Pascal Beaujeant, François Beuvens, Adrien Coyette, Didier Dulait, Juan Manuel Gonzalez Calleros, Josefina Guerrero, Francisco Martinez, Efrem Mbaki, Jérémie Melchior, Hildeberto Mendonca, Kênia Sousa, Ricardo Tesoriero, and Jean Vanderdonckt

Université catholique de Louvain,

Belgian Laboratory of Computer-Human Interaction

Place des Doyens, 1 – B-1348 Louvain-la-Neuve (Belgium)

bchi@uclouvain.be

T: + 32 (0) 10 47 85 25, F: + 32 (0) 10 47 83 24

1. Interest in the workshop

The Belgian Laboratory of Computer-Human Interaction (BCHI) is conducting research, development, and consulting services in the domain of user interface engineering. This domain is located midway between software engineering, human-computer interaction, and usability engineering. The long term goal of BCHI is to be able to rely on a global approach for engineering the software development life cycle of the user interface (UI) of interactive systems. We developed our first model-based UI development in 1988: Trident (Tools for an Interactive Development Environment) which was working in the DEC Windows, OSF/Motif environment. Since that time, we have pursued continuous research and development first in model-based UI design and then in UI model-driven engineering throughout several projects such as: FP4 Syrecos, FP5 Cameleon, FP6 Similar, FP7 Human, FP7 Selfman, FP7 Serenoa, Salamandre, MulPlex. In 1993, we introduced the concept of Abstract Interaction Object (AIO) that was implemented in the Trident project that was based on the DSL language (Dynamic Specification Language). Since that time, the lab has been working towards a better definition of a User Interface Description Language (UIDL). More recently, we ensure the scientific coordination of the ITEA2 Call3 European project UsiXML.

2. Model-Based User Interface with UsiXML

To cope with the ever increasing diversity of Markup languages, programming languages, toolkits, and User Interface (UI) development environments, UsiXML (which stands for User Interface eXtensibe Markup Language – http://www.usixml.org) consists of XML-compliant User Interface Description Language (UIDL) that supports the “µ7” concept. It should be possible to specify a user interface for an interactive application that should support multi-device, multi-platform, multi-user, multi-linguality / culturality, multi-organisation, multi-context, or multi-modality capabilities. A complete environment has been created based on conceptual modelling framework of user interfaces organized around three axes:

1. the *models* that characterize various facets of a µ7 user interface from the end user’s viewpoint and the UIDL that allows designers and developers to specify such interfaces;

2. a *method* for developing interfaces in forward, reverse, and lateral engineering based on these models;
3. a suite of *software tools* that support designers in applying the method based on the models.

This suite is compatible with the Model Driven Architecture (MDA) recommendations: all models adhere to the principle of separation of concerns and are compliant with a meta-model defining the semantics of the UIDL, along with definitions of abstract and concrete syntaxes, as well as stylistics, when any. A UI transformation language can be applied in order to perform model-to-model transformation and rendering engines (as compiler or interpreter) support model-to-code production.

UsiXML is different from a pure UI authoring language in that it could also be used as a specification language. The ultimate goal is not only to generate code, but also to have the capability to reason about the UI specifications, such as model checking, UI evaluation, model-driven engineering, maintenance of repository of UI cases or patterns, static and dynamic analysis, model testing.

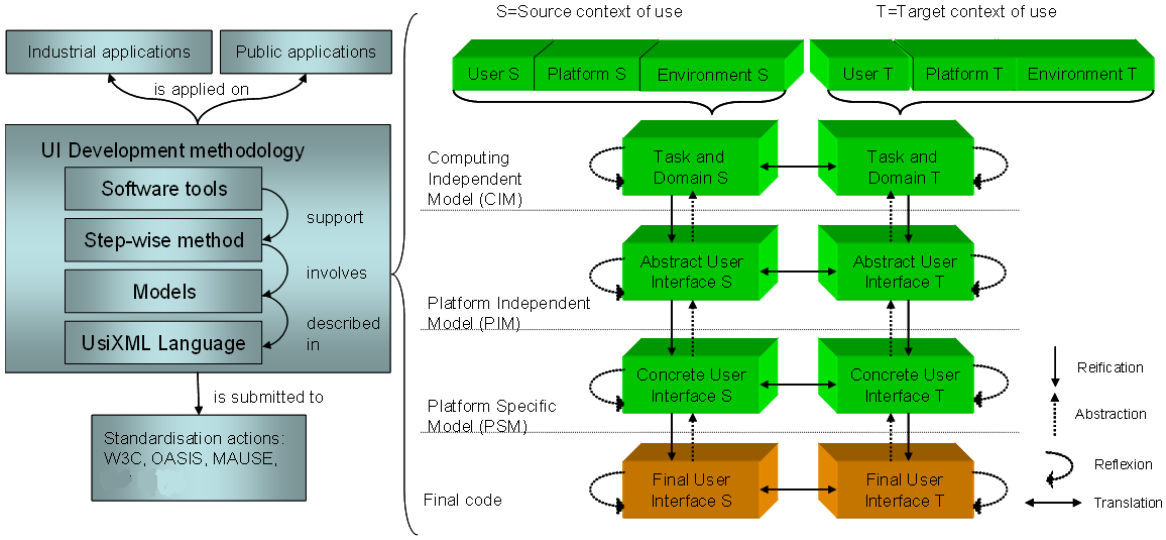

Figure 1 Overview of the UsiXML structured according to Cameleon Reference Framework [1].

The UI development method is based on the Cameleon Reference Framework (CRF) [1], which defines UI development steps for multi-context interactive applications. Its simplified version structures development processes for two contexts of use into four development steps (each development step being able to manipulate any specific artefact of interest as a model or a UI representation) (fig. 1):

1. *Task & Concepts* (T&C): describe the various users' tasks to be carried out and the domain-oriented concepts as they are required by these tasks to be performed.
2. *Abstract UI* (AUI): defines abstract containers (AC) and abstract individual components (AIC) [7][9] two forms of Abstract Interaction Objects (AIO) [17] by grouping subtasks according to various criteria (e.g., task model structural patterns, cognitive load analysis, semantic relationships identification), a navigation scheme between the container and selects abstract individual component for each concept so that they are independent of any modality.

3. *Concrete UI* (CUI): concretizes an abstract UI for a given context of use into Concrete Interaction Objects (CIOs) [17] so as to define widgets layout and interface navigation. A CIO is defined as an entity that users can perceive and/or manipulate (e.g., a push button, a list box, a check box, a sound). The actual specification realizes an abstraction of widget sets found in popular toolkits: 2D graphical (Java Swing, HTML 4.01, Flash) and auditory (earcons and VoiceXML 2.0).
4. *Final UI* (FUI): is the operational UI i.e. any UI running on a particular computing platform either by interpretation (e.g., through a Web browser) or by execution (e.g., after compilation of code in an interactive development environment).

The framework also exhibits three types of transformation types: *Abstraction* (respectively, *Reification*) is a process of eliciting artefacts that are more abstract (respectively, concrete) than the artefacts that serve as input to this process. Abstraction is the opposite of reification. Translation is a process that elicits artefacts intended for a particular context of use from artefacts of a similar development step but aimed at a different context of use. Therefore, when there is a need to switch from one context of use (e.g., one platform) to another one (e.g., another platform), the development process can cope with adaptation to the new context of use at any level of abstraction. Of course, the higher the level of abstraction the adaptation is performed, the more flexibility we have in the resulting processes.

To support the application of a particular development path, a suite of tools (Figure 2) has been developed and is currently being expanded. The most significant tools belonging to this suite are:

- TransformiXML is a Java application for defining, storing, manipulating, and executing model-to-model transformations expressed as graph transformations contained in graph grammars.
- IdealXML [10] is a Java graphical editor for the task model, the domain model, and the abstract model. It can also establish any mapping between these models either manually (by direct manipulation) or semi-automatically (by calling TransformiXML).
- KnowiXML [4] consists of an expert system based that automatically produces several AUIs from a task and a domain models for various contexts.
- GrafiXML [10] is a UsiXML high-fidelity editor with editing of the CUI, the context model and the relationships between. It is able to automatically generate UI code in HTML, XHTML, XUL and Java thanks to a series of plug-ins.
- VisiXML [18] is a Microsoft Visio plug-in for drawing in mid-fidelity graphical UIs, that is UIs consisting exclusively of graphical CIOs. It then exports the UI definition in UsiXML at the CUI level to be edited by GrafiXML or another editor.
- SketchiXML [4] consists of a Java low-fidelity tool for sketching a UI for multiple users, multiple platforms (e.g., a Web browser, a PDA), and multiple contexts of use.
- Several renderers are currently being implemented: FlashiXML [5] opens a CUI UsiXML file and renders it in Flash, QtkXML in the Tcl/Tk environment, and InterpiXML for Java.

- VisualiXML [12] personalizes a UI and produces one thanks to generative programming techniques for Visual C++ V6.0.
- ReversiXML opens a HTML file and reverse engineers it into UsiXML at both the CUI and AUI levels.
- DistriXML [7] consists of a set of software modules developed to support a particular form of Distributed User Interfaces (DUIs): attachable and detachable user interfaces.
- MigriXML [13] is a virtual reality system representing the user's real environment, based on UsiXML models: the platforms found in that environment, the UI of interactive graphics applications that are executed on these platforms, and the user.
- MultimodaliXML [13] tool is an assembly of five software modules for computer-aided design of multimodal user interfaces.

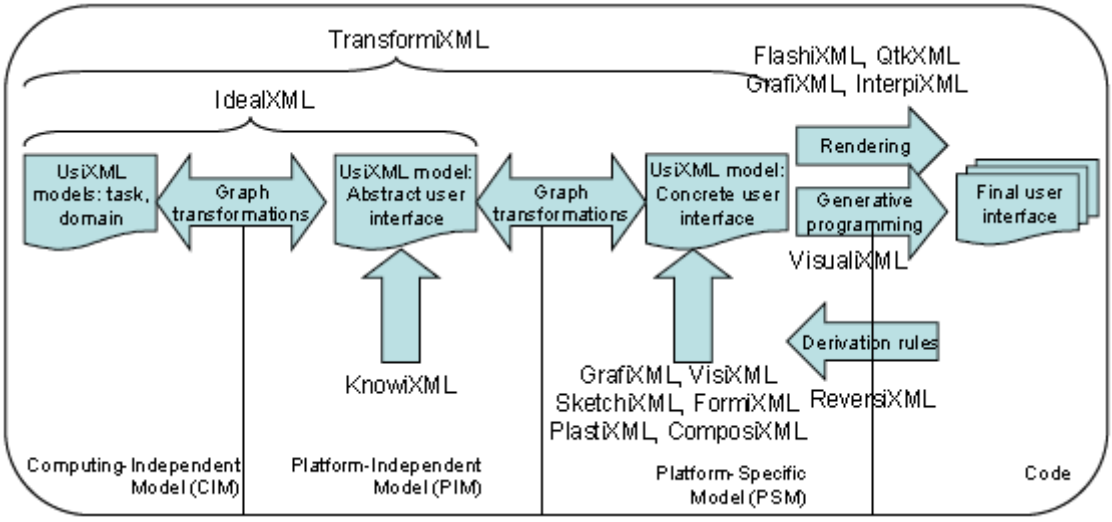

Figure 2 The suite of UsiXML tools structured according to the MDA classification

REFERENCES

[1] Calvary, G., Coutaz, J., Bouillon, L., Florins, M., Limbourg, Q., Marucci, L., Paternò, F., Santoro, C., Souchon, N., Thevenin, D., Vanderdonckt, The CAMELEON Reference Framework, Deliverable D1.1

[2] Calvary, G., Coutaz, J., Thevenin, D., Limbourg, Q., Bouillon, L., Vanderdonckt, J.: A Unifying Reference Framework for Multi-Target User Interfaces. Interacting with Computers, Vol. 15, No. 3 (June 2003) 289–308.

[3] Constantine, L.L.: Canonical Abstract Prototypes for Abstract Visual and Interaction Design. In: Proc. of 10th Int. Workshop on Design, Specification, and Verification of Interactive Systems DSV-IS'2003 (Funchal, June 11-13, 2003). Lecture Notes in Computer Science, Vol. 2844. Springer-Verlag, Berlin (2003) 1–15. Accessible at http://www.foruse.com/articles/abstract.pdf

[4] Coyette, A., Vanderdonckt, J.: A Sketching Tool for Designing Anyuser, Anyplatform, Anywhere User Interfaces. In: Proc. of 10th IFIP TC 13 Int. Conf. on Human-Computer Interaction INTERACT'2005 (Rome, 12-16 September 2005). Lecture Notes in Computer Science. Springer-Verlag, Berlin (2005)

[5] FlashiXML available at http://www.usixml.org/index.php?mod=pages&id=24

[6] Furtado, E., Furtado, V., Soares Sousa, K., Vanderdonckt, J., Limbourg, Q.: KnowiXML: A Knowledge-Based System Generating Multiple Abstract User Interfaces in UsiXML. In: Proc. of 3rd Int. Workshop on Task Models and Diagrams for user interface design TAMODIA'2004 (Prague, November 15-16, 2004). ACM Press, New York (2004) 121–128

[7] Grolaux, D., Vanderdonckt, J., Van Roy, P., Attach me, Detach me, Assemble me like You Work, Proc. of 10th IFIP TC 13 Int. Conf. on Human-Computer Interaction Interact'2005 (Rome, 12-16 September 2005), Lecture Notes in Computer Science, Vol. 3585, Springer-Verlag, Berlin, 2005, pp. 198-212

[8] Limbourg, Q., Vanderdonckt, J., Michotte, B., Bouillon, L., Lopez, V.: UsiXML: a Language Supporting Multi-Path Development of User Interfaces. In: Proc. of 9th IFIP Working Conference on Engineering for Human-Computer Interaction jointly with 11th Int. Workshop on Design, Specification, and Verification of Interactive Systems EHCIDSVIS'2004 (Hamburg, July 11-13, 2004). Springer-Verlag, Berlin (2005).

[9] Limbourg, Q., Multi-path Development of User Interfaces. Ph.D. thesis. Université catholique de Louvain, IAG-School of Management. Louvain-la-Neuve (Nov. 2004).

[10] Michotte, B., Vanderdonckt, J., GrafiXML, A Multi-Target User Interface Builder based on UsiXML, Proc. of 4th International Conference on Autonomic and Autonomous Systems ICAS'2008 (Gosier, 16-21 March 2008), IEEE Computer Society Press, Los Alamitos, 2008

[11] Montero, F., Lozano, M., González, P.: IDEALXML: an Experience-Based Environment for User Interface Design and pattern manipulation. Technical Report DIAB-05-01-4. Universidad de Castilla-La Mancha, Albacete (2005).

[12] Schlee, M., Vanderdonckt, J.: Generative Programming of Graphical User Interfaces. In: Proc. of 7th Int. Working Conference on Advanced Visual Interfaces AVI'2004 (Gallipoli, May 25-28, 2004). ACM Press, New York (2004) 403–406

[13] Stanciulescu, A., Limbourg, Q., Vanderdonckt, J., Michotte, B., Montero, F., A Transformational Approach for Multimodal Web User Interfaces based on UsiXML, Proc. of 7th Int. Conf. on Multimodal Interfaces ICMI'2005 (Trento, 4-6 October, 2005), ACM Press, New York, 2005, pp. 259-266

[14] Vanderdonckt, J., Mendonça, H., Molina Massó, J.P., Distributed User Interfaces in Ambient Environment, Proc. of AmI-07 Workshop on "Model Driven Software Engineering for Ambient Intelligence Applications" MDA-AMI'07 (Darmstadt, November 7-10, 2007), Lecture Notes in Computer Science, Springer-Verlag, Berlin, 2007, pp. 44-52

[15] Vanderdonckt, J., A MDA-Compliant Environment for Developing User Interfaces of Information Systems, Proc. of 17th Conf. on Advanced Information Systems Engineering CAiSE'05 (Porto, 13-17 June 2005), O. Pastor & J. Falcão e Cunha (eds.), Lecture Notes in Computer Science, Vol. 3520, Springer-Verlag, Berlin, 2005, pp. 16-31.

[16] Vanderdonckt, J., Bouillon, L., Chieu, K.C., Trevisan, D.: Model-based Design, Generation, and Evaluation of Virtual User Interfaces. In: Proc. of 9th ACM Int. Conf. on 3D Web Tech. Web3D'2004 (Monterey, April 5-8, 2004). ACM Press, New York (2004) 51–60.

[17] Vanderdonckt, J., Bodart, F.: Encapsulating Knowledge for Intelligent Automatic Interaction Objects Selection. In: Proc. of the ACM Conf. on Human Factors in Computing Systems INTERCHI'93 (Amsterdam, 24-29 April 1993). ACM Press, New York (1993) 424–429.

[18] VisiXML available at http://www.usixml.org/index.php?mod=pages&id=13