HAL
open science

A Software Engineering Method for the Design of Mixed Reality Systems

Sophie Dupuy-Chessa, Guillaume Godet-Bar, Jorge-Luis Pérez-Medina, Dominique Rieu, David Juras

To cite this version:

Sophie Dupuy-Chessa, Guillaume Godet-Bar, Jorge-Luis Pérez-Medina, Dominique Rieu, David Juras. A Software Engineering Method for the Design of Mixed Reality Systems. E. Dubois and P. Gray and L. Nigay. The Engineering of Mixed Reality Systems, chapter 15, 15, Springer, pp.313-334, 2009. hal-00953692

HAL Id: hal-00953692

https://inria.hal.science/hal-00953692v1

Submitted on 28 Feb 2014

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

A Software Engineering Method for the Design of Mixed Reality Systems

S. Dupuy-Chessa, G. Godet-Bar, J.-L. Pérez-Medina, D. Rieu, D. Juras
Laboratory of Informatics of Grenoble, Grenoble Université
385 rue de la bibliothèque, B.P. 53
38041 Grenoble Cedex 9, FRANCE
{Firstname.Lastname}@imag.fr

Abstract. The domain of Mixed Reality systems is currently making decisive advances on a daily basis. However, the knowledge and know-how of HCI scientists and interaction engineers, used in the design of such systems, is not well understood. This paper addresses this issue by proposing a software engineering method that couples a process for designing Mixed Reality interaction with a process for developing the functional core. Our development method features a Y-shaped development cycle that separates the description of functional requirements and their analysis from the study of technical requirements of the application. These sub-processes produce Business Objects and Interactional Objects, which are connected to produce a complete Mixed Reality system. The whole process is presented via a case study, with a particular emphasis on the design of the interactive solution.

Keywords: Mixed Reality, interaction design, model, functional core, process

1 Introduction

Mixed Reality systems, which include tangible user interfaces, augmented reality and augmented virtuality seek to smoothly merge physical and digital worlds to improve usability. Many prototypes in various domains [1][2][3][4] have been developed to demonstrate the technical feasibility and the interest of such interaction techniques. But developing mixed systems often remains based on the realization of ad hoc solutions, which do not facilitate the reusability of the systems and do not capitalize on previous solutions. It is now required to focus on design approaches rather than adopting technology-driven approaches only.

A design approach is based on the study of users' requirements in order to propose a suitable solution. It needs approaches, like models or processes, to reason about solutions, compare them and choose the most appropriate one according to users' needs and good practice. However, good practice for Mixed Reality systems is still being identified. Existing knowledge and know-how must be exploited in order to be integrated into industrial practice. At the present time, software designers and developers tend to build graphical user interfaces, which are easy to develop, but which are not always adapted to the interaction situation. Therefore exploiting and

spreading design knowledge and know-how about Mixed Reality systems design can be a way to facilitate the acceptance of Mixed Reality systems in industry.

The first step to make use of design knowledge is to integrate it into models. Given the particularities of the Mixed Reality systems domain, current design approaches in Human Computer Interaction (HCI) are no longer sufficient. Several proposals (for instance, by Dubois et al. [5] and by Coutrix and Nigay [6]) have been made for guiding the design of Mixed Reality systems. They provide a rationale for how to combine physical and digital worlds. They are used in addition to traditional user-system task description in order to identify physical and digital objects involved in interaction techniques and the boundaries between real and virtual worlds.

A second step in helping designers is proposing design processes. The lack of maturity of the mixed systems domain suggests processes based on experimentations, as presented by Kulas et al. [7], rather than models. Nevertheless the use of models and their associated processes, which has been described by Nigay et al. [8] or by Gauffre et al. [9], facilitates the link with classical software engineering (SE) practices and the integration of new interaction techniques into applications.

Our goal is to propose a design method based on models integrating both an SE method for the development of the functional core and HCI practices for designing the interaction. Compatibility between design methods for interactive systems and for the functional core is a recurring problem that has already been subject to specific studies by Tarby and Barthet [10] and Lim and Long [11]. In particular, Gulliksen and Göransson [12] and Sousa and Furtado [13] propose extending the Rational Unified Process with the design of interaction, in a user-centred approach. Constantine et al. [14] also describe a process unifying the design of interaction and that of the functional core but in a usage-centred approach. These studies illustrate an interest in considering both the interaction and functional aspects while designing a system. Nevertheless, none of these studies addresses Mixed Reality-specific aspects, such as the integration of interaction devices like Head-Mounted Displays, positioning systems, etc. Moreover, they offer a weak formalization of the proposed processes, which renders their application difficult for designers.

The second section introduces software engineering principles that are used as a foundation for the design of Mixed Reality systems and the Symphony Y-cycle software method [15] that we will use as a medium for integrating those principles. We then describe how a Y-shaped development cycle can be applied to the design of Mixed Reality systems by using specific models and processes. The functional aspects, which aim at designing a functional and interactive solution without considering the technical aspects, are described in Section 3. Then Section 4 describes the technical analysis that aims at choosing the most appropriate devices, software architectures and platforms for supporting the Mixed Reality system under development. In Section 5, the junction of both concerns poses the challenge of merging technical choices with the models elaborated in the functional analysis. Finally, we conclude by considering lessons learnt from the use of the method. We also give details of evaluations of the method, both carried out and envisaged.

2 Extending a SE Method for Mixed Reality Systems

2.1 Extending Symphony for the Design of Mixed Reality Systems

Our approach is mainly based on the practices of the Rational Unified Process [16]. We apply three of them to the design of Mixed Reality systems:

- The process is **iterative**. Given today's sophisticated software systems, it is not possible to define the entire problem, design the entire solution, build the software and then test the product at the end. An iterative approach is required for allowing an increasing understanding of the problem through successive refinements, and for incrementally growing an effective solution over multiple iterations. Thus, we envisage building Mixed Reality systems incrementally and iteratively. Additionally, sub-processes are identified for allowing shorter iterations when the activities focus on crucial elements of the system, such as the design of the interaction.
- The Symphony process is driven by **use cases and scenarios** to manage requirements. We choose a compatible approach based on scenarios [7] to complement the process with design steps specific to the design of Mixed Reality systems (for instance, the choice of interaction devices or the design of interaction techniques). These scenarios are used as a pivot model for all specialists.
- The process uses **graphical models of the** software to capture the structure and behaviour of components. Graphical models such as UML diagrams help to communicate different aspects of the software, enable a developer to see how the elements of the system fit together, maintain consistency between a design and its implementation and promote unambiguous communication among developers. For the design of Mixed Reality systems, we complement classical HCI models like task trees [21] with specific models such as ASUR [5].

Additionally, the original Symphony method was centered on the early use of reusable and reused components. It provides a systematic approach for defining a solution using new and existing components. When the solution is precise enough to be described by computerized objects, we propose to structure the interaction space with interactional objects [17], which are user interface-oriented components. In parallel, the business is designed into components called Business Objects.

During the process, models and scenarios are continuously refined. SE and HCI-oriented activities are realized either in cooperation or in parallel, by design actors specialized either in SE and/or HCI. All actors collaborate in order to ensure consistency of adopted design options.

However Mixed Reality system design is not yet a fully mastered task. The design process needs to be flexible enough to evolve over time. Therefore, our Symphony method extended for Mixed Reality systems contains black box activities which correspond to not fully mastered practices: a black box only describes the activity's principles and its purpose without describing a specific process. In such a case, we let the designer use her usual practices to achieve the goal. For instance, "preparing user experiments" is a black box activity, which describes the desired goal without making explicit the way in which it is achieved.

The method also proposes extension mechanisms, in particular, alternatives. Indeed, we can imagine alternatives corresponding to different practices. For instance, different solutions can be envisioned to realize the activity « analyzing users' tasks». They are specified as alternative paths in our process.

Additionally, Symphony is based on a Y-shaped development cycle (**Fig. 1**). It is organized into three design branches, similar to 2TUP [19]. For each iteration, the whole development cycle is applied for each functional unit of the system under development [15]:

- The **functional (left) branch** corresponds to the traditional task of domain and user requirements modeling, independently from technical aspects. Considering the design of a Mixed Reality system, this branch is based on an extension of the process defined by [8]. It includes interaction scenarios, task analysis, interaction modality choices and mock-ups. This branch ends by structuring the domains with Interactional and Business objects required to implement the Mixed Reality system.
- The **technical (right) branch** allows developers to design both the technical and software architectures. It also combines all the constraints and technical choices with relation to security, load balancing, etc. In this paper, we limit the technical choices to those related to Mixed Reality systems support, that is, the choice of devices and the choice of the global architecture.
- The **central branch** integrates the technical and functional branches into the design model, which merges the analysis model with the applicative architecture and details of traceable components. It shows how the interaction components are structured and distributed on the various devices and how they are linked with the functional concepts.

In the rest of the paper, the extended Symphony method and its design principles will be detailed in a case study, which concerns the creation of an inventory of premise fixtures.

2.2 Case Study

Describing the state of a whole premise can be a long and difficult task. In particular, real-estate agents need to qualify damage in terms of its nature, location and extent when tenants move in as well as when they move out. Typically, this evaluation is carried out on paper or using basic digital forms. Additionally, an agent may have to evaluate changes in a premises based on someone else's previous notes, which may be incomplete or imprecise.

Identifying responsibilities for particular damage is yet another chore, which regularly leads to contentious issues between landlords, tenants and real-estate agencies.

In order to address these issues, one solution may be to consider improving the computerization of the process of making an inventory of premise fixtures. In particular, providing better ways to characterize damage and to improve the integration of the process into the real-estate agency's information system, as well as its usability, all of which could add considerable value to this activity.

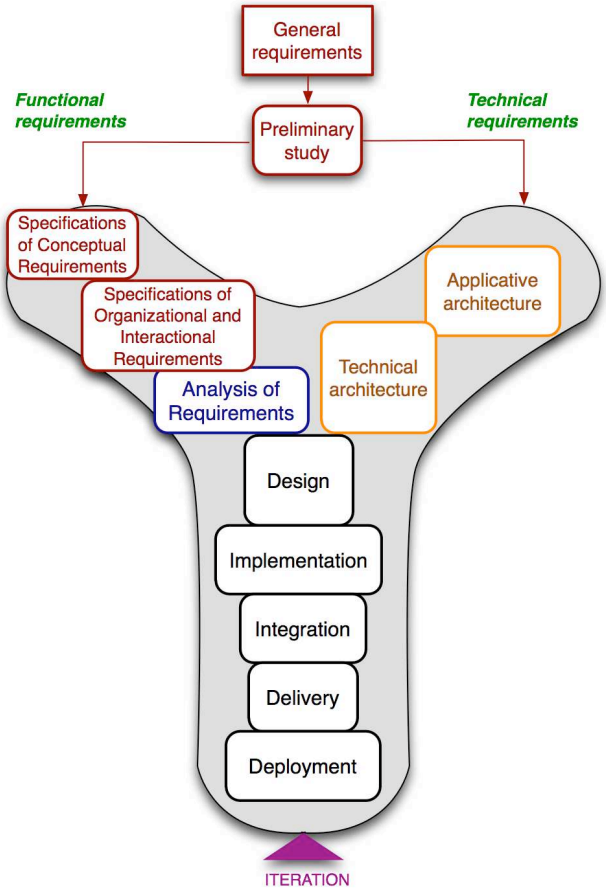

Fig. 1. Symphony design phases

In the following sections, we detail the application of our method to this case study.

3 The Functional Branch

3.1 Introduction

This section presents the development activities of the extended Symphony method's functional branch, from the preliminary study to the requirements analysis.

In particular, the focus of our concern is on the collaborations that HCI specialists may have with SE experts. Therefore, we provide an excerpt of our development process centered on these aspects, in **Fig. 2**. Please note that, for the sake of conciseness, both the SE and HCI processes have been greatly simplified. For instance, we present collaborations between domains (i.e., a group of actors sharing the same concerns), not between the particular actors; in fact, several collaborations occur between the functional roles of each domain during the application's development cycle.

The activities that appear across the swimlanes correspond to cooperations: the development actors must work together to produce a common product. For instance, the "Description of weaving model between Business Objects and Interactional Objects" involves having both the HCI and SE experts identifying which Interactional Objects correspond to projections of Business Objects and laying this down in a specific model.

Coordination activities are not represented as such in the central swimlane because the experts do not need to produce a common product. On the contrary, they must

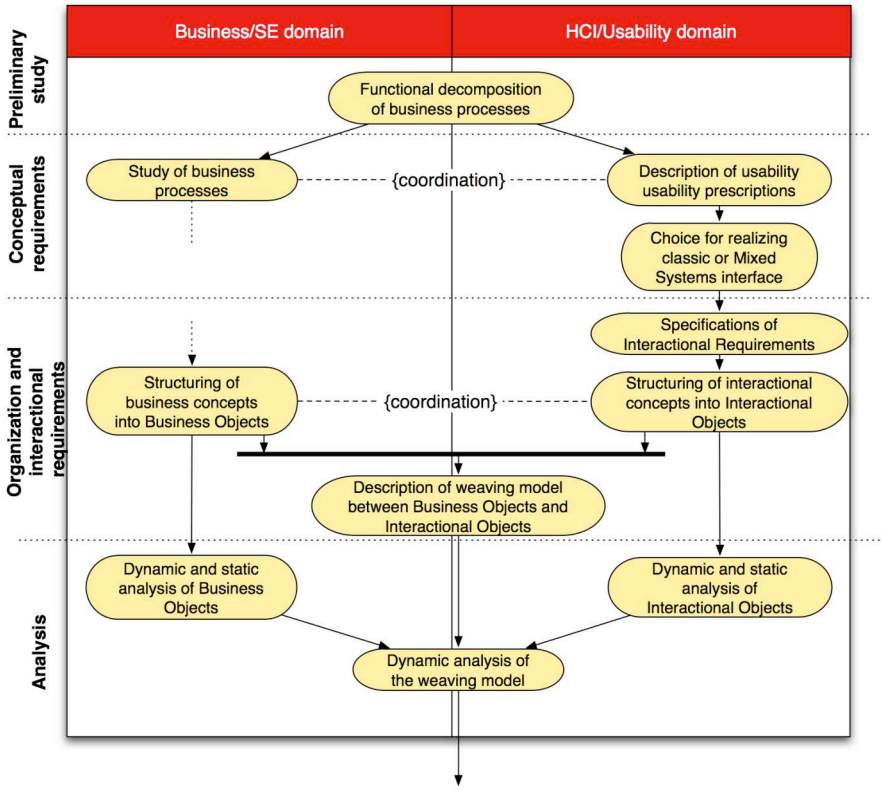

Fig. 2. Collaborations between the HCI and SE domains during the functional development

compare products from their respective domains in order to validate their design choices. In our example, the coordination between the "Structuring of business concepts into Business Objects" and "Structuring of interaction concepts into Interactional Objects" implies that the SE and HCI experts identify whether they may need to modify their model in order to facilitate the ulterior weaving of the two models. The details of these activities, as well as the possible business evolutions they may trigger, have been covered in [18].

3.2 Initiating the Development

Before starting a full development iteration, a preliminary study of the business is realized. Its aim is to obtain a functional decomposition of the business as practiced by the client, in order to identify business processes and their participants. A business process is defined as a collection of activities taken in response to a specific input or event, which produces a value-added output for the process' client. For instance, our inventory of fixtures case study corresponds to a fragment of a larger real-estate management business, in which several business processes can be identified: "management of tenants", "management of landholders" and "management of inventories of fixtures". An essential issue of this phase is therefore to identify the stakeholders' value, for each process.

This study is described using high-level scenarios in natural language shared by all development specialists (including usability and HCI specialists), so as to provide a unified vision of the business, through the description of its constitutive processes.

In this phase, the usability specialist collaborates with the business expert for capturing prescriptions based on the current implementation of the business processes, as well as for defining a reference frame of the application's users, using the participants identified by the business specialist during the writing of the scenarios.

3.3 Conceptual Specifications of Requirements

In the original description of the Symphony method, this phase essentially comprises the detailing of the subsystems that constitute the different Business Processes and of the actors that intervene at this level, in terms of sequence diagrams and scenarios.

In our extension of the method for HCI, we associate the usability specialist with the description of the Business Processes' scenarios. In collaboration with the HCI specialist, and based on the scenarios and usability prescriptions (from the preliminary study), the usability expert determines the types of interaction that may be envisaged for the application, such as Mixed Reality, post-WIMP or classical interfaces according to the context and needs. In collaboration with the other actors from the method and stakeholders, an estimation of the added value, cost and risks of development associated with the interaction choice is realized, for each subsystem of the future application. Considering our example, the "Realization of an inventory of fixtures" subsystem is a good candidate for a Mixed Reality interaction for a variety of reasons including:

- The case study typically features a situation where the user cannot use a desktop workstation efficiently while realizing the activity,
- Textual descriptions of damage are both imprecise and tedious to use, especially for describing the evolution of damage over time and space,
- Several manual activities are required for thoroughly describing the damage, e.g. taking measures, photographs etc.,
- The data gathered during the inventory of fixtures cannot be directly entered into the Information System (except if the user operates a wireless handheld device),
- Standard handheld device such as PDAs would only allow the use of textual, form-filling approaches, with the aforementioned limitations.

3.4 Organizational and Interaction-oriented Specification of Requirements

Once the Business Processes are identified and specified, the Organizational and Interaction-oriented Specification of Requirements must determine the "who does what and when" of the future system. Concerning the business domain, the SE specialist essentially identifies Use Cases from the previous descriptions of Business Processes, and from there refines business concepts into functional components called Business Objects.

We have extended this phase to include the specifications of the interaction, based on the choices on the style of interaction made during the previous phase.

Three essential aspects are focused on in this phase: firstly, the constitution of the "Interaction Record" product, which integrates the synthesis of all the choices made in terms of HCI; secondly, the realization of prototypes, based on "frozen" versions of the Interaction Record; thirdly, the elaboration of usability tests, which use the Interaction Record and the prototypes as a basis.

The design of these three aspects is contained as a sub-process within a highly iterative loop, which allows testing multiple interactive solutions, and therefore identifying usability and technological issues before the actual integration of the solution into the development cycle.

The construction of the Interaction Record is initiated by creating a projection of the users' tasks in the application under development: the HCI specialist describes "Abstract Projected Scenarios" [8], based on usability prescriptions proposed by the usability expert. However, at this point the description remains anchored in a business-oriented vision of the user task, as we can see in **Table 1**.

Table 1. Abstract Projected Scenario for the "Create damage report" task

Theme(s)	{Localization, Data input}
Participant(s)	Inventory of Fixtures Expert
Post condition	The damage is observed and recorded in the Inventory of Fixtures
(…) The Expert enters a *room*. Her *position* is indicated on the *premises plan*. The past inventory of fixtures does not indicate any damage or wear-out (*damage*) that needs to be checked in this room. She walks around the room and notices a dark spot on one of the walls. She creates a new damage report, describes the observed damage, its position on the premise's plan and takes a *recording* (photograph or video) of the damage and its context(…).	

From this basis, the HCI specialist moves her focus to three essential and interdependent aspects of the future interaction: the description of interaction artefacts, of interaction techniques and of the device classes for supporting the interaction. This latter activity is optional, given that it is only necessary when designing Mixed Reality interfaces. Additionally, this activity is undertaken in collaboration with the usability expert.

We expect the following types of results from these activities:

- A textual description of the interaction artefacts, including lists of attributes for each of these artefacts. A list of physical objects that will be tracked and used by the future system may also be provided (even though at this point we do not need to detail the tracking technology involved),
- Dynamic diagrams may be used for describing the interaction techniques. Both physical action-level user task models (e.g. ConcurTaskTrees [21]) and more software-oriented models (e.g. UML statecharts [16]) may be used. For instance, the following interaction technique for displaying a menu and selecting an option may be described using a task model: the expert makes a 1-second pressure on the Tactile input; a menu appears on the Augmented Vision display with a default highlight on the first option; the Expert then makes up or down dragging gestures to move the highlight; the Expert releases the pressure to confirm the selection,
- Static diagrams for describing device and data-flow organization for augmented reality interactions, such as ASUR [5], complement the description of artefacts and interaction techniques. **Fig. 3** presents an example of such a diagram for the "Create a damage report" task.

The user, identified as the Expert, is wearing an Augmented Vision Display ("==" relation), which provides information (→ relation symbolizes physical or numerical data transfer) about the Marker and Vocal Note virtual objects. Additionally, the Mobile Display device, which is linked to the Tactile input ("==" relation), provides information about the Marker, Vocal Note and Situational Mesh (i.e., the 3D model of the premises) virtual objects. The Marker is a numerical representation ("-->" relation) of physical damage in the physical world. The Expert can interact with the Marker and Situational Mesh objects using Tactile Input. She can interact with the Vocal Note using a Vocal Command input. The Expert's position in the premises is deduced from the Positioning system, which sends information to the virtual objects for updating the virtual scene as the Expert moves.

The descriptions of artefacts, interaction techniques and classes of devices are then integrated into "Concrete Projected Scenarios" [8], which put into play all the concepts elaborated previously, as an evolution of the Abstract Projected Scenarios. An extract of Concrete Projected Scenario is presented in **Table 2**.

Finally, high-level user task models are deduced from the Concrete Projected Scenarios, both in order to facilitate the the evaluation process and to validate the constructed models.

Following each iteration of the Interaction Record's development, we recommend the construction of paper and software prototypes (Flash or Powerpoint simulations, HCI tryouts…) putting into play the products of the Interaction-oriented Specifications. Beyond the advantage of exploring design solutions, the prototypes allow the usability expert to set up usability evaluations for validating the specifications. **Fig. 4** presents an early prototype for the Augmented Inventory of Fixtures application, with both the Expert wearing an augmented vision device and the data displayed.

From these different products of the interface's design process, the usability expert compiles recommendations and rules for the future user interface, such as its graphic chart, cultural and physical constraints, etc. Based on the prototypes, the Interaction Record and the Concrete Projected Scenarios, the usability expert may start elaborating validation tests for all the products of the Interaction-oriented Specification. We use "Black box activities" for describing these steps, as described in Section 2.

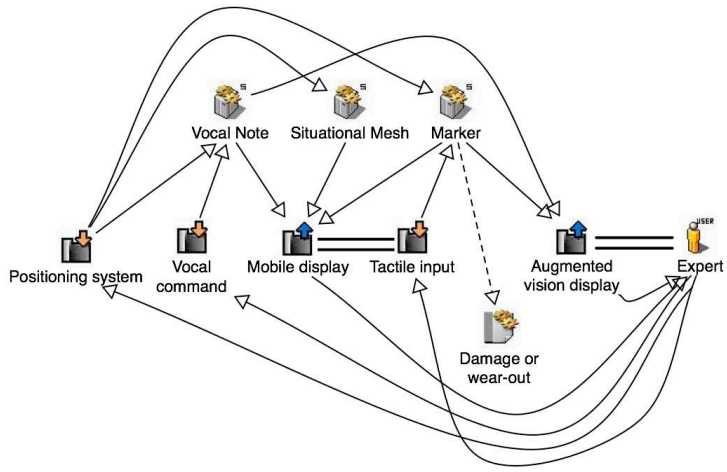

Fig. 3. ASUR model representing the "Create damage" task

Table 2. Concrete Projected Scenario for the "Create damage" task

Supporting device(s)	{Mobile tactile display, Augmented vision device, Vocal input device, Positioning system}
Interaction artefact(s)	{Marker, Situational Mesh, Vocal note}
(…) The Expert enters a *room*. Her *position* is indicated on the *Situational Mesh*, which is partially displayed on the *Mobile tactile display*. The past inventory of fixtures does not indicate any damage or wear (*Marker object*) that needs to be checked in this room. She walks around the room, notices a dark spot on one of the walls. She creates a new *Marker* and positions, orients and scales it, using the *Mobile tactile display*. Then she locks the *Marker* and describes the damage by making a *Vocal note with* the *Vocal input device* (…)	

Depending on the results of the usability tests, a new iteration of the Specification of Interaction-oriented Requirements may be undertaken, returning to the Abstract Projected Scenarios if necessary.

Finally, once the Interaction-oriented Requirements are validated, the HCI expert proceeds with the elicitation of the Interactional Objects, deduced from the interaction concepts identified previously. The criteria for this selection are based on the concepts' granularity and density (i.e., if a concept is anecdotally used or is described by only a few attributes, then it may not be a pertinent choice for an Interactional Object).

As was mentioned in **Fig. 2**, a cooperation activity between HCI and SE experts aims at mapping these Interactional Objects to the Business Object they represent through a "represent" relationship. For instance, the "Marker" Interactional Object is linked to the "Damage" object, through a "represent" relationship (see **Fig. 5**).

In the following section, we detail how the Symphony Objects that have been identified previously are refined and detailed.

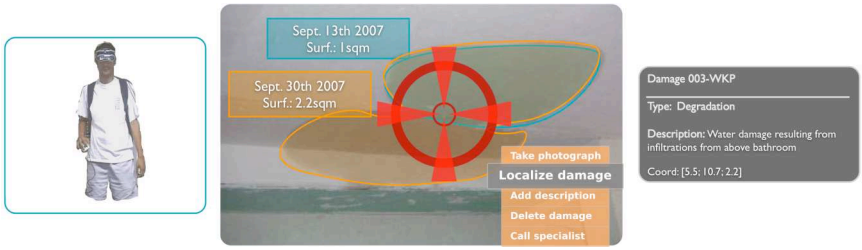

Fig. 4. Augmented Inventory of Fixtures prototype

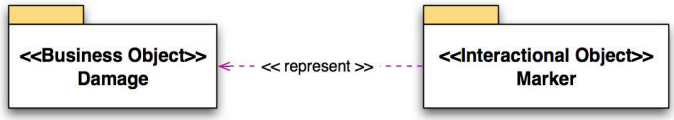

Fig. 5. "represent" relationship between an Interactional Object and a Business Object

3.5 Analysis

This phase describes how the Symphony Objects are structured, in terms of services and attributes, and how they behave. The latter aspect is detailed in a dynamic analysis of the system, while the former is elaborated in a static analysis. Following these analyses, details of the communication between the business and interaction spaces, indicated by the "represent" relationship, are described.

Concerning the business space, the dynamic analysis consists of refining the Use Cases identified during the previous phase into scenarios and UML sequence diagrams, for identifying business services. These services are themselves refined during the static analysis into the Symphony tripartite structure (i.e. methods and attributes are identified, see **Fig. 6**). The left-most part describes the services proposed by the object, the central part describes the implementation of the services, and the right-most part (not used in this example) describes the collaborations the object needs to set up in order to function. A "use" dependency relationship, allows Symphony Objects to be organised (e.g., the "Inventory of fixtures" depends on the "Damage" concept during its lifecycle).

Concerning the interaction space, the dynamic analysis is similar to that of the business space, but based on the high-level user task models elaborated at the end of the Specification of Organizational and Interaction-oriented Requirements phase. They are complemented by UML statechart diagrams for describing the objects' lifecycles. For instance, the "Marker" Interactional Object can be described using a simple two-state statechart diagram: it may be "Locked" and immovable, or "Unlocked" and moveable.

Similar to the business space, the static analysis refines these studies into tripartite components. We propose the same, model-oriented representation of Business Objects and Interactional Objects (i.e., "Symphony Objects") [17], in order to facilitate their integration into MDE tools, as well as their implementation.

At this point, both the business and interaction spaces have been described in parallel, from an abstract (i.e., from technological concerns) point of view. Now HCI and SE specialists need to detail the dynamic semantics of the "represent"

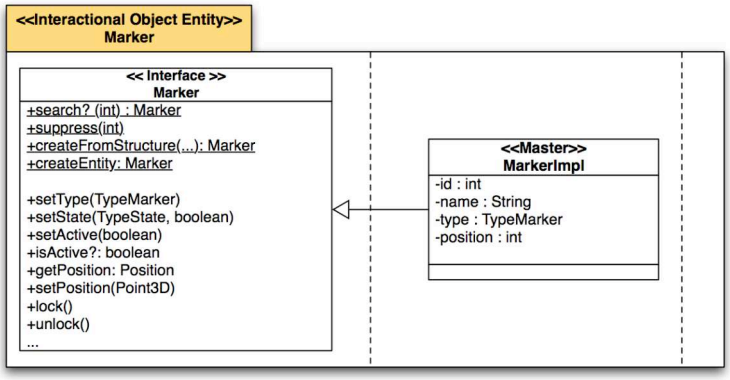

Fig. 6. Interactional Object example

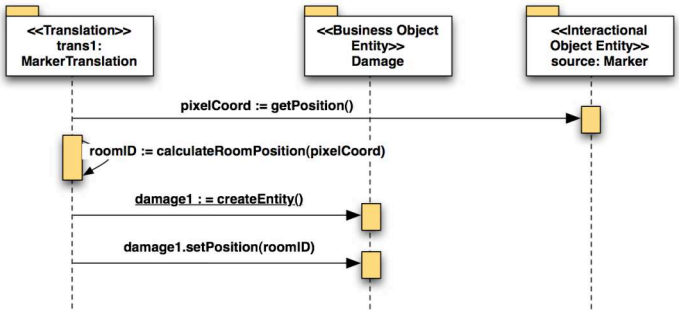

Fig. 7. Translation semantics corresponding to the "createMarker" event

relationships that were drawn during the Specification phases.

It is first necessary to identify the services from the interaction space that may have an impact on the business space. For instance, creating a “Marker” object in the interaction space implies creating the corresponding “Damage” object in the business space. Second, the translation from one conceptual space to the next needs to be described. In our example, it is necessary to convert the pixel coordinates of the “Marker” into the architectural measures of the premises plan.

These translations from one conceptual space to another are managed by “Translation” objects. There is one instance of such object for each instance of connection between Interactional Objects and Business Objects. In the Analysis phase, the Translation objects are described using sequence diagrams for each case of Interactional-Business Object communication. **Fig. 7** illustrates the consequence of the “create Marker” interaction event in terms of its translation into the business space. Note that the reference to the Marker object (i.e. the “source” instance) is assumed to be registered into the Translation object. Details of this mechanism are presented in Section 5.

3.6 Main points discussed

The extension of the Symphony method for the development of Augmented Reality systems capitalizes on the features we discussed in Section 2.1. Similarly to what is achieved in terms of the business description, we have proposed the following principles for the development of the interaction space:

1. Early description of interaction components (i.e., “Interactional Objects”), during the specification of requirements,
2. Structuring of interaction components based on the same tripartite structure as the business components (i.e., “Business Objects”),
3. Parallel and collaborative description of the business and interaction specifications and analysis,
4. Late connection between the business and interaction spaces, using Translation objects.

As we can see, principles 1 and 2 reproduce what was initially proposed for the business space. However, principles 3 and 4 aim at allowing HCI experts and

usability experts to design ambitious interfaces early enough in the development cycle for permitting their integration into the final system. Additionally, the design of the user interface involves regular prototyping and evaluation activities, contained in a highly iterative sub-process, as recommended by ISO 13407 for UCD (User-Centered Design).

As a consequence of the application of these principles, the end of the development of the functional branch provides developers with two exhaustively detailed sets of products structuring the business and interaction spaces, as well as the necessary elements for realizing the junction between these spaces (i.e. the Translation classes).

Additionally, we have integrated regular collaborations between the HCI specialist and the usability expert during all the phases of the functional branch. The usability specialist herself involves future users in the design process (for instance through the observation of the business practices and the usability tests). These practices greatly increase the overall usability and efficiency of the future system. In this respect, we follow once again the indications of ISO 13407 for UCD.

4 The Technical Branch

The technical branch of the original Symphony method allows developers to design the applicative and technical architectures. Their goal is the analysis of all the constraints and technical choices related to security, pervasiveness, and load balancing, for example. We saw previously that in order to design Mixed Reality systems, our method considers the choice of techniques of interaction, interaction artefacts and classes of devices. Consequently, this has led us to propose extensions for the technical branch.

The applicative architecture corresponds to the organization of technical and functional components amongst applicative tiers, as well as the description of the software architectures used for supporting the execution of Business and Interactional Objects detailed during the Analysis phase. Additionally, patterns and design rules are identified and defined for the applicative architecture. The technical architecture corresponds to the hardware and technology (e.g., frameworks) solutions that will allow the application to be run. Finally, note that both phases are realized concurrently with the functional requirements.

4.1 Description of the Applicative Architecture

The goal of this phase is to identify and describe the rationale concerning the selection of the software architecture, which must allow the integration of the technical and functional components. First, the distribution of components amongst applicative layers is described, before a software architecture for efficiently supporting the interaction is superimposed on the applicative layers.

Description of applicative layers. The original Symphony method recommends a placing the components in a five-tier architecture, for classic systems. The tiers are built following the Layer pattern [22], which enables designing, developing and

testing each tier independently from the others. The extension of the method preserves this decomposition of the Information System, and describes the distribution of Business Objects and Interactional Objects amongst the five layers. This applicative choice facilitates the maintenance, reuse and evolution of future software.

Description of the software architecture. The original Symphony method recommends the MVC pattern for the design of the Presentation layer, which allows isolating the business concerns of the application from the interface's control and view concerns. However, this solution needs to be complemented for taking into account the distribution of Interactional Objects, as well as the technical constraints imposed by Mixed Reality systems, such as multimedia rendering loops. **Fig. 8** illustrates how we envisage this dispatch of components amongst the applicative tiers, using an adapted MVC software architecture:

- The Interactional Objects correspond to the abstract, logical part of the MVC's "View", while the "OpenGL Client" corresponds to the technical implementation of the Interactional Objects; the "Controller" is an adaptation of the View to the user input (e.g., pressing a given button triggers a notification, which allows matching this event with a given action),
- The Business Objects correspond to the "Model" part of the MVC pattern,
- The Translation objects correspond to facets of MVC's "Controller" in the sense that they manage the bridge between the "Model" and "View" aspects.

4.2 Description of the technical architecture

This section describes the technical analysis for choosing the most adequate devices and platforms for supporting Mixed Reality systems.

Choice of interaction devices. Once the need to develop a Mixed System (in this case, an Augmented Reality System) is identified (see Section 3.3), it is necessary to choose the most satisfactory interaction devices. This selection must be made on the basis of characteristics of the generic devices identified during the Specification of Organizational and Interaction-oriented Requirements (see Section 3.4).

Our method for facilitating the selection of devices is based on the QOC[1] notation

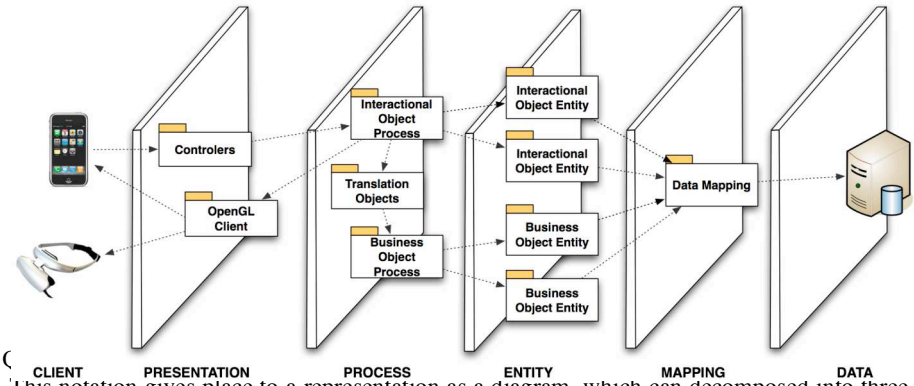

Fig. 8. Applicative architecture for Mixed Reality systems

[1] QOC: This notation gives place to a representation as a diagram, which can decomposed into three columns, one for each element of the notation (questions, options, criteria), and links between the elements of these columns.

[22]. It enables designers to classify and to justify the design decisions. In the context of the inventory of fixtures, the study of the interaction has determined the need for an augmented vision display, a tactile input and mobile display, a positioning system (i.e. orientation and localization) and a vocal command device (see **Fig. 3**). For the sake of conciseness we will only present in **Fig. 9Erreur ! Source du renvoi introuvable.** the selection of the augmented vision display.

Based on the QOC diagram, the usability expert has determined that the most adequate device for the inventory of fixtures is an HMD device: it allows user mobility, guarantees low energy consumption, and offers an acceptable image quality, low weight and a wide space of visualization.

Description of the interaction's devices. This activity consists of detailing the technical specifications of each interaction device chosen during the previous activity. For instance, **Table 3** shows the technical specifications of the HMD device.

4.3 Main points discussed

In this section we have presented a methodological guide adapted to HCI concerns

Table 3. Description of the interactive devices

Device	Technical specifications
HMD	Model AddVisor 150. Full color image on 1280x1024 pixels (SXGA), excellent image quality, high brightness and contrast. Superposed image with up to 35 % see-through or fully immersed. Designed for a 46 degree diagonal 100% overlap field of view. Low weight.

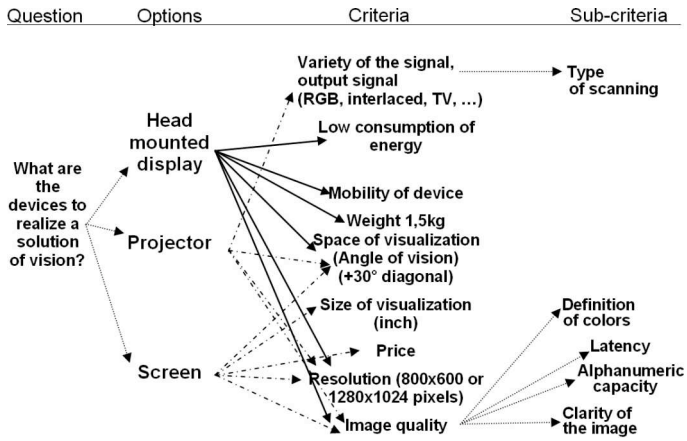

Fig. 9. Selection criteria for the vision support device

that aims at selecting the most adequate device, software architectures and platforms to support the design of Mixed Reality systems, based on the use of principles underlying recognized software architectures. We use the QOC notation, which

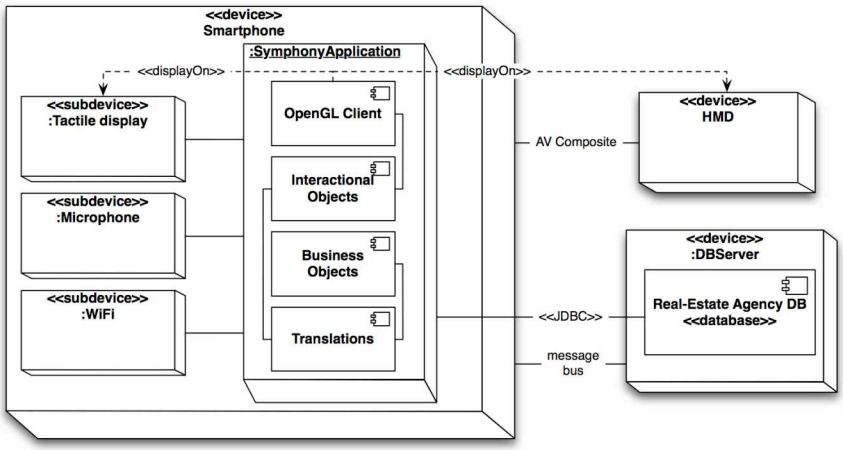

Fig. 10. Deployment diagram for the functional and technical components of the Augmented Inventory of Fixtures application

allows technical aspects to be organized and classified, for facilitating the choice of devices and software architectures adapted to Mixed Reality systems.

5 The Junction of the Functional and Technical Branches

5.1 Design

Two essential aspects need to be addressed at this point in the development cycle: the blending of the functional analysis with the technical choices (i.e., the organization of the functional and non-functional software components), and the deployment of these components on the technical infrastructure.

For the sake of conciseness, the latter point is not extensively discussed in this paper: the choices made are in fact summarized in **Fig. 10**. We use a tactile smartphone device for running the application, which is linked to the real-estate agency's database through a wireless message bus. Additionally, the "Vocal Command" and "Positioning System" mentioned in **Fig. 3** correspond to the internal microphone and Wi-Fi/accelerometers (used for position triangulation purposes) sub-devices of the smartphone. The "Tactile Input" is managed by the "Tactile Display" sub-device. An OpenGL client, which corresponds to a technical presentation component (c.f. Presentation tier in **Fig. 8**) managed by the Interactional Objects, manages the display of the adequate data for both the HMD and the Tactile Display. Finally, note that the Interactional Objects, Business Objects and Translations are dispatched amongst the Process and Entity tiers.

The OpenGL client runs the main rendering loop of the 3D scene and dispatches user events to the Controller elements, while another thread loop taps into captured

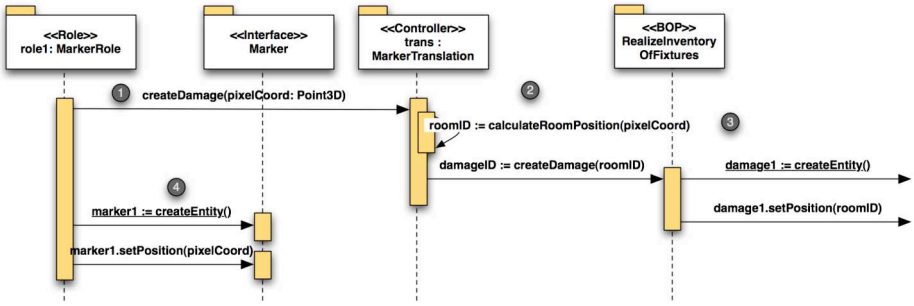

Fig. 11. Simplified view of the Interaction Object - Business Object connection at the Design level

Wi-Fi and acceleration data in order to infer the user's location, which is then notified to the main loop. The initial position of the user in the premises is obtained using manual calibration (i.e., the user pinpoints her position in the premises on the smartphone before starting the inventory of fixtures).

We focus now on the design choices made for realizing the connection between Interactional Objects and Business Objects, via the Translation objects.

In the technical branch, we noted that the use of the MVC architectural style corresponds to the requirements for our Augmented Inventory of Fixtures. Consequently, we need to adapt the Analysis models concerning the structure and behaviour of the Symphony Objects to this architectural style, while preserving the semantics of the functional services they provide.

As we saw previously, every collaboration between Symphony Objects (represented by "use" relationships in Section 3) occurs through a "Role" class (see the comments concerning **Fig. 6**).

In order to respect the MVC architectural style, we chose to overload the responsibilities of the "Role" classes with the management of the Interactional Object-Business Object (IO-BO) communications.

Concretely, when an event occurring in an Interactional Object needs to be reflected in the business space, the communications between the interaction space and the business space (identified by "represent" relationships in Section 3) occur through the "Role" classes. **Fig. 11** presents an example of such a communication, where the Interactional Object that uses the "Marker" object needs to manage the creation of a new "Marker" object, through the "MarkerRole" Role object:

1. The "MarkerRole" object calls the "MarkerTranslation" object to create a new damage report in the business space,
2. The "MarkerTranslation" object translates the pixel coordinates of the marker into the corresponding position in the premises and calls the "RealizeInventoryOfFixtures" Business Object Process (which holds the applicative logic for creating damages),
3. The "RealizeInventoryOfFixtures" Business Object Process calls the Business Object Entity for creating the damage in the right room of the premises,
4. The "MarkerRole" calls the "Marker" Interactional Object for creating a marker at the appropriate location in the 3D model of the premises (i.e., the "Situational Mesh" from Section 3).

Thus, all the occurrences of Interactional Object-Business Object (IO-BO) communications are concentrated into "Role" classes. Therefore, the coupling between the interaction and business spaces is limited to clearly identified entities.

5.2 Main points discussed

We saw in this section an overview of how the Symphony method realizes the blending of the functional analysis with the technical choices. After briefly showing how the functional (i.e., Symphony Objects) and technical components may be deployed, we studied how the organization of between Symphony Objects and Translations (see Section 3) may be adapted to the MVC common architectural style.

Let us mention that the design solution that we provided in this section is in no way exclusive. Indeed, we will show in future work how other design solutions may be efficiently implemented, independently of the technical choices, provided that developers respect the structure of the Symphony models from the Analysis phase.

6 Conclusion and future works

The Symphony method extended for Mixed Reality systems permits a Mixed Reality system design to be considered in its entirety without neglecting either the functional or the interaction part. However, if it is clear that interaction cannot be designed without considering functionality, we have pointed out in [18] that the interaction choices also influence the functional part. Therefore the method needs to include cooperative activities in order to guarantee global design consistency, but also to maximize the benefits obtained from the analysis of each conceptual space.

The method also allows a clear separation of concerns, as well as design traceability for each domain. We focused in this chapter on how these properties are achieved in the interaction space.

The method has been applied to two case studies. Each study was realized by teams of three persons. These teams consisted of average and expert programmers, some of whom are authors of this paper, others being interns in our research team.

The first application we developed concerned the simulation and evaluation of airport security. It had to respect precise requirements from another research team in our laboratory, and most particularly it had to fit into a large test workflow. Even though not strictly speaking a Mixed Reality system, the application nevertheless features a very complex interface (several windows, animated elements). It has met its requirements and is currently being transferred for actual deployment.

The other application (i.e., the Mixed Reality inventory of fixtures) is well-advanced but still under development. At the current stage of development, screenshots are quite similar to the prototype presented in **Fig.** 4. Additionally, we are currently working on a robust localization system.

Even if these studies are not comparable to industrial designs, they forced us to define more precisely the design process and modify some of its steps.

However, we also noted that some developers and testers who intervened during the development or the evaluation of our method had trouble handling the volume of the specifications. Indeed, the amount of documentation produced during the development cycle is currently quite high (over 50 pages of specifications for a project of about 4000 lines of codes, for instance). This volume is generally necessary for allowing the integration of applications into large information systems (such as that of a real-estate agency) and for permitting the maintenance and evolution of the system. Nevertheless, extensive documentation is only necessary when the development team is not familiar with either the business domain or the interaction style. In more well-known contexts, large parts of the method may be either skipped or less strictly documented, thus making the development methodology more lightweight.

Finally, a large number of models are constructed during the development cycle of our Mixed Reality systems. Even though most of these models are either refinements of previous models or partially generated, we admit that the members of the development team currently spend too much time using a plethora of modeling tools for single iterations of their models, and not enough time discussing and iterating over these models.

Consequently, one of our long-term perspectives is to propose an open platform for supporting collaborative modeling activities and short iterations over these models. This platform may be based on the model-driven engineering approach (where models would be first class elements). We hope that such support, coupled with our methodological guide, will facilitate the design of mixed reality systems.

Acknowledgments. We are grateful to the Foundation “Gran Mariscal de Ayacucho, the university UCLA-Venezuela” for their financial support.

References

[1] Wellner, P.: Interacting with paper on DigitalDesk. Communications of the ACM (36)7 (1993)

[2] Müller, D., Mixed Reality Learning and Working Environments - The MARVEL approach. In proceedings of the 12th European Conference for Educational and Information Technology (Learntec'04), Karlsruhe, Germany, (2004).

[3] Troccaz, J., Lavallée, S., Cinquin, P., Computer augmented surgery, Human Movement Science 15, (1996), 445-475.

[4] Ullmer, B., Ishii, H. and Glas, D.: mediaBlocks: Physical Containers, Transports, and Controls for Online Media. In proceedings of SIGGRAPH '98, Orlando, Florida USA, (1998), ACM Press, 379-386.

[5] Dubois, E., Gray P., Nigay, L.: ASUR++: a Design Notation for Mobile Mixed Systems. Interacting With Computers 15(4) (2003) 497–520.

[6] Coutrix, C., Nigay, L.: Mixed reality: A model of mixed interaction. In: Proceedings of the 8th International Conference on Advanced Visual Interfaces AVI'2006, Venezia, ACM Press (2006) 43-50.

[7] Kulas C., Sandor C., Klinker G.: Towards a Development Methodology for Augmented Reality Users Interfaces, In Dubois E, Gray P. and Nigay L. (eds) : proceedings of MIXER '04, Exploring the Design and Engineering of Mixed Reality Systems, Proceedings of the IUI-CADUI'04 Workshop on Exploring the Design and Engineering of Mixed Reality Systems, Madeira (2004).

[8] Nigay L., Salembier P., Marchand T., Renevier P., Pasqualetti L.: Mobile and Collaborative Augmented Reality: A Scenario Based Design Approach, In F. Paterno (ed.): proceedings of the 4th International Symposium on Mobile HCI 2002, LNCS 2411, Springer, Pisa, Italy (2002) 241-255.

[9] Gauffre, G., Dubois, E., Bastide, R.: Domain Specific Methods and Tools for the Design of Advanced Interactive Techniques, 3rd Workshop on Model Driven Development of Advanced User Interfaces at MoDELS'07, Nashville, TN - USA, CEUR-WS Proceedings (2007).

[10] Tarby J.C., Barthet M.F. : Analyse et modélisation des tâches dans la conception des systèmes d'information : la méthode Diane+. In : Analyse et conception de l'IHM, Hermès (2001) 117-144 (In French).

[11] Lim K. Y., Long J. : The MUSE method for usability engineering, Cambridge University Press (1994).

[12] Gulliksen, J., Göransson, B.: Usability design: Integrating user-centred systems design in the systems development process. In: Tutorial at CHI'2005, Portland, USA (2005).

[13] Sousa, K., Furtado, E.: From usability tasks to usable user interfaces. In Dix, A., Dittmar, A., eds.: TAMODIA '05: Proceedings of the 4th international workshop on Task models and diagrams, New York, NY, USA, ACM Press (2005) 103-110.

[14] Constantine, L., Biddle, R., Noble, J.: Usage-centered design and software engineering: Models for integration. In Harning, M.B., Vanderdonckt, J., eds.: Proceedings of the IFIP TC13 workshop on Closing the gaps: Software engineering and Human-Computer Interaction. (2003)

[15] Hassine, I., Rieu, D., Bounaas, F., Seghrouchni, O.: Symphony: a conceptual model based on business components. In: SMC'02, IEEE International Conference on Systems, Man, and Cybernetics. Volume 2. (2002)

[16] Jacobson I., Booch G., Rumbaugh J. : The Unified Software Development Process, Addison-Wesley, 1999.

[17] Godet-Bar G., Rieu D., Dupuy-Chessa S., Juras D., *Interactional Objects : HCI concerns in the analysis phase of the Symphony method*, Proc. of the 9th International Conference on Enterprise Information System (ICEIS'2007), Madeira (2007).

[18] Godet-Bar G., Dupuy-Chessa S., Rieu D., When interaction choices trigger business evolution, 20th International Conference on Advanced Information Systems Engineering (CAiSE'08), LNCS 5074, Springer, 16-20 June 2008, Montpellier, France, pp 144-147.

[19] Roques P., Vallée F.: UML2 en action, de l'analyse des besoins à la conception J2EE, collection Architecte Logiciel, Eyrolle, 2004 (In French).

[20] Rolland, C., Prakash, N., Benjamen, A.: A Multi-Model View of Process Modelling. J. Requirements Engineering. 4, pp 169-187, 1999

[21] Paterno, F.: ConcurTaskTrees: An Engineered Notation for Task Models. In: The Handbook of Task Analysis for Human-Computer Interaction, Lawrence Erlbaum Associates, pp. 483-503, 2003

[22] Buschmann F., Meunier R., Rohnert H., Sommerlad P., and Stal M., Pattern-Oriented Software Architecture - A System of Patterns. John Wiley and Sons, 1996.

[23] Maclean A., Young R.M., Bellotti V.M.E, Moran T.P., "Questions, Options and Criteria: Element of Design Space Analysis", Human-Computer Interaction, Vol. 6, No. 3, pp. 201-250, 1991.