Integrating HCD into BizDevOps by Using the Subject-Oriented Approach

Peter Forbrig(✉) and Anke Dittmar

University of Rostock, Albert-Einstein-Str. 22, 18051 Rostock, Germany
{peter.forbrig,anke.dittmar}@uni-rostock.de

Abstract. The DevOps-approach becomes more and more important because of the success of agile software development in conjunction with the continuously changing reality. It aims at unifying development and operations. A common team is responsible for both domains. Additionally, there are approaches like Continuous Software Engineering with the intention to unify business administration (Biz) and development. Even tool chains for BizDevOps are possible. The paper discusses aspects of BizDev and BizDevOps using a subject-oriented approach for supporting aspects of HCD. The focus lies on modeling user activities and business processes. Additionally, the role of domain-specific textual languages is discussed. Most important is the fact that methods from HCI like task modeling or storytelling can support BizDev and BizDevOps.

Keywords: Model-based approach · Stories · DevOps · BizDev
BizDevOps

1 Introduction

Changing requirements of interactive software systems are rather the rule than the exception. Therefore, software solutions have to be continuously updated during operation in order to respond properly to contextual changes. Classical development methods fail to address this challenge and are more and more replaced by agile approaches. Together with the increasing popularity of agile software development methods an increasing need for continuously deploying software arose. The idea of DevOps as a clipped compound of development and operations was born. It comes with automatic quality assurance and continuous delivery.

Unfortunately, agile methods such as SCRUM often lack a human-centered design perspective. Design activities such as those recommended in the ISO 9241-210 on the human-centered design process are insufficiently integrated into agile software development. For example, agile methods often do not support a systematic exploration of alternative solutions. Existing approaches to integrate HCD activities into agile process models mainly focus on the design of the user interface (UI). However, it is common ground in the fields of human-computer interaction (HCI) and interaction design that UI design needs to be informed by the analysis of the organizational context and the users' tasks and needs. It is assumed that business process models, task models, or user stories as results from such broader analysis help to derive UI models of higher quality.

Published by Springer Nature Switzerland AG 2019
C. Bogdan et al. (Eds.): HCSE 2018, LNCS 11262, pp. 327–334, 2019.
https://doi.org/10.1007/978-3-030-05909-5_21

This paper suggests a better integration of subject-oriented business modeling and task-modeling activities into agile development approaches.

2 DevOps, BizDev and BizDevOps

DevOps is currently discussed a lot in industry. It is related to development technologies and organizational aspects. "DevOps is defined as a paradigm or set of principles focuses on software delivery through enabling continuous feedback, quick response to changes and using automated delivery pipelines resulting in reduced cycle time" [13].

Very important aspects are the monitoring of the running software and the feedback for improvements in short intervals. DevOps is also discussed in the context of continuous software engineering [5]. First ideas of a disappearing boundary between development-time and run-time were published by Bares and Ghezzi [1]. In their abstract, they state: "Models need to continue to live at run-time and evolve as changes occur while the software is running." Business process models and task models seems to be candidates for such an approach. According to Humble and Farley [12], Continuous Delivery and DevOps have common goals and are often used in conjunction. However, there are subtle differences. "While continuous delivery is focused on automating the processes in software delivery, DevOps also focuses on the organization change to support great collaboration between the many functions involved" [12].

Fitzgerald and Stol [5] argue that there has to be a continuous integration of business strategy and software development. They use the term BizDev for this purpose. In the framework of Continuous Software Engineering BizDev and DevOps are separated. However, a combination of both is possible. Gruhn and Schäfer address with BizDevOps "the boundary between IT and business departments in order to allow business departments to participate hands-on in the development of parts of the system and at the same time having measures in place that allow IT to safeguard the development process" [11]. They argue that the approach makes sense for systems that reflect business innovations. It might not be appropriate for general-purpose software development.

3 Subject-Oriented Business-Process Modeling

Fleischmann et al. [6] characterize subject-oriented business-process management (BPM) as socially executable BPM. They further argue: "As organizations need to act flexible in the continuously changing landscape of the digital economy, their process work is increasingly driven by valued interactions among stakeholders [14]. Traditional … BPM does no longer fit to this changing view of processes".

It is therefore necessary to have a specification language that on the one hand is simple to learn and to use. On the other hand, the specification should be executable. The best would be if domain experts can specify their business process models by

themselves. S-BPM seems to be a solution for that. Practitioners from industry report on the Metasonic[1] web page about such success stories. S-BPM specifies business processes from the perspective of subjects that communicate via messages and provides a simple notation. Subjects can be humans or software agents.

S-BPM [7] is a graphical specification language that has five language elements only. These elements are subject, message, send state, function state, and receive state. S-BPM specifications start with modeling of a communication diagram. It represents possible communications of subjects via messages. The big picture of an application is specified in this way. Details of the behavior of each subject are specified later by finite state machines. Figure 1 provides an example form [7] of a communication diagram for a vacation request of an employee. The request goes to a manager who decides about acceptance. If the request is approved human resources (HR) and the employee are informed accordingly by an approval message. If the request is turned down, only the employee gets a denial message. The subject employee is able to start the communication.

A communication diagram visualizes possible message exchanges. However, the sequence and dependences of messages are not specified. This is done in a behavioral diagram. Each subject is characterized by exactly one diagram. It consists of states and messages. Figure 2 provides in its left part a model for the dynamic behavior of an employee that consists of five states and five messages.

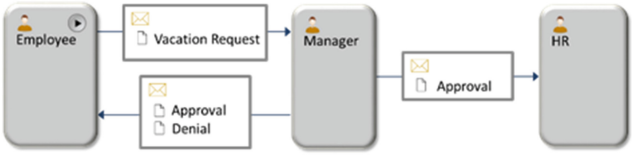

Fig. 1. Example of an S-BPM communication diagram

According to the provided specification, an employee fills a form first and sends afterwards the filled form to a manager. Having done this, the employee has to wait for an answer message from a manager. If the request is accepted, the employee can go on vacation and afterwards go to work. Otherwise, the employee has to go to work immediately. It is the intention of S-BPM to provide tool support for end-user modelling. Stakeholders should be able to edit their own behavior model.

There is the saying that "a picture is worth a thousand words". However, sometimes it is good to have a textual domain specific language as an alternative to graphical specifications.

It was our intention to have a look at a textual representation of behavioral models of S-BPM as well. We modified the grammar of the example from Fowler in such a way that behavioral specifications of S-BPM can be expressed. Right part of Fig. 2 expresses the specification of the left part of Fig. 2 in a domain-specific language.

[1] https://www.metasonic.de/en, last visited January 21, 2018.

Based on the textual specification java code can be generated and executed. This provides a further perspective and different experience with the domain model.

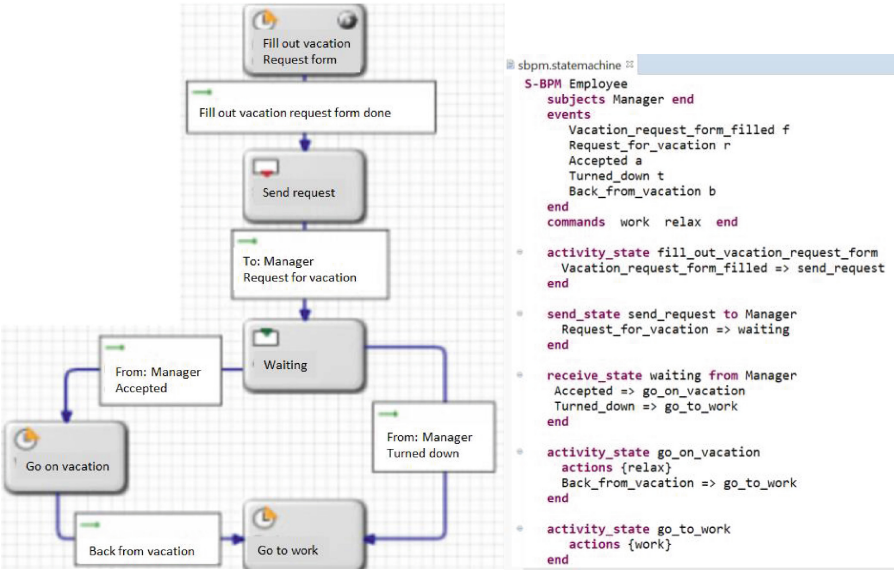

Fig. 2. Behavioral diagram for subject employee (adapted from [7]) on the left and specification in a textual DSL on the right.

Alternatively, to the suggestion of S-BPM, activities of subjects could be specified as task models as well. The following section will discuss this aspect.

Task models are traditionally applied for designing and developing interactive systems. They describe the logical activities of users and can be the basis for user interface design. Each role of users is specified by a separate task model. The concept of roles is equivalent to the concept of subjects. Therefore, task models can specify the behavior of subjects. The DSL-CoTaL [3] was designed for such a purpose. It uses the temporal operators from CTT [4]. Additionally, preconditions can be specified in an OCL-like language. Figure 3 provides the example of the behavior of subject employee in the context of a vacation request.

```
role Employee for Vacation {
    root Asking_for_vacation = Fill_form >> Ask >> (Go_on_vacation [] Go_to_work);
       task Ask pre HR.oneInstance.Announce_holidays;
       task Go_on_vacation pre Manager.oneInstance.Accept_request;
       task Go_to_work pre Manager.oneInstance.Turn_down_request;
}
```

Fig. 3. Task model for employee.

For a vacation request, an employee has to fill a form. Afterwards (≫-enabling), the employee has to ask a manager. After the employee asked, there is a choice between two tasks, going on vacation or going to work. Asking for vacation is only possible if the corresponding precondition is fulfilled. One instance of the subject human resources had to have performed the task of announcing the possibility of asking for holiday requests – in short announce holidays.

Xtext[2] can be used together with Xtend[3] for code generation. In this way, task models specified in DSL-Cotal can be visualized in different tools (see [10]).

Stories can help as well. In computer science, the term "user story" is used in different ways. They make things more interesting and improve the engagement of participants. The scenario-based approach from Rosson and Carroll [15] is based on descriptions of the use of the envisioned system.

Storytelling is used in many cultures as a means of education. However, stories are not only used for software development. They are also used in business management. Fog et al. characterize "storytelling as a Management Tool" [8]. They mention: "The stories we share with others are the building blocks of any human relationship. Stories place our shared experiences in words and images". It seems to be an excellent communication tool for BizDevOps.

4 Combining HCI Approaches with BizDevOps

S-BPM with its restricted number of language elements lets users successfully specify the behavior of their own role, their subject. Task models have been used for requirements analysis and for user-interface design. Traditionally each role is specified by a task model. This can be considered as subject-orientation. Therefore, it makes sense to use task models for business processes as well. They can replace behavioral models of S-BPM. The different models of S-BPM and task models open new perspectives that can be even further broadened by textual domain specific languages. Tool support for language engineering exists by Xtext and Xtend. It is also possible to

```
team Vacation {
    root Handling_vacation_requests =
         HR.Announce_holidays >> Process_request{*} [> HR.Finish;
       task Process_request =
            Employee.Ask >> Handle_decision >> Act_accordingly;
         task Handle_decision =
              Manager.Accept_request [] Manager.Turn_down_request;
         task Act_accordingly =
             Employee.Go_on_vacation [] Employee.Go_to_work;
}
```

Fig. 4. Team model for the holiday request example.

[2] https://www.itemis.com/en/xtext/, last visited January 20, 2018.

[3] http://www.eclipse.org/xtend/, last visited May 7th 2018.

specify the cooperation of subjects by task models. It is called team model in DSL-CoTaL [2, 9]. A team model is a counterpart to the communication diagram of S-BPM.

The team model consists of tasks that are executed in cooperation. A task can be preceded by a subject name. This means that the task has to be performed by an instance of this subject. Handling vacation requests is started and finished by an instance of subject HR by executing task Announce-holidays and Finish respectively.

Storytelling seems to be a good method as well for people from business administration as for developers. Textual DSLs can support the specification of task models based on stories. However, they can also be the basis for creating stories. The team model from Fig. 4 inspires the following story.

After Paula from HR sends an email to all employees to inform them that they can ask for holidays. After Fred asks for vacation, Manager Chris turns down the request from Fred and Fred goes to work. Afterwards Susan asks for vacation and Chris accepts the request from Susan and she goes immediately on vacation. Finally, Paula finishes the vacation request period.

While creating the story one might recognize that the strict order in the iterations is not reflecting the reality. Instance iteration ({#}) is a better model. It allows the start of a new iteration before the previous one was finished.

Figure 5 reflects the situation after Fred asked for vacation by animated model instances.

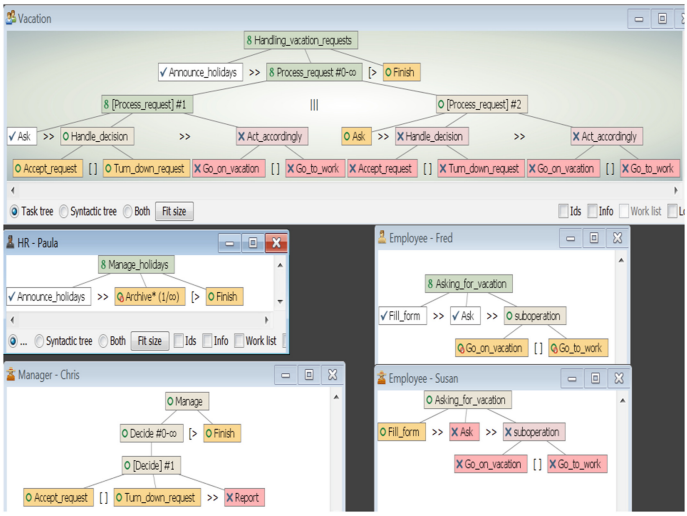

Fig. 5. Animated team model and subject model instances

On the top of Fig. 5 one can see the animated team model with instance iteration. A first iteration for a request is started. The second instance of an iteration is already prepared to be executed in parallel. One can see below, that Paula already announced holidays. Fred filled his form and asked already. Susan and Chris did not do anything yet. However, both can act while Fred has to wait for a decision.

The animation with CoTaL allows the dynamic creation of subject instances. Therefore, stories can really be well explored and improved. These stories help to validate models and the final application. The different kinds of knowledge representation (statecharts, task models, stories) should be used intertwined. Figure 6 describes the intended application of models and tools for functional requirements.

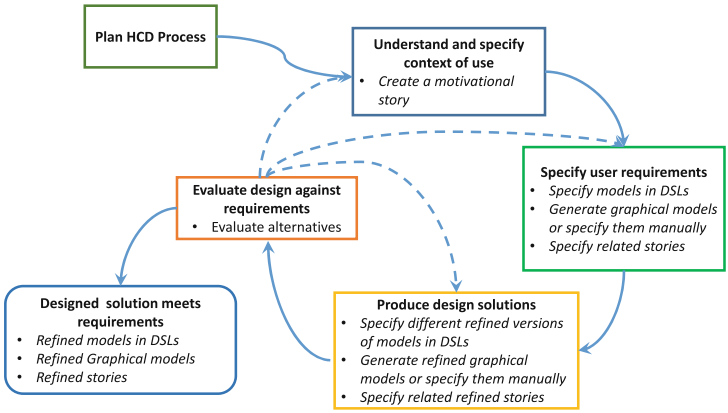

Fig. 6. Suggested human-centered-design process for functional requirements for BizDevOps

The structure of the process corresponds to the design process from ISO 9241-210. Additionally, the intended use of textual specification in DSLs, graphical specifications, and stories is added. All specifications are optional. Let us know discuss a possible tool chain for BizDevOps and the subject-oriented approach.

Gruhn and Schäfer [11] provided a software architecture for BizDevOps. The discussed tools for task modeling can be integrated as app with a corresponding plugin. For S-BPM there exists already a runtime environment that is used in several industrial companies. It can be integrated into the discussed architecture as well. In this way, the subject-oriented approach can get support by DevOps features like continuous deployment and continuous monitoring.

5 Summary and Outlook

The approaches DevOps, BizDev, and BizDevOps were discussed in the context of HCD. Communication between stakeholders is identified as the most important aspect. Therefore, social, cultural, and communication skills are necessary for all stakeholders.

Stories have been successfully used in HCI and in business administration. Therefore, they seem to be a perfect tool for BizDev and BizDevOps. They should be combined with subject-oriented notations and methods. Statecharts and task models are candidates for knowledge representation for subject behavior. Domain specific textual languages were suggested in conjunction to graphical notations. Providing different perspectives (states &tasks & stories, graphics & text) allows further insights into the

domain. Tool support supports the creation of different versions of models. Therefore, alternatives can be specified without many efforts. Stories and models can be explored and the HCD process can be applied to DevOps, BizDev, and BizDevOps.

It might be possible to create even more abstract textual DSLs that fit to cognitive models of people from the business domain. A language allowing expressing stories could be a candidate for that. This language must have the option to add general information like iteration to certain expressions. In this way, models can be extracted from such specifications.

References

1. Baresi, L., Ghezzi, C.C.: The disappearing boundary between development-time and run-time. In: Future of Software Engineering Research (2010)
2. Buchholz, G., Forbrig, P.: Extended features of task models for specifying cooperative activities. PACMHCI **1**(EICS), 7:1–7:21 (2017)
3. CoTaSE: https://www.cotase.de/. Accessed 20 Jan 2018
4. CTTE: http://hiis.isti.cnr.it:4500/research/CTTE/home. Accessed 15 Jan 2018
5. Fitzgerald, B., Stol, K.-J.: Continuous software engineering and beyond: trends and challenges. In: Proceedings of 1st International Workshop on Rapid Continuous Software Engineering - RcoSE 2014, pp. 1–9. ACM, New York
6. Fleischmann, A., Schmidt, W., Stary, C.: Subject-oriented BPM = socially executable BPM. In: Proceedings of the 15th IEEE Conference on Business Informatics (CBI 2013), Vienna, pp. 399–406. IEEE Computer Society (2013)
7. Fleischmann, A., Schmidt, W., Stary, C.: Open S-BPM = open innovation. In: Fischer, H., Schneeberger, J. (eds.) S-BPM ONE 2013. CCIS, vol. 360, pp. 295–320. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36754-0_19
8. Fog, K., Budtz, C., Munch, P., Blanchette, S.: Storytelling as a management tool. In: Storytelling. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-540-88349-4_6
9. Forbrig, P., Buchholz, G.: Subject-Oriented specification of smart environments. In: S-BPM ONE, p. 8 (2017)
10. Forbrig, P., Dittmar, A., Kühn, M.: A textual domain specific language for task models - generating code for CoTaL, CTTE, and HAMSTERS. In: Proceedings of EICS 2018, Paris (2018). https://doi.org/10.1145/3220134.3225217
11. Gruhn, V., Schäfer, C.: BizDevOps: because DevOps is not the end of the story. In: Fujita, H., Guizzi, G. (eds.) SoMeT 2015. CCIS, vol. 532, pp. 388–398. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-22689-7_30
12. Humble, J., Farley, D.: Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation. Pearson Education, London (2010)
13. Jabbari, R., bin Ali, N., Petersen, K., Tanveer, B.: What is DevOps?: a systematic mapping study on definitions and practices. In: Proceedings of the Scientific Workshop Proceedings of XP2016 (XP 2016 Workshops), Article 12. ACM, New York (2016)
14. Li, C., Bernoff, J.: Groundswell: Winning in a World Transformed by Social Technologies. Harvard Business Press, Boston (2008)
15. Rosson, B., Carroll, J.M.: Scenario-based design (Chap. 53). In: Jacko, J., Sears, A. (eds.) The Human-Computer Interaction Handbook Fundamentals, Evolving Technologies and Emerging Applications, pp. 1032–1050. Lawrence Erlbaum Associates, Mahwah (2002)