Association for Information Systems
AIS Electronic Library (AISeL)

BLED 2018 Proceedings | BLED Proceedings

2018

Adaptation of enterprise architecture efforts to an agile environment

Robin Duijs
HU University of Applied Sciences, rduijs@sligro.nl

Pascal Ravesteyn
HU University of Applied Science, pascal.ravesteijn@hu.nl

Steenbergen, vav Marlies
University of Applied Sciences Utrecht, marlies.vansteenbergen@hu.nl

31ST BLED ECONFERENCE: DIGITAL TRANSFORMATION: MEETING THE CHALLENGES
JUNE 17 - 20, 2018, BLED, SLOVENIA, CONFERENCE PROCEEDINGS
A. Pucihar, M. Kljajić Borštnar. P. Ravesteijn, J. Seitz & R. Bons

Adaptation of enterprise architecture efforts to an agile environment

ROBIN DUIJS, PASCAL RAVESTEIJN & MARLIES VAN STEENBERGEN

Abstract Agile ways of working have become mainstream, with many organisations practising a form of agile. Agile maturity among those organisations differs. In a research conducted by VersionOne Inc. (2016), 82% of the participating organisations stated to be at or below the level of 'still maturing'. Existing agile and architecture methods have begun to incorporate some aspects of each other, with agile methods including architecting, such as the Scaled Agile Framework (SAFe), and architecture frameworks such as TOGAF (the Open Group Architecture Framework), adding agile elements (Poort, 2014). This study addresses the question how to shape the architecture function to effectively achieve compliance with architecture regulations, of solutions realised in an agile environment. To answer this question a multiple-case study was done, studying three different organisations. The findings are translated into seven propositions.

Keywords: • Agile • Enterprise Architecture • Agile maturity • SCRUM •

CORRESPONDENCE ADDRESS: Robin Duijs, MSc., HU University of Applied Sciences, Padualaan 99, Utrecht, The Netherlands, e-mail: rduijs@sligro.nl Pascal Ravesteijn, Ph.D., Professor Process Innovation and Information Systems, HU University of Applied Sciences, Padualaan 99, Utrecht, The Netherlands, e-mail: pascal.ravesteijn@hu.nl. Marlies van Steenbergen, Ph.D., Professor Digital Smart Services, HU University of Applied Sciences, Padualaan 99, Utrecht, The Netherlands, e-mail: marlies.vansteenbergen@hu.nl

DOI https://doi.org/10.18690/978-961-286-170-4.26 ISBN 978-961-286-170-4

Available at: http://press.um.si.

1 Introduction

Agile methods are based on the agile manifesto (Beck et. al., 2001). One of the twelve agile principles states that good architectures, requirements, and designs tend to be constructed by self-organizing teams. According to the report 'the state of agile 2015' by VersionOne, Inc. (2015), nearly 70% of respondents use agile methods such as Scrum or Scrum/XP hybrid. According to Wilson and Altman (2013) the role of the architect does not exist in Scrum because of bad experiences with 'ivory tower' architects and resistance to 'big design up front'.

Petersen and Wohlin (2009) also mention that agile approaches are lacking in architectural design. More recently Yang et al. (2016) confirmed that the role of architecture in agile development is often overlooked. This leads to agile teams struggling due to the lack of good architecture and architects distrusting agile projects (Wilson & Altman, 2013).

According to The Open Group Architecture Framework (TOGAF, 2009) ensuring the compliance of individual projects with the enterprise architecture is an essential aspect of architecture governance. TOGAF offers compliance review process descriptions, checklists and guidelines. There have also been several studies on architecture compliance checking approaches. Knodel & Popescu (2006) define architecture compliance as a measure that compares the implemented architecture in the source code to the planned architecture and determines to which degree these differ. However, architecture frameworks such as TOGAF are not tuned to agile teams. TOGAF (2009) describes the use of its framework in an iterative process, but ever since the latest release of TOGAF 9 on the 4th of February 2009 a lot has changed in the world of IT and agile.

Agile methods implement a different way of working. An important difference between traditional and agile development, when it comes to architecture, is transparency. Transparency on everything done and decided as well as the reasoning behind everything. This is related to the greater autonomy of agile teams in comparison to traditional development teams. Whereas traditional teams might accept the task they are given, agile teams would like to be involved and as such demand to know why certain decisions are made.

Scrum, one of the leading agile methods, states "Scrum is founded on empirical process control theory, or empiricism. […] Three pillars uphold every implementation of empirical process control: transparency, inspection, and adaptation." (Schwaber & Sutherland 2017, p.4). One of the Scrum values is on 'Openness', stating "the Scrum Team and its stakeholders agree to be open about all the work and the challenges with performing the work" (Schwaber & Sutherland 2017, p.5). Because of this, transparency is a key property architecture should possess.

Apart from transparency a major difference is the influence a Product Owner has on the product. Architects tend to feel less powerful in an agile environment because PO's

overrule architecture for the sake of business value. This leads to architects stressing the importance of architecture and focussing on cost and risk aspects.

According to Poort (2014), the debate seems to be starting to settle down: today some agile methods include architecting, such as the Scaled Agile Framework (SAFe), and some architecting frameworks, such as TOGAF, are adding agile elements. Still, organisations are struggling how to adapt their architecture function to an organisation that is becoming more agile. There still is little support when it comes to architecture guidelines for agile development that match the organisation's maturity. Based on this the following research question is formulated:

In what way can enterprise architecture efforts be shaped (i.e. form and function) to effectively achieve compliance with architecture regulations of solutions realised in an agile environment?

To answer this question a multiple-case study is conducted in three organisations that are currently applying agile development.

The remainder of this paper is organized as follows. In section 2 the research method is described, followed by the findings in section 3. A discussion on the findings is provided in section 4. Finally, in section 5, conclusions are drawn and implications, limitations and suggestions for further research are described.

2 Research method

The goal of this study is to generate new theory on how to effectively apply architecture in an agile environment. According to Zikmund et. al. (2009), a theory consists of a coherent set of general propositions that offer an explanation of some phenomenon, by describing the way other things correspond to this phenomenon. A proposition explains the logical linkage among certain concepts by asserting a universal connection between concepts (Zikmund et. al., 2009).

In this study, an explorative multiple-case study paired with semi-structured interviews is used. Case study research is used, because the topic of this study requires inquiry into the way how certain processes are structured and why this is so. Multiple cases are necessary because of the interest in similar and contrasting results. Yin (2014) calls this a "replication logic". Eisenhardts (1989) process of theory building from case study research is used to structure the research process. According to Eisenhardt the theory building process should be free from prejudice as much as possible. To achieve this the researcher needs to retain theoretical flexibility. This flexibility limits the possibility of bias which could influence the findings. At the outset of the process, the study begins as close as possible to no theory under consideration and no hypothesis to test. However, as specifying constructs helps shape the design of the study, a preliminary conceptual model is created, based on the commonly distinguished aspects of any function, process, product and people.

Three cases were selected from different industries and investigated using semi-structured interviews and subsequently template-coding the transcriptions of the interviews. Within-case analysis was followed by cross-case analysis, leading to the formulation of propositions. The propositions were further validated by comparing them with extant literature. Because this theory building process is intimately tied to evidence, it is very likely that the resultant theory will be consistent with empirical observation (Eisenhardt, 1989). Overall, tying the emergent theory to existing literature enhances the internal validity, generalisability and theoretical level of theory building from case study research (Eisenhardt, 1989). The external validity is supported by choosing organisations from different industries. As can be seen in table 1, the first case is a university, the second a power company, and the third an IT service provider.

Table 1: Case study characteristics

Label	Company	Architecture – Agile project type
C1	University	Has some agile projects but also uses traditional development methods; an architect advises the project team on architectural concerns in both types of projects.
C2	Power company	All projects are agile, using some of the practices as described by SAFe. External architect manages multiple teams and coordinates architecture concerns within a certain predefined domain. The domain architect also keeps contact with tech leads from various teams and the enterprise architects of the company.
C3	IT Service Provider	Provides services to government. Agile projects have a project architect which is responsible for the project architecture. The project architect is, however, no part of the architecture function of the organisation. The architects in the architecture function must review and reach an accord on the architecture vision of the project.

When examining the cases, multiple variations are seen. All cases have overarching EA teams influencing agile teams in an indirect manner. All cases show architecture involvement in agile teams. C2's agile development team actually contained a full-time delegate architect (under direct supervision of a domain architect). The other cases show either part-time architecture involvement or direct architecture influence through a project architect. Despite the variations the traditional architecture work is still done by people in (delegated) architecture roles.

Each case is analysed individually (within-case analysis) and subsequently compared to the other cases (cross-case analysis). Next, based on all three cases, propositions are created. For guidance during the case study interviews, an interview guide is constructed. This guide is based on the preliminary conceptual model depicted in Figure 1.

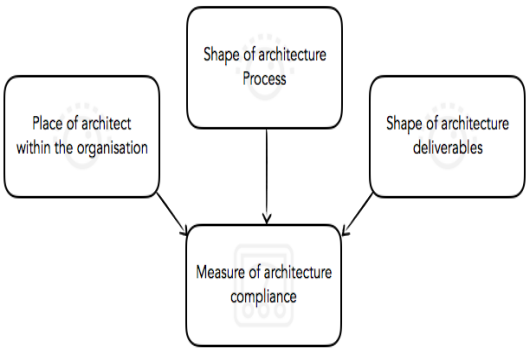

Figure 11: Preliminary Research Constructs

The goal of the interviews is to gain insight into the preliminary constructs and how they influence one another. In each organisation an agile team member as well as an architect is interviewed. Thus, both perspectives are included. All interviews are recorded, and data collected from the interviews (which are transcribed) are coded with a priori codes for within-case data analysis. The a priori codes are based on existing knowledge and theory regarding architecture frameworks and principals. During coding or analysis, additional codes emerged and were added to the existing codes. The codes used are involvement of architect, architecture design, architecture principles, architecture deviations, architecture decisions, architecture role, and standards. The last two codes emerged from the interviews. The codes were used to sort the statements from all interviewees and compare them both within the cases and across the cases.

Based on the findings from the within-case analysis and cross-case analysis certain constructs appeared. These refined the initial research model constructs as shown in figure 1. In the process of generating propositions based on these constructs, additional literature was used to establish the propositions as feasible findings or disprove them.

3 Results

The case study reveals that the preliminary research constructs need refinement (see Figure 2).

Acceptance of architecture
First, it appears that compliance with architecture regulations is not the main objective in an agile environment, but that acceptance of architecture by agile teams is more

important. In an agile environment the focus is on creating the architecture in collaboration with the agile team under supervision/direction of the architect while making decisions, taking into account the current situation and the existing constraints at this moment in time. Therefore, the dependent variable in the preliminary conceptual model is refined to '*acceptance of architecture*', which is defined as the measure to which architecture is a part of the way of working, i.e. agile teams are receptive to architecture.

The independent variables, too, are refined by the case study, as described below.

Transparency and documentation of reasoning
A main topic which came forth from the cases is transparency on reasoning. The cases show that architecture principles should be applied in a transparent manner. The reasoning behind architecture decisions must be described clearly and transparently and communicated to all stakeholders. This is supported by literature. In their 'Understanding/Acceptance Principle', Slovic and Tversky (1974) describe that understanding results in acceptance. The more someone understands and comprehends something the more that person will accept it. By documenting reasoning of architecture decisions, the understanding can be increased, leading to greater acceptance. And being transparent on the procedure of how a decision was made also leads to greater acceptance (Skitka & Mullen, 2002). This leads to the independent variable 'transparency and documentation of reasoning', which is defined as a clear understanding for all stakeholders on how decisions were made and making the reasoning behind decisions available for reference.

Direct and active involvement of architects
All cases show a need from the team perspective that an architect should be involved from the start and during the entire project. Architects should work together with the team, share knowledge and create architecture documents. The architecture perspective shows this same finding in only C1. The other cases vouch for fit for purpose or impact-based involvement, where the last of those two is paired with architecture by proxy. However, this is indicated to be ineffective because of disagreements or differences of opinion between the proxy and the architect. The idea that an architect only makes the design up front and then goes on to the next challenge is not valid in an agile environment. The teams themselves demand more from an architect. Grant (1996) describes 4 mechanisms for integrating knowledge (Rules and directives, Sequencing, Routines & Group problem solving and decision making). Traditionally, in enterprise architecture, a rules and directives mechanism is applied. This provides "low cost communication between specialists and a large number of persons" (Grant, 1996). The use of agile knowledge sharing, means moving away from rules and directives to group problem solving and decision making. Group problem solving is to be used when dealing with high "task complexity and task uncertainty". This demands "high-interaction, non-standardized coordination mechanisms" (Grant, 1996). So, the architect working together with, and maintaining a collaborative relationship with the agile team(s) and product owners is important when working with agile teams. This leads to the independent variable 'direct and active involvement of architects', which is defined as the architect working together

with and maintaining a collaborative relationship with the agile team(s) and product owners.

Fit for purpose architecture
All cases point out two things about architecture design. First, it should match the target audience, tailoring the information in the design to the stakeholders and containing the essential information relevant to them. Second, the design should be shared in a way that stakeholders are used to. Because of different ways of working within organisations, the architecture function cannot have a one size fits all approach. Organisations in transition to a more agile way of working will have waterfall development happening alongside agile development. Architecture tailored for its intended use, i.e. fit for purpose, making it suited to a certain audience will increase its acceptance. Fit for purpose is hard to define, since it suggests the possibility of multiple shapes and sizes. Van Steenbergen (2011) states that approaching Enterprise Architecture (EA) from a client perspective, i.e. expressing EA models in a format that is aligned with the way of thinking of the stakeholders, is a good technique to stimulate acceptance of the architecture. Hence, the independent variable 'fit for purpose architecture', which is defined as architecture processes and products tailored to the specific needs of the initiative on hand.

Focus on technical debt and quality
The agile team members of C2 and C3 agree that the architect should have a holistic view of the organisation. The architect must however also be able to go into the technical details if needed, also focussing more on quality aspects. This view is shared by the architects of C1 and C2. The cases show that in an agile environment, the focus of the architecture function is moving to quality aspects of products and to diminishing technical debt levels. According to Ernst et. al. (2015) the dominant source of technical debt is bad architectural decisions. Bad architecture choices stood out from others at 296 of the top three responses (54%) in this study. In terms of the mean rank, the top three were 'Bad architecture choices', 'Overly complex code', and 'Lack of code documentation' (Ernst et. al., 2015). If architecture is the main cause of technical debt, it deserves the architect's attention. However, whether the technical debt caused by architects has anything to do with agile teams taking the role of architects remains to be studied. The independent variable 'focus on technical debt and quality' is defined as keeping track of, and actively managing technical debt levels and the ISO quality aspects to a level accepted by the organisation.

Being flexible when applying rules and principles
In agile environments, the enforcing of principles and rules does not work as it might have done before. The role of architect shifts towards a solution-oriented, supporting expert who is willing to work towards a feasible solution at this moment in time and accepting imperfections for the sake of business value. The architect is challenged to find the right balance between granting teams enough freedom and protecting the IT landscape from unnecessary complexity. Some agile team members claimed that because of a too benevolent approach, the IT landscape suffered. The independent variable 'being flexible when applying rules and principles' is defined as being flexible when it comes to applying

architecture principles and willingness to work towards a feasible solution at this moment in time and accepting imperfections for the sake of business value.

Using patterns and standards

Case C1 points out that standards are an effective way for agile teams to work with architecture and it relieves architecture workload. C3 supports this by stating that because standards and patterns add predefined solutions they aid in the creation of architectures, thereby lowering the workload. Also, at C3, teams create standards and patterns in collaboration with architects. This, in turn, helps them to work effectively. There is a clear trend noticeable across the cases in this research with regard to the use of standards and patterns in relation to agile development. Cases C1 and C3 show that standards and patterns help them work effectively with agile teams: "If we see a gap in a certain solution, we consult the solution architect and together we search for a solution which eventually becomes a new pattern." Having standards and patterns helps achieve synergy across teams and initiatives. Existing literature shows that patterns represent implicit knowledge, which if documented, can be reused by others with the intent of managing the cognitive load while architecting complex systems (Cloutier & Verma, 2006). While developing open source software, improved communications between team members of the architecture and design teams was a measured and quantified result of using patterns (Hashler & Koch, 2004). Hence, the independent variable 'using patterns and standards', which is defined as the application of predefined best-practice solutions and industry standards or self-defined best-practice solutions and organisation standards.

Architecture awareness of the product owner

The final finding from the cases is that when working agile, the product owner (PO) decides the priority of the backlog and with it the work being done on the product. Because the product owner is mostly interested in adding new or improved functionality, architecture and overall quality is put on the back burner. This proves to be unwise since degradation of software ultimately leads to the product being unusable or inhibited to change. When awareness is lacking, the persuasion of the architect is more of a necessity. The 'Understanding/Acceptance Principle' (Slovic & Tversky, 1974) can also be applied to the awareness of the PO. If the PO understands what architecture is, what the risks and opportunities are and why it has added value for the PO, then the perceptibility of the PO towards architecture should increase. This leads to the final independent variable, 'architecture awareness of the product owner', defined as the measure of insight a product owner has into the implications of (not) working with enterprise architecture.

With the insights from the case studies the preliminary conceptual model depicted in figure 1 is revised. The final model has gained additional, more detailed constructs and relations (figure 2).

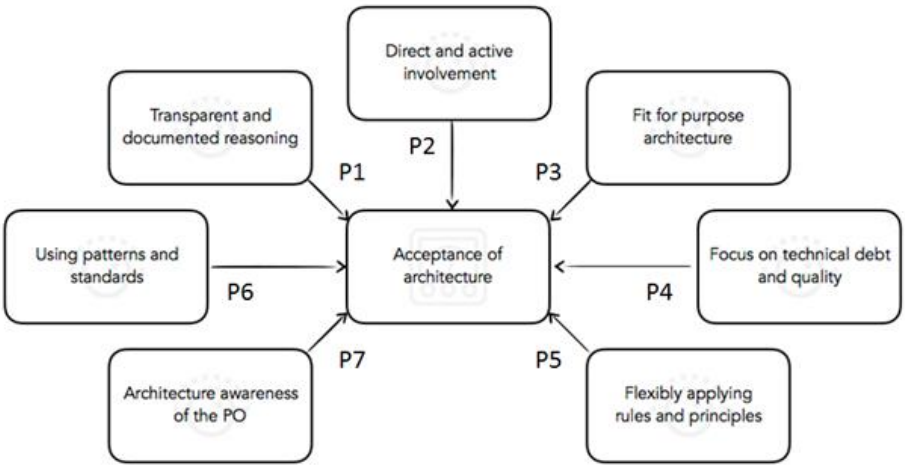

Figure 2: Revised research model

The relations between the variables in the model are expressed with the following seven propositions:

1. Transparency and documentation of reasoning has a positive impact on acceptance of architecture
2. Direct and active involvement of architects has a positive impact on acceptance of architecture
3. Fit for purpose architecture has a positive impact on acceptance of architecture
4. Focus on technical debt and quality has a positive impact on acceptance of architecture
5. Being flexible when applying rules and principles has a positive impact on acceptance of architecture
6. Using patterns and standards has a positive impact on acceptance of architecture
7. Architecture awareness of the PO has a positive impact on acceptance of architecture

Two of the propositions are more like prerequisites whereas the rest of them are activities to perform in a certain way. One of the prerequisites is being flexible when applying rules and principles and avoid forcing them on teams by having a regulatory mentality. The second prerequisite is the increased awareness of the PO. PO's overruling architecture is an enduring challenge and requires the architects' attention. All interaction with the PO should be aimed at increasing awareness and making them understand their responsibility.

From the case study a number of differences between agile and traditional waterfall development emerge. A main insight from the case study, is that compliance should not be the goal when dealing with an agile environment. The goal should be to reach a

measure of acceptance by agile teams where architecture is perceived as an essential ingredient for creating a solution.

Architecture deliverables in an agile environment differ from traditional deliverables in the sense that they are, more than before, aimed at the intended use. Use of standards and patterns by agile teams is also a reason why architecture deliverables have shrunk in size. Everything already described in standards and patterns is left out in architecture deliverables and most new issues and topics are translated into new standards or patterns. This almost makes the architecture a set of standards to apply and patterns to use in the scope of a project or change initiative.

Architecture regulations are not enforced like they are in a traditional environment. Deviations tend to happen in a more collaborative fashion. This means the architect is involved in the process and can influence the deviation and the gravity of the deviation. Because of this collaborative deviation process, deviations can be managed more easily. Deviation is made explicit and can immediately be addressed as technical debt. Because the PO understands the effect of the technical debt (or the architect stresses the importance) choices are made to either fix the technical debt or continue to live with it. Either way the responsibility lies with the PO.

5 Conclusion

The main contribution of this study is the final conceptual model and seven propositions relating architecture efforts to acceptance of architecture by agile teams. The propositions are a first step towards a theory of effective architecture function in an agile development environment. The exploratory nature of this study implies that the propositions and model need to be validated through scientific testing.

Findings from this study add to the understanding of the reasons why architecture and agile development struggle to cooperate. By speaking to team members and architects and analysing the cases a truthful image has been documented and propositions have been generated. These valuable insights provide a foundation to expand upon.

Apart from the scientific contribution this study also has a practical contribution. The findings of the study can be used by practitioners to further develop their architecture and agile development function. The propositions can be translated into the following activities and instructions for architects:

- Architecture functions can start documenting all decisions for everyone to see
- Architects can start attending agile team meetings and working with them on a daily basis
- Architecture deliverables can be adapted to the interests of stakeholders
- The law enforcing attitude can be exchanged for a more collaborative one, discussing the needs for deviation instead of enforcing principles

- Architects can start documenting standards and guidelines in collaboration with the various agile teams to create a catalogue of standards that all teams need to use

By taking the steps described above, the gap between agile teams and architecture functions may close, even if it is just a little and at a slow and steady pace.

This study has some limitations. First, the case selection process was not as thorough and strict as research methods prescribe. Due to discouraging and disappointing reactions by prospective cases, some convenience sampling was applied. Also, Eisenhardt (1989) states that there is no ideal number of cases, but a number between 4 and 10 usually works well to generate complex theory. In that regard, this study lacks one case. However, all three cases are vastly different. This diversity means the current findings are not biased to one industry sector. The generated propositions are likely to be applicable to other similar organisations. Analysis, elaboration and triangulation with existing literature confirmed their relevance. Further testing of the propositions must be done to formalise them.

By using interviews as one of the research techniques there is a risk of subjects being biased. Subjects express lots of opinions and perceptions which could be misleading or false. During this study, no measures were taken to verify whether allegations were truthful. This could lead to results being less reliable (Bryman & Bell, 2011).

The number of people interviewed was limited due to scoping of activities and time limitations.
The two people from each case were however selected to represent their colleagues' perspectives as well. So, the architecture perspective should represent the entire architecture function of that case. The same holds true for the agile team perspective.

The study leaves some interesting venues for further research. First and foremost, the propositions generated by this study need to be tested. Formalising and validating the propositions can be done by focussing on single propositions and testing them using additional cases. Another option is to do a quantitative study using a survey.

As another option for further research, additional cases of different levels of maturity can be studied. This might lead to a maturity model and recommendations for organisations at a certain maturity level. To achieve this, case selection should include criteria which clearly distinguish the difference in maturity.

The culture of an organisation is something which could have an impact on how agile an architecture teams operate. Culture is not an aspect which is analysed in this study. It is however very interesting to investigate what impact an organisations culture has on the propositions generated in this study and how cultural aspects influence collaboration between architecture functions and agile teams.

References

Beck, K. & Beedle, M. & van Bennekum, A. & Cockburn, A. & Cunningham, W. & Fowler, M. & Grenning, J. & Highsmith, J. & Hunt, A. & Jeffries, R. & Kern, J. & Marick, B. & C. Martin, R. & Mellor, S. & Schwaber, K. & Sutherland, J. & Thomas, D. (2001). Agile Manifesto. Retrieved on Wednesday 14th march 2018 from http://agilemanifesto.org/

Eisenhardt, K.M. (1989). Building theories from case study research. Academy of management review, 14(4), pp.532-550. DOI: 10.5465/AMR.1989.4308385

Ernst, N.A., Bellomo, S., Ozkaya, I., Nord, R.L. and Gorton, I. (2015) Measure it? manage it? ignore it? software practitioners and technical debt. In Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering, pp. 50-60. ACM. DOI: 10.1145/2786805.2786848

Galbraith, J.R. (1977). Organization design. Reading, MA: Addison Wesley Publishing Company.

Grant, R.M. (1996). Toward a knowledge based theory of the firm. Strategic management journal, 17(S2), pp.109-122. DOI: 10.1002/smj.4250171110

Hashler, Koch, S (Editor) (2004). Free/open source software development. Hershey PA: Idea Group Publishing.

Kelley, K., Clark, B., Brown, V., & Sitzia, J. (2003). Good practice in the conduct and reporting of survey research. International Journal for Quality in health care, 15(3), 261-266. DOI: 10.1093/intqhc/mzg031

Knodel, J., & Popescu, D. (2007). A comparison of static architecture compliance checking approaches. In Software Architecture, 2007. WICSA'07. The Working IEEE/IFIP Conference on (pp. 12-12). IEEE. DOI: 10.1109/WICSA.2007.1

Petersen, K. and Wohlin, C. (2009). A comparison of issues and advantages in agile and incremental development between state of the art and an industrial case. Journal of systems and software, 82(9), pp.1479-1490. DOI: 10.1016/j.jss.2009.03.036

Poort, E.R. (2014). Driving agile architecting with cost and risk. IEEE Software, 31(5), pp.20-23. DOI: 10.1109/MS.2014.111

Schwaber, K. & Sutherland, J. (2017). The Scrum Guide; The Definitive Guide to Scrum: The Rules of the Game. Retrieved on Wednesday 14th march 2018 from https://www.scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-US.pdf

Skitka, L.J. and Mullen, E. (2002). Understanding judgments of fairness in a real-world political context: A test of the value protection model of justice reasoning. Personality and Social Psychology Bulletin, 28(10), pp.1419-1429. DOI: 10.1177/014616702236873

Slovic, P. and Tversky, A. (1974). Who accepts Savage's axiom?. Systems Research and Behavioral Science, 19(6), pp.368-373. DOI: 10.1002/bs.3830190603

Steenbergen, M. van (2011). Maturity and effectiveness of enterprise architecture. Doctoral dissertation, Utrecht University.

Van Haren Publishing (2011). TOGAF Version 9.1. http://pubs.opengroup.org/architecture/togaf9-doc/arch/

VersionOne, Inc. (2015). 9th annual state of agile survey. Retrieved on Wednesday 14th march 2018 from https://explore.versionone.com/state-of-agile/9th-annual-state-of-agile-report-2

Wilson, N. & Altman, R. (2013). Agile Teams Need Application Architects. Retrieved on Wednesday 14th march 2018.

Yang, C., Liang, P. and Avgeriou, P. (2016). A systematic mapping study on the combination of software architecture and agile development. Journal of Systems and Software, 111, pp.157-184. DOI: 10.1016/j.jss.2015.09.028

Yin. R.K. (2014). Case Study Research Design and Methods (5th ed.). Thousand Oaks, CA: Sage.

Zikmund, W.G., Babin, B.J., Carr, J.C. and Griffin, M. (2009). Business research methods. Mason, OH South-Western, Cengage Learning.