Limiting Practices in Developing and Managing Software-Intensive Systems: A Comparative Study

Peter Wallin, Stefan Cedergren, Stig Larsson, Jakob Axelsson
Mälardalen University, School of Innovation, Design and Engineering, Sweden

Abstract--**Within the automotive industry, up to 90% of all new features are dependent on electronics and software. Consequently, the amount of software and electronics in vehicles are rapidly increasing. The same trend has been observed in other domains, such as telecom, avionics, trains, and more. An important factor in dealing with this inherent complexity is the use of a system architecture. The architecture is typically an enabler for both efficiency and effectiveness in the development of software-intensive systems but not directly connected to the customer needs. For example, the architecture can increase the agility of upcoming product releases, in order to cost effectively satisfy future customer needs.**

By combining two parallel multiple case studies, one focusing on the architects view, and the other one focusing on the managerial perspective, we have identified six limitations. Our results indicate that the focus is on customer requirements for the current product, on the expense of the internal requirements related to the development of the architecture and long-term profitability. Further, even if the early phases of development are identified as a success criterion, they are still not given enough attention.

I. INTRODUCTION

It has been 30 years since the first piece of software was used in a vehicle [3]. That particular software was used to control the ignition of the engine. The first software systems in vehicles were local and did not have any communication between different systems. Since then, a lot has happened and today almost all new functionality involves advanced control of electronics and software.

A similar evolution has happened for other type of industries, e.g. hardware control circuits for controlling electrical machines have been replaced by algorithms in software, and protection relay solutions have been moved from electromechanical solutions to be implemented in software-intensive systems.

Some parameters that make it hard to develop systems in both utility, process and vehicle industries are the long operational life time. At the same time, many of the functions controlled by electronics are safety critical and periodic maintenance cannot be assumed. Furthermore, the complexity is increased due to the different variants with many different configurations. The reason for this is partly due to varying customer demands but also due to the legal requirements of each country where the products are sold. To handle the different variants many industrial companies use a product line approach and many models share a common system architecture or platform.

The architecture affects the qualities of the system in all of these types of products [1]. Even though the architecture is an enabler for successful development of systems, the importance of the architecture is often neglected. This is mostly due to the fact that it is difficult to see any direct customer value provided by the architecture. However, if an organization is unsuccessful in the architectural work, adding functionality could be costly, the required quality might not be achieved, or it could even be impossible to include new functionality at all.

A. *System Architecture*

The term architecture and system are frequently used when developing software intensive systems. However, there is not always a common understanding of what an architecture is. During interviews with employees at different automotive manufacturers we asked for their view of what an architecture is. The placement of physical components and software was one respondent's idea of an architecture. Another said it is only the cabling, harnesses, and power consumption that are part of the architecture. One respondent claimed that an architecture is the guiding rules for how to build a system and also the composition of elements and their relationship. Through discussions with industry experts and our own observations, we have seen that the understanding of what constitutes an architecture is equally vague for software intensive systems in general.

The IEEE recommended term for architectural description of software-intensive systems is:

> *"The fundamental organization of a system embodied in its components, their relationships to each other, and to the environment, and the principles guiding its design and evolution"*[12]

This definition is quite general and most of the respondents own ideas on what an architecture is, can be mirrored in this definition.

B. *Performance Measurements in Product Development*

By nature and definition, product development has a long-term effect, and is often subjective in its value to the organization and is frequently intangible. Because of these features, traditional performance-based measures are, in the main, inappropriate [28]. Existing models of performance in product development are almost exclusively focused on the resulting artifacts instead of the performance of the activities required for its development [19]. The Stage-Gate model proposed by Cooper [6] establishes the main guidelines for analyzing the product development process from a performance evaluation perspective. This model presents the product development activities as a complex system that consists of two independent and parallel processes: the development process itself and the evaluation process [13]. The Stage-Gate model, consists of a series of stages, where the project team undertakes the work, obtains the needed information, and does the subsequent data integration and analysis, followed by gates, where go/terminate decisions are made i.e., to continue to invest in the project or not [5].

The performance measurement literature includes researchers from various academic fields such as accounting, operations management, marketing, finance, economics, psychology and sociology [16]. An important contribution to develop a common body of knowledge within this rich but scattered field of research was made by Neely, by editing the book Business Performance Measurements [16]. However, in this body of knowledge is an explicit focus on the product development activities missing. Performance measurements within product development is an elusive subject due to the multiplicity of meanings associated with performance measurement, the different roles and customers of performance measurement [29]. Performance measurements are important as an aid to determine priorities, e.g. within different activities, and as means of providing direction to teams by highlighting how they are performing and where improvements would be most beneficial. However, performance measurements must be kept in perspective; they must support the product development process and goal attainment [18] based on the business strategy.

The importance of performance measurements is evident. Sink and Tuttle [26] argue that the main focus of the performance measurement system is to provide managers with the needed information to be able to make decisions about what actions to take in order to improve the performance of the organization. Moreover, Lynch and Cross argue that the purpose of performance measurements is to motivate behavior leading to continuous improvement in customer satisfaction, flexibility, and productivity. Performance measurement can be defined as the process of quantifying action, where measurement means the process of quantification and the performance of the operation is assumed to derive from the actions by its management [27]. Without performance measurements in product development, fundamental managerial questions such as "how well are we doing", "what have we learned", and "what should we do in the future" cannot be answered [29]. *What gets measured gets done* [21] and *You are what you measure* [11] are two well known statements related to the use of performance measurements. Since performance measurements are so powerful it is important to align the measurement system with the strategic priorities of the organization [7, 17]. Performance measurements may function as the primary strategic deployment tool. The basic function of any performance measurement system lies in its integration into operative processes and in its actual use for taking action upon improvements leading to improved performance in the area targeted [10].

C. Motivation

The motivation of comparing the data of two different studies is that we have seen a big convergence in the results. The first study focuses on the technical perspective when developing the system and software architecture, and the second study, focuses on how management view performance measurement in development, of software-intensive systems. Fig. 1 shows the outcome of each study.

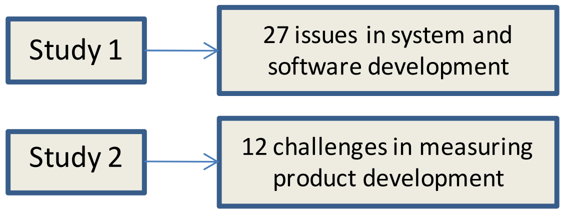
Study 1
27 issues in system and software development
Study 2
12 challenges in measuring product development

Fig. 1. Outcome from studies.

One of the benefits of comparing the result from these two studies is that we can make claims that are valid for a larger part of an organization.

D. Research question

The architecture design process can be viewed as a subset of the product development process as shown in Fig. 2. In the first case study, the focus was on identifying issues that relate to the architecture design process. However, one of the conclusions is that many of the identified issues, even if asked from a technical context, is heavily dependent on other areas. The second case study focused on finding what the challenges are with the current performance measurement system for the product development process. Through the combination of the studies we can identify practices performed in the organizations that limit the performance. The research question we seek to answer by combining the data from these two case studies is therefore as follows:

> *What limiting practices can be identified by combining issues related to the architecture design process with challenges in performance measurement?*

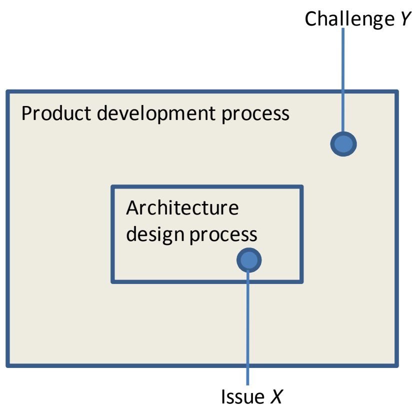

Fig. 2. Architecture design process as a subset of the product development process

The rest of this paper is structured as follows: In Section 2, the chosen research method to answer the research question is explained followed by Section 3, where the result with the six limiting practices is presented. In Section 4, the results are discussed and in Section 5, the paper is concluded with conclusions and future work.

II. RESEARCH METHOD

Similar research methods were used for both studies. Here we describe the generic method used in both studies and with some specifics for each study. Below we have used the acronym APS for the Architectural Perspective Study and PPS for the Performance Perspective Study. Fig. 3 shows a brief explanation on how the data from the different studies are combined to elicit the limiting practices.

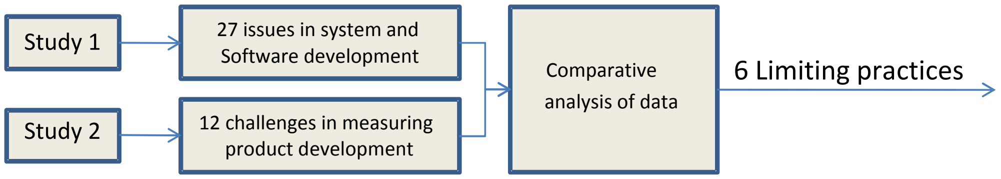
Study 1
Study 2
27 issues in system and Software development
12 challenges in measuring product development
Comparative analysis of data
6 Limiting practices

Fig. 3. Overview of research approach.

A. *Initial study decisions*

We choose the case study methodology since we wish to explore what people working in the organization think are the most challenging issues within the development of software-intensive systems. As our tool to collect data we used semi-structured interviews as it provides us with the flexibility to change direction based on the answers we get. Semi-structured interviews have predetermined questions, but the order can vary based on the interviewer's perception of what seems most appropriate [24] Additional questions can also be constructed during the interview and it is also possible to remove questions that seem inappropriate.

Another important advantage of using interviews instead of for example surveys, is the ability to explain a question further if the respondent is unsure about how to interpret the question. It is also possible for the interviewee to ask to respondent to further elaborate the answer when necessary. However, interviews are time consuming compared to a survey. More information about different methodologies can be found in [24, 32] and [31].

B. *Planning and Preparation*

In the Architectural Perspective Study (APS) the unit of analysis was the department of system and software development in three global automotive companies. The PPS study included five global organizations developing software intensive systems in the domain of telecom, transportation and process automation. Some organizations were included in both studies and to extend the overlap in organizations between the studies, a survey based on the issues from study 1 were constructed.

In total 81 interviews were conducted divided between the two studies, 27 for the APS and 54 for the PPS. Initially persons were selected by a contact working at the company with extensive knowledge about each organization and therefore suitable to identify people with different roles within the company. To avoid that the data solely relied on the recommendation of one person, a snowballing technique was used to locate more potential respondents.

For the APS, people interviewed included a project manager, a technical leader, a senior technical advisor, a system architect, a software architect, a senior manager, and a technical expert. The people interviewed in the PPS included senior managers, product owner, and persons responsible for the measuring system.

All companies in both studies have a matrix organization and roles from both the line and project organizations were included. This selection increases the probability that the data collection covers all major aspects of the architecture development. After the selection was made, invitations were sent out and interviews booked. None of the interviewees have any strong formal relationship to the authors or the different contacts at each company, which reduces the risk to get insincere answers.

C. *Interviews*

All interviews were semi-formal and questions were asked in a way that would encourage the respondent to talk about what they considered important. An example of a question for the APS is ***"How do you make architectural decisions today?"*** and for the PPS, questions were typically ***"How do you measure performance during development today?"***. Questions were added based on the answers from the respondents, and there were very large differences between different interviews regarding what topics were discussed and how much time was spent on each area. To further ensure that the respondent spoke as freely as possible, no recording devices were used. Two researchers were present during all interviews, one taking notes and the other one asking most of the questions. All interviews lasted between 50 and 120 minutes. All notes were transcribed directly after each interview to avoid any misinterpretation of the notes made.

The interviews were anonymous and no names were printed on the transcripts. All names of respondents were kept in a separate file to facilitate traceability in case the data needed to be complemented in any way.

D. Data Analysis

The data were extracted from the transcribed documents by categorizing data into a spreadsheet. The result from the data analysis was a long list of issues and factual statements. Similar issues were grouped together and a high level issue was constructed based on the low level issues. Each high level issue was constructed based on opinions from at least two respondents. A chain of evidence was upheld by a case study database or similar to uphold traceability as described by Yin [32]. All data analysis was done by at least two researchers jointly enabling a discussion about how to interpret the data.

For the APS, the data collection was complemented by a survey. This survey was conducted both with the companies participating in the interviews, and expanded with additional companies. It served several different purposes. The first was to validate that the issues extracted from the APS, were general issues and could be confirmed with the opinion of the different respondents in the study. The second motivation for the survey was to investigate to what extent these issues occurs at different companies, outside the ones participating in the initial APS. Apart from the three automotive manufacturers, three more organizations were included in the survey, all developing software-intensive systems.

E. Comparative analysis of data

To elicit limiting practices, all challenges from the PPS and the issues from the ASP were analyzed. Challenges and issues that touched upon the same problem but from different perspectives were grouped together. In some cases more than two issues were grouped with one challenge, and in a few cases one issues formed a group with several challenges. In such situations, the two issues or challenges were merged to one. As with the original data analysis, this step were done with at least two researchers and later reviewed by senior people with extensive knowledge in both academia and industry.

III. RESULTS

In the result section, we present the identified limiting practices. Each practice is presented with a short description followed by the architectural perspective and the performance perspective. For each practice, we present possible consequences concerning what might happen if these limitations are not managed. We conclude each practice with some ideas about what an organization needs to do in order to minimize these limiting practices. Table 1 summarizes the limiting practices.

TABLE 1. SUMMARY OF LIMITING PRACTICES

Nr	Limiting practice
1	Short term cost savings triumph long term success
2	Quality problems are not recognized until after product launch
3	Estimating business value is challenging and therefore often neglected
4	Project performance is more important than product portfolio performance
5	Technology is not part of the performance evaluation and therefore not prioritized
6	A cancelled project is seen as a failure and is not acceptable

A. Limiting practice 1 - Short term cost savings triumph long term success

The objectives of the architecture include supporting multiple products over a long time span. Usually architectural projects run over a number of years and it is hard to see the benefit of such within one fiscal year.

Architectural perspective: There is a lack of clear long-term architectural strategy

There is a lack of clear strategy for how the architecture should look in the future. A consequence of this is that new solutions sometimes are developed under stress with a result that does not appear satisfactory. A typical example from the automotive industry is attempts to cut cost on components leading to a bandwidth overload on networks causing a late restructuring of the network topology.

Performance perspective: The fiscal year budget process is stronger than the projects budget process

On a senior management level the fiscal year reporting has a higher priority since it reflects the organization to its' external stakeholders. As one of the mangers within accounting highlighted as an issue is the reporting of the organization that follows a calendar year and is design to also fit with reporting to the stock market. While the project's financial reporting is very much dependent on where it is in the project lifecycle.

Consequence

It is hard to express the value of architectural development and change in monetary terms, which makes it difficult to advocate for a long-term strategy. However, this increases the risk for major revisions of the architecture with a higher cost and reduced quality of the product.

Example of what can happen if this limiting practice is not properly addressed:

- **Decreased project performance.** Needed changes may not be possible to incorporate within a reasonable time and cost.
- **Increased architectural complexity and decreased product performance.** Shortcuts are introduced violating architectural integrity and creating suboptimal solutions.
- **New functionality not supported.** When long-term considerations are limited, new functionality might be harder or sometimes impossible to incorporate without major revisions of the architecture.

Possible solution

We see a need for a method to evaluate long-term architectural decisions enabling organization to balance short-term functional needs with strategically long-term targets as well as the effect of not taking any long-term decisions. In [9], the same problem is investigated and they believe that prospect theory [14] can be used to educate management to make them more conscious about the possible consequences when deciding between system improvements and adding new functionality.

B. Limiting practice 2- Quality problems are not recognized until after product launch

There is a long time between cause and measured effect of the development activities. This is especially true for product quality since it is measured mostly after the product has been launched. The focus on quality is not visible until after the product has been launched and not in each step of the development process as it should.

Architectural perspective: There is no method or model for measuring and follow up of quality problems during development

Lack of quality is not identified until the product reaches the market. Today, the actual quality achieved is not seen until late in the development process. For example, it is unclear how much a quality issue costs compared to choosing a more reliable and expensive solution from the beginning.

Performance perspective: Quality measurements are typically focused on the artifact, and not on the process

Quality is typically measured through MTBF, quality deficiency costs, error reports from the field etc. One internal study within one of the studied organizations concluded that all the faults identified in the testing and verification process could have been found in the previous step. The measurement system in use had a clear focus on reporting what had already happened i.e. had a lagging perspective.

Consequence

The long time between cause and measured effect makes it hard to estimate the cost of quality issues. It is particularly hard to learn from ones mistakes causing the same quality issues to appear in several products.

Example of what can happen if this limiting practice is not properly addressed:

- **Decreased product quality.** Many quality problems are not detected until the product has been released. When quality problems appear in the field they always get attention and the most experienced engineers will be occupied fixing quality problems of already released products instead of spending time in the early phases, of developing the new product when it is crucial with experienced people.
- **Increased maintenance cost.** Fixing quality issues in the field is always expensive. For some products it might be a remote software update can also be a recall of several thousands of cars for an automotive manufacturer.
- **Major rework is needed.** Quick fixes will most likely cause non-intended shortcuts, creating a bigger need for major revisions more frequently.

Solution

It is important that each step in the development result in the expected and planned quality. To get quick feedback on the product quality, the focus should be to understand that the process is followed and that it is efficient and effective in each step. A clear strategy is needed for the verification steps, from reviews of requirements and architectural solutions, through design and code review, to the integration and product testing. For all these verification steps, metrics should be defined that indicates that the activities are performed, and what the results from these are.

C. Limiting practice 3- Estimating business value is challenging and therefore often neglected

Both studies indicate that the value perspective is lacking, from an architectural standpoint as well as from a product perspective. Instead of focusing on the created value, focus is on reducing cost.

Architectural perspective: There is a lack of method or model to evaluate the business value when choosing the architecture

The connection between customer benefit and architectural decisions is hard to make, and the understanding of the relation between the architecture and the business is poor. A consequence is that many decisions are based on short-term cost requirements rather than long term strategic trends. Many respondents indicated that this may be due to the fact that each project must carry its own cost, but sometimes an investment in the architecture does not give any benefits until later in the lifetime of the platform. A better model for sharing this kind of investment between projects is needed. The consequence of such event-driven development is that a cheaper product cost can result in a complex system that is costly to maintain in the long run.

Experience is important when it comes to understanding the architecture. Today, architectural decisions are often made by experienced individuals based on gut feeling. There is a lack of a structured method for making these decisions. It is not clearly stated in the interviews that this results in poor architectures, but nevertheless some respondents ask for better arguments and statistics as a basis for making these decisions.

In [2], Bosch states that there is a need for such model for product line architectures but none existing yet. This issue is probably valid in many domains although it might not be possible to create a model that satisfies the need for many domains. One example is by Favaro et al. [8], where three approaches to value based software reuse is suggested.

Performance perspective: Measurements of value creation are missing

No measurements of value created or value to be created was identified. When asked about value creation a typical response was that it is difficult to demonstrate the value of a new product that is the result of incremental development and primarily a new version of a product already in the market. All of the case companies do have a structured process to collocate a clear business case in order to initiate a development project. This information is used in order to gather internal funding for the project, but not to understand how value is created throughout the projects.

Consequence

Since the value perspective is missing it becomes hard to value both architectural decisions as well as the overall architectural process. A typical dilemma is that a customer function will most likely be prioritized over an architectural change since the revenue of adding that customer function can easily be calculated, while the revenue of increasing the flexibility in the architecture to support multiple products is difficult to calculate.

Example of what can happen if this limiting practice is not properly addressed:

- **Inappropriate solutions may be selected.** Solutions are selected by mostly using cost as the deciding factor instead of using the value gained.
- **Selected solutions may be more expensive from a life cycle perspective.** Since short term development and product cost are the deciding factors, even for an architecture that supports multiple products with a life cycle of several years.

Solution

Develop a method or model that can visualize the value of the architecture. A special focus should be to illustrate the portfolio value of all products building on the same architecture. Approaches such as the Architecture Tradeoff Analysis Methods, ATAM [23], can provide some guidance to include business objectives in the process of designing the architecture. However, the ATAM cannot be used to evaluate the benefit for the portfolio of products building on the same architecture. Lindgren et al. [15] suggest a simple four step method to balance system qualities with adding new functionality.

D. Limiting practice 4 – Project performance is more important than product portfolio performance

The current measurement systems focus more on measuring the project manager, i.e. how well the individual project performs in terms of time and cost, not the performance of the product portfolio. This is further enforced by that each project has to carry its own cost.

Architectural perspective: Decisions are easily made that suit one's own project, team or component even though it leads to a poorer overall solution

Sub-optimizations are common and sometimes lead to a more complex overall solution than necessary. Optimization is made within one's own project or team and does not consider the potential of a favorable overall optimization. Each project is supposed to carry its own cost and this means that no one is prepared to compromise in favor of commonality. Everyone thinks that commonality is good as long as "my project" doesn't have to adapt in any way. This relates to Conway's law [4] from 1968 that says: "Any organization which designs a system will inevitably produce a design whose structure is a copy of the organization's communication structure".

Performance perspective: The focus of the performance measurement system is to report project progress

The primary focus of the performance measurement system is to report project progress to upper management. As one manger expressed it “the performance measurements focus on the project manager and not what enables high performance”.

Consequence

Projects are optimized without considering other projects that might build on the same architecture.

Example of what can happen if this limiting practice is not properly addressed:

- **Decreased overall profitability.** The project portfolio is sub-optimized because focus in on optimizing the performance of each project.

Solution

There is a need to focus more on optimizing the portfolio of all development projects as a whole and not purely on individual project success. This is difficult with the measurement system used today. We have not been able to identify a promising solution to this particular limitation.

E. Limiting practice 5 – Technology is not part of the performance evaluation and therefore not prioritized

The qualitative results from both studies indicate that it is important to incorporate the technology aspect in decisions. However, if the technology is not part of the measurement system it is hard to get the attention it needs.

Architectural perspective: Technical parameters are regarded as less important than cost when selecting components or suppliers

The price strongly drives the choice of component. The purchasing department choose the supplier and sometimes technical parameters are traded for a lower price. This can sometimes lead to lower quality and hardware problems for modules mounted in a harsh environment. "You get what you pay for", as one respondent stated. On the other hand, the price is a very tangible parameter, whereas quality issues are often speculative at the time when the supplier choice is made.

Performance perspective: Technology is not used in the performance evaluation

The technology used in the products is not part of the performance evaluation, even though it affects the performance. Only one of the five case companies had some indicators related to their technology and how it affects the performance. However, the technology aspect such as the product architecture was stressed as one important factor for both the effectiveness and the efficiency in the development of software-intensive products.

Consequence

When the technology is not used in the performance evaluation it is directly reflected in the product. A solution that is cheaper, but might be less flexible in terms of adding new functionality at a later stage is usually chosen.

Example of what can happen if this limiting practice is not properly addressed:

- **Decreased product performance.** When the technology aspect is neglected it will reduce both effectiveness and efficiency in the development of software intensive systems.

Solution

A strategic decision is needed to divide the investments in product development into functional and architectural development. Another study made by Ozkaya et al. [20], architects indicates that the only way to get acceptance for an architectural upgrade is to include them when adding new functionality. The approach of including (i.e. hiding) the architectural upgrade can be problematic since the need for the architectural upgrade will never be highlighted.

F. Limiting practice 6 –A canceled project is seen as a failure and is not acceptable

If no projects are terminated it is hard to get resources for all of them. Especially the ones that probably should have been terminated will consume a lot of resources, making potential successful projects unsuccessful.

Architectural perspective: Pre-development projects have low priority and to increase the priority they are merged into development projects too early

Too little effort is put into advanced engineering projects or early concept and technology development. The projects are included too early in a delivery project to increase the attention and priority of the project. This is due to the fact that many resources are spent in the end of the delivery project making the advanced engineering projects short on resources. This severely increases the uncertainty in the delivery project. A more structured way of dealing with advanced engineering projects and stricter demands about when an advanced engineering project should be allowed in a vehicle project is needed and also it would be beneficial to try to move from back load to front load development. A problem is that legal requirements might force an advanced engineering project to be included earlier than what is preferable. One reason that the organization usually ends up in this situation might be that old development projects cannot keep their deadlines and are therefore utilizing resources that were allocated for advanced engineering projects. This issues with a possible solution is discussed in [30].

Performance perspective: A project started is a project completed

Almost no development projects are terminated once they have been initiated. Even though all of the organizations have some kind of stage-gate process in use, the termination of projects were never or almost never the case. When it occurred, the project was more paused and conducted later, due to budget cuts or similar. Hence the business side of the projects is not reevaluated along the progress of the project as both the technical and market uncertainty decreases.

Consequence

Initiated projects are directly included in the proposed delivery where it is costly to remove a project.

Example of what can happen if this limiting practice is not properly addressed:

- **Decreased overall projects.** Since resources are limited and if no project is canceled it is likely that the overall quality will decrease.
- **Cannot keep project deadlines.** Risky projects will consume a lot of the available resources causing many projects to deliver late.

Solution

Allocate more resources to pre-development project as well as allow projects to be terminated. A possible solution is to use Coopers Stage-Gate model [6] that explicitly has a go or terminate decision at a certain gate. However some of the companies use a stage gate model but don't seem to apply it as intended.

IV. DISCUSSION

We have by comparing the data from two case studies seen that there are several limiting practices when it comes to the development of software-intensive systems. A reoccurring theme is that the early phases are not as prioritized as they should be. The reason for this could be that the company's existing measurement systems focus on the later phases of development. However, in the later phases of development the solution space is limited.

In several cases, we have seen that one reason for not prioritizing the early phases is that most senior people are involved in fire fighting activities, described by Repenning in [22]. This means that key persons are involved in projects that already left development, but due to quality issues, they still become highest priority of the development organization.

Another observation is that there seem to be consistency between the perspective of the architects and the managerial perspective, that the architecture is an important part and that it should be prioritized. Still, when resources are limited it is still among the first one to go. A possible explanation to this is that if you remove all architectural work, the products will still come out, with possible consequences such as; lack of quality, increased product portfolio cost, decreased possibilities to add new functionality, and increased development time for each product. However, if you remove all software developers, no software will be produced, hence these software-intensive products will not work at all, despite how much resources you put into designing an architecture. The architecture is more an enabler but usually not a necessity for adding new functionality and does therefore not provide direct customer value. The same comparison can be made with performance measurement in product development. Even if you decide to not measure the performance, products will still be delivered. However, without knowing how we can improve in future releases.

The objective of the architecture, when developing software intensive system, is to support several products over a long time span. When several products utilize the same architecture it is extremely hard to value the benefit of the architecture for each product. However, in a product portfolio perspective it should be able to see the benefit of the architecture. Thus, typical for the architecture is that both the dynamic and behavioral complexity discussed by Senge and Roth in [25]. A large dynamic complexity indicates that there is a long time between cause and effect i.e. an architectural decisions to chose a more powerful processor to meet the demands on new functionality in 5 years and also possibly support other products. Our data indicates that the decision in this case will most likely be to choose a processor that only meets the current needs. We have seen two reasons for this: The first one is that other projects are neglected since only the result of each project is measured and little or no consideration about how the project created a value for other projects. The second one is that the fiscal year budget is more important than long-term profitability. A large behavioral complexity indicates that there is a high diversity in aspirations, mental models, and values among decision makers [25]. In the process of architecting this behavioral complexity is characterized by the different requirements from different stakeholders such as, product owner, marketing department, customer, developing organization and maintenance organization.

Architects say they need more resources but are usually cut short, but according to this result, the managerial perspective and the architectural perspective coincide. However, existing performance measurement systems is focusing on parameters that neither the architects nor management sees as important. This could partly be solved by adapting the existing performance measurement system to focus on the value perspective for the product portfolio and not on time and cost for specific projects.

V. CONCLUSIONS AND FUTURE RESEARCH

In this paper, we present six limiting practices that we have identified in organizations developing software-intensive systems. The comparative analysis indicates that there is a big need to focus more resources on the early phases of development. One of the reasons that this is not done today is that existing performance measurement systems focus on the later phases of development. The result further indicates that there is a strong need for a way to value the architecture, especially in the context of a portfolio of products building on the same architecture. As a complement to existing measurement techniques, we see a great need for an architecture performance management system. The direction for future contribution in the area is to identify and develop an architecture performance measurement system, and to further develop, pilot, and make available solutions for the presented limiting practices.

REFERENCES

[1] Bass, L., P. C. Clements, and R. Kazman, *Software architecture in practice (2nd edition)*. Pittsburgh, PA: Addison-Wesley Professional, 2003.

[2] Bosch, J., "Product-line architectures in industry: A case study," in *Proceedings of the 21st international conference on Software engineering* Los Angeles, California, United States: ACM, 1999.

[3] Broy, M., I. H. Kruger, A. Pretschner, and C. Salzmann, "Engineering automotive software," *Proceedings of the IEEE,* vol. 95, pp. 356-373, 2007.

[4] Conway, M. E., "How do committees invent?," *Datamation,* vol. 14, pp. 28-31, 1968.

[5] Cooper, R. G., "Perspective: The stage-gate® idea-to-launch process - update, what's new, and nexgen systems," *The Journal of Product Innovation Management,* vol. 25, pp. 213-213, 2008.

[6] Cooper, R. G., "Stage-gate systems: A new tool for managing new products," *Business Horizons,* vol. 33, pp. 44-55, 1990.

[7] Dixon, R. J., A. J. Nanni, and T. E. Vollmann, *New performance challenge: Measuring operations for world-class competition*. New York: McGraw-Hill Professional Publishing 1990.

[8] Favaro, J., K. Favaro, and P. Favaro, "Value based software reuse investment," *Annals of Software Engineering,* vol. 5, pp. 5-52, 1998.

[9] Fogelström, N., S. Barney, A. Aurum, and A. Hederstierna, "When product managers gamble with requirements: Attitudes to value and risk," in *Requirements engineering: Foundation for software quality*, 2009, pp. 1-15.

[10] Godener, A. and K. E. Soderquist, "Use and impact of performance measurement results in r&d and npd: An exploratory study," *R & D Management,* vol. 34, pp. 191-219, 2004.

[11] Hauser, J. and G. Katz, "Metrics: You are what you measure!," *European Management Journal,* vol. 16, pp. 517-528, 1998.

[12] IEEE, "Ieee recommended practice for architectural description of software-intensive systems." vol. IEEE 1471-2000: IEEE, 2000.

[13] Jiménez-Zarco, A. I., M. P. Martínez-Ruiz, and Ó. González-Benito, "Performance measurement system (pms) integration into new product innovation: A literature review and conceptual framework," *Academy of Marketing Science Review,* vol. 10, 2006.

[14] Kahneman, D. and A. Tversky, "Prospect theory: An analysis of decision under risk," *Econometrica,* vol. 47, pp. 263-291, 1979.

[15] Lindgren, M., A. Wall, R. Land, and C. Norstrom, "A method for balancing short- and long-term investments: Quality vs. Features," in *Software Engineering and Advanced Applications, 2008. SEAA '08. 34th Euromicro Conference*, 2008, pp. 175-182.

[16] Neely, A., *Business performance measurement*, 2 ed. Cambridge: Cambridge University Press, 2007.

[17] Neely, A., M. Gregory, and K. Platts, "Performance measurement system design: A literature review and research agenda," *International Journal of Operations & Production Management,* vol. 25, pp. 1228-1263, 2005.

[18] Nixon, B., "Conference report: Performance measurements for r&d," *R & D Management,* vol. 27, pp. 87-90, 1997.

[19] O'Donnell, F. J. and A. H. B. Duffy, "Modelling design development performance," *International Journal of Operations & Production Management,* vol. 22, pp. 1198-1221, 2002.

[20] Ozkaya, I., P. Wallin, and J. Axelsson, "Architecture knowledge management during system evolution: Observation from practisioners," in *Fifth Workshop on SHAring and Reusing architectural Knowledge - SHARK* Cape Town, South Africa, 2010.

[21] Peters, T., "Tom peters revisited: What gets measured gets done," *Office Solutions,* vol. 19, pp. 32-33, 2002.

[22] Repenning, N. P., "Understanding fire fighting in new product development," *The Journal of Product Innovation Management,* vol. 18, pp. 285-300, 2001.

[23] Rick Kazman, Mark Klein, and Paul Clements, "Atam: Method for architecture evaluation," 2000.

[24] Robson, C., *Real world research*, Second ed.: Blackwell publishing, 2002.

[25] Roth, G. L. and P. M. Senge, "From theory to practice research territory, processes and structure at an organizational learning centre," *Journal of Organizational Change Management,* vol. 9, pp. 92-&, 1996.

[26] Sink, D. S. and T. C. Tuttle, *Planning and measurement in your organization of the future*. Norcross, GA: Industrial Engineering and Management Press, 1989.

[27] Slack, N., S. Chambers, and R. Johnston, *Operations management*, 5 ed.: Pearson Education Limited, 2007.

[28] Stainer, A. and B. Nixon, "Productivity and performance measurement in r&d," *International Journal of Technology Management,* vol. 13, p. 486, 1997.

[29] Tatikonda, M. V., "Product development performance measurement," in *Handbook of new product development management*, C. H. Loch and S. Kavadias, Eds. Oxford: Butterworth-Heinemann, 2008.

[30] Thomke, S. and T. Fujimoto, "The effect of 'front-loading' problem-solving on product development performance," *Journal of Product Innovation Management,* vol. 17, pp. 128-142, 2000.

[31] Wohlin, C., P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in software engineering: An introduction* vol. 6: Springer, 2000.

[32] Yin, R. K., *Case study research: Design and methods*, 3 rd ed. Newbury Park: Sage Publications, 2002.