Model-Based User Interface Design in the Context of Workflow Models

Renate Kristiansen and Hallvard Trætteberg

Department of Computer and Information Science,
Norwegian University of Science and Technology,
Sem Sælands vei 7-9, N-7491 Trondheim, Norway

Abstract. Within ERP systems, workflow models are used by business analysts to specify which business processes the system supports. The workflow model specify which actors that performs what activity in what sequence and the required resources. Within user interface (UI) design task models are used to develop task-centric user interfaces. Task-centric UIs can increase systems' usability as it focuses on the end-user. In this article we will show how task models together with other models used in the field of model-based UI design can be created within the context of already existing workflow models. We show how standard tasks can be defined as editable UI components allowing role-based composition of the UI with support from the workflow model.

Keywords: ERP, MBUID, Workflow, Task modeling.

1 Introduction

Enterprise Resource Planning (ERP) systems are off-the shelf business applications providing a tightly integrated solution to organizations' information system needs [27]. ERP benefits include best practice business processes, real-time access to information and shared practices across the entire enterprise. One important characteristic of ERP systems is the fact that they are pre-built software packages designed to meet the general needs of a business sector instead of the unique requirements of a particular organization [1]. To be able to deliver such huge software packages, ERP vendors use different business process models in their overall description of the system to describe the supported processes and organizational structures together with the structure of data and objects [13]. The reference models are founded upon what the vendor considers being the industrial best practices, that is, the most efficient way the business processes should be structured [5]. SAP uses Event Process Chain (EPC) models to document the system's functionality [12] while Microsoft uses Business Process Modeling Notation (BPMN) to describe the business domain. These are descriptive models documenting the existing software (in contrast to prescriptive models that are used as a specification of what to create) [15].

In this article we use models and information collected from a large company developing ERP systems and show how prescriptive task models can be connected to descriptive workflow models. The company currently runs a project

M. Winckler, H. Johnson, and P. Palanque (Eds.): TAMODIA 2007, LNCS 4849, pp. 227–239, 2007.

where the ERP system's functionality is modeled using workflow modeling. The intention is to use the models as documentation in implementation projects. In addition there is an interest in investigating how these models can be reused in other contexts. We want to show how they can take advantage of model-based user interface design (MBUID) to allow flexible role-centered composition of user interfaces in the context of the workflow models. Role-based access and portal solution is considered the answers to the severe usability problems identified in ERP systems [7].

A challenge with role-based systems is how to keep the number of roles on a manageable level. When new functionality is added, should this result in the creation of a new role? A single person typically fulfills several roles, and the combination of roles users have differs among companies. Flexibility in creating user interfaces (UI) for various combinations of roles is therefore important. We will explore a systematic way to define what needs to be included in the UI for one particular user based on her participation in the workflow process. The workflow model defines what tasks need to be fulfilled and their possible ordering; hence the workflow model is suitable as a 'frame' for creating task models. A task model typically focuses on modeling the work of an individual user.

A short introduction to task and workflow modeling is given in section 2, and we discuss how MBUID and workflow models by virtue of coming from different research traditions have differences in concepts, focus and pragmatics. Our work take advantage of existing modeling languages proved useful in one context, and proposes how they can be combined to add value in an industrial context. Section 3 describes relevant aspects of the ERP vendor organization, and describe our approach by showing a practical example. In section 4 we explain how to make use of pattern structures to compose role-oriented user interfaces so that the highly detailed, executable dialogue models can be wrapped into easier to work with lesser detailed components. We have discussed our approach with the user interface developers in the company and report some of their first-hand comments. Finally, in section 5 we conclude and give some notes on future work.

2 Different Modeling Traditions and Their Relation

We will give a short introduction to task modeling and explain how task models relate to other models used in MBUID. Workflow modeling is then introduced before the relationship between task modeling and workflow modeling is discussed. Based on our discussion we argue for the choice of modeling languages used in the case study.

2.1 Task Models and Model-Based User Interface Design

Task modeling is often used first in the analysis phase to understand and communicate the problem domain (resulting in a descriptive model), and later on as a prescriptive task model for the system to be designed (as e.g. the DUTCH

method using GTA [33]). Examples of task modeling languages are: Méthode Analytique de Description des tâches (MAD)[26], Task Knowledge Structure (TKS) [11], GroupWare Task Analysis (GTA) [32] and ConcurTaskTrees(CTT)[19] which all support designers by hierarchically decomposing tasks, defining objects manipulated and the role responsible for performing the task.

The vast number of task modeling notations results in semantic and syntactic differences which are discussed by e.g. [14] and [35]. Based on their analysis a uniform task model is created which includes concepts like: task and goal hierarchies, operators that express temporal constraints between task, some role concept to deal with co-operative aspects, and objects with possible actions.

Task models are considered one of the viewpoints in the model-based community [20]. Viewpoints are related to both abstraction level and focus of the model. Is the level of detail high and is the focus on the task or on the UI? Models with different viewpoint are:

1. *Task model and object model* represent the highest level of abstraction and their focus is on user's goals, tasks and what objects that are manipulated (the object model is often referred to as a domain model).
2. The second layer is the *abstract user interface* describing the structure and behaviour of the user interface [29].
3. The third level involves building a *concrete user interface* specification defining the platform dependent look and feel of the interface.
4. The fourth level is the *final user interface* which is the running interface implemented on a specific software environment.

Model-based user interface design (MBUID) processes often start with a task related model that is evolved through an incremental approach to the final UI [4]. In each of the transformation phases the designer has the possibility to manually change the generated artefact, and the modification is preserved when regenerating the UI.

The concept of tasks is very similar to that of processes (in a workflow); the difference is mainly that of scope and focus. Processes typically relate directly to organizational goals, while tasks focus on the goal and actions of individual users playing a role. Hence, a task model may be seen as a refinement of a process model, in the context of a specific user role [28].

2.2 Workflow Modeling

Workflow models focus on how work is done to accomplish some organizational goals. It defines how documents, information and tasks are passed between human or other actors in the enterprise [25]. Important workflow characteristics are tasks/activities that are performed by role-playing persons using supporting tools that give access to various shared information resources [17] [2].

In the literature there is confusion about the differences on business process models and workflow models. According to [10] a business process is defined by a process definition and managed by a workflow management system. Hence the process model includes the workflow model.

Many workflow modeling languages have formal semantic built on Petri nets [25]. A Petri net is a directed graph with a mathematical formalism facilitating visual modeling on the one hand and formal analysis, verification and validation of the model on the other. An example of such a language is Yet Another Workflow Language (YAWL) [31]. Informal workflow modeling languages includes Event-Process Chain (EPC) [12], Action Port Model (APM) [2] and Business Process Modeling Notation (BPMN) [8]. BPMN is defined by the Object Management Group and offers a rich notation for workflow modeling. The notation supports decomposition of processes into sub-processes and tasks. A task is an atomic activity and cannot be decomposed further. A task can usually be performed by an end-user and/or an application [8].

2.3 The Relation Between Models

We will use a two-dimensional representation framework to discuss how different modeling notations used by different research traditions relate to each other. The representation space is shown in figure 1. The problem/requirement-solution/design dimension say something about how tight the model is connected to the final design. While problem-oriented notations describe goals and requirements to the design in abstract manners, solution-oriented notations describe aspects of the artifact we are designing and give specific details on the environment the artifact will act in. Along the problem-solution axis models have different granularity, which is the second dimension. Business processes (BP) have high granularity as they describe the activities businesses undertake to reach their business goals. Workflow models (WF) need to include more details to be executable by a workflow management system. The task model (TM) is partly overlapping the workflow model since the lowest level of workflow models usually is a task performed by one actor. A task performed by one actor is typically the highest level in a task model describing which sub-task that must be completed to reach the goal of its parent task. Dialogue models (DM) add details of the functionality and the interaction that the UI provide. Dialogue models and task models together cover the same area of the problem-solution axis as workflow models, but have lower granularity. A model of the concrete interaction (CIM) is very close to the final system, and has a low granularity specifying both visual details (e.g. layout, widget usage, etc.) and interaction (keyboard and mouse).

As we have explained, different modeling notations cover different areas of the representation space. We emphasize the following differences between workflow and task modeling:

- **Different research tradition:** Workflow models have their origin in organizational theory. Hammer stated in [9] that usability was a "second order issue" and should only be considered when all other functionality has been considered. "The important thing for automated office application were: (1) functionality; (2) functionality; (3) nothing; (4) functionality; and only then, (5) everything else" [6, page 119]. Task models come from the field of human

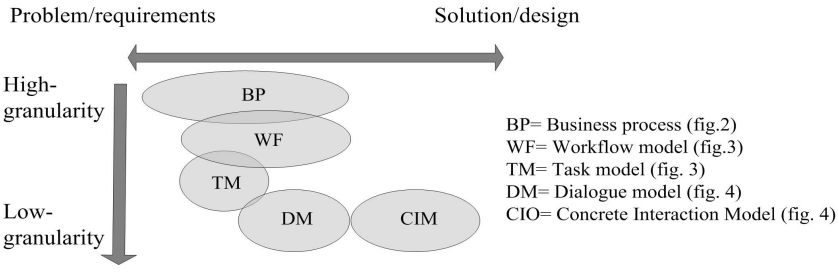

Fig. 1. A representation framework for classification of modeling languages

factors [21] and are used with the aim of increasing usability of computerized systems. This naturally leads on to the second difference:

- **Difference in focus:** In workflow models the focus is on how to reach organizational goals. In task modeling the focus is on the goals of individual users. It is important to be aware of the difference between organizational goals, the individual goals and how they are related as they might not be aligned [34].
- **Differences in concepts:** Section 2 pointed to the mixture in concept definition between task models, and the same mixture is present if we consider concepts across task models and workflow models. As [21] point out, a concept defined in a task model can be used in a workflow model with a different meaning. This gives a pragmatic problem across modeling languages.

When a combination of workflow models and task models are considered these differences must be taken into consideration. In the next section we choose which modeling languages we will use in our case study.

2.4 Selection of Modeling Languages

Our case study company use BPMN for workflow modeling, so these models are kept and used directly. Because of the considerable overlap in workflow and task modeling concepts, we have considered the possibility of extending BPMN so that it also can be used for task modeling. However, because of the difference in focus and use of concepts, we think that it is useful to have two separate notations and instead emphasize that the focus shifts from being about organizational goals to considering individual users' goals. BPMN uses swimlanes for modeling the responsibility of actors, which generally is problematic when it comes to decompositions. For these reasons we have chosen to use Taskmodl [29] which is a task modeling language with its origin both from the workflow tradition and the tradition of task modeling and analysis. It was created with the aim to narrow the gap between workflow and task modeling and it is based on the workflow modeling language APM [2]. The main APM concepts are interpreted in the context of task modeling [29] resulting in a notation supporting the traditional hierarchical sequence-oriented style typical for task modeling languages.

Taskmodl supports decompositions of BPMN tasks into user-centric task hierarchies, specify resources, actors and sequence constraints. To model the user interface we will use Diamodl [29] [30] which is an executable modeling notation for abstract user interface specification developed to be used together with Taskmodk. Central for our approach is that the UI models should be editable and result in a running UI. Diamodl satisfies this requirement.

3 Models from a Real World Company and Their Relation to MBUID

We will show how a workflow model developed at a large international ERP vendor can be used as a starting point for creating a task-oriented user interface. We call the vendor ProERP and explain the models they currently use in their developing organization before we show how a MBUID approach can be pursued in connection to this information.

To aid the software development ProERP uses a model of the business domain as a common point of reference. The model is split into two representations:

- **User and Organizational Model** has the individuals and their organizational relationship as focus. The users are described using Personas [3] [24]. A Persona is an archetype of an actual user and included in the Persona description is information stating what roles a Persona can take and what tasks he or she is responsible for. The numerous Personas are grouped into departments, and each department is illustrated by organizational charts.
- **Business Process Model** has a supply chain perspective and is decomposed into the activities involved in the business process. The processes are grouped together and placed within departmental borders, showing which department is responsible for which processes. The business process shown in figure 2 is one of seven business processes grouped under the "Operations" department.

The two model representations describe the same world, but with different perspective. The information used in this case study is based on this generic model together with documentation that was provided by two other projects. ProERP had a project that decomposed the business process model into BPMN diagrams and the uppermost diagram in figure 3 is from that project. In addition, documentation from a user interface development project lead by the UI design team is used.

When new functionality is designed, the Personas that should participate are identified and used as leading actors when developing scenarios [22] describing the functionality. Detailed information concerning the business domain and what is required for the new functionality is provided by domain experts participating on the design project. The UI design specification consists of sketches of the user interface drawn with a drawing tool and supplemented by textual description of the interaction. For usability evaluation Powerpoint slides are used.

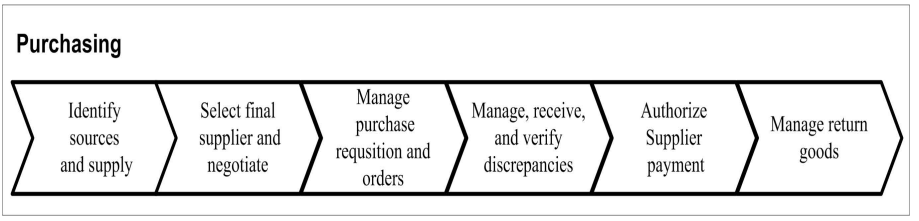

Fig. 2. Example of business process

The process steps in figure 2 are further decomposed into BPMN diagrams showing which activities that are carried out to execute the process. In topmost diagram in figure 3 one of the decompositions under "Manage purchase requisitions and orders" is shown. Sara (which is a Purchaser) first creates a purchase order (PO). The PO is then transferred to the stack of outgoing, awaiting PO's and she can choose when she wants to communicate the PO by sending it to the supplier. The supplier must send an order acknowledgment before a pre-defined time has elapsed, otherwise a rule triggers and the PO is put into the stack of PO's awaiting for order acknowledgment. A reminder should then be sent to the supplier.

Each of the boxes in the BPMN model is a task suitable for one person and can be considered the highest level in a task model hierarchy modeled in a task modeling language. The decomposition of the Create PO task is shown in the task structure in the lower part of the figure 3. The rounded rectangles are tasks, with an identifier and a name in the top compartment. The lower compartment is optional and contains the resources necessary for the task and the actor performing the task (shown in the parent task). A middle compartment can be added with a task description, but we have not used this compartment in this figure. The resources that are sent between tasks are flow resources triggering the execution of the following task (e.g. PO and Product). The circle enclosing the arrow means that the tasks need to be executed in a fixed sequence. To create a purchase order Sara has to find the products, add them to a requirement list and then generate a purchase order from the requirement list. How to accomplish a task is a question of design and requires domain knowledge and knowledge about constraints in the software. In the current user interface design process, the designers in ProERP create scenario descriptions and sketches of the user interface to describe the functionality.

The task model describe "what to do" but does not include the "how to do it" knowledge describing how users accomplish a task in the UI using interaction objects, state and data flow specification. This is information typically specified in a dialogue model. Dialogue models are suitable for representing abstract interaction tasks as selecting an element from a set of elements or pushing a button to trigger some functionality. We have developed dialogue models for each of the leaf node tasks of the Create PO tasks tree in figure 3 based on the UI designers description of how the UI should look like and behave. As it seems to be essential for the user interface designer to have complete control of the design process, we have not pursued a more formal derivation of the dialogue model from the task model as done e.g. by [16].

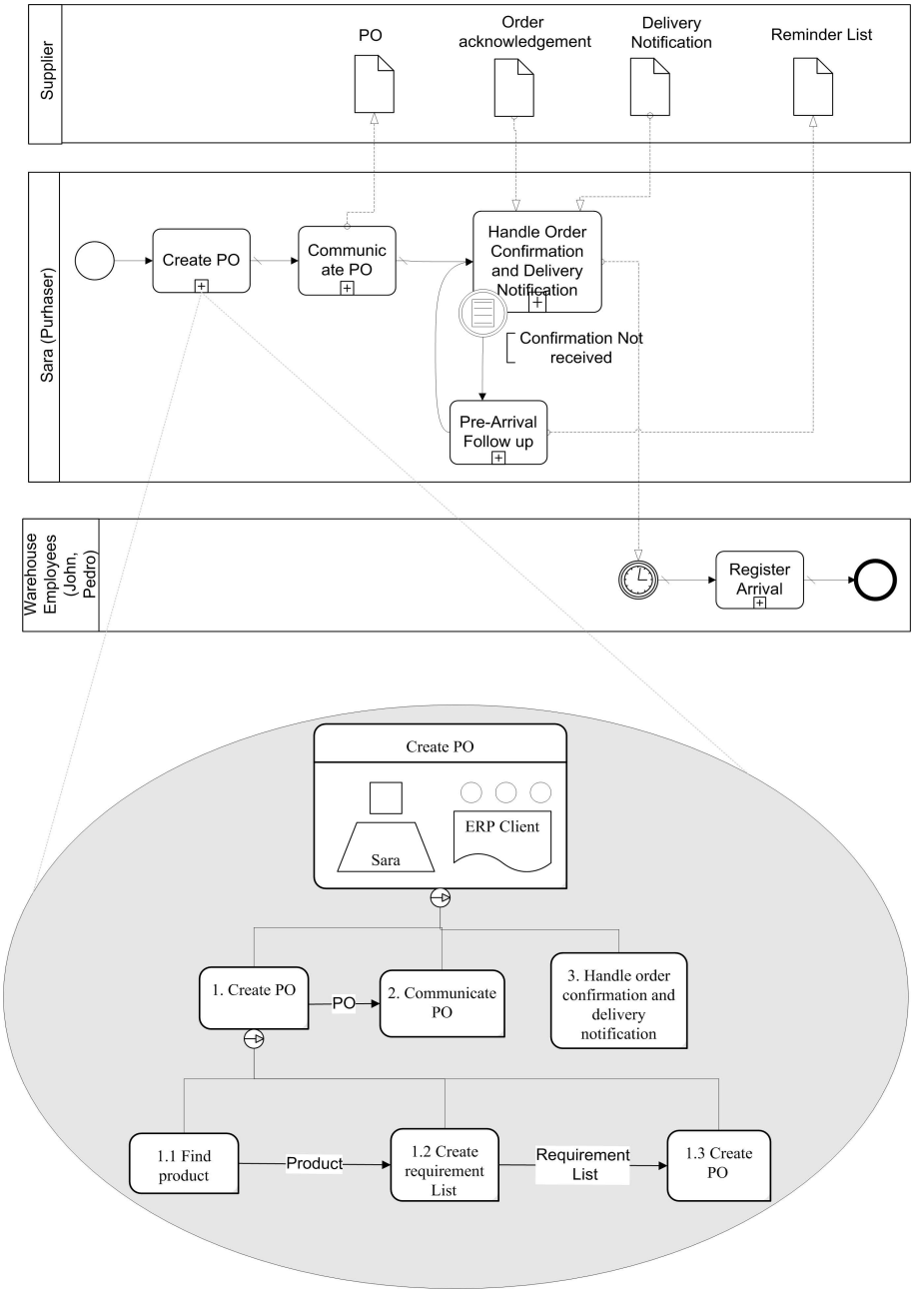

Fig. 3. Workflow model showing the process in which a PO is manually created and a task model that show the decomposition of one of the tasks

The abstract user interface model is drawn using DiaModl notation, and figure 4 shows the dialogue model for the task 1.1 Find Product. *Interactors* are drawn as rectangles with a name describing their functionality. Attached to the interactor's left side are *gates* which define user input (pointing outwards) and output to the user (pointing inwards). The free floating triangle is a computation with functionality as indicated by the description (match). The edges between elements are connections, and define flow of data. The Product object is from the domain model. To find a product the user first search for the product by typing the product number. For each digit the user types, a match function filters the product list and highlights the first product that match. Some of the attributes of the supplier and product object is displayed to the user.

The dialogue model shows an abstract model of the interaction which can be used as a specification for the concrete implementation of the UI. The interactor's input/output signature determines a set of concrete interactor objects (e.g. a set of standard widgets) that can replace the abstract user interface component. The lower model in figure 4 show how the abstract interactors are replaced with concrete ones matching their input/output signature.

4 A Role-Oriented Approach to Dialog Composition

When creating a homepage for a specific user with a different role composition than the ones in the pre-defined Persona description, the BPMN diagram can be used as a starting point for creating task models. Using task models, the necessary steps for solving the BPMN tasks identified in the workflow model can be modeled in a user-centric way. Our experience from the case study indicate that each of the BPMN tasks are candidates for being the top level of a task structure accessible directly from the employee's personal homepage.

Since the low-level tasks encapsulate a dialogue structure, a task-oriented user interface can be created by assembling the dialogue fragments for the required set of tasks. As noted by [18] modeling the user interface of an interactive system in sufficient details to be "run" soon becomes an overwhelming task - and an abstraction mechanism is required to get the "big picture" of the system. To reduce this complexity we suggest using task model components as patterns for how standard tasks can be solved. Patterns give a generic solution to a problem and should be adapted to the specific problem [19]. Composing a UI then will consist of defining which tasks are needed, plug together the dialogue fragments and do possible adaption to the standard structure. For example if a specific user needs to search for a product using supplier name instead of product number as the abstract dialogue model in figure 4 prescribe, it is possible to edit the model to support search on supplier name by adding the supplier object as a resource in the interactor.

In large software development organizations like ProERP, UI designers and developers work in different, disperse teams. The designer wants to be in charge of the UI design, but as paper prototypes "do-not-fly" in ProERP, they need to spend much time drawing the UI and "implementing" the interaction using

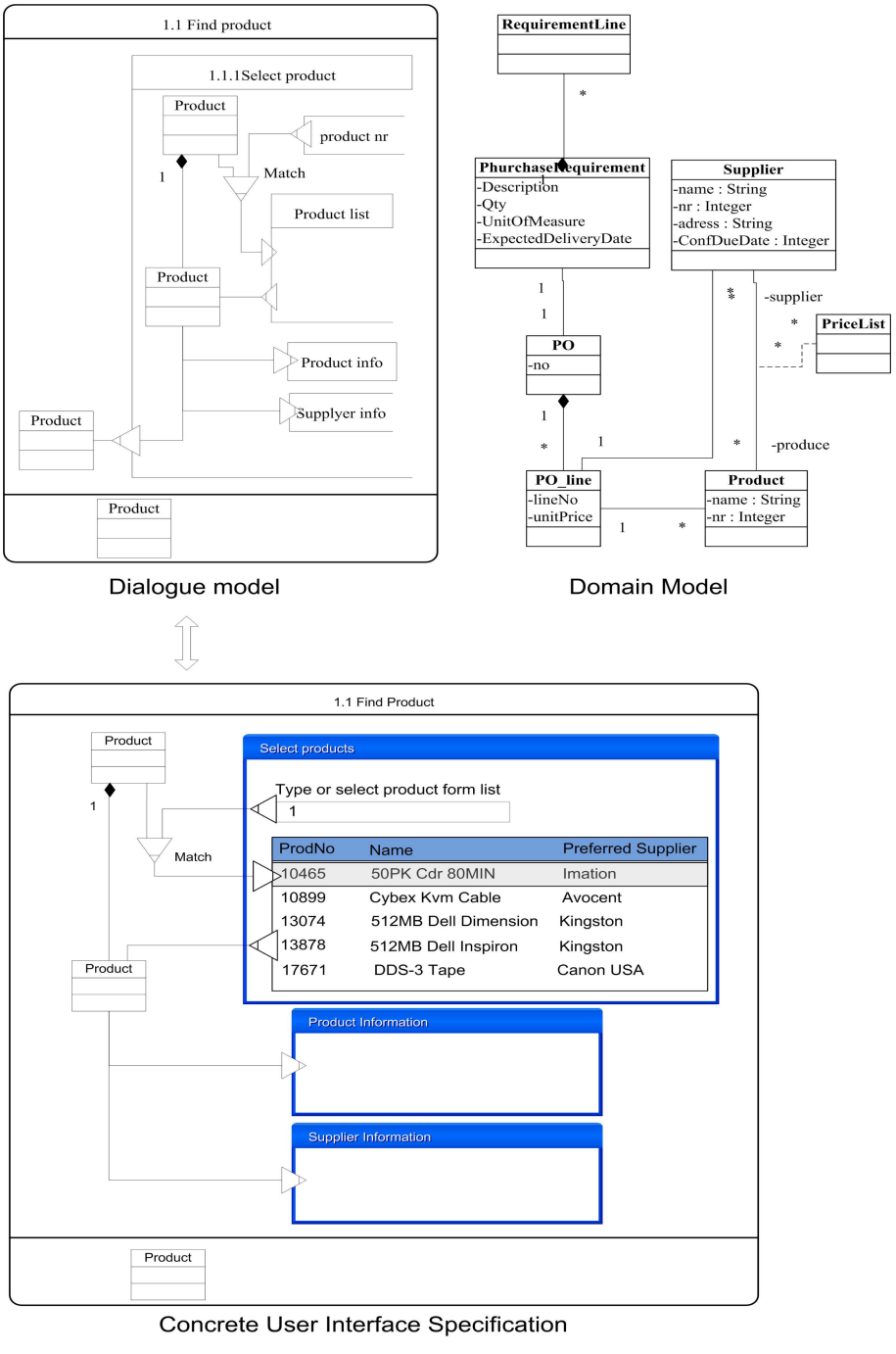

Fig. 4. Dialogue model of the task "1.1 Find Product", the domain model and a concrete UI specification of the abstract dialogue model

power-point slides. On the developer side it is required to have a précis description of the UI and what specific business objects that are used. Having a library of standard UI components will make it quicker for the UI designer to create a prototype of a running UI which can both be used in usability testing and as design specification to developers. It is important that the designer is still in charge of the UI design and has complete control of the model.

Presenting our approach to UI designers at ProERP they suggested that if they could be in charge of designing and testing such standard components, the developers could implement them and return them to the UI design team. The design team could then use the components as a resource when designing new UIs. Since designers and developers are grouped into two disperse teams the UI design ideas are communicated through scenarios for some illustrative example cases that demonstrate the principles the UI should follow. The information is supplemented with design guidelines handed over to the developing team. As a consequence the UI designers have not complete control over the look-and-feel of the final UI. Having a library of standard UI components for solving common tasks will help assure consistence across different UIs.

In this paper the approach is presented as a linear process moving from the workflow diagram towards the final user interface using several models along the way. However, this is done purely for explanatory reasons. For UI designers, it is also relevant to move from a concrete UI design towards the abstract design. After all, people tend to prefer to think in concrete terms instead of abstract terms. It is equally relevant to support starting with a concrete design for thereafter specifying its abstract and formal structure and behavior.

It is important that models are used as design aid for the UI designers. We do not believe that an automation of the design process will be appreciated. Presenting this approach to the UI designers in ProERP they commented that having a library with standard tasks (designed by them) that are editable would be a feature, not a limitation for their work.

5 Conclusion and Future Work

In an ERP domain many of the same or similar tasks are performed by different people having different subsets of roles within an organization. We have proposed an approach where models from the field of model-based user interface design are used in the context of workflow models to allow role-centric composition of ERP systems' UI. As the suggested UI components are defined using an executable modeling notation, they can be edited and thereby allowing tailoring of the UI. Typical cases where editing would be relevant is when a user should be allowed to take shortcuts compared to what is considered the standard process (e.g. create a purchase order without getting a requisition from the manager).

In the suggested approach the transition from a task model to a dialogue structure is a matter of design decisions from the UI designer. We do not provide design support for determining a useful mapping from the task model to the abstract user interface model as done in the methodology proposed by [23].

They provide a decision tree for selecting an abstract interaction object fitting the task. We need to consider whether such support would be appreciated by the UI designers in the ERP domain. Also, the appropriate size of the UI components needs to be investigated further.

References

1. Brehm, L., Heinzl, A., Markus, M.L.: Tailoring erp systems: A spectrum of choices and their implications. In: Proceedings of the 34th Hawaii International Conference on System Sciences (2001)
2. Carlsen, S.: Conceptual Modeling and Composition of Flexible Workflow Models. PhD thesis, Norwegian University of Science and Technology (1997)
3. Cooper, A.: The inmates are running the asylum: Why high-tech products drive us crazy and how to restore the sanity. Sams Publishing (1999)
4. Cuppens, E., Raymaekers, C., Coninx, K.: A model-based design process for interactive virtual environments. In: Gilroy, S.W., Harrison, M.D. (eds.) Interactive Systems. LNCS, vol. 3941, pp. 225–236. Springer, Heidelberg (2006)
5. Curran, T.A., Ladd, A.: SAP R/3 Business Blueprint: Understaning Enterprise Supply Chain Management. Prentice-Hall, Englewood Cliffs (2000)
6. Diaper, D., Sanger, C.: Tasks for and tasks in human-computer interaction. In: Interacting with Computers, vol. 18, pp. 117–138. Elsevier B.V, Amsterdam (2006)
7. Gilbert, A.: Business apps get bad marks in usability (2003), Accessed at: `http://news.com.com/2100-1017-980648.htmlon8/12/2006`
8. Object Management Group. Business process modeling notation specification, final adopted specification dtc/06-02-01 (2006)
9. Hammer, M.: The oa mirage. Datamation 30, 36–46 (1984)
10. Hollingsworth, D.: Workflow management coalition -the workflow reference model. Document Number TC00-1003 (January 1995)
11. Johnson, P., Johnson, H., Waddington, R., Shouls, A.: Task-related knowledge structures: Analysis, modeling and application. In: People and Computers IV, pp. 35–62 (1988)
12. Keller, G., Taufel, T.: SAP R/3 Process-Oriented Implementation. Addison-Wesley, Reading (1998)
13. Klaus, H., Rosemann, M., Gable, G.G.: What is erp? Information Systems Frontiers 2(2), 141–162 (2000)
14. Limbourg, Q., Vanderdonckt, J.: Comparing task models for user interface design. In: Diaper, D., Stanton, N. (eds.) The Handbook of Task Analysis for Human-Computer Interaction, vol. 6, pp. 135–154. Lawrence Erlbaum, Mahwah (2003), `http://citeseer.ist.psu.edu/limbourg03comparing.html`
15. Ludewig, J.: Models in software engineering an introduction. In: Software and Systems Modeling, vol. 2, pp. 5–14. Springer, Heidelberg (2003)
16. Luyten, K., Clerckx, T., Coninx, K., Vanderdonckt, J.: Derivation of a dialog model from a task model by activity chain extraction. In: Jorge, J.A., Jardim Nunes, N., Falcão e Cunha, J. (eds.) DSV-IS 2003. LNCS, vol. 2844, pp. 203–217. Springer, Heidelberg (2003)
17. Marshak.: Workflow: Applying automation to group processes. In: Coleman, D. (ed.) Groupware - Collaborative Strategies for Corporate LANs and Intranets, pp. 143–181. Prentice Hall PTR, Englewood Cliffs (1997)

18. Paquette, D., Schneider, K.: Interaction templates for constructing user interfaces from task models. In: Jacob, R.J.K., Limbourg, Q., Vanderdonckt, J. (eds.) Computer-Aided Design of User Interfaces IV, pp. 223–234. Springer, Heidelberg (2004)
19. Paternó, F.: Model-Based Design and Evaluation of interactive Applications. Springer, Heidelberg (2000)
20. Paternó, F.: Model-based tools for pervasive usability. Interacting with Computers , 1–25 (2004)
21. Pontico, F., Farenc, C., Winckler, M.: Model-based support for specifying eservice egovernment applications. In: Coninx, K., Luyten, K., Schneider, K.A. (eds.) TAMODIA 2006. LNCS, vol. 4385, Springer, Heidelberg (2007)
22. Preece, J., Rogers, Y., Sharp, H.: Interaction Design: beyond human computer interaction. John Wiley & Sons, Chichester (2002)
23. Pribeanu, C., Vanderdonckt, J.: A methodological approach to task-based design of user interfaces. Studies in Informatics and Control 11, 145–158 (2002)
24. Pruitt, J., Grudin, J.: Personas: practice and theory. In: DUX 2003: Proceedings of the 2003 conference on Designing for user experiences, pp. 1–15. ACM Press, New York (2003)
25. Salimifard, K., Wright, M.: Petri net-based modelling of workflow systems: An overview. European Journal of Operational Research 134, 664–676 (2001)
26. Scapin, D., Pierret-Golbreich, C.: Towards a method for task description: Mad. In: Work With Display Units (WWU 1989) (1989)
27. Shehab, E.M., Sharp, M.W., Supramaniam, L., T.A.: Spedding. Enterprise resource planning: An integrative review. Business Process Management Journal 10(4), 359–386 (2004)
28. Trætteberg, H.: Modeling work: Workflow and task modeling. In: CADUI 1999 (1999)
29. Trætteberg, H.: Model-based User Interface Design. PhD thesis, Norwegian University of Science and Technology (2002)
30. Hallvard., Trætteberg.: A hybrid tool for user interface modeling and prototyping. In: Computer-Aided Design of User Interfaces V. Springer Science+Business Media B.V. (2006)
31. van der Aalst, W.M.P., ter Hofstede, A.H.M.: Yawl: Yet another workflow language (revised version). Information Systems 30, 245–275 (2005)
32. van der Veer, C.C., Lenting, B.F., Bergevoet, B.A.J.: Gta:groupware task analysis-modeling complexity. Acta Psycologica 91, 297–322 (1996)
33. van der Veer, G., van Welie, M.: Task based groupware design: putting theory into practice. In: Proceedings of DIS 2000 (August 2000)
34. van Welie, M.: Task-based User Interface Design. PhD thesis, Vrije universiteit (2001)
35. van Welie, M., van der Veer, G.C., Eliëns, A.: An ontology for task world models. In: Proc. Int'l Eurographics Workshop Design, Specification, and Verification of Interactive Systems (DSV-IS 1998), pp. 57–70 (1998)