4], we propose to organize the UIs' models in a graph but enriched with informal models such as sketches and prototypes.

Nodes of the graph

Nodes of the graph are UIs' models defined at one of the CAMELEON levels of abstraction: Concepts and tasks (C&T), Abstract UI (AUI), Concrete UI (CUI) and Final UI (FUI). Each node is enriched with a level of precision. This level ranges from "rough sketch" to "formal definition", covering all levels of fidelity in prototyping.

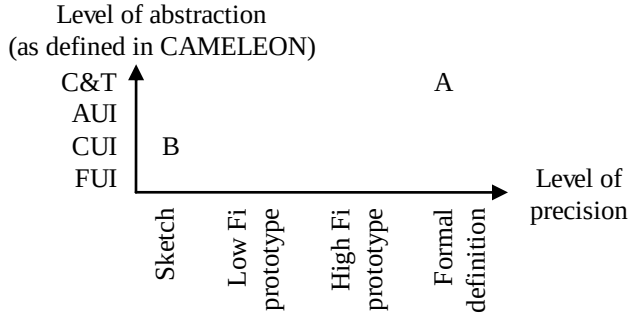

Figure 1: Nodes are characterized by a level of abstraction and a level of precision. A and B are two samples detailed below.

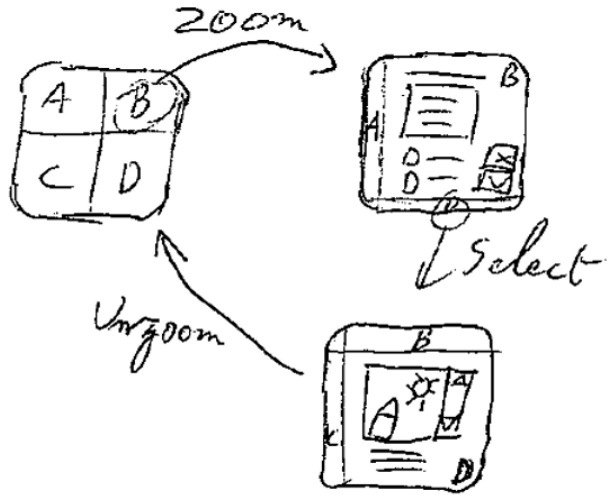

Figure 2: An example of Point B in Figure 1: the node is the interleaving task operator. It is defined at the CUI and Sketch level.

Point A in Figure 1 may correspond to a formal definition of the interleaving task operator. Such a definition could be based on CTT [10]. More concrete descriptions of this operator could be provided. For instance, point B is a concrete description of this operator but at a sketch level of precision only. Figure 2 provides an example of such a CUI-Sketch definition.

Arcs of the graph

The arcs of the graph model the relationships between UI models. Arcs can be seen as transformations that produce target UI models from source UI models. A transformation is defined by:

- A level of precision ranging from informal to formal;
- The context of use (in terms of platform, user and environment) the transformation requires;
- A degree of originality that conveys how much the know-how expressed in the arc is spread over designers: is it shared by the whole HCI community, or just by a part of it? This attribute gives designers clues on how well established or how innovative the transformation is.

Figure 3 illustrates a possible classification of transformations. This classification goes beyond usual transformations that are limited to the levels of abstraction they manipulate (Abstracts and Concretizes). Thanks to our classification, transformations can also be used for:

- Changing the level of precision of UI models (e.g., providing a formally defined UI model from an informal prototype).
- Making the composition of a UI model explicit (e.g., a task tree is composed of subtasks and task operators).
- Expressing that a UI model is another version of another one. This can be useful for knowing that UI alternatives exist.

Overall, transformations are a means for expressing the design rationale of an evolution in the design process.

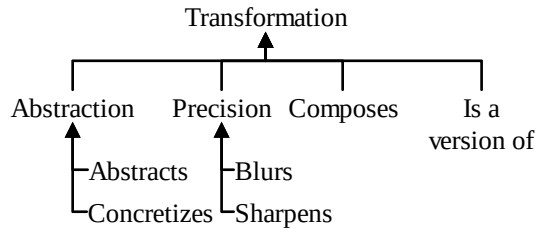

Figure 3: Classification of transformations.

SEMAIS'11, Feb 13 2011, Palo Alto, CA, USA

EXPLOITATION OF THE GRAPH OF MODELS

This section develops how powerful the graph is to support evolution both at design time and at runtime. Figure 4 is used to support explanation.

At design time

At design time, the graph serves two purposes: 1) to inspire designers by capitalizing the know-how in UI design, and 2) to provide a space to store and access UIs produced by designers during the design process.

The graph provides a means for designers' teams to structure their production of sketches and prototypes. Relationships between the UI models can embed the design rationale of the design process (the motivations of the design choices). For instance, in Figure 4, a project starts by a sketch of a C&T description. Neither the tasks nor the concepts are well defined, but stakeholders agree on an informal description of the project. Then this description is sharpened to a formal C&T model (here a CTT model). Nodes C, D, E, F and G describe one possible design evolution: from the C&T model, designers explore two paths: C followed by E and G, in parallel with F. C is more thoroughly explored. Several design versions are proposed and explained. The last version (G) sharpens parts of the design.

The graph stores the evolutions, discussions, and choices along with their rationale. Thus designers can later on go back to understand where an idea comes from, or start a new branch while keeping memory of alternatives. Indeed, different parts of the design may evolve at different places in the graph, or along different paths. In a same node, some parts can be highly detailed denoting a high level of confidence in the design choice, whilst other parts can still be roughly sketched (for instance node G in Figure 4 where only a part of the UI is sharpened).

Designers can select parts of a drawing and link them to other nodes, or parts of other nodes. For instance, designers can specify that one part of the C&T model represents the "Manage contacts list" and link it with the corresponding nodes. They can also link it to the circle part in node C. This possibility to identify parts of models is particularly useful when applied together with the "Composes" relationship. Designers can specify that a node is composed of several sub-nodes. In the case of a C&T model, sub nodes may represent sub tasks involved in the model. The "Composes" relationship makes it possible to split problems carried out by models into sub-problems. This is key for reducing complexity by finding, capitalizing and reusing solutions to smaller problems.

Designers can then explore possible solutions by assembling solutions of sub-problems together. As sub-problems can be decomposed in turn, this leads to a combinatory explosion and makes it impossible for designers to explore all of them. Thus one solution is to let the exploration of the combinations to search algorithms. Masson [9] proposed to use genetic algorithms to produce examples of UIs designs. Based on an external database that capitalizes widgets at several levels of abstraction (C&T to FUI), it takes a C&T model in input and produces a set of transformations to be applied on the C&T model to produce final UIs. However this approach focuses on widgets at a very high level of precision only. As a consequence, the generated UIs might not be suitable for early design phases. This approach can be extended to sketches and prototypes.

At runtime

Designers can rely on nodes and arcs at the formal definition level to propose automatic UI generators that can produce UI adapted to a given context of use. Indeed, for a given task, one can go through arcs and nodes to retrieve all possible implementations of this task. For each of these implementations, the path that links it with the original task informs about the context of use it is designed for. For instance, in Figure 4, one can follow the concretization arcs from the interleaving node to find all possible solutions to represent it. This process can be guided by the information about the context of use the node requires. By doing so, it is possible to retrieve all CUI/FUIs adapted to a given context of use. This was explored in [4]. It is related to a service broker devoted to HCI.

The graph, used as a service broker, could be integrated in automatic UI generation algorithms like SUPPLE [5]. The richer the graph is for a given task, higher the chance is to produce adapted UIs. Thus the openness and extendibility of the graph is key compared to closed or non explicit approaches that enumerate possible renderings for tasks or tasks operators. Actually, algorithms like SUPPLE [5] can be seen as a concretization arc in the graph that produces a CUI/FUI (at the formal definition level precision) based on a C&T description (at the formal definition level precision), a user model (his/her UI preferences, Fitts parameters and typical traces) and the targeted platform (widgets set and screen size). Applying SUPPLE to a particular task tree results in adding an arc in the graph starting from the node that embeds the C&T description to a node that describes the generated CUI/FUI. For instance, in Figure 4, SUPPLE can be applied to the C&T node that describes the instant messenger to produce a CUI (B in Figure 4) optimized for the platform P and user characteristics U.

CONCLUSION

Considering sketches and prototypes as models in MDE is promising to avoid the "fast-food UI" limit. It should enable UI designers to take advantages of these powerful approaches while taking benefit of the strong know-how HCI has in MDE.

We explore how capitalizing models in a graph can be useful both at design time and runtime to get inspired and

make the exploration of the design spaces easier. The implementation of this graph and the related exploration tools is currently work-in-progress. We plan to involve UI designers in their design as well.

REFERENCES

1. Buxton, B. *Sketching User Experiences: Getting the Design Right and the Right Design*. 448 pages, Morgan Kaufmann (March 30, 2007). ISBN-13: 978-0123740373.
2. Coutaz, J., *User Interface Plasticity: Model Driven Engineering to the Limit!*, in ACM, Engineering Interactive Computing Systems (EICS 2010) International Conference. Keynote paper. Pages 1-8. 2010.
3. Coyette, A., Faulkner, S., Kolp, M., Limbourg, Q., Vanderdonckt, J., *SketchiXML: Towards a Multi-Agent Design Tool for Sketching User Interfaces Based on UsiXML*, Proc. of 3rd Int. Workshop on Task Models and Diagrams for user interface design TAMODIA'2004 (Prague, November 15-16, 2004), Ph. Palanque, P. Slavik, M. Winckler (eds.), ACM Press, New York, 2004, pp. 75-82.
4. Demeure, A., Calvary, G., Coutaz, J., and Vanderdonckt, J. *The comets inspector: Towards run time plasticity control based on a semantic network*. In TAMODIA'06.
5. Gajos, K., and Weld, D. S. *Supple: automatically generating user interfaces*. In IUI '04: Proceedings of the 9th international conference on Intelligent user interface (New York, NY, USA, 2004), ACM Press, pp. 93–100.
6. Landay, J.A. and Myers, B.A. *Interactive sketching for the early stages of user interface design*. Proceedings of the SIGCHI conference on Human factors in computing systems, 1995.
7. Lee, B. and Srivastava, S. and Kumar, R. and Brafman, R. and Klemmer, S.R. *Designing with interactive example galleries*. Proceedings of the 28th international conference on Human factors in computing systems, 2010, pp. 2257--2266
8. Lin, J. and Newman, M.W. and Hong, J.I. and Landay, J.A. *DENIM: finding a tighter fit between tools and practice for Web site design*. Proceedings of the SIGCHI conference on Human factors in computing systems, 2000.
9. Masson, D. and Demeure, A. and Calvary, G. *Magellan, an Evolutionary System to Foster User Interface Design Creativity*. In proceedings of EICS'10, Berlin, 2010.
10. Paterno, F. and Mancini, C. and Meniconi, S. *ConcurTaskTrees: A Diagrammatic Notation for Specifying Task Models*. In Proceedings Interact'97, July'97, Sydney, Chapman&Hall, 1997, pp. 362-369.
11. Tohidi, M., Buxton, W., Baecker, R., and Sellen, A. *Getting the right design and the design right*. In CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems (New York, NY, USA, 2006), ACM, pp. 1243–1252.

SEMAIS'11, Feb 13 2011, Palo Alto, CA, USA

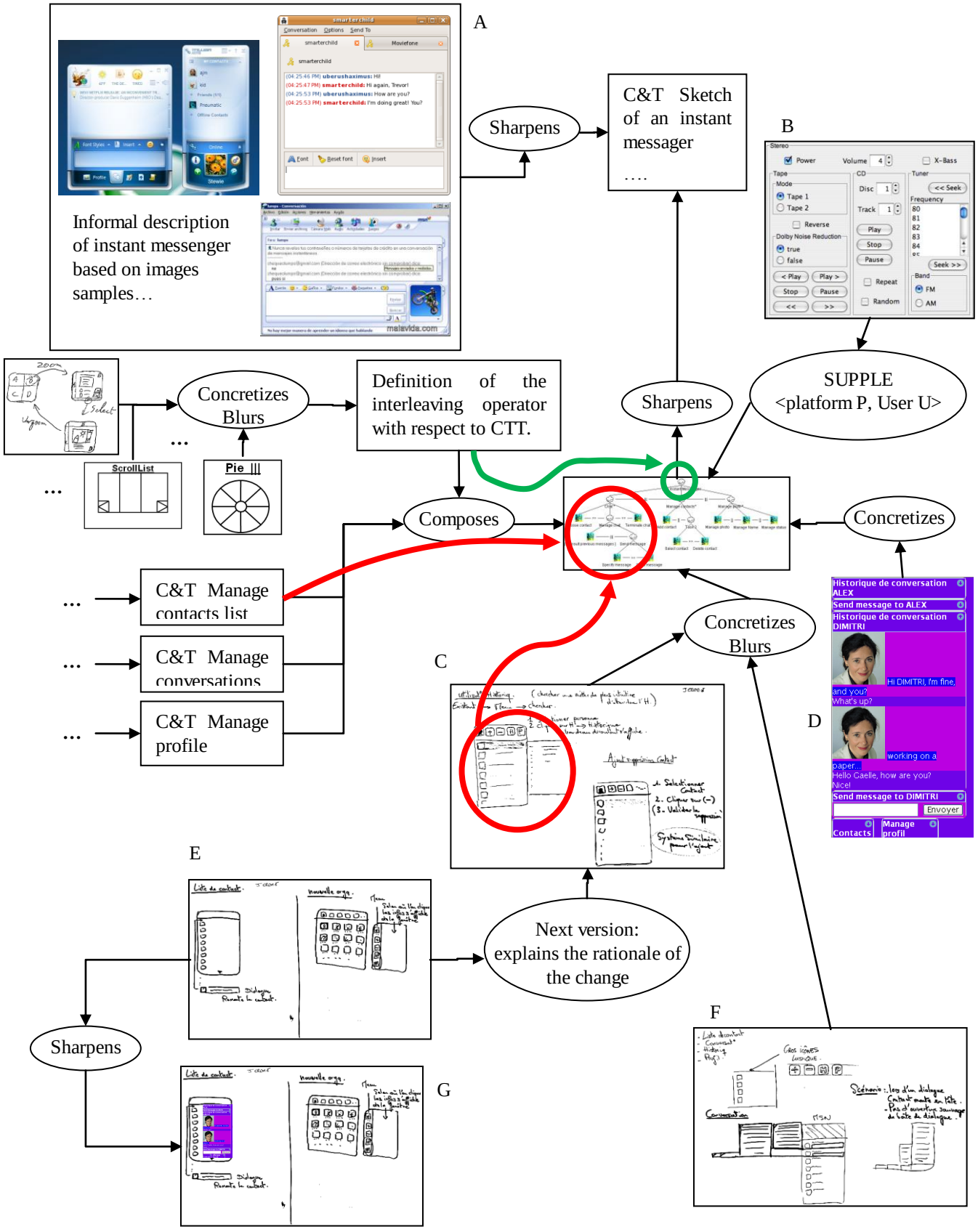

Figure 4: Excerpt of a graph of UI models.