Customizing Domain Analysis For Assessing The Reuse Potential Of Industrial Software Systems

Experience Report

Dominik Domis, Stephan Sehestedt, Thomas Gamer, Markus Aleksy, Heiko Koziolek

ABB Corporate Research Germany
Wallstadter Str.59
68526 Ladenburg, Germany
{firstname}.{lastname}@de.abb.com

ABSTRACT

In companies with a large portfolio of software or software-intensive products, functional overlaps are often perceived between independent products. In such situations it is advisable to systematically analyze the potential of systematic reuse and Software Product Lines. To this end, several domain analysis approaches, e.g., SEI Technical Probe, have been proposed to decide whether a set of products with a perceived functional overlap should be integrated into a single product line. Based on the principles of those approaches we devised our own approach. One important property is the inherent flexibility of the method to be able to apply it to four different application cases in industrial software products at ABB. In this paper we present our refined approach for domain analysis. The results and lessons learned are meant to support industrial researchers and practitioners alike. Moreover, the lessons learned highlight real-world findings concerning software reuse.

Categories and Subject Descriptors

D.2.13 [**Software Engineering**]: Reusable Software—*Domain engineering*; D.2.11 [**Software Engineering**]: Software Architectures—*Domain-specific architectures*; D.2.9 [**Software Engineering**]: Management—*Life cycle*

Keywords

Software Product Lines, Domain Analysis, Software Reuse

1. INTRODUCTION

Large companies often have perceived opportunities to introduce software product lines or systematic reuse where several products have functional overlap. ABB is a large corporate company with divisions and business units all over the world. The extensive portfolio of power and automation products has historically grown inside business units, e.g., to serve different markets and domains or due to mergers and acquisitions. This results in a large landscape of software intensive products such as automation controllers, or software tools for the engineering, configuration, monitoring, and control of devices. Some of these products have a perceived functional overlap, even within the same business unit or industrial domain. Naturally, this has attracted the attention of managers who want to assess the potential of systematic reuse between the different products in order to harvest the advantages, such as lower development and maintenance cost, or possibly higher product quality.

It is important to note that a perceived functional overlap is merely an indicator for reuse potential. Thus, the real opportunities need to be identified systematically before decisions are made. The potential benefits and drawbacks need to be assessed; arguments and rationale need to be provided before building reusable software components for many products or merging different products into a software product line.

For this purpose, we adopted the principles of domain analysis approaches such as the SEI technical probe [1] to devise our own approach and applied this in two industrial application cases on several ABB power and automation products. These two cases were previously described in [2]. Based on the findings, we adapted and extended the approach and used it in the analysis of two further domains of industrial power and automation. Particularly for practitioners, who want to identify and assess reuse potentials in a set of products, we present in this paper the improved and more time-efficient domain analysis approach as well as our experiences from both new application cases. Beyond this, we provide new lessons learned with respect to customization of the method, required efforts, and conducting the interviews. Furthermore, we present a summary of general and domain specific reuse success factors collected in the interviews of more than 20 products. At the same time, we give pointers for scientists where more research is required.

In [3], we collected empirical evidence with respect to five research questions and corresponding hypotheses from all four application cases. The hypotheses address the results of domain analyses, the correlation of a positive return on investment (ROI) and decisions for SPLs among business units, the correlation of required granularity with size and complexity of products, the projected return on investment,

SPLC '14, September 15 - 19 2014, Florence, Italy

http://dx.doi.org/10.1145/2648511.2648547.

and the reusability of domain analysis steps. However, in [3], we put less emphasize on the more practical contributions presented in this work.

The remainder of this paper is structured as follows: Related work is summarized in Section 2. Section 3 presents the adaptations to our domain analysis approach. Section 4 and 5 present the new application cases and the achieved results. Lessons learned and reuse success factors are provide in Section 6. Lastly, Section 7 concludes the paper.

2. RELATED WORK

A general overview of software product line engineering can be found in several publications [1, 4]. Linden et al. [5] provide an industrial perspective and the Software Engineering Institute (SEI) has cataloged numerous reports of industrial case studies in the Product Line Hall of Fame [6]. Khurum and Gorschek offer a comprehensive overview of domain analyses for software product lines [7]. The SEI Product Line Technical Probe [8] examines the readiness of a company to succeed with a software product line approach. The probe consists of a series of structured interviews followed by data analysis. The Family Evaluation Framework [5] follows the CMMI philosophy and assesses an organization's maturity for software product line engineering along different dimensions, e.g., business, architecture, or process. In each dimension, an organization is ranked within five levels, so that improvement potential can be identified. The Reuse Capability Model (RCM) [9] includes a model for assessing an organization's strength and improvement opportunities for reuse. Critical factors, such as management, application development, and process factors, are evaluated to implement reuse initiatives. Our approach follows similar steps as these approaches, but focuses more on technical aspects and high-level architecture.

PulSE Eco from Schmid [10] intends to analyze functional overlaps between different products and consists of three steps: 1) Mapping relevant products and their corresponding features, which are grouped into feature domains and subdomains, in a product or feature map. 2) Assessing the benefits and risks of including a feature domain in a software product line. 3) Performing a quantitative evaluation of the features with respect to required effort and business goals. We applied step 1 and partially step 2 of PulSE, as described by John et al. [11] as a light-weight variant.

There are automated and semi-automated approaches for identifying reusable components and features. For example, MAP [12] semi-automatically extracts architecture information from code, identifies patterns and styles, and evaluates the potential for software product lines. Frenzel et al. [13] use reflexion and clone detection for identifying ad-hoc copied code between different products. However, due to the heterogeneous landscape of ABB's legacy systems being based on diverse technologies and programming languages, we decided to use a manual, interview-based approach for architecture reconstruction.

3. DOMAIN ANALYSIS

The aim of the domain analysis is to compare a set of products with regard to their relevant criteria in order to identify, assess, and recommend viable scenarios for systematic reuse, ranging from single components to a full Software Product Line. The analysis consists in general of eight steps grouped into three phases, as listed in Table 1. The order, concrete implementation, and execution of the steps is not fixed. In [2], we found out that preparing larger parts of the interview documents in advance based on the available documentation lets us collect better and more detailed information in the interviews. The more detailed preparation facilitated the understanding of the products and allowed for starting their comparison directly in the interviews. This improved the results of the evaluation and saved efforts in the interview phase. Beyond this, we put more emphasis on adapting and extending the approach according to given requirements and the prioritization of reuse potential criteria for each domain. The resulting process looks in brief as follows. More details are provided in Sections 4 and 5, describing the two new application cases.

Table 1: Domain Analysis Approach

Phase	Nr.	Step
Preparation	1	List products and information sources
	2	Establish criteria for reuse potential
	3	Collect and analyze documentation
	4	Prepare intitial interview documents
Interview	5	Conduct interviews
Evaluation	6	Evaluate results and identify opportunities
	7	Create business case
	8	Handover results and discuss actions

Table 2: Generic List of Reuse Potential Criteria

Category	Example reuse potential criteria
Business	purpose, scope, market, business model, business case
Technical	features, architecture, technology, standards
Processes	process maturity, release and change management
Organization	units, roles, funding
Reuse culture	knowledge, motivation, lessons learned
Outlook	reuse potentials and future trends

The PREPARATION PHASE starts with identifying and listing the products to be analyzed as well as corresponding information sources (Step 1). Usually an initial list is provided by the management team of a business unit, who need to agree on the final scope of the analysis and who need to nominate responsible persons for supporting the analysis. Another part of the scope discussion is to agree on and prioritize the relevant reuse potential criteria to be considered in the analysis (Step 2). Based on literature [5, 1] and our experiences from the previous case studies [2], we created the generic list of reuse potential criteria, as shown in Table 2. These are used as input for the discussion. However, new reuse potential criteria can be added in Step 2, if required, for a particular domain analysis. In Step 3, documentation such as user manuals, requirements, or architecture documents are collected for analysis from the product stakeholders or intranet.

Based on the reuse potential criteria and their priorities defined in Step 2, we derive a specific questionnaire from a generic questionnaire in Step 4. Questions may be prioritized or removed altogether and preliminary answers may be entered. If we receive enough documentation about the architecture, we prepare a preliminary architecture sketch (see Fig. 1) in the Fundamental Modeling Concept (FMC) notation [14]. We use FMC, because it has only few modeling elements compared to UML and is quickly understood also

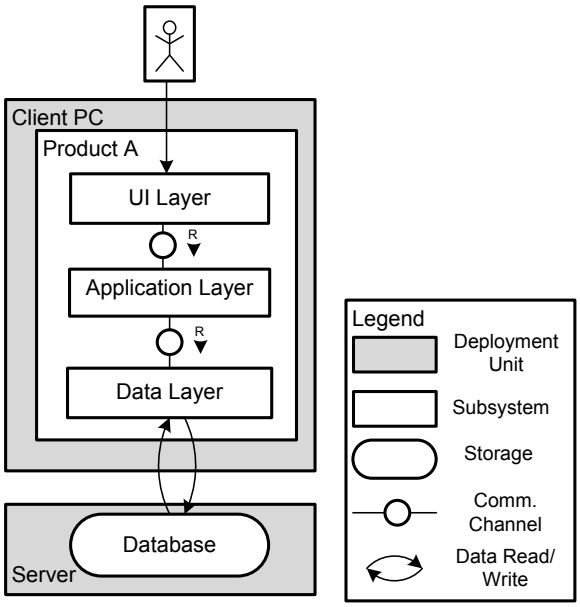

Figure 1: Obfuscated excerpt of an architecture sketch from an interview in FMC notation [14]

Table 3: Obfuscated excerpt of a feature map

Feature Domain	Feature Sub-domain	Feature	Product 1	Product 2	Product 3	Product 4	Product 5	Product 6
Domain 1			◑	○	●	◑	◑	◑
	Subdomain 1.1		●	●	●	◑	◑	◑
		A	●	●	●	●	●	●
		B	●	●	●	○	○	○
		C	●	●	●	●	○	●
		D	●	●	●	○	●	○
	Subdomain 1.2		○	○	●	○	○	○
		E	●	○	●	○	○	○
		F	○	○	●	○	○	○
		G	○	○	●	○	○	○
		H	○	○	●	○	○	○
Domain 2			●	◑	●	○	◑	◑
	Subdomain 2.1		●	◑	●	○	◑	◑
		I	●	●	●	○	●	○
		J	●	○	●	○	○	●
		K	●	○	●	○	○	○
		L	◑	●	○	○	●	○
	Subdomain 2.2		◑	○	●	○	◑	◑
		M	●	○	●	○	○	●
		N	○	○	●	○	●	○

Legend			
●	supported	○	not supported
		◑	partially supported

by non-computer scientists [2]. Even when UML or other models are already available, redrawing the high-level architectures in FMC helps to describe all products on a similar level of abstraction.

Features are documented in a feature map or table [10], which shows for each product (as columns) the supported feature domains, feature subdomains, and features (as rows) (see Table 3). Additionally to the product documentation, industry standards, similar products, and current technology trends also serve as input for the feature map.

It is advisable to time-box Step 3 and 4 to avoid spending too much time. A trade-off must be found between the prioritized reuse potential criteria, the available documentation, and the availability of the domain experts in the interviews.

In Step 5, the INTERVIEWS are usually conducted as face-to-face meetings at the development site of the respective product by one or two interviewers. Together with the interviewees, we go through questionnaire, architecture, feature map, and the other artifacts. The interviewee(s) might be different for each reuse potential criterion, in order to save efforts. For example, for drawing the architecture only the architect is required. If some things were prepared beforehand, such as interview answers, the feature map, or the architecture sketch, we verify these together with the interviewee(s) in the interview and add or correct information. In between the interviews of the different products, artifacts such as the feature map, which are used again in the next interview, are already cleaned up and pre-analyzed.

In the EVALUATION PHASE, questionnaire, architecture, and feature map are cleaned up. Then, the products are compared with each other to identify and assess reuse opportunities (Step 6). These might be reusable components, or starting or extending a software product line. For each identified potential reuse scenario, we describe assumptions, scope, as well as pros and cons with regard to the reuse criteria. If the business units define this as important reuse criteria, we calculate the return on investment (ROI) or net present value of promising reuse scenarios in Step 7. For this purpose, we are using, e.g., the formulas and assumptions defined by Boeckle et al. [15]. In Step 8, the results of the domain analysis are documented in a report and presented and discussed in a final presentation or workshop with managers and interviewees. The goal of the final meeting is to agree with the management on concrete follow up activities, e.g., to implement the identified reuse scenarios that have a high potential with regard to the defined reuse potential criteria.

4. APPLICATION CASE 1: DEVICE ENGINEERING TOOLS

4.1 Domain and Challenges

ABB sells products and solutions in various industrial sectors, e.g., process and factory automation, power generation including renewables, substations, oil & gas. These products and solutions need to be configured, commissioned, and maintained by ABB, customers, or integrators by using dedicated PC tools. In this application case, we analyzed four different ABB device engineering tools of a single industrial sector.

We started the domain analysis with a set of three existing tools, which are developed by three different teams at three different sites and which are owned by two different business units. All three tools have a history of several years and versions. Tool A is based on an outdated technology and needs to be reimplemented. Tool B has been developed for a particular market segment. However, due to the products' success new market segments will be addressed requiring new tool variants. Tool C is already a software product line: A PC tool framework that is extended by dedicated plug-ins for configuring dozens of different device types. Based on the intermediate results after analyzing the three tools, the

management of the industrial sector decided to extend the domain analysis with a fourth new tool (D) that is planned to be implemented in the coming years based on its market and technical requirement specification. This is to ensure that this new product can directly benefit from and support the systematic software reuse opportunities that are identified during the domain analysis.

4.2 Domain Analysis

1) List products and information sources.

We received a list of relevant ABB software products in the target industrial sector from the involved managers. This initial list was discussed and then reduced to only focus on the already mentioned four device engineering tools based on their assumed functional overlap and assumed roadmaps. The managers also named contact persons for each of the tools.

2) Establish criteria for reuse potential.

We started with the generic reuse criteria from Table 2. Due to their long history, the management asked us to put specific emphasis on lessons learned from the existing products and on the future roadmaps. Finally, as management appeared to be open minded towards reuse for future developments, reuse culture also received a high priority. As a result, processes and market fit received lower priority to still be able to execute the entire domain analysis process in time.

3) Collect and analyze documentation.

As a refinement of the originally described process, we had a kick-off video meeting of 30 minutes with the named contacts, the architect and the product manager, separately for each product. There, we motivated the domain analysis, explained the management motivation, and already clarified, which inputs and documents would be required from them before and during the interviews. Furthermore, we identified who of our contacts would be the main interviewee for each of the different reuse criteria.

We received documentation on the different engineering tool products as well as the devices to be engineered, e.g., product manuals, specification sheets, and getting started guides. For the tool framework, we also received detailed internal technical documentation, such as an architecture review of a previous tool version.

4) Prepare initial interview documents.

Based on the received documentation, particularly the extensive documentation of the device engineering tool framework (C), we were able to create an initial feature map with roughly 200 features categorized into 10 feature domains. Each feature domain contained several feature subdomains, features, and sometimes subfeatures. The initial feature map already covered approximately 80-90 % of the final features. Mainly the naming and grouping of features into feature domains and subdomains had to be revised during and after the interviews. For extracting features from free text documents such as user manuals, we used the patterns defined by John [16]. Even though the patterns are straight forward such as searching features in headings of sections, they provide good guidelines particularly for newbies to domain analysis.

For the framework, we created a first architecture sketch, based on the received architecture documentation and the old architecture review. Moreover, the architecture review document in particular also contained known challenges and limitations as well as unique selling points and opportunities, which helped us to prepare questions for the interviews. For the other two existing products, the architecture sketches had to be drawn in the interviews.

5) Conduct interviews.

We scheduled one and a half days for the interviews with lead architect, product manager, and line manager for each product. We started each interview with a short tool and device demonstration to get a common understanding and a better idea about the functionality of the most important features. During the interviews we spent most time discussing and refining the feature list, followed by reconstructing the architecture, lessons learned, future roadmap, as well as reuse culture.

The detailed initial feature map of the previous step accelerated the interviews and facilitated deeper discussions with the interviewees. As a side effect, the feature map enabled an early and quick clarification of wording and terms during the interviews. When analyzing multiple products it is essential to develop and maintain such a shared glossary. To avoid results of one product influence the interview results of another product, the interviewees were only shown their respective tool in the feature map.

In the interview for the new product, the feature map was also valuable to give the product manager an idea about feature domains that are also relevant for his product, but have not yet been in the focus of the requirements specifications. For the tool framework (C), we discussed the pre-prepared architecture sketch and open questions from the architecture review with the architect. We corrected several aspects in the architectural model that have been changed during the latest releases of the platform, partially triggered by the architecture review. Due to these changes and the large experience of the architect from the long history of the framework, we collected a large number of positive and negative lessons learned from building and maintaining such a platform.

For the two existing tools A and B, we created the architecture sketches in FMC notation from scratch during the interviews and already included their ideas for extending their products in the future. We used the CrossModel Architecture Discovery Pattern Language from Fairbanks [17] to prepare the reconstruction sessions. In particular, the Refinement Patterns and Expert Interviewing Patterns were helpful. While most of the CrossModels patterns are quite generic and mainly based on common sense–e.g., Many Eyes, Gain Consensus, Technical Rationale or Single Level of Abstraction–it still was very useful to have such guidelines and be able to prepare for the architecture reconstruction session. Moreover, there are controversy patterns available, e.g., Blackbox-Whitebox Refinement or Passive and Active Facilitation for Expert Interviews, which allows for catering to the respective personality type of the architect. For the new tool D, no architecture sketch could be drawn as the development was still in a very early stage.

To gather lessons learned, future roadmap, as well as reuse culture, we prepared some open questions in the questionnaire such as “what are positive lessons learned?” and “what

are negative lessons learned?". In all interviews, this initiated productive brainstorming and discussions of up to two hours, e.g., for the lessons learned, resulting in long lists of valuable findings.

In one interview, we were not able to complete the questionnaire in the one and a half days on site. Therefore, we performed two two-hour video conferences to complete the questionnaire afterwards. These video sessions were less productive and produced not as good results as the face-to-face interviews.

6) Evaluate results and identify opportunities.

In the feature map, the feature support of all products was compared, some duplicated features identified and merged, as well as the grouping of features in feature (sub-)domains revised. Afterwards, we checked and documented the overlap of the products for each feature domain. The result was a large overlap in the basic engineering tool features of this particular industrial sector such as several dialog types as well as device communication related features. However, each product supported unique feature domains. Each of these unique feature domains reflects a particular aspect (or feature) of the respective device to be configured by the tool. There was only a medium functional overlap between the different tools, i.e., less than 50 % of the features.

The architecture sketches did not require extensive rework. Architecture and technology of all considered products are quite different in the current versions and thus do not suggest a high reuse potential. However, some components might fit for reuse or re-factoring in a new common platform.

We consolidated and documented the lessons learned and reuse culture in a report and derived recommendations for systematic reuse. We also documented successful systematic reuse cases in the history of the products as well as known organizational and technical learnings and challenges. Furthermore, we compared the future roadmaps of the products and identified strong relations as well as many ideas which show that the overlap between the device engineering tools will probably increase in the future, due to a stronger collaboration and alignment of the tools as well as of the devices to be engineered by the tools.

7) Create business case.

Based on the results of the domain analysis, in particular the future roadmaps of the products, the management of the business units created the business case for a common tool platform. In this calculation, the potential savings due to more reuse as described in literature [15] is only one factor to be considered. Additional factors are addressing new markets as well as gaining more market share in existing markets. These existing markets could benefit from a better integration of the tools and products through better usability and easier integration of new tool features.

8) Handover results and discuss actions.

We held a one-hour telephone conference to present and discuss our findings. Furthermore, a written report, the architectural maps of the tools, as well as the feature maps were provided to the stakeholders.

4.3 Results

Based on the results of our domain analysis, the management of the business unit created the business case calculation and decided to go for a common platform for the three individual tools A, B, and D, due to their future roadmaps and intended increased functional overlap and integration. The fourth analyzed product, the existing device engineering tool framework, will be continued as a separate platform as it addresses a different market segment and would add significant complexity to the platform development.

It has been decided to develop the new platform architecture as a framework with defined extension points (hot spots). Tools derived from the platform would be realized via these extension points. In order to support the development of the new common device engineering tool platform, we extracted high level requirements from the feature map and the future roadmaps of the products. This information is to be used as input for the market and technical requirements specification of the platform. The high level requirements cover both commonalities as well as tool/market/customer specifics. Currently, we support the business units in developing for the platform: 1) the feature model [18], 2) the common data model with variable and common parts, as well as 3) the platform architecture.

5. APPLICATION CASE 2: ENTERPRISE INFORMATION SYSTEMS

5.1 Domain and Challenges

In this application case, we analyzed a set of ABB enterprise information systems, which are used to store large data sets and to provide them to different kinds of users via corresponding views, filters, and search functions. The analysis comprised a family of five products and additionally, two individual products that have grown independently of each other. Today, the two individual products (A and B) and one product from the family (C) are used to support similar business processes, i.e., some users require all three systems to manage related sets of data for particular tasks. The four remaining products of the family have different users each and handle independent data.

The product managers wanted to know whether and how the three related products could be merged, integrated, or built on the same platform. Assumed potential advantages were, for instance, reduced development and maintenance efforts and easier usage.

To better understand the relationships of the considered enterprise information systems, we extended the domain analysis: We used a comparison of the database schemata to analyze the relation and overlap of the data managed by the products. Additionally, we conducted an evaluation of the underlying business processes to assess the tool overlap on this level as well as the impact of merging the tools.

5.2 Business Processes

An overview of different business process definitions can be found in [19]. Davenport and Short [20] define business processes as a *"... set of logically related tasks performed to achieve a defined business outcome."*. According to the authors, information technology (IT) should support new or redesigned business processes while business processes should consider the capabilities provided by IT.

Table 4: Used business process template

ID	Business Process	Roles	Input	Processing	Output
1	BP_1	role_X	system_U: data_Z	action_7	system_V: data_B
2	BP_2	role_Y	system_V: data_B	action_8	system_W: data_A

Usually, the business process life-cycle starts with process discovery and should consider various criteria [21], such as the process space, the process topology, and the process attributes. Process space describes all the relevant processes and their integration points. Process topology describes the process steps or activities and the flow logic. Process attributes (e.g., owner and purpose of the process) and activity attributes (e.g., roles, resources, data) provide detailed information required for process analysis.

In our domain analysis, we discovered and analyzed the business processes of different products on a high-level to better understand and compare the required functionality of the products, to identify their relations, commonalities, and differences, and to assess whether the products can be integrated, merged, or share common features. For documenting the business processes, we did not use a comprehensive model such as the Business Process Model and Notation [22], because these are too detailed for our interviews. Therefore, it was more practical to use the business process description shown in Table 4. The table lists all business processes that are supported by a particular resource, i.e., IT product. For each business process, the roles of all people that perform this process are summarized as well as the required data inputs (incl. source systems), the processing step(s) performed by the product, and the data outputs generated by the product (incl. receiving systems). The details of the business process activities and flow logic are not relevant for the domain analysis, in this step.

5.3 Domain Analysis

1) List products and information sources.

The product managers provided the list of products to analyze as well as the contact persons. The development of the product family and the two individual products was distributed among three different teams at three different sites.

2) Establish criteria for reuse potential.

We conducted a 30 - 60 minutes kick-off meeting via phone or video conference with each product's responsible person to explain and plan the analysis and to discuss the reuse potential criteria based on the generic list in Table 2. The product responsible persons decided that all criteria should be considered. High priority was given to the criteria features and architecture. Furthermore, two new criteria were added with a high priority: Data models and business processes.

3) Collect and analyze documentation.

In the kick-off meeting, we used a table to collect for each reuse potential criteria the available documentation as well as the roles and persons to be interviewed. After the kick-off meeting, the contact persons provided the respective documentation. It included product overviews, user manuals, database schemata, UML use case descriptions, and high-level management summaries of the architectural descriptions.

4) Prepare initial interview documents.

From the collected documentation, we were able to prepare approximately 70 % of the final feature map and significant parts of the architecture sketches before the interviews. Similar to the feature map, we prepared a table that mapped the data model entities (rows) to products (columns). We mainly used the product overviews and user manuals of the products as input for the data entity map. The database schemata were too detailed and complex for a high level mapping.

Based on use case documentation, we were able to create a draft version of a business process description of one of the products. The corresponding document already included a subset of the involved user roles and information how the product is used. This information can serve as a starting point for a business process comparison. However, it provides only a subset of the required information. One reason for this is the fact that a use case describes only the corresponding interactions with the involved system(s). Thus, activities that are not directly connected to the functionality of the system(s) are not covered. This fact was of special interest because we considered multiple systems in the domain analysis. Therefore, further investigations were needed for which we created a template document (cf. Table 4).

5) Conduct interviews.

The interviews were separated into two parts. As for the other application case, we performed the interviews on site together with the interviewees. The focus of the interviews was, as defined by the criteria priorities, the questionnaire, the feature map, the data entity map, and the architecture sketch. We split off the interviews for the business processes, as they were performed by one of our experts for business process analysis and as we required several additional product experts and users as interviewees.

The ON-SITE INTERVIEWS required one and a half days for each product. We interviewed an architect and a product manager or owner of each product or family. In the first step, the interviewees gave us a 15 minutes live demonstration of the products, which increased our understanding and started useful discussions, e.g., about the purpose and usability of some features. Then, we went through the questionnaire based on the previous criteria prioritization. Subsequently, we reviewed the preliminary feature map (created beforehand) and added missing feature support. In each interview, we showed the feature support of all products to the interviewees. In this way, the interviewees were able to compare the feature support of their product with the other products. Some interviewees disputed the features supported by other products or argued that despite missing support in their own product, a work around would be possible. These discussions were valuable, as they allowed to cross check the feature map and to achieve a better understanding of the differences between the products. It also supported the mapping of features that have different names in different products. This review took two to three hours for each tool.

Between the interviews, we cleaned up the feature map. The final feature map contained more than 250 features grouped into feature domains and subdomains.

In the same way as for the feature map, we went through the data entity map and checked and marked the data model entities that are managed by a product. We also merged data entities that have different names in different products, but a similar meaning. This step took half an hour using an 80 % completed table as a starting point.

In the last step of the interview, we reviewed the pre-prepared FMC architecture sketches and completed them together with the interviewees. The original architecture documentations used UML and proprietary notations and described the architecture on different levels of abstraction for each product. Representations of the architectures could be created on a similar abstraction level using FMC. Importantly, all interviewees understood the notation immediately and were able to provide corrections and additions. For supporting the review, we asked questions regarding the understanding of the architectures as well as about parts where we suspected missing details. In this way, we were able to review and complete the architecture sketch of each tool in one to two hours. After the interviews, we sent the questionnaire, the feature map, and the architecture sketch to the interviewees for offline review and approval. Only minor corrections were required in this step.

The Business Process analysis was performed iteratively in three to four video conference interviews with domain experts of each product. Each interview took up to two hours. In advance, a template or draft document together with corresponding questions was sent to the interview partners, who provided their feedback by e-mail. Afterwards, it was consolidated by the business process expert and discussed in the forthcoming video conferences. This procedure was repeated three to four times until the business process descriptions of all systems reached sufficient levels of quality and granularity to compare them with each other. In the end, we collected around 30 business processes for each product. Because we performed the business process interviews iteratively via phone or video conference and relied completely on their results, we decided to not show the business process of one product to the domain experts of another product, in order to avoid mutual interferences. Hence, the mapping of the business processes had to be performed afterwards in step 6.

6) Evaluate results and identify opportunities.

After the interviews, we compared the products with each other with a particular focus on their business processes, features, architectures, and data models. For comparing the business processes, we mapped them onto a related reference business process specification and visualized the comparison as a table. In this way, we identified a large overlap between the individual product A and product C from the family, although some processes are instantiated differently.

We cleaned up the feature map again and restructured the assignment of features to feature domains and subdomains according to, e.g., the product feature support. We assessed the functional overlap for each feature domain and subdomain by rating it high, medium, and low and added a rationale statement. As for the business processes, the individual product A and product C from the family showed a large functional overlap on a generic level, i.e., both products have implemented the features differently, but if the products would be merged, they could use the same implementation. The functional overlap inside the product family is lower and mainly covers basic tool functionality such as open, edit, and store data sets. The same is true for B with respect to A and the product family.

From the architecture sketches, we compared logical structuring and used technologies. The two individual products and the tool family had been implemented in different technologies, which hinders direct reuse of components between the products. The logical structures of the products were also different, but the cores of the individual product A and the product family were similar and provide potential for a common platform.

For comparing the data models, we reviewed and compared the data entity map from the interviews and, based on this, the database schemata of the products with each other. The structure of the models as well as the exact names and types of the entities were different. We identified similar elements in different models, i.e., the entities have different names with a similar meaning and their types are also similar or can be transformed into each other. The largest overlap was found between the individual product A and product C. The models are different in structure and details, but they are similar enough to merge them into a single common model in the future. The overlap with product B is very low and non-existent with the rest of the tool family.

In conclusion, the comparison of the business processes, features, architectures, and data models has shown a large overlap between the individual product A and product C of the product family. In some business processes, Product B exchanges data with product A and product C, but this covers only a small part of their data models and they only have some basic features in common. The product family shares only the basic functionality and no data or business processes.

Based on these results, we elaborated several scenarios for integrating the products: 1) data integration of A and C via corresponding interfaces, 2) merging A and C into a single product, 3) building A and C based on a common platform, and 4) building a software product line for implementing product A, product B, and the family (including C). For all scenarios, we listed the most important advantages and disadvantages such as usability, complexity, maintainability, time to market, and investment cost. We also sketched migration roadmaps between the scenarios.

7) Create business case.

We performed a return on investment calculation for the common platform scenarios 3) and 4), because they seemed to be the most interesting ones based on the intermediate results. The business units provided the past development and maintenance costs for the calculations, and we estimated a 50 percent higher effort for building generic, shared parts instead of specific, separated parts as in literature [15]. Although the feature map indicated a high functional overlap, we conservatively assumed the overlap to be 50 percent after consulting the architects. With these and other assumptions, we expected a positive return on investment for scenario 3) after seven years at latest. Due to the larger

number of products, the return on investment of scenario 4) would likely be sooner.

8) Handover results and discuss actions.

The results of the domain analysis were documented and discussed in an additional meeting with the interviewees, product managers, and other responsible stakeholders from the business units.

5.4 Results

The business unit first implements scenario 1) (data integration) for better usability of products A and C in common business processes and will align their business processes for preparing a deeper integration scenario. After the alignment of the business processes, the proposed integration scenarios will be reassessed for the long term planning.

6. LESSONS LEARNED

Required efforts.

Table 5 provides an overview of the four domain analyses we conducted in the past three years. The cases *Industrial control systems* (A) and *Commissioning and monitoring tools* (B) were previously presented in [2]. The rows *# products*, *# developers per product*, and *Lines of code per product* are an indication of the relative magnitude of the application cases.

In case A, the efforts for *Evaluation* are very high compared to the other cases. This has mainly two reasons. First, the *Industrial control systems* were the most complex products we analyzed. Second, we applied the domain analysis the first time in this application case. We extensively compared the products with regard to all reuse potential criteria and created a detailed documentation of the high-level architecture of each product. In case B, we had only a very small budget and, therefore, prioritized the reuse potential criteria and were very efficient during the interviews, which resulted in the low efforts for *Evaluation*. In case C and D, we applied the improved approach, customized and prioritized the reuse potential criteria, and invested more time in the preparation phase. In case C, one product was added during the analysis based on the intermediate results, which increased the efforts of the *Evaluation* compared to case B and D.

There are many factors that influence the time required for a domain analysis and Table 5 can only give some orientation. However, the table shows that one can estimate per product up to one week for preparation and up to two days for the interviews. The evaluation should be planned with two to three weeks for similar cases as B, C, and D.

Customization.

Our experiences show that the domain analysis can and should be customized regarding the prioritization of reuse potential criteria. This is critical in order to produce relevant results within the given budget. In the case *device engineering tool* (C), we focused on future roadmaps, lessons learned, and feature map and put less emphasis on markets and architecture. In the case *enterprise information system* (D), the main focus was on feature map, architecture, and the new criteria business processes and data model. In the case *commissioning and monitoring tool* (B), features and architecture received the highest priority. In the case *industrial control system* (A), which was the first application case, all reuse potential criteria from Section 3 have been analyzed with a particular focus on comparing the architectures.

Preparation based on documentation.

Preparing feature map, answers in the questionnaire, and architecture sketches based on provided documentation before the interview is our preferred approach. The reasons are twofold: First, the interviewers start with a more broad and deep understanding of the domain and the respective products, enabling them to ask the right questions. Second, in our experience the time the stakeholders can be expected to invest in the interviews is limited. A thorough preparation ensures that there will be enough time to deal with the high priority reuse criteria sufficiently rather than having to cover many basics first.

Importance of interviewers and interviewees.

Although we are using a questionnaire with many comments and answer hints for each question, the quality of the results very much depends on the experience and skill of the interviewers in domain analysis and, of course, on the depth of knowledge of the interviewees regarding the product. An interview can in principal be conducted by a single interviewer. However, having a second interviewer allows for a better pace and more follow up questions and challenging of answers, while still being able to document all important aspects during the interview. Hence, we recommend having two interviewers.

A lot of domain knowledge is gained in each interview. Therefore, it is beneficial to have one interviewer to attend all interviews. The acquired knowledge helps in conducting the following interviews and in achieving comparable results for all considered products. Another important aspect is the role of the interviewees. Preferably, two persons should be interviewed to capture both technical and business perspective. Architects and developers have significant technical knowledge and are most often interested in reuse. Product managers provide information about customers, markets, business and organizational aspects. However, technical discussions may suffer from political influences.

Phone and video conferences cannot substitute face-to-face interviews.

Although we tried to perform all interviews as face-to-face meetings, we had to use phone and video conferences in addition. In each of these cases, the interviews were not as productive as face-to-face meetings and did not provide the same quality of results. This is particularly true for the feature map as well as for drawing the architecture sketches. This is an important observation for globally distributed companies and we recommend to have face-to-face meetings at least for certain aspects of the domain analysis.

Independence vs. comparability of results.

It has to be decided whether or not to share the results of previously conducted interviews with the interviewees. This is especially true for the feature map, but also for the other artifacts. The trade-off is to prevent that the results of the different interviews mutually influence each other versus the

Table 5: Overview of the application cases

Application case	A: Industrial control systems	B: Commissioning and monitoring tools	C: Device engineering tools	D: Enterprise information systems
Type of systems	Client-server	Desktop applications	Desktop applications	Web client + server
# products	6	4 + 2 families of 10	3 + 1 framework	2 + 1 family of 5
# developers per product	<10 to >40	up to 10	up to 5	up to 3
Lines of code per product	1 - 6.5M	300K - 2M	300 - 500K	10 - 100K
Required Efforts				
Preperation per product	1-3 days	1-2 days	1-5 days	1-5 days
Interview per product	1-2 days	1-2 days	1-2 days	2 days
Evaluation	15 weeks	1 week	3 weeks	2 weeks

efforts required for the interviews as well as for information consolidation and analysis after the interviews.

If the involved parties are mostly interested in reuse opportunities, sharing the results is recommended. Especially for the feature map we observed that most of the data consolidation gets done during the interview with very little overhead. However, if there is a lot of competition between the products it is better to not share the results until the final report is ready for review. As these are only our observations more empirical evidence on the matter would be very welcome.

Market share is important in business case.

In three of our four application cases, we analyzed a set of PC tool products which belong to the same sector or subdomain of industrial automation. For each of these three sets of tools, reduced maintenance costs are only one factor in the motivation and business case for systematic reuse or software product lines. Of equal or higher importance are opportunities to increase market share and revenues by better meeting customer needs. Moving to a common platform could achieve that by 1) sharing the best features between the tools, 2) providing a common look and feel, and 3) providing new features for seamless integration or improved workflows. Hence, reduced development and maintenance cost as well as customer needs have to be taken into account.

Success factors for systematic reuse.

A lot of pitfalls and success factors for Software Product Lines can be found in literature [5, 1, 23]. In the domain analysis interviews, we collected several lessons learned, which confirm these factors and give an additional industrial power and automation domain perspective on some of them:

1) Components must be *systematically* developed for reuse; ad-hoc reusing arbitrary artifacts often failed.

2) Systematic reuse and, in particular, software product lines need a clear decision and support from higher management, which needs to set the priorities for the involved people and projects accordingly. Because there is a high risk that the development splits again after the first common version, *management support* is required over the entire lifetime of the SPL and needs a long term focus.

3) The *organization and processes* need to be defined at the beginning of a software product line. Often mentioned as important were: communication between platform engineering and product engineering, deciding about new features, release planning, maintenance, and how new stakeholders can join a running project. Particularly, if a product line covers multiple organizational units, a platform manager is required, who is perceived as politically unbiased and is able to align the requirements and expectations of the different product units.

4) The *funding* of the development and maintenance of reusable artifacts needs to be set up [24].

5) Previous negative experience increases reluctance for reuse, i.e., seeing only additional efforts and overhead, but not the overall benefits. This must be addressed very early by transparency and by providing "personal" motivation for all *involved people*.

6) It is more important that a software product line fits into the *business models* and roadmaps of the participating business units than being technically feasible, as technical challenges can most often be solved.

7) The platform needs to be continuously re-factored to keep it maintainable and usable by the existing and new products. Because this requires a significant amount of time, it is important to explicitly plan the re-factoring steps.

8) A strategy needs to be in place on how to handle backward and forward compatibility between different versions of platform, products, devices, communication, and data models. For example, full backward compatibility between platform and older versions of different devices to be engineered requires testing a large number of combinations, which might become very expensive.

9) For engineering tool frameworks, there must be a strong governance of interfaces and guidelines (e.g., UI) in order to efficiently utilize the platform, avoid double work, and provide a common look and feel.

10) Separating data model, business logic, and tool framework facilitates reuse particularly for engineering tools.

11) Using a modern technology is risky but often made the platform more sustainable and increased its lifetime.

7. CONCLUSIONS

In this paper, we presented our new domain analysis approach, which we improved based on the findings from two former application cases [2]. The proposed changes increase the time efficiency of the interview-based approach by investing more time in the interview preparation. This improves the results of the interviews and facilitates their evaluation as well as the comparison of the products. Beyond this, we put more emphasize on adapting the approach for different application domains by prioritizing reuse criteria for the analysis. In order to help other practitioners to perform domain analyses in large to medium-sized companies, we particularly focused on practical and concrete aspects such as execution of the interviews, selection of involved people, used artifacts, and required efforts. While applying our improved domain analysis approach on two further ABB application cases, we collected various lessons learned, including success factors for systematic reuse. Presenting these in this paper, practitioners as well as researchers might benefit in their future work.

Currently, the results of the domain analysis of the *device engineering tool* case are being used in the development of a new tool platform. Beyond this future work encompasses investigating patterns and automation for increasing the efficiency of feature analysis and architecture reconstruction.

8. REFERENCES

[1] Paul Clements and Linda Northrop. *Software Product Lines: Practices and Patterns*. Addison-Wesley, 2001.

[2] H. Koziolek, T. Goldschmidt, T. de Gooijer, D. Domis, and S. Sehestedt. Experiences from identifying software reuse opportunities by domain analysis. In T. Kishi, S. Jarzabek, and S. Gnesi, editors, *SPLC*, pages 208–217. ACM, 2013.

[3] H. Koziolek, T. Goldschmidt, T. de Gooijer, D. Domis, S. Sehestedt, and M. Aleksy. An exploratory case study on domain analysis to identify reuse potential. *Submitted to the Journal of Empirical Software Engineering – Special Issue on Empirical Evidence in Software Product Line Engineering.*

[4] Klaus P., Günter B., and F. van der Linden. *Software Product Line Engineering: Foundations, Principles and Techniques*. Springer, 2005.

[5] Frank J. van der Linden, Klaus Schmid, and Eelco Rommes. *Software Product Lines in Action: The Best Industrial Practice in Product Line Engineering*. Springer, 2007.

[6] Carnegie Mellon University - Software Engineering Institute. Product Line Hall of Fame. http://splc.net/fame.html, 2013. last visited 2013-01-21.

[7] M. Khurum and T. Gorschek. A systematic review of domain analysis solutions for product lines. *J. Syst. Softw.*, 82(12):1982–2003, December 2009.

[8] Carnegie Mellon University - Software Engineering Institute. Software Product Lines. http://www.sei.cmu.edu/productlines/, 2013. last visited 2013-01-21.

[9] T. Davis. The reuse capability model: a basis for improving an organization's reuse capability. In *Software Reusability, 1993. Proceedings Advances in Software Reuse., Selected Papers from the Second International Workshop on*, pages 126 –133, mar 1993.

[10] K. Schmid. A comprehensive product line scoping approach and its validation. In *Proc. 24th Int. Conf. on Software Engineering*, ICSE '02, pages 593–603, New York, NY, USA, 2002. ACM.

[11] I. John, J. Knodel, T. Lehner, and D. Muthig. A practical guide to product line scoping. In *Software Product Line Conference, 2006 10th International*, pages 3 –12, 0-0 2006.

[12] C. Stoermer and L. O'Brien. Map - mining architectures for product line evaluations. In *Software Architecture, 2001. Proc. Working IEEE/IFIP Conference on*, pages 35 –44, 2001.

[13] P. Frenzel, R. Koschke, A. P. J. Breu, and K. Angstmann. Extending the reflexion method for consolidating software variants into product lines. In *Proc. 14th Working Conference on Reverse Engineering*, WCRE '07, pages 160–169, Washington, DC, USA, 2007. IEEE Computer Society.

[14] P. Tabeling. Home of Fundamental Modeling Concept. http://www.fmc-modeling.org/home, 2014.

[15] G. Böckle, P. Clements, J.D. McGregor, D. Muthig, and K. Schmid. Calculating roi for software product lines. *Software, IEEE*, 21(3):23 – 31, may-june 2004.

[16] I. John. Using documentation for product line scoping. *IEEE Software*, 27(3):42–47, 2010.

[17] G. Fairbanks. *Just Enough Software Architecture: A Risk-Driven Approach*. Marshall & Brainerd, 1st edition, 2010.

[18] S. Apel, D. Batory, C. Kästner, and G. Saake. *Feature-Oriented Software Product Lines: Concepts and Implementation*. Berlin/Heidelberg, 2013.

[19] Siegel, J. OMG's OCEB Certification Program, What is the Definition of Business Process? *An OCEB Certification Program White Paper*, May, 2008.

[20] Davenport, Thomas H. and Short, James E. The New Industrial Engineering: Information Technology and Business Process Redesign. *Sloan Management Review*, Summer:11–27, 1990.

[21] Verner, L. The Challenge of Process Discovery. *BPTrends*, May, 2004.

[22] Object Management Group. *Business Process Model and Notation (BPMN)*. 2011.

[23] M. Morisio, M. Ezran, and Colin Tully. Success and failure factors in software reuse. *IEEE Trans. Software Eng.*, 28(4):340–357, 2002.

[24] T. de Gooijer and H. Koziolek. Agreements for software reuse in corporations. In B. Meyer, L. Baresi, and M. Mezini, editors, *ESEC/SIGSOFT FSE*, pages 679–682. ACM, 2013.