CanonSketch and TaskSketch: Innovative Modeling Tools for Usage-Centered Design

Larry Constantine, IDSA
Constantine & Lockwood Ltd
Rowley, MA 01969
+1 978 948 5012
lconstantine@foruse.com

Pedro Campos
Department of Mathematics and Engineering
University of Madeira
Funchal, Madeira, Portugal
+351 291 705287
pcampos@uma.pt

ABSTRACT
Two experimental tools to support usage-centered design using essential use cases and canonical abstract prototypes are described. The models and methods of usage-centered design are outlined and the new tools are described briefly.

Categories and Subject Descriptors
D.2.2 [**Software Engineering**]: Design Tools and Techniques – *user interfaces.* H.5.2 [**Information Interfaces and Presentation**]: User Interfaces – *graphical user interfaces (GUI), prototyping, theory and methods.* H.1.2 [**Information Systems**]: User/Machine Systems – *human factors.*

General Terms
Design, Human Factors.

Keywords
canonical abstract prototype, abstract user interfaces, user interface design, usage-centered design essential use cases, task cases, task modeling.

1. INTRODUCTION
Because all software ultimately serves human needs, the design of the interface between software and its users is a necessary and integral part of software engineering. Nevertheless, software engineering models and methods, as well as the tools supporting them, have largely ignored user interface (UI) design. Despite its ambitious moniker, the Unified Modeling Language (UML) [14], a pastiche of discrete models of diverse heritage, also fails in this arena. UML is particularly weak when it comes to expressing visual and interaction design, appropriating a Procrustean patchwork of models originally intended for other purposes [13]. Ironically, UML is also weak in a critical aspect of use cases, one of the core models of object-oriented software engineering [12]. Widely employed for requirements modeling in software engineering, use cases are also commonly used in user-centered and usage-centered design for modeling user tasks. While recognizing use cases as modeling objects, neither UML nor the software tools that implement it define or recognize the internal structure and content of use cases. Unfortunately, it is these details of user interaction with software that are important drivers for effective UI design.

OOPSLA'05, October 16–20, 2005, San Diego, California, USA.
ACM 1-59593-193-7/05/0010.

CanonSketch [2, 4] and TaskSketch [1, 3] are experimental tools that address these critical shortcomings of UML and UML-based tools and techniques, providing improved support for usage-centered design [8, 10, 11]. Usage-centered design (U-CD) is a model-driven refinement of conventional user-centered design [5]. In U-CD, both the user interface and the internal structure of software are derived from simple, robust models of user roles, user tasks, and the architecture of the user interface. The detailed visual and interaction design of the user interface is derived from a canonical abstract prototype (CAP) [4, 6] representing the content and organization of the user interface, the abstract prototype is based on a rich task model in the form of essential use cases (EUC) representing the needs and intentions of users [7, 9], which, in turn, is based on a user role model representing the roles users play in relation to the system being designed. TaskSketch supports modeling with EUCs and CanonSketch supports modeling with CAPs.

These proof-of-concept tools were developed in Objective-C for Mac OS X using object-oriented software engineering techniques such as the Model-View-Controller pattern. Both tools use industry-standard object modeling notation (UML) and compatible extensions and integrate with standard tool suites through XMI export. Beta releases of the tools are available at http://dme.uma.pt/canonsketch and http://dme.uma.pt/tasksketch.

2. CANONSKETCH
CanonSketch supports abstract user interface prototyping using canonical abstract components (CAC). As shown in Figure 1, a CAP is a kind of wire-frame schematic constructed from a highly refined set of standard components (CACs). A CAC models a particular interactive function as viewed by the user in a standardized manner divorced from details of appearance and behavior of any specific realization. A CAP thus serves as an intermediary between task and object models on the one hand and working user interface prototypes on the other, allowing the designer to separate architectural issues and decisions from design details.

Examples (see Figure 1) include an abstract container, which holds information for presentation to the user, or a toggle that enables the user to alternate between states. CAPs have proved to be effective in promoting innovative visual and interaction designs that better support enhanced user performance. CanonSketch enables rapid modeling and prototyping with CACs through three synchronized views at different levels of abstraction: UML class model, canonical abstract prototype, and functioning HTML prototype.

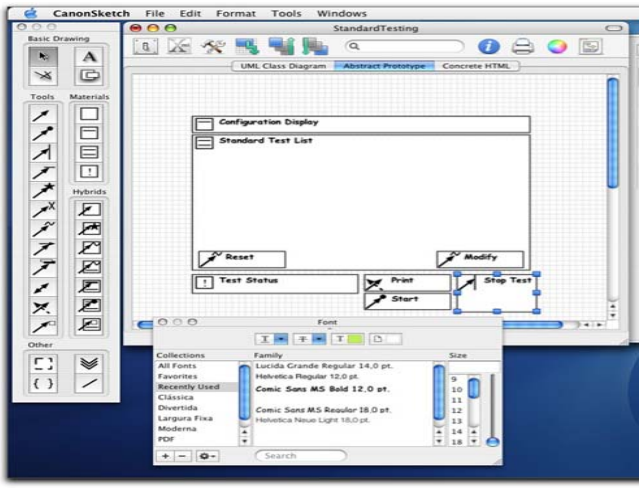

Figure 1. CanonSketch showing sample CAP.

3. TASKSKETCH

TaskSketch (Figure 2) is an interactive requirements elicitation and modeling tool focused on linking and tracing essential use cases. An EUC is a use case representing a single, discrete end-user intention expressed in essential form, that is, as an abstract, generalized, simplified, and technology- and implementation-independent narrative [7, 9]. TaskSketch supports collaborative modeling by multiple stakeholders, including clients, marketing staff, and software engineers. It is unique in facilitating the development and exploration of the conceptual architecture based on use case narratives developed in essential form. It enables tracing system requirements, in terms of user intentions and system responsibilities, to the conceptual architecture of the system, making it easy to extract the software architecture from task flows and to prioritize development of classes.

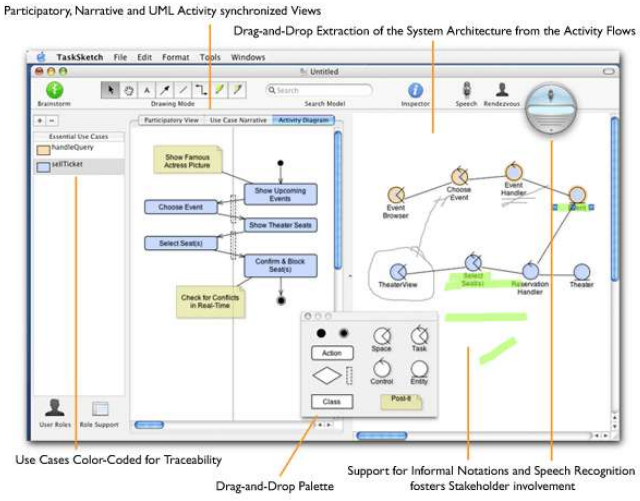

Figure 2. TaskSketch illustrated.

4. REFERENCES

[1] Campos, P. Task-driven tools for requirements engineering. In *13th International Requirements Engineering Conference (RE'05), Doctoral Consortium.* August 30, Paris, France.

[2] Campos, P. and Nunes, N. A UML-based tool for designing user interfaces. In *UML Modeling Languages and Applications: UML 2004 Satellite Activities* Lisbon, Portugal, Oct. 11-15, 2004 Revised Selected Papers. Springer-Verlag, 2004.

[3] Campos, P. and Nunes, N. J. A human-work interaction design approach by modeling the user's work styles. In *Workshop on Describing Users in Contexts - Perspectives on Human-Work Interaction Design, INTERACT'2005,* 12-16 September 2005, Rome, Italy.

[4] Campos, P. and Nunes, N. Canonsketch: A user-centered tool for canonical abstract prototyping. In *Proceedings of DSV-IS'2004, 11th International Workshop on Design, Specification and Verification of Interactive Systems.* Springer-Verlag, 2004.

[5] Constantine, L. Beyond user-centered design and user experience. *Cutter IT Journal, 17* (2), February, 2004.

[6] Constantine, L. Canonical abstract prototypes for abstract visual and interaction design." In J. Jorge, N. Jardim Nunes, and J. Falcao e Cunha, Eds. *Interactive Systems: Design, Specification, and Verification. Proceedings,* 10th International Workshop, DSV-IS 2003. Lecture Notes in Computer Science, Vol. 2844. Springer-Verlag, 2003.

[7] Constantine, L. Essential modeling: use cases for user interfaces. *ACM Interactions, 2* (2): 34-46, April 1995.

[8] Constantine, L., Biddle, R., and Noble, J. Usage-centered design and software engineering: models for integration. In IFIP Working Group 2.7/13.4, editor, *ICSE 2003 Workshop on Bridging the Gap Between Software Engineering and Human-Computer Interaction,* Portland, Oregon, 2003.

[9] Constantine, L., and Lockwood, L. Structure and style in use cases for user interfaces. In M. van Harmelan, Ed., *Object Modeling and User Interface Design.* Boston: Addison Wesley, 2001.

[10] Constantine, L., and Lockwood, L. *Software for Use: A Practical Guide to the Essential Models and Methods of Usage-Centered Design.* Reading, MA: Addison-Wesley, 1999.

[11] Constantine, L., and Windl, H. Usage-centered design: scalability and integration with software engineering. In C. Stephanidis and J. Jacko (Eds.) *Human-Computer Interaction: Theory and Practice. Proceedings of the 10th International Conference on Human-Computer Interaction,* Crete, Greece, 22-27 June 2003. Mahwah, New Jersey: Lawrence Erlbaum Associates, 2003.

[12] Jacobson, I., Christerson, M., Jonsson, P. and Övergaard, G. *Object-Oriented SoftwEngineering: A Use-Case Driven Approach.*. Reading, MA: Addison-Wesley, 1992.

[13] Kruchten, P., Ahlqvist, S., and Byland, S. User interface design in the Rational Unified Process. In M. van Harmelan, Ed., *Object Modeling and User Interface Design.* Boston: Addison Wesley, 2001.

[14] Rumbaugh, J., Jacobson, I., and Rumbaugh, J *The Unified Modeling Language reference Manual.* Reading, MA: Addison-Wesley, 1999