Stories, Use-Case Slices and Behavioral Models: Unifying Stakeholders' Views

Peter Forbrig and Anke Dittmar

University of Rostock, Department of Computer Science Chair of Software Engineering, Albert-Einstein-Str. 22, 18055 Rostock, Germany
[peter.forbrig | anke.dittmar]@uni-rostock.de

Abstract. Software development is a multidisciplinary process with typically many stakeholders being involved. This paper looks at stories as a means to consider their different viewpoints. Although stories are generally appreciated to increase empathy and evoke discussion, the understanding of what forms a story differs widely from sub-discipline to sub-discipline. The paper discusses applications of different kinds of stories and their cross-pollination with other behavioral models. Domain-specific languages are used to specify story-related behavior by task models and statecharts. The organization of business trips is used as a running example to illustrate the ideas.

Keywords: Domain-specific languages; Task models; Subject-oriented specifications

CEUR-WS.org/Vol-2503/paper2_2.pdf

1 Introduction

The development of interactive systems requires the consideration of multiple viewpoints raised by various stakeholders. User-centred design responds to this challenge by an iterative development with early and active involvement of users and other stakeholders. Agile approaches as they are currently popular in the software development community facilitate quick responses from customers and users. Artefacts, such as prototypes that can be understood by all stakeholders play an important role in these design approaches as they evoke discussion and facilitate the development of a shared understanding. In this position paper, we reflect on the use of stories as design artefacts. We first look at the different understandings of stories and their utilization in different disciplines, such as interaction design and business process modeling. We then suggest how to relate these different "types of" stories and integrate them with other design artefacts needed for software development such as behavioral models of subject-oriented specifications in S-BPM [8]. The paper continues our work in [5]. The organization of business trips is used as a running example throughout the position paper.

Copyright © 2019 for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

2 Related Work

This section gives a short overview of the understandings and use of stories in interaction design, business-process modeling, agile development, and requirements engineering.

2.1 Stories in Interaction Design

According to Quesenbery and Kevin Brooks [18], "Stories have always been part of user experience design as scenarios, storyboard, flow charts, personas, and every other technique that we use to communicate how (and why) a new design will work. As a part of user experience design, stories serve to ground the work in a real context by connecting design ideas to the people who will use the product". The authors report five advantages of user stories.

1. They help to gather and share information about users, tasks, and goals.
2. They put a human face on analytic data.
3. They can spark new design concepts and encourage collaboration and innovation.
4. They help to understand the world by giving us insight into people who are different.
5. They can even persuade others of the value of a contribution.

Stories help to describe a problem and to focus on the context of an application. They can also be used to explore design decisions by describing the consequences of new designs in an exemplary way. Stories are most effective in evoking the empathy of stakeholders, if they convey a complex and nuanced understanding of involved characters, their motives, activities, and conflicts. Scenarios as they are developed in the scenario-based approach by Rosson and Carroll [19] to describe current and envisaged situations are more complex narratives in this sense. The authors comment: "Narrative descriptions of envisioned usage episodes are then employed in a variety of ways to guide the development of the system that will enable these user experiences".

Here is an example of a rather 'flat' Cooper-like persona description as it is also used in some design contexts. It is related to the example of organizing business trips which is used for illustration throughout this paper.

Susan is an IT Manager at Important Limited. She is member of the local cultural association in Smalltown. Susan has to go for several business trips every month. Because she is very much interested in culture she tries to combine her business duties with visits to museums and concerts. Susan is therefore very much interested to book ticket for trains and the reservation of hotel according to events that fit to her business activities.

2.2 Stories in Business-Process Modeling

Stories are also used in business-process modeling, for example, as motivational stories. Fog et al. [9] characterize *storytelling* as management tool and state: "

The stories we share with others are the building blocks of any human relationship. Stories place our shared experiences in words and images. They help shape our perception of "who we are" and "what we stand for". Likewise, stories are told and flow through all companies".

In the context of business administration, Denning [3] suggests eight different story patterns: (1) Sparking action, (2) Communicating who you are, (3) Transmitting values, (4) Communicating your brand, (5) Fostering collaboration, (6) Taming the grapevine, (7) Sharing knowledge, and (8) Leading people into the future. The Sparking action pattern describes how a successful change was implemented in the past, but allows listeners to imagine how it might work in their situation. For transmitting values Denning gives the hint to use believable characters and situations. The brand is usually told by the product or service itself. To foster collaboration, the stories should recount a situations that listeners have experienced. They should be animated to share their own stories. Taming the grapevine refers to storytelling with gentle humor about some aspect of a rumor that reveals it to be untrue. Sharing knowledge focuses on problems and how they were corrected. The story should have an explanation of why the solution worked. Leading people into the future evokes the future that is planned to be created.

Thiel provides in a more recent book [21] techniques for knowledge management in enterprises. In [22] she discusses storytelling as tool for reaching customer and employee loyalty.

In business-process modeling stories can be supportive as well. Simões et al. [20] state that process stories increase the expressiveness of process models. They report: " In this particular study, the creation of individual process stories by staff unveiled numerous de facto practices that were not captured by traditional process elicitation and modeling".

2.3 Stories in Agile Software Development

Part of the agile software development are so called *user stories*. Compared to above discussed approaches, there exists a totally different understanding of a story. A user story consist of one sentence only which has the following structure.

- **As a** ¡role¿, **I want** ¡feature¿ **so that** ¡reason¿.

The *story* of Susan could look as follows:

- **As an** employee, **I want** to manage my business trip **so that** I can combine the trip with cultural events.

While the first kind of stories provides a lot of motivation and detailed context information, the second kind of stories is mainly focused on functional requirements. However, they can be specified on very different levels of abstractions as well. Lawrence [17] provides nine *story-splitting patterns*. They are called *workflow steps, operations, business rule variation, variations in data, break out spike,*

interface variations, major effort, simple/complex, and *defer performance*. Conditions and resulting actions are described. We will recall in detail the workflow step pattern only. It is described by Lawrence [17] by the following example:

- **As a** content manager, **I want** to publish a news story **so that** it is available on the cooperate website.

" It turned out that just to get a few sentence news story on the corporate website required both editorial and legal approval and final review on a staging site. There's no way 6-10 stories like this would fit in an iteration. In a workflow like this, the biggest value often comes from the beginning and end. The middle steps add incremental value, but don't stand alone. So it can work well to build the simple end-to-end case first and then add the middle steps and special cases" [17]. The story was split into several stories like:

- **As a** content manager, **I want** to publish a news story **so that** it is directly available on the cooperate website.
- **As a** content manager, **I want** to publish a news story **so that** it is published with editor review.
- **As a** content manager, **I want** to publish a news story **so that** it is published with legal review.

From our point of view, it makes sense to start with a motivational story and extract parts of the story in the sense of agile software development. These stories can be further refined by *story-splitting patterns*.

2.4 Stories and their Relationship to Use Cases in Requirements Engineering

Use cases [13] are very popular for requirements engineering. They describe a system as it appears to users. They "represent the things of value that the system performs for its actors" [1]. Such actors can be users or other systems.

Use cases can specify a set of interaction sequences between a system and its actors. Use cases descriptions often conform to a structured template containing title, goal, primary actor, level, precondition, main success scenario, alternatives, extensions, etc. A widely used structured template was provided by Cockburn [2].

A use case specification contains a set of scenarios: one main success scenario and several alternative scenarios. Jacobson et al. [15] refer to use-case scenarios as stories. However, they are not stories in the sense discussed above.

Nevertheless, these stories can be characterized by one sentence like those of the agile development.

In the context of agile development with sprints of about four weeks, use cases are considered to be sometimes too complex. Therefore, a new concept was introduced in conjunction with Use Case 2.0 [14] [15]. It is called use-case slice. A use-case slice consists of one or several stories (action scenarios).

According to Jacobson et al. [15], a use-case slice "is created by selecting one or more stories for implementation..., acts as a placeholder for all the work required to complete the implementation of the stories..., and evolves to include the equivalent slices through design, implementation and test". Fig. 1 provides an abstract visualization of a complete use case specification on the left hand side, and its stories (scenarios) and possible slices on the right hand side.

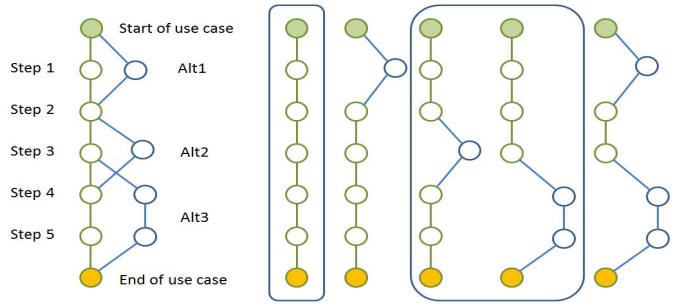

Fig. 1. Use case, use-case slices and their 'stories' (scenarios) according to Jacobson.

On the left hand side of Fig. 1 one can see the main success scenario with its alternatives. The straight line represents the main success scenario. Additionally, alternative paths are sketched. On the right hand side possible stories are shown in detail. They represent one specific path through the use case each. One or several stories can be grouped to use-case slices. One slice is intended to be implemented in one development iteration (sprint). The rest of the stories might be grouped to slices later or are not intended to be supported by the software under development.

The splitting of a use case to different scenarios looks very similar to the splitting patterns of stories. Additionally to the used pattern above, the pattern Variations in Data would result in stories like in Fig. 1.

Let us have a look at an example. Software has to be developed that supports a company in organizing business trips. The story could be characterized as follows:

- **As an** employee, **I want** to get the permission of a business trip, **so that** train ticket and hotel are booked.

The corresponding use case can be called organize a trip. The primary actor is Employee and there are two secondary actors called Manager and Agent. A manager has to give the permission of a business trip and an agent supports the booking of hotel and train ticket. Fig. 2 presents a corresponding use-case diagram.

The main success scenario could be: ask for permission, wait for answer, ask for booking, wait for documents, travel.

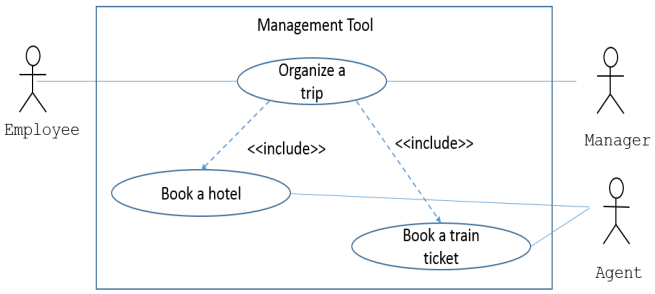

Fig. 2. Use-case specification for organizing business trips.

However, there might be alternatives that an employee books a hotel by himself according to his private preferences. Another alternative could be that the employee books the train ticket as well according to private preferences themselves.

3 Behavioral Models

With S-BPM [7] there exists a quite successful notation for business-process modeling. It follows the idea of a subject-oriented approach. Subjects communicate with other subjects via messages. The behavior of subjects is specified by a variant of state-transition diagrams.

Subjects can be considered as roles like actors in use-case diagrams. This is supported by Fleischmann [6]: "in UML actors in use case diagrams represent a kind of subject...". Therefore, behavior specifications characterize the behavior of several instances of subjects. However, not all of the instances of subjects behave in the same way. Hence, it makes sense to slice the behavior specification of subjects as well. Let us first have a look at parts of the S-BPM specification for our business trip example. Message exchange between subjects Employee, Manager, and Agent is specified in the interaction diagram of Fig. 3.

The behavioral model for subject Employee is presented in Fig. 4. States are similar to Moore states. Actions are performed inside such states. Specific symbols characterize the first and the last state. The same is true for sending and receiving states. In Forbrig [10] a textual DSL for S-BPM was presented. However, the graphical specification needs fewer space. Therefore, we use the graphical representation.

Let us assume that the specification in Fig. 4 is the representation of a whole use-case specification. It specifies four different stories. One can consider the story that a requested trip of an employee is not permitted to be the first one. The other three are real success stories.

The specification in Fig. 4 represents the following four stories in the sense of Jacobson et al. [15]:

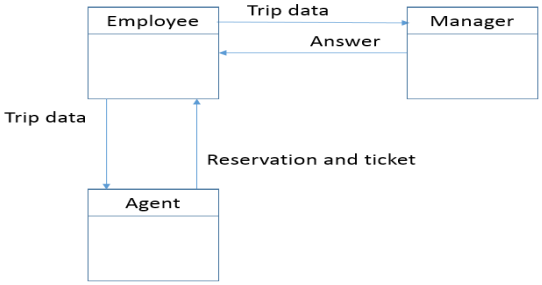

Fig. 3. S-BPM subject interaction diagram.

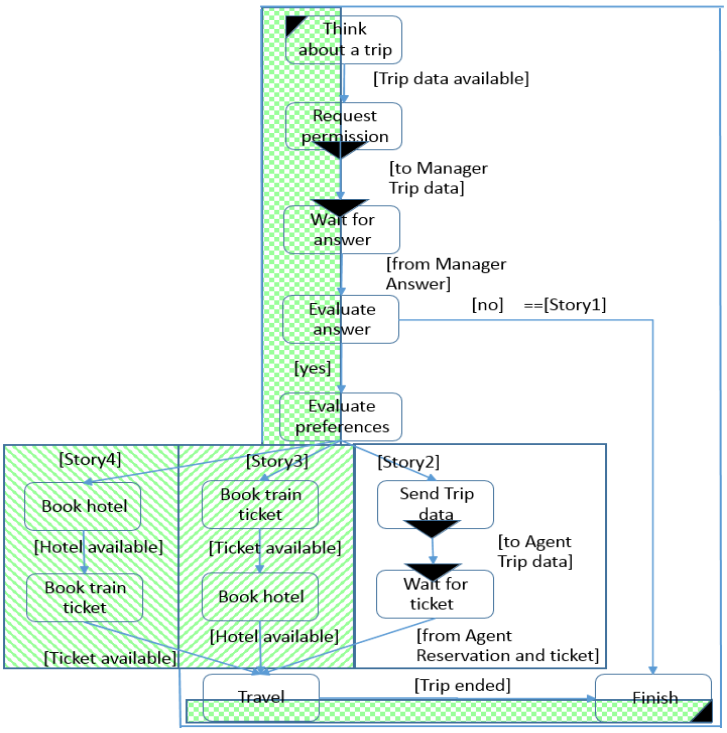

Fig. 4. Behaviour specification for subject Employee.

1. Think about a trip, Request permission, Wait for answer, Evaluate answer, Finish
2. Think about a trip, Request permission, Wait for answer, Evaluate answer, Evaluate preferences, Send Trip data, Wait for ticket, Travel, Finish
3. Think about a trip, Request permission, Wait for answer, Evaluate answer, Evaluate preferences, Book train ticket, Book hotel, Travel, Finish
4. Think about a trip, Request permission, Wait for answer, Evaluate answer, Evaluate preferences, Book hotel, Book train ticket, Travel, Finish

It might make sense to combine stories to two slices for the agile development process. In this way implementation is distributed to two implementation cycles. In our case it was decided that the first slice consists of rejected request (story (1)) and the automatic booking (story (2)) while the second one groups the two stories of organizing the trip by the employee himself (story (3) and (4)). Fig. 4 is the attempt to visualize this fact. The white area represents the first slice, while the shaded area represents the second slice. If states are partly shaded and partly white they belong to both slices. This reflects the fact that all stories start and end in the same way.

Sometimes, it makes sense to provide different views on a behavioral model. With task models [4] the behavior of actors can be specified as well. It is possible to specify stories by specific sub-tasks in this notation. In the textual task specification language DSL-CoTaL (Collaborative Task Modeling Language) [11] this looks like presented in Fig. 5. Additionally to providing a new view on the problem domain, it also demonstrates how task models can be used for agile specifications.

```
role Employee {
  root provide_information =
          think_about_trip >> ask_for_allowance >> (act [] do_not_travel_Story1);
    task act = make_reservation >> travel;
      task make_reservation =
              behave_like_Story2 [] behave_like_Story3 [] behave_like_Story4;
        task behave_like_Story2 = involve_agent;
        task behave_like_Story3 =
                book_train_ticket >> make_hotel_reservation;
        task behave_like_Story4 =
                make_hotel_reservations >> book_train_tickets;
}
```

Fig. 5. Task model in DSL-CoTaL specifying three stories for the business trip example.

4 Discussion

The understanding and use of stories is different across different domains. The application of motivational stories ranges from software development, business

process modeling to general management aspects. Stories might play a more important role in the future. They seem to be a very good representation of different stakeholder's views. For agile software development, stories are used as structured sentences. They can be split by story-splitting patterns or refined by action sequences. Fig. 6 provides an overview of the discussed concepts. First, actors have to be identified for an application. Users and other stakeholders should be asked to write motivational stories involving one or several actors. After analyzing those stories agile stories should be identified. Each motivational story has several of them. Each final agile story is refined by a use-case story. Several of those use case stories are combined to a use-case slice. A behavioral model has to reflect the slice and the stories. Behavioral models can be represented by state machines or task models. With domain-specific languages one could also combine both approaches in one language.

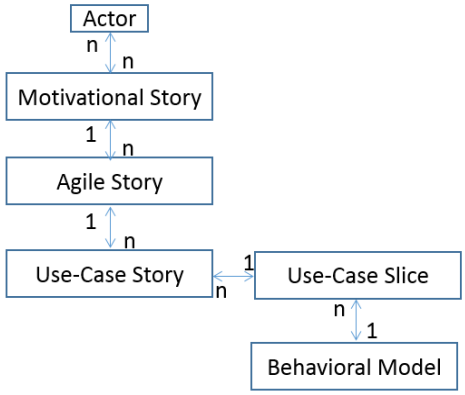

Fig. 6. Relations between concepts.

Khanh et al. [16] used the term *human story* for a combination of user-story and persona-story. They provide an example for which they analyze: "...without the real story we could not know what it (the solution) looks like and how to do it"[16]. Wang et al. [23] analyzed that agile methods still need methods from requirements engineering like use cases, user-stories and process models. Most important is the expression of stakeholders' perspectives as stories [12]. Domain-specific-languages might help to find a notation that can be used for specifying structured stories in such a way that stakeholders can write them themselves. Additionally, those stories could be the basis for further transformations that deliver behavioral models.

5 Summary

The role of stories and their different interpretations was discussed. It was shown how actors, motivational stories, agile stories, use-case stories, use-case slices

and behavior models are related and can be combined. They represent different views of stakeholders. Additionally, it was demonstrated how stories and slices can be represented in behavioral specifications like state-machine models and task models.

References

1. Bittner, K., Spence, I.: Use Case Modeling. Addison-Wesley (2002)
2. Cockburn, A.: Writing Effective Use Cases. Addison-Wesley (2000)
3. Denning, S.: The Leader's Guide to Storytelling: Mastering the Art and Discipline of Business Narrative. Jossey-Bass a Wiley Imprint (2011)
4. Diaper, D.: Task Analysis for Human-Computer Interaction. Prentice Hall PTR, Upper Saddle River, NJ, USA (1990)
5. Dittmar, A., Forbrig, P.: Integrating personas and use case models. accepted for INTERACT 2019 (2019)
6. Fleischmann, A.: S-BPM ONE – Setting the Stage for Subject-Oriented Business Process Management, Communications in Computer and Information Science, vol. 85, chap. What Is S-BPM?, pp. 85–106. Springer, Berlin, Heidelberg (2010)
7. Fleischmann, A., Kannengiesser, U., Schmidt, W., Stary, C.: Subject-oriented modeling and execution of multi-agent business processes. In: 2013 IEEE/WIC/ACM International Conferences on Intelligent Agent Technology, IAT 2013, 17-20 November 2013, Atlanta, Georgia, USA. pp. 138–145 (2013). https://doi.org/10.1109/WI-IAT.2013.102, https://doi.org/10.1109/WI-IAT.2013.102
8. Fleischmann, A., Schmidt, W., Stary, C., Augl, M.: Agiles prozessmanagement mittels subjektorientierung. In: HMD Praxis der Wirtschaftsinformatik. 2, vol. 50, p. 64–76 (2013). https://doi.org/https://doi.org/10.1007/BF03340797
9. Fog, K., Christian Budtz and, P.M., Blanchette, S.: Storytelling: Branding in Practice, chap. Storytelling as a Management Tool. In: Storytelling., pp. 131–160. Springer Verlag (2010)
10. Forbrig, P.: Bizdevops and the role of S-BPM. In: S-BPM ONE. pp. 1:1–1:8. ACM (2018)
11. Forbrig, P., Dittmar, A., Kühn, M.: A textual domain specific language for task models: Generating code for cotal, ctte, and HAMSTERS. In: Proceedings of the ACM SIGCHI Symposium on Engineering Interactive Computing Systems, EICS 2018, Paris, France, June 19-22, 2018. pp. 5:1–5:6. ACM (2018). https://doi.org/10.1145/3220134.3225217, https://doi.org/10.1145/3220134.3225217
12. Hudson, W.: User stories don't help users: Introducing persona stories. Interactions **20**(6), 50–53 (Nov 2013). https://doi.org/10.1145/2517668, http://doi.acm.org/10.1145/2517668
13. Jacobson, I., Christerson, M., Jonsson, P., Övergaard, G.: Object-Oriented Software Engineering: A Use Case Driven Approach. Addison-Wesley (1992)
14. Jacobson, I., Spence, I., Bittner, K.: The guide to succeeding with use cases (2011), https://www.ivarjacobson.com/sites/default/files/field_iji_file/article/use-case_2_0_jan11.pdf, last visited March 7, 2019
15. Jacobson, I., Spence, I., Kerr, B.: Use-case 2.0. Commun. ACM **59**(5), 61–69 (2016)

16. Khanh, N.T., Daengdej, J., Arifin, H.H.: Human stories: A new written technique in agile software requirements. In: Proceedings of the 6th International Conference on Software and Computer Applications. pp. 15–22. ICSCA '17, ACM, New York, NY, USA (2017). https://doi.org/10.1145/3056662.3056680, http://doi.acm.org/10.1145/3056662.3056680
17. Lawrence, R.: How to split a user story (2009), http://agileforall.com/resources/how-to-split-a-user-story/, last visited January 20, 2019
18. Quesenbery, W., Brooks, K.: Storytelling for User Experience: Crafting Stories for Better Design. Rosenfeld Media (2011)
19. Rosson, M.B., Carroll, J.M.: The human-computer interaction handbook. chap. Scenario-based Design, pp. 1032–1050. L. Erlbaum Associates Inc., Hillsdale, NJ, USA (2003), http://dl.acm.org/citation.cfm?id=772072.772137
20. Simões, D., Antunes, P., Carriço, L.: Eliciting and modeling business process stories. Business & Information Systems Engineering **60**(2), 115–132 (2018)
21. Thier, K.: Storytelling. Springer Verlag, Berlin Heidelberg, New York (2017)
22. Thier, K.: Storytelling mit der 3-Akt-Struktur: Wie Sie mit der 3-Akt-Struktur authentische Geschichten erzählen und Kunden sowie Mitarbeiter binden - der Leitfaden (Quick Guide). Springer Verlag, Berlin Heidelberg, New York (2017)
23. Wang, X., Zhao, L., Wang, Y., Sun, J.: The role of requirements engineering practices in agile development: An empirical study. In: Zowghi, D., Jin, Z. (eds.) Requirements Engineering. Communications in Computer and Information Science, vol. 432, pp. 195–209. Springer, Berlin, Heidelberg (2014)