

Use Cases, User Stories and BizDevOps

Peter Forbrig

University of Rostock, Chair in Software Engineering,
Albert-Einstein-Str. 22, 18055 Rostock
Peter.forbrig@uni-rostock.de

Abstract. DevOps is currently discussed a lot in computer science communities. BizDev (business development) is only mentioned once in a computer science paper in connection with Continuous Software Engineering. Otherwise it is used in the domain of business administration only. Additionally, there seems to be a different interpretation of the term in the two communities.

The paper discusses the different interpretations found in the literature. Additionally, the idea of BizDevOps is discussed in conjunction with further ideas of taking new requirements for software features into account. These requirements can be described by models on different level of granularity. Starting points are use cases, use-case slices, user stories, and scenarios. They have to be communicated between stakeholders. It is argued in this paper that storytelling can be a solution for that. It is used in as well in software development as in management.

Keywords: BizDev, DevOps, BizDevOps, Continuous Requirements Engineering, Continuous Software Engineering, Agile Software Development, Storytelling.

1 Introduction

Developing Businesses is often related with the development of software because nowadays business processes have to be supported by IT. Continuous requirements engineering has to be performed to provide continuously the optimal support. Continuous business development seems to be a good description for the combined development of software and business. This term is not used so very often. Nevertheless, it exists like in [17].

However, more often the abbreviation BizDev (business development) is used. It seems to be well known in the business administration community but not in the computer science community. A current search (January 20, 2018) for BizDev in the ACM digital library (dl.acm.org) delivered one result only. The result refers to the paper of Fitzgerald and Stol [7] about continuous software engineering. It provides a framework Continuous* that specifies continuously performed activities of software engineering. DevOps (development and operation) plays a central role in this framework.

Copyright 2018 for this paper by its authors. Copying permitted for private and academic purposes.

“DevOps is a set of practices intended to reduce the time between committing a change to a system and the change being placed into normal production, while ensuring high quality” [1]. It aims at shortening the development cycles from months to days or even hours. This is only possible with a corresponding quality insurance by a software tool chain in an agile way.

For many software development projects the combination of BizDev and DevOps to BizDevOps seems to be useful. The paper discusses the idea of BizDevOps and the consequences for requirements specifications. It evaluates different known notations requirements specification notations for the usage in a BizDevOps context. In the following paragraph, BizDevOps will be discussed in more detail. Additionally, the most important requirements specification techniques are shortly revisited. Their usage from different perspectives the relation between these perspectives will be highlighted. Storytelling seems to be an interesting technique to communicate requirements. The paragraph closes with a discussion. Finally, a summary and an outlook are presented.

2 BizDevOps

2.1 Introduction to BizDevOps

BizDev or sometimes its abbreviation BD is a term coming from the business administration domain. In [6] Andrew Dumont characterizes it as: “in its simplest form, BD can be described as connecting similar businesses to similar goals”. He mentions later that BD can be characterized as filtering and building. Filtering refers to the selection activity of the right business partners and the right deals. Building can be related to an IT infrastructure and again finding the right partners for that.

James Cohane mentions that a lot of “Sales” people refer themselves to “Business Development” [4]. He mentions that business development is more than sales. He provides the following definition instead: “Business Development is the function at the company responsible for identifying, securing, and/or managing relationships with organizations outside of the company (excluding customers and suppliers) that helps other key functions at the company achieve their respective goals.”

A nice visualization of BizDevOps is provided by Baumgartner [2]. It is shown in Figure 1.

For the above discussed definitions, aspects of software development play a minor role. IT aspects are covered but not outstandingly important. One can even say that they play a minor role. However, different perspectives exist. Fitzgerald and Stol [7] consider the cooperation of people from business and software development as the most important aspect of BizDev. “We argue that a closer and continuous linkage between business and software development functions is important, and term this BizDev, a phenomenon which complements the DevOps one of integrating more closely the software development and operations functions”.

From our point of view, this is an important aspect. We suggested some modifications to the framework of Fitzgerald and Stol in [7] by continuous requirements engi-

neering, continuous requirements modelling and continuous human centered design. Additionally, we mentioned the idea of using S-BPM [9] for BizDevOps in [10].

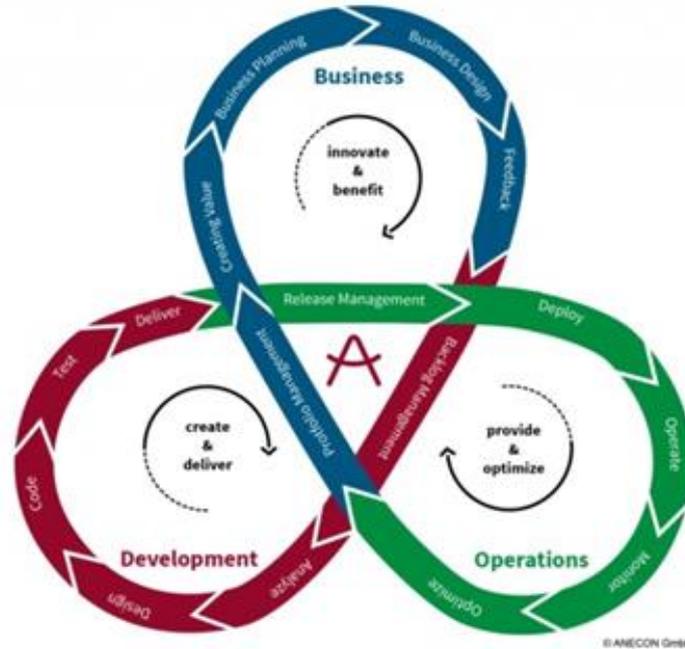


Fig. 1. Visualization of the domain of BizDevOps ([2]).

Already in 2015, Gruhn and Schäfer [11] discussed the fact that DevOps is not the end of the story. “The BizDevOps approach addresses the boundary between the two distinct disciplines: it aims at redistributing responsibilities between IT (who are professionals in rendering stable and reliable IT systems) and business departments (who understand the rationale of IT systems from business perspective)” [11]. They provide a lot of arguments that characterize advantages of BizDevOps.

If people from business and software development have to work together, there is still the question how requirements are developed and specified.

2.2 Requirements Specification Notations and Knowledge Representations

Scenarios. A scenario is a sequence of actions. It can be specified on different level of abstractions. It can be a sequence of events only or like in scenario-based design [10] a narrative descriptions of envisioned usage episodes. Such scenarios are employed to guide the development of the system. Additionally, they offer potential users of the system under development a smell of envisaged use experience. Carroll characterizes the user interaction scenario as a sketch of use.

| Hazards of the solution-first approach | How scenario-based design can help |
|--|--|
| Designers want to select a solution approach quickly, which may lead to premature commitment to their first design ideas | Because they are concrete but rough, scenarios support visible progress, but also relax commitment to the ideas expressed in the scenarios |
| Designers attempt to quickly simplify the problem space with external constraints, such as the reuse of familiar solutions | Because they emphasize people and their experiences, scenarios direct attention to the use-appropriateness of design ideas |
| Designers are intent on elaborating their current design proposal, resulting in inadequate analysis of other ideas or alternatives | Because they are evocative and by nature are incomplete, scenarios promote empathy and raise usage questions at many levels |

Fig. 2. Benefits of scenario-based design (from [16])

Rosson et al. [16] characterize scenarios even as stories that will be discussed in the next paragraph.

User Stories. The term “user story” is used in different ways. The first perspective comes from interaction design.

“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” [15]. Whitney Quesenbery and Kevin Brooks [15] mention 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.

User stories have a structure like beginning part, middle part and ending part. They put important information into a nice context. This makes things more interesting and improves the engagement of participants in a software development project. They can describe a problem, focus on the context of an application, explore design decisions and describe the consequences of new design.

Technically, user stories can be written or spoken. Alternatively, comics, animations, or videos are possible.

We already mentioned the scenario-based approach from Mary Beth Rosson and John Carroll [16]. It is based on descriptions of the use of the envisioned system.

They characterize the nature of their scenarios as follows: “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” [16].

Based on this characterization and on the provided examples scenarios and stories can be considered as synonyms.

Those stories describe the main problems and possible solutions. In this way discussions can be focused on specific aspects and a common ground between stakeholders can be established.

Rosson and Carroll distinguish between problem scenarios, activity scenarios, information scenarios, and interaction scenarios.

Fig. 3 provides a visualization of their scenario-based design (SBD) framework.

During the process of software development the scenarios are continuously refined, extended, or rewritten. They are a very important artefact for software development because they are one of the sources for requirements.

More details can be found in [3].

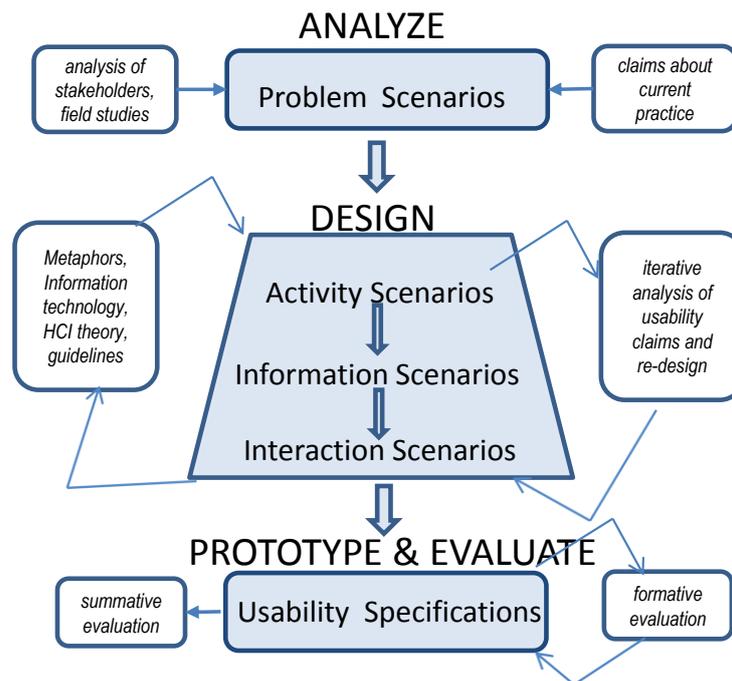


Fig. 3. Scenario-based design framework according to [16].

In the context of agile software development, user stories are totally different. They consist of one sentence only and are written on index cards. Such sentences can have the following structure:

- As a <role>, I want <feature> so that <reason>.

Examples of corresponding user stories could look like the following two stories

1. As a user, I want to upload messages so that I can share them with others.
2. As an administrator, I want to approve messages before they are posted so that I can make sure they are appropriate.

While the first kind of stories provides motivation and a lot of context information, the second kind of stories is focused of functionality. It seems to be that currently storytelling is considered to be a successful technique for business management. We will come back to this aspect in one of the next paragraphs.

Use Cases. Use cases were introduced by Ivar Jacobson [12]. They are a generalized description of a set of interaction sequences between a system and actors. An actor can be a user or a system. Use cases can be written in unstructured text. However, more often they conform to a structured template containing title, goal, primary actor, level, precondition, main success scenario, alternatives, extensions etc.

As overview a use case diagram can be drawn. It provides an abstraction of the written specification by presenting several use cases, their relations, and actors. For details, the textual specification has to be consulted always.

Use Case Slice. Use cases were introduced several years ago. During that time, agile development methods did not play an important role. For agile software development, use cases were considered to be too heavy or complex. Therefore, Use Case 2.0 has the concept of use case slices. A use case can be considered as a set of stories, which are the main success scenario and several alternative scenarios.

A use case slice consists of one or two of those scenarios. These scenarios are called also stories. However, they are not stories in the sense of the above discussed stories in interaction design or the one sentences used in agile methods like SCRUM. They are a sequence of action that is neither a real story nor can be put into one sentence.

A use case slice is a collection of one or more of such stories:

“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” [13].

Fig. 4. Use case slices and their stories (scenarios) visualizes a symbolic use case, its stories, and the grouping of stories to slices.

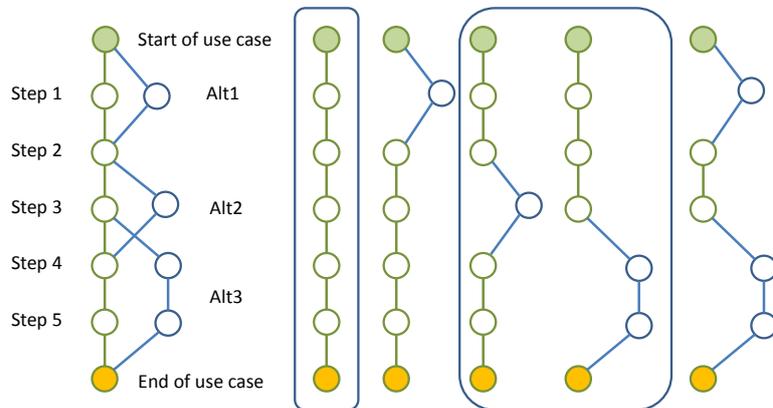


Fig. 4. Use case slices and their stories (scenarios)

On the left hand side of **Fig. 4** one can see a complete use case with its alternatives. The straight line represents the main success scenario. On the right hand side specific scenarios are shown. They represent one specific path through the use case. Two groupings represent two use case slices. The rest of the stories might be grouped to slices later. The life cycle of use cases and use case slices according to Jacobson et al. is shown in **Fig. 5**.

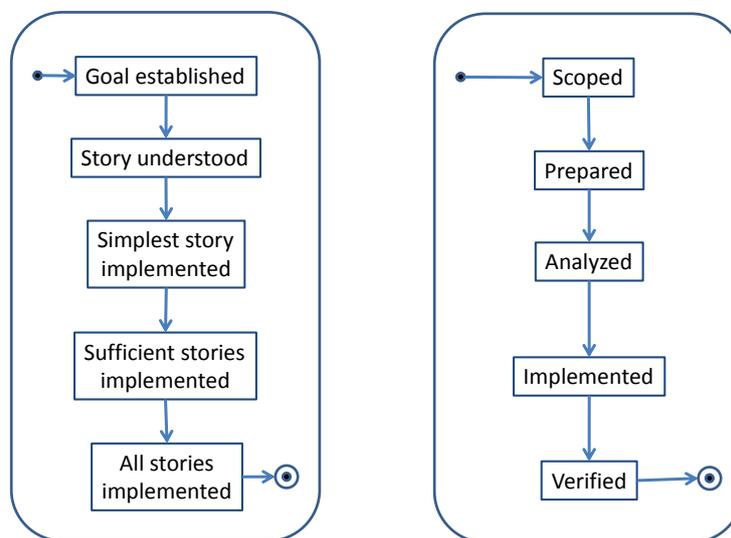


Fig. 5. Life cycle of use case (left hand side) and use case slice (right hand side) implementation (according to [11])

In their Fig. 9 “Use-Case 2.0 Work Products” Jacobson et al. [11] mention the concept of a story without explaining it in detail. One can read from the diagram that use cases are explored by telling stories. Additionally, they are assigned to use-case slices. The story itself is influenced by flows, special requirements and system wide requirements.

Storytelling. The social activity of sharing stories is called storytelling. As we already discussed in connection with scenario-based design, stories are shared as a means of education. However, stories are also used in business management. They help managers solving conflicts, addressing issues and facing challenges. Fog et al. published a paper about “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. They help shape our perception of “who we are” and “what we stand for”. Likewise, stories are told and flow through all companies”.

This is supported by Dennings [5] remark: “Storytelling works as a supplement to traditional management tools. For manager, the task is to use storytelling to anchor the company's values, visions, and culture within the organization.”

Denning presented in [5] eight different story patterns:

- Sparking action
- Communicating who you are
- Transmitting values
- Communicating your brand
- Fostering collaboration
- Taming the grapevine
- Sharing knowledge
- 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.

Karin Their published a book [18] about success examples of applying stories in different areas of business administration. The title of her book is “Storytelling: A Method for Change-, Brand-, Project- and Knowledge Management”. She provides a process model for developing stories. It presented by **Fig. 6**.

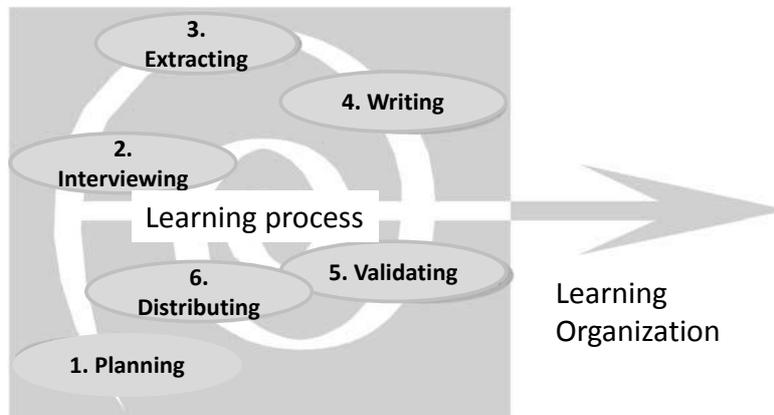


Fig. 6. Storytelling process (adopted from [18])

According to the provided model organizations should organize their knowledge management by including stories into the knowledge base.

2.3 Discussion

In the previous paragraphs different notations for requirements specifications and knowledge representations were presented. From the idea of continuous software engineering and BizDevOps arises the need of communication between stakeholders from business and from software development.

Stories seem to be a tool that is used in both communities. Storytelling has advantages as well for interaction design as business management. It is used in a minimized way in agile development and can support continuous requirements engineering by constantly updating the stories.

The provided story patterns by Denning were introduced and used for business management purposes only. They were not intended to be used for software development. However, the patterns “sharing knowledge” and “leading people” into the future can be adapted for this purpose. In combination with the experience in scenario-based design the tool of stories should be very helpful for BizDevOps.

Lawrence [14] provides nine patterns for spitting user stories (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. However, currently those patterns are mainly for the one sentences stories. Nevertheless, they might be a good starting point for further research.

3 Summary

The ideas of BizDevOps were discussed based on their roots in BizDev and DevOps. The main purpose of those methods is the cooperative work of business, development, and operations people. Consequences of monitoring running systems have to be considered for consequences for business strategies and processes.

Business processes have to be supported by software systems. Requirements for those systems have to be specified in such a way that business people are able to write them or at least they have to be able to read those documents.

Scenarios, user stories, use cases, and use case slices were discussed. Differences and similarities were shown. Additionally, the potentials as communication means were evaluated. It seems to be that stories are an appropriate tool for communication between business, development, and operation people. Furthermore, they can be used for management purposes to motivate people.

Refinements of stories to more precise specifications like use cases, use-case slices, tasks, objects, and user stories for backlogs are possible. This can be done based on existing patterns. However, it might be possible to identify further patterns

4 References

1. Bass, L., Weber, I., and Zhu, L.: DevOps: A Software Architect's Perspective. ISBN 978-0134049847.
2. Baumgartner, M.: Quality Leadership Circle: DevOps - ein Zukunftsbild mit Erfolgsgarantie?, <http://www.anecon.com/blog/qlc2-devops/> from 2016/06/14. Last visited January 5, 2018.
3. Carroll, J.M.: Making Use: Scenario-Based Design of Human-Computer Interactions. MIT Press, Cambridge (2000)
4. Cohane, J.: WTF is Biz Dev? (aka What is "Business Development"?), <http://jamescohan.com/wtf-is-biz-dev-aka-what-is-business-development/>, last visited January 20, 2018.
5. Denning, S.: The Leader's Guide to Storytelling: Mastering the Art and Discipline of Business Narrative, Revised and Updated, John Wiley, 2011.
6. Dumont, A.: Role of Business Development at a Startup, April, 2014, <http://andrewdumont.me/role-of-business-development-at-a-startups/> last visited January 20th 2018.
7. Fitzgerald, B. and Stol, K.-J.: Continuous software engineering: A roadmap and agenda, Journal of Systems and Software, Volume 25, July 2015, Pages 1–14.
8. Fog K., Budtz C., Munch P., and Blanchette S.: Storytelling as a Management Tool. In: Storytelling. Springer, Berlin, Heidelberg, https://doi.org/10.1007/978-3-540-88349-4_6, 2010.
9. Forbrig, P.: Continuous Software Engineering with Special Emphasis on Continuous Business-Process Modeling and Human-Centered Design, In Proc. S-BPM ONE 2016.
10. Forbrig, P.: Does Continuous Requirements Engineering need Continuous Software Engineering?, CRE'17, Workshop at REFSQ, Essen, 2017, <http://ceur-ws.org/Vol-1796/cre-paper>, last visited January 20, 2018.

11. Gruhn, V., and Schäfer, c.: "BizDevOps: Because DevOps is Not the End of the Story," in Proceedings of the 14th International Conference on Intelligent Software Methodologies, Tools and Techniques (SoMet 2015), Communications in Computer and Information Science, Vol 532. Ed. By H. Fujita and G. Guizzi. Springer, 2015, pp. 388-398.
12. Jacobson, I., Christerson, M., and Jonsson, P.: Object-Oriented Software Engineering - A Use Case Driven Approach, Addison-Wesley, 1992, ISBN 0-2015-4435-0
13. Jacobson, I.; Spence, I., and Bittner, K.: USE-CASE 2.0: The Guide to Succeeding with Use Cases, https://www.ivarjacobson.com/sites/default/files/field_iji_file/article/use-case_2_0_jan11.pdf, visited last time January 15, 2018.
14. Lawrence, R: How to Split a User Story, <http://agileforall.com/resources/how-to-split-a-user-story/>, last visited January 20, 2018.
15. Quesenbery, W., and Brooks, K: Storytelling for User Experience: Crafting Stories for Better Design, Rosenfeld Media, 2011, ISBN-13: 978-1933820477.
16. Rosson, B., and Carroll, J. M.: Scenario-Based Design, Chapter 53 in J. Jacko & A. Sears (Eds.), The Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies and Emerging Applications. Lawrence Erlbaum Associates, 2002, pp. 1032-1050.
17. The Guardian: keeping professional development continuous, <https://www.theguardian.com/careers/careers-blog/keeping-professional-development-continuous>
18. Thier, K.: Storytelling: Eine Methode für das Change-, Marken-, Projekt- und Wissensmanagement, 3. Auflage, Springer 2016.