A Service-Oriented Approach for interactive computing systems

Abstract. The introduction of new technologies leads to a more and more complex interactive systems design. In order to describe the future interactive system, the human computer interaction domain uses specific models, design processes and tools in order to represent, create, store and manipulate models. The aim of our work is to facilitate the work of model designers and project managers by helping them in choosing processes, modeling environments adapted to their specific needs. This paper details the use of a service-oriented approach for model management. Our propositions are related to three different abstract levels: the operational level to choose the appropriate tool, the organisational level to select a process and the intentional level to define modelling goals.

Keywords: HCI, model, service, model management, modeling tools, modeling services.

1 Introduction

In the Human Computer Interaction (HCI) domain, interactive systems are increasingly complex: they can use everyday life objects to propose tangible interfaces; they can couple the virtual and the physical worlds in augmented reality systems; they can adapt themselves to the user context, etc. Then they are increasingly difficult to design.

The HCI community made many proposals to address this design complexity. Some of them are based on ad-hoc code centric, while others use usability properties (as the ISO/IEC 9126[1]) or ergonomic requirements [1], [2] [27], in order to make user interfaces (UIs) more usable. These recommendations refers to ergonomic properties (as usefulness, users experience, etc.) that often cannot be formalizes as process and models, because they may contain a part of subjective appreciations.

On the other hands, substantive efforts have been devoted to the definition and use of models, and extensive development of software support has been achieved. We are interested in those propositions that are based on models.

The HCI community uses different models to support the design of interactive systems. In particular, the HCI design is often based on task analysis, which is classically represented by task trees. Moreover the use of these models can be guided

[1] The ISO/IEC 9126 series are part of the stands defined by the software engineering community for the standardization of “quality of use” on a software product.

by specific processes. Many interactive system design method are proposed. They are often based on task analysis [22], [28], [30]. Many efforts are also related to contextual design [4], [11], scenario-based approaches [8], [25] and iterative prototyping [14].

The choice of such processes is a strategic decision that depends on the goals expected by the model designers. For example, in the HCI domain, the Human is the most important aspect in all phases of the development process. For consequent, a modeling goal is the "study the user interaction". During this study, the designer can discover that instead of a classical WIMP interface, the design of a mixed reality system is more appropriate. So he will have a new goal "design a mixed reality system" for which he needs a specific process. Therefore, a rigid method is no longer desired and there is a need to support method definition and adaptation.

To face these needs of adaptation and flexibility in the design, we propose to help model designers and project managers in choosing processes according to their modeling goals. For example, based on the goal "study the user interaction", a model designer will be able to choose the methods of fragments of processes, such as the Organizational and Interactional Specifications of Requirements of the Symphony method [9] or the Elaboration phase of the extended RUP proposed by [16].

The choice of a process determines the models to use, and then their modeling environments. For instance, selecting the process proposed to design an Interactive system give rise to use of the ConcurTaskTress (CTT) notation and them to the choice of CTTE [19] for its support.

Our approach is based on the reuse of existing processes and technological solutions in order to find solution to the goals of designers and managers. It concerns the adaptation of Service Oriented Architecture (SOA) [18] to model management. It is based on three abstract levels: the operational level, the organizational level and the intentional level: 1) operational services carry out automated operations (e.g. editing a model); 2) organizational services propose fragments of design methods i.e. processes; 3) the intentional services correspond to the goals proposed by any person or organization handling models. This paper focuses on the intentional and the organization levels which are the main contributions of our work.

This paper is organized as follows. Section 2 describes two experimental scenarios. These examples are based on interactive system design methods. Section 3 presents our approach based on service-oriented models management. Sections 4 and 5 detail the models of the intentional and organizational layers. Section 6 presents a platform support for our service-oriented approach. Finally, conclusions and perspectives are presented.

2 Interactive Design Method

The finality of our study is not situated in the choice of ideal interactive system design methods. We aim in the choice of appropriate processes and modeling tools. To explain our approach, we focus on two interactive system design methods. Our goal is

to demonstrate how our service-oriented approach can support the construction of modeling environment used for interactive system design.

The first method is an extension of the Symphony design method [13], used as a medium for merging HCI and software engineering development processes. Symphony is a user-oriented, business component-based development method originally proposed by the UMANIS Company. It has been extended lately to integrate the design of Augmented Reality systems [9].

In this article, we concentrate on the phase of the "Organizational and Interactional Specifications of Requirements" (Figure 1). This phase aims to analyze a business process at organizational level (the "who does what and when" of the future system), and to envisage a satisfactory interaction to realize activities of different stakeholders.

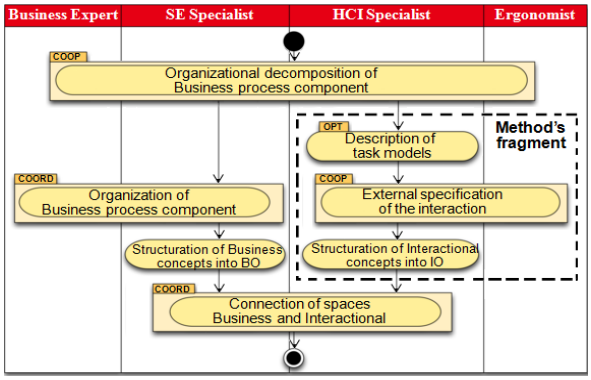

Fig. 1. A phase of the Symphony method [9]

Concerning the HCI aspects (box of Figure 1), the activities proposed by the design method are: **description of task models** to clarify the interactions of the internal actors with the system, **external specification of the interaction** to define the user interface (UI) and its supporting devices, and S**tructuration of Interaction concepts into a specific model** composed of reusable components **called Interactional Objects (IO)**. These actions must be driven by the ergonomics and the HCI specialist.

The second approach [29] used for model-based design of user interfaces. It is founded on the Cameleon Reference Framework [7]. The approach allows designers defining a model-based user interface development (UID) method according to the problem domain or context of the project by analyzing goals and activities. It is composed of four-step reification process: 1) create conceptual models (e.g. task model, data model, user model) which bring the descriptions produced by the designers for that particular interactive system and that particular context of use; 2) create Abstract UI (AUI) which specify objects in a UI independent of device, create Concrete UI (CUI) which makes explicit the final look and feel of the Final User Interface considering device constraints, and create Final UI (FUI) which is expressed in source code.

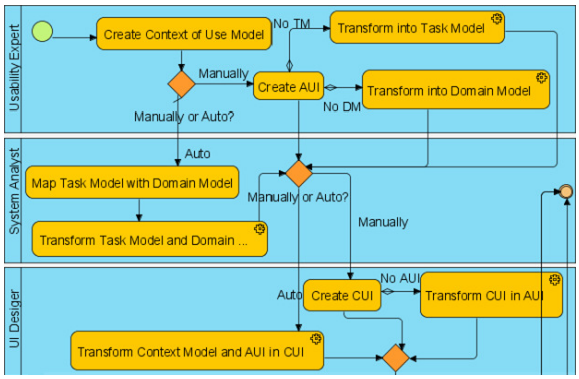

Fig. 2. An interactive system modeling process [29]

Figure 2 illustrates a global vision of this process. Concerning the HCI aspects of the Usability Expert, the activities proposed by the design method are: **Create Context of Use Model** to design user-centered UIs, **Create AUI** to specify objects in a UI independent from devices, **Transform into Task Model** to automate the generation of specification of UIs (receive AUI as input and generate task model), and finally **Transform into Domain Model** to automate the generation of UIs focused on the application domain for describing the manipulated data.

In the remainder of this paper, these two examples are used to illustrate our approach for model management.

3 General Approach

3.1 Introduction

This section proposes the concepts of our service-oriented approach for models management with services. In our approach, modeling services enables to structure the set of knowledge which is necessary to the description of goals, in order to facilitate and to automate software development steps using models (e.g. model edition, model transformation, etc).

3.2 The basic service-oriented architecture

Service-Oriented Computing (SOC) is a paradigm that uses services as fundamental elements for developing applications [17]. This development approach speeds the application development and facilitates the composition of distributed applications. A service is a set of self-contained software modules and auto-descriptive applications that can be described, published, discovered, composed, and negotiated on demand by a customer. **Services** perform functions, which can be

anything from simple requests to complicated business processes [18]. This approach defines an interaction between software agents as an exchange of messages between service requesters (customers) and service providers (Figure 3). **Customers** are software agents that request the execution of a service. Customers must be able to find the description(s) of the services they require and must be able to bind them. **Providers** are software agents that provide the service. Agents can be simultaneously service customers and providers. Providers are responsible for publishing a description of the service(s) they provide.

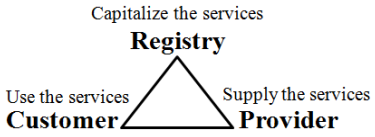

Fig. 3. Service approach

3.3 Three levels of services

Our approach based on services relies on three modeling levels (see Figure 4) where providers, clients and services are different.

The first level corresponds to the **operational layer**. This layer offers services for model designers, to facilitate the building of their modeling environment. The customers are designers who want to manage models in an individual or collaborative way (with other designers). So, they should define and adjust their modeling environment to theirs needed functions in terms of models management. For example: an "HCI designer" can need a modeling environment that offers support for editing "task models" and transforming these models into a "concrete user interface" (CUI) for a specific device.

The organizational layer decomposes information system development methods as fragments. Several types of fragments have emerged in the literature. The most known of these different kind of representation are method fragments [6], chunks [24], components [32], and method services [12]. Historically, the term fragment was the first one to appear, long before component, chunk, and so on. In this article, we will use the term: "method fragment".

The main role of a method fragment is to provide guidelines to the system engineer for realizing some specific system development activity (for example: specify context of use, user requirement specifications, produce design solutions, evaluate design, etc.) as well as to provide definitions of concepts to be used in this activity [20].

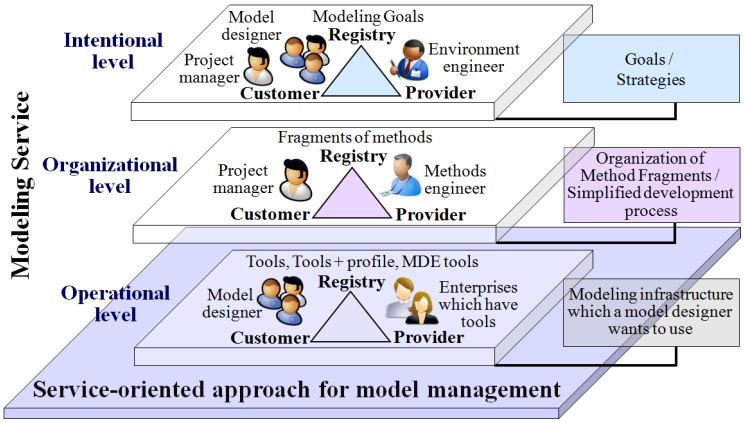

Fig. 4. Three levels of service

In our work, the organizational layer enables the modeling of reusable method fragments. In this layer, the activities are expressed in terms actions on models. The objective is to capitalize method fragments in order to provide them to designers, who have a role in the project group. Customers are, in this case, projects managers which need to define and manage roles and activities in their development process.

The organizational layer uses the operational services in a coordinated way. Project managers can choose some organizational services (part of design process) that require the implementation of operational services for model management. Thus, they create the model management environments for designers involved in their development process.

The **intentional layer** (Figure 4) deals with modeling goals. It conceptualizes strategic modeling needs required by a specialist, a group of specialists, a unity of work or any organization involved in the development process. So, this layer uses the organizational services. The provider corresponds to the environment engineer who plays a new role in charge of the administration and management of the service platform. The customers are still those of the organizational and operational layers, e.g. the models designers and the project managers. For these customers, the services are the goals proposed by the environment engineer (e.g. “Specify an Interactive System”).

In this section we introduced the principles of our service-oriented model management approach. In the following sections, we detail the intentional and the organizational levels on the interactive system design methods presented in section 2.

4 Modeling an intentional service

This section presents the model of the interactional layer. An **intentional service** is a business-oriented service described from an intentional point of view (e.g. specify an interactive system, study the usability…). It corresponds to the modeling goals. It can

be composed by other intentional services. A complex intentional service consists of elementary services and they are realized by organizational services that correspond to methods fragments (Figure 5).

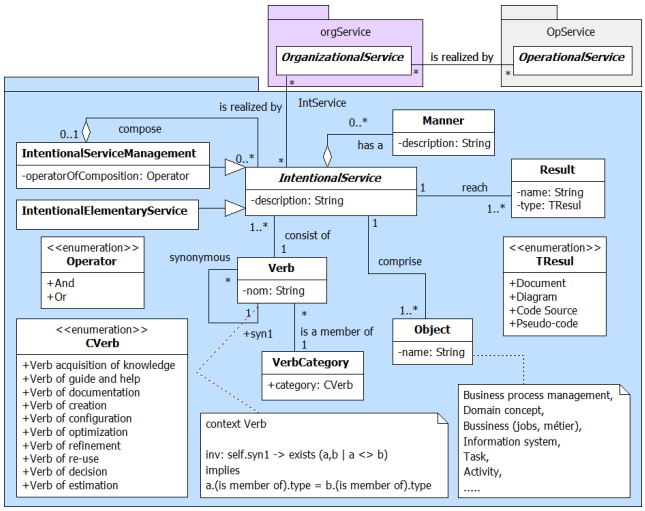

Fig. 5. Intentional model service

The service is characterized by **a verb** that is associated with objects and complemented by a refinement. We have used the ontologies of goals proposes by [11] which describe development's problems. From these ontologies we have identified a set of verbs that describe specific intentions for models management (e.g. **study** interactive system usability, **design** UIs considering users' mental models to perform their task, **automate** the generation of UIs considering many devices …).

A verb belongs to a **category of verbs**. It corresponds to the type of activities which models designers implement during the development (e.g. the acquisition of knowledge, the guidance and help, documentation …). In addition, a verb has several **synonymous verbs**. So, synonymous verbs belong to the same category of verbs.

The object is a modeling concept that is defined or reused by the verb (e.g. Interactional Object, Abstract UI, Task, Interactive System …).

The result is an elaborated artifact, modified or used by a service. We take the classification of a result proposed in the unified process [15] and [12]. The types of results identified are: diagrams, documents, source code, and pseudo-codes. **The manner** is a feature that plays specific roles concerning the verb. It corresponds to the way as the service must be realized (e.g. in the purpose to "specify the software architecture with interaction devices choice", the phrase "with interaction devices choice" corresponds to the way to solve the goal achieved.

We use a linguistic approach to formulate an intention. Our purpose is to express the intentional services defined by the meta-model presented in Figure 5. This approach relies on the structural declaration of an intention proposed by Rolland [23].

We have adapted this general statement to the needs of model management. So, in our context work, the structure of an intention is:

Intention: Verb<Object>,<Result(name,type)>[Manner]

The element "Manner" that are in hooks "[]" correspond to optional element. The general structure of the intentions corresponds to several cases. We present below some combinations followed by an example.

Intention 1: Verb<Object><Result(name,type>

For the intention: **"Specify an Interactive System",** the general structure is: Verb("**Specify**")<Object("**an Interactive System**")><Result(Interactive System, code source>.

Intention 2: Verb<Object><Result(name,type>[Manner]

For the intention: **"Study the user interaction by task modeling"**, the general structure is: Verb("**Study**")<Object("**the user interaction**")>Result(task model, diagram)[Manner("**by task modeling**")].

4.1 Examples

At the intentional level, we must determine the goals of the two methods presented in section 2. These strategic goals are those required HCI specialists, who participate in the development process of interactive systems. Studying the two examples, we define the main goal "specify an interactive system", which can be decomposed into other goals. So, based on the intentional model, we create the appropriate intentional services to develop interactive systems (figure 6).

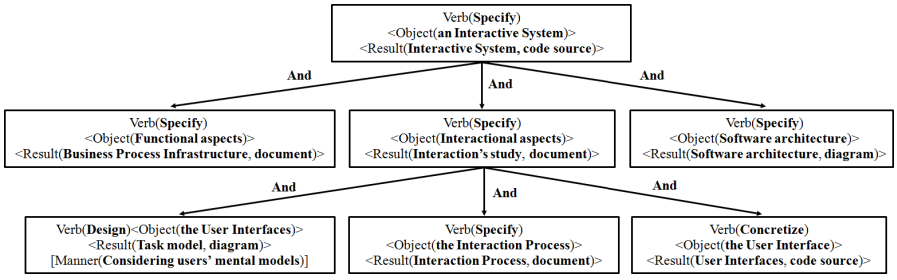

Fig. 6. A composition of intentional services

In the figure above, rectangles correspond to the services and the links describe the composition of intentional services. For example: the service "Specify an Interactive System" is decomposed into three other services: "Specify functional aspects", "Specify interactional aspects" and "Specify software architecture". Similarly, these services can be decomposed. For the sake of conciseness, we concentrate only on the sub-decomposition of the service "Specify Interactional Aspects". It is decomposed intro three other services: "Design the User Interfaces considering users 'mental

models", "Specify the Interaction Process" and "Concretize the User Interface". As we will see in the next section, these services are linked to organizational services in order to propose a solution (in terms of processes) to the specified goals.

5 Modeling an organizational service

This section presents the model of the organizational layer. Our organizational model service (Figure 7) is inspired by the work of Ralyté et al. [21] who propose a method engineering process model approach, which permits to represent any method as an assembly of the reusable method fragments. In our work, we use the notion of service to support the construction of modeling processes by assembling method fragments.

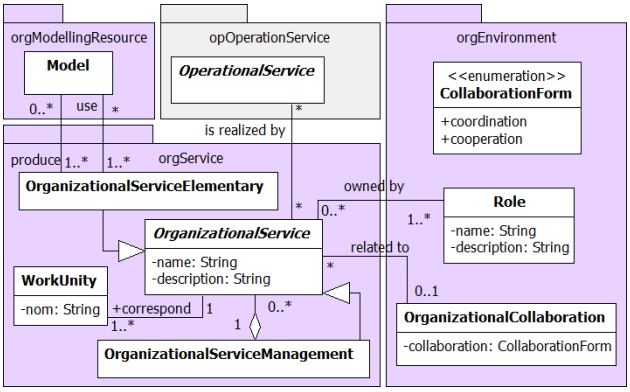

Fig. 7. Organizational model service

An organizational service consists in a composition of development method fragments that can be reused by model designers, in response to the intentional needs. So, an organizational service complex is composed of organizational services. A method fragment is represented by an organizational elementary service that is defined in terms of model manipulation. For example, in the definition of a transformation process to generate an Abstract UI, an aspect is the edition of the source meta-model. This activity consists in the production of a model that can be used by other method fragments.

An organizational service is carried out by one or more roles. A model designer who plays a role can define and reuse several organizational services. At this level, the collaboration term is used for coordination and cooperation tasks between designers [5]. The coordination activities consist of the decomposition of work in activities with similarly goals. The cooperation activities are based on a common modeling goal. Each designer provides their models and the cooperation permits the production of consensual or common models.

The **work unity** defines the action executed during the interactive system design process. It determines in which case the use of organizational service is appropriate (i.e. requirements, analyze, design, validation, implementation,…).

Another aspect considered by our organizational model is the fact that the organizational services are realized by operational services. It means that organizational elementary services must use operational service to support the management of modeling activities.

5.1 Examples

As we have commented in the previous section, we consider that an intentional service can be realized by several organizational services. Concerning the sequence of modeling activities of the intentional service "**Specify Interactional aspects**" of an Interactive System, one of the possible processes that answers to this goal is a part of the phase "Organizational and Interactional Specifications of Requirements" as defined by the Symphony method described in section 2.

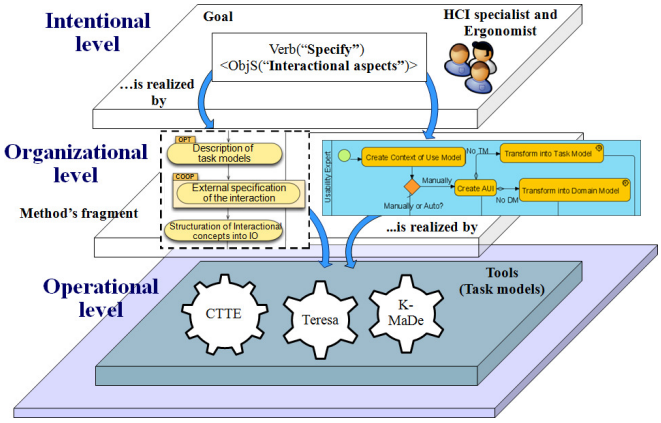

Fig. 8. Relation between an intentional service and an organizational service

This method fragment (box of Fig. 1) must be defined in terms of organizational services. It corresponds to an organizational service composed of three sub-services (one by activity). The composite organizational service is linked to the intentional level by the goal: "Specify Interactional aspects" (Figure 8).

The elementary services are described by actions on used or produced models. For example, the activity "**description of task models**" is expressed at the organizational level as an elementary service that carries out the action: "edition of task models". Moreover an organizational service can be supported for several tools defined in terms of operational services. The action "edition of task models" can be carried out by tools with support to task models (CTTE [19], Teresa [3] KMAde [26]...).

Another possible process that answers to the goal "**Specify Interactional aspects**" of an Interactive System is a part of the method engineering for model-driven user interface development proposed by [29]. This goal deals with the UI activities of the role "Usability Expert". They correspond to an organizational service composed of three elementary services: "**Create task model**" to describe task in a hierarchical

manner, "**Create context of use model**" to describe the users' characteristics, the platform used and environment; and "**Create domain model**" to describe the manipulated data. As previously, the action "create task model" needs operational services supporting task modeling. These operational services are those (i.e. services for CTTE, Teresa or KMade…) which are already linked to the action "edition of task models".

In this example, we have shown how an intentional service can be realized by two different organizational services. We also illustrate how an organizational service is composed of other organizational services, which can be linked to operational services. To complete this example, we need a support to facilitate the selection of services. In the following sections, we present an overview of the platform that we are currently developing for our service-oriented approach.

6 Platform support for service-oriented models management

6.1 Platform overview

This section presents an experimental prototype which supports our approach. It allows the registration, consultation, research and design of our three services levels.

The prototype has been implemented with two independent but complementary blocks. The first block (Figure 9b) considers the implementation of a service Repository with the integrated development environment for service composition management "ChiSpace" [33]. The aim of this environment is to simplify the work of developers when developing service-based application within domain specific context. ChiSpace was implemented based on the Eclipse Modeling Framework (EMF) platform and the JET 2 technology. The environment is composed of a set of editors within a customized perspective of eclipse [33].

The service Repository corresponds to the database that stored the descriptions of our three services levels. These descriptions are based on the modeling services presented in the previous sections.

The second block is a tool to add, view, select and validate services which are stocked in the service Repository. This realization is based on the Eclipse Rich Client Platform (RCP). So, Eclipse RCP is used to develop the UIs, which will allow use the features of the platform for customers and providers. Figure 9a shows the search intentional services interface. The other UIs are not presented in this article because space constraints.

Our prototype currently contains these two blocks. It permits actually, add and search intentional services. We have conceived the UIs for the others services (organizational and operational), but the functionalities of theses UIs are not completed.

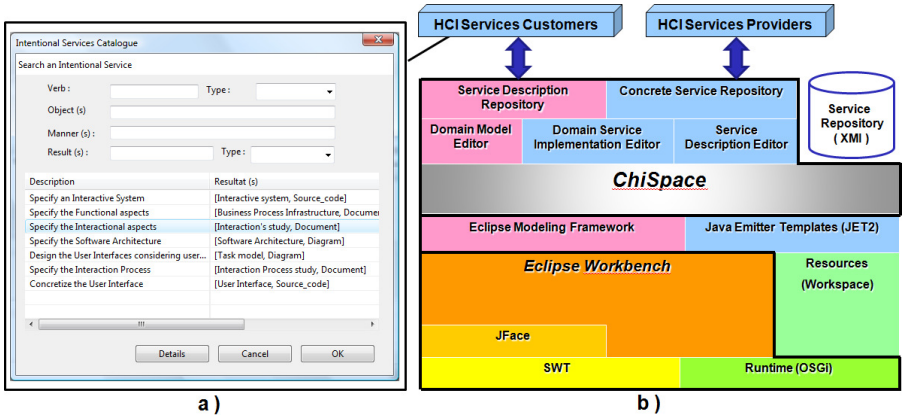

Fig. 9. The service-oriented platform

6.2 Global vision of use of the service-oriented platform

Nowadays, we do not have a formal process of helping customers to use the platform. Our global vision is "**top-down** and **bottom-up**". Thus, the platform that we propose can be used in different ways, depending on the needs of customers. Table 1 summarizes some of the scenarios that we have considered.

Table 1. Global vision of use of the service-oriented platform

The vision of Customer. For the customer the goal is search the services depending on their needs.		
	scenario of use	**Vision**
An intentional support	**From the intention towards the tools** The customer has the intention to study the user interaction and he wants to use a tool which has support to task models within uses a modeling process. So, the information concerning to the object, the manner and the result of an intentional service facilitate the request to find the tools required by the user. **From the intention towards the method fragments, then the tools** The customers must seek and choose an intentional service according to the objectives to reach. The goal selected corresponds at organizational level to organizational services. Then, customers choose among the organizational services associated with the intentional service. Finally, customers can use operational services offering a modeling environment support of the chosen process. This is the process that we illustrate in the previous sections.	These examples correspond to the **top-down** vision in which the modeling environment choice is a strategic decision supported by the selection of modeling goals.
The service dynamic	**From tool towards the method fragments** The customer knows perfectly one of the tools that he wants to use. However, he needs to rely on processes that suggest using this tool. These processes suggest the use of other tools to complement his work. Par example: a customer knows perfectly KMAde [26] and needs to specify the user interaction for mixed system, then, he explores the organizational services that use this tool. He find that the method proposed by [10] suggest that a complementary tool for the goal of specify the interaction is the tool GuideMe [31].	The organizational model service research is guaranteed by a vision **bottom-up**.

7 Conclusion

We have presented in this paper a set of principles of our service-oriented models management approach, designed to helping to model designers in choosing processes and modeling environments adapted to their specific needs. Our work relies on three modeling levels (where providers, customers and services are different): the operational layer to define the modeling environment for model designers; the organizational layer to enable the reuse of operational services in a coordinated way, but also the creation and the management of method fragments; the intentional layer permits to define the modeling goals that can be implemented by design processes described at organizational level.

We also present an overview of the platform under development that will allow the management of our three services levels. Our prototype currently permits add and search intentional services.

Future works include finalizing the service-oriented platform. With the aid of this platform the next step is to test our service-oriented approach in different projects. It will enable us to validate our propositions and to analyze the impact and the usability of our approach. The realization of user experiments will also enable us to explore other functionalities, and integrated them in our solution. Finally, we plan to propose a formal process of use and operation of the platform to facilitate the modeling activities of model designer.

References

1. Abowd G., Coutaz J., and Nigay L.: Structuring the space of interactive system properties. In: Engineering for HCI, vol. A-18, pp. 113--129. North-Holland (1992)
2. Bastien J., Scapin D-L.: A validation of ergonomic criteria for the evaluation of human-computer interfaces. In: International Journal of HCI, vol. 4, pp 183-196. (1992)
3. Berti S., Correani F., Mori G., Paternó F., Santoro, C.: TERESA: A Transformation-Based Environment for Designing Multi-Device Interactive Applications, in Proc. of CHI'04, pp. 793--794. ACM Press, New York (2004)
4. Beyer H.; Holtzblatt K.: Contextual Design. Defining Customer-Systems. Morgen Kaufmann, San Francisco (1998)
5. Blanco E., Grebici K., Rieu D.: A unified framework to manage information maturity in design process. In: International Journal of Product Development. Volume 4, Number 3-4, pp. 255--279, (2007)
6. Brinkkemper S.: Method Engineering: engineering of information systems development method and tools, Information and Software Technology, 38(7), (1996)
7. Calvary G., Coutaz J., Thevenin D., Limbourg Q., Bouillon L., Vanderdonckt J.: A Unifying Reference Framework for Multi-Target User Interfaces. Interacting with Computers 15(3), pp. 289--308 (2003)
8. Constantine L., Biddle R., Noble J.: Usage-centered design and software engineering: Models for integration. In Proc. of the IFIP TC13 workshop on Closing the gaps: Software engineering and Human-Computer Interaction (2003)
9. Dupuy-Chessa S., Godet-Bar G., Pérez-Medina J-L., Rieu D., Juras D.: Technical and Functional Issues for Integrating Mixed Systems into Information Systems. Engineering of Mixed Reality Systems. Editors: Dubois E., Gray P., Nigay L. Springer, (2009) (To appear)

10. Gauffre G., Dubois E., Bastide R.: Domain-Specific Methods and Tools for the Design of Advanced Interactive Techniques. In: Models in Software Engineering. Holger Giese (Eds.), Springer, pp. 65-76, Vol. 5002, Lecture Notes in Computer Science, (2008)
11. Gulliksen J., Goransson B.: Usability design: Integrating user-centred systems design in the systems development process. In: Tutorial at CHI'05, Portland USA (2005)
12. Guzélian G., Cauvet C.: SO2M: Towards a Service-Oriented Approach for Method Engineering. In: IKE'07, Las Vegas USA, (2007)
13. 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)
14. Hix D., Hartson H.: Developing User Interfaces, Ensure Usability Through Product & Process. pp. 381. John Wiley & Sons, Inc. (1993)
15. Jacobson, I., Booch, G., Rumbaugh, J.: The Unified Software Development Process, Addison-Wesley, (1999)
16. Kruchten, P.: Introduction of Rational Unified Process, number ISBN 2 212 09104 4. Editions Eyrolles, 282 pages, Paris France (2000)
17. Papazoglou, M.P.: Service-Oriented Computing: Concepts, Characteristics and Directions. In: 4th International Conference on Web Information Systems Engineering (WISE'03), pp. 10--12. IEEE CS, (2003)
18. Papazoglou M-P., Traverso P., Dustdar S., Leymann F., Service-Oriented Computing: A Research Roadmap, Dahstuhl Seminar 05462, 1-29, (2005)
19. Paternò F.: ConcurTaskTrees: An Engineered Notation for Task Models. In: The Handbook of Task Analysis for HCI, Lawrence Erlbaum Associates, pp. 483-503, (2003)
20. Ralyté J., Backlund P., Kühn H., Jeusfeld M.: Method Chunks for Interoperability. In: 25th Int. Conf. on Conceptual Modelling (ER'06). LNCS 4215. pp. 339-353, Springer-verlag, Tucson, Arizona, USA (2006)
21. Ralyté J., Rolland C.: An Approach for Method Reengineering. In: ER'01. Hunii H., Jajodia S., Solvberg A. (Eds.), LNCS 2224, Springer-Verlag, pp. 471-484. Yokohama Japan (2001)
22. Redish J., Wixon D.: Task analysis: The human-computer interaction, fundamentals, envolving technologies and emerging applications. pp. 922-940. New York (2003)
23. Rolland C.: Capturing System Intentionality with Maps, Conceptual Modeling in Information Systems Engineering. pp. 141--158. Springer, Berlin, Germany, (2007)
24. Rolland C., Plihon V., Ralyté J.: Specifying the reuse context of scenario method chunks. In: 10th Conf. on Advanced Information Systezms Engineering. Pisa Italy (1998)
25. Rosson M-B., Carroll J-M.: Usability Engineering: Scenario-based Development of Human-Computer Interaction. Academic Press. London (2002)
26. Scapin D-L., Lai-Chong Law E.: Report and Refine Usability Evaluation Methods (R3UEMs), COST294-MAUSE 3rd International Workshop, Athens (2007)
27. Shneiderman B.: Designing the User Interface. 638 pages. Addison-Wesley (1998)
28. Sousa K., Furtado E.: From usability tasks to usable user interfaces. In: TAMODIA'05, pp. 103--110. ACM Press, New York (2005)
29. Sousa K., Mendonça H.,Vanderdonckt J.: Towards Method Engineering of Model-Driven User Interface Development. In: TAMODIA'07, pp. 112—125. Toulouse France (2007)
30. Tarby J-C., Barthet M-F.: The DIANE+ Method. In: CADUI'96, pp. 95--120. Namur (1996)
31. Viala J., Dubois E., Gray P.: GUIDE-ME: graphical user interface for the design of mixed interactive environment based on the ASUR notation. In: 1st French-speaking conf. on Mobility and ubiquity computing UbiMob vol. 64, pp. 74--77. Nice France (2004)
32. Wistrand, K., Karlsson, F.: Method components: Rationale revealed. In: CAISE'04, Springer-Verlag. Riga, Latvia (2004)
33. Yu J., Lalanda P., Chollet S.: Development Tool for Service-Oriented Applications in Smart Homes. In: Proc. of SCC'08, pp. 239--246. DC USA, (2008)