A Method for Modeling Interactions on Task Representations in Business Task Management Systems

Todor Stoitsev, Stefan Scheidl

SAP AG, SAP Research, Germany,
{todor.stoitsev, stefan.scheidl}@sap.com

Abstract. Task modeling approaches facilitate the design of interactive systems by bridging the gap from understanding human tasks to designing interfaces to support these tasks. Business Task Management (BTM) systems provide explicit task representations for managing and coordinating work items, by further requiring definition of how such task representations can be created, distributed and monitored throughout an organization. This paper presents a method for modeling interactions on task representations in BTM systems. It introduces generic task-centric roles as useful abstractions, encapsulating different perspectives on tasks and related interactions. This allows generic, domain-independent views on tasks resulting in enhanced adaptability of BTM systems in different application contexts. The method is implemented in the Collaborative Task Manager (CTM) tool.

Keywords: Task management, interactions modeling, end user development

1. Introduction

The need to develop adaptable software applications which can be swiftly tailored to specific end user needs and application domains has resulted in flexible, model-driven software engineering approaches. Task modelling approaches have proven highly efficient for designing interactive applications. CTT [16] enable system designers to describe the logical activities that an interactive application should support and facilitate model-driven software engineering from requirements analysis to user interface design. Further approaches like GOMS [8] take a goal-oriented view and provide comprehensive description of activity sequences and tasks' interrelations. GTA [21] combines task analysis methods from human computer interaction with ethnographic methods as used in computer supported cooperative work and provides comprehensive methodology for the design of groupware systems. While the above approaches focus on describing user activities and the interactions needed to support them, they do not consider interactions on explicit task representations in formal systems. Such representations are used in Business Task Management (BTM) systems to manage and coordinate work items, and can be e.g. to-do items in personal task lists or in a central work list of a company department. Modelling of interactions on task representations can bring flexibility to BTM systems and make them adaptable to different business domains and application contexts. This requires abstractions of the possible interactions from personal and from organizational point of view, which can allow clustering of requirements and detection of generic interaction patterns on task representations.

Molina et al. consider deficiencies in known approaches for modelling collaborative aspects of human work and propose pattern-based techniques for designing groupware applications [13]. However, their methodological framework starts with modelling of the given organizational structure, which binds the approach to a given enterprise and business domain. Organizational patterns for early requirements analysis are discussed in [9]. These patterns are however elicited based on case studies in concrete enterprises and do not provide a high-level generalization of organizational roles and basic interactions, needed to support collaborative work.

This paper presents a generic modelling approach, which is not confined to a given business domain or concrete organizational structure. The approach provides abstractions for defining high level interactions on explicit task representations in BTM systems based on generic, task-centric roles. This enables flexible adaptation of the BTM system in different usage contexts. Domain-specific extensions are enabled through mapping of the task-centric roles to organizational roles. The presented approach is implemented through the Collaborative Task Management (CTM) prototype.

The remainder of the paper is organized as follows. In section 2 we discuss the background of the presented modelling method. The method is described in section 3. In section 4 we present the implementation of the method in the CTM prototype. In section 5 we give conclusions and future research directions.

2. Background

The presented work founds on empirical research based on site visits and interviews at three companies from different industries: textile (120 employees), software (ca. 500 employees), automotive (ca. **150** employees) and is consolidated with extensive literature research. As part of our activities we investigated end user interactions with software systems within the context of day-to-day work. Our purpose was to reveal basic user demands and pain-points in the area of task management and software support for agile business processes.

There is plenty evidence that some aspects of human work are similar in different business domains, organizational or individual day-to-day activities. Reappearing interaction schemes are discussed in related literature in the field of interactive systems design, i.e. as task patterns, providing reusable structures for task models [6, 14, 15]. The idea for reappearing interactions is in the background of the presented work. We suggest that as a first step towards user-centric task management observations it is essential to determine basic user segmentation. It should provide high level abstractions for different user activity types and thereby also basic directions for the detection of significantly different interaction schemes on task representations.

There is a common notion of basic user activity types. These are divided into strategic, tactical and operational and are especially used in decision-making and planning studies [5, 17]. Further studies use this segmentation to reduce the complexity for system observations from significantly different perspectives [23]. An extended view, focusing on intellectual capital governance is presented in [22], where additionally ‘Global/Societal’ and ‘Implementation/Application’ perspectives are introduced. We suggest that this segmentation can be considered also in task management context. For our studies we used the following basic activity types:

- **Strategic:** Refers to activities with high degree of unpredictability and strong innovation character, further involving extended collaboration and people management. Example activity – strategy planning.
- **Tactical:** Refers to activities with higher need for flexibility and rapid adaptation. Such activities often imply ‘on demand’ innovation and creativity to increase efficiency, avoid bottlenecks and workaround unanticipated problems. Example activities – supply chain management, production planning.
- **Operational:** Refers to activities with higher degree of predictability and repeatability. Such activities mostly consist of routine tasks. Example activities – support center activities, sales order processing.
- **Implementation/Application**: Refers to activities, where existing abstract knowledge is transformed to tangible implementations and processes. It is different from operational as there is a noticeable creative element. There is also a significant difference to the tactical level as the focus is set on the actual implementation and not on the overall planning. Example activity – software implementation, non-automated production/crafting.
- **Societal/Educational**: Refers to activities with strong societal character, where new knowledge is created, systematized and transferred. Example activity – teaching/training course preparation.

The presented activity types provide abstract categories for tasks of significantly different nature. Nevertheless, it should be considered that it is not always possible to match all day-to-day activities of a system user to only one activity type. For example a project manager, who is usually executing tactical activities, may have also operational tasks, deriving from common organizational practices like e.g. performance feedback or quarterly reports.

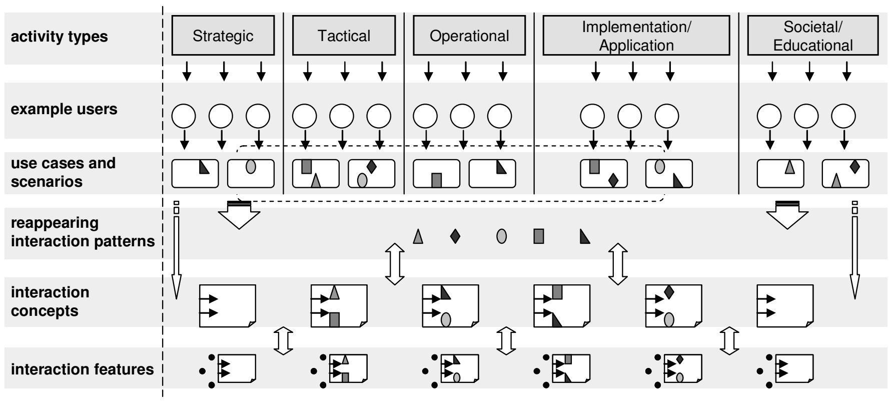


Fig. 1. Example users (circles on second layer from top) with different activity types (top layer) are examined in concrete use cases and scenarios. End-to-end scenarios (dotted line on use cases and scenarios level), comprising the activities of various users with different activity types reveal, how the system should mediate between users with different business roles. Various reappearing interaction patterns (triangles, circles, diamonds and squares) are detected and extracted from the different use cases and scenarios. The patterns are linked to generic interaction concepts (rectangles containing arrows and interaction pattern symbols). Some concepts are directly extracted from the use cases and scenarios (arrows on the right and on the left and nearest empty concept rectangles containing only arrows). Concrete interaction features (black dots beside interaction concepts' symbols on lowest level) for supporting given interaction concepts are identified where possible.

An overview of the background information that we collected for the elaboration of the presented method is shown in Figure 1. The three top level layers build the foundation for a business centric top-down view on a task management system. We examined the different activity types described above based on identified example users from different business domains. These were 10 employees from the textile production company, 9 employees from the software company and 7 from the automotive company. In the given overview on Figure 1 a possibly wide activity type scoping is presented for completeness. We however did not explore the societal/educational level as our focus was on business users which were not involved in any educational activities. The involved users in each company covered all other activity types. The users' organizational roles in the textile company ranged for example from brand manager (strategic/tactical) and chief sales officer (tactical) to sales officers (tactical/operational) and IT employees (implementation/application). The users were put in the context of scenarios and use cases, which revealed their work practices and helped to identify their demands and pain-points regarding task management. We elaborated 3 mainstream scenarios in each company, detailed through 2 to 3 further supportive scenarios, i.e. for special case handling or peripheral activities. For the textile company mainstream scenarios were the initiation of special sales procedures, e.g. consignment and annual discount sales, and the binding of new partner enterprises for electronic data interchange. For the software company these were the preparation of new product package, new software release and support center scenarios. In the automotive company we explored prototype development and mature prototypes' transfers from prototyping to manufacturing. The user studies were conducted using contextual enquiry techniques [2] at the site of the respective company in the familiar work place surrounding of the interviewees to preserve their context as far as possible. The interviews were recorded using a digital voice recorder and transcribed for analysis. A thorough examination of the elaborated scenarios and use cases led to the identification of reappearing interaction patterns with a software system related to task management. Such were e.g. the creation of calendar entries with reminders, the sending of meeting requests or the management of to-do items in Microsoft Outlook task lists. These patterns reappeared in cross-functional areas or sometimes repeated in different scenarios for the same user type.

The two low level layers on Figure 1 constitute the bottom-up view towards a task management system. They provide task management concepts and features which can support a system implementation. As the concepts provide lower granularity, a concept can be relevant for more than one of the reappearing interaction patterns. In some cases concrete features could be mapped to the detected concepts. For example, a concept 'Awareness', referring to the ability to keep the user informed of possibly approaching bottlenecks or escalations, can be supported by a feature, displaying warning dialogs for approaching task deadlines.

We ordered the elaborated content in a semantic mediawiki which provided a highly interlinked structure with certain evaluation mechanisms like e.g. querying the (number of) occurrences of interaction patterns and related concepts in the various scenarios and use cases. This helped to evaluate the resulting requirements towards the interactions in real-life context. Based on these observations we were able to clearly identify several common perspectives on task representations. These were associated with the attitude of the persona towards a task which was always noticeable in the background of the interactions. These perspectives were able to describe the complete end user interaction landscape on explicit task representations in all scenarios. We extracted them as generic task roles.

3. Task Roles

Task roles aim at providing generalizations that group certain interactions with an explicit task representation and basically state the question: What interactions should be supported, when a user is acting in a given task role? Task roles hence provide task-centric perspectives that reveal different aspects of task management and enable domain-independent abstractions of the necessary interactions on task representations. The task roles are shown in Figure 2. The dotted line areas mark different aspects, which influenced the derivation of the roles. The Requester and the Recipient roles are related to the collaborative handling of a task. The right hand side area contains roles, for which relationships to the organizational structures and hence closer connections to the business context can be discovered. Thereby we suggest that collaborative aspects are orthogonal to organizational aspects as collaboration is performed throughout the complete organizational structure. It is hence reasonable to emphasize on these aspects through explicit roles. A brief description of the derived task roles is given in the following, where the term 'agent' is used to identify the role owner. An agent is generally a system user, but it can also be a software component, which is able to create or process tasks based on given rules.

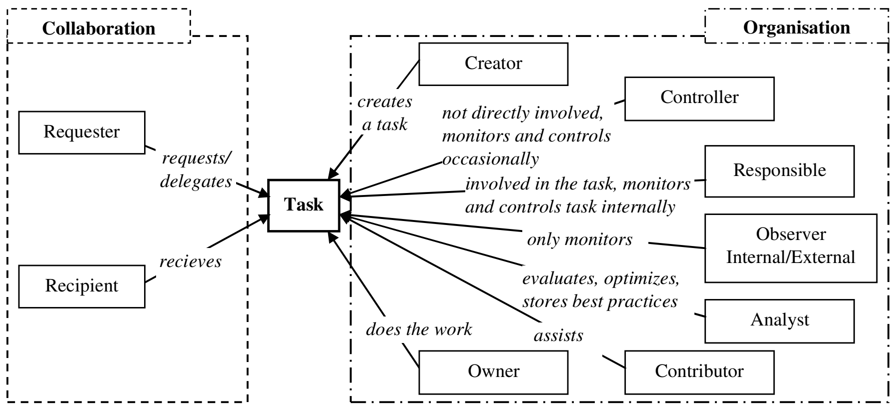

Collaboration
Organisation
Creator
creates a task
Controller
not directly involved, monitors and controls occasionally
Requester
requests/ delegates
Responsible
involved in the task, monitors and controls task internally
Task
only monitors
Observer Internal/External
recieves
Recipient
evaluates, optimizes, stores best practices
Analyst
does the work
assists
Owner
Contributor

Fig. 2. Task roles.

- **Creator** – An agent which creates a task. We here only refer to the action of creating a task, excluding its further processing.
- **Owner** – An agent which is executing on a task and should deliver the results.
- **Controller** – An agent which is able to monitor and interfere in the task activities, and which is not being directly involved in the task. Such could be e.g. a senior manager, not directly contributing or performing on the task but who is able to occasionally view the task and eventually trigger escalations.
- **Responsible** – An agent which is in charge of the successful completion and coordination of a task. Such can be e.g. a project manager responsible for sub tasks distributed in the project team. Thereby a responsible has lower level expertise on a task than a controller.
- **Observer**
  - **Internal** – An agent which can view the evolution of others' tasks without interfering. Such would be e.g. team members, who can view each other's tasks or various shared tasks.
  - **External** – An agent which does not belong to a company or team but is able to view given company's or team's tasks externally. Such is e.g. a customer, who is able to track the processing of his order.
- **Analyst** – An agent which evaluates the outcomes of a task, considers optimizations, and saves reusable data as best-practices or recommendations. This role is clearly related to knowledge management functions.
- **Contributor** – An agent which is informally connected to a task, without being involved in it. The contributor is occasionally delivering information or resources to a task, without being responsible for it, executing on it or even having access to the complete task contextual information.
- **Requester** – An agent which delegates a task to another party.

- **Recipient** – An agent which receives a task from another party.

A similar representation of the task roles is given also in [7]. The Owner and Recipient roles are discussed also in [19]. The next section presents how task roles are used for interactions modeling in the CTM system.

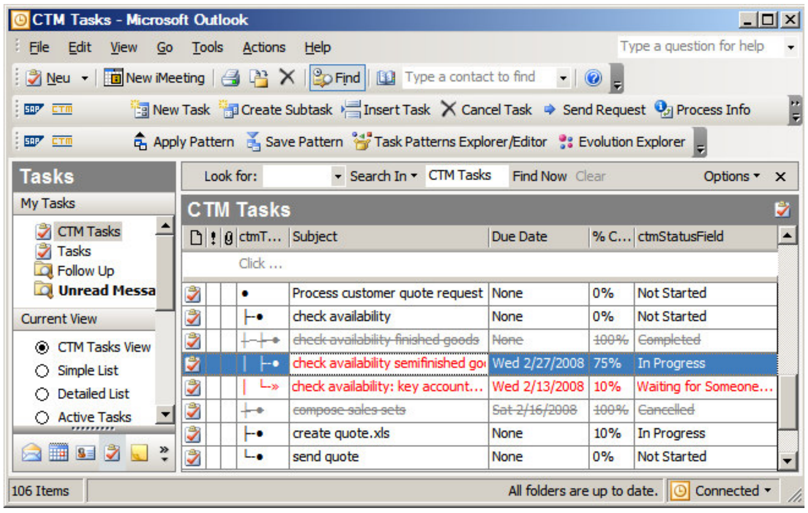
4. Modeling Interactions with Task Roles

We have realized the task roles as extensions to the task model used in the Collaborative Task Manager (CTM) in order to enable enhanced adaptability of the system in different application contexts. CTM is a process-enhanced groupware system which provides advanced End User Development (EUD) capabilities and aims at enabling users with different IT and business background to efficiently participate in business process composition and management. EUD is defined as "a set of methods, techniques, and tools that allow users of software systems, who are acting as non-professional software developers, at some point to create, modify, or extend a software artefact" [11]. In CTM, a process model is considered as a software artifact, which can be adapted and enacted to support human-centric business processes.

4.1 The CTM Prototype

This section gives a general overview of the CTM functionalities with respect to the presented method for interactions modeling. A more comprehensive description of the CTM prototype is given in [20]. CTM generally involves end users in process composition by providing added value on personal task management and leveraging their experience with standard tools for task management (to-do lists) and collaboration (email) towards definition of process models. The solution provides a "gentle slope of complexity" [12] for process tailoring by closely integrating the process definition in the actual user working environment and unfolding emergent processes behind the scenes in an unobtrusive manner. For achieving this it uses enterprise-wide "programming by example" [10] by implicitly reconciling data on personal task management of multiple process participants to end-to-end process execution examples.

In order to ensure integrated support in a common user working environment, the CTM front-end is designed as a Microsoft Outlook (OL) add-in. CTM extends OL mail and task items and enables "programming by example" by capturing OL events and using web services to replicate task data in a tracking repository, residing in a Database (DB) on the CTM server. The CTM To-Do List (TDL) is shown in Figure 3. Extensions to the standard OL tasks enable end users to create hierarchical to-do lists. When the end user is creating or editing a CTM task they work with the familiar OL task fields. Files can be added to CTM tasks as common OL attachments. An email can be as well saved as a CTM task, whereby the email subject, text and attachments are transferred to the resulting task.



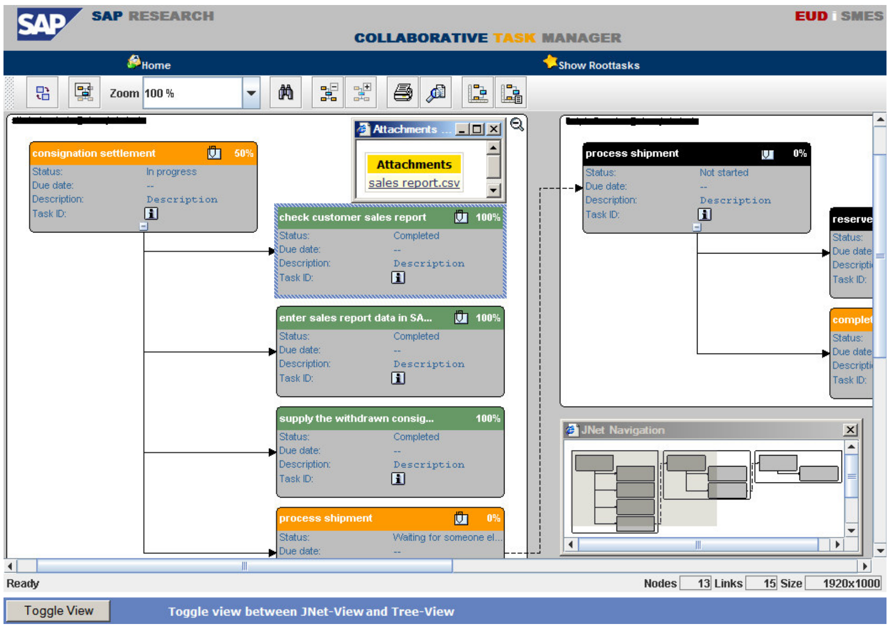
CTM Tasks

ctmT…	Subject	Due Date	% C…	ctmStatusField
Click …				
•	Process customer quote request	None	0%	Not Started
├•	check availability	None	0%	Not Started
├├•	~~check availability finished goods~~	~~None~~	~~100%~~	~~Completed~~
│ ├•	check availability semifinished go	Wed 2/27/2008	75%	In Progress
│ └»	check availability: key account…	Wed 2/13/2008	10%	Waiting for Someone…
├•	~~compose sales sets~~	~~Sat 2/16/2008~~	~~100%~~	~~Cancelled~~
├•	create quote.xls	None	10%	In Progress
└•	send quote	None	0%	Not Started

Fig. 3. CTM To-Do List (TDL).

Tasks can be delegated over email, whereby the recipients can further break down the received tasks and delegate resulting (sub)tasks to other end-users. A CTM task is delegated through a preformatted “Request” message, which recipients can “Accept”, “Decline” (similarly to meeting requests in OL) or “Negotiate”. The latter action enables iterative negotiations for additional clarifications on tasks. When a request is accepted, and later on completed by a recipient, the latter issues a “Declare Complete” message. Hereupon the requester can respond with “Approve Completion” or “Decline Completion” message. These actions allow negotiation of deliverables before the final completion of a delegated task. The actual discourse takes place in the email text, independently from the given message type. This allows open-ended collaboration and prevents from submitting user behavior to strict speech-act rules, which is a known limitation in speech-acts adoption [3].

Tracking of email exchange for task delegation integrates the personal to-do lists of different process participants to overall Task Delegation Graphs (TDG) [19] on the server. TDGs can be inspected through a web front-end as shown on Figure 4. They represent weakly-structured process models which are captured as actual process execution examples and contain all task data including artifacts (attachments) and stakeholders’ information. Tasks of different users are contained in different user containers. TDGs provide a workflow-like overview of evolving user activities, aiming to facilitate “the creation of a shared understanding leading to new insights, new ideas, and new artifacts as a result of collaboration” [4]. In a TDG users can view status of related tasks, identify potential bottlenecks and evaluate work distribution, which is not possible by using common email and to-do items. Currently, due date, task processing status and percent complete indications are provided. Attachments, added in OL tasks, are replicated in a central DB-based Artefacts Repository (AR) on the CTM server, and are accessible in the task nodes.



[redacted]

consignation settlement 50%
Status: In progress
Due date: --
Description: Description
Task ID:

check customer sales report 100%
Status: Completed
Due date: --
Description: Description
Task ID:

enter sales report data in SA... 100%
Status: Completed
Due date: --
Description: Description
Task ID:

supply the withdrawn consig... 100%
Status: Completed
Due date: --
Description: Description
Task ID:

process shipment 0%
Status: Waiting for someone el...
Due date: --

[redacted]

process shipment 0%
Status: Not started
Due date: --
Description: Description
Task ID:

reserve
Status:
Due date
Descripti
Task ID:

complet
Status:
Due date
Descripti
Task ID:

Ready
Nodes 13 Links 15 Size 1920x1000

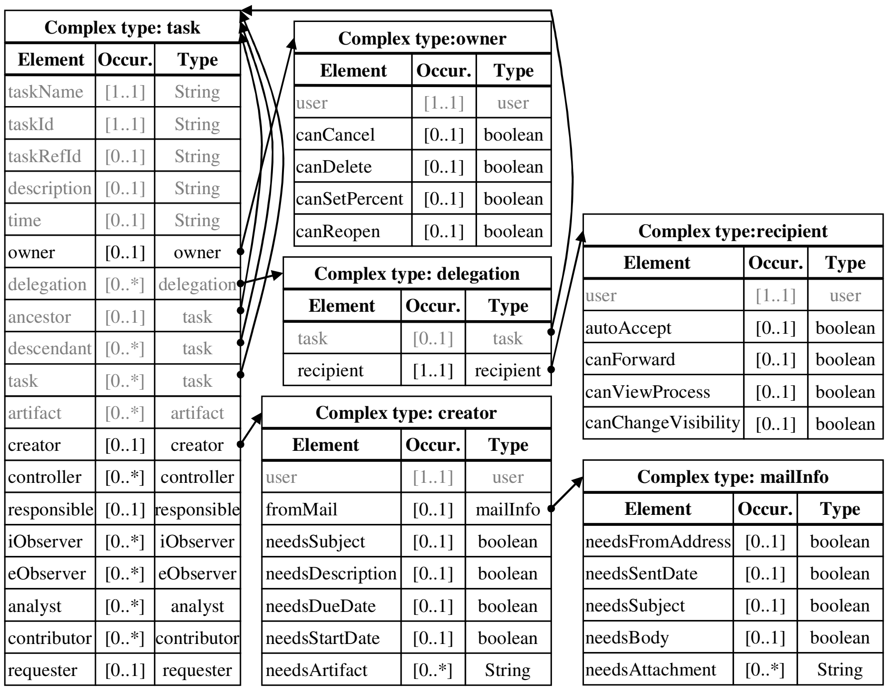
Fig. 4. Task Delegation Graph (TDG).

As end users have different levels of technical expertise and attitudes towards maintaining process data, we suggest that it is important to consider possibilities for “seeding, evolutionary growth, and reseeding (SER)” [4] of user-defined process models for their iterative refinement and complementation. CTM enables SER of weakly-structured process models through extraction, adaptation and reuse of Task Patterns (TP) [19]. We consider a TP as a reusable task structure, comprising one task with its sub task hierarchy and the complete context information of the contained tasks like e.g. description, used resources, involved persons etc. A TP hence represents a high level task as a step in an ad-hoc business process and corresponds to the notion introduced in [18]. In the literature ‘task patterns’ are discussed also regarding reusable structures for task models in the field of interactive systems design [6, 14, 15]. However, such observations focus on low-level interactive activities, like e.g. searching, browsing or providing generic system input, and deviate from the notion of TP that we use. In CTM, TPs can be enacted to create a new process instance and execute it along the provided example flow. This flow can be altered by changing suggested task recipients or reusing referenced TP hierarchies. Task evolution through TPs’ adaptation and reuse is traced through task instance-based ancestor/descendant relationships [19]. These enable end users to establish best-practices and to trace best-practice deviations in different application cases.

4.2 Modeling Interactions on CTM Tasks

CTM enables end users to implicitly develop end-to-end process execution examples as TDGs, which provide a shared context between all involved process participants. This results in a need for enhanced system adaptability due to the different requirements towards sharing and managing task content in different processes. Task roles are applied to the task model defined in [19] as optional XML elements in 'task' elements as shown on Figure 5. In CTM, task roles and the corresponding interaction properties are stored as OL task item properties and tracked with the other task data to the CTM server. Through this they define the interactions with CTM tasks throughout the system. The implemented task roles are discussed in the following.

Creator: This role is taken by the users while they are creating a CTM task and defines the interactions for entering the required task input. We suggest that modeling of task item creation is important as users often define tasks in an underspecified manner [1], which may lead to omission of important information and defer the task processing later on. For example, if the needsDescription element (see Figure 5) is omitted or set to false in the task model, this means that no description for the task may be specified. If this element exists and its value is true, the OL dialog for creating a new CTM task will not close until a description is specified. All other 'needs' elements function in an analogous manner. The needsArtifact element can occur multiple times and accepts a text command in Character Data (CDATA) form, specifying the artifacts (attachments) which need to be added to a task upon creation. These commands represent standard regular expressions for file names like e.g. '*.doc' or 'sales report.*'. For example, the processing of a weekly sales order settlement in the textile company (cf. section 2) is based completely on a customer sales report which is sent by the customer as a file in Comma Separated Values (CSV) format. It is necessary to ensure that the report will be attached in the initial process (root) task. This can be accomplished by adding a needsArtifact element with '*.csv' command text in the task model.



Complex type: task		
Element	**Occur.**	**Type**
taskName	[1..1]	String
taskId	[1..1]	String
taskRefId	[0..1]	String
description	[0..1]	String
time	[0..1]	String
owner	[0..1]	owner
delegation	[0..*]	delegation
ancestor	[0..1]	task
descendant	[0..*]	task
task	[0..*]	task
artifact	[0..*]	artifact
creator	[0..1]	creator
controller	[0..*]	controller
responsible	[0..1]	responsible
iObserver	[0..*]	iObserver
eObserver	[0..*]	eObserver
analyst	[0..*]	analyst
contributor	[0..*]	contributor
requester	[0..1]	requester

Complex type:owner		
Element	**Occur.**	**Type**
user	[1..1]	user
canCancel	[0..1]	boolean
canDelete	[0..1]	boolean
canSetPercent	[0..1]	boolean
canReopen	[0..1]	boolean

Complex type: delegation		
Element	**Occur.**	**Type**
task	[0..1]	task
recipient	[1..1]	recipient

Complex type: creator		
Element	**Occur.**	**Type**
user	[1..1]	user
fromMail	[0..1]	mailInfo
needsSubject	[0..1]	boolean
needsDescription	[0..1]	boolean
needsDueDate	[0..1]	boolean
needsStartDate	[0..1]	boolean
needsArtifact	[0..*]	String

Complex type:recipient		
Element	**Occur.**	**Type**
user	[1..1]	user
autoAccept	[0..1]	boolean
canForward	[0..1]	boolean
canViewProcess	[0..1]	boolean
canChangeVisibility	[0..1]	boolean

Complex type: mailInfo		
Element	**Occur.**	**Type**
needsFromAddress	[0..1]	boolean
needsSentDate	[0..1]	boolean
needsSubject	[0..1]	boolean
needsBody	[0..1]	boolean
needsAttachment	[0..*]	String

Fig. 5. Task model extensions with task roles - only the owner, recipient and creator roles are given for simplicity. The task role extensions embody the user information where each role is defined as a complex type, additionally allowing embedding of XML elements for specifying concrete interactions. Grayed-out elements are discussed in the original task model [19].

The *fromMail* element defines the applying of email contents to a CTM task. Several employees in the sales department of the textile company required to be able to create a CTM task from an email over a mouse click. CTM hence enables creation of a new (root) task from an email and application of email content to existing CTM tasks. The latter operation is especially relevant for tasks, resulting from TP reuse. When this operation is performed, all available tasks from the TDL are given in a tree view with check boxes where the user can select target task(s) for the email content. During this operation, the *mailInfo* properties take effect and define which data should be transferred from the email. If for example the *needsBody* element exists and is set to true, the email body will be transferred to the CTM task(s) as task description. The *needsAttachment* element functions analogously to the *needsArtifact* element, discussed above, and accepts a text command in CDATA form. This allows filtering of artifacts while transferring them from an email to a CTM task. For example, if a user has applied a TP for weekly settlement of a customer sales order, the CSV sales report file from the original execution (previous settlement) will be available in the CTM root task. By applying the contents of a new customer email with attached CSV sales report file to the reused root task and using the appropriate '*.csv' filtering command, the task content will be updated with the new sales report from the customer email, excluding any email attachments with other file extensions. The *needsFromAddress* and *needsSentDate* are true by default. They transfer the corresponding sender and sent-date information as OL task properties to activate a control for searching for the original email in the common OL mail folders.

Owner: This role is generally taken by the users when they have created (or, in case of delegation, received and accepted) a CTM task in the CTM TDL and have to process it. This role comprises all necessary interactions for processing a task in a BTM system. CTM supports modeling of several interactions on active task representations. It can be specified for example, if a user should be able to cancel an active task, to set the percent complete of tasks with sub tasks (alternative is to automatically increase percentage of a parent task based on sum of sub tasks' percentage), delete a task or reopen a completed task (cf. Figure 5). The set of supported actions can be limited due to limited functionality of a system or due to intentional limitations. For example, we set the delete capability to false during a case study at the textile production company to deactivate any delete controls on tasks and preserve the whole amount of generated ad-hoc tasks for evaluation. The reopen capability is still unsupported in CTM due to the required complex compensation handling in TDGs.

Recipient: This role is taken by a user, which receives a CTM task. If the *autoAccept* option (cf. Figure 5) is true, a requested task will be automatically inserted in the recipient's TDL. This option is helpful e.g. for work distribution in support center scenarios as we observed them at the software company (cf. section 2), where the number of error reports handled by each employee and in the whole support center are monitored and tasks are automatically delegated according to the work load. The *canForward* option is also applicable in such scenarios and specifies, if the recipient is allowed to forward a requested task to another person. The *canViewProcess* element defines if a task recipient should be able to view the overall TDG of a process, to which the received task belongs. Such viewing can be deactivated, e.g. if a person from a given department is asked to accomplish a task in the context of a confidential process, managed in another department and containing sensitive customer data. The *canChangeVisibility* option enables administration of the visibility of recipients' tasks in the TDG. If this option is true, the recipient will be allowed to specify if their user container will appear blank in the TDG (cf. Figure 4) or it will show their personal task hierarchy.

Requester: A user is taking this role, while they are delegating a task. An element *canSuggestPattern* enables a requester to add (attach) suggested TPs as recommendations for the further task processing in a CTM request message. A *canSetRecipientProperties* element defines if a requester should be able to customize the recipients' properties described above, i.e. to specify if the recipients are allowed to view the process tree and to change the visibility of the resulting accepted tasks' hierarchies. This is accomplished in a CTM property dialog, which can be displayed if the above property is set to 'true'.

Controller: In CTM controllers can be explicitly set in an additional dialog, which can be shown on CTM tasks in the TDL or during TP editing (explicit task modeling). Controllers are specified based on their user email address. When a user opens a TDG for an active process, they log in with their email address and the system determines the tasks, for which the user is specified as controller. These tasks receive further buttons in the web form, which allow controllers to add comments, increase priority or trigger task escalations (enabled/disabled accordingly through *canAddComment*, *canIncreasePriority* and *canTriggerEscalations* elements in the model).

Contributor: Contributors are explicitly set in tasks through additional property dialogs accessible from the TDL and during TP editing, and are as well specified through a user email address. When a user logs in to view a given TDG, the task nodes for which they are specified as contributors contain additional controls, which allow them to upload attachments in the artifacts list and to set comments on tasks (enabled through *canAddAttachments* and *canAddComments* elements in the model). Controls in task nodes for changing due dates, priority and triggering escalations will not be active as these are only relevant for Controllers.

Responsible: This role results from the task owner role after a task is delegated, and contains properties for automated notifications on task changes, like e.g. percent complete changes, structure changes (sub task creation), content changes (subject or description), due date changes, delegations. The relevant elements in the task model are respectively: *notifyOnPercentComplete*, *notifyOnSubjectChange*, *notifyOnDueDateChange*, *notifyOnBrakedown*. Further properties define if the responsible should be able to cancel, complete or delete a task (*canCancelFromAbove*, *canCompleteFromAbove*, *canDeleteFromAbove*), which results in cancellation, deletion or completion of a delegated task and the underlying task hierarchy.

Internal & External Observer (i/eObserver): These roles are set explicitly on tasks based on the user email address in the TDL or during TP editing. When the users log in for viewing a TDG, CTM determines their role and shows different view of tasks for the different roles. Internal observers may be allowed to view task descriptions, attachments or task delegation dialogs, i.e. through properties respectively *canViewDescription*, *canViewAttachments*, *canViewDialog*. External observers may be allowed to view only high level description of the processes where user containers are substituted with generalized containers for departments, which are for example processing the customer (external observer) order (if property *canViewPersonalizedTaskList* is false).

Analyst: This role comprises interactions for extracting reusable task and process knowledge. This includes e.g. interactions for viewing the task execution history (changes) and task evolution (ancestors/descendants), for extracting TPs and publishing them to central TP repositories. The relevant properties in the task model are: *canViewDialog*, *canViewExecutionHistory*, *canViewEvolutionHistory, canExtractTDG, canSaveGlobalTP*. The controls for the latter two operations are not enabled for other users, to avoid generation of multiple, concurrent best-practice definitions. This role basically targets at consolidation of the captured process experience.

4.3 Summary

In CTM some task roles (controller, contributor, internal and external observer) can be explicitly defined during task execution, while work is managed and user-defined task hierarchies evolve. This can be accomplished in property dialogs, where users can select the appropriate roles on tasks, assign them to different users based on email addresses and set the appropriate options for interactions in CTM in the scope of a given task role. This results in runtime task modeling of the interactions on the evolving weakly-structured processes. Other task roles are implicitly taken over by the system users, while they are creating CTM tasks (creator), managing them in their TDL (owner), delegating tasks (requester), receiving tasks (recipient), managing delegated tasks (responsible) or extracting best-practice definitions (analyst). The interactions, necessary to act in these roles and in the previously mentioned explicit roles, can be predefined in the task model of a TP. When this TP is applied (enacted), the pre-modeled interactive behavior is activated for the tasks in the resulting ad-hoc process instance.

Task roles enable enhanced flexibility of BTM systems, as they provide an additional abstraction layer. While the interactions are defined and modeled on a generic level through the task roles, these roles can be mapped to organizational roles to enable domain-specific adaptations of the BTM system. For example, an organizational role ‘manager’ can be mapped to the task role controller in the DB on the CTM server. This will provide the corresponding interactions on a task in the TDG when a user with manager permissions logs in. If the ‘manager’ role is further mapped to the analyst task role, additional interactions will be enabled, through which a manager will be able to extract TPs from the task tracking repository and to store them as global best-practice prescriptions.

5. Conclusions

In the presented paper we describe a method for modeling interactions on task representations in BTM systems, which uses generic task-centric roles to enable domain-independent, flexible adaptation of the user interface and the available interactions on task representations. We have shown how task roles can be enriched with a set of application-specific interaction descriptions, supporting various aspects of task management – from task creation to delegation, controlling, contributing to tasks and analyzing user-defined task hierarchies. The study reveals how task roles can provide an abstracted, high-level view for modeling interactions from significantly different perspectives of BTM system usage and can result in enhanced system flexibility.

As further research topics we consider the adding of runtime-dependent interaction properties in the scope of task roles and the cascading of (parent) task properties to emerging (sub)tasks during process execution.

Acknowledgments

The reported work was supported financially by the German "Federal Ministry of Education and Research" (BMBF, project EUDISMES, number 01 IS E03 C). We thank to all participants in our user studies for their time and cooperation.

References

1. Bellotti, V., Dalal, B., Good, N., Flynn, P., Bobrow, D.G., Ducheneaut, N.: What a To-Do: Studies of Task Management towards the Design of a Personal Task List Manager. In: CHI'04, pp. 735–742. ACM Press, New York (2004).
2. Beyer, H., Holtzblatt, K.: Contextual Design: Defining Customer-Centered Systems. Morgan Kaufmann (1998).
3. Button, G.: What's Wrong With Speech-Act Theory. Computer Supported Cooperative Work, vol. 3, no.1 (Mar. 1994), pp. 39-42. Springer (1994).
4. Fischer, G., Giaccardi, E., Ye, Y., Sutcliffe, A., Mehanjiev, N.: Meta-Design: A Manifesto for End-User Development. Communication of the ACM, vol. 47, no. 9 (Sep. 2004).
5. Flynn, P., Curran, K., and Lunney, T.: A decision support system for telecommunications. International Journal of Network Management, vol. 12, no. 2 (Mar./Apr. 2002), pp. 69-80. John Wiley & Sons (2002).
6. Gaffar, A., Sinnig, D., Seffah, A., Forbig, P.: Modeling patterns for task models. In: Proceedings of the 3rd annual Conference on Task Models and Diagrams, pp. 99–104. ACM Press, New York (2004).
7. Grebner, O., Ong, E., Riss, U., Brunzel, M., Bernardi, A., Roth-Berghofer, T.: Task Management Model. http://nepomuk.semanticdesktop.org/xwiki/bin/view/Main1/D3-1.
8. John, B., Kieras, D.: The GOMS family of analysis techniques: Comparison and contrast. ACM Transactions on Computer-Human Interaction, vol. 3, no. 4 (1996), pp. 320-351.
9. Kolp, M., Giorgini, P., J. Mylopoulos.: Organizational Patterns for Early Requirements Analysis. In: Proceedings of the 15th International Conference on Advanced Information Systems Engineering (CAiSE'03), Velden, Austria, June 2003.
10. Lieberman, H.: Your Wish is My Command: Programming by Example. Morgan Kaufmann (2001).
11. Lieberman, H., Paterno, F., Wulf, V.: End-User Development. Springer (2006).
12. MacLean, A., Carter, K., Lövstrand, L., Moran, T.: User-tailorable systems: pressing the issues with buttons. In: Proc. CHI 1990, pp. 175-182. ACM Press, New York (1990).
13. Molina, A., Redondo M., Ortega, M.: Applying Pattern-Based Techniques to Design Groupware Applications. LNCS vol. 4101, pp. 225-233. Springer (2006).
14. Palanque, P., Basnyat, S.: Task Patterns for Taking into Account in an Efficient and Systematic Way Both Standard and Abnormal User Behaviour. In: IFIP 13.5 Working Conference on Human Error, Safety and Systems Development, Toulouse, France, pp. 109–130 (2004).
15. Paternó, F.: Model-Based Design and Evaluation of Interactive Applications. Springer, Heidelberg (2000).
16. Paterno, F., Mancini, S., Meniconi, S.: ConcurTaskTree: a diagrammatic notation for specifying Task Models. In: Proceedings Interact'97, pp. 362-369. Chapmann & Hall (1997).

17. Rabelo, L., Eskandari, H., Shalan, T., Helal, M.: Supporting simulation-based decision making with the use of AHP analysis. In: Proceedings of the 37th conference on Winter simulation, pp. 2042 – 2051. Winter Simulation Conference (2005).
18. Riss, U., Rickayzen, A., Maus, H., v. d. Aalst, W.: Challenges for Business Process and Task Managemen. Journal of Universal Knowledge Management vol. 0, no. 2 (2005), pp. 77-100.
19. Stoitsev, T., Scheidl, S., Spahn, M.: A Framework for Light-Weight Composition and Management of Ad-Hoc Business Processes. LNCS 4849, Springer (2008).
20. Stoitsev, T. Scheidl, S., Flentge, F., Mühlhäuser, M.: Enabling End Users to Proactively Tailor Underspecified, Human-Centric Business Processes. In: Proceedings of the 10th International Conference on Enterprise Information Systems, Barcelona, Spain (2008).
21. Veer, G. v. d., Lenting, B., Bergevoet, B.: GTA: Groupware task analysis - modeling complexity. Acta Psychologica 91 (1996), pp. 297-322.
22. Wiig, K. M.: People-focused knowledge management: how effective decision making leads to corporate success. Elsevier Butterworth–Heinemann (2004).
23. Winter, A.F., Ammenwerth, E., Bott, O.J., Brigl, B., Buchauer, A., Gräber, S., Grant, A., Häber, A., Hasselbring, W., Haux, R., Heinrich, A., Janssen, H., Kock, I., Penger, O.-S., Prokosch, H.-U., Terstappen, A., Winter, A.: Strategic information management plans: the basis for systematic information management in hospitals. International Journal of Medical Informatics vol. 64, no. 2 (Dec. 2001), pp. 99–109. Elsevier Science, Ireland (2001).