Agile Business Engineering: From Transformation Towards Continuous Innovation

Barbara Steffen(✉), Falk Howar, Tim Tegeler, and Bernhard Steffen

TU Dortmund University, Dortmund, Germany
{barbara.steffen,falk.howar,tim.tegeler}@tu-dortmund.de,
steffen@cs.tu-dortmund.de

Abstract. We discuss how to overcome the often fatal impact of violating integral quality constraints: seemingly successful (software) development projects turn into failures because of a mismatch with the business context. We investigate the similarities and differences between the today popular DevOps scenarios for aligning development and operations and the more general alignment problem concerning software and business engineering based on 33 structured expert interviews. It appears that both scenarios are driven by creativity in a continuous collaboration process relying on continuous goal validation. On the other hand, differences appear when considering Thorngate's trade-off between accuracy, generality and simplicity: the different level of accuracy is the main hurdle for transferring the automation-driven DevOps technology. The paper closes with the hypothesis that this hurdle may be overcome by increasing the accuracy within the business context using domain-specific languages, a hypothesis supported by the interviews that now needs further confirmation via case studies.

Keywords: Agile business engineering · Software engineering · Integral quality constraint · DevOps · Domain-specific languages

1 Introduction

Today, organizations are under continuous pressure of their industry's evolution to survive the natural selection in a world of everchanging customer preferences, new technologies and competitors' developments and offers [21]. This selection is won by the organizations most responsive to or even driving the change. The Red Queen effect underlines the challenge: "[...] it takes all the running you can do, to keep in the same place" (p.2 [10]) [4]. Thus, organizations have to continuously track the external developments to initiate the according iteration and transformation measures internally.

Daepp et al. found that independent from the industry an organization's half-life is only roughly a decade based on a sample of 25.000 publicly traded organizations in North America [12]. In Christensen's opinion the worst enemy of industry leaders are the disruptive innovations/technologies they do not take seriously

T. Margaria and B. Steffen (Eds.): ISoLA 2021, LNCS 13036, pp. 77–94, 2021.
https://doi.org/10.1007/978-3-030-89159-6_6

enough due to an initial underperformance of their offers, but that still have the power to drive them obsolete [11]. This trend amplified with the fourth revolution including Industry 4.0 and digitalization shortens the innovation and change cycles dramatically with the effect that traditionally successful big bang[1] transformations are increasingly replaced by continuous change/transformation [1].

There are two strategies to tackle this continuous request for change. First, organizations can develop dynamic capabilities to ensure the organization's survival in the long run [26,33]: The more dynamic an organization is, the better it can adopt new technologies and thus adapt to new trends. Second, organizations may decide to go beyond this 'reactive' approach and to attack by driving the industry's change via designing radical innovations continuously challenging their own offers or via applying the blue ocean strategy of creating entirely new markets/customer segments [25].

Thus, the holy grail of surviving is becoming an ambidextrous organization and to simultaneously exploit current technologies and offers via further incremental innovations while, at the same time, exploring new paths via radical innovations and disruptive technologies [33]. In fact, Ries postulates that any (established) organization should have an entrepreneurship department in order to ensure that exploration receives the needed attention [34].

Invasive changes and innovations require cross-departmental collaboration to ensure the solution's fit. Due to different backgrounds, experiences, and set performance targets these collaborations face diverging agendas and semantic gaps complicating the smooth and aligned understanding and collaboration [8,28].

A great example of the status-quo and business engineering's shortcomings is Bosch's lawnmower Indigo Connect for roughly € 1200. Bosch is known for its high quality products. Thus, it was not surprising that the lawnmower's advertisement stated easy, live and remote controllability via smartphone. However, Keese summarized his experience as spending 3,5 days on his knees to install the boundary wires in his garden and a fight with an app that was never up to date [23].

How can it happen that the product's marketing promises diverge so much from the customer experience? Concerning the app performance, the answer is easy: Engineering developed, tested and pitched a lawnmower with two (identical) chips, one for driving autonomously and one for tracing and sending the position. The marketing campaign was based on this experience and promised an 'active and live control via app'. Controlling, on the other hand, considered the two chips as too expensive which led to the final product only having one chip. Together with the decision for a reduced data line (again a cost factor) this caused a totally unacceptable app performance, in particular, for a high-end product of a market leader [23].

The problem was the silo structure of today's market leaders with completely different competencies, objectives and metrics [19,36]. This silo structure supports local (department-centric) optimizations that are all too often in conflict with organization's global interest [23].

[1] https://airbrake.io/blog/sdlc/big-bang-model (last access 15th June 2021).

Mismatches like this are even greater in the business engineering (BE) and software engineering (SE) context where the integral quality constraints (that the product must adhere to the existing (IT) infrastructure, process and products) are much less tangible [38]. It is therefore a major challenge to motivate all involved stakeholders to support the changes/solutions, and to align and adapt the different objectives, requirements, and preferences [36]. This requires continuous communication and cooperation between the different stakeholders in order to establish a common understanding and vision by doing to reach the status of a scalable agile organization [19,27].

As stated in a Fraunhofer report (2021) [2], organizations need to understand software as enabler of business engineering and not just as internal business process support or add-on functionality of hardware devices.

The research presented in this paper is motivated by two observations:

- The integration problems due to misalignment between the information systems (IS) development methods and the business development context that may cause failure of seemingly successful SE projects [13].
- The success story of DevOps for aligning SE development (Dev) methods and outputs with the requirements of operations (Ops) [3,5]

These observations lead to two research questions:

1. What are the essential similarities and differences between the BE/SE and the Dev/Ops scenarios?
2. Is it possible to transfer some of the DevOps techniques to the BE/SE scenario?

The paper's structure is as follows: After the introduction, Sect. 2 outlines the foundations, parallels and differences of BE and SE. Section 3 details the methodology. Section 4 summarizes the interview study results. In Sect. 5 we derive the implications and answer the research questions before reflecting on the results in Sect. 6. This paper finalizes in Sect. 7 with a conclusion, limitations and outlook.

2 State of the Art

We briefly compare concepts, terminology, and methods of the BE and SE disciplines by summarizing the state of the art in both fields as found in the literature. We then make some initial observations on parallels, key differences, and potentials for alignment in Sect. 2.3. The observations provide a basic conceptual framework for the empirical study.

2.1 Business Engineering

In this paper we use Österle's definition of Business Engineering from 1995 [41]. It focuses on adapting and transforming the business in accordance to internal and external developments and is divided into optimization and development driven changes. These transformations are ideally implemented in a structured top-down fashion focusing on three specifications: strategy, organization and IT system [40,42]: From the business strategy encompassing the organization's strategy and the derived goals to the organization's business processes and finally to the definition of the corresponding IT system support and implementation. This rather engineering (also referred to as plan-driven) process is suggested to handle the transformation's complexity and interdisciplinary collaboration and alignment to ensure that all relevant aspects are considered in the right order.

However, since the continuous introduction of technology innovations businesses face the pressure to constantly observe and react to the industry's dynamics leading to rather invasive and radical innovations and adaptations of current internal processes and/or business models [1,6]. In these challenging settings of high uncertainty regarding the project's business execution and business development method this plan-driven approach does not suffice. Thus, organizations need to embrace change-driven project execution.

Many organizations already try to leverage agile methods like design thinking, scrum, SAFe etc. to reduce the uncertainty and adapt implementation measures and processes based on new learnings [9,27]. This focus on collaboration and regular meetings addresses and reduces the semantic gap (e.g. misunderstandings due to different backgrounds and experiences) and supports alignment and buy-in along the process. In practice this leads to better and holistic outputs but is a very time-consuming process to derive at acceptable compromises.

Nevertheless, Dahlberg & Lagstedt observed that even successful (Information Systems (IS)) projects that benefited from the necessary competencies and a well-defined plan may never be successfully integrated and used in the business environment [13]. From the business perspective the reason for this failure boils down to a violation of the integral quality constraint [38]: great and well functioning products and/or solutions do not fit the needs of the actual business development context in which they shall be integrated. Unfortunately, this mismatch often just becomes visible after the product is finalized and ready to be integrated.

The risk for failure increases with the scope of change and the underlying uncertainty: In engineering it is still comparatively simple to detail the machine specifications and to ensure its fit into the production line. It becomes more complicated with increasing degrees of freedom.

Software projects are known for their high degrees of freedom (see Sect. 2.2) as are invasive business changes: Both often depend on many parameters that are typically hardly constrained by something like physical laws. E.g. new business models and internal processes typically depend on cross-departmental collaboration and alignment and therefore on individuals with their specific character and their willingness to cooperate [35]. In such complex scenarios it is virtually impossible to sufficiently predict the solution's and the business context integration's requirements upfront. Rather a flexible approach is required that allows to react to arising challenges. To summarize, today's businesses face major uncertainties and thus need to embrace an internal continuous improvement approach.

This requires ongoing learning by doing in interdisciplinary teams via developing creative ideas and innovations which are continuously tested and validated.

2.2 Agile Software Engineering

In the 1990 s, Agile Software Engineering (ASE) arose as a response to two decades of failing waterfall-oriented software development projects, which had aimed at controlling risks via detailed contract specifications [7,30]. The observation that software projects are very hard to specify upfront because customers are typically unable to express their wishes in sufficient detail for experts to decide on adequate implementations was central to the paradigm shift. This problem, also known as the semantic gap, led to the **Agile Software Engineering manifesto** [17] that postulated the following four key insights:

1. Individuals and interactions over processes and tools
2. Working software over comprehensive documentation
3. Customer collaboration over contract negotiation
4. Responding to change over following a plan

The manifesto considers software system development[2] as a mutual learning process in which customers and developers converge towards a mutual understanding. Key to convergence is the incremental development style in which partial products serve as unambiguous means for common design decisions.

DevOps [3,5] complements ASE in this line by (semi-) automatically supporting partial product construction, management, and validation. More concretely, experts of operations are involved to bridge the gap between the logical design (e.g. the program) and the product running on some complex physical infrastructure. This comprises:

1. **Construction:** Version-controlled development supporting roll back and merge.
2. **Management:** Continuous version-based documentation in 'one shared repository' style where essential dependencies and design decisions are maintained in a combined fashion.
3. **Cooperation:** Dedicated domain-specific languages (DSLs) supporting the Dev/Ops cooperation.
4. **Validation:** Automated test environments enabling continuous validation via so-called CI/CD (Continuous Integration/Continuous Deployment) Pipelines.

The combination of ASE and DevOps supports an incremental, collaborative development style which continuously maintains running partial products (extremely high-fidelity prototypes) that successively converge towards (successful) products which oftentimes differ quite drastically from the initially conceived product.

[2] Here meant to comprise SE and IS.

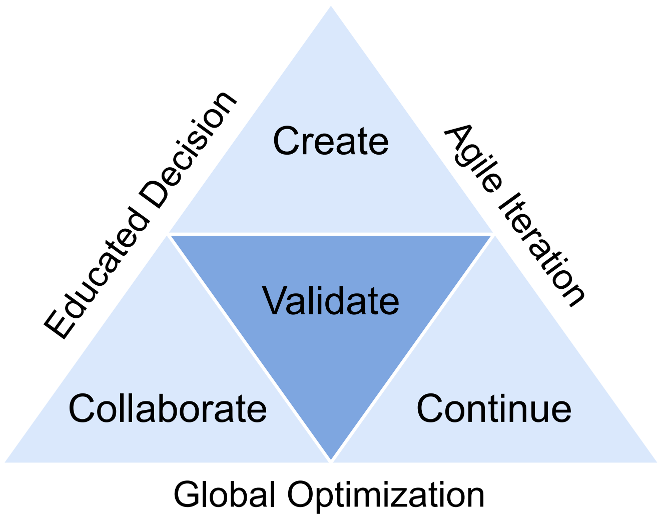

Fig. 1. BE and SE Parallels

The early integration of the operations team does not only lead to better infrastructure for scaling up operations and performance but it allows all stakeholders to experience and test the intended product during its development in its foreseen environment [18].

2.3 Parallels, Differences, and Potentials

As detailed in the previous two sections, BE and SE have a similar problem domain. Both disciplines face the challenges of (a) continuously dealing with change which requires (b) creative solutions that, to be successful, can only be found in (c) interdisciplinary collaboration. Moreover, in particular, due to the semantic gaps there is a strong need to continuously (d) validate the state of the creative collaboration process to detect misconceptions early. Figure 1 sketches the aspects of the problem domain and their interplay.

The labels on the outer sides of the triangle name requirements for solutions pertaining to the involved aspects: methods must enable global optimization through agile iteration, based on educated decisions. Key enabling techniques for the success of this approach concern the continuous validation of the reached achievements according to the strategic goals.

The main conceptual difference between BE and SE can elegantly be characterized by the well-known trade-off between accuracy, generality, and simplicity (see Fig. 2 [37]): by its nature, BE has to drastically simplify its complex highly heterogeneous scenarios, and, due to the high level of inherent uncertainty to aim for generality rather than accuracy. SE, in contrast, addresses software, i.e., descriptions precise enough to run on a computer. Programming languages are in other words generic, in order to allow programmers to potentially solve all

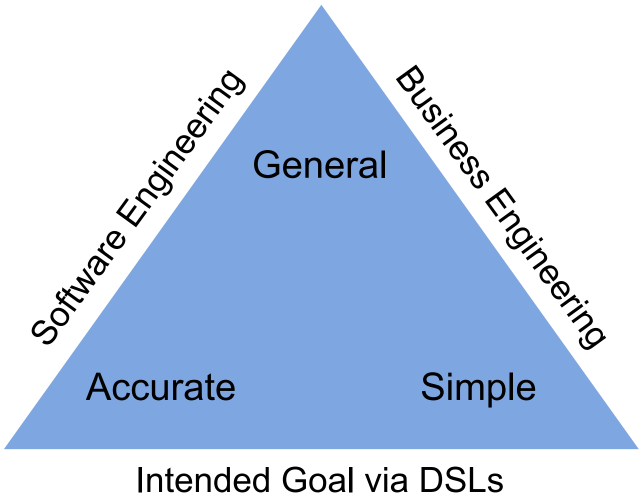

Fig. 2. Based on Thorngate's Trade-off

computable problems. Thus, SE only leaves room for trading between generality and simplicity.

The trade-off between simplicity and generality can be observed between general purpose programming languages and so-called domain-specific languages (DSLs, [16,22]) which are arguably one of the key enablers of DevOps [14].

Key characteristics of DSLs: Whereas programming languages are traditionally universal and consequently intricate, there is an increasing trend towards using (graphical) DSLs that aim at allowing application experts to cooperate in a no/low code style on specific problems. DSLs can be regarded as an ideal means to trade generality against simplicity in application-specific contexts. Together with corresponding Integrated Development Environments (IDEs, [24,29,31]) that typically provide sophisticated development support DSLs have the potential to become adequate alternatives to classical tool support whenever these are conceived to be too restrictive.

In the context of DevOps, this is witnessed by the success of DSLs that provide a dedicated support in particular for configuring IT infrastructure and to develop required CI/CD pipelines in an infrastructure as code style.

In the realm of business engineering, graphical DSLs that are designed on the basis of BE-oriented graphical notations (BPMN, CMMN, ER Diagrams, Organigrams, Canvases (BMC), etc.) may turn out to be good candidates for aligning the BE and ASE/Ops (Dev/Ops) cooperation and to transfer supporting technology for achieving (more) automation.

Our corresponding experience of combining the Business Model Canvas [32] with graphically modelled ontologies as sketched in Fig. 3 was very promising. It allowed us to (semi) automatically derive data structures for organizational structures without writing a single line of code.

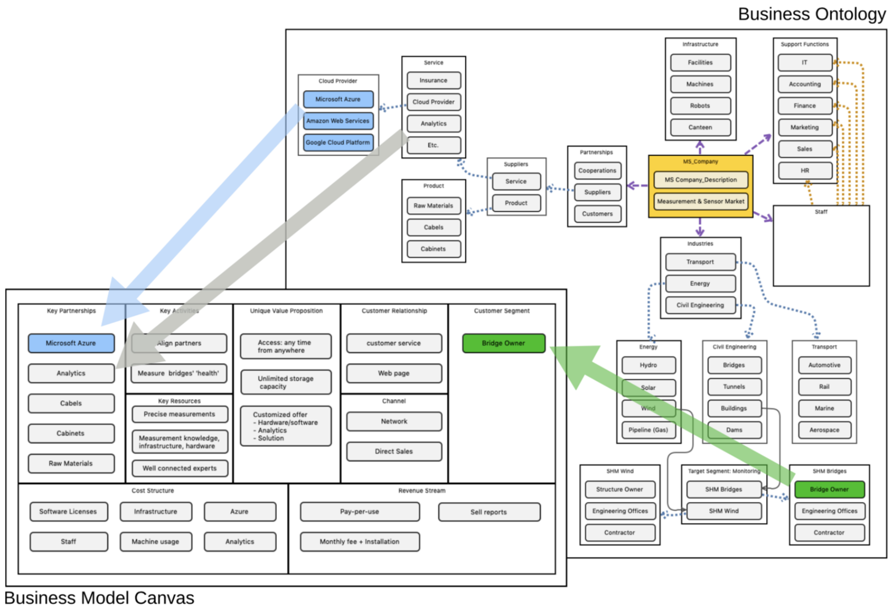

Fig. 3. Graphical DSL-based Enablement of Canvases

3 Method

In this section we define the paper's design method and interview study details. The design method is based on Hevner's three cycle view (see Fig. 4, [20]) which consists of the relevance, design and rigor cycles. In our case this led to a 10-step procedure as also sketched in Fig. 4:

- *Step 1 (Relevance):* The study of this paper was triggered by the observation of Dahlberg and Lagstedt (2021) [13] that the results of successful development projects may nevertheless lead to failure due to problems during the integration into the business context.
- *Step 2 (Design):* Based on this motivation and the corresponding research questions we designed this paper's method.
- *Step 3 & 4 (Rigor):* We reviewed the literature to define and compare BE and SE.
- *Step 5 (Design):* To further detail the parallels and differences between BE and SE we designed a structured interview guideline with closed and open questions.
- *Step 6 (Relevance):* To ensure the applicability of the structured interview guideline a pilot with four interviewees (two with IT and two with a business background) were conducted.
- *Step 7 (Design):* Based on the feedback gathered and issues identified via the pilot interviews we iterated the interview guideline accordingly.

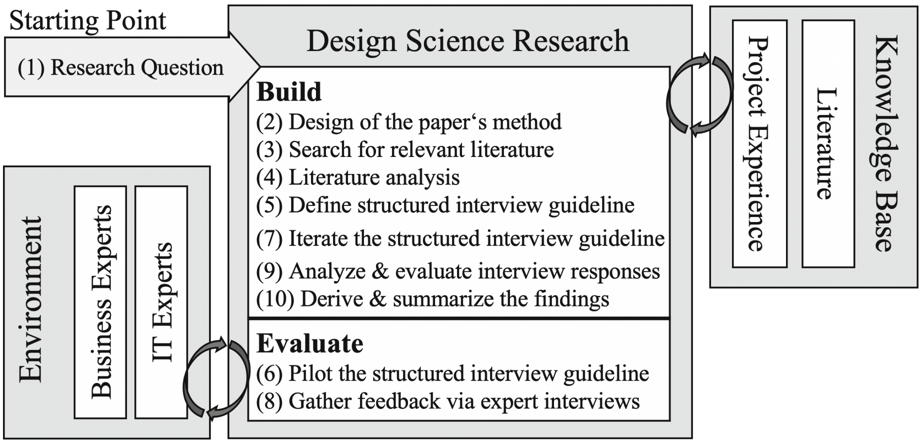
Starting Point
(1) Research Question
Design Science Research
Build
(2) Design of the paper's method
(3) Search for relevant literature
(4) Literature analysis
(5) Define structured interview guideline
(7) Iterate the structured interview guideline
(9) Analyze & evaluate interview responses
(10) Derive & summarize the findings
Evaluate
(6) Pilot the structured interview guideline
(8) Gather feedback via expert interviews
Environment
Business Experts
IT Experts
Knowledge Base
Project Experience
Literature

Fig. 4. Research Method Based on Hevner's Three Cycle View [20]

- *Step 8 (Relevance):* Then we gathered the business and IT expert feedback via 33 highly structured interviews.
- *Step 9 & 10 (Design):* Once all interviews were conducted, we analyzed and evaluated the responses to derive at the paper's final results and findings.

To ensure that the research questions can be answered based on actual experiences and assessments of the status quo and preferred outlooks we decided to conduct expert interviews. We interviewed a total of 33 interviewees:

- *7 business experts:* four senior consultants, two entrepreneurs, and one assistant to the board.
- *9 mixed business & IT experts:* four members of the board, one founder, three professors (who also led or worked in organizations), and one team leader with dedicated responsibility for digitalization. This group is especially important as they can compare the differences between BE and SE and their corresponding mind-set and tool-support firsthand.
- *10 IT experts:* four with dedicated DevOps experience and six with dedicated DSL experience four of which working as team leaders.
- *7 IT students:* all are almost finished with their masters, have experience with DSL application and participated in interdisciplinary IT projects.

Due to SARS-CoV-2 we executed the highly structured interviews via telephone and online (e.g. via Zoom). Each interview took roughly 45 to 60 minutes depending on the interviewees' level of detail. The interviews comprised a total of 57 questions and four additional questions regarding the general background. The 57 questions consisted of 25 quantitative (20x Likert-scale based response options from 1–5 and 5x multiple choice questions) and 32 qualitative (open questions) response options. The questions covered the interviewees' experiences and opinions on the following topics: confrontation with changes on the job, transformation vs. continuous improvement focus, agility of the work environment, regression potentials, validation methods and processes, (interdisciplinary) collaboration and knowledge management. Open questions were e.g., 'How do you analyze and manage risks?', 'Which tools do you use to support transformation/change?', What is the biggest challenge to enable agility' and 'How do you measure and analyze success?'. Examples of closed questions are shown in Fig. 5. We chose this mix to simultaneously allow for a direct and easy comparison/assessment of responses while leaving sufficient room for differences and examples to benefit from the advantage of expert interviews e.g. the openness towards new essential input [15]. Our highly structured interview approach ensured comparability between the results because neither interviewees nor respondents could deviate from the pre-defined procedure. All interviews follow the same structure reducing potential information/discussion biases independent of the interviewer. In our case three authors conducted interviews to reduce the interviewer bias: we matched the expertise and background of the respondents with the most similar interviewer to reduce potential semantic barriers and to increase the responses' objectivity [8,15,39].

4 Interview Results

In this section we sketch the results of the interview study. First, we will elaborate on the qualitative responses. Here, the main differences observed concern the understanding and status of the role of tools in BE and SE. Then we will briefly sketch and discuss the eight most relevant outcomes of the quantitative questions (Q1 to Q8) for our conclusion (see Fig. 5).

Via category-coding the named IT tools (used in the interviewees' work contexts) according to their purpose (communication/knowledge, operation, management, success metrics, requirements/validation, modelling, configuration management, and test/quality assurance), we observed that IT professionals frequently named tools (e.g. GitLab, GitHub, and CI/CD pipelines) automating tasks like operations, tests, and configuration management. In comparison, business experts, with two exceptions that also named tools to measure success metrics (e.g. Power BI and OKR Software), only mentioned tool support for communication/knowledge (e.g. MS Teams, Slack, Zoom, MS Office, Wikis, and SharePoint), information systems (e.g. SAP) and requirements/validation (e.g. survey tools and (software) prototypes).

More concretely, technical support for requirements elicitation (including prototypes) is present in both groups, indicating that both groups value early validation and use prototypes for improving the shared understanding. Business experts named significantly more tools for communication as IT professionals, but most answers qualified rather as conceptual frameworks than as tools in the sense of SE.

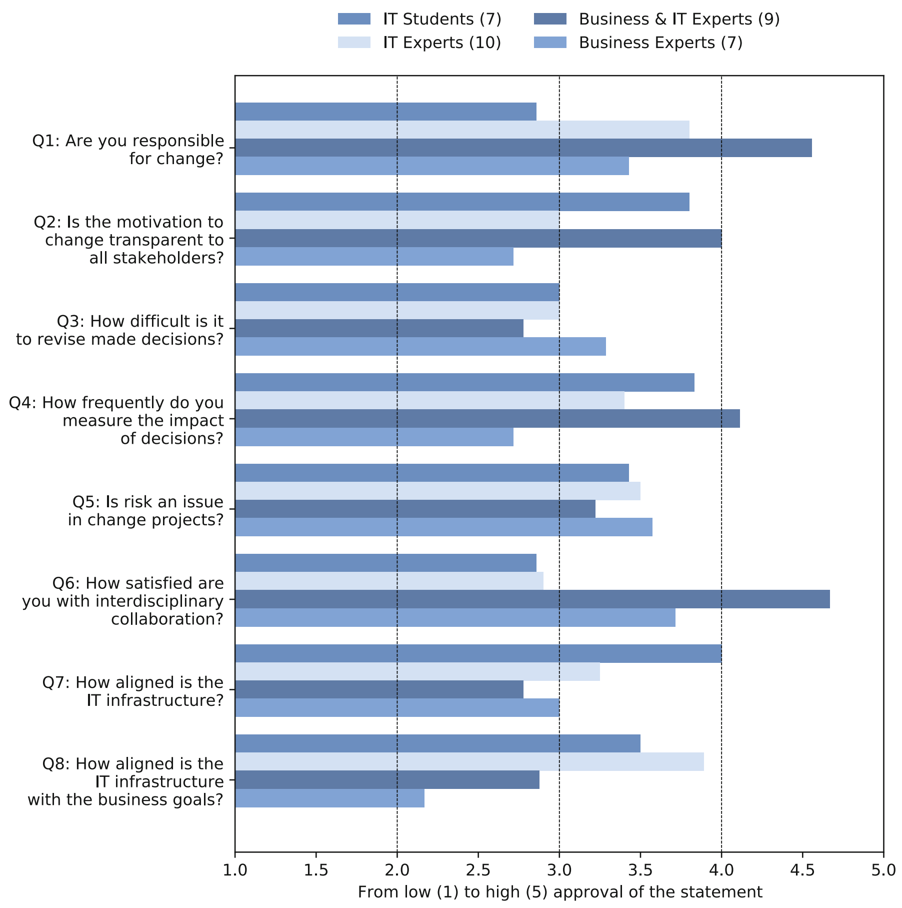

Fig. 5. The Results of the Exemplary Questions (Q1 to Q8)

For the group of professionals that qualify as both, the results were mixed, in some cases closer to the group of IT professionals (e.g. with regard to configuration management) and in some cases closer to the group of business experts (e.g. communication/knowledge). The higher the respective organization's IT core competency the more tools are used for (automated) support.

Interestingly business experts mentioned mostly decision and alignment support in the form of processes, methods and frameworks which currently do not benefit from direct tool support e.g. discussions in meetings, calls and workshops for SWOT, canvas and stakeholder analysis. Here one can see that today business experts handle the business transformation's complexity via meetings and shared documentation.

When asked whether they would prefer additional tools supporting their daily tasks they do not see the need and applicability. One respondent made it clear when he stated that "our daily tasks are too complex and different from project to project to benefit from tool-support. The cost-benefit ratio would not support the development of suitable tools". (We translated this statement from the original language to English.)

It seems that business experts trust people's understanding and complexity matching more than tools. The drawback of this approach is that it builds on intuitive semantics which unfortunately are open to individual and subjective interpretation and thus semantic gaps and diverging understanding. This impairs the overall transparency of the projects and excludes the possibility of automated support, very much in contrast to the accuracy-driven DevOps scenario.

The observed differences are a consequence of more fundamental differences between the BE/SE and the Dev/Ops scenarios: The different degrees of complexity and accuracy whose consequences are also visible in the responses (see Fig. 5) to

- the perceived transparency of the
  motivation for change to all stakeholders (2),
- the satisfaction with the interdisciplinary work (6) and
- the perception of risk in the context of change (5).

The replies to Q2 show the differences between experts who are in charge of their work and projects and those who are dependent on their boss' decisions. The less dependent on others the higher was the transparency rated. 'Business & IT experts' value the benefits and satisfaction of interdisciplinary work particularly high. One could argue that the more often one encounters interdisciplinary work the more one values and gets used to it. Interestingly Q5 shows that all four groups perceive risk as issue when dealing with change. However, not particularly high. The higher frequency of measuring the impact of decisions concerning both the SE and the Ops perspectives as seen in replies to Q4 indicates the impact of DevOps supporting higher degree of automation. As a consequence integral quality is continuously guaranteed in the DevOps scenario which prohibits bad surprises as reported in [13] where the results of a successful IS project never became operational because of a mismatch with the BE development contexts.

Put together with the other responses to the open questions the following picture arises:

- Automation support for configuration management, testing, operation, and continuous delivery (aka the DevOps support) is ubiquitous in SE but virtually absent in BE.
- Software engineers and business engineers have a different concept of what a tool is and likely also a different understanding of the degree of automation that can be achieved and the associated benefits.

– The agile mindset has already permeated all groups to some degree (as indicated by mentions of tools for requirements and early validation).
– Modeling tools (especially for business processes) have some success in BE.

These different perceptions of tools and their potential towards automation is essential for the proper understanding of the following answers to our two research questions.

5 Implications

In the following we will answer the paper's two research questions:

Research Question 1: What are the essential similarities and differences between the BE/SE and the Dev/Ops scenario?

There was a strong overall agreement in response to the qualitative and quantitative questions that both BE/SE and Dev/Ops face the challenge to continuously deal with and manage change. Here, interdisciplinary collaboration is particularly relevant to develop creative solutions like invasive and/or radical innovations. Moreover, as already mentioned in Sect. 2.3 and now confirmed by the interviews there is a strong need to continuously validate the progress of the creative (collaboration) process and the decisions' impact to detect misconceptions and misalignments early and allow for early and effective countermeasures.

On the other hand, the interviewees' responses revealed clear differences between the two scenarios when it comes to the required systematic support of the continuous and creative collaboration process. Particularly striking is the difference when it comes to the role of tools and validation:

1. Whereas in BE/SE there are hardly tools that support more than standard administrative tasks, DevOps is supported by a wealth of tools that (semi) automate most of the CI/CD pipelines comprising documentation, versioning and roll back.
2. Whereas in BE/SE processes typically follow some assumed best practices but are typically not tool supported or automated in any way, DevOps aims at automating the entire build process.
3. Whereas in BE/SE the gap between the SE/IS development methods and BE development context is considered too large to be bridged via standards and tools, DevOps explicitly addresses this gap with corresponding common DSLs in order to support automation.
4. Whereas the tool landscape of BE/SE is typically neither aligned itself (see Q7) nor towards the company goals (see Q8), DevOps is characterized by aligned tool chains.

Research Question 2: Is it possible to transfer some of the DevOps techniques to the BE/SE scenario?

The answers to the qualitative questions revealed that SE experts, in particular those with some experience with (graphical) DSLs, were quite optimistic concerning the transferability of the methods. The main reason mentioned was that DSL-based frameworks are much more flexible than ‘classical’ tools and may therefore proof to be able to bridge the larger gap. In fact, DSLs are also explicitly mentioned as the essential reason for DevOps to overcome the semantic gap between SE and Ops [14].

In fact, one of the nine ‘business & IT’ experts was a team leader of a larger software house where agility principles and tools known from software development and DevOps start to also enter the business level. In this company, organizing even customer presentations and company events like an agile software development project showed automation potential, easier goal adaptation, better prototyping and therefore, in particular, better cross stakeholder communication. These benefits even reached the board level and entered an explicit company-wide agile manifesto.

Certainly, this success story very much depends on the fact that it takes place within a software company, and that the application of software (development) tools is considered standard there. The feedback of the interviewees with experience in applying DSLs in customer projects suggests, however, that, using adequate DSLs, this success can be leveraged in a larger scope.

Our answer to the second research question can therefore be formulated as a hypothesis that cannot be confirmed by interviews but requires further systematic case studies and pilot projects:

Hypothesis for future research:
DSLs can be regarded as an enabler for tool-based automation in BE.

6 Reflections

Stepping back, it appears that compared with SE in particular DevOps the technological state of the BE scenario has certainly reasons in

– its much higher complexity in particular
  concerning the interdisciplinary scope
– its much higher level of variety and uncertainty

which both lead to the mindset that standardization, tools, and automation imply unacceptable restrictions that strongly impair the potential of BE. This explains the poor BE tool landscape and the appreciated value of informal best practice patterns (e.g. continuous improvement cycles and canvases). In particular the observed lack of automation hinders agility, as e.g., prototyping (in the sense of minimum viable products) or validations, e.g., via simulations, are extremely expensive and therefore hardly performed (at least in comparison to SE where, e.g., daily automated builds are kind of standard).

Our **Hypothesis for Future Research** indicates a way that may allow to overcome this situation: DSLs for interdisciplinary communication may provide a level of precision that allows for automation via dedicated generators for providing stakeholder-specific views, executable prototypes, or KPI analyzes. This may, in particular, also help to control the risk. Figure 5 is interesting in this respect: The 'business & IT expert' group which also considered a BE-oriented notion of risk sees fewer problems than the business expert group.

Please note that, in particular, the numbers of the other two groups are misleading in this respect as they were thinking, e.g., of security risks introduced by e.g. third party components (which the 'business & IT expert' group was also aware of), a phenomenon that was not considered by the interviewed business experts.

Reducing the (perceived) risk of a change is of vital importance for an agile organization and increasing the transparency of the impact of changes is a good way to guarantee the acceptance by all stakeholders. Thus, every means supporting validation is crucial.

7 Conclusion, Limitations, and Outlook

This paper **contributes** to the co-development potentials of BE and SE. It analyzed the similarities and differences between the BE/SE and DevOps contexts to derive an assessment on the applicability of the DevOps approaches, tools and mind-sets to BE/SE. We have identified that both contexts face continuous change which requires interdisciplinary collaboration as creativeness and innovation mostly originates from the intersection of several disciplines and especially requires them for their successful implementation. Moreover, as uncertainty increases with the number of stakeholders and the depth of change, frequent validation is crucial to enable educated decision-making and to achieve organization-wide acceptance.

To further detail our understanding we conducted an interview study with 33 experts. We chose to address four categories of interviewees: IT students with interdisciplinary experiences, IT experts, business experts and 'business & IT experts' with long experience in both fields of expertise enabling them to provide a rather objective view of the BE and SE contexts and their corresponding mind-sets.

Based on the highly structured interviews including quantitative and qualitative questions we have identified that these groups show major differences regarding current and wished for tool support.

The BE context faces more global challenges and greater interdisciplinarity than SE and in particular DevOps. This asks for rather manageable (as simple as possible) and generally/globally applicable solutions at the expense of accuracy. Today, these challenges are addressed via frequent meetings and presentations rather than concrete tool support.

SE and in particular DevOps on the other hand excel at accuracy to allow for (semi) automation and continuous tool support in addition to frequent meetings and awareness of diverging priorities. Here, DSLs allow for accurate and simple tools and solutions which fit in particular domain-specific contexts.

Based on these findings we derived at the following hypothesis for **future research**: DSLs can be regarded as an enabler for tool-based automation in

BE and, similar to DevOps, align cross community communication. Thus, the DSL-based approach would allow e.g. for tools/DSLs specifically designed for the cross-departmental cooperation required in a given project to achieve integral quality and alignment.

Due to the current exploration phase we focused on rather qualitative feedback on our questions at the expense of the generalizability of our findings. We propose to address this **limitation** via additional quantitative analyses to increase the reliability of our results. Further, in order to better meet the complexity of the BE context additional areas of expertise (e.g. finance and operations) should be addressed to account for their custom requirements and derive a more realistic picture of the overall complexity.

References

1. McKinsey Digital: Industry 4.0: How to navigate digitization of the manufacturing sector
2. Aichroth, P., et al.: Wertschöpfung durch software in deutschland: Aktueller zustand, perspektiven, handlungsempfehlungen
3. Allspaw, J., Hammond, P.: 10+ deploys per day: Dev and ops cooperation at flickr. In: Velocity: Web Performance and Operations Conference
4. Barnett, W.P., Hansen, M.T.: The red queen in organizational evolution (1996)
5. Bass, L., Weber, I., Zhu, L.: DevOps: A Software Architect's Perspective. Addison-Wesley Professional, Boston (2015)
6. Bechtold, J., Lauenstein, C., Kern, A., Bernhofer, L.: Industry 4.0 - the capgemini consulting view (2014)
7. Beck, K., et al.: The agile manifesto (2001)
8. Bloice, L., Burnett, S.: Barriers to knowledge sharing in third sector social care: a case study. J. Knowl. Manag. **20**, 125–145 (2016)
9. Brosseau, D., Ebrahim, S., Handscomb, C., Thaker, S.: The journey to an agile organization. McKinsey Insights 10
10. Carroll, L.: Through the Looking Glass. Project Gutenberg
11. Christensen, C.: The Innovator's Dilemma: When New Technologies Cause Great Firms to Fail. Harvard Business Essentials, Harvard Business School Press (1997). https://books.google.de/books?id=vWvixwEACAAJ
12. Daepp, M.I.G., Hamilton, M.J., West, G.B., Bettencourt, L.M.A.: The mortality of companies. J. Royal Soc. Interface **12**(106), 20150120 (2015)
13. Dahlberg, T., Lagstedt, A.: Fit to context matters-selecting and using information systems development methods to develop business in digitalization contexts (2021)
14. Debois, P., et al.: Devops: a software revolution in the making. J. Inf. Technol. Manag **24**, 3–39 (2011)
15. Fink, A.: How to Conduct Surveys: A Step-by-Step Guide. SAGE, Thousand Oaks (2015)
16. Fowler, M.: Domain-Specific Languages. Pearson Education, London (2010)
17. Fowler, M., Highsmith, J., et al.: The agile manifesto. Softw. Dev. **9**(8), 28–35 (2001)
18. Hemon, A., Lyonnet, B., Rowe, F., Fitzgerald, B.: From agile to devops: smart skills and collaborations. Inform. Syst. Front. **22**(4), 927–945 (2020)

19. Hessenkämper, A., Steffen, B.: Towards standardization of custom projects via project profile matching. In: Fernandes, J.M., Machado, R.J., Wnuk, K. (eds.) ICSOB 2015. LNBIP, vol. 210, pp. 186–191. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-19593-3_17
20. Hevner, A.R.: A three cycle view of design science research. Scand. J. Inform. Syst. **19**(2), 4 (2007)
21. Hirst, D., Darwin, C., Ramm, N., Bynum, W.: On the Origin of Species. Penguin classics, Penguin Books Limited (2009). https://books.google.de/books?id=e3hwBxBTh24C
22. Hudak, P.: Domain-specific languages. Handb. Program. Lang. **3**(39–60), 21 (1997)
23. Keese, C.: Silicon Germany: Wie wir die digitale Transformation schaffen. Penguin Verlag, München
24. Kelly, S., Lyytinen, K., Rossi, M.: Metaedit+ a fully configurable multi-user and multi-tool case and came environment. In: International Conference on Advanced Information Systems Engineering, pp. 1–21
25. Kim, W., Mauborgne, R.: Blue Ocean Strategy. Harvard Business School Press, Brighton
26. King, A.A., Tucci, C.L.: Incumbent entry into new market niches: the role of experience and managerial choice in the creation of dynamic capabilities. Manag. Sci. **48**(2), 171–186 (2002)
27. Korpivaara, I., Tuunanen, T., Seppänen, V.: Performance measurement in scaled agile organizations. In: Proceedings of the 54th Hawaii International Conference on System Sciences, pp. 6912–6921 (2021)
28. Kukko, M.: Knowledge sharing barriers in organic growth: a case study from a software company. J. High Technol. Manag. Res. **24**, 18–29 (2013)
29. Ledeczi, A., et al.: The generic modeling environment, vanderbilt university. Institute For Software Integrated Systems
30. Martin, R.: Clean Code - a Handbook of Agile Software Craftsmanship. Prentice Hall, Hoboken
31. Naujokat, S., Lybecait, M., Kopetzki, D., Steffen, B.: Cinco: a simplicity-driven approach to full generation of domain-specific graphical modeling tools. Int. J. Softw. Tools Technol. Transf. **20**, 327–354 (2018)
32. Osterwalder, A., Pigneur, Y.: Business model generation: a handbook for visionaries, game changers, and challengers, vol. 1. John Wiley & Sons, Hoboken (2010)
33. O'Reilly, C.A., III., Tushman, M.L.: Ambidexterity as a dynamic capability: resolving the innovator's dilemma. Res. Organ. Behav. **28**, 185–206 (2008)
34. Ries, E.: The Startup Way: How Entrepreneurial Management Transforms Culture and Drives Growth. Portfolio Penguin (2017). https://books.google.de/books?id=er-iswEACAAJ
35. Steffen, B., Boßelmann, S.: GOLD: global organization alignment and decision - towards the hierarchical integration of heterogeneous business models. In: Margaria, T., Steffen, B. (eds.) ISoLA 2018. LNCS, vol. 11247, pp. 504–527. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-03427-6_37
36. Steffen, B., Boßelmann, S., Hessenkämper, A.: Effective and efficient customization through lean trans-departmental configuration. In: Margaria, T., Steffen, B. (eds.) ISoLA 2016. LNCS, vol. 9953, pp. 757–773. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-47169-3_57
37. Thorngate, W.: "in general" vs. "it depends": Some comments of the gergenschlenker debate. Personal. Soc. Psychol. Bull. **2**(4), 404–410 (1976)

38. Weiber, R., Ferreira, K.: Transaktions- versus geschäftsbeziehungsmarketing. In: Backhaus, K., Voeth, M. (eds.) Handbuch Business-to-Business- Marketing, pp. 121–146. Springer Gabler, Wiesbaden
39. Wilson, M., Sapsford, R.: Asking questions. In: Sapsford, R., Jupp, V. (eds.) Data Collection and Analysis, pp. 93–123. SAGE, London
40. Winter, R., Müller, J., Gericke, A.: Der st. galler ansatz zum veränderungsmanagement. In: Organisationsentwicklung, pp. 40–47
41. Österle, H.: Business Engineering: Prozeß- und Systementwicklung. Springer-Verlag, Berlin Heidelberg, Berlin
42. Österle, H., Höning, F., Osl, P.: Methodenkern des business engineering - ein lehrbuch