Bridging the gap: developing 2D and 3D user interfaces with the IDEAS methodology

José Pascual Molina, Pascual González, M. Dolores Lozano, Francisco Montero and Víctor López-Jaquero

LoUISE – Laboratory of User Interaction and Software Engineering
IIIA – Instituto de Investigación en Informática de Albacete
University of Castilla-La Mancha
Escuela Politécnica Superior, Campus Universitario s/n,
02071 Albacete, Spain
{jpmolina, pgonzalez, mlozano, fmontero, victor}@info-ab.uclm.es

Abstract. Most user interface development methodologies have been conceived based on the experience from the development of traditional PC-based systems. However, computer and displays technology is changing. The size spectrum now ranges from large surrounding displays to small mobile devices. Besides, 3D graphics is no longer limited to graphics workstations, as most PCs are shipped with specialized hardware, and 3D standards are being developed for mobile devices. User interface engineering should not be left out his progress. With that concern in mind, the IDEAS methodology is presented, a novel environment which allows the development of both 2D and 3D user interfaces for a wide range of devices. A case of study is used to show the details of the proposed development process.

1 Introduction

The graphical user interfaces (GUIs) that are known as WIMP (which stands for Windows, Icons, Menus and Pointing) have been dominant for more than two decades. In a time when computer tools and applications became used by millions of people, software engineers realized that the interface was one of the main factors that determined the success of an application. WIMP user interfaces provided a "de facto" standard that, thanks to the existent consistencies in the look and feel of application interfaces, gave the user ease of learning and ease of use. As a result, user interface design was introduced in the life cycle of software development, and many methodologies have been proposed which support the development of user interfaces.

However, those methodologies have primarily been designed and created based on experience from the development of traditional PC-based systems, such as word processors or spread sheets. This situation is changing rapidly, though. The range of computer devices today includes not only desktop PCs, but also small but powerful internet-enabled mobile devices such as cell phones and PDAs. Besides, new interaction paradigms deserve more attention as technology matures, such as wearable computers, virtual reality, augmented reality and mixed reality. Interface technology

advances towards a fourth generation of user interfaces, a "post-WIMP" generation which is not only based on 2D widgets such as menus, forms or toolbars, but also on 3D user interfaces that enable a more natural and intuitive style of interaction [15].

Most of these new interfaces are largely designed by intuition, engaging developers in cycle of prototype creation and evaluation. A fundamental lesson learned in software engineering is that improvements in design and quality assurance processes require systematic work practices that involve well-founded methods [14]. Just as software engineering helps in the development of WIMP interfaces, new interfaces should also take advantage of a systematic development process.

In this paper, the IDEAS methodology is presented, a user interface development environment which, following a model-based approach, enables the developer to design and generate user interfaces for a wider range of devices and applications. IDEAS has been successfully applied to the creation of WIMP GUIs, either desktop or Web [9], but can also used to develop 3D user interfaces that aim to increase productivity by engaging users' spatial cognition and perception [11]. The generated 3D user interfaces are similar to the Win3D Windows front-end (www.clockwise3d.com) or those 3D virtual environments that can be designed following the IBM RealPlaces Guidelines [6], examples of 3D user interfaces that transforms the standard 2D desktop into a more intuitive virtual environment, where the user can navigate using arrow keys as in a 3D game, but also clicking on the target place where the user wants to move to.

This paper is structured in three main sections. First section describes the methodology and the abstract models that are used at each of the four abstraction levels that IDEAS defines. In the second section, a case of study is introduced, which is then used to demonstrate the user interface development process of the proposed methodology, with special attention to the development of both a 2D and a 3D GUI versions of the same system. Finally, some discussion is made about the proposed methodology and the user interfaces that are built with it, conclusions are presented and some lines of further research are sketched.

2 User interface development with the IDEAS methodology

IDEAS [10] aims to be an automatic interface development system integrated within the framework of the object oriented model OASIS [8] to support the automatic production of high-quality user interfaces. IDEAS is based on abstraction models that are used to understand and model the reality. The UI development process (Fig.1) is structured in four abstraction levels: requirements, analysis, design and presentation. This vertical structuring shows the reification processes followed from the first and most abstract level to finally reach the system implementation, which constitutes the last level. At each abstraction level, different models are used to describe the multiple aspects of the GUI. A brief description of them follows.

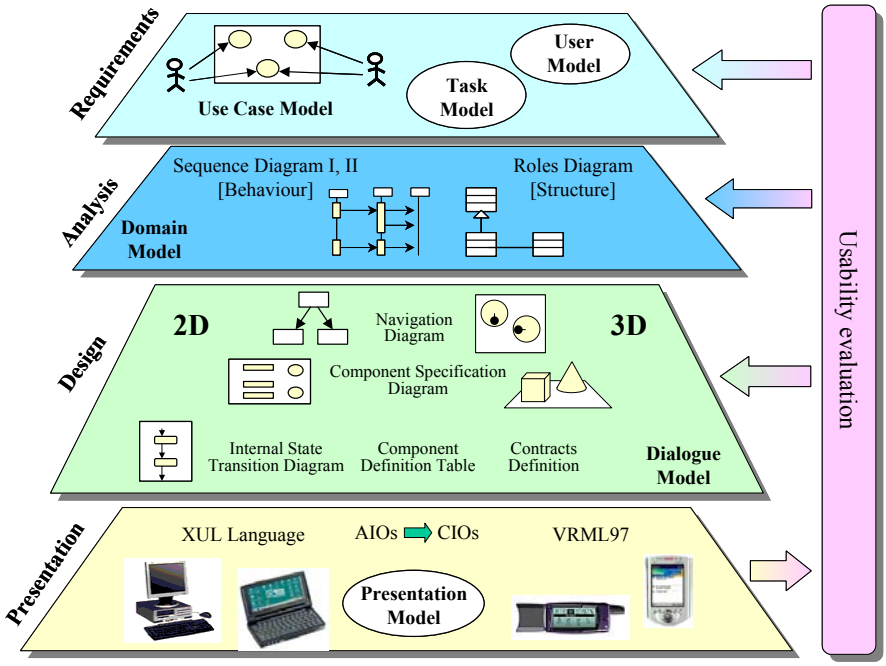

Fig. 1. IDEAS: 2D and 3D user interface development for a wide range of devices

2.1 Requirements

This level offers a number of abstraction tools that help the developer to gather the user requirements and to classify tasks and their subtasks. Three models are created: Use Case Model, Task Model and User Model. The *Use Case Model* captures the use cases identified within the information system. Then, for every one of the use case there will be one or more tasks which the user may perform to accomplish the functionality defined by the use case. These tasks are compiled and described in the Task Model. The *Task Model* defines the ordered set of activities and actions the user has to perform to achieve a concrete purpose or goal. A template, based on the one proposed by Cockburn [3], is used to describe all these issues using natural language. The *User Model* describes the characteristics of the different types of users. The purpose of this model is to support the creation of individual and personalized user interfaces.

2.2 Analysis

At analysis level the *Domain Model* is generated. This model consists of two diagrams. The first one is the Sequence Diagram, which defines the system behavior. The second one is the Role Model, which defines the structure of the classes that take

part in the associated sequence diagram together with the relationships among these classes, specifying the role of each one of them.

2.3 Design

At design level the *Dialog Model* is developed. All the models that have been generated up to now do not contain any graphical aspect of the final user interface. The proposed development process assumes that the hierarchy of tasks is independent on whether the user interface is 2D or 3D. It is from now on that these issues start to be addressed, describing the syntactic structure of the user-computer dialog by means of several diagrams. These diagrams capture the navigation and interaction aspects of the user interface, defines when the user can invoke commands, select or specify the input data and when the computer can require data from the user or display the output data.

On the one hand, the Navigation Diagram establishes which are the different interaction spaces and how the user jumps from one to another. On the other hand, the Component Specification Diagram, the Component Definition Table and the Internal State Transition Diagram defines how the user-system interaction will be performed at each interaction space by means of Abstract Interaction Objects (AIOs). The relationship between user interface objects and the domain objects is established by means of Contracts Definition.

2.4 Presentation

At this level, the *Presentation Model* is created. This model describes the final GUI according to the final platform, whether it is a mobile device, a desktop or a more sophisticated virtual reality system. The starting point for the generation the GUI is the Dialog Model developed at design level, which models the structure and the behavior of the GUI by means of AIOs. These AIOs are then translated into Concrete Interaction Objects (CIOs) which depends on the style of user interface and the widget toolkit.

Currently, a 2D user interface generation is performed by using XUL, an XML-based mark-up language, in order to make it as independent as possible from the final platform where the application is going to run. With regard to 3D user interfaces, the generation could be performed using VRML97 and JavaScript, Java3D or a virtual reality toolkit, such as Sense8 WorldUp. Once the translation is made, the developer can finally refine the GUI based on different style guidelines and other aesthetic issues, such as repositioning widgets or customizing colors.

3 A case of study: Multimedia zone

The Win3D desktop developed by ClockWise Corp. consists of several places, each one aimed to support a specific set of tasks. The M-Media place is committed to

multimedia, a place where not only the user can play the music he or she likes but also choose a wallpaper to personalize his or her desktop (Fig.2).

Fig. 2. ClockWise Win3D desktop: M-Media space

Based on this space, in this paper we propose a similar system as a case of study to show how IDEAS can be used to develop both 2D and 3D user interfaces. In our Multimedia Zone, the user can play a music album, as in the previous example, selecting the song he or she wants to listen to and tuning the volume up and down. The wallpaper has been replaced by a slide projector, which allows the user to view a presentation, deciding when to go to the next slide or go back to the previous one. In the following sections, the development process will be detailed. Some of the diagrams shown here are image captures from the IDEAS CASE application, a tool developed to help the user interface designer throughout the whole process.

3.1 Requirements

The abstract models used at the requirements level have been introduced in a previous section. This level consists of three main models. The first one is the Use Case Model. In our case of study, the *Initial Use Case Diagram* should show that the "User" is the main actor and that he or she is able to "View a Presentation" or "Listen to Music" (Fig. 3, left). Each use case can be enriched by adding entities, rules or other use cases that are included in them or are an extension of them. For instance, the "Listen to Music" use case has two entities, "Album" and "Track", and some rules, such as "No next song at the end of the album" (Fig. 3, right).

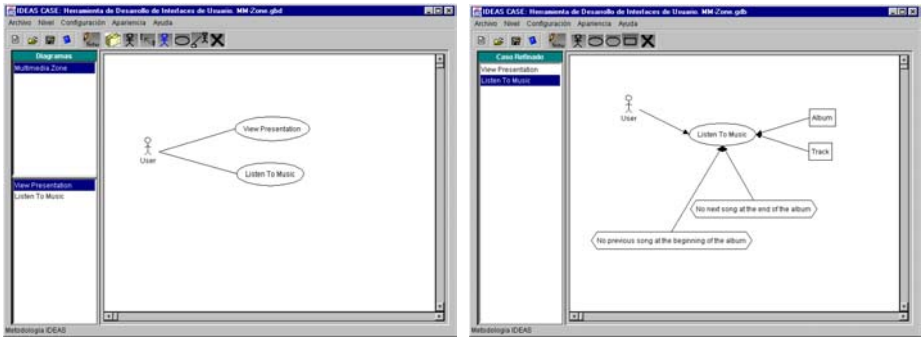

Fig. 3. Initial Use Case Diagram (left) and "Listen to Music" Use Case Diagram (right)

The next step is to identify user tasks from use cases, which represents the Task Model. In this case, only one task has been identified for each one of the two use cases. Thus, the "Listen to Music" use case corresponds to the task of the same name. In order to detail the actions involved, a *Task Template* is used. With this template, the UI designer specifies some general information related to a task, a normal scenario and its possible variants, and some other valuable information. The general information section includes data such as the goal of the task, its primary actor or the action that triggers that task. This general information can also refer to "where" the task is performed, information that can be useful when designing a 3D user interface. For instance, the "Listen to Music" task is related to our multimedia zone (Fig. 4, left).

Task Model: Template (Example: Listen To Music)

Task: Listen to music.

GENERAL FEATURES:
- GOAL: Listen to a music album.
- PRECONDITION: The music album is available.
- SUCCESS CONDITIONS: The music album has been played.
- FAILURE CONDITIONS: The music album has not been played.
- PRIMARY ACTOR: User.
- SECONDARY ACTOR: None.
- TRIGGER ACTION: The user launchs the music player application.
- WHERE: Multimedia zone.

NORMAL SCENARIO:
- The user makes the first song to be played.
- The next song is played*.
- The last song is played.

VARIANTS:
- The user skips current song.
- The user goes back to the previous song.
- The user stops playing the music album.

EXTENSIONS:
- The user turns the music up.
- The user turns the music down.

RELATED INFORMATION:
- Priority: Normal.
- Duration: -
- Frequency: -

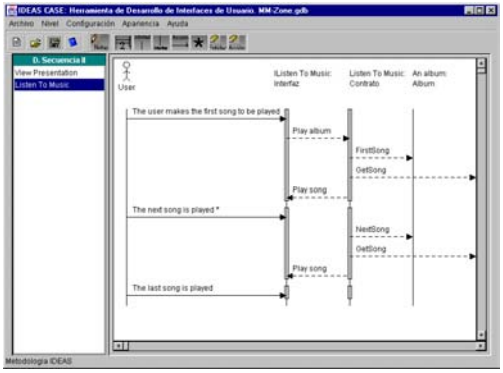

Fig. 4. "Listen to Music" Task Template (left) and Sequence Diagram (right)

The last abstract model of this level is the User Model. With this model, the developer gathers information related with each user, such as the tasks that he or she is allowed to execute. Users can also express their preferences about the type of visualization that he or she prefers to use in each task, including 2D and 3D information visualization. This data is compiled in the User Model too.

3.2 Analysis

The first diagram generated at analysis level is the *Sequence Diagram*, which models the system behavior. Tasks and their related actions are the starting point to describe a sequence diagram, which also involves interface and entity classes. For instance, in the normal scenario of the "Listen to Music" task (Fig. 4, right), the user can perform the action "Play album", which is received by the interface class and then to the system, which finds the first song to be played. The structure of classes are described in the Role Model, which uses UML class diagrams to show the different classes and their relationships.

3.3 Design

Once the behavior and structure aspects of the system have been detailed, the developer designs the user interface from a platform-independent point of view. Taking the hierarchy of tasks and the domain objects as the input, the design level deals with the generation of the Dialog Model, which includes five diagrams. This diagrams capture the navigation and interaction issues of the GUI, keeping a strong relationship with the style of user interface. This means that the diagrams that are used to sketch a 2D user interface differ from those created for a 3D user interface of the same system, though they support the same tasks. To illustrate the differences in design, a comparison of the Navigation Diagram and the Component Specification Diagram for both 2D and 3D user interfaces is made in the next paragraphs.

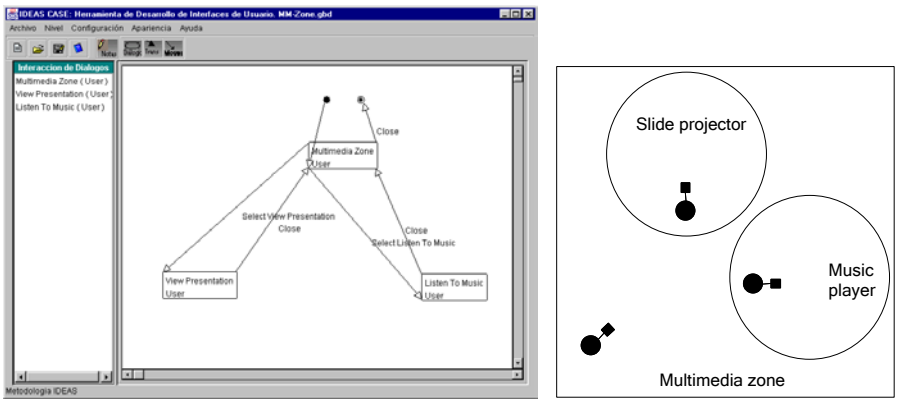

Fig. 5. Navigation Diagrams: 2D user interface (left), 3D user interface (right)

On the one hand, the *Navigation Diagram* allow the designer to specify the different interaction spaces that are derived from the task analysis. In the case of a 2D user interface (Fig. 5, left), this diagram shows the different windows and the links that the user follows to navigate from one to another in order to complete his or her tasks. In our case of study, a main window, "Multimedia Zone", where two options are given to the user, either "View a Presentation" or "Listen to Music". Selecting one takes the user to the window which corresponds to the desired task.

In contrast, the Navigation Diagram that is used to sketch a 3D user interface takes the form of a map (Fig. 5, right). This map relates the tasks and subtasks of the Dialog Model to places in the 3D space. This map sketches the layout of the scene from a gods-eye view of the world, looking down onto the Earth. This is a 2D map because, as it can be seen in the 3D environments referenced in the introduction section, many of them constrain the user to navigate along the horizontal plane. This makes sense, as many of them simulate a real environment where gravity keeps the user on the ground, making navigation a more familiar task. In this map, a *Place* represent an area of the space where a task is carried out. A place has also a *Zone of Influence*, which is the region of the space where the user is able to manipulate the interaction objects of the dialog related with that place, and a *Preferred Point of View*, an optimal location of the observer to interact with that dialog. These concepts are similar to the ones introduced in the IBM RealPlaces Guidelines.

In our example, the "Multimedia Zone" is a place which defines a zone of influence (represented with a circle in the map) and a preferred point of view (depicted as a Pinocchio camera). In the same way a task can have subordinated tasks, a place can also include other places, which also creates a hierarchy of zones of influence and points of view that can be used to help the user to navigate through the environment. Thus, the "Multimedia Zone" includes two more places which corresponds to the two main tasks: "View a Presentation" and "Listen to Music". The designer can use several maps to outline the arrangement of higher-level and lower-level tasks in the space, in the same way an architect draws up plans of the same building with different levels of detail.

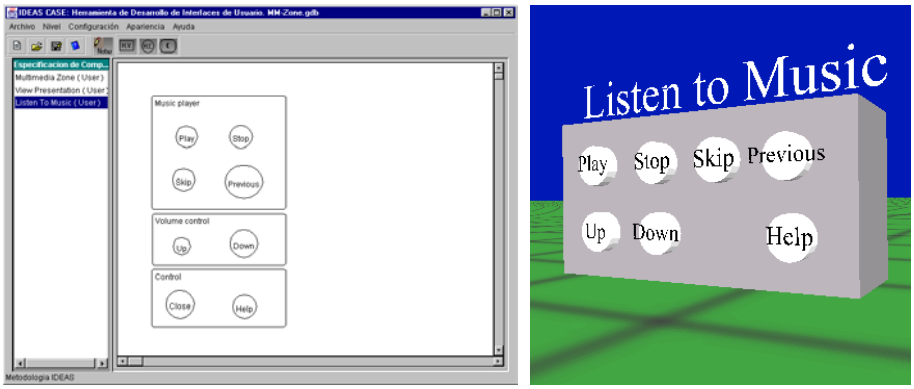

Fig. 6. Component Specification Diagram: 2D user interface (left), 3D user interface (right)

On the other hand, the *Component Specification Diagram* is used to detail the interaction objects that corresponds to each task. Following our comparison, in the case of a 2D user interface (Fig. 6, left) this diagram represents a window where the designer places the Abstract Interaction Objects (AIOs). The IDEAS methodology defines three types of AIOs: Controls, Presentors and Components. A component is used to enclose a set of controls and presentors, which are the elemental mechanisms that enable user-system interaction. Controls and presentors are related to the actions

that the user performs and the data that the system outputs described in the Task Model.

With regard to a 3D user interface, the Component Specification Diagram is also used to place the AIOs that support the tasks the user performs (Fig. 6, right). In this case, a component takes the form a geometric primitive, such as a box, a cylinder, a cone or a sphere, while controls and presentors are place on the surface of the component. Placing AIOs in 3D space can be done using three orthographic projection views, just like CAD systems. However, it can be simplified using a perspective view of the scene, bearing in mind the constraints of the 3D user interface that is being designed. On the one hand, the primitives that represent the components are constraint to the horizontal plane, and its positioning is as simple as placing furniture in a 3D interior design application. On the other hand, controls and presentors are constraint to the surface of a component, which makes its positioning as simple as placing body parts to a Mr. Potato toy.

In any case, by generating the Dialog Model, the designer obtains a approximate version of the user interface which is platform-independent. The next step is to customize the designed user interface for each device, obtaining a running version, task that is carried out at the next abstraction level.

3.4 Presentation

This is the lowest abstraction level, where the Presentation Model is built from the Dialog Model taking into consideration implementation issues. The Dialog Model previously built represents a hierarchy of windows or places, components, controls and presentors. The AIOs specified at design level are now translated into Concrete Interaction Objects (CIOs) depending on the style of interaction and the final platform (Fig. 7, left). For instance, a component with controls can be translated into a panel with buttons in a WIMP desktop interface, a frame with hyperlinks in a Web interface, or a detailed 3D object with 3D buttons in a 3D user interface.

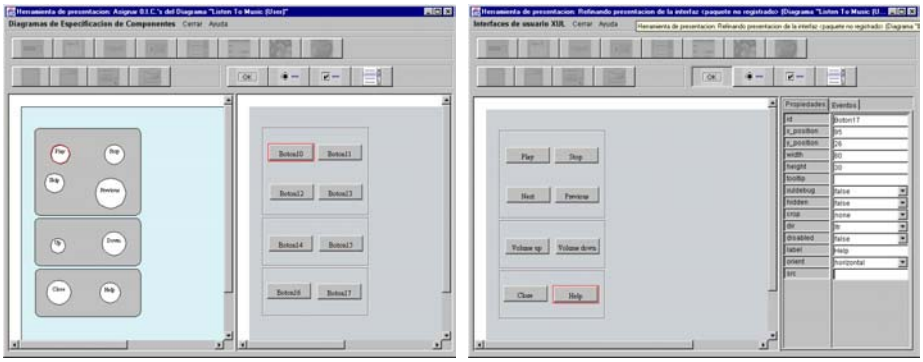

Fig. 7. Translating AIOs into CIOs (left) and fine-tuning of the 2D user interface (right)

Once the translation is made, the designer performs a fine-tune of the user interface, which depends on both usability and aesthetic considerations (Fig. 7, right). For instance, in a 3D user interface, a different color would be applied to each place, so that the user can easily identify the different interaction spaces, while more detailed textures would be mapped to objects depending on the user's preferences.

Fig. 8. 2D Graphical User Interface (left) and 3D Virtual Environment (right)

It should remarked that both the 2D and 3D user interfaces that IDEAS generates (Fig. 8) have the same functionality. In other words, both support the same tasks as a result of the development process. However, it could be argued that the 3D user interface is more complex to use, and so the 2D GUI is preferable. Quite the opposite, the generated 3D user interface does not require much more user input actions than the WIMP user interface. Within the 3D environment, the user can travel from one place to another by simply clicking on the target place, just as the user selects an item on a 2D menu window. Besides, this navigation technique allows the developer to deploy this kind of 3D user interface to any type of platform which provides some kind of pointing device, such as a PDA with a stylus. On the other hand, the user can also move through space using the cursor keys, just as in a first-person 3D game. Indeed, providing a game-like interaction style is one of the goals of current windows 3D front-ends, such as Win3D –previously mentioned– and 3DNA Desktop (www.3dna.net). 3D user interfaces could also be criticized about some well-known problems, such as user disorientation. However, it should be taken into account that disorientation is not a exclusive problem of 3D environments, as it also appears in hypermedia systems such as Web sites, which are 2D user interfaces. In any case, this a research topic that falls out of the scope of this paper.

4 Comparison with other proposals and current research concerns

This section assesses the IDEAS methodology from two distinct points of view. First, it is evaluated as a WIMP user interface development environment, comparing it with other proposals and remarking its main contributions. Then, a similar discussion is made considering IDEAS as a 3D user interface creation tool. Finally, some current research concerns are described.

As a WIMP user interface development tool, IDEAS is a model-based approach which includes a high-level abstract representation of the system under development, using use case, task, domain, user, dialog and presentation models. Such a completeness can only be compared with very few current tools, as for instance MOBI-D [12] in the academic field and OVID [13] in the industrial field, while most of the environments simply focus on the specification of the task and domain models, as for instance TRIDENT [1]. Thus, by means of widely accepted and proven techniques, this methodology allows the user interface designer to refine the conceptual models in every stage of the development. Besides, IDEAS establishes a technologically well-supported relationship between the conceptual representation of the system and the final running user interface, allowing the implementation of the GUI in any platform or environment, either a web environment or a PC, a mobile phone or a PDA.

As a 3D user interface creation tool, IDEAS offers a systematic approach that comprises every activity of the development process. The software engineering notion 3D design is usually limited to a study of user tasks, structural modeling, the coding phase and the evaluation of the 3D user interface [2] [5]. In contrast, IDEAS provides a wider range of abstraction tools, such as the use case, domain and user models for requirements gathering. However, there is another notion of 3D design which deals with aesthetic and usability issues of the final environment, the enhancing and optimizing tasks that are usually carry out by 3D and virtual reality experts [4] [7]. For that purpose, IDEAS also allows the designer to build a conceptual model of the environment at design level, which is useful to sketch the user interface arranging task supportive objects without worrying about details, which are then taken into account at the presentation level of the IDEAS methodology as well.

Taken as a whole, the proposed methodology offers a novel environment which integrates both styles of interaction. However, there are still research issues that deserve further consideration. First, the proposed development process allows the user interface designer to generate both 2D and 3D user interfaces of the same system, sharing both requirements and analysis phases. However, at design level the developer must make a decision on the style of interaction, generating different diagrams based on that decision. Apart form the task hierarchy, it would be desirable to have another abstraction tool that showed the aspects that the two styles of user-system share at design level, if any. The second research issue concerns virtual reality interfaces. IDEAS support the description of 3D environments which can be run either on a PDA, a PC or a virtual reality system. However, virtual reality interaction techniques design is a currently a major topic of research, and require new abstraction tools that IDEAS should provide in order to, for instance, allow the generation of multi-modal interfaces. Both concerns are currently under research.

5 Conclusions

Most software engineering approaches to user interface design have been conceived based on the experience from the development of traditional PC-based systems. However, technology advances are bringing considerable changes to computer and displays. The size spectrum now ranges from large-screen displays to small but powerful PDAs. Besides, 3D graphics are no longer limited to expensive graphics workstations and virtual reality systems, as specialized hardware can be found in most PCs, and new standards are being developed to allow PDAs and mobile devices render 3D applications. The range of user interfaces grows as fast as these advances and user interface engineering must not be left out of this progress.

In this paper, the IDEAS methodology has been presented. IDEAS is a model-based user interface creation tool that guides the UI designer from the first stages of system requirements gathering to the generation of the final user interface for a given platform, evaluating and refining the solution following a iterative process. Taking advantage of the experience from the development of WIMP GUIs, IDEAS offers a number of abstraction tools that not only allows the UI designer to create standard 2D user interfaces but also three-dimensional environments. This feature of IDEAS has been the main focus of this paper, using a case of study to show the details of the development process. In the last section, a comparison with other proposals has been made, and some current research concerns have been outlined.

Acknowledgements

This work is supported in part by the Spanish CICYT TIC 2000-1673-C06-06 and CICYT TIC 2000-1106-C02-02 grants.

References

1. Bodart, F., Hennebert, A.M., Leheureux, J.M., Provot, I., Sacre, B., Vanderdonckt, J.: Towards a Systematic Building of Software Architectures: the TRIDENT Methodological Guide. In: Design, Specification and Verification of Interactive Systems. Springer, Wien (1995) 262-278
2. Bowman, D.A., Kruijff, E., LaViola, J.J., Poupyrev, I.: An Introduction to 3-D Interface Design. Presence, Vol. 10, No. 1 (2001) 96-10
3. Cockburn, A.: Writing Effective Use Cases. Addison-Wesley (2001)
4. Fencott, C.: Towards a Design Methodology for Virtual Environments. User Centered Design and Implementation of Virtual Environments (1999) Available at the workshop Website: http://www.cs.york.ac.uk/hci/kings_manor_workshops/UCDIVE/
5. Gabbard, J., Hix, D., Swan, J.: User-centered design and evaluation of virtual environments. IEEE Computer Graphics & Applications, Vol. 19, No.6 (1999) 51-59
6. IBM RealPlaces Design Guide. Available on-line at IBM Ease-of-Use Website: http://www-3.ibm.com/ibm/easy/eou_ext.nsf/Publish/580

7. Kaur, K.: Designing virtual environments for usability. PhD thesis, Centre for HCI Design, City University, London (1998)
8. Letelier, P., Ramos, I., Sánchez, P., Pastor, O.: OASIS version 3.0: A Formal Approach for Object Oriented Conceptual Modelling. SPUPV-98.4011. Edited by the Technical University of Valencia, Spain (1998)
9. Lozano, M., González, P., Montero, F., Molina, J. P., Ramos, I.: A Graphical User Interface Development Tool. Proceedings of the 16th British HCI Conference, Vol. 2 (2002) 62-65
10. Lozano, M., Ramos, I., González, P.: User Interface Specification and Modeling in an Object Oriented Environment for Automatic Software Development. IEEE 34th International Conference on Technology of Object-Oriented Languages and Systems, TOOLS-USA (2000) 373-381
11. Molina, J.P., González, P., Lozano, M.: Developing 3D user interfaces using the IDEAS Tool: A case of study. Accepted for presentation at the HCI International Conference 2003
12. Puerta, A.: The Mecano Project: Comprehensive and Integrated Support for Model-Based Interface Development. CADUI, Namur University Press (1996) 19-36
13. Roberts, D., Berry, D., Isensee, S., Mullaly, J.: Designing for the User with OVID. Macmillan Technical Publishing (1998)
14. Sommerville, I.: Software Engineering. 5th edition, Addison-Wesley (1999)
15. van Dam, A.: Post-WIMP User Interfaces. Communications of the ACM, Vol. 40, No. 2 (1997) 63-67