Causes and Consequences of Application Portfolio Complexity – An Exploratory Study

Pouya Aleatrati Khosroshahi, Jannis Beese, Florian Matthes, Robert Winter

Causes and Consequences of Application Portfolio Complexity – An Exploratory Study

Pouya Aleatrati Khosroshahi[1], Jannis Beese[2]
Florian Matthes[1], and Robert Winter[2]

[1] Technical University of Munich, Chair for Informatics 19 (sebis)
Boltzmannstr. 3, 85748 Garching b. Munich, Germany
{p.aleatrati,matthes}@tum.de
[2] University of St. Gallen, Institute for Information Management
Unterer Graben 21, 9000 St. Gallen, Switzerland
{jannis.beese,robert.winter}@unisg.ch

Abstract. Application Portfolio (AP) complexity is an increasingly important and strongly discussed issue by both researchers and practitioners. Application portfolios in large organizations have become more and more difficult to understand, resulting in costly efforts to maintain and operate them. Although this is an urgent topic in large organizations, researchers and industry experts do not yet have a common understanding of this phenomenon and lack appropriate methods to measure and manage the respective complexity. We conduct an exploratory case study with the central enterprise architecture management (EAM) governance team and ten application owners of a large European automotive company to identify and link root causes and consequences of AP complexity. Furthermore, we evaluate possible solutions to decrease or manage this complexity from an application owners perspective. The results are interpreted from a socio-technical systems perspective.

Key words: Application Portfolio Complexity, Complexity Management, Socio-Technical Theory

1 Introduction

Technological advances, such as new possibilities for customer interactions enabled by digital platforms, require various industry sectors to fundamentally adapt their business models [1, 2]. Furthermore, increasing regulatory pressure also necessitates changes in the enterprise architecture (EA) domain due to a lack of transparency about enterprise information and poor data quality [3, 4]. Consequently, todays organizations need to undergo fundamental changes in their EA in general, and in their Application Portfolio (AP) in particular, and face multifarious obstacles in this transformation process [5, 6]: poor AP documentations leads to time-consuming and error-prone initiatives. As a result, enterprises are unable to efficiently adapt to changes since they are missing essential information about their AP.

Lacking a complete and consistent high-level view, organizations tend to introduce further services and applications to fulfill business needs, which leads to a perceived growth of complexity in the enterprise in the EA domain [7] and a growth of investments in the operation of information systems [8]. This manifests in a large number of heterogeneous information systems, which are costly to maintain and lack flexibility with regard to business changes [9].

Although the challenge of increasing AP complexity was already highlighted in research [10, 11] and by industry experts [12, 13], there is still a lack of research that explicitly addresses how to tackle this issue [14]. This is compounded by the fact that there exist multiple interpretations of the term AP complexity that depend on the specific context in which it is used [11, 15, 16, 17]. Based on our conducted literature review and state of the art research (see section 2), we define AP complexity as the compilation of organizational and technical characteristics in an enterprise that lead to avoidable costs and decreased agility of the AP. In order to identify root causes and possible solutions of this phenomenon, we conduct an explorative qualitative case study, as proposed by Yin [18], at a large European automotive company. Our analysis relies on data gathered from ten expert interviews, meetings with the central enterprise architecture management (EAM) team, and data from previously conducted complexity assessments. We employ socio-technical theory, in particular the *Punctuated Socio-Technical Information Systems Change* (PSIC) model [19], for organizing, grouping, and interpreting this data and corresponding results.

First, to gain a better understanding of the phenomenon at hand, we identify *root causes* of AP complexity as perceived by application owners in todays organizations. These root causes are then linked to specific *consequences* that negatively impact the organization. Finally, we evaluate technical and organizational *solutions* for managing AP complexity based on the identified root causes and their consequences. We address the following research questions (RQ):

- *RQ1: What root causes for AP complexity do application owners perceive in their daily activities?*
- *RQ2: What are the consequences of these root causes?*
- *RQ3: What kind of technical or organizational actions can help to control identified AP complexity?*

The rest of this paper is organized as follows: in section 2 related literature on AP complexity is reviewed and socio-technical systems theory is introduced as a lens for organizing and interpreting our findings. We then elaborate our methodology and data collection process in section 3. In section 4 we present our results, comprising root causes of AP complexity (*capacity*, *code quality*, *subjective complexity*, *technical support*, *design of data flows*, *quality of interfaces*, *IT authority of business*, *change management plan*, and *role allocation*), consequences of AP complexity (*lack of time / quality*, *data quality issues*, *performance issues*, *chain reaction to other functions*, *avoidable efforts*), and solutions to control AP complexity (*increased capacities*, *technical support*, *pool of experts*, *stronger IT governance*, *code reviews*, *automated checks*, *stronger data management*, *improved*

knowledge management, and *technical renewals*). The interpretation, applicability and consequences of these findings are then discussed in section 5. The paper concludes with a discussion of implications and limitations of this research.

2 State of the Art

There exist diverse and multi-faceted understandings of AP complexity in extant literature, which has been investigated from a number of different perspectives by previous researchers [10, 11, 14, 20, 21, 22, 23, 24, 25]. Thus, we review conceptualizations of AP complexity and how these are used in practice, noting that research is still at an early stage regarding the identification of complexity drivers of APs and the development of technical and organizational actions to control this phenomenon.

At the beginning of the 2000s the scope of complexity exploration in the information systems domain was enlarged from single applications to entire APs. The definition of the term AP complexity is, however, still fragmented: Following Schneider et al. [11], the view of AP complexity in the EA domain comprises different categories – such as subjective versus objective complexity or perceived versus objective complexity – each considering this phenomenon from a different perspective. Similarly, Beetz et al. [14] point out the variety of the term complexity, showing that various initiatives have taken place in this context and concluding with a research gap on this topic. Thus, when analyzing the increasing complexity of APs in today‘s organizations “a number of statements in the academic and consulting literature that include several implicit propositions on causes as well as on impacts“ [23] need to be considered, such as the age of applications or a decreasing agility of APs [26].

Notwithstanding the difficulties in conceptualizing and operationalizing AP complexity, several studies find dependencies between drivers of AP complexity, e.g., the age of applications, interdependencies, and redundancies, and related effects such as maintenance and operating costs [23]. An increasing number of components in an AP and an increase in their dependencies to each other negatively affect the flexibility with regard to architectural changes [20]. Proposed measures for AP complexity both in literature (e.g. [21, 22, 25]) and in practice [24] thus usually include the number of used components, their heterogeneity, and interdependencies between them, such as interfaces or information flows. Research in this area generally aims to identify and uncover hidden structures in APs to guide enterprise transformation [10]. For example, heterogeneity-based metrics can be employed to measure the complexity of employed applications within an portfolio, and the Design Science Matrix proposed by Lagerström et al.[21, 22] was found useful for assessing the criticality of IS change projects [24].

3 Research Methodology

Previous studies on AP complexity [3, 23] follow a quantitative approach to identify dependencies between business application characteristics (e.g., interfaces, type of application) and dependent variables (e.g., the amount of created incident tickets and operation costs of applications). While the conducted analysis allows to study statistical dependencies between the considered constructs, it turns out that AP complexity is also affected by organizational choices in a more complicated way: interdependencies and interactions may lead to emergent properties that are not easily captured by statistics [27]. Thus the extant quantitative results would benefit from a complementary qualitative investigation. To better understand the complicated ways in which AP complexity manifests and is affected by organizational choices, we employ an exploratory case study research, following the recommendations of Yin [18]. The conducted research approach is divided into five stages and is illustrated in Fig. 1.

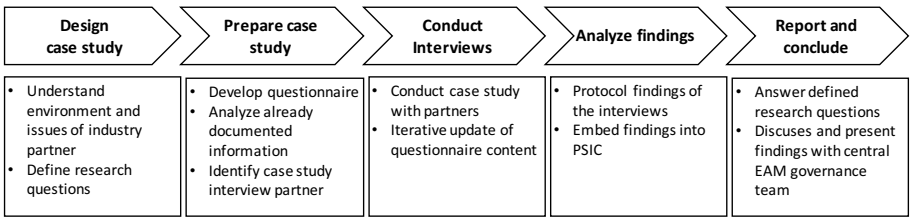

Fig. 1. Case Study Process

3.1 Case Study Approach

Our discussion of related literature (see section 2) shows that current research on AP complexity includes both qualitative and quantitative approaches. After reviewing the mentioned sources, we subsequently defined the research questions and decided on a research partner to conduct a case study in order to investigate the phenomenon of interest.

Case company description: The investigated organization is a large automotive company with over 100.000 employees. The headquarter of the company is located in Europe, whereas the plants are distributed in all continents and the dealers operate on an international level. Being one of the largest companies in its industry and currently investing significantly in AP complexity management initiatives, this company provides deep insights into the phenomenon of AP complexity. The first author has been involved with the ongoing efforts of the central EAM governance team since April 2015, allowing us to acquire rich data over a sustained period of time from internal complexity assessments, participation in

meetings, and access to relevant interview partners. The IT section of the automotive company is organized in twelve main departments and employs over 3.500 internal employees. The EAM governance team acts as an own department. All information about the deployed applications in the AP is documented in a central EA repository. Previous initiatives of the central EAM governance team on AP complexity revealed organizational and technical issues in the IT section, leading to decreased agility in projects and high operational costs to run the AP. It also turned out that the IT section is characterized by a silo mentality between the main departments, leading to missing transparency about deployed applications.

Design case study: We designed the content and structure of the case study in cooperation of the EAM governance team, aiming to identify technical and organizational actions (RQ3) to tackle AP complexity through group discussions with affected stakeholders. Consequently, we decided to interview two types of stakeholders: first, application owners, who are confronted with the consequences of AP complexity in their daily business and projects, and second, the EAM governance team, who has an aggregated view on this topic trough the complete organization. We discussed and finalized our proposed RQs with the EAM governance team, based on the findings of our literature review and experiences of the central EAM governance team.

Prepare case study: Based on our defined RQs and the findings of our literature review, we developed a questionnaire in cooperation with the EAM governance team. In developing the questionnaire we aimed to define the questions in a way that elicits concrete root causes of AP complexity as perceived by the application owners, and that allows to identify specific consequences and solutions for AP complexity, rather than strategic advice and general issues. The questionnaire is divided into six parts (general information, technical infrastructure and interfaces, problem / incident management, release management, software quality). The first part ensures the correctness of general information that was gathered before the interview (e.g., the name of the application, the application ID, and data about productive users). The following parts aim to identify current issues of the application on the respective topic. The application owners are asked to name root causes for each issue, its consequences and possible solutions to solve it.

In order to select a subset of applications from the company's AP for a detailed investigation, we started with all applications that were used productively by the company as a basis for further selection, excluding pure infrastructure components. From this set, comprising more than 7.000 applications, only those with significant costs for maintenance and errors were selected. Next, we employed data from internal complexity assessments, including information about application interfaces, monthly changes, incidents, and releases as well as sourcing and vendor information and information about the technical architecture. We include only applications for which this complexity index exceeds a predefined threshold, indicating that these applications are somehow more complex.

Finally, we limit our analysis to lead applications, i.e., applications, which the company considers to be fundamentally important for the operation of the enterprise. This set of 105 applications was then discussed with the EAM governance department, and 10 applications, deemed to be the most relevant, critical, and interesting, were selected as a final set for a detailed analysis together with the respective application owners.

Conduct interviews: We then conducted a series of ten semi-structured interviews with the application owners during November and December 2015. These interviews covered the areas identified in our research questions, namely, (RQ1) perceived root causes of AP complexity, (RQ2) consequences of these root causes, and (RQ3) technical or organizational actions employed to deal with AP complexity. All interviews were conducted face-to-face and followed a semi-structured approach in order to discuss a wide range of aspects [28].

Analyze findings: We then employed the PSIC model of Lyytinnen et al. [19] to group root causes of EA complexity and link these with consequences and applicable actions. The allocation of the findings to the PSIC model were conducted by our research team. To ensure the correctness of the findings, we presented and discuss our allocation with the EAM governance team (see step five *Report and conclude*).

Report and conclude: The results of the expert interviews and the allocation of the PSIC model were presented to and discussed with the company's EA governance department and the head of application portfolio management. It was considered a useful tool for dealing with problems arising as a consequence of AP complexity. Aside from the company-internal evaluation, the results were also presented by one of the authors and discussed at a two-day focus group on EA, involving senior enterprise architects and IT managers from five large European organizations in February 2016 [29].

3.2 Socio-Technical Systems Theory

This research relies on the PSIC model of Lyytinnen et al. [19] for organizing and interpreting results, since an analysis of AP complexity requires a comprehensive framework that also captures the dynamics and interactions between a multitude of different organizational elements [27, 30]. Socio-technical systems theory has been a useful perspective for ordering these diverse elements and interactions, thus allowing researchers to make sense of and reason about complex systems, such as enterprise architectures [31]. The PSIC model provides an established framework that also allows to reason about temporal causalities, such as the connection between root causes of AP complexity and related consequences.

Following this model, socio-technical systems comprise a social subsystem, consisting of actors and structure, and a technical subsystems, consisting of technology and tasks (Fig. 2). The overall behavior of the system is then determined by the interactions between all of these components. As a very general

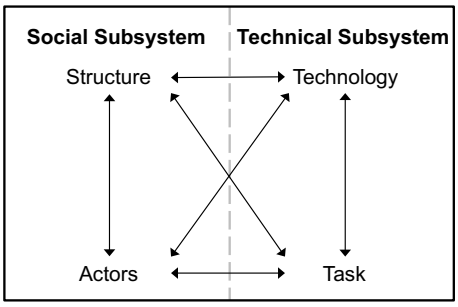

Fig. 2. Socio-technical systems theory [19]

example, an enterprise can be considered as humans (*actors*) using IT systems (*technology*) to perform work (*task*), which they have been assigned according to their role and position (*structure*). Transformation processes in large enterprises comprise a series of local changes within the organization, often in reaction to new and evolving external requirements [23, 32]. While these local adaptations manage to temporarily fulfill the requirements, a series of such changes across different parts of the organization generally introduces inconsistencies, unnecessary redundancies or dependencies, which are typical drivers of complexity [10, 33]. At some point, the misalignment between at least two socio-technical components will be noticeable and large enough to require the EA to undergo fundamental changes, termed *punctuated changes* [19]. For example, people will take action to change the system if the IT applications cannot handle new processes (technology-task misalignment) or if an application is too complicated for people to understand (actor-technology misalignment). Thus, root causes of AP complexity may be interpreted in the context of the related misalignment in the AP, i.e., *technology-people*, *technology-task*, *technology-structure*, *task-structure*, *task-people*, and *people-structure*.

4 Results

We use socio-technical systems theory as a lens to group and interpret our findings. First, identified root causes for AP complexity are discussed, which are considered as misalignments between any two socio-technical system components (see Fig. 2) and are grouped accordingly. Table 1 lists all identified root causes and also links them to related consequences of AP complexity. Finally, we present potential types of technical and organizational actions that are expected to deal with the root causes or to offset the consequences of AP complexity (see Table 2). The identified causes, consequences and actions are the result of aggregating similar elements found in the interviews through group discussions between the authors, also relying on feedback from meetings within the company and data from the complexity assessments. The results were discussed and validated with the EAM governance team to ensure the correctness of our findings.

4.1 Root Causes

In total, we identified 13 root causes of AP complexity (see Table 1). Most of the named root causes relate to technical issues within the EA landscape as a result of either misalignments between the technical system and the people that use and develop this system or misalignments between the technical system and the tasks that this system is supposed to carry out. The identified root causes are based on statements from multiple experts (application owner and employees of the central EAM governance team).

Technology/Actors: Issues with code complexity relate to the inability of people to use and maintain the technical system adequately since they are unable to make sense of too complex software code. Similarly, code quality relates to issues due to poorly written or documented code. Discussions with application owners revealed that one major problem is missing knowledge and documentation about single applications. This makes it difficult to steer the AP in an efficient way. The source code of old and highly customized applications often includes unused lines of code that cannot be deleted due to missing knowledge about the content and possible consequences. This results in complex and costly change activities in respective IT projects and additional maintenance activities. The application owners often mentioned unnecessary transitive interfaces in legacy systems. The quality of these interfaces is usually also lacking in terms of technical design and documentation. These circumstances decrease the transparency about the AP and lead to further workarounds to fix issues within data flows and thus to increased AP complexity. This is compounded by people lacking time and other resources to perform maintenance and development activities.

Technology/Task: Application owners state that applications are frequently missing adequate technical support such as dedicated testing instances for new deployments. One further issue is the quality of the implemented interfaces and the respective data flows. Often, these do not fit the required data formats, lack plausibility checks or include erroneous implementations. As a consequence, the transferred data might include useless information for the recipients and hinder the performance of planned tasks and thus the fulfillment of business requirements. Also, the quality of the source code might deliver wrong results and thus hinder the performance of the planned tasks.

Technology/Structure: Two application owners stated that the IT authority of business stakeholders leads to fundamental issues within the EA and increases the respective complexity. In their cases, business stakeholders have the permission to implement technical scripts within applications. As a consequence, enterprise architects face the challenge of redundant implementations that do not follow defined data dictionary standards and thus lead to missing transparency and a lack of knowledge about the state of the application. This also affects the knowledge about the AP and ends in inefficient decisions in daily

projects such as the introduction of redundant applications.

Task/Structure: Application owners named the setup of change management plans as another root cause for AP complexity. The change cycle of several applications in the organization is faster than the planned changes within one fiscal year. As a consequence, IT projects face the challenge of outdated data and have to perform several workarounds to fix these issues. Also the IT authority of business stakeholders was highlighted as a structural issue that hinders the fulfillment of planned tasks.

Task/Actors: The capacity issue, already explained for the misalignment technology/actors, also relates to this misalignment. Projects often lack resources, in particular people and time, what affects the fulfillment of tasks.

People/Structure: Large applications require collaboration between multiple stakeholders such as operations-, maintenance-, and defect-managers. Application owners identified the missing role allocation between these stakeholders as one root cause for AP complexity. Missing communication and the lack of a common language lead to undesirable conditions within the respective application such as missing maintenance activities.

4.2 Consequences

The identified root causes were linked to the following five consequences (see Table 1). The identified root causes are based on the conducted expert interviews (application owner and employees of the central EAM governance team).

Lack of time / quality: As a consequence of technical (C8) and organizational issues (C1, C6, C9), the implemented AP does not fit the defined requirements. Consequently, business stakeholders face the challenge of missing information and enterprise architects do not complete projects in time. This often leads to a number of manual and undocumented workarounds, which increase the amount of activities in the operation of the AP and thus lead to missing transparency.

Data quality issues: Application owners stated that business stakeholders have extensive permission rights in the investigated applications (C8), leading to the implementation of redundant scripts that lack a comprehensive picture of the application. Moreover, these scripts might include business-related errors, due to missing testing activities and ad-hoc implementations of scripts. There is the risk that the implemented scripts do not match the defined data quality standards within the application - such as the required granularity - ending in data quality issues. In several cases the design of ingoing or outgoing interfaces (C6) does not match the required data format or lacks necessary plausibility checks, which also leads to hidden data quality issues.

Socio-technical misalignment	Root cause	Consequences				
		Lack of time / quality	Data quality issues	Performance issues	Chain reaction to other functions	Avoidable efforts
Technology/Actors	(C1) Capacity	×				
	(C2) Code quality				×	×
	(C3) Subjective complexity					×
	(C4) Code complexity					×
Technology/Task	(C5) Technical support	×				
	(C6) Design of data flows		×			
	(C2) Code quality		×	×	×	
	(C7) Quality of interfaces					×
Technology/Structure	(C8) IT authority of business	×				
Task/Structure	(C9) Change management plan	×				
	(C8) IT authority of business		×			
Task/Actors	(C1) Capacity	×				
Actors/Structure	(C10) Role allocation					×

Table 1. Identified Root Causes of AP Complexity and Related Consequences

Performance issues: The quality of the source code of the applications often leads to major performance issues: calculations and report preparations exceed available time slots of batch jobs, which then result in automated cancelations of these jobs. As a consequence, the employees have to take further efforts in order to fulfill business requirements.

Avoidable efforts: The implementation of manual workarounds, data cleansing activities, and other efforts within IT projects could be avoided if the number of AP complexity root causes were decreased. Missing transparency leads to the implementation of redundant applications and thus to further efforts for maintaining and operating the complete AP. Moreover, the named technical (C2, C4, C3, C7) and organizational issues (C10) require manual efforts that also need a time-consuming coordination between different stakeholders and project teams.

Chain reaction to other functions: Technical and business related issues within applications often also affect other related functions within the organization. Application owners stated that the quality of the source code might hinder the fulfillment of business requirements due to cancelations of batch jobs. This in turn leads to missing information in other departments.

4.3 Solutions

The interviewed experts identified the following technical and organizational actions that are expected to reduce AP complexity or offset respective negative

consequences (see Table 2). All listed solutions were named by application owners in interviews and were validated with the EAM governance team.

Technical renewals: Experts suggested selected technical renewals of the source code. While it is not necessary to shut down complete applications within the AP, the renewal of single elements, e.g., lines of source code or outdated interfaces, may increase the quality of applications and reduce the extent of the consequences, leading to an improved steering of the AP. The identification of renewal candidates, however, requires time-consuming analysis activities of the source code and group discussions between application owners and enterprise architects in order to evaluate the added value of such renewals.

Solutions	**Consequences**				
	Lack of time / quality	Data quality issues	Performance issues	Chain reaction to other functions	Avoidable efforts
Increased capacities	×				
Technical support	×				
Pool of experts	×				
Stronger IT governance		×			×
Code reviews					×
Automated data checks		×			×
Stronger data management					×
Improved knowledge management					×
Technical renewals		×	×	×	×

Table 2. Named Solutions for Each Consequence

Improved knowledge management: Missing information about the AP directly leads to inefficient steering of it. It is crucial to define clear knowledge management initiatives in order to ensure a high transparency. As an example, we note that the automotive company uses an EA repository that acts as a single point of truth for technical-, business-, process-, and application-architecture information. The interviewed experts suggested to further increase the documentation of single applications and data flows between them, which is expected to increase transparency about already available technical solutions within the organization.

Stronger IT governance: The IT governance department needs to clearly define and implement rights and obligations of IT and business stakeholder. The business-side should not implement technical scripts. Upcoming projects should

verify AP changes within a blueprint process, e.g., by employing a business capability map, in order to ensure up-to-date information about the operating AP.

Pool of experts: The operating automotive company runs over 7.000 applications within their AP, including a large stack of used technologies and standards. Enterprise architects are required to make decisions in projects without having a deep understanding of the respective technologies. There is a risk that stakeholders make wrong decisions, e.g. by implementing functionally redundant technologies, leading to an inefficient AP. The interviewed experts recommend to establish a pool of experts for all used technologies within the AP blueprint, which can be consulted for respective decisions in projects.

Further solutions are an increased technical support and capacity when operating the application portfolio. These approaches simply provide additional resources for overcoming extant problems. In a similar manner, detailed code reviews were mentioned, which might help to identify hotspots within the AP. Moreover, a stronger implementation of automated checks at interfaces as well as an improved data management can prevent the origination of data quality problems within the AP.

5 Discussion

In section 1, we defined three RQs, aiming to evaluate root causes (Q1) and consequences (Q2) of AP complexity and to identify solutions (Q3) that decrease AP complexity or offset negative consequences. The identified root causes were embedded in the PSIC model in order to provide a structured and coherent overview of our findings. Each root cause leads to at least one consequence and each consequence is ameliorated by at least one proposed solution (see Table 1) and 2.

Considering Q1, we identified 13 root causes for AP complexity. The identified issues paint a comprehensive yet diverse picture, including technical (e.g., code quality), process-driven (e.g., change management plan), and organizational (e.g., capacity) findings, revealing that the phenomenon of AP complexity is a result of the interplay between different factors within an enterprise. A concluding discussion of our findings with the EAM governance team of our research partner confirms this: the technical reassessment of AP parts is not sufficient to decrease the respective complexity in long term. The controlled reduction of AP complexity also has to consider non-technical enterprise conditions, such as process management issues, e.g., change management plans, and requires sophisticated knowledge management processes.

The second research question revealed consequences of AP complexity in daily projects. We identified five consequences of AP complexity, including technical (data quality, performance) and business (lack of time / quality) related

consequences. The interviewed application owners emphasized the importance of AP complexity in IT projects and in their daily business, but also highlighted that critical business processes, i.e. processes that are related to the operation of plants in the automotive company, are not strongly affected by this issue. This may be due to a high amount of attention being allocated to these functions, resulting, for example, in a close monitoring of the application landscape, sharper governance principles and increased capacities for the operation of these business processes. This statement reflects the findings of our third research question: the suggested solutions, illustrated in Table 2, mainly include initiatives that aim to increase supporting capacities for the AP (e.g. pool of experts) and stronger monitoring operations (e.g., automated checks). However, to resolve the historically grown AP complexity application owners also suggest technical improvements in the currently operated applications (e.g. renewals, code reviews).

6 Conclusion

Our research aims to identify root causes, consequences, and possible solutions for AP complexity in large enterprises. We employed a case study approach, including ten expert interviews with application owners and group discussions with the central EAM governance team of a large European automotive company with an application portfolio of over 7.000 applications. Our results reveal the diverse issues related to AP complexity, including technical and organizational root causes, consequences, and solutions of this phenomenon. Our research extends current research on AP [10, 21, 22, 23, 9, 11, 25] by analyzing specific instances of real-world problems in connection with proposed solutions that might decrease AP complexity. The research results discover concrete characteristics of AP complexity in large organizations, which might be useful for further research in order to evaluate further solutions that might tackle this issue in practice. This is in line with calls to move research in this area away from abstract speculation towards an analysis of real-world issues [23].

The generalizability of these results requires further verification, in particular from organizations operating in different industries. A first step was made by discussing our findings in a focus-group with seven senior enterprise architects and IT-Managers from four other companies in the banking, logistics and insurance sectors. This discussion indicated that our results are applicable to other companies, as issues with AP complexity and attempted solution approaches are similar across different industries. Further research should specify the outlined solutions and define concrete procedures and methods. The expert interviews and group discussion reveal that the emergence of AP complexity is not observed for all functions of the organization: critical functions, in this case the operation of plants in the automotive sector, seem to be less affected by complexity in achieving their objectives. An evaluation of the technical and organizational factors that lead to the success of such functions seems to be promising.

References

1. Baharadwaj, El Sawy, O.A., Pavlou, P.A., Venkatraman, N. : Digital Business Strategy: Toward a Next Generation of Insights. In: MIS Quarterly, vol. 37 (2), pp. 471–483. (2013)
2. Malhotra, Y.: Knowledge management and business model innovation. Hershey: Idea Group Publishing, Hershey (2001)
3. Aleatrati Khosroshahi, P., Beese J., Aier, S.: What drives Application Portfolio Complexity? An Empirical Analysis of Application Portfolio Cost Drivers at a Global Automotive Company. In: 18th IEEE Conference on Business Informatics (CBI 2016), Paris (2016)
4. Aleatrati Khosroshahi, P., Roth, S., Hauder, M.: Impact of Solvency II on the Enterprise Architecture of Insurances: A Qualitative Study in Germany. In: Multikonferenz Wirtschaftsinformatik, Paderborn (2014)
5. Purchase, V., Parry, P., Valerdi, R., Nightingale, D., Mills, J.: Enterprise Transformation: Why Are We Interested, What Is It, and What Are the Challenges?. In: Journal of Enterprise Transformation, vol.1(1), pp.14-33. (2011)
6. Rouse, W. B., Marietta, L. B.: Enterprise transformation. In: Communications of ACM, vol. 49(7), pp. 66–72. (2006)
7. Ashkenas, R.: Simplicity-Minded Management. Harvard Business Review vol. 85, pp. 202–209. (2007)
8. Zammuto, R. F., Griffith, T. L., Majchrzak, A., Dougherty, D. J., Faraj, S.: Information Technology and the Changing Fabric of Organization. In: Organization Science, vol. 18(5), pp. 749–762. (2007)
9. Schmidt, C., Buxmann, P.: Outcomes and success factors of enterprise IT architecture management: empirical insight from the international financial services industry. In: European Journal of Information Systems, vol. 20(2), pp. 168–185. (2011)
10. Beese, J., Aier, S., Winter, R.: On the role of complexity for guiding enterprise transformations. In: 5th Enterprise Engineering Working Conference, pp. 113–127. Prague (2015)
11. Schneider, A. W., Zec, M., Matthes, F.: Adopting Notions of Complexity for Enterprise Architecture Management. In: Proceedings of the 20th American Conference on Information Systems, Savannah (2014)
12. Akella, J., Buckow, H., Rey, S.: IT architecture: Cutting costs and complexity, 2009. [Online]. Available: "`http://www.mckinsey.com/insights/business_technology/it_architecture_cutting_costs_and_complexity`".[Accessed: 10- Apr- 2016]
13. Reiner, J., Schaper, M.: Tackling IT complexity in product design, 2010, [Online]. Available: "`http://www.mckinsey.com/insights/business_technology/tackling_it_complexity_in_product_design`". [Accessed: 13- Apr- 2016]
14. Beetz, K. R., Kolbe, L. M.: IT Organization and Business-IT Power Sharing for the Ability to Manage IT Complexity: An Agenda for Action. In: The International Multi-Conference on Complexity, Informatics and Cybernetics, Orlando (2010)
15. Cookie-Davies, T., Cicimil, S., Crawford, L., Richardson, K.: Were Not In Kansas Anymore, Toto: Mapping the Strange Landscape of Complexity Theory, and Its Relationship to Project Management. In: Project Management Journal, vol. 38 (2), pp. 50–61. (2007)
16. Dewar, R., Hage, J.: Size, Technology, Complexity, and Structural Differentiaton: Toward a Theoretical Synthesis. In: Administrative Science Quarterly, vol. 23(1), pp.111–136. (1978)

17. Geraldi, J., Maylor, H., Williams, T.: Not lets make it really complex (complicated), A systematic review of the complexities of projects. In: International Journal of Operations "&" Production Management, vol. 31 (9), pp. 996–990. (2011)
18. Yin, R.K. Case study research: Design and methods. Sage publications, Thousand Oaks (2013)
19. Lyytinen, K., Newman, M.: Explaining information systems change: a punctuated socio-technical change model. In: European Journal of Information Systems, vol. 17(6), pp. 589–613. (2008)
20. Dreyfus, D., Wyner, G. M.: Digital Cement: Software Portfolio Architecture, Complexity, and Flexibility. In: Proceedings of the 17th Americas Conference on Information Systems, Detroit, submission 62, (2011)
21. Lagerström, R., Baldwin, C. Y., Maccormack, A. D., Aier, S.: Visualizing and Measuring Enterprise Application Architecture: An Exploratory Telecom Case. In: Harvard Business School Working Paper, no. 13-103. (2013)
22. Lagerström, R., Baldwin, C. Y., Maccormack, A. D., Dreyfus, D.: Visualizing and Measuring Enterprise Architecture: An Exploratory BioPharma Case. In: Harvard Business School Working Paper, no. 13-105. (2013)
23. Mocker, M.: What Is Complex About 273 Applications? Untangling Application Architecture Complexity in a Case of European Investment Banking. In: Hawaii International Conference on System Sciences, pp.1–14. Big Island (2009)
24. Schneider, W., Reschenhofer, T., Schtz, A., Matthes, F.: Empirical Results for Application Landscape Complexity. In: Hawaii International Conference on System Sciences, Kauai (2015)
25. Schuetz, A., Widjaja, T., Kaiser, J.: Complexity in Enterprise Architectures - Conceptualization and Introduction of a Measure from a System Theoretic Perspective. In: 21st European Conference on Information Systems, pp. 1–12. Utrecht (2013)
26. Ross, J. W., Weill, P., Robertson, D. C.: Enterprise Architecture As Strategy. Harvard Business Scholl Press, Boston (2006)
27. Kulik, B. W., Baker, T.: Putting the organization back into computational organization theory: a complex Perrowian model of organizational action. In: Computational and Mathematical Organization Theory, vol. 14(2), pp. 84–119. (2008)
28. Kvale, S. Doing interviews. Sage Publications, London (2008)
29. Tremblay, M. C., Hevner, A. R., Berndt, D. J.: Focus Groups to Artifact Refinement and Evaluation in Design Research. In: Communications of the Association for Information Systems, vol. 27(27), pp. 599–618. (2010)
30. Hanseth, O., Lyytinen, K.: Design theory for dynamic complexity in information infrastructures, the case of building internet. In: Journal of Information Technology, vol. 25(1), pp. 1–19. (2010)
31. Wu, P. P-Y., Fookes, C., Pitchforth, J., Mengersen, K.: A framework for model integration and holistic modelling of socio-technical systems. In: Decision Support Systems, vol 71, pp. 14–27. (2015)
32. Vessey, I., Ward, K.: The Dynamics of Sustainable IS Alignment: The Case for IS Adaptivity. In: Journal of the Association for Information Systems, vol. 14(6), pp. 283-311. (2013)
33. Dooley, K. J., Van de Ven, A. H.: Explaining Complex Organizational Dynamics. In: Organization Science, vol. 10(3), pp. 358–372. (1999)