Self-service kits to scale knowledge to autonomous teams - concept, application and limitations

Alexander Poth, Mario Kottke, Andreas Riel

Computer Science and Information Systems 20(1):229–249 https://doi.org/10.2298/CSIS211112048P

Self-Service Kits to Scale Knowledge to Autonomous Teams – Concept, Application and Limitations

Alexander Poth[1], Mario Kottke[1], and Andreas Riel[2]

[1] Volkswagen AG, Berliner Ring 2
D-38436 Wolfsburg, Germany
{alexander.poth, mario.kottke}@volkswagen.de
[2] Grenoble Alpes University, Grenoble INP, G-SCOP Laboratory
F-38031 Grenoble Cedex 1, France
andreas.riel@grenoble-inp.fr

Abstract. In large organizations, it is not trivial to spread knowledge to all teams. Often, individual teams need to handle similar topics and re-invent the wheel. Another scenario is that a group of people with a common role (for example "guild" in Spotify model) has to distill their practices to make them shareable. Trainings should have empower participants so to apply the learnings easily in their daily businesses. To realize this, the proposed Self-Service Kit (SSK) approach can be used in the context of a holistic methodology that fosters team autonomy while leveraging knowledge spread and sharing throughout a large organization. Such a methodology is presented and instantiated in an enterprise context in facing the mentioned challenges.

Keywords: computer science, information systems, agile, learning organization, efiS® framework.

1. Introduction

The VUCA challenges (Vulnerability, Uncertainty, Complexity and Ambiguity) [1] of our modern economic, ecologic and social context makes that more and more enterprises are confronted with the need to initiate and drive forward their agile transformations in order to be able to control change and react to change as fast as possible. Although agile frameworks, methods and practices have been largely published and widely adopted already, the company-specific introduction and scaling of whichever framework is always a huge challenge requiring enormous investments in terms of time, effort, and more often than not, also failure. One of the key reasons at the origin of this is the difficulty of making agile change propagate in the organization at a vast scale, in particular in large organizational settings. Departmental borders and, more generally, organizational silos [2] in classical hierarchical corporate organizations lead to the fact that agile transformation efforts frequently fail reaching the critical mass of entities required to make the change happen on a large scale. Overcoming these boundaries requires specific strategies that typically rely on the presence and active involvement of agile trainers and coaches in a huge number of departments [3]. These coaches have the mission to introduce agile mindsets and practices in a way that is most appropriate to the

current organizational culture and practices of that very unit. While this approach is mostly effective on a local (department/team) scale, it often fails to efficiently scale up higher, cross-departmental levels. Furthermore, the number of coaches required in order to adopt this approach in large organizations is in general hardly compatible (at least limited by the scalability of the coaches) with the resources that companies can or want to invest in transformation efforts, which will translate to tangible economic benefits only in the mid- and long-term [4].

This article investigates an alternative approach to this challenge that relies on team autonomy at its ultimate level for scaling. Its core concept is to provide teams with digital agile "transition kits" that they have developed and keep extending and improving themselves [5]. Agile coaches accompany this process, however at a far more distant and selective scale than in the classical approach. This concept essentially relies on teams knowing themselves best their needs and desired interpretation of agile frameworks and methods. Only initially coaches can just support them in identifying those, and they push for the creation and evolution of electronic agile self-service kits (SSKs) inspired by the concrete needs of the product/service teams, and moderated by experienced competence and knowledge leaders in the organization. The authors put this SSK-driven approach in the context of the large-scale [6] agile setups especially the efiS® framework they created and implemented in the Volkswagen Group IT. With respect to size, global outreach, cultural mindset, and diversity this global enterprise providing several IT services to the Volkswagen AG poses challenges to agile transformation that are largely comparable to those companies in other sectors are facing when adopting agile practices. The key point this paper is making lies in the demonstration of the potential of the SSK-based agile transformation approach through concrete success stories at the Volkswagen AG.

To this aim, this article is organized as follows: section 2 elaborates on related work in the field of large-scale agile frameworks and organizational learning-based transformation approaches. Section 3 explains the methodology applied to coming up with efiS® and the SSK's constituting this framework. Section 4 presents four use cases implemented at the Volkswagen AG in order to prove the practical relevance, efficiency and effectivity of SSKs in the large-scale setting of the Volkswagen Group IT. Section 5 points out the limitations of the SSK approach and its initial instantiations at the Volkswagen Group IT most notably with respect to the validity and transferability of the insights and evaluation results to other corporate environments. Section 6 concludes by summarizing the key contributions that this paper attempts to make both to the academic and practical community and includes the outlook which describes next steps.

2. Related Work

Works on the challenges that traditional organizations are confronted with when they undergo agile transformation are numerous like [7]. Many of those challenges rely on the fact that in agile environments, people are key success factors for the outcomes [8]. Taking this essential aspect into account adequately grows in difficulty with the size of the organization. In [9] we investigate widely adopted large-scale agile frameworks such as SAFe®, LeSS, Scrum@Scale™, and others, with respect to the ways they integrate

facilities for team autonomy, self-governance, and knowledge scaling. Our conclusion from this investigation was that although all three aspects are addressed to some extent in each framework, there is no consistent focus on autonomy in any of them, and self-service approaches are hardly addressed. However, as autonomous teams are key elements in agile organizations for product development [10], self-service approaches for knowledge scaling in large organizations is a subject that has been identified as relevant in several contexts, and considered as non-trivial [11]. Among them, Problem-Based Learning (PBL) [12] relies on collecting and documenting particular problems and their solution approaches. If applied by the teams themselves, and enriched by guidance [13], the gathered knowledge can be compiled in SSK's.

Given the fact that in agile teams, the social component is in general well developed [14], knowledge sharing support through social network sites [15] and communities of practice (CoP) [16] are especially appropriate, since agile teams are based on the specific competencies of their individuals [17]. Furthermore, autonomy and self-organizing teams come together and need cross-functionality, which is based on sharing of knowledge [18] that is available both within and outside the teams.

Another important driver for self-service approaches is the increasing geographical distribution of agile organizations [19] and teleworking [20]. Self-paced distant learning based on goal-based scenarios [21] is more and more relevant, combined with blended learning [22]. Self-paced and asynchronous formats [23][24] are extended by synchronous online formats for facilitating tight collaboration [25].

In corporate environments, Web-based training (WBT) approaches are increasingly common [26]. They transfer information; however they fail guiding knowledge spread. Giving an adequate design context to the knowledge spread approach is therefore a key success factor [23], since it is a prerequisite to enable learners to re-contextualize the provided learning content. Labs used for practical instruction-guided training combined with e-learning constitute a blended learning method [27]. Such labs provide guidance, however they represent only a limited set of real-world scenarios. SSK's inspired by and deployed in real-world scenarios and created by Communities of Practice can overcome these limitations, however care has to be taken in the problem identification and solution guidance to avoid significant failures [28] leading to harm [29] either by misguidance, misuse or even by accident.

To the best of our knowledge, combining the knowledge concentration and dynamics of Communities of Practice with the rapid and convenient spread of web-based of corporate SSK's has not been investigated in literature so far, although it promises interesting scaling power.

3. Methods

3.1. Research Goals

The goal of this research is to come up with an autonomy-focused approach to facilitate agile transitions for large-scale enterprise settings without the need of extensive team

coaching. The focus of the proposed approach is to fosters team autonomy to a maximum while assuring the fast and deep propagation of agile mindsets and practices across the entire company. In this context, knowledge scaling is an important building block which is addressed by SSK's. To shape the term transition and transformation in this context, we define transformation with a defined target state which can be reached and then it is done. A transition is more used in a Cybernetics [30] thinking. In this case agility is like a variable and the variable can change the state to any value. There is no final state defined – it leads to a continuous step-by-step transition to more agility. Depending on the teams transition progress they have an agile maturity which leads to the ability to act autonomous. However, the team specific "maximum" depends on the team maturity and their products [31]. The autonomy enables the teams to define and shape their delivery procedures within their value streams – the autonomy leads to createablity [32]. For this, the knowledge-scaling focused SSK approach shall be extended to be easy to use, adaptable to the specific needs and maturities of organizational units and their teams, and be deployable in any agile or non-agile framework. Particular design constraints are given by the need for the minimization of frequent and broad agile coach interventions in individual teams for the efficiency reasons explained in the introduction to this article. Furthermore, the approach shall be based on the SSK paradigm defined in [33]. The concept of the SSK to offer knowledge like learnings which become knowledge or later on specific Intellectual Property (IP) of an organization to the autonomous teams. The instantiation of this knowledge can differ from the information which is shared. To support this a SSK defines a kind of common structure and a basic set of relevant meta data. Furthermore, the detailed SSK implementation is open to address the consumers behavior. To foster loosely coupled autonomous teams the knowledge have to be available in an asynchronous way – else a more intensive coupling for synchronization between the teams for knowledge transfer is needed. With the SSK concept teams have to possibility to become prosumers. In their early stages with lower maturity they consume the SSK with the knowledge or IP shared from others. With their growing maturity teams can become to producers of knowledge with their learnings. They can extend existing SSKs are build new SSKs if they have opened a new knowledge domain. The objective of SSKs is that they are like 24*7 available, independent and self-contained knowledge units.

Based on the fundamental research objectives specifically defined for the SSK transition approach in [34], we aim at building, distributing and relating SSKs to each other such that they help constitute a holistic aid in adopting (agile) practices for specific topics and subject areas that are of particular importance for the specific organization.

Based on the identified aspects and observations requirements for a holistic SSK approach can be derived. More specifically, we define the Key Requirements (KR) to the SSK-based agile transition framework as follows:

1. SSKs as defined in [33] shall be applicable for "stand-alone" knowledge topics in a way that topic experts initiate the SSK creation and foster their continuous improvement and share the learnings with other teams. SSK application through teams shall happen in a "team-pull", rather than in a "coach-push" effort wherever possible. These SSKs shall drive the building up of knowledge in a particular subject area on a large organization scale.

2. SSKs shall also serve as a means of delivery for building blocks of holistic agile frameworks as efiS®. Compared to 1), this requirement implies an increased need for complementary design of individual SSK's and consistency among them.
3. SSKs shall also be designed in a way that hands-on trainings can be built from them. The integration of agile transition means (such as SSKs in this case) in trainings not only fosters efficiency, but it also allows collecting direct feedbacks on the understandability, relevance and needs for improvement for these SSKs.
4. Given 1)-3), SSKs shall also serve as part of knowledge management instruments within the organization. If applied on a large scale on several core knowledge topics of the company in bidirectional form (i.e., employees get knowledge from them, but also feed in new knowledge into them), they can be seen as a living representation of the diversity and actuality of core knowledge elements. Knowing about the actuality requires life cycle management facilities built into SSKs.

3.2. Research Context Design

The Volkswagen Group IT is the field of research. The Group IT is like a virtual organization which includes the IT capabilities of the brands. Additionally, the brands act autonomous within their business areas. So, it is possible to group e.g. specific technology expertise logical in communities. However, all legal entities can organize and decide about their projects etc. within their autonomy. Agile transitions are made in different ways. First it can come bottom up – a team decides to work aligned to an agile approach they find suitable for their setup and context. In this case the team can self-organize this transition journey or get help by agile coaches. The second approach is more top-down. In this case e.g. a program management decides to establish an agile setup. Here, the approach is predefined and the teams have to align with the program management decision. Often facilitation is given by the program management to the teams to become part or the programs transition journey. Coaches often facilitate transformations to reach pre-defined goals in a time-framed period. Then the enabled teams go ahead autonomous within their agile transition. As the demand about specific agile knowledge can vary significantly over time, approaches are needed to deliver the demanded knowledge just-in-time. Human coaches are not able to scale in/out with the demand-peaks. To handle the volatile demand other approaches are needed. One facilitation for the dynamic demands are self-services.

This complex and heterogeneous research environment makes it difficult to define synthetic investigation setups which are representative to ensure a-priori that outcomes can be transferred to the real world. This leads to selecting a research approach which can handle real-world environments and deliver its outcomes direct into the demanding organization or team. The objective is to work directly with a set of users to ensure that all real-world effects are addressed and the outcome produces value for them. In a second step, the "general availability" scales the outcome.

3.3. Methodology

The methodology fits to the research environment and the research question (RQ). The RQ is how SSKs concept can be used on a broader base to address typical scenarios of knowledge sharing in enterprises? This leads to the hypothesis that the current SSK concept needs selected extensions to fit the set of typical knowledge sharing use cases of enterprises. All new concepts to extend the existing SSK approach are introduced in this work. The existing capabilities and features of SSKs are described in [20]. To identify the use cases different experts from the Agile Center of Excellence (ACE) and Test & Quality Assurance (TQA) were requested to show typical cases in which knowledge is needed. This can be used to derive the use cases and specific demands about SSKs. Consequently, the methodology chosen to develop the efiS® framework – an agile framework for enterprises - and one of its core building block SSK [33] is Action Research (AR) [35]. Oriented on the AR approach the specific AC for this work is derived. The experimental field included several organizational entities within the Volkswagen Group IT. In their double-roles as researchers and agile coaches, the authors have chosen the approach of semi-formal interviews with different product/service teams in order to accompany them on their ways of creating and instantiating SSKs for different purposes linked with the agile transition. The extended SSK concept was developed within a working group of ACE and TQA members. This was useful also to get fast feedbacks in the daily work of coaching and facilitation of teams and organization. The prosumer approach of SSKs is based on the collaborative agile mindset by design. In order to assure the complementary nature of the SSKs, an SSK assured the uniform core structure and the facilitation of SSK use and understanding. An important further structural element is the common life-cycle developed for SSKs, which comprises a fundamental state-model which can be adapted and extended if needed. Some meta-data attributes were also defined from the outset facilitating the management, structuring, and linking of SSKs. Some of these attributes are mandatory (e.g. last revision, the name of the owner/maintainer), others are optional and/or context-specific (e.g. the maturity of the content, dependencies of documents, other SSKs etc.).

The action research approach within the corporate environment of the Volkswagen Group IT intrinsically included the validation of the developed SSK approach. The latter's more general validity and applicability has been considered consistently through all phases by making sure that any concepts chosen to build the efiS® framework and its SSKs are not specific neither to the company nor to the industrial sector it is active in. The validation measures and observations were conducted in real product or service environments. Figure 1 shows the flow which combines the research (analyze potentials and evaluate solution) and development (identify scope and develop solution) of the SSK concept. The iterative approach is started by the initial demand and terminated by the released solution. The termination condition for this specific AR flow is the "sufficient fit" between demand and the current solution, by identifying no further significant improvement potential which justifies another development iteration. All steps of the flow for the SSK development are mainly conducted by the SSK working group, only the step evaluation solution was mainly done by teams who have to use the potential released solution in the future. In this step the working group was "observer" to learn also about usage, adaption issues etc.

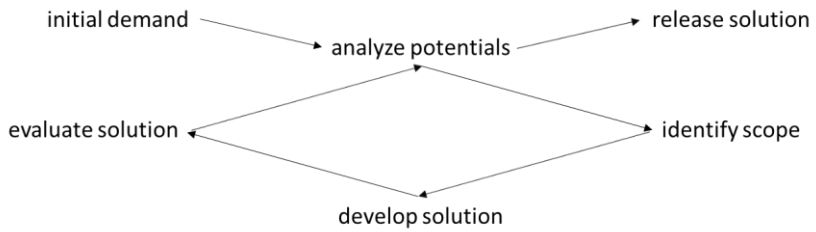

Fig. 1. The iterative research and development flow with the entry and exit points.

3.4. Design of model capabilities

Derived Use Cases. In order to meet the research goals, we decided to define representative use cases for SSKs with a high practical relevance. This helps to avoid over-engineering of the SSK approach extension for scaling agile practices and knowledge. Such use cases have been derived from typical enterprise settings. The following use cases are addressed:

- Stand-alone SSK (Research Goal 1): to package knowledge of one topic to make it available to other people to address KR1
- Network of SSKs (Research Goal 2): to relate SSKs to address KR2 with each other in the context of
  - Frameworks with more or less dependencies between their "building-blocks": establish a harmonized set of SSKs for topic-specific building blocks to build complex frameworks to realize additional effects beyond the specific building blocks on the system level of the framework; this requires a harmonized release of all involved SSK of the network for assuring consistency.
  - Knowledge base with maturity "indicator" for the practice: extends the life-cycle of SSK to additional attributes. Beyond the versions semantic aspects to manage for example maturity of the SSK content.
- SSK as training "take-away" package (Research Goal 3): bundle specific topics to work with them in the training on examples and later on re-use them in the real-world context to address KR3.

All 3 research goals together contribute to address KR4. These use cases require additional attributes and meta data for SSKs to model the demanded characteristics for the enterprise setting. The additional information attributes have to be designed to be optional or additional to keep the compatibility to the SSK approach v1.0 as proposed in [33]. The additional attributes define v1.1 with the extended modeling capabilities. For the attributes and their values context specific options have to be identified like pre-defined values usable as tags or references like links.

Demanded as optional meta data, the dependency and the maturity description is needed. To keep the application of the SSK approach as simple and intuitive as possible, the design results of the v1.1 have to be lean and easy to understand.

Dependency handling. These data have to be designed to handle references to other SSKs. Three different types of dependencies are needed to handle a network of dependent SSKs:

- No dependency – no action needed (trivial case).
- Uni-directional dependency – needs checks to evaluate the "source" update to a new version.
- Bi-directional dependency – on every update of both sides, a check is needed.

Existing formal approach for dependency modeling like [36]. However, typical users do not have the formal modeling approach in mind, if they want to express a relation. This leads to investigating more intuitive approaches. Mark the dependency on the SSK visible for everybody to make it transparent. This can be realized as an attribute to the SSK metadata like for e.g. "dependency to:". Then, the one or more dependencies are listed. An intuitive way to do this are e.g. hyper-links, which are well-know and understood by mostly all people.

In a second step, identify the semantic level of the dependency to avoid SSK dependencies where possible by smart SSK design. A strategy is the indication of a semantic dependency, but avoid to have content which is subject to change in the future. The result is that the overall network is described, but changes driven by the dependency are rare or completely avoidable. These are weak dependencies. An example is to have a dependency identified and then only link to the "master description" and avoid partly repletion of the master description as a copy which has to reworked in case of changes in the master description.

In case of established sematic dependencies, ensure validations for new versions to enable blocked releases of the dependent SSKs if needed. Blocked release packages have strong dependencies. In case of semantic dependency to the content of the master description the version of the master description has to be part of link to avoid inconsistency. To make this kind of dependency transparent, it is useful to add the version of the dependent SSK to the dependency attribute. So, everybody can independently identify on their own cases where potentially inconsistent versions are used. As a positive side effect of this information, it helps to initiate updates to newer versions of dependent SSKs, because users can have local outdated copies of SSKs.

The presented approach with links for dependency modeling and optional version binding offers an easy to understand approach with a lot of options to model different kind of dependencies.

However, this dependency handling approach does not help to identify dependencies in a (semi-)automated way – this is still the work and responsibility of the SSK team.

Maturity Description. Modeling of a maturity life-cycle for SSKs can be realized by a state diagram. For most cases the state diagram is straight forward, like: concept → evaluated → established → decommissioned. Any SSK version can have only one state. This makes it easy to realize, each state as a "tag" in the simplest modeling way. The current maturity state respective the representation as tag is associated with the SSK. The approach of using "tags" has the benefit that it also can be used to model more complex maturity life-cycles. A more complex case is for example the case in which different domains have to be represented by a set of states like: evaluated in domain x, established in domain y and decommissioned in domain z. The important point is that each domain can model its life-cycle flow independent with the tags. This can lead to n domains * m states (Figure 2). In the worst-case, n*m tags have to be assigned to the

maturity attribute. However, in practice, this is untypical (due to a very simple design of the SSK maturity description) and is only a theoretical issue to have a large amount of tags. The approach to describe the maturity with tags is useful for a trivial maturity life-cycle staging, but it also allows building more complex stage-models if needed. This makes it adaptable to many application scenarios.

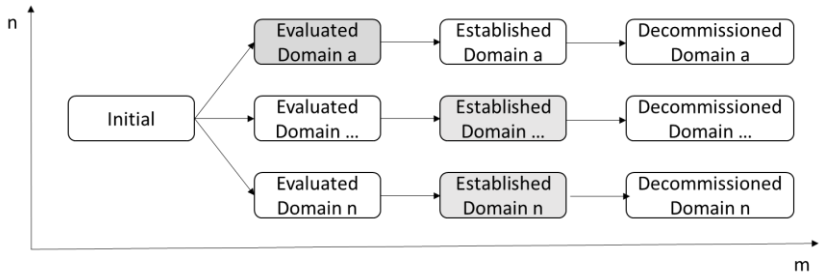

Fig. 2 Example of a complex set of tags to model maturity with business domain orientation – each domain can have an individual usage "state" represented by the grey tagging.

4. Evaluation

The evaluation presents examples from the AR setup of different use cases of the Volkswagen Group IT to demonstrate the potentials and applications of SSKs. The use case specific demand to SSKs for the new proposed meta data attributes are presented, too. The four case studies are structured in a way as to explain the objectives of each use case, and the role of SSKs in their implementation. Then, the learnings and results for each use case are presented. While we did not undertake agile maturity evaluations at the beginning and end of a certain evaluation period, we directed the evaluation focus on how the SSK's were perceived, accepted, and integrated by the teams under evaluation. From that perspective, agile teams reporting a felt usefulness of the SSK's, as well as contributing to the latter's extension and improvement, leads to a positive evaluation. Each sub-chapter addresses one of the four KR and all together contribute to the RQ. Each sub-chapter presents the needed extensions to reach its KR. This confirms the hypothesis, that with only a few selected extensions use cases in real enterprise setups can be solved with the SSK concept.

4.1. Case study A: "Stand-alone" knowledge topic

Use Case: Use the SSK approach to deliver a topic like Quality Assurance (QA) for Machine Learning (ML). The presented evAIa approach [34] is a representative example for this use case. The evAIa approach is used to identify adequate safeguarding actions for ML based products and services. This identification should be realized by autonomous teams – this motivates an SSK for the evAIa approach. The SSK can be

built on the basic structure initially presented in [33]. No additional attributes are needed. Figure 3 shows a part – an overview - of the SSK. The content of the SSK is directly related to the content of the evAIa publication. A special aspect is the reference to the PQR approach [37] to identify product specific quality risks. This is an additional aid to ensure the right focus for the evAIa application. It is referenced as a kind of useful pre-condition. However, there is no direct dependency to the PQR SSK established. This example shows how two SSKs one for the evAIa and one for the PQR method can have a relationship by reference without having a direct dependency.

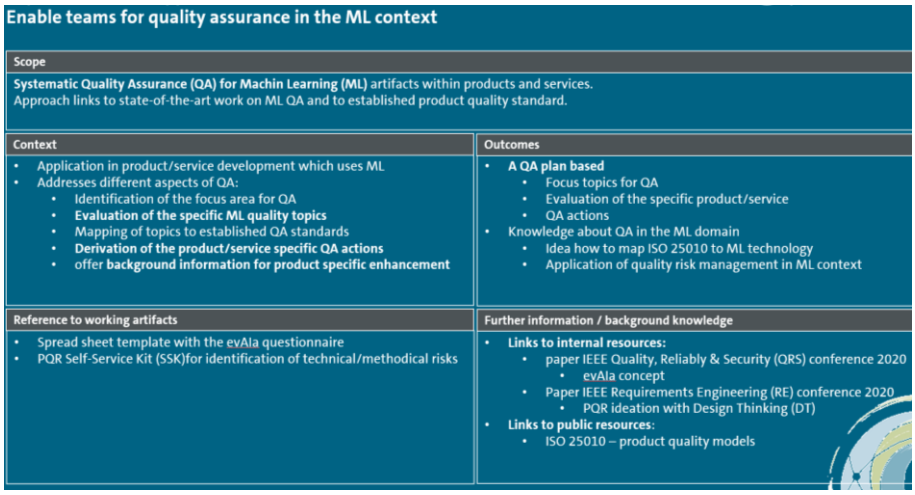

Fig. 3. Extract from the evAIa approach SSK.

Setup: The evAIa approach was developed in a cross-brand team from Audi, Cariad and Volkswagen [34]. By design the documentation was made with the SSK concept. The evaluation took place in ML teams of different brands. The feedbacks were used to enhance the SSK and the questionnaire of evAIa. The authors of the SSK observed the usage by the teams. The iteration phase terminates as no significant improvement potentials were identified. This termination criteria is based on the feedback and the observations which indicate an intuitive and fluent application of the SSK as a specific definition for "ease to use". Key feedback included a clear facilitation of QA approaches to Machine Learning based software. Teams reported being more comfortable in the selection of appropriate QA methods through the guidance by the evAIa SSK. This contributes to an increased level of team autonomy with respect to this selection.

Learning: SSKs are easy to use and open for individual refinement of non-defined methodologies and model aspects like the pre-condition handling of the evAIa approach with PQR. Applicable in insolated setups without the complexity to separate a semantic content into more SSKs.

Results: The SSK development kit 1.0 as facilitation tool for other SSKs was designed to build independent SSKs. Dependencies are not explicitly managed. They can be handled in many individual ways. The evAIa example shows this with the "pre-condition" approach.

Based on SSK development kit. Adaption: no

4.2. Case study B: Building blocks for a combined knowledge package

Use Case: Use the SSK approach as a generic template for building blocks of a network of more or less dependent knowledge "elements". An example for a semantic network of knowledge elements is a framework. The presented evaluation uses the efiS® framework [38] as a real-world case. The efiS® framework delivers itself and its building blocks with SSKs. The core objective of this use case is to show how different building blocks can be combined to a more holistic and dependent knowledge package. The presented case study focuses on the efiS® framework - the holistic knowledge package. The four pillars of the framework are comprised amongst others of aTWQ [39], TTM [32], PQR [36], LoD [38] and SSK. This example focusses only on these selected building blocks to show how the combination of different building blocks described by SSKs can be assembled to a network with dependencies to realize a holistic knowledge package. The aTWQ (agile Team Work Quality) approach supports evaluating the generic team maturity. It is based on an agile team maturity model in the sense of [40] and a team-autonomous maturity evaluation approach. The TTM (Technical Team Maturity) is a technology-focused extension helping to direct team maturity evaluation on their mastery of technologies (e.g. cloud technology). The PQR (Product Quality Risk) approach supports identifying product/service specific quality risks to mitigate them adequately during the development. The LoD (Level of Done) approach supports focusing on regulation and standard requirements relevant to the specific product/service domain. The LoD layer approach is an optional LoD extension fostering reuse of domain specific sets of regulations. Figure 4 presents an overview of the efiS® framework and a mapping of some of its building blocks to the four pillars (empowerment, focus, integrate and scale) of the autonomous value streams. The goal of the SSK dependency model is to offer a consistent set of SSKs as a knowledge package of the efiS® framework to autonomous organizations and teams.

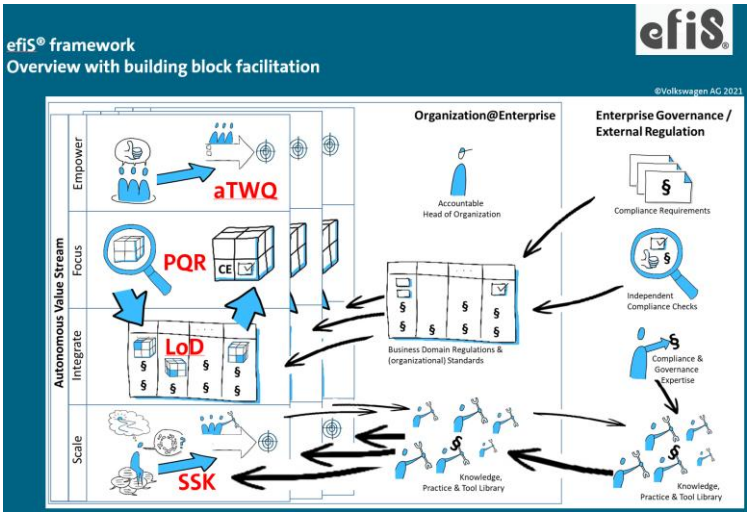

Fig. 4. The efiS® framework with its 4 pillars and selected building blocks (red) mapped to pillars.

Setup: The efiS® framework is assembled by building blocks. The building blocks were developed by expert teams around the building block topics – the SSK developer. Often at least one of the SSK developers becomes SSK maintainer over its life-cycle. From their view each building block is independent. However, the efiS® framework creates by the smart assembly a holistic framework. The work was to identify the level of dependencies between the building blocks and the framework. This was made by experts of the efiS® framework. The resulted SSK network usage was observed in a large SAFe® instantiation with 10 teams. The iteration phase terminates as no significant improvement potentials were identified.

Model the dependencies. In the case of the efiS® framework, the empowerment drives the autonomy of the team. This influences the capability to modify the delivery procedures. This can impact the LoD and the PQR handling, too. These are unidirectional dependencies.

The integration with a LoD defines the compliance aspects of the product development and specific domain knowledge of the organization for the delivery procedures. The LoD directly impacts the product development and influences the deliverables like products and services according to Conway's law [41]. Conway's law says that "systems image their design groups" also the relation between organization and product/delivery architecture/structure. This is a unidirectional dependency.

The focus on products with PQR identifies product/service specific aspects. Some of them have to be handled via procedures and actions handled by the LoD. This leads to a strong interaction of the PQR and LoD. This is a bi-directional dependency.

The scaling with SSK empowers the team with knowledge in most cases. In some cases, new knowledge is scaled by the team into the organization by adding or creating SSKs. This dependency is additive not changing structures. This is a bi-directional dependency.

Figure 5 shows the described dependencies.

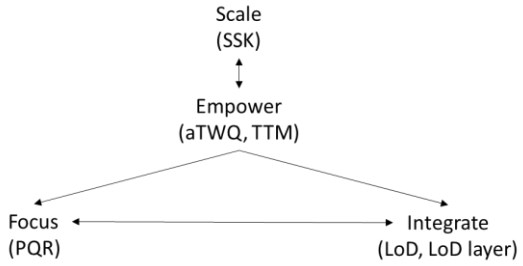

Fig. 5. Dependencies between the efiS® framework pillars.

Avoid content dependencies. Try to describe SSK content without content dependencies. Find the level of real impact/interaction of content – try to avoid the interaction to have the SSKs independent were possible. This also applies to SSKs within the same efiS® framework pillar. An example are the aTWQ and TTM approach. Both are designed to be orthogonal to avoid influencing each other. One handles the generic agile team capabilities the other specific technology capabilities. This makes it possible to have independent SSKs for aTWQ and TTM.

Via the scale pillar, knowledge comes into the team and empowers it. However, the content that is delivered can have dependencies on a semantic level. As described for the empower pillar, it is the work of the SSK designers to split the content to independent SSKs. This has been achieved for the SSKs of the efiS® framework. For other content, this has not been assured and therefore has to be handled in the team – especially in case of contrary "recommendations". The same is in case of contributing or building new SSKs by empowered teams. This insight should be mentioned as a recommendation in the SSK development kit that this insight should be handled during the SSK development. In some case this can lead to refactoring existing SSKs and new releases of aligned SSKs.

The empower pillar is driven by the knowledge delivered by SSKs to the teams e.g. the SSK for aTWQ or TTM. Furthermore, the team specific maturity profile, which typically grows over time, influences the team autonomy. This autonomy can have impact to the focus and integrate pillar, because the autonomy can lead to optimizations by the teams. The semantic level of impact is the adoption of the integrate pillar. By design of the LoD SSK this semantic dependency is handled by offering only an example which can be adopted. The same semantic level is with the focus pillar and the SSK for PQR approach, which is designed to be adoptable.

The focus pillar with its PQR SSK can have impact to the integrate pillar by adding product specific actions to the LoD. To handle this semantic dependency, the LoD SSK is by design open for the kind of extensions.

The integration pillar with its compliance handling impacts the product with the derived tasks and actions. The LoD SSK by design handles this interaction from and to the product focus pillar of the efiS® framework. The efiS® framework indicates this with the arrows between the pillars (Figure 4). This semantic dependency is modeled by the efiS® framework and is a design constraint for the SSK. This modeling is motivated as it is a fundamental part of the ISO 9001 requirements with "8.1 The organization shall plan, implement and control the processes, .., needed to meet requirements of relevant interested parties and the quality objectives for the provision of products and services …to realize opportunities and mitigate risks…" and "8.5.5 As applicable, the organization shall meet requirements for post-delivery activities associated with the products and services. In determining the extent of post-delivery activities that are required, the organization shall consider: a) the risks associated with the products and services;" This makes it easy for all teams to ensure that the efiS® framework implementation fits to the extracted examples as representatives for typical demands of the ISO 9001 and similar derived quality requirements of other standards and regulation which often are inspired by the ISO 9001 (or at least do not contradict to the ISO 9001). This makes it explicit that not the LoD implementation alone handles the specific product/service risks adequately.

Check for release packages. As it was avoidable to establish semantic and content dependency for all pillars and their SSKs, no release packages have to be defined. However, the LoD and PQR SSKs have to be controlled every change to keep this level of "weak" dependency without release packages.

Learnings: The efiS® framework SSK team realized that the LoD and the optional layer approach can be described in one SSK to avoid redundancies between highly dependent SSKs for the case that they are delivered by a LoD and LoD-layer SSK. This leads to a refactoring to one SSK which contains the LoD and LoD layer approach.

Furthermore, this learning was integrated in the generic SSK development kit to remember SSK developer to avoid dependencies wherever possible. All other investigated dependencies were handled intuitively adequate in the efiS® framework SSK setup.

Results: The instantiation of the efiS® framework based on the SSKs is possible. The authors have been accompanying programs which are continuously adopting the efiS® framework. As an example, one program adopted the efiS® framework with its core building blocks with approximately 3 Program Increments (PI) of their SAFe® instantiation without explicit agile coaching by ACE efiS® facilitators or efiS® trainings. In each release train, selected building blocks were established. Equipped only with the SSK and some broad agile knowledge, they started with the efiS® framework SSKs to understand the underlying concepts. They identified that the initial setup of the LoD will take its time due to the stakeholder feedback iterations in principal, however also other factors linked with the adoption of new practices. Then they progressed with the product risk evaluation to get "scope". In a next step, they initiated with scoped and "open mind" teams with the aTWQ and TTM. A weekly 15 minutes call was established to talk about "issues". Sometimes this call was omitted, or an extra review/feedback session was added for outcomes like the LoD. Overall, this accompaniment was less than 1h per week and was a "nice to have" because the basic adoption of the efiS® framework was made based on the SSK. The advanced specific feedback and coaching advices/recommendation are outside the SSK scope. The systematic dependency refinement approach helps to go a step towards the long-term goal: self-explanatory of efiS® based on a basic set of SSKs. The presented simplified use case example is applicable also to networks of more SSKs. The entire efiS® framework is currently described by 7 SSKs. The relation of the SSKs builds a logical flywheel in which the whole framework is more than the sum of the building block parts.

Based on SSK development kit. Adaption: semantic dependency needs synchronized versions.

4.3. Case study C: Training Take-away

Use case. The integration of SSKs into the practical parts or exercise lessons of trainings ensures that the learnings are "packaged" into a useable setup for real world application. This is motivated by the revised Bloom's taxonomy [42] and the competency driven curriculums of trainings. This has been applied in the efiS® trainings and TaaS trainings for the Volkswagen Group Academy and the IT Academy, respectively. All the mentioned SSKs existed before the trainings were designed. This constraint led to the point that the trainings had to "reuse" an as-it-is SSK. The implication is that the case for the practical lessons needed to be designed to fit to the training curriculum and be a good reprehensive case for the SSK. The effort of the case design in practice requires less effort than to develop the practical lessons from scratch. This positive side-effect fosters the reuse of SSK for trainings, too. An issue is that the life-cycle of the SSK is independent from the training life-cycle. Always using the newest version of the SSK, may require some adjustment of the trainings exercise case. However, in practice, a smart design of the exercise case oriented on the recommendations of the dependency section avoids this in most cases. The evaluation in two TaaS and two efiS® framework

trainings were positive examples for easy integration, low creation effort (compared to building them from scratch), as well as a high practical relevance. The added value for the trainees for the understanding and adoption of the learned concepts into their daily practical work environment was reported as high. A useful side effect of the "lab character" of the trainings is that one can observe the trainees in their practical SSK applications during the trainings. This is helpful to get inspirations and feedback for improving the SSK for better usability.

Setup: Typically, the trainings are developed and maintained by different persons than the SSKs. At least the point in time in which they are developed are different, because trainings typically follow the demand. However, training demand comes after the "content" is existing and not reverse. Parts of the training content can be described in SSKs. In this case the trainers used feedbacks of the trainees about the usability of the SSKs in the exercises. The iteration phase terminates as no significant improvement potentials were identified.

Learnings: It is possible to integrate existing SSKs into trainings. For such trainings, case studies have to be designed which use the SSKs in the learning context of the training. With this setup, it is possible to train people how to work with SSK and how to get most value out of them. Furthermore, it can be used as case study to observe the application of SSKs to get ideas to improve them.

Results: The integration of SSKs into trainings is a step forward to blended-learning. The SSK can be used during the training or during preparation or "home-work". After the training, the SSK can directly be used in the trainees' organizations in their specific project contexts. This helps bundling training content for real world application without additional transfer work by the trainees. This is a big step forward to ensure that training content is relevant and usable for trainees.

Based on SSK development kit. Adaption: no

4.4. Case study D: Body of Knowledge

Use case. Expert teams use SSKs to structure and offer practices with their groups. To indicate the maturity of practices, an additional maturity-attribute is used. It is a collection of expert experience which grows in maturity with application by others (validation and enhancement over time) – from a successful implementation to a best practice. The attribute is realized as a set of tags. The tag values of this maturity attribute have the following significance:

- Explore: practices under development – for experiment in a limited risk environment
- Verify: practices with a disruptive potential – for early adopters
- Open mind: emerging practices on the way to be main-stream
- Must have: used for practices which are recommended as state of the art or main-stream
- Deprecated: practice is not recommended to use anymore

The IT Quality Manager community uses this maturity attribute to build a body of knowledge for their communities. This approach enables all to contribute and use the practices. Depending on their maturity and application context, the "portfolio" of practices is more or less huge. The moving to a new maturity stage of practices is made

in the cyclic community meetings. This makes it transparent to everybody what is new or what has been changed. The intention behind this approach is that the experts can build their own set of practices for their daily businesses and keep it up to date within the learnings of the entire community. Every community member can participate and contribute to practices with new insights and learnings form their product and service domains of the entire Volkswagen Group IT.

Setup: The experts in the IT Quality Manager community are prosumers of the SSKs. So, they can decide what is missing and what should be enhanced to fit their quality expectation for their Body of Knowledge. The iteration phase terminates as no significant improvement potentials were identified.

Learnings: The beginning was not easy, because it was new for the community to be themselves the authors of the practices they want to spread and establish within the organization. This needs time and examples to show what should be in the practice collection and in which maturity state. However, the pressure to add practices was "build" with the strategy that there was no central team defining and maintaining the practice collection like often done by central expert or governance teams. This turned out to be an important lever for team autonomy and self-governance, while fostering mutual collaboration and knowledge sharing among the teams at the same time

Results: After initial practice collection during setup a slowly growing set of practices for the IT Quality Manager community. However, slow growth it not a negative indicator, because a practice set should not be an encyclopedia. It should rather be selection of practices that are compatible with the working culture of the specific organization. The important point is that this case study shows that a community can work with SSK's to build their domain-specific body of knowledge.

Based on SSK development kit. Adaption: semantic life-cycle requires additional attributes

4.5. Generic Topic: Source and Updates of SSK

Table 1. Meta data for SSKs to model advanced use cases

Attribute	Content
Dependency	Dependency with other SSK and optional info weak/strong
Maturity	Tag(s) for state
Source	URL(s)
Version	x.y.z
Maintainer	Email(s)

The SSK approach creates the significant challenge of keeping up to date the different SSK versions create by the users in different project teams and organizational entities. A recommendation is to establish one official source to avoid many update-sources. However, this only supports the SSK maintainer team. As a meta data of SSKs, the source tag is a useful information for users, to know where they can check for updates. The version helps to see the type of update if the notation x.y.z is used. X is the major version indicating non-backward compatible changes. Additionally, it is useful to have a history/change-log in the SSK to show what was really changed. Y is the minor version

used for extensions or enhancements and z for bug fixes. The producer or maintainer is given as contact person.

Table 1 presents the proposed meta data attributes which were useful during the holistic evaluation of advanced use cases for SSKs.

5. Limitations

In order to point out the limitations of the proposed approach, we distinguish between the SSK approach itself as defined by its methodology, and the evaluation setup with its constraints.

5.1. SSK Approach

The SSK approach is not a formal modeling method. This is a limitation by design, but needed to enable most people in agile organizations to produce and maintain SSKs without deep formal description knowledge. The minimal formal modeling extensions are a benefit in more complex use cases, but they are not automatically identified like a SSK dependency and model checked which can lead to errors and failures during the SSK life-cycle of creating, updating and deleting, if it is no longer useful.

5.2. Evaluation Setup

The case study evaluations cannot cover all possible real-world scenarios. The selected scenarios are from a real-world enterprise context, but they are only indicators that "typical" setups get benefits from the presented approach. Another limitation in the setup is that by design the distribution to autonomous teams makes measuring difficult. Metrics can only partly cover samples of observed SSK applications like trainings. Furthermore, samples are not randomized, since training participants represent a specific group of people who are interested in the training topic. These people have the skills and knowledge "required" for the training, and may not represent the entire organization. Additionally, the researchers act as designer and observers of the application of the designed artefacts. This can lead to some selective observations. Selective observation can lose sights of some potential interesting improvement aspects or issues by the application. Due to this, there might also be some bias in the evaluation of the presented approach's effectivity and efficiency. The corporate context made that the authors could not avoid this setting. However, they did their best to let the teams ("clients" of this approach) judge, even if not necessarily based upon strictly quantifiable criteria.

6. Conclusion and Outlook

The paper focuses on the definition of requirements for extending SSKs to becoming a means of fostering the agile transition in large company settings, especially in distributed on-line (virtual) organization settings like during the Covid pandemic. These enhanced SSKs open a new dimension of application and adoption scenarios in practice and provides new insights from an academic perspective. More precisely, this work shows the practical feasibility of fostering large-scale agile transitions with a self-service approach that relies on team autonomy both in terms of its design and its deployment and evolution by the experimental application of SSKs in the wide range of evaluation setup. SSKs providing the fundamental basis of this approach, we have shown that by extending the life cycle model of such SSKs and integrating the consistent handling of SSK dependencies. The SSKs can be effective drivers for the implementation of a holistic agile practices framework. We have shown this in the form of practical use cases in the complex setting of the Volkswagen Group IT. These use cases covered stand-alone application of subject knowledge SSKs, related and complementary SSKs constituting a knowledge framework, as well as a bundle of SSKs instantiated in different organizational units and representing a body of knowledge. We have also shown that classical trainings can complement the self-service approach most effectively when they are based on the SSKs themselves.

Practically, we have shown advanced design options of SSKs for different use cases that lead to more self-explanatory SSKs that reduce training/coaching efforts on large-scale agile transition journeys. New business cases for SSKs have been made possible through dependency modeling and SSK life-cycle management. So far, we have not yet shown the specific, quantifiable influence of SSK application on VW Group IT's way from a specific point A to a specific point B in their agile transition journey. Our objective was rather to show that SSK's can and already did give a significant contribution to increasing the autonomy and knowledge-sharing of distributed, diverse teams in the large organization.

Possible next steps shall move further in the direction of self-explanatory SSKs. However, it is not trivial to ensure that without adding more attributes while keeping complexity of the approach low, which is important to ensure that mostly everybody can still build and use SSKs. Furthermore, the measurement based on generic KPIs and the management of distributed SSKs is still an open topic which needs more investigation in the future. Here, the focus could be to establish representative sampling or "sending home" some meta data, but this requires some kind of (temporally) online-connection. This would also enable something like an (auto-)update-function to ensure that only the latest versions of an SSK are used.

References

1. LEADERSHIP SKILLS & STRATEGIES V U C A world [online]. Available: https://www.vuca-world.org/ (current March 2022)
2. Serrat O. (2017) Bridging Organizational Silos. In: Knowledge Solutions. Springer, Singapore. https://doi.org/10.1007/978-981-10-0983-9_77. (current March 2022)

3. Kowalczyk, M., Marcinkowski, B., Przybylek, A.: Scaled agile framework: Dealing with software process-related challenges of a financial group with the action research approach. In: Journal of Software: Evolution and Process, (2022) →in review
4. R. M. Parizi, T. J. Gandomani and M. Z. Nafchi, Hidden facilitators of agile transition: Agile coaches and agile champions, *8th. Malaysian Software Engineering Conference (MySEC)*, 246-250, doi: 10.1109/MySec.2014.6986022. (2014)
5. Poth A., Kottke M., Riel A.: Scaling Agile – A Large Enterprise View on Delivering and Ensuring Sustainable Transitions. In: Przybyłek A., Morales-Trujillo M. (eds) Advances in Agile and User-Centred Software Engineering. LASD 2019, MIDI 2019. Lecture Notes in Business Information Processing, vol 376. Springer, Cham. https://doi.org/10.1007/978-3-030-37534-8_1 (2019)
6. Moe, N. B., Olsson, H. H., & Dingsøyr, T.: Trends in large-scale agile development: a summary of the 4th workshop at XP2016. In Proceedings of the Scientific Workshop Proceedings of XP2016 (1-4). (2016)
7. Dikert, K., Paasivaara, M., & Lassenius, C.: Challenges and success factors for large-scale agile transformations: A systematic literature review. Journal of Systems and Software, 119, 87-108. (2016)
8. Przybylek, A., Albecka, M., Springer, O., Kowalski, W.: Game-based Sprint retrospectives: multiple action research. In: Empirical Software Engineering 27, 1, (2021), [Online]. Available: https://doi.org/10.1007/s10664-021-10043-z (current March 2022)
9. Poth A., Nunweiler E.: Develop Sustainable Software with a Lean ISO 14001 Setup Facilitated by the efiS® Framework. In: Przybyłek A., Jarzębowicz A., Luković I., Ng Y.Y. (eds) Lean and Agile Software Development. LASD 2022. Lecture Notes in Business Information Processing, vol 438. Springer, Cham. https://doi.org/10.1007/978-3-030-94238-0_6 (2022)
10. Patanakul P, Jiyao C, Lynn G.S.: Autonomous teams and new product development. Journal of Product Innovation Management 29(5). 734-750. (2012)
11. Lee LL.: Knowledge sharing metrics for large organizations. Knowledge Management: Classic and Contemporary Works, The MIT Press, 403-419. (2000)
12. Hung W, Jonassen DH, Liu R.: Problem-based learning. Handbook of research on educational communications and technology 3(1). 485-506. (2008)
13. Hmelo-Silver CE, Barrows HS.: Goals and strategies of a problem-based learning facilitator. Interdisciplinary Journal of Problem-Based Learning, 1(1): 21-39. (2006)
14. Chau T, Maurer F, Melnik G. Knowledge sharing: Agile methods vs. tayloristic methods. WET ICE 2003. Proceedings. Twelfth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, IEEE, 302-307. (2003)
15. Ellison NB, Gibbs JL, Weber MS.: The use of enterprise social network sites for knowledge sharing in distributed organizations: The role of organizational affordances. American Behavioural Scientist, 59(1). 103-123. (2015)
16. Paasivaara, M. and Lassenius, C.: Communities of practice in a large distributed agile software development organization - Case Ericsson, Information and Software Technology, vol. 56, pp. 1556-1577 (2016)
17. Cockburn A, Highsmith J.: Agile software development, the people factor. Computer, 34(11). 131-133. (2001)
18. Hoda R, Murugesan LK.: Multi-level agile project management challenges: A self-organizing team perspective. Journal of Systems and Software (117). 245-257. (2016)
19. Dorairaj S, Noble J, Malik P.: Understanding team dynamics in distributed Agile software development. International conference on agile software development, Proceedings, Springer, Berlin, Heidelberg, 47-61. (2012)
20. Poth, A., Kottke, M. and Riel, A.,: The implementation of a digital service approach to fostering team autonomy, distant collaboration, and knowledge scaling in large enterprises. Human Systems Management. 1-16. (2020)

21. Schank RC, Berman TR, Macpherson KA.: Learning by doing. Instructional-design theories and models: A new paradigm of instructional theory, 2(2). 161-181. (1999)
22. Hoic-Bozic N, Holenko Dlab M, Mornar V. Recommended system and web 2.0 tools to enhance a blended learning model. IEEE Transactions on Education 59(1). 39-44. (2015)
23. Hoic-Bozic N, Mornar V, Boticki I.: A blended learning approach to course design and implementation. IEEE Transactions on Education 52(1). 19-30. (2009)
24. Latchman H, Salzmann C, Gillet D, Bouzekri H. Information technology enhanced learning in distance and conventional education. IEEE Transactions on Education 42(4). 247-254. (1999)
25. Singh H. Building effective blended learning programs. Educational Technology 43(6). 51-54. (2003)
26. Williams SW.: Instructional Design Factors and the Effectiveness of Web-Based Training/Instruction. In: The Cyril O. Houle Scholars in Adult and Continuing Education Program Global Research Perspectives: Volume II. Compiled by Cervero RM, Courtenay BC, Monaghan CH, 132-145. (2002)
27. Dukhanov A, Karpova M, Bochenina K.: Design virtual learning labs for courses in computational science with use of cloud computing technologies. Procedia Computer Science 29. 2472-2482. (2014)
28. Raspotnig C, Opdahl A.: Comparing risk identification techniques for safety and security requirements. Journal of Systems and Software, 86(4). 1124-1151. (2013)
29. IEC 61508.: Functional safety of electrical/electronic/programmable electronic safety-related systems. International Electrotechnical Commission, 2nd ed. (2008)
30. Ashby W. Ross: Introduction to Cybernetics. Chapman & Hall, London. (1956)
31. Poth A., Jacobsen J., Riel A. : A Systematic Approach to Agile Development in Highly Regulated Environments. In: Paasivaara M., Kruchten P. (eds) Agile Processes in Software Engineering and Extreme Programming – Workshops. XP 2020. Lecture Notes in Business Information Processing, vol 396. Springer, Cham. https://doi.org/10.1007/978-3-030-58858-8_12. (2020)
32. Poth, A., Kottke, M. and Riel, A.: Measuring team work quality in large-scale agile organizations. IET Software, John Wiley & Sons, Inc. (2021)
33. Poth, A., Kottke, M. and Riel, A.,: The implementation of a digital service approach to fostering team autonomy, distant collaboration, and knowledge scaling in large enterprises. Human Systems Management. 1-16. (2020)
34. Poth, A., Kottke, M. and Riel, A.,: September. Scaling agile on large enterprise level with self-service kits to support autonomous teams. In 2020 15th Conference on Computer Science and Information Systems (FedCSIS) (pp. 731-737). IEEE. (2020)
35. Järvinen, P.: Action research is similar to design science. Quality & Quantity, 41(1), pp.37-54 (2007)
36. Chaohui Z.: A knowledge base with dependencies | IEEE Conference Publication. IEEE Xplore (2014). [Online]. Available: https://ieeexplore.ieee.org/abstract/document/6980868 (Current March 2022
37. Poth, A. and Riel, A.,: August. Quality requirements elicitation by ideation of product quality risks with design thinking. In 2020 IEEE 28th International Requirements Engineering Conference (RE). 238-249. IEEE. (2020)
38. Poth, A., Kottke, M., Heimann, C. and Riel, A.: The EFIS framework for leveraging agile organizations within large enterprise, XP'21 (2021)
39. Poth, A., Kottke, M. and Riel, A.: Agile Team Work Quality in the Context of Agile Transformations–A Case Study in Large-Scaling Environments, European Conference on Software Process Improvement. 232-243. (2020)
40. J. Becker, R. Knackstedt, J. Poeppelbuss. Developing maturity models for IT management. Bus. Inf. Syst. Eng., 1. 213-222. (2009)

41. Conway M. E.: How Do Committees invent?. (1968). [Online]. Available: https://hashingit.com/elements/research-resources/1968-04-committees.pdf (Current March 2022)
42. Bloom BS, Krathwohl DR, Masia BB.: Bloom taxonomy of educational objectives. Allyn and Bacon, Pearson Education, (1984)

Alexander Poth studied computer engineering at Technical University Berlin. He is IT Quality Manager in the Group IT at Volkswagen AG. He is product owner of Testing as a Service (TaaS) and cares about many other interesting things like the Quality innovation NETwork (QiNET) of Volkswagen to ensure that IT service/product development and delivery can fit the quality expectations.

Mario Kottke studied Business economics and computer science at Martin-Luther-University Halle. He is an Agile Coach at Volkswagen AG. His working field is the first contact for projects who want to work agile and consulting during the start. His experience is 4 years working as a scrum master and 3 years as an agile coach.

Andreas Riel received an MSc (Dipl.-Ing.) in Telematics from Graz University of Technology, Austria, a PhD in Meachtronics from Vienna University of Technology, Austria, as well as a habilitation (HDR) in Industrial Engineering from Grenoble Alps University, France. He is an independent coach, consultant and trainer in Systems Engineering, Quality Engineering, Innovation Management, as well as Digitalization and Industry 4.0 to major European industry, especially automotive. He is also a part-time Professor at the Institute of Engineering of Grenoble Alps University.

Received: November 12, 2021; Accepted: July 29, 2022.