Extended Abstract: Managing Complexity in Activity Specifications by Separation of Concerns

Peter Forbrig, Gregor Buchholz

University of Rostock
Albert-Einstein-Str. 22
18051 Rostock, Germany
Tel.: +49 381 498 7620

[peter.forbrig|gregor.buchholz@uni-rostock.de

Abstract. The specification of activities of the different stakeholders is an important activity during the requirements engineering phase of software development. Currently, a lot of specification languages like task models, activity diagrams, state charts, and business specifications are used to document the results of the analysis of the domain in most projects. In the paper the aspect of the separation of concerns of global cooperation and individual work by subject-oriented specifications is discussed. It will be demonstrated how task models can be used to support subject-oriented specification by so called team models and role models. This can be done in a more precise way than S-BPM specifications. Restrictions in instances of roles can be specified.

Keywords: Activity Specification, Business-Process Modeling, WorkflowSpecification, Subject-oriented BPM, Subject-oriented Task Models, Cooperative-Task Execution.

1 Introduction

Modeling seems to become more and more significant for the implementation of interactive systems. Specification languages like UML or BPMN are very important for academia and industry. From our point of view, modeling is an attempt to reduce the complexity of software development. Separation of concerns in models might be another approach. Subject-oriented specifications in S-BPM [3] can support this idea. It can be characterized by the separation of models for different subjects and a joined communication model. The paper is structured in the following way that the advantages of S-BPM will be discussed first. This will be followed by an approach with more precise specifications using task models. A summary will close the paper.

The full version of the paper will be submitted to Issue 8 of the journal Complex Systems Informatics and Modeling.

2 Subject-oriented Business-Process Modeling

Fleischmann et al. [4] provided an approach that allows the rapid execution of business-process specifications. It is called Subject-oriented Business Process Modeling and is supported by a language that is called S-BPMN [3]. This approach perfectly fits to the idea of separation of concerns. There is no unified model of all activities but an overview by a communication model and separated process models for all subjects involved. These subjects are most of the time humans.

For S-BPM it is suggested that you start with modeling the communication of subjects via messages first. In this way, the big picture is specified. Later, the dynamic behavior of each subject is specified by finite automata. The available symbols of S-BPM are presented in Fig. 1.

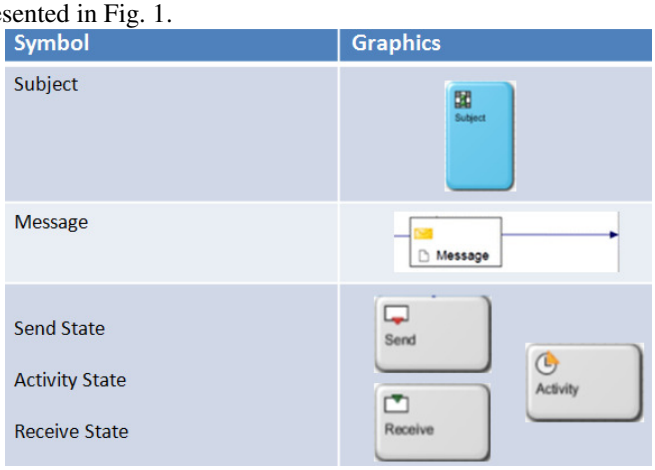

Fig. 1. S-BPMN notation

Requirements are the basis of a communication model that itself is the basis for the implementation. There are only two subjects in **Fig. 2**. They also exchange only one message. The subject *Customer* sends an *Order* to subject *Supplier*.

The details of the behavior of the subjects are defined by specific models. These models specify how subjects react on messages and under which condition they send messages.

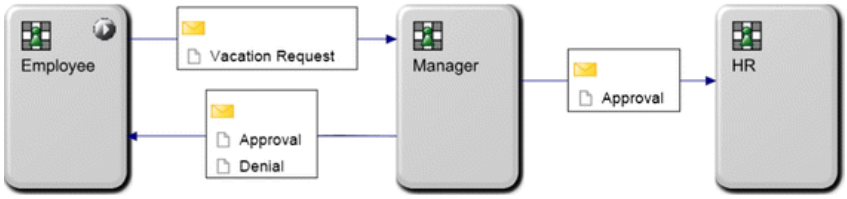

Fig. 2. Example of message exchange between subjects (from [2])

We will recall an example from [2]. It is based on the following natural language description.

"An employee fills in a holiday application form. He/She puts in a start and end date of his/her vacation. The responsible manager checks the application and informs the employee about his/her decision; the holiday request might be rejected or get approved. In case of approval the holiday data are sent to the human resource department which updates the days-off in the holiday file." ([2], p. 220)

The specification of the behavior of an employee is presented in Fig. 3. It provides details of the business process from the perspective of the subject Employee. It specifies the behavior of an employee. This includes the communication with the subject Manager. It can be seen from this specification in Fig. 3. that a function in a function state (e.g. "Fill out Vacation request Form") leads automatically to a message that informs about the ending of the performance of the function (e.g. "Fill out Vacation request form done") and this leads to the action of "Sending it". This action results into a message being sent to the manager.

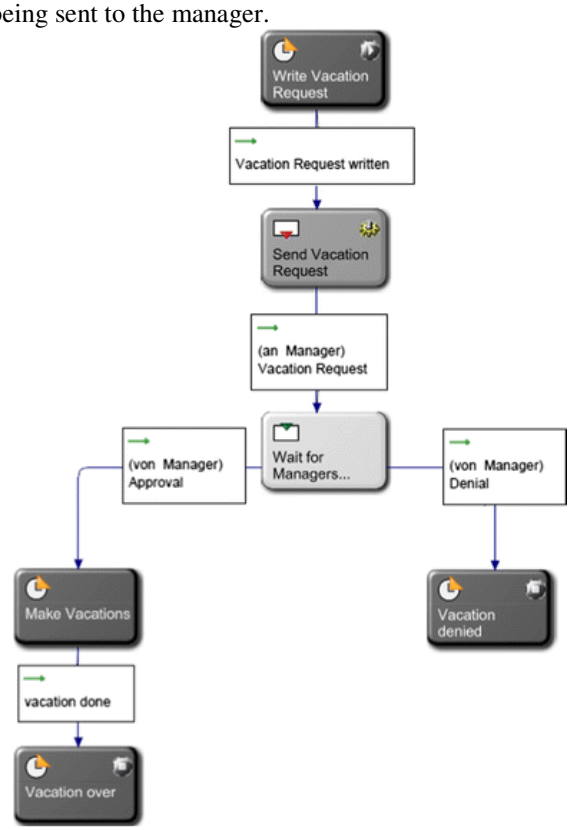

Fig. 3. Example specification of the behavior of subject *Employee* (from [2])

A S-BPM specification drastically reduces the number of elements of BPMN. It provides an interesting combination of overview diagrams and detailed diagrams. These detailed specifications are very much reduced in their complexity in this way. However, there is no need to use always the S-BPM notation.

Task models have been used for several years to develop interactive systems [6]. We presented ideas to use task models for workflow systems [1] and we used task

models for providing assistance in smart environments [8]. For this purpose, the specification language CTML was developed [7]. It consists of a team model, certain task models of stakeholders (described as roles), and state machines for the specification of the behavior of devices.

The team model was further refined to support the subject-oriented approach. We will show how the team model can play the role of the communication model and how task models describe the behavior of subjects.

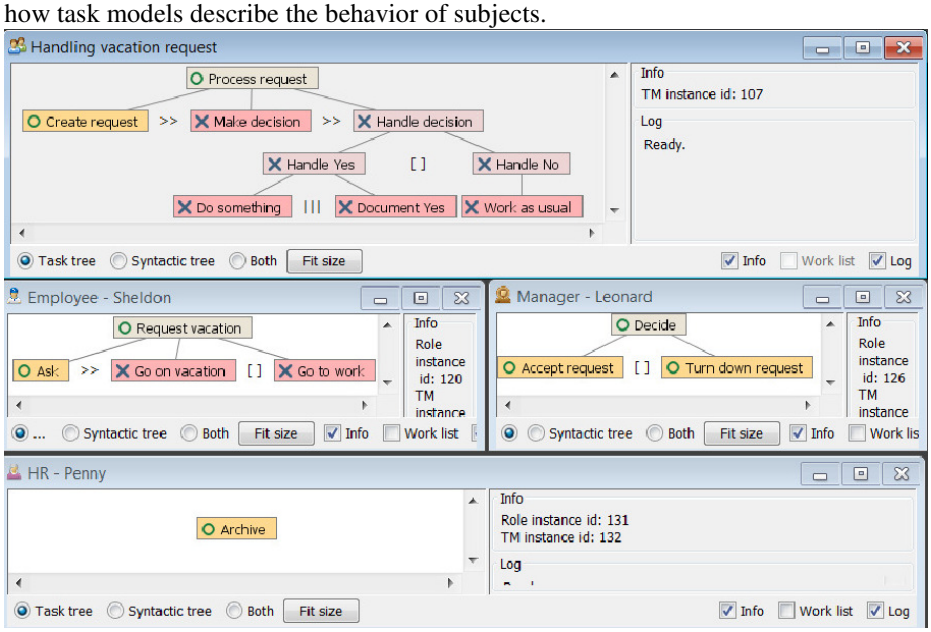

Fig. 4. Team model instance and role model instances

The team model is the upper part of Fig. 4. It specifies that for processing a vacation process the request has to be created first. Afterwards a decision has to be made and finally the decision has to be handled. This can be done in two ways. Either a positive or a negative decision has to be handled. In the positive case, the employee can have holidays and do something. In parallel, the vacation has to be documented. In the negative case, the employee has to continue his work. The middle part of Fig. 4 presents the task model for the role Employee. There can be several instances of such a model. Currently the instance for Sheldon is shown. Leonard performs the role Manager. Sheldon has first to ask for vacation, afterwards she can go on a vacation or go to work. Leonard can accept or turn down the request. The lower part is the simple model for the role HR (human resource – responsible for personal). Penny has to archive the vacation request only.

These models are already presented in an animated mode. The green cycle signals the corresponding task can be started. Leafs of animated instances of role models can be animated only. One cannot interact with the team model. Its state changes are based on the other models. They provide the necessary events and conditions. However, the team model can restrict other models. Certain task may not be able to be executed while the team model is in a certain state.

The start trigger of the team task Create request is Employee.oneInstance.Ask.start, which means that it is started when any instance of Employee started Ask. The end trigger might be Employee.oneInstance.Ask.end. The language for the conditions is a kind of simplified predicate logic in the notation of OCL. Currently, one can specify oneInstance or allInstances in the conditions. According to our experience, there is no urgent need for more expressive expressions.

The specified conditions for the team model fit to our mental model if only one instance of the role Employee exists. Otherwise, start and end could be executed by different instances. This might not be the intention of the modeler. Therefore, the idea of binding context and references to role instances was introduced. The first time an instance provides a trigger, its identification can be stored in a variable.

Assuming that for role Employee the variable E1 is define, the conditions can change to E1.Ask.start and E1.Ask.end. In this case, the same instance is referred to. The same can be specified for the role Manager. The same manager that starts the decision has to end it. However, this might not be necessary. Any other manager might have the right to end the decision. For an employee it might be fine if any manager gives the okay. Nevertheless, the employee that is allowed to go on vacation has to be the same as that who asked for vacation.

These details can be seen in the specification below. This is a more precise specification than the S-BPM model presented above. S-BPM does not allow specifying such details. It is simply assumed that always the same instances are involved. However, this might not always be the case.

The main part of the team model looks like this:

```
<task name="Process request" bindingContext="C1">
<roleVar role="Employee" var="E1"/>
<roleVar role="Manager" var="M1"/>
<roleVar role="HR" var="H1"/>
<task name="Create request" operator="enabling"
startTrigger="E1.Ask.start" endTrigger="E1.Ask.end" />
<task name="Make decision" operator="enabling"
startTrigger="M1.Decide.start" endTrigger="M1.Decide.end" />
<task name="Handle decision">
<task name="Handle Yes" operator="choice">
<task name="Do something" operator="interleaving"
startTrigger="E1.GoOnVacation.start"
endTrigger="E1.GoOnVacation.end">
<bindPrec source="M1.AcceptRequest" target="E1.GoOnVacation" />
</task>
<task name="Document Yes" startTrigger="H1.Archive.start"
endTrigger="H1.Archive.end" />
</task>
<task name="Handle No">
<task name="Work as usual" startTrigger="E1.GoToWork.start"
endTrigger="E1.GoToWork.end" />
</task>
</task>
</task>
```

Recently, the interpreter for the task models was implemented in a cloud. In this way task models can be executed via the Internet in a cooperative way. This allows a subject-oriented development of interactive systems for different platforms.

3 Summary & Outlook

Subject-oriented modeling of business processes allows reducing complexity by separation of concerns. Separate models help to focus on specific details without the need of looking at the global relations all the time.

Specific task models supporting the subject-oriented specification of business processes were introduced. The expressiveness of this approach was shown by a simple example from the subject-oriented community. This specification is more precise than the corresponding S-BPM specifications. The corresponding tool support of an interpreter was implemented in Java.

Some more examples have to show the appropriateness of the suggested approach.

4 References

1. Brüning, J., Forbrig, P.: TTMS: A Task Tree Based Workflow Management System. In: Halpin, T., Nurcan, S., Krogstie, J., Soffer, P., Proper, E., Schmidt, R., Bider, I. (eds.) BPMDS 2011 and EMMSAD 2011. LNBIP, vol. 81, pp. 186–200. Springer, Heidelberg (2011).
2. Fleischmann, A., & Stary, C. (2012). Whom to talk to? A stakeholder perspective on business process development. Universal Access in the Information Society, 11(2), 125-150, free download http://link.springer.com/article/10.1007/s10209-011-0236-x (2011)
3. Fleischmann, A., Schmidt, W., and Stary, C.: Open S-BPM= open innovation. In S-BPM ONE-Running Processes, pp. 295-320. Springer Berlin Heidelberg, (2013)
4. Fleischmann, A., Schmidt, W., and Stary, C.: Requirements Specification as Executable Software Design – A Behavior Perspective, in 0 p. 9-18, 2015.
5. Matulevičius, R. et al. (Eds.): REFSQ Workshop proceedings, http://ceur-ws.org/Vol-1342/ (2015)
6. Puerta A. R.: A Model-based Interface Development Environment, IEEE Software, 14(4):40-47, July/August (1997)
7. Wurdel, M. Sinnig, D. and Forbrig, P.: CTML: Domain and task modeling for collaborative environments. Journal of Universal Computer Science 14(19) pp. 3188-3201 (2009)
8. Zaki, M., Forbrig, P.: Making task models and dialog graphs suitable for generating assistive and adaptable user interfaces for smart environments. In: Proc. PECCS 2013, pp. 66-75, Barcelona, Spain (2013)