Modeling User Interfaces to Workflow Information Systems

Josefina Guerrero[1], Jean Vanderdonckt[1], Juan M. Gonzalez[1], Marco Winckler[1,2]
[1] *Université catholique de Louvain, Louvain School of Management*
Place des Doyens, 1 – B-1348 Louvain-la-Neuve (Belgium)
[2] *LIIHS-IRIT, Université Paul Sabatier, 118 route de Narbonne, Toulouse F-31062 (France)*
{guerrero, jean.vanderdonckt@uclouvain.be, gonzalez, winckler}@isys.ucl.ac.be

Abstract

This paper addresses the need for designing user interfaces (UIs) to workflow information systems by adopting a model-centric approach. We introduce a conceptual workflow model to determine system functionalities of workflow. The model is exploited in order to systematically derive UIs to access these functionalities. The workflow model is recursively decomposed into processes which are in turn decomposed into tasks. Each task gives rise to a task model whose structure, ordering, and connection with the domain model allows the automated generation of their corresponding UIs, using a transformational approach. Each UI is specified in a User Interface Description Language, workflow, process, and task models are specified in the same specification language. The language is exploited to generate multiple source codes, depending on how their corresponding tasks are structured in the process model; and by a workflow execution engine to produce running workflow systems. A real-world case is outlined to exemplify the process.

1. Introduction

The introduction of Workflow Management Systems (WfMS) in organizations has emerged as a major advantage to plan, control, and manage organization's business processes. Implemented properly, workflow applications enable companies to reengineer and streamline business processes; for this reason, the interest in workflow systems has grown dramatically over the last years. Recently, the Web had become a privileged platform for implementing workflow systems. The Web provides ubiquitous access to information; supports inherent distribution of business process, and consist of platform-independent user interfaces (UIs). However, workflow-based Web applications are far more complex than traditional Web applications; currently available methods do not provide comprehensive support to integrate business process analysis and advanced techniques for Web application design, which open this topic for improvements [2]. This paper presents a conceptual framework for the development of workflow-based Web application from the UI point of view. This framework considers business process, user tasks and UIs as independent layers that can be conjointly used to build the application. Section 2 presents the underlying conceptual model of the framework. Section 3 provides a description of the method to generate the UI of a workflow model. Section 4 provides a real life case study illustrating the method, a workflow-based Web applications for reviewing documents. Section 5 presents the related work. Finally, section 6 presents the conclusion of this work.

2. Conceptual Modeling of Workflow

The underlying conceptual model [7] is composed of workflow, process, task and organizational models, see Figure 1. The *workflow model* is recursively decomposed into processes which are in turn decomposed into tasks. The definition of a *process model* indicates the ordering of processes in time, space, and resources. Each process gives rise to a process model structured and ordered with *process operators*. Process operators determine whether the flow of work is *sequential, parallel split, exclusive choice* or *multi choice;* with the corresponding merger operators, *synchronization* and *simple merge.* A *task model* represents a decomposition of tasks into sub-tasks linked with task relationships. The model was adapted from the ConcurTaskTrees (CTT) [11] in order to take into account a couple of missing task relationships: inclusive choice and disabling with information passing. The *organizational model* is composed of organizational units, jobs, and resources with their correspondent agenda.

2.1 Conceptual Mapping Model

Transformations are applied in cascade though the workflow layers using a mapping model. In order to support the mapping between the layers, predefined relationships provided by UsiXML (USer Interface eXtensible Markup Language – http://www.usixml.org) [10] were used. UsiXML is a User Interface Description Language (UIDL) that is independent from code.

0-7695-3093-1/08 $25.00 © 2008 IEEE
DOI 10.1109/ICAS.2008.27

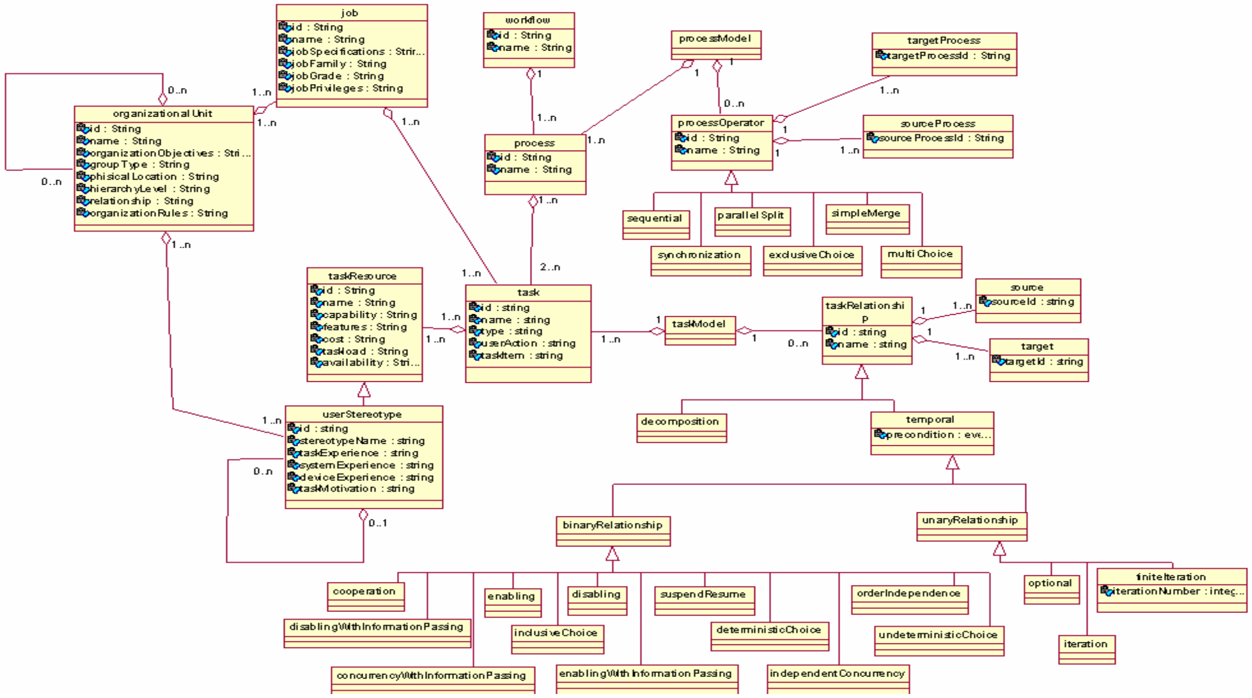
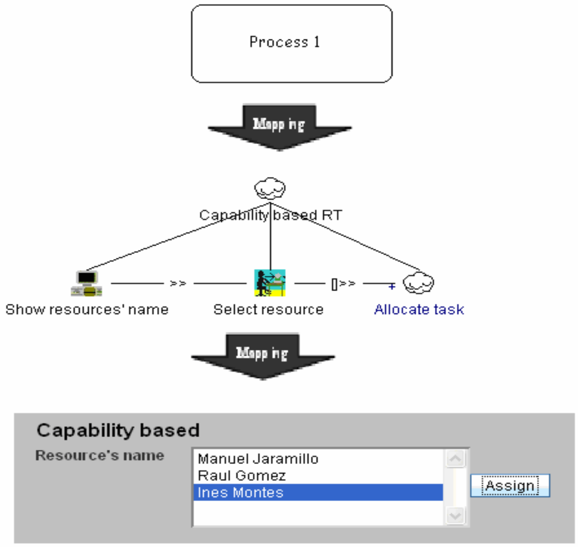


Figure 1 Overview of Conceptual Model

The workflow model consists of three layers: process, task, and UI (see Figure 2). The process layer defines the business process; each process can be considered as an independent building block connected to the workflow. User tasks are covered by task models. This layer focuses on user activities rather than processes. By exploiting task models description, different solution scenarios can be modeled. Each scenario represents a particular sequence of actions to be performed. Task models do not impose any particular implementation so that user tasks can be better analyzed without implementation constraints; it is, even possible to analyze user activities. The third layer, the UI derived from scenarios extracted from task models using a transformational approach.

As in a model-based approach all the components are models [14], transformations between models and relationships are described in terms of a meta-model. Several relationships have been defined to explicit the relationships between the domain model and the UI model. The current UsiXML mappings were extended by adding the mappings: *Is Grafted On* (when a task is grafted on another one, even at run-time), *Is Defined By* (when a task is defined by a userStereotype), *and Is Allocated To* (when a task is allocated to some resource, that is assigned to a *taskResource*) [7]. Each mapping model is also stored as mapping instances in UsiXML.



Manuel Jaramillo
Raul Gomez
Ines Montes

Figure 2 Overview of Conceptual Mapping Model

2.2 Conception Criteria

Fulfilling a form and sending it for evaluation are activities that can easily be consider either two independent processes or one process composed of two tasks, depending on the designer. When combining workflow and task models the boundaries of each model must be specified in order to avoid design mistakes. Designers should consider the following guidelines composed of a set of parameters to identify the different layers of the model. In addition, we introduce the life cycle of each component, important aspect to provide a feedback, for instance, the process is not *started* due to a lack of workers.

A workflow model is associated to the operational and/or administrative objectives of organizations. It is defined inside the same organization; the work is carrying out and is associated to the automation of a business process. A process refers to the use of the same group of resources in a continuous period of time, with a specific ordering of tasks. The work is developed within groups, among groups or by a group as a whole. In addition, a process can be: primary (production), secondary (support), or tertiary (managerial). Finally, a task is performed in same place, by the same type of resource, in the same period of time.

The *Starting Criteria* of each model represents the event that triggers the execution of each component. The workflow requires an initial analysis which identifies the various processes, rules and associated control data, defining an initial phase, specific ordering of tasks. The process starts when an input (time, human or message) triggers the execution of the process. Once the task is defined, it could be initialized. To the *Stopping Criteria*, i.e., when the component is assumed like finished; a final evaluation is analyzed from a perspective based on the executions protocols/rules. The workflow arrives to a terminate phase or abort phase. When the process is completed, aborted or terminated an output can be sent indicating that the process terminates its execution. Finally, the task is finished in a horizontal form when the status is finish, cancel or fail; and in a vertical form the action is considered as the atomic leaf of the task model and cannot be further decomposed.

3. A Method to Generate the User Interface of a Workflow Model

A framework not just to generate UIs automatically but also to specify workflows and task models, integrating the concepts that we propose in previous section, is composed on the following steps: 1) define the organizational units, 2) define the workflow, which includes process model, 3) define the task models, 4) mapping model from task models to UIs.

The method proposed expands the current description of the User Interface Description Language (UIDL) UsiXML [10] and introduces a higher level of description, the workflow model. UsiXML is based on the Cameleon Reference Framework [3], which defines UI development steps for multi-context interactive applications.

The current steps are: *Tasks & Concepts* level describes the interactive system specifications in terms of the user tasks to be carried out and the domain objects of these tasks. An *Abstract User Interface* (AUI) abstracts a *Concrete User Interface* (CUI) into a definition that is independent of any interaction modality (such as graphical, vocal or tactile). A CUI abstracts a *Final User Interface* (FUI) into a description independent of any programming or mark-up language in terms of Concrete Interaction Objects, layout, navigation, and behavior. A FUI refers to an actual UI rendered either by interpretation (e.g., HTML) or by code compilation (e.g., Java).

In order to pass from one step to another, some transformational rules [9] are applied. The method proposed here (see Figure 3) starts from the specification of a workflow model, based in workflow patterns [15]. The result of such specification has two related results, on the one hand the UI required to handle the workflow, i.e. the agenda, tasks operations, such as delegation, jobs assignation. On the other hand the UIs of each workflow task can be described using the extended version of the task model and then it can be transformed into its correspondent UI, using the UsiXML current approach. The main purpose of this work is to develop a method for designing the UIs for a workflow information system; for this reason is important to consider usability guidelines [12] at design time for the building or the evaluation of UIs in order to respect cognitive and sensory-motor capabilities of users. However, this is out of the scope of this paper.

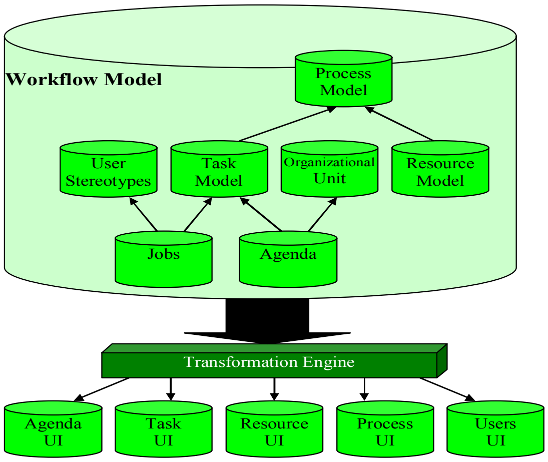


Figure 3 Method building blocks.

4. Case Study: Workflow for Equipment Management

In order to exemplify the above concepts (in particular the decomposition of a workflow into processes which are, in turn, decomposed into tasks that are attached to their UIs), a real-world case study is summarized in this section. This case study has been fully implemented with the model editors, all of them have being developed in Java 1.5 with Swing widget set. This set has been selected because it offers more capabilities to automatically generate UI code (here, Java code) from internal representations (here a UsiXML file).

Case study. A funding agency provides organizations with funds for buying laboratory equipment (e.g., electronic microscopes), hardware (e.g., graphic tablets) and software (e.g., dedicated scientific applications) or a combination of them (e.g., a special microscope with dedicated software). The workflow goals are threefold: (i) allow any organization manager to register to the system and apply for equipment funding; (ii) insert any equipment description and usage modalities once an equipment has been funded and acquired; and (iii) share the usage of acquired material by renting them.

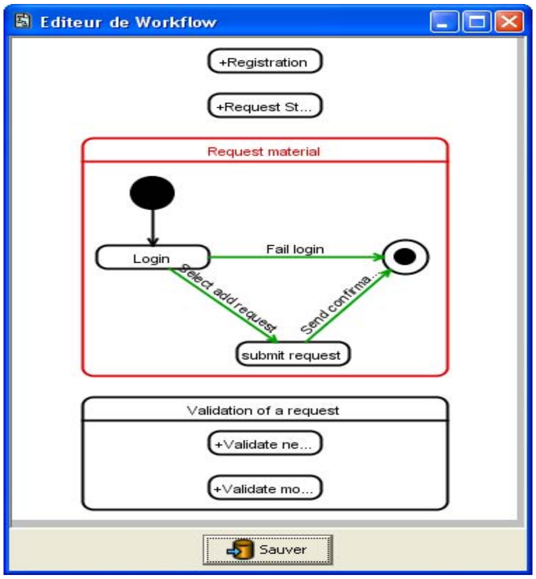


Figure 4 Screenshot of the Workflow editor.

Step 1 – Building the workflow model. Figure 4 reproduce the workflow editor where the workflow is progressively decomposed into processes to end up with tasks. Each workflow is hierarchically defined in embedded rectangles, which could be expanded or reduced as the modeling process is progressing. Each last level rectangle is then filled with a particular process, which is decomposed into tasks (represented by rounded rectangle with a label inside) connected by arcs (represented by arrows). Starting states (represented by filled circles) and ending states (represented by double circles) are added to complete each process. Then arcs are augmented with the conditions imposed to ensure a transition in the workflow. The first step is to register to the system, after, is possible to request new equipment. Once a document is filled, it is sent to “waiting validation” state where an e-mail is automatically sent to the workflow manager to check. Then, each document could be validated (then accepted in the workflow) or not (then rejected in the workflow or returned to the initial state).

Step 2 – Building the task model. For each rounded rectangle in Figure 4, a task model is designed for each user to decompose her task into sub-tasks, to end up with actions. For instance, Figure 5 reproduces the task model associated to the task “Register a manager”, which is particular instance of the rounded rectangle for validation process. Temporal operators specify the temporal constraints governing how information is manipulated in the task. In Figure 5, registering a manager consists in filling the form and in sending this form for validation. Filling the form consists in providing all individual data in a concurrent manner (hence, the ‘|||’ concurrency operator). The address is itself decomposed in street and number, filled in any order.

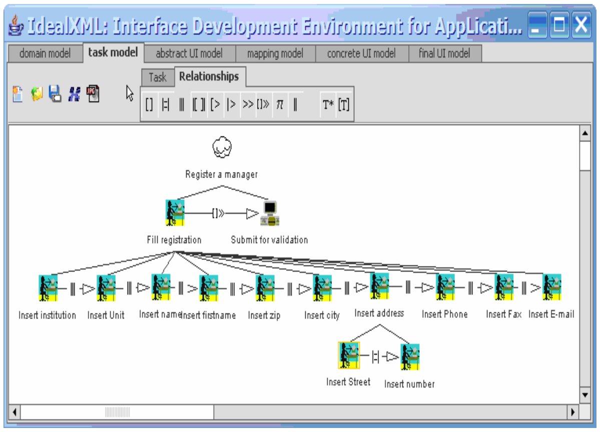
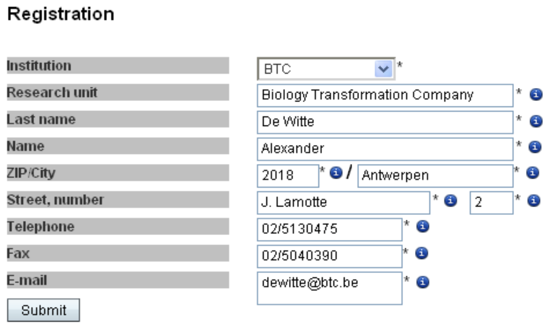


Figure 5 Screenshot of the registration UI generated in HTML.

Step 3 – Deriving the UI model and generating the UI code. Once the task model is provided for each task, a UI model is systematically derived by model-to-model transformation as described in [10]. The explanation of this transformation is beyond the scope of this paper. In short, the decomposition of a task in subtasks, their properties and the temporal operators are exploited to generate a UI model. This is submitted in turn to a model-to-code transformation. For instance, Figure 6 reproduces the UI corresponding to the task model of Figure 5 the first time a manager will register. In this case, a Java/Swing UI is automatically generated. When this document is submitted for validation through an extranet, a HTML version is generated instead from the same model, except that the user name and the password are not visible to the validation person. The Java/Swing version is used for on-line registration while the HTML version is used for validation through the extranet. When the person who is responsible for validation connects to the extranet, the web ble for validation connects to the extranet, the web form is produced instead.



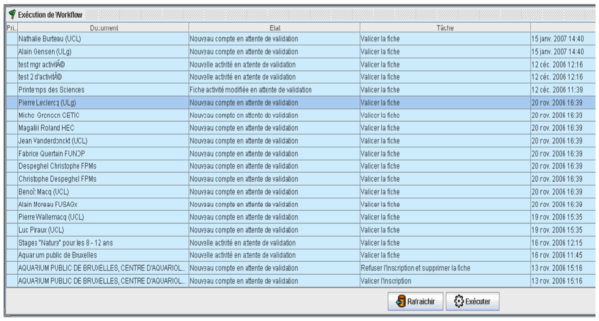
Registration

Institution	BTC *
Research unit	Biology Transformation Company *
Last name	De Witte *
Name	Alexander *
ZIP/City	2018 * / Antwerpen *
Street, number	J. Lamotte * 2 *
Telephone	02/5130475 *
Fax	02/5040390 *
E-mail	dewitte@btc.be *

Figure 6 Screenshot of the registration UI generated in Java/ Swing.

Step 4 – Running the workflow. Once the workflow model and the task model have been produced with the corresponding UIs, the workflow could be executed thanks to a workflow execution engine. It interprets the workflow specification written in UsiXML V1.8 in order to produce workflow agenda for any actor having a job in the workflow. For instance, Figure 7 reproduces a list of pending tasks which have been automatically generated from the workflow model: each time a task has been carried out, a task instance is added in the agenda of the person who is responsible for validation.

Step 5 – Running the workflow. Once the workflow model and the task model have been produced with the corresponding UIs, the workflow could be executed thanks to a workflow execution engine. It interprets the workflow specification written in UsiXML V1.8 in order to produce workflow agenda for any actor having a job in the workflow. For instance, Figure **7** reproduces a list of pending tasks which have been automatically generated from the workflow model: each time a task has been carried out, a task instance is added in the agenda of the person who is responsible for validation.
Each time one of these task instances is selected (thanks to a highlighting bar in Figure **7**), a particular task could be triggered and completed, thus proceeding the workflow to the next step for each instance. For instance, each task will result in an acceptation or a rejection. Since the workflow defined the automated sending of an e-mail, this represents an automatic task which is ensured by the workflow engine, but not by any other person. It is possible, though, to keep an explicit control over automated task by allowing them to be confirmed or differed by the workflow manager. This completes the full process.



Pri...	Document	Etat	Tâche	
	Nathalie Burteau (UCL)	Nouveau compte en attente de validation	Valider la fiche	15 janv. 2007 14:40
	Alain Gensen (ULg)	Nouveau compte en attente de validation	Valider la fiche	15 janv. 2007 14:40
	test mgr activitÃ©	Nouvelle activité en attente de validation	Valider la fiche	12 déc. 2006 12:16
	test 2 d'activitÃ©	Nouvelle activité en attente de validation	Valider la fiche	12 déc. 2006 12:16
	Printemps des Sciences	Fiche activité modifiée en attente de validation	Valider la fiche	12 déc. 2006 11:39
	Pierre Leclercq (ULg)	Nouveau compte en attente de validation	Valider la fiche	20 nov. 2006 16:39
	Michel Gronoen CETIC	Nouveau compte en attente de validation	Valider la fiche	20 nov. 2006 16:39
	Magaliii Roland HEC	Nouveau compte en attente de validation	Valider la fiche	20 nov. 2006 16:39
	Jean Vanderdonckt (UCL)	Nouveau compte en attente de validation	Valider la fiche	20 nov. 2006 16:39
	Fabrice Quertain FUNDP	Nouveau compte en attente de validation	Valider la fiche	20 nov. 2006 16:39
	Despeghel Christophe FPMs	Nouveau compte en attente de validation	Valider la fiche	20 nov. 2006 16:39
	Christophe Despeghel FPMs	Nouveau compte en attente de validation	Valider la fiche	20 nov. 2006 16:39
	Benoît Macq (UCL)	Nouveau compte en attente de validation	Valider la fiche	20 nov. 2006 16:39
	Alain Moreau FUSAGx	Nouveau compte en attente de validation	Valider la fiche	20 nov. 2006 16:39
	Pierre Wallemacq (UCL)	Nouveau compte en attente de validation	Valider la fiche	19 nov. 2006 15:35
	Luc Piraux (UCL)	Nouveau compte en attente de validation	Valider la fiche	19 nov. 2006 15:35
	Stages "Nature" pour les 8 - 12 ans	Nouvelle activité en attente de validation	Valider la fiche	16 nov. 2006 12:15
	Aquarium public de Bruxelles	Nouvelle activité en attente de validation	Valider la fiche	16 nov. 2006 11:45
	AQUARIUM PUBLIC DE BRUXELLES, CENTRE D'AQUARIOL...	Nouveau compte en attente de validation	Refuser l'inscription et supprimer la fiche	13 nov. 2006 15:16
	AQUARIUM PUBLIC DE BRUXELLES, CENTRE D'AQUARIOL...	Nouveau compte en attente de validation	Valider l'inscription	13 nov. 2006 15:16

Figure 7 Screenshot of the workflow being executed.

5. Related Work

The control of workflow in organizations has been addressed using several formalisms and notations such as Petri Nets [17], Statecharts Diagrams [18,21] and UML Activity Diagrams [5]. Currently, several models and design methods [16,19,20] support the development of complex workflow-based applications providing notations for describing rich business process including tool support for designing. Testing and executing processes are also available [6,8] but only a few have been addressed to the development of workflow-based applications over the Web [2]. Similarly, not many investigations are concerned with the automatic derivation of UIs from workflow specifications [7].

All these approaches describe the use, in some way, of tasks/goals to specify a process. However, none defines the identification criteria to recognize the boundaries between process and user task, which is essential to the development of usable UIs. The lack of appropriate analysis of user tasks quite often leads to the implementation of poor user interfaces [1,4]. Then, when UI generation problem is addressed another model should be considered; user's task model. Task models play an important role in UI design because they support the systematic representation of the user activity as opposed to the system activity. Task models indicate the logical activities that an application should supports to reach user's goals [13].

6. Conclusion

This paper has addressed the need for developing a UI corresponding to a workflow model that is explicitly based on the organization business logics. So far, the focus has been put mostly on the system functionalities as opposed to the UI for accessing these functionalities, which is the main contribution of this paper. For this purpose, a conceptual modeling approach integrates the following concept defined through a metamodel: workflow, process, task, domain, job definition, organizational structure, and resources. These concepts along with their attributes have been integrated in UsiXML V1.8, the last version available of UsiXML today. A transformational approach has been followed to progressively decompose the workflow model into processes which are in turn decomposed into tasks. These three models adhere to the principle of separation of concerns: any modification of a process (e.g., in its structure or its temporal ordering while keeping the same tasks) do not influence the task model. The reasoning also holds between the workflow and its underlying processes. In this way, it is possible to optimize the workflow by modifying the underlying process, but without affecting the involved tasks. A real-world case study has been reported and summarized to demonstrate the feasibility of this approach in an industrial context.

Acknowledgments. We gratefully thank the support from the SIMILAR network of excellence, supported by the 6th Framework Program of the European Commission, under contract FP6-IST1-2003-507609 (http://www.similar.cc), the Alban program (www.program alban.org) supported by European Commission, and the CONACYT (www.conacyt.mx) program supported by the Mexican government. All information regarding UsiXML is accessible through http://www.usixml.org. We thank the company DefiMedia (www.defimedia.be) for the development of the editors and the case study exemplified in this paper and for allowing us to use them in the context of this research.

7. References

1. Bodart, F., Vanderdonckt, J. (1994). *On the Problem of Selecting Interaction Objects*. Proc. of BCS Conf. HCI'94 "People and Computers IX" (Glasgow, 23-26 August 1994), G. Cockton, S.W. Draper, G.R.S. Weir (eds.). Cambridge University Press, Cambridge, pp. 163-178.
2. Brambilla, M., Ceri, S., Fraternali, P. *Process Modeling in Web Applications*. ACM Transactions of Software Engineering and Methodology, vol. 5, no. 4, 360-409, 2006.
3. Calvary, G., Coutaz, J., Thevenin, D., Limbourg, Q., Bouillon, L., and Vanderdonckt, J. (2003). *A Unifying Reference Framework for Multi-Target User Interfaces.* Interacting with Computers, vol. 15, no. 3, 289–308, 2003.
4. Dix, A., Finlay, J., Abowd, G., Beale, R. *Human-Computer Interaction*. Prentice Hall; 3Rev Ed edition. 832 p.
5. Dumas, M. and ter Hofstede, A.: *UML Activity Diagrams as a Workflow Specification Language*. In M. Gogolla and C. Kobryn, editors, Fourth International Conference on the Unified Modeling Language (UML 2001), pp. 76-90, Toronto, Canada, 2001.
6. Esposito, D. *Getting Started with Microsoft Windows Workflow Foundation: A Developer Walkthrough.* September 2005. http://msdn.microsoft.com/winfx/reference/workflow
7. Guerrero, J., Vanderdonckt, J.Gonzalez, J.M., *FlowiXML: a Step towards Designing Workflow Management Systems*, Journal of Web Engineering, vol. 4, no. 2, 2008, to appear.
8. IBM Lotus Workflow. http://www-306.ibm.com/ software/lotus/
9. Limbourg, Q. (2004) *Multi-Path Development of User Interfaces.* PhD-Thesis, Université catholique de Louvain, Belgium.
10. Limbourg, Q., Vanderdonckt, J., *UsiXML: A User Interface Description Language Supporting Multiple Levels of Independence*, in Matera, M., Comai, S. (Eds.), «Engineering Advanced Web Applications», Rinton Press, Paramus, 2004, pp. 325-338.
11. Mori, G., Paternò, F. and Santoro, C. *CTTE: Support for Developing and Analyzing Task Models for Interactive Systems Design.* IEEE Transactions on Software Engineering, September 2002.
12. Palanque, P., Farenc, Ch., & Bastide, R. *Embedding Ergonomic Rules as Generic Requirements in a Formal Development Process of Interactive Software.* In Proceedings of IFIP TC 13 Conference on Human-Computer Interaction INTERACT'99 (Edinburg, Scotland, 1-4 September 1999.
13. Paternò,F. *Model-based design and evaluation of interactive applications.* Applied Computing, Springer. 1999.
14. Puerta, A.R. A Model-Based Interface Development Environment. *IEEE Software* 14, 4 (1997), pp. 41-47.
15. Russell, N., van der Aalst, W.M.P., ter Hofstede, A.H.M. and Edmond, D. *Workflow Resource Patterns.* In the 17th Conference on Advanced Information Systems Engineering (CAISE'05). Porto, Portugal. 13-17 June.
16. Stavness, N. and Schneider, K.: *Supporting Flexible Business Processes with a Progression Model*. Workshop: Making Model-based UI Design Practical: Usable and Open Methods and Tool, 2004.
17. van de Aalst, W.M.P.: *The Application of Petri Nets to Workflow Management. Journal of Circuits, Systems, and Computers*. Vol. 8, No. 1, February 1998, pp. 21-66.
18. W3C: State Chart XML (SCXML): *State Machine Notation for Control Abstraction 1.0.* W3C Working Draft 5 July 2005. http://www.w3.org
19. White, S. *Business processing modeling notation* (BPMN), version 1.0, http:// http://www.bpmn.org/
20. WfMC 2006. *Workflow Management Coalition.* http://www.wfmc.org/
21. Wodtke, D., Weikum, G. *A Formal Foundation for Distributed Workflow Execution Based on State Charts*. In Proc. of the 6th International Conference on Database Theory (ICDT '97), Springer-Verlag, LNCS Series, p. 230-246, 1997.