# Using Task Modelling Concepts for Achieving Adaptive Workflows

Carsten Eichholz, Anke Dittmar, Peter Forbrig

University of Rostock,
Institute of Computer Science,
A.-Einstein-Str. 21,
18059 Germany
{eichholz|ad|pforbrig}@informatik.uni-rostock.de

**Abstract.** Business processes are usually described by abstract workflow specifications. However, existing workflow descriptions are often too restricted to reflect the true nature of work. For instance tasks might be added or deleted during execution. The presently available workflow management systems insufficiently support the desired flexibility for workflows. In this article we present an approach, how certain kinds of adaptability can be achieved on the base of task modelling combined with the principle of "Order & Supply". Task models offer means to describe the way humans perform tasks in cooperation focussing on the individual level. We show that the principles of task modelling can also be used for cooperative workflow models providing means on group level.

## 1. Introduction

The introduction of workflow management systems (WFMS) in companies has emerged as a major advantage to plan, control, and organise a company's business processes. Workflow processes can be modelled and executed, thus the business process is assisted by a software while it is running. Chiefly, the flow of documents through a process, but also scheduling, notification, and other communicative tasks are assisted.

Although these advantages are of great help, it is often desired to keep workflows more flexible. The definition of a business process cannot be completely foreseen at its beginning. A lot of changes and adaptations are done while the process is already running. The presently available workflow management systems do scarcely support adaptability for workflows as the following statements show: "Traditionally, workflow management systems have not been designed for dynamic environments requiring adaptive response."[1] "It is widely recognised that workflow management systems should provide flexibility. [...] However, today's workflow management systems have problems dealing with changes."[13]

This is constituted by several problems, e.g. changing workflows should be possible even during execution, but what happens with already started tasks? Are the renewed or the extended tasks in execution still consistent to old tasks that have been

finished in the workflow? Because of these and other questions, *adaptive workflows* have become an important research area.

In this paper, we present an approach for dealing with workflow adaptability by using task models. Recent approaches in task modelling offer means to specify more flexible task models. We show that certain kinds of adaptability for workflows can be solved using task models. In section 2 we introduce the ideas behind the concepts of task analysis and workflows, and show that the similarity between them can be a basis for our approach. Section 3 gives an overview of the question of adaptation in workflows. Different aspects of adaptability are presented, mainly based on the paper of van der Aalst [13]. In the subsequent section 4, we show with our method of "Order & Supply" how certain aspects of adaptation can be solved by using task models. This method is finally illustrated in an example presented in section 5. Some related approaches concerning adaptivity in workflows are shown and compared in section 6 while in the last section some conclusions of our approach are summarised as well as some perspectives on future expectations are presented.

## 2. Task Models and Workflows

In this chapter we briefly characterise the two main concepts our approach is based on, namely task models and workflow specifications. Trætteberg compared workflow models and task models in [12]. He states that both "essentially describe the same domain, but at different levels". While workflows support work on the organisational and group level, task models rather consider the individual level of work. We show that the similarity between these concepts allows an implementation of certain adaptation aspects desired in workflows by use of task models.

### 2.1. Task Models

Task models play an important role in the model-based design of user interfaces for interactive systems. The process of interaction—the process of working with a software system—is modelled with the aim "to have a structured method for allowing designers to manage such a complexity"[7] as it emerges in user interface design. According to [7], task models can be useful for the purpose of:

- Understanding an application domain
- Recording the results of interdisciplinary discussions
- Designing new applications consistent with the user conceptual model
- Analysing and evaluating usability of an interactive system
- Supporting the user during a session
- Documenting interactive software

In addition, we propose to use task models for coordinating tasks and activities in a more general way, i.e. coordination of activities in business processes.

Tasks consist of activities that are performed to reach a goal, which can be considered as modifying a system into a desired state. Tasks are structured hierarchically (see hierarchical task analysis, HTA [2]), forming so-called task-trees. Thus, tasks can be described at different levels of abstraction and detail. Between activities exist certain dependencies defining the order of execution. Often, such dependencies are described by a set of temporal equations, using predefined temporal operators. Task models can therefore be seen as a combination of HTA and a description of temporal execution. Paternò et al. developed ConcurTaskTrees, a method for task modelling using these principles. They define temporal operators [9] like:

- T1|||T2 Interleaving (parallel execution)
- T1|=|T2 Order independency
- T1>>T2 Enabling (sequential execution )
- T1[>T2 Deactivation
- T1[]T2 Choice
- [T] Option
- T* Iteration

In the ConcurTaskTree notation, the dependencies between activities in the task tree are included into the diagrammatic notation. Unary operators are marked at a task's name (e.g. "*" at "enter terms" in **Fig. 1**) and binary operators are put between two tasks, read from left to right (e.g. "|||" between "collect terms" and "define terms").

Different types of tasks are identified: abstract tasks, user tasks, interaction tasks, and application tasks. Later extensions of this method introduce cooperative trees, where sub-tasks can be distributed to and performed by different roles/employees (cf. [8]). This allows modelling task execution not only at individual but at group level as well. **Fig. 1** shows an example for a cooperative task tree, as it can be modelled in CTTE, a tool supporting the ConcurTaskTree modelling approach.

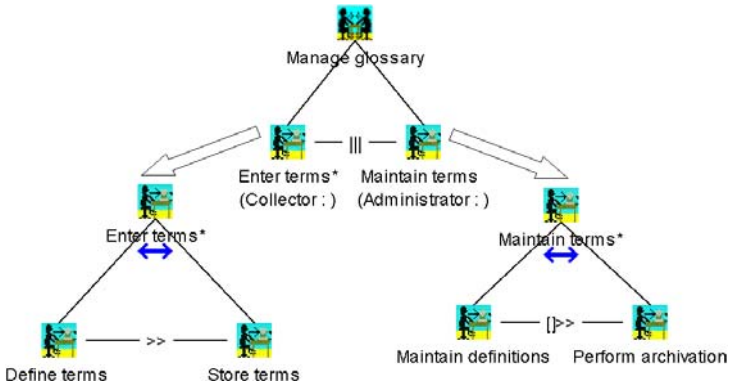


**Fig. 1.** Cooperative task tree for the task "manage glossary".

This example models the task of managing a glossary. The sub-tasks "enter terms" and "maintain terms" are assigned to the roles "Collector" and "Administrator"

respectively. Each role is assigned a sub-task tree and performs the execution of it. The broad arrows symbolise the distribution of work (not part of the CTT notation). The double arrows mark the sub-tasks as being part of a cooperative task. CTTE allows to animate the execution of such a cooperative model.

### 2.2. Workflow Models

Processes in an organisation require to be constantly reconsidered and optimised to meet the market's claims, as well as to fit new requirements in changing environment, like availability of resources etc. Workflow technology facilitates the modelling, redesign and administration of processes in an organisation.

Georgakopoulos et al. define workflow as "a collection of tasks organized to accomplish some business process" and the definition of "the order of task invocation or condition(s) under which tasks must be invoked, task synchronization, and information flow (data flow)"[5]. According to this, business processes can be described by specifying workflows. Business processes can be implemented as material processes (mainly production processes focussing on the manipulation of physical objects) or information processes (partly or fully automated transaction processes). One of the main reasons for using workflow technology in organisations is to understand business activities and thus have a means for improving customer satisfaction, increasing efficiency, and reducing costs.

Yet, it is necessary to periodically reconsider the business activities by so-called business process engineering (BPR) to fit new requirements. BPR addresses issues of customer satisfaction. It is complemented by information process reengineering (IPR) which addresses system efficiency and costs and describes the process requirements for information system functionality and human skills [5]. Conversely to the periodical reconsideration through business process reengineering, a continuous process examination, known as continuous process improvement (CPI) becomes more and more important (see [1]). As we see in the next section, workflow adaptation while the workflow is running comes with a number of difficulties.

Workflows are commonly classified in three categories: (I) *ad-hoc workflows*, with low complexity, few participants and short-term activities, (II) *administrative workflows*, with repetitive and predictable processes where the coordination of tasks may be automated, and (III) *production workflows*, which typically have a high complexity and the processes are, like in administrative workflows, repetitive and predictable (cf. [5,10,1]).

In the following, the definitions are given according to the Workflow Management Coalition (WfMC), an organisation of practitioners as well as researchers, who have provided a glossary of standardised terms of workflow technology, to have a more precise understanding of what workflow is [14]:

> *Workflow*: The automation of a business process in whole or part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules.
>
> *Business Process*: A set of one or more linked procedures or activities which collectively realise a business objective or policy goal, normally within the

context of an organisational structure defining functional roles and relationships.

*Process Definition*: The representation of a business process in a form which supports automated manipulation, such as modelling, or enactment by a workflow management system. The process definition consists of a network of activities and their relationships, criteria to indicate the start and termination of the process, and information about the individual activities, such as participants, associated IT applications and data, etc.

*Workflow Management System (WFMS)*: A system that defines, creates and manages the execution of workflows through the use of software, running on one or more workflow engines, which is able to interpret the process definition, interact with workflow participants and, where required, invoke the use of IT tools.

Workflows represent business processes. Business processes are modelled by process definitions and executed/interpreted by a workflow management system.

As we have seen, workflow management deals with coordination as well as execution. Buhler and Vidal [1] express the idea of workflow in the aphorism workflow = activities + processes, in analogy to the view on software programs as application = calculation + coordination. Here we see activities as the de facto executable components (called coordinables in [1]) while a process (coordinator in [1]) comprises the structuring of the activities, i.e. the activities' coordination.

Buhler's and Vidal's idea of introducing flexibility in workflows is based on web services and agents. Web services are components that execute a task and deliver results. Agents are used to coordinate the provided results of web services according to a certain goal. Buhler and Vidal speak of adaptive workflow engines = web services + agents, in analogy to the previously given equations.

### 2.3. Business Processes Modelled as Tasks

As stated in [12], workflow models and task models address the same domain, namely, how can tasks and activities be coordinated in such a way that their execution accomplishes the business goals. The difference between these two means lies in the different levels. Workflow models mainly focus on collaborative work, while task models primarily represent the individual task execution [12]. In the following, we are using the principles of task modelling to model group activities in a more flexible way by introducing distributable sub-tasks.

In our approach we call the parts, into which a business process is structured, *tasks*. Such tasks are assigned to groups or single persons for execution. We call the assignment of a task *order*, following the notions from business perspective. According to [3], we can distinguish tasks and orders in the way, that tasks are interpreted subjectively, while an order necessarily has objective characteristics. Thus, when an order is given to a group or person, it has to be transformed into a task. Fig. 2 illustrates this relation between the notions "task" and "order".

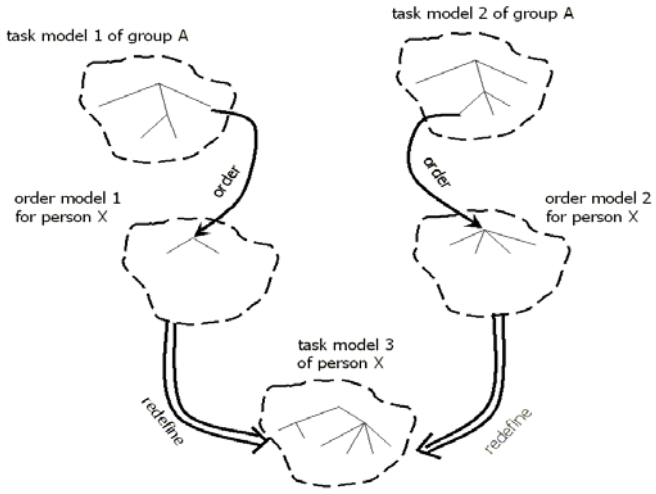


**Fig. 2.** Relation between task models and order models (according to [3]).

Group A planned two task models for different tasks and orders a certain person X (possibly a member of another group) with both tasks. Person X now has a set of tasks to do and has to compose his own task model from these two orders. This means, person X has to transfer the given (objective) orders into an own (subjective) task model.

In this transferring step, person X can make certain adaptations in the allowed range of the predefined structures of the orders. The following section gives a brief overview of different aspects of adaptation in connection with workflows.

## 3. Aspects of Adaptation

When speaking of flexibility in workflows, one can imagine several aspects of change. Van der Aalst et al. [13] made a comprehensive classification of these changes, of which we present an overview in this section.

Process definitions—our workflow specifications—are an aggregation of cases, i.e. runs through processes. Thus, a process definition is an abstraction of a concrete workflow, sometimes also called workflow schema. From this process definition, instances are created for the enactment in a WFMS. So the possible instances (or runs) can be seen as cases and the process definition comprises the description of a number of cases. Similarly, a task model comprises a set of different runs according to the task model definition.

Based on this idea, we can distinguish between two main types of change [13]:

- **ad-hoc changes,** where a change only inflicts one instance. Such a change might occur on exceptions or errors, or maybe special demands. In this case the workflow

description stays untouched. For ad-hoc changes, it has to be checked what kinds of changes are allowed at all. It is possible to allow changes at any time, so-called changes *on-the-fly*, or to restrict changes in an instance just when it starts (*entry time*) and then no more.

- **structural changes,** where the workflow description itself is changed and is thus affecting all new instances. This, of course, involves some conflicts like: What happens with already started tasks?, or: Is the old running workflow still consistent with the new definition? In [13] the following, three strategies for structural changes are distinguished: *restart* all running instances, or *proceed* the existing instances according to the old definition and start new instances according to the new one, or finally *transfer* the existing instances to satisfy the new definition.

In [13] the main kinds of changes in a workflow, no matter if structural or ad-hoc, are classified as follows:

1. **Extending tasks:** A new task is inserted in a sequence, or added as being processed parallel to existing, or added as an alternative of an existing task.
2. **Replacing tasks:** An existing task is replaced with a new one.
3. **Reordering tasks:** The order of execution of existing tasks is changed.

Besides these three kinds of changes we introduce some additional kinds of change, that affect the set of possible instances of a workflow model:

4. **Selecting tasks:** Alternative and optional tasks, as defined in the task definition, can be constrained, thus the degrees of freedom, the set of possible runs, can be reduced. This means, an option may be made obligatory, or alternatives may be removed. This kind of change may be done before the actual execution and renders the task definition more precisely.
5. **Constraining:** The existing structure of task execution is further constrained by additional global rules, which means rules that may be defined over tasks in any layers of the task tree. Thus, the set of possible runs through the model is being reduced.

The latter two types of change lead us to some concrete adaptation approaches as explained in the context of different aspects of change. According to [13], the aspects of changes cover the following branches:

- **control perspective**: covers the allocation and introduction of new resources to processes.
- **system perspective**: covers the infrastructure and configuration of the WFMS.
- **task perspective**: covering the adaptation of the set of possible runs.
- **resource perspective**: Resources may influence the order, respectively the choice of tasks in a business process.
- **process perspective**: covers the adaptation of the process definition.

We understand the task perspective as a reduction of degrees of freedom in the definition of a task., mainly using the idea of *constraining* the structure (see the fifth head point of kinds of change above). This can be done by introducing additional rules (temporal equations) besides the rules for each node. These additional rules create relations using any activities, not just those of a sub-tree. This idea is already presented in [2] and illustrated there by an example.

As regards the resource perspective, exhausted resources can constrain the options and alternatives for certain tasks. We understand this perspective as a way of using resources as a means of control. Thus, assigning resources to tasks can be used as a control criteria for preferred choices and thus prioritise possible alternative task executions.

The process perspective covers the idea of extending tasks. During its execution a task is refined by adding new sub-tasks (*extending*) or determining alternatives and options (*selecting*) in the predefined structure. The selecting is done, before the execution starts. This will be the basis for our approach of Order & Supply as described in the next section.

## 4. Workflow Adaptation by "Order & Supply"

Since a business process cannot be completely modelled in all details in the planning phase, adaptation has to be done by different employees after the enactment of a model. An adaptation in our approach can lead to either extending a task by new sub-tasks, or making a choice for alternative or optional tasks.

Considering the execution of a complex business process, we follow the metaphor of "Order & Supply", which means, in cooperative work, an employee A wants the execution of a task done by another employee B, i.e. A *orders* B to perform the task.

Often, an order comes with some predefined task structure. We assume that tasks and orders resemble the same structural description (see also [3] for more detail). Thus, an order already might have defined some constraints for its execution (cf. **Fig. 1** above: when interpreting the sub-trees of collector and administrator as orders, then we see that their orders already have a predefined structure).

B has to redefine A's order to his own task. In this redefinition process, B can adapt the order according to the degrees of freedom that are allowed within the predefined structure. Additionally, B can order some tasks further to another employee C, who again may adapt this order to his task. Such ordering can be done recursively.

After having solved the ordered task, the employee returns his results to the employee who ordered, i.e. he supplies the results. Thus, when B completed the task, he gives the results back to employee A, thus B supplies results for A. The principle of order and supply is summarised in **Fig. 3**.

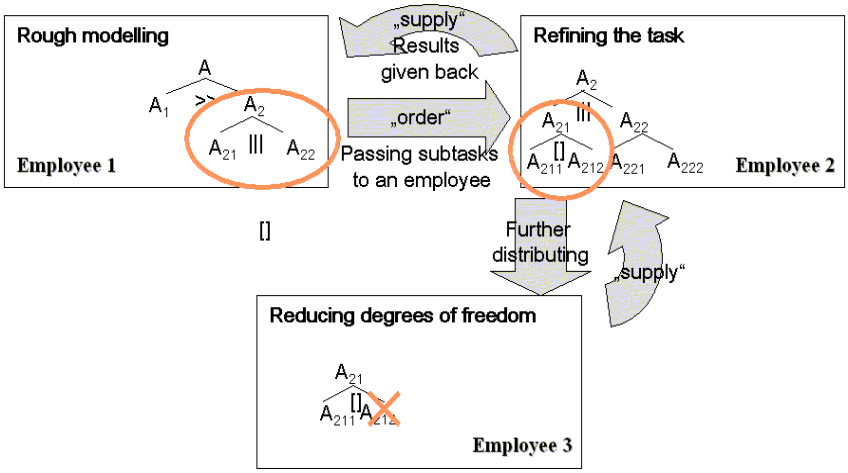


**Fig. 3.** The principle of Order and Supply.

Regarding our task model, an order corresponds to passing a sub-tree of the task tree to another employee. In the following, we consider business processes as being structured like tasks. We suggest a number of steps, how the above introduced Order & Supply principle can be realised.

Step 1. **coarse modelling**: Before a business process comes into enactment, it has to be modelled at least roughly to have a basis for the work. The main task has to be defined (this will be the root node of the corresponding task-tree) and the sub-tasks have to be determined. As described in [9] the task model is built in three phases: (i) hierarchical logical decomposition of the tasks forming a tree-like structure. (ii) identification of the temporal relationships between tasks. (iii) identification of the objects associated with each task. We neglect objects here and concentrate on the tree structure and temporal relations. After building a task model in such a way, we have a more or less coarse model.

Step 2. **distribution of tasks**: After the coarse model is built, it is being instantiated and the tasks are executed according to the defined rules. The execution of a business process, is planned by distributing it in parts which have to be performed by actors in certain roles. A role model maps the set of employees to the set of roles necessary for our business process. We call the distribution of a task to an employee *order*. When distributing an order, i.e. a sub-tree, the sub-tasks of this task may give a predefinition which can be adapted by the receiving employee as we see in the next step. Each employee has one or more tasks (task-trees) to process and each employee can further distribute parts of his task-tree(s) to other employees. The distribution should consider the workload of the employee for efficient and balanced processing. Hence, one can imagine monitoring the workload. Additionally, an employee should have the possibility to accept/deny a given order. An employee who receives a task as an order uses it as his view on the business process. All other tasks are

hidden and not accessible. So any adaptation does generally not influence other tasks in the business process.

Step 3. **adaptation of task**: When an employee receives an order, he is going to adapt it when necessary. On the one hand, the adaptation of a task can happen before starting to execute the task. This comprises appending new sub-tasks, thus refining and specifying the task in more detail (according to adaptation by *extending, reordering* or also *replacing* as described in the section above). In our approach, we neglect the adaptation by reordering and replacing, rather we presuppose an intention in the given task tree, that means the employee who gives the order has put his imagination into the model that he distributes. On the other hand the task can be adapted after the enactment of the model, i.e. while executing it. This means, alternatives are chosen and options are taken or rejected (according to adaptation by *selecting*, see above). It is, of course, also imaginable to select alternatives/options before starting the execution, for example if the employee has enough information to make such a decision. All adaptations made in this step are local and in the current instance only (cf. ad-hoc change, in the above section), so we avoid problems of inconsistency.

Step 4. **execution of task**: This step means de facto performing the task during the enactment of the model. The sub-tasks are executed according to the defined temporal equations. In this phase, *selecting* is still possible, although selecting during the execution means no adaptation, rather it characterises a concrete run. Only the leaves of the task tree are actual operations that are executed. Non-leaf-nodes just serve for structuring the task. When all leaves of a node are completed, the node itself is marked as complete as well.

Step 5. **returning results**: This is the *supply* phase of the process. After the employee has completed his task tree, the results are given back to the employee, who has ordered it. This is done recursively through the whole tree until all nodes (sub-tasks) are completed and the global goal of the task tree is achieved and the business process is finished. Mainly, the results consist of certain artefacts, documents or notifications (like acceptance or denial of requests).

These steps should illustrate, how to perform the whole or parts of a business process. Steps 1 and 2 are done at the beginning of processing a workflow. Steps 3, 4, and 5, as well as step 2, when further distributing, are then performed until the task is complete. The business process in a whole can be seen as one big and complex task model in the background which is processed and adapted continuously during runtime. The participating employees only see their view on parts of the business process. To describe the global task model of the business process, one can use XML descriptions, for instance as suggested by Stavness and Schneider[11].

In the next section we show, how this method can be put into practice by illustrating the principles at the example of maintaining a web glossary.

## 5. An Example: Maintenance of a Web Glossary

In this section, we illustrate the above described method of Order & Supply in a simple example. Lets consider the business process of maintaining a web-based glossary. This process can be classified as a certain kind of content management.

In our example, a research group is responsible for setting up and maintaining a web glossary. Necessary tasks are: adding new notions and definitions, editing existing notions like adding a figure or a reference, or removing terms from the glossary database. These tasks are done by the members of the research group. A first rough version of the task "maintain web glossary" might be modelled as shown in **Fig. 4**.

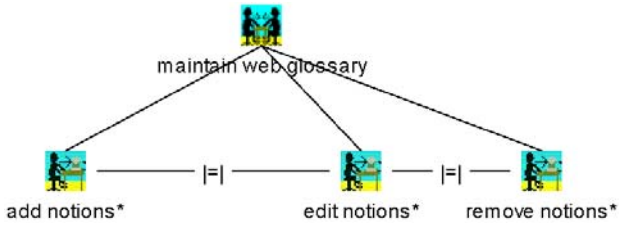


**Fig. 4.** Coarse model.

We can divide the maintenance in the way that each member of the group is responsible for a different subject, lets say one employee maintains notions from the area of object oriented technologies, another employee maintains notions in usability engineering, and a third employee is responsible for programming languages. In the following, lets concentrate on adding a notion to the glossary. **Fig. 5** shows, how a refinement of our first draft might look like and how we distribute tasks to employees, i.e. our experts in OO, Usability, respectively PL, thus realising ordering.

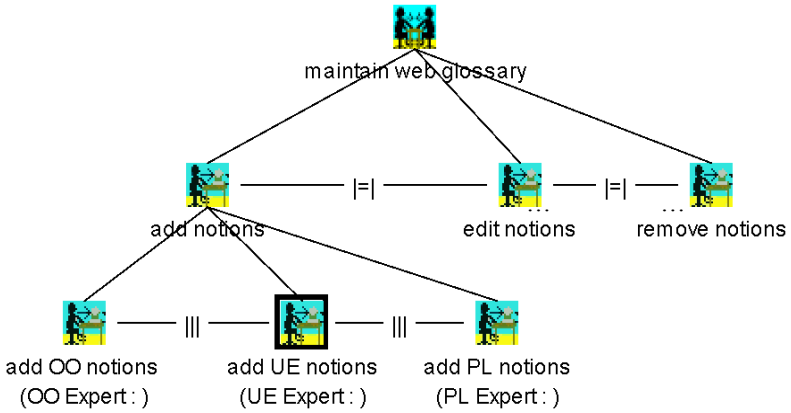


**Fig. 5.** Distributing sub-tasks.

Lets take a look at the activities of the OO expert. As we explained in the section before, the employees can adapt their tasks before they are executing them as well as during execution. Adding notions to a glossary might be structured by predefinition and could be as illustrated in **Fig. 6**. Hence, a definition needs the definition text, and definition reference, while figures and links are optional.

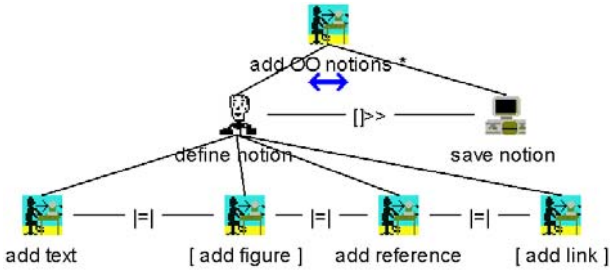


**Fig. 6.** Predefined sub-tree distributed to an employee.

If an employee is adding a notion, he has certain degrees of freedom. He might give an own definition text or do a research about the notion and referencing to the source (alternatives). He might add a figure to his definition text or not (option). The Employee adapts his task tree by adding further tasks and making a decision about optional tasks. **Figure 7** illustrates possible points of adaptation.

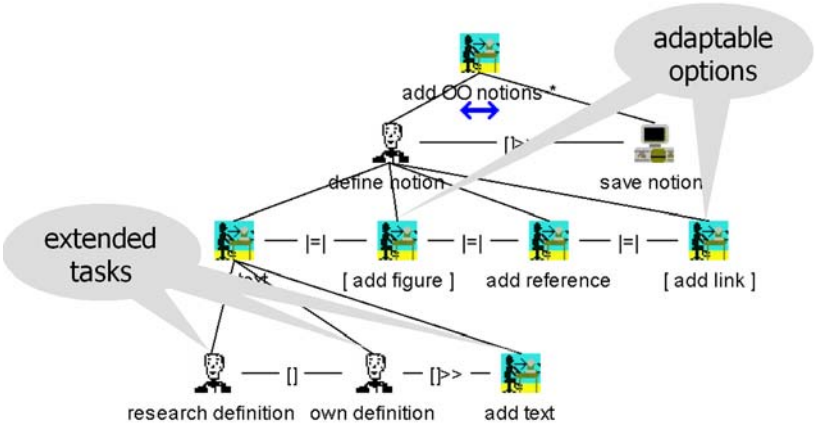


**Fig. 7.** Adaptation possibilities

Our employee decided to research a definition. Also, he is not adding a figure nor a link to his description. The adapted task tree of our software expert might look like in **Fig. 8**.

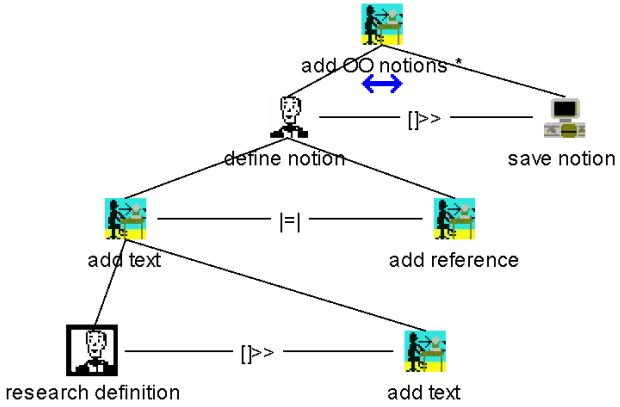


**Fig. 8.** Task tree after adaptation.

In this example, we have illustrated adaptation before execution starts. The employee can as well make decisions during performing his task. For instance, he might decide to delegate the sub-task "save notion" to an assistant who just inserts all collected information into the system.

We have modelled the diagrams in CTTE, an environment to model tasks according to [9]. Although the environment does not allow adaptation as we described above (except deciding for options or between alternatives), nor does it support distribution of subtasks, it serves as a good means of visualising task-trees in cooperative work.

## 6. Related Works

As shown in the introduction, keeping workflows adaptable is an important research area. Various techniques and approaches for dealing with adaptability in workflows can be found in the literature. Van der Aalst et al. implement dynamic change in business processes by using petri nets [13]. Odgers and Thompson consider aspect-oriented process engineering, combining techniques from the aspect-oriented programming with business process management [6]. Edmond and ter Hofstede use reflection and meta-object protocols [4]. They introduce task meta-objects for appropriate abstraction, thus allowing reasonable adaptation of a process' structure.

Furthermore, the idea of using Agents and Web Services for realising adaptation in workflows as described by Buhler and Vidal [1] is a promising topic for further enquiry. In a more general view, the subject of adaptive workflows can be seen as a new paradigm in software engineering, in terms of the new view described in [1]. This subject transcends to the area of structure dynamic systems and self organization from general systems theory.

## 7. Conclusions

We have seen that task models are an appropriate way of describing workflows, at least covering the group-level-oriented workflows. It comprises main aspects of workflow modelling. Using task models for describing workflows opens new ways of dealing with adaptation as we tried to show by examining the process perspective with our "Order & Supply" principle. This principle resembles the delegation in object-oriented technologies from a technical point of view. From the business perspective, "ordering" means to distribute tasks to different institutions. This may become clearer especially when tasks are distributed across a company's borders. In this context, the results of a solved *order* are *supplied* to the ordering *customer*.

We can distinguish adaptation before and while performing a task, e.g. Certain temporal relations, like option and choice allow to be processed before runtime as well as during runtime. We speak of adaptation of the workflow definition when options and alternatives are constrained before execution.

All adaptations we considered, only concern a reduction of degrees of freedom or extending tasks in a closed sub-tree. We did not inquire complete structure changes in processes. The general problem of adaptation in systems can be identified as structure-dynamic systems, a challenging area and large application field not only in the ambit of workflow modelling.

## References

1. Buhler, P. A., Vidal, J. M.: Towards Adaptive Workflow Enactment Using Multiagent Systems. In Information Technology and Management Journal, 2003.
2. Dittmar, A., More Precise Descriptions of Temporal Relations within Task Models. in P. Palanque and F. Paternò (eds.), Interactive Systems: Design, Specification, Verification; LNCS 1946, pp. 151–168, Springer 2000.
3. Dittmar, A., Ein formales Metamodell für den aufgabenbasierten Entwurf interaktiver Systeme. PhD Thesis, University of Rostock, 2002.
4. Edmond, D., ter Hofstede, A. H. M.: Achieving Workflow Adaptability by Means of Reflection. In Proceedings of CSCW-98 Workshop Towards Adaptive Workflow Systems, Seattle, USA, 1998.
5. Georgakopoulos, D., Hornick, M., Sheth, A.: An Overview of Workflow Management: From Process Modeling to Workflow Automation Infrastructure. In Distributed and Parallel Databases, vol. 3, No. 2, pp. 119–153, 1995.
6. Odgers, B., Thompson, S. G.: Aspect-oriented Process Engineering (ASOPE), Workshop on AOP at European Conference on Object-oriented Programming, Lisbon, Portugal, 1999.
7. Paterno, F.: Task Models in Interactive Software Systems. In S. K. Chang (ed.), Handbook of Software Engineering & Knowledge Engineering, World Scientific Publishing, 2001.
8. Paterno, F.: Model-Based Design and Evaluation of Interactive Applications. Springer, 2000.
9. Paternò, F., Mancini, C., Meniconi, S.: ConcurTaskTrees: A Diagrammatic Notation for Specifying Task Models. In Human Computer Interaction – INTERACT'97, pp. 362–369, 1997.
10. Plesums, Ch.: An Introduction to Workflow. Workflow Handbook 2002, Workflow Management Coalition, 2002.

11. Stavness, N., Schneider, K.: Supporting Workflow in User Interface Description Languages. Workshop on Developing User Interface Description Languages, AVI2004, Gallipoli, Italy, 2004.
12. Trætteberg, H.: Modeling Work: Workflow and Task Modeling. In J. Vanderdonckt and A. Puerta (eds.), Computer-Aided Design of User Interfaces II (CADUI); Louvain-la-Neuve, Belgium, Kluwer, 1999.
13. van der Aalst, W. P. M., Basten, T., Verbeek, H. M. W., Verkoulen, P. A. C., Voorhoeve, M.: Adaptive Workflow — On the interplay between flexibility and support. In J. Filipe and J. Cordeiro (eds.), Proceedings of the first International Conference on Enterprise Information Systems, vol. 2, pp. 353–360, Setúbal Portugal, March 1999.
14. Workflow Management Coalition: Terminology & Glossary, Document Number TC-1011, 3rd version, http://www.wfmc.org/standards/docs/TC-1011_term_glossary_v3.pdf.

## Discussion

[Tom Ormerod] I am interested in your claim that your adaptations can be made at the local level without running into dependency problems. For example, if I was teaching RE and someone made a change to the OO course, this would have implications. So how can local effects be accounted for?

> [Carsten Eicholz] If there is such an influence, it must be modelled explicitly, at a higher level of abstraction. A dependency would mean that we could not, in the example, have paralellism, since paralellism means that there is no dependency.

[Tom Ormerod] I'm wondering how you could slice that in a way that you can guarantee that there are no dependencies.

> [Carsten Eicholz] It depends on the expectations that you have of the model. In our study we have modeled complex independence. When there is a dependency, you cannot slice things in this way. Perhaps you could have a single lecturer who is responsible for both lectures.

[Simone Barbosa] How do you deal with an order that cancels another order that was partially executed? Would you then need to model all the other partially executed tasks?

> [Carsten Eicholz] There is nothing in our model to explicitly handle this. Perhaps one would need to specify each "canceling" workflow separately and have it selected if needed.

[Simone Barbosa] So you would have to model these as separate independent workflows?

> [Carsten Eicholz] Yes, we would need a new workflow model to do that.

[Michael Harrison] The reason for modeling workflow is so you can ask questions of the workflow. E.g. an auditor would want to know who signs off on purchases. Have you thought about how you would inspect workflows.

> [Carsten Eicholz] No, we have a straight-forward approach where the absract modelling is only done at the beginning. We don't save all of the adaptations.

We have the idea of saving such a library, where we save and preserve all these tasks for analysis, to see what can be optimized. But this is not currently included.

[Juergen Ziegler] How do you model splits and joins in this model.

[Carsten Eicholz] The splits should be clear--parallel execution. A join--in what case do we have a join?

[Juergen Ziegler] In some processes you have joins, e.g. building a car you have separate processes that have to come together.

[Carsten Eicholz] Our approach is completely different from net-based approach that is common in process modelling. We are hierarchical. So a join must be represented as the super-task of two sub-tasks. It cannot be visualized by a join as in an activity diagram.