# 3D User Interfaces for Information Systems Based on UsiXML

**Juan Manuel González-Calleros, Jean Vanderdonckt**
Université catholique de Louvain
Place des Doyens 1, 1348
Louvain-la-Neuve, BELGIUM
+32 (0)1 047 8349
{juan.m.gonzalez, jean.vanderdonckt}@uclouvain.be

## ABSTRACT
Since many years, 3D interactive systems have demonstrated some benefits in reproducing adequately the reality, in improving it, and even in augmenting it by providing the user with unprecedented actions. 3D User Interfaces are becoming the primary subject of interest of a growing community of researchers and developers adopting different approaches for specifying and creating 3DUIs. Providing development methods and software support for 3DUIs is a complex problem. In this paper, we argue that developing 3DUIs for Information Systems is an activity that would benefit from the application of a model-driven development methodology composed of: a set of models defined according to an ontology, a language that expresses these models, and a structured method manipulating these models.

## CATEGORIES AND SUBJECT DESCRIPTORS
D2.2 [**Software Engineering**]: Design Tools and Techniques – *Modules and interfaces; user interfaces*. D2.m [**Software Engineering**]: Miscellaneous – *Rapid Prototyping; reusable software*. H.1.2 [**Information Systems**]: Models and Principles – *User/Machine Systems*. H5.2 [**Information interfaces and presentation**]: User Interfaces – *Prototyping; user-centered design; user interface management systems (UIMS)*.

## General Terms
Design, Experimentation, Human Factors, Verification.

## Author Keywords
User Interface, 3D User Interface, 3DUI, Model-Based User Interface Development, Information Systems design.

## INTRODUCTION
In HCI, a *Graphical User Interface* (GUI) is often understood as a User Interface (UI) that involves graphical widgets that are displayed as planar regions in *xy* planes according to their abscissa and ordinates. As such, they are considered as *two-dimensional UIs* (2DUIs). Many desktop environments support overlapping of widgets, pseudo-relief effects (such as shadows) and depth effects, thus raising the level up to *two-dimensional and half user interfaces* (2D½UIs). In contrast, *three-dimensional user interfaces* (3DUIs) involve graphical widgets that are rendered as volumes in *xyz* spaces according to their 3D coordinates.

2DUIs, as well as 3DUIs, could be used to support the user to achieve a 2D or a 3D interactive task. A 2D task in nature, e.g., navigating on a map, can be supported by a 2DUI and does not necessarily require a 3DUI. A 3D task in nature, e.g., controlling a satellite in space, can be supported by both 2DUI and 3DUI, but not with the same quality. Table 1 categorizes UI types according to two axes: the nature of the task (2D *vs.* 3D) and the associated front-end (2D *vs.* 3D). For instance, navigating on a map, a 2D task in nature, can be rendered as a GUI on a 2D desktop, but also as a flat object in a 3D environment, although this is not particularly interesting. Organising tasks in windows, a 2D task in nature, is typically achieved in a 2DUI, but also benefit from a 3DUI (Figure 1). Controlling a satellite in space, a 3D task in nature, would really benefit from a 3DUI, although it could be projected into 2D planes as a GUI in 2D. In most situations, 3DUIs have shown some benefits with respect to 2DUIs, but also some shortcomings. 3DUIs are not automatically superior or inferior to 2DUIs. Moreover, some transitions may become desirable from 2D to 3D and vice versa in order to ensure appropriate representation of a change of context and reasons exist why maintaining 2D contents in 3D [10].

| | Nature of the task | |
|---|---|---|
| **Front-end** | **2D** | **3D** |
| **2D** | 2DUI, such as a GUI | GUI in 2D |
| **3D** | GUI in 3D | 3DUI |

**Table 1. Nature of the task and its front-end.**

Some reasons to care about 3DUIS are the following: some users, not all, prefer the use of 3DUIs [6] although they are helpful for specific tasks, but not all [40]; human perceptual mechanisms to analyze the world into structures of 3D primitives are better compared to 2D representations [5]; human visual bandwidth is much larger in 3DUIs than in 2DUIs [39]; users tend to remember better objects shapes and location in 3DUIs than in 2DUIs [38]; companies venturing into virtual worlds virtual worlds, such as IBM for hosting virtual meetings, has been rewarded with an increment in collaboration [28].

Thus, 3DUIs are becoming the primary subject of interest of a growing community of researchers and developers [3,27] adopting different approaches for specifying and creating 3DUIs. Providing development methods and software support for 3DUIs is a complex problem. Researchers are at a stage where they are developing new interaction

In Faure, D., Vanderdonckt, J., (Eds.), *Proc. of 1st Int. Workshop on User Interface Extensible Markup Language UsiXML'2010* (Berlin, 20 June 2010), Thales Research and Technology France, Paris, 2010. ISBN 978-2-9536757-0-2

techniques, gestures and metaphors for 3DUIs [33]. Most research and development is focusing on technological issues, as reported in a survey of major publications on 3DUIs [43]. The research is mainly focusing on how to overcome hardware and software issues [11]. Little or no attention is devoted to the design knowledge that should drive the development life cycle of 3DUIs.

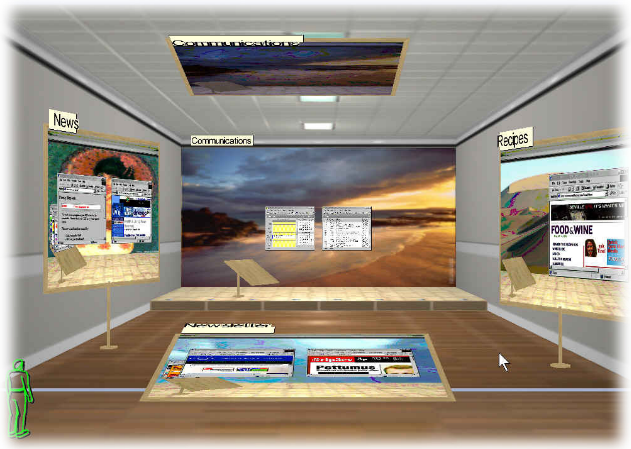


**Figure 1. TaskGallery, an example of a 2D task in 3D [38].**

In this paper we argue that developing 3DUIs for Information Systems is an activity that would benefit from the application of a development method which is typically composed of: (1) a set of models defined according to an ontology, (2) a language that expresses these models, and (3) a principle-based approach manipulating these models based on principles.

## STATE OF THE ART

3DUIs can be the result of different development approaches: programmatic, toolkit based, method.

In the *programmatic approach*, the 3DUI is obtained by directly coding in its target computer language, e.g., C++. The programmatic approach is frequently preferred for applications where performance is a priority such as in games. Most games are written in C++ although, some may use C to try to get even more speed (at the cost of not having built in Object Oriented support). There are also some other application programming interface (API) that can be used in most common programming languages, for instance, Google O3D, Microsoft XNA.

A 3DUI can be also coded as a XML-based language, examples are: X3D, VRIXML [7,9]. Programming and maintaining 3DUIs without any method could be not as simple as it gives no guarantee for regularity. There are no evaluation criteria to consider unless the final result is achieved. The tendency is on the “rush to code” approach without any structure favours a “trial and error” method.

A *toolkit approach* allows a straightforward implementation of a final interface once modelled in a tool. Using a predefined set of objects that help developers in their programming task, the toolkit approach offers the best solution to draw 3DUIs elements with an immediate feedback. A lot of toolkits exists for 3D modelling, open source (Google SketchUp, Vivaty Studio), and commercial (Autodesk, Anark). While they are very similar in their basic capabilities, there are some differences more related to the interoperability with other technologies. For instance Autodesk Maya or 3ds Max can import 2D diagrams from AutoCAD and making the process of creating 3D content very simple. A toolkit is always useful and it was for us for designing our 3D objects. However does not provide design knowledge for developing 3DUIs.

The *methodological approach* relies on the step-wise frameworks for developing 3DUIs. There is a plethora of methods VR-Wise [35], CoGenIVE [9], InTML [13], Contigra [8], Tres-D [31], Desktop 3DUIs [25,26,42], Participative [ 34], Task analysis [14,37] to develop 3DUIs, we might have skipped some significant work in this area but we are not trying to be exhaustive but just reviewed some significant exiting work. Although these different methods share some similarities on their steps, several conceptual dissimilarities differentiate them.

Consequently, it is challenging to transfer one abstraction from one method to another in order to have one consistent framework supporting the development lifecycle of 3DUIs. They decompose the software life cycle into steps and substeps, but these methods rarely provide the design knowledge that should be typically used for achieving each step. Most approaches cover aspects such as task modelling, dialog modelling and implementation aspects. While some knowledge in explicit of existing method most of the time it is hard to find the abstracted models, the transformational approach along with its rules.

To identify shortcomings on existing work, a series of comparative analyses were conducted using the three axes recommended in [1]: descriptive part: a common ground is needed to describe every piece of work (for this purpose we use the Cameleon reference framework); comparative part: a set of criteria were defined to compare the different works that we described using a common syntax (next paragraph describe such criteria); generative part: new work emerges from the comparative analysis as a result of the identification of limitations or potentiality of the literature review (the proposed methodology is the result of this process).

The properties analyzed in the comparison were: models manipulated by the toolkit, all of them compliant with UsiXML models with the following notation: Di = dialog, AUI=abstract presentation, CUI=concrete user interface, FUI = Final User Interface, U = user, C = context. Along with this models to support a transformational approach then a series of inter model transformation is needed. When such transformational approach exists then it is explicitly denoted using the following notation: **(A, …, B)** indicates that A, …, B are grouped models that are done at the same level; **A↔B** indicates that A derivates B and B is reengineered from A; **A→B** indicates that A derivates B; **A≈ ↔B** indicates that model A concepts could be manually linked to model B concepts and that B can be manually reengineered to A; **A≈→B** indicates that model A concepts could be manually linked to model concepts B. This means that the rules and the models exist but not a tool to support the automatic transformation.

| Methods | Models | Inter Model Transformation |
|---|---|---|
| **VR-Wise** | CUI | CUI → FUI |
| **CoGenIVE** | T, Di, CUI | ( T, Di, CUI ) ↔ FUI |
| **InTML** | Di, CUI | T ≈→ (Di, CUI),<br>(Di, CUI) ≈→ FUI |
| **Contigra** | CUI, Di | (CUI, Di) → FUI |
| **Tres-D** | T, U, C, Di, Do | T, Do≈→ CUI, CUI≈→ FUI |
| **Desktop 3DUIs** | T, CUI, Di | T≈→CUI, CUI → FUI |
| **Participative** | T, CUI, Di | T≈→CUI, CUI ≈→ FUI |
| **Task analysis** | T, CUI, Di | T≈→CUI |

**Table 8 Comparison of Model-Based methodologies**

## 3D USER INTERFACES FOR INFORMATION SYSTEMS

Similarly to a 2DUI, a 3DUI can be decomposed into two parts: the presentation part (also called *front-end*) and the semantic core part (also called *back-end*) equipped with the semantic functions, the system data storage, and the communication layer. 3DUIs are often associated to various 3D systems, such as those in virtual reality, mixed reality, augmented reality, and 3D desktop environments [29]. We focus of the 3DUIs we propose is Information systems (IS) for desktop-based systems. An IS for an organization is a construction made up of four blocks [2]: (1) data, a partial representation of facts that interest the organization; (2) processes, that represent means to acquire, search, store, present, and con-vey information; (3) organizational rules, governing the implementation of informational treatments; and (4) human & Technical Resources, required for the functioning of IS.

An IS supports management tasks according two axes [2]: functioning level, which ranges from operational, decisional to strategic; and the structure level, which ranges from structured to informal. We primarily consider management tasks that are operational and structured activities typically corresponding to administrative tasks, which are defined to deal with routine activities [2]. In this scenario, a context of use is assumed to be quasi-constant: the physical environment is assumed to be an office setup; the user has known skills required conducting these administrative tasks, and a desktop computer is considered as the main computing platform. Therefore, we consider that 3DUIs that correspond to other tasks than such administrative tasks are beyond the scope of this thesis, as well as contexts of use that significantly depart from this assumption. For instance, a mobile traveler could conduct administrative tasks, but the resulting context no longer satisfies our assumption. And so do 3D games, volumetric displays, Organic UIs.

For administrative tasks supported by IS several interaction styles are candidates [45]: form filling, multi-windowing, direct manipulation, iconic interaction, graphic interaction, multimedia interaction, and 3DUIs. 3DUIs were chosen as a potential interaction style for IS since this option is underexplored. IS examples that are not covered by the present work includes: non interactive contents (e.g., a surgery room), information visualization (e.g., 3D statistics), and custom 3D contents (e.g., a stadium).

## MODEL-BASED DEVELOPMENT OF 3DUIs

### Models

In this section we define concepts related to 3DUIs, i.e., the representation of 3D widgets, their characteristics, and how from a user task we can derive the 3DUI. These models were incorporated to UsiXML to support:

- 3D Rendering of 2D User Interfaces. The UsiXML model that considers 2D GUI was specialized to consider the attributes specific for their 3D rendering. The extension specialize the Concrete Interaction Objects (*CIOs*) by adding extra attributes to define their size (height, width) and position (top, left) when it applies.
- 3DUIs A second approach involves a true 3D presentation of the 3DUI. By true representation is meant going beyond an imitation of their 2D GUI counterpart. In that scenario it was not enough to rely on the existing model as concepts such as appearance, texture, shape, and behaviour were needed.
- Hapgets Haptically enhanced 3D widgets [23,24].

We create the 3DUI model that includes concepts to express haptic interaction and different representations for 3D UIs depicted in Figure 2.

### Language

In order to be fully-MDA compliant, this works need a User Interface Description Language (UIDL). Some environments may includes models, and a transformational approach but do not have a genuine modeling language behind. It is not just because there is a XML language that a genuine modeling language may exist [46]. A genuine UIDL must be strongly defined based on a trilogy (semantics, syntax, stylistics), this is the case of UsiXML. A review of the literature of existing UIDLs was conducted in [15] to justify the selection of UsiXML as UIDL for our work.

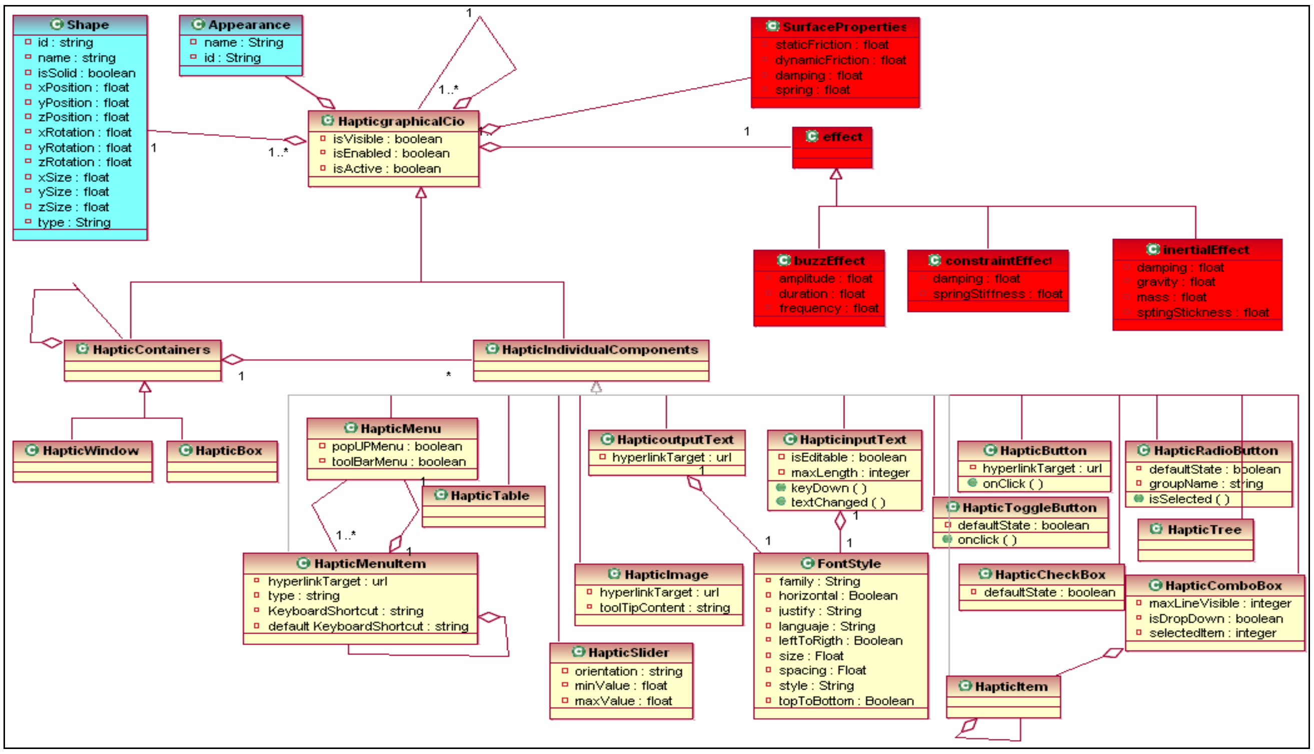


**Figure 2 UsiXML 3DUI Concrete User Interface Model.**

Regarding the UIDL trilogy we briefly comment that: the semantics are expressed as Meta-Models using the UML class diagram notation; the syntax is abstract expressed as XML Schemas and concrete expressed as a XML file; finally, the stylistics are the graphical representation used to represent the different artifacts involved in the methodology.

## Software Tools

The set of software tools required to support the development of 3DUIs includes: model editors to assist the designer in constructing the models; design critics providing a designer with quality assessment facilities, models capturing explicit properties of the artefact are an ideal representation to perform evaluation; implementation tools translate a specification into a representation that can be used by a compiler, an interpreter or an interface builder; and transformation tools provide support to the designer to edit, store and execute model transformation rules.

Finding the right tool is a trade-off between six main criteria [41]: partial support of the tool not supporting the whole development process, learning time, building time, communication with other subsystems, extensibility and modularity. Supporting the evolution and the reuse of software remains a challenge. Although, it is hard to keep a valance we select the software to be used based on extensibility and modularity.

The software modules that support the methodology are:

- IdealXML [32]. Task, domain and AUI models are designed sing this tool. The semantics and models is from UsiXML. The notation of the task model is based on the stylistics from CTT [35] while its operators on the semantics of LOTOS operators. The domain model uses as stylistics UML class diagrams. The AUI uses an innovative expression for gathering abstract concepts related to the AUI model. Transformations from task and AUI model are supported.
- Usability Adviser [44]. Usability guidelines were introduced for each development step. Usability guidelines can be evaluated automatically using the Usability Adviser this software determines the usability of any UI specified in UsiXML.
- 3D Modelling was possible by relying on non-commercial tool: Alice (www.alice.org), Blender (www.blender.org), VUIToolkit [31] and Vivaty (http://developer.vivaty.com/).
- Software Support for Transformations. An analysis was reported for the transformation engines [18] used to support the transformational approach of our methodology.
- Haptic support was tested via the haptic browser, which was reported in [23,24].

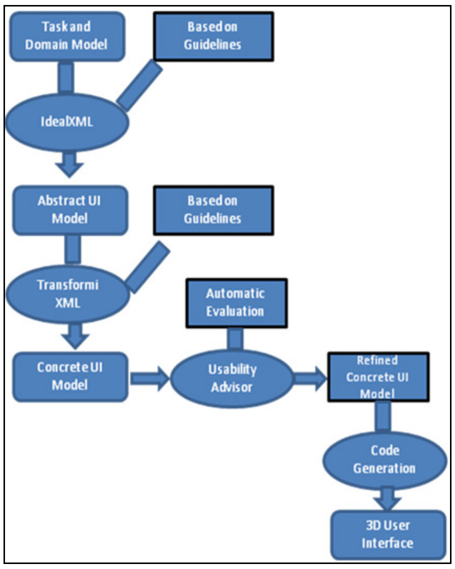


**Figure 3. Outline of the method for developing 3DUIs.**

## METHOD AND CASE STUDY

The goal is not to come up with yet another Software Development Method but to reuse existing work and structure it accordingly. The result is a method that structures the development life cycle of a 3DUI of an IS in a principle-based way. The method follows an exploratory approach as its goal is to show a variety of possibilities to encourage design. It is said that is structured as it is based on a structured Framework, the Cameleon Framework [4]. Second, a set of instruments (evaluation guidelines [19,21], task patterns [21], canonical task types [20], rules for model to model transformations [17,18]) guide de use of the method then making it principle-based. The method (Figure 3) includes an evaluation of each development step, which can be performed manually or automatically. In order to provide another possibility to evaluate a 3DUI, our method could be an option to cope with the budget problem that rises when costly user experiments including experts are needed [Bowm02]. We rely on principles expressed as guidelines for modelling the task and abstract user interface (AUI) model. They are applied manually and refine the models. At the concrete user interface (CUI) model direct evaluation over the code is performed by evaluating rules and conventions or recommendations. The refined model is then used for the code generation. The next subsections details the development steps. Due to space reasons the following case study will not include haptic interaction in the final result.

### Step 1: Task and Domain Modeling

In order to provide some means to designers for task modelling we propose [21]: 1) to reuse existing successful solutions existing as task patterns and 2) to follow a set of guidelines that might lead to a consistent task model that later can be object of transformations.

The 3D Universal interaction tasks patterns described in [3] are expressed in UsiXML task model notation. It is not the scope of this paper to go beyond this description since most issues related to pattern-based design is extensively addressed in the literature. Even that the use of pattern-based development life-cycle is contradictory to model-driven engineering, as patterns are poorly structured or in many different ways. There has been some works [15] showing the potential of modelling patterns for task models and to reuse successful solutions. This could be easily extended into more detailed pattern Markup languages such as PLML (Pattern Language Markup Language.

Guideline based evaluation of task modelling is based on principles reflecting broad knowledge about task modelling coming from the literature. For instance, promote the use of systematic mechanism for the selection of task attributes, such as the task types [20].

*Task Model Case Study*

In this example we refer to a simple navigation task in a virtual office, with two interactive objects: an interactive table and an interactive screen. The user can interact with a table, navigate through the room and interact with a screen. For the navigate room, the task model uses the identified task pattern travel and Wayfinding. The interaction with big Screen indicates as the user interacts with screen task refers to a turn on/off a screen than renders video or images.

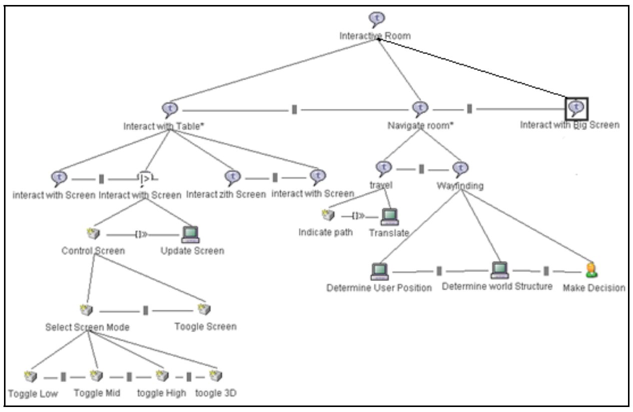


**Figure 4. Virtual Office Task Model.**

### Step 2: Task and Domain Modeling

An AUI model can be generated automatically or produced manually from a task model following a set of heuristics. Various set of heuristics may fit this purpose depending on the type of AUI to be obtained: an AUI that reflects the task structure, an AUI minimizing navigation, an AUI compacting input/output. Most of them are compliant with the knowledge base from UsiXML. Although this level is independent of any modality, some guidance is still desired on how AUI might be structured considering further reifications into 3D CUI objects. Several metaphors have been introduced in order to display information or windows. If we imagine for instance a cube to render the different tasks as an Abstract Container (AC), then authors need to add inputs with navigation facets in order to guarantee the cube transitions.

*AUI Model Case Study*

The task model of previous section (Figure 4) is reified into the AUI model (Figure 5), using the rules are similar as in the previous example so there is no need to explain the internal process of the tool. What is relevant is to notice, that even that we put in the task model the make decision task for wayfinding, at the CUI model this kind of task do not appear, as they are not part of the UI.

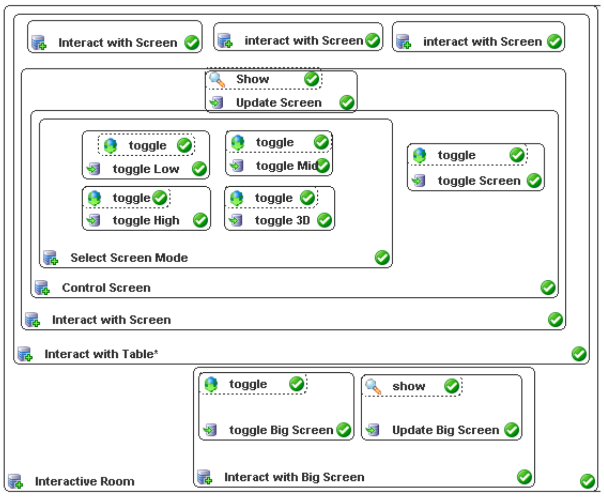


**Figure 5. Partial Abstract User Interface Model of the Virtual Office.**

### Step 3: Concrete User Interface Modeling

Abstract Interaction Objects (AIOs) can be selected based on the facet of the Abstract individual component (AIC), the action type and action item of the task [20]. Unfortunately, it is not enough while the action type and action item combined with the facet to properly select the AIO. An example can be used to clarify this situation. Assuming that the UI action type corresponds to a select of a collection of elements, then, several are the potential AIOs that can be used such as: combo box, radio button group, text fields. The problem became then on deciding the appropriate AIO depending on the context of use, the type of value to be selected, and the domain. For that purpose, UsiXML meta-models can be used. Our aim is on the use of them and to provide guidelines on the proper selection of AIOs. How to differentiate 2D and 3D tasks working on 2D or/and 3D objects? Another question is related to the final code. What is the appropriate representation of 3DUIs? Should the 2D desktop metaphor still be used or are there alternative visualizations or metaphors. Several attempts go towards defining a new toolkit of 3D objects which are natively appropriate to 3D applications. Again, this represents an advantage to have a predefined collection of such 3D-widgets, but then the interaction is reduced by what they offer natively.

The primary problem to solve at this level of abstraction consists in determining which mapping rules can be defined in order to transform an AUI into one or several CUIs. The selection has been largely reported [17, 19,21,22]. The second, not less important, issue is regarding the selection of AIOs. For that purpose we build a taxonomy of 3DUIs. The taxonomy can be of some help in making that decision [3] as a design space of potential representations of the AIO. During a second phase, selecting 3D presentations based on the questions and answers method as we did for the hapgets [23,24].

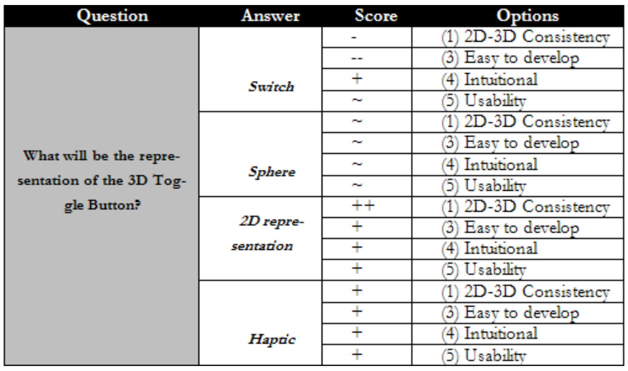
The problem is not just a matter of 3DUI representation, but also the selection of the appropriate representation. The possible mappings and guidelines to support the correct transfer from task model to 3DUI widgets are also relevant. Because it is not enough just to use any arbitrarily selected widget, for instance, a combo box for selecting a value instead of a radio button group. The selection of a simple value can be mapped to a radio button group or a list box, the difference relies on the number of possible values to select. This characteristic could be part of the design in UsiXML [30].

*CUI Model Case Study*

The third step implies a transformational system that is composed of necessary rules for realizing the transition from AUI to CUIs. We won't consider in this example the attachment of objects to any surface, we just create a direct mapping, for each component and then in the high level editor each component is put in the corresponding shape. This sub-step involves the highest number of rules of all transformation sets as the different combinations of facet types, data types, cardinalities, are numerous. Figure 6 provides the subset of rules applied in this case study. The designer can choose among the different alternatives provided by the rules. We illustrate in Figure 7 how the taxonomy can be used for proper selection of the representation of the widget. The meaning of the simbols are: the darkest solid line (++) means strongly supported, dark solid line (+) means supported, solid line (~) means neutral, dash lines (-) means denied and dot lines (..) means strongly denied.

| Abstract Interaction Component | Facet Specification | Information to take into account | Possible Concrete Interaction Component |
|---|---|---|---|
| "Navigation" | Navigation | The platform used, an internet Browser with any pug-in. | There is no need for any concretization of this task or any subtask |
| Interact with big screen/ Screen | Toggle + Element (Screen/Button) | The big screen reacts on touch, the small screen react with a toggle button. This is also a design rule. | Toggle button, touch screen |
| "select screen mode" | Select attribute value + selection values known | The set of possible values are in subtask instead of a domain list of values. | A group of toggle button acting as a radio group. |
| "Toggle Low/Mid/High/3D" | Toggle + element | None | Toggle button |
| "Update Screen" | Communicate (Show) + Element (Image or video) | Attribute, data type, domain characteristics | An image component or video component |

**Figure 6. Correspondence between AIOs and CIOs.**



| Question | Answer | Score | Options |
|---|---|---|---|
| What will be the representation of the 3D Toggle Button? | *Switch* | - | (1) 2D-3D Consistency |
| | | -- | (3) Easy to develop |
| | | + | (4) Intuitional |
| | | ~ | (5) Usability |
| | *Sphere* | ~ | (1) 2D-3D Consistency |
| | | ~ | (3) Easy to develop |
| | | ~ | (4) Intuitional |
| | | ~ | (5) Usability |
| | *2D representation* | ++ | (1) 2D-3D Consistency |
| | | + | (3) Easy to develop |
| | | + | (4) Intuitional |
| | | + | (5) Usability |
| | *Haptic* | + | (1) 2D-3D Consistency |
| | | + | (3) Easy to develop |
| | | + | (4) Intuitional |
| | | + | (5) Usability |

**Figure 7. Questions and answer criteria to select a toggle button.**

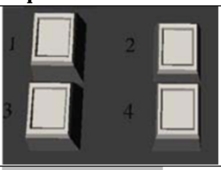
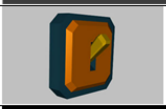
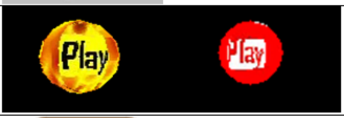
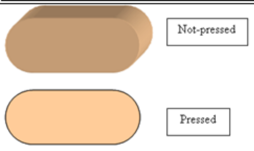
| | Representation |
|---|---|
| 2D Representation |  |
| Switch |  |
| Sphere |  |
| Haptic |  |

**Figure 8. Graphical representation for a toggle button.**

In this particular example the decision was towards the use of a more 2D related representation, see options in Figure 8. The resulting specifications are obtained by realizing the above transformational development sub-steps. Figure 9, present a mock-up of the graphical UI corresponding to the CUI model.

### Step 4: Final User Interface

The resulting CUI 3DUI can be encoded in a software tool, such as: Vivaty studio, Blender. The idea is not to start from scratch your models, this can be done but it is preferable to have a direct mapping in the tools and reuse existing work. Of course authors might produce their own objects if needed. The target languages in which we rely to concretize 3DUIs are: Java3D, VRML, X3D. The final rendering can be evaluated using any usability evaluation method. We are not aware of any system that performs usability evaluation directly on the code of a 3DUI (e.g., on VRML), although this could be a future avenue for automated evaluation. This is because at this stage, it is very complicated to analyze the code in a meaningful way. This is why we propose the use of automatic evaluation over the CUI model where semantics expressing the 3DUI can be object of evaluation [21]. For the final rendering guidelines can be followed before generating the code. We rely on existing knowledge on guidelines and some proposed such as those related to the hapgets [24].

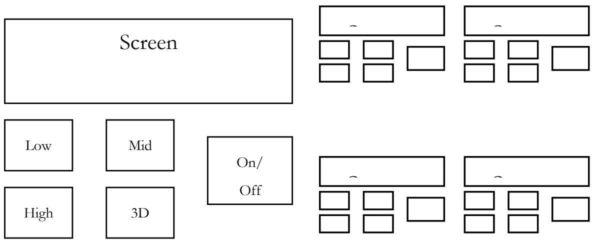


**Figure 9. Mock-up of the Control Screen and Interacting table.**

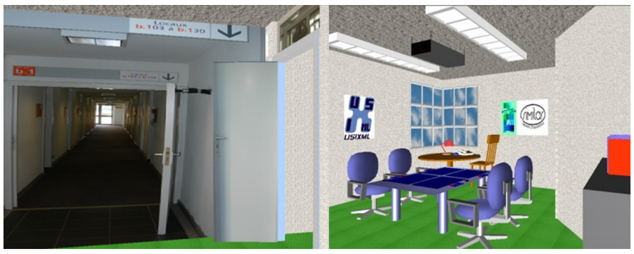


**Figure 10. Virtual office with an interactive table.**

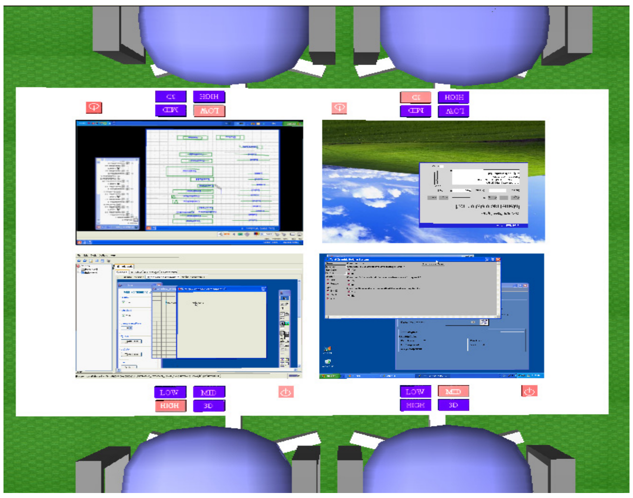


**Figure 11. Top View of the virtual table.**

*FUI Rendering of the Case Study*

The screenshot of Figure 10 shows the virtual office. Notice that the office follows the guideline "Virtual objects should be similar as much as the real objects" [25]. The Big screen, Figure 11 shows the interactive table composed of four control screens. The navigation task is controlled by the Cortona player plug-in., which support the navigation with the mouse and keyboard, as input devices, the user just decide where to go.

## EVALUATION

Several cases studies have been developed around this methodology for different domain of application. Their goal was to serve as a proof of the concept of the different principles introduced in the methodology, and to prove the feasibility of method through a set of case studies.

The first case study [17] was devoted to the development of an opinion polling system, a reasonably scaled example of a typical information system. A second case study [19,21] is dedicated to the development of an administrative application for a management school. Even that the problem complexity was moderate it has been chosen to illustrate the design diversity that offers 3DUI. A third case study [22] was devoted to the development of a flight navigation system for an aircraft. The complexity of this system is high but is more related to the algorithms that compute aircraft behaviour during the flight, while all this already exist. We conduct an in depth study to provide a MDA to the development of the Navigation Display UI and namely using a 3DUI instead the 2DUI. Finally, the fourth case study [23,24] was dedicated to the rendering of web site in the haptic web browser.

## CONCLUSION

The development methodology relies on three main axes: models, method and language. The contribution to *UsiXML models* is summarized: more than 200 attributes, 90 classes, 100 relationships were added to the UsiXML models that corresponds mainly to aspects for 3DUIs and some more to task modelling concepts. A sanity check of the resulting models was made to consolidate them.

The *method* aspects adhere to the MDE paradigm. Models and transformations are explicitly defined and used, around 85 transformations rules (25 old, 15 adapted, 45 new). The method relies on the Cameleon reference framework then is said that is structured. It just provides means for forward engineering. A set of principles decomposed in: guidelines for the different development steps for evaluating the resulting models, task models relying on patterns and a canonical list of task type. All these principles promote systematicity when modelling 3DUIs.

The development steps reinforce existing knowledge on 3DUI development methods at different levels. In terms of transformational explicitness, abstraction layers independent of the modality of interaction. We provide some means to identify the diversity of concretizations of the 3DUI (i.e. its representation) for a task.

3DUIs are not intended for everybody. For the increasing number of users interested by 3DUIs, organizations need to be present in virtual online applications. For those users who are not interested by 3DUIs, our method benefits from using UsiXML and its MDA approach introduced in this paper that supports multi-path development, i.e., one source, many targets, an important benefit. These targets vary in computer platforms (e.g., mobile phone, desktop), programming languages (e.g., Java, Html). The goal of this work was not to prove that the method is better than another, neither the usability of the 3DUI produced. The goal was to define a method to develop 3DUI in a principle-based way as opposed to an opportunistic way. Therefore, the appropriateness of this method in this paper is for sure the major piece of work to investigate in the near future.

## ACKNOWLEDGMENTS

We gratefully acknowledge the support of the ITEA2 Call 3 UsiXML project under reference #20080026.

## REFERENCES

1. Beaudouin-Lafon, M., Instrumental Interaction: an Interaction Model for Designing Post-WIMP User Interfaces. *Proc. of ACM Human Factors in Computing Systems* (*CHI'2000*), La Haye (The Netherlands), April 2000, ACM Press, CHI Letters 2(1), pp.446-453.
2. Bodart, F. Pigneur, Y., Conception assistée des systèmes d'information : méthode, modèles, outils, *Masson*, 1989.
3. Bowman, D.A., Kruijff, E., LaViola, J., Poupyrev, I., 3D User Interfaces: Theory and Practice, *Addison Wesley*, Boston, July 2004.
4. Calvary, G., Coutaz, J., Thevenin, D., Limbourg, Q., Bouillon, L., and Vanderdonckt, J. A Unifying Reference Framework for Multi-Target User Interfaces. *Interacting with Computers 15*, 3 (June 2003) 289–308.
5. Carrol, J., HCI models, theories, and frameworks: toward a multidisciplinary science, *Morgan Kaufmann*, San Francisco, 2003.
6. Cockburn, A., McKenzie. 3D or not 3D? Evaluating the Effect of the Third Dimension in a Document Management System. In *Proc. of the ACM Conf. on Human factors in computing systems CHI'2001*, Seattle (USA), 31 March-5 April 2001, ACM Press, New York, 2001, pp. 434–441.
7. Cuppens, E., Raymaekers, Ch., Coninx, K, A Model-Based Design Process for Interactive Virtual Environments, *Proc. of Int. Workshop on Design, Specification, and Verification of Interactive Systems* (*DSV-IS'2005*), Newcastle upon Tyne (England), 13-15 July 2005, Lecture Notes in Computer Science, Vol. 3941, Springer, Berlin, 2005, pp. 225-236.
8. Dachselt, R., Hinz, M., Meißner, K. CONTIGRA: An XML-Based Architecture for Component-Oriented 3D Applications, *Proc. of 7th Int. Conf. on 3D Web Technology* (*Web3D'2002*), Tempe (USA), 24-28 February 2002, ACM Press, New York, 2002, pp. 155–163.
9. De Boeck, J., Raymaekers, Ch., Coninx, K., A Tool Supporting Model Based User Interface Design in 3D Virtual Environments. In *Proc. of the 3rd Int. Conf. on Computer Graphics Theory and Applications GRAPP'2008* (Funchal, January 22-25, 2008). INSTICC - Institute for Systems and Technologies of Information, Control and Communication, 2008, pp. 367-375.

10. Deléglise, E., Paul, D., Fjeld, M., 2D/3D Web Transitions: Methods and Techniques, *Proc. of the 5th Int. Conf. on Web Information Systems and Technologies* (*WEBIST'2009*), Lisbon (Portugal), 23-26 March 2009, INSTICC Press, 2009, pp. 294-298.

11. Dünser, A., Grasset, R., Seichter, H., Billinghurst, M., Applying HCI Principles in AR Systems Design, *Proc. of the 2nd Int. Workshop on Mixed Reality User Interfaces: Specification, Authoring, Adaptation* (*MRUI'07),* Charlotte (USA), 11 March 2007.

12. Fencott, C., Isdale, J., Design Issues for Virtual Environments, *Proc. of Int. Workshop on Structured Design of Virtual Environments and 3D-Components at the Web3D 2001 Conference*, Paderborn (Germany), 2001.

13. Figueroa, P., Green, M., Hoover, H. J., InTml: A Description Language for VR Applications, *Proc. of 7th Int. Conf. on 3D Web Technology* (*Web3D'2002*), Tempe (USA), 24-28 February 2002, ACM Press, New York, 2002, pp. 53-58.

14. Gabbard, J.L., Hix, D., Swan, J.E. II, User-Centered Design and Evaluation of Virtual Environments, *IEEE Computer Graphics and Applications*, 19 (6), November 1999, pp. 51-59.

15. Gaffar, A., Sinnig, D., Seffah, A., Forbrig, P., Modeling patterns for task models, *Proc. of 3rd Int. Workshop on Task Models and Diagrams for user interface design* (*TAMODIA'2004*), Prague, 15-16 November 2004, Ph. Palanque, P. Slavik, M. Winckler (Eds.), ACM Press, New York, 2004, pp. pp. 99-104.

16. Guerrero García, J., González Calleros, J.M., Vanderdonckt, J., and Muñoz Arteaga, J. A Theoretical Survey of User Interface Description Languages: Preliminary Results. In *Proc. of Joint 4th Latin American Conference on Human-Computer Interaction-7th Latin American Web Congress LA-Web/CLIHC'2009* (Merida, November 9-11, 2009), E. Chavez, E. Furtado, A. Moran (Eds.). IEEE Computer Society Press, Los Alamitos, 2009, pp. 36-43.

17. Gonzalez-Calleros, J.M., Vanderdonckt, J., and Muñoz-Arteaga, J.M. A Method for Developing 3D User Interfaces of Information Systems. In *Proc. of 6th Int. Conf. on Computer-Aided Design of User Interfaces CADUI'2006* (Bucharest, 6-8 June 2006). Chapter 7, Springer-Verlag, Berlin, 2006, pp. 85-100.

18. González-Calleros, J.M., Stanciulescu, A., Vanderdonckt, J., Delacre, J.P., and Winckler, M. Comparative Analysis of Transformation Engines for User Interface Development. In *Proc. of 4th Int. Workshop on Model-Driven Web Engineering MDWE'2008* (Toulouse, 1 October 2008). N. Koch, G.-J. Houben, A. Vallecillo (Eds.), CEUR Workshop Proceedings, Vol. 389, 2008, pp. 16-30.

19. González-Calleros, J.M., Vanderdonckt, J., and Muñoz-Arteaga, J. A Structured Approach to Support 3D User Interface Development. In *Proc. of 2nd Int. Conf. on Advances in Computer-Human Interactions ACHI'2009* (Cancun, 1-6 February 2009). IEEE Computer Society Press, Los Alamitos, 2009, pp. 75-81.

20. González-Calleros, J.M., Guerrero-García, J., Vanderdonckt, J., and Muñoz-Arteaga, J. Towards Canonical Task Types for User Interface Design. In *Proc. of Joint 4th Latin American Conference on Human-Computer Interaction-7th Latin American Web Congress LA-Web/CLIHC'2009* (Merida, 9-11 November 2009). E. Chavez, E. Furtado, A. Moran (Eds.), IEEE Computer Society Press, Los Alamitos, 2009, pp. 63-70.

21. Gonzalez-Calleros, J.M., Vanderdonckt, J., and Muñoz-Arteaga, J. A Structured Methodology for Developing 3D Web Applications. In T. Spiliotopoulos, P. Papadopoulou, D. Martakos, and G. Kouroupetroglou (eds.), "Integrating Usability Engineering for Designing the Web Experience: Methodologies and Principles", Chapter 2, IGI Global Inc., Hershey, 2010, pp. 15-43.

22. Gonzalez-Calleros, J.M. A Model-Driven Approach for Developing Three-Dimensional User Interfaces of Information Systems in a Principle-based Way, PhD Thesis, Université catholique de Louvain Press, Louvain-la-Neuve, February 2010.

23. Kaklanis, N., González-Calleros, J.M., Vanderdonckt, J., Tzovaras, D. Hapgets, Towards Haptically-enhanced widgets Based on a User Interface Description Language. In *Proc. of Workshop on Multimodal Interaction Through Haptic Feedback MITH'2008* (Naples, 31 May 2008).

24. Kaklanis, N., González-Calleros, J.M., Vanderdonckt, J., Tzovaras, D. Haptic Rendering Engine of Web Pages for Blind Users. In *Proc. of 9th Int. Conf. on Advanced Visual Interfaces AVI'2008* (Naples, 28-30 May 2008). ACM Press, New York, 2008, pp. 437-440.

25. Kaur, K., Designing virtual environments for usability, Ph. D. Thesis, City University, London, 1998.

26. Kim, G.J., Kang, K.C., Kim, H., Lee, J., Software Engineering of Virtual Worlds. In *Proc. of ACM Symposium on Virtual Reality Software and Technology VRST'98* (Taipei, November 1998). ACM Press, New York, 1998, pp. 131-139.

27. Latoschik, M.E., Reiners, D. Blach, R., Figueroa, P., Dachselt, R., *Proc. of Workshop of Software Engineering and Architectures for Realtime Interactive Systems SEARIS'2008* (Reno, March 2008). IEEE Computer Society Press, Los Alamitos, 2008.

28. Laudon, K.C., Laudon, J.P., Management Information Systems: Managing the Digital Firm. Pearson, 2009.

29. Laviola, J., Bringing VR and Spatial 3D Interaction to the Masses through Video Games. *IEEE Computer Graphics and Applications 28*, 5 (2008), pp. 10-15.

30. Limbourg, Q., Vanderdonckt, J., Michotte, B., Bouillon, L., and 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 EHCI-DSVIS'2004* (Hamburg, July 11-13, 2004). Lecture Notes in Computer Science, Vol. 3425, Springer-Verlag, Berlin, 2005, pp. 200-220.

31. Molina, J.P., A Structured Approach to the Development of 3D User Interfaces. Ph.D. thesis, University of Castilla-La Mancha, Albacete, Spain, 29 February 2008.

32. Montero, F., López-Jaquero, V., Vanderdonckt, J., Gonzalez, P., Lozano, M.D., and Limbourg, Q. Solving the Mapping Problem in User Interface Design by Seamless Integration in IdealXML. In *Proc. of 12th Int. Workshop on Design, Specification, and Verification of Interactive Systems DSV-IS'2005* (Newcastle upon Tyne, 13-15 July 2005). S.W. Gilroy, M.D. Harrison (Eds.), Lecture Notes in Computer Science, Vol. 3941, Springer-Verlag, Berlin, 2005, pp. 161-172.

33. Myers, B., Hudson, S. E., Pausch, R., Past, Present and Future of User Interface Software Tools, *ACM Transactions on Computer Human Interaction 7*, 1 (March 2000), pp. 3-28.

34. Neale, H., Nichols, S., Designing and developing Virtual Environments: methods and applications, *Proc.of Workshop on Visualization and Virtual Environments Community Club VVECC'2001*, Designing of Virtual Environments, 2001.

35. Paternò F., Mancini C., and Meniconi S. ConcurTaskTree: A diagrammatic notation for specifying task models. In *Proc. of IFIP TC 13 Int. Conf. on Human-Computer Interaction Interact'97* (Sydney, 14-18 July 1997), Kluwer Academic Publishers, Boston, 1997, pp. 362-369.

36. Pellens, B., Bille, W., De Troyer, O., Kleinermann, F.: VR-WISE: A Conceptual Modeling Approach for Virtual Environments, *Proc. of the Methods and Tools for Virtual Reality Workshop MeTo-VR'2005*, Gent (Belgium), 2005.

37. Polys, N.F., Hetherington, R., Brutzman, D., Gracacin, D., Engineering Virtual Environments with X3D, Tutorial at ACM Web3D 2005 Symposium, 10th Int. Conf. on 3D Web Technology, Bangor, March 2005.

38. Roberts, N., Andersen, D., Deal, R., Garet, M., Shaffer, W., Introduction to Computer Simulation, a system dynamics modeling approach, *Productivity Press*, 1983.

39. Shneiderman, B., 3D or Not 3D: When and Why Does it Work?, Human-Computer Interaction Laboratory & Department of Computer Science University of Maryland, Talk in Web3D, Phoenix (USA), 26 February 2002.

40. Shneiderman, B., Why Not Make Interfaces Better than 3D Reality, Virtualization Viewpoints. Theresa-Marie Rhyme (Ed.), November-December, 2003.

41. Shneiderman, B., Plaisant, C., Designing the User Interface. 4th Edition, *Addison Wesley*, Reading, 2004.

42. Smith, S., Duke, D., Design Support for Virtual Environments. Proc. of Workshop: Design of Virtual Environments, Oxford shire, 2001.

43. Swan, J.E., Gabbard, J.L. Survey of User-Based Experimentation in Augmented Reality? *Proc. of 1st Int. Conf. on Virtual Reality,* Las Vegas (USA), 2005.

44. Vanden Bossche, P., Développement d'un outil de critique d'interface intelligent : UsabilityAdviser, M.Sc. thesis, *Université catholique de Louvain*, Louvain-la-Neuve, 1 September 2006.

45. Vanderdonckt, J., A Small Knowledge-Based System for Selecting Interaction Styles, *Proc. of Int. Workshop on Tools for Working with Guidelines* (*TFWWG'2000*), Biarritz (France), 7-8 October 2000, Springer-Verlag, London, 2000, pp. 247-262.

46. Vanderdonckt, J., Model-Driven Engineering of User Interfaces: Promises, Successes, and Failures, *Proc. of 5th Annual Romanian Conf. on Human-Computer Interaction* (*ROCHI'2008*), Iasi (Romania), 18-19 September 2008, S. Buraga, I. Juvina (Eds.), Matrix ROM, Bucarest, 2008, pp. 1-10. ISSN 1843-4460