International Conference on Advanced Technologies, Computer Engineering and Science (ICATCES'18), May 11-13, 2018 Safranbolu, Turkey

Software Architecture Documentation in Agile

C.CAKIR, F. CETIN, M. SAVASCI and O. FINDIK

Siemens A.S., Istanbul/Turkey, caglar.cakir@siemens.com
Siemens A.S., Istanbul/Turkey, fatihcetin@siemens.com
Siemens A.S., Istanbul/Turkey, mustafa.savasci@siemens.com
Karabuk University, Karabuk/Turkey, oguzfce@gmail.com

***Abstract* - Over the last decade, agile practices become very popular amongst software development. According to one of the phrases of the Agile Manifesto "Working software over comprehensive documentation" sometimes could lead practitioners to the misunderstanding of "documentation is not valuable" or "not needed at all". Because of this understanding from software community, agile practitioners do not give enough attention to architectural related documents. However, documentation is also a communication way between people and this communication should be also simple and lean considering agile principles. Commonly used traditional architectural documentation is very comprehensive and detailed. Creation and maintenance of this architectural documentation take too much effort for agile teams. Therefore, existing architectural template documents cannot serve agile teams in the best way. Rather than using existing architectural documents, this article presents a new lightweight architectural documentation template that can be used maintained easily in agile projects.**

***Keywords* – Agile, Software Architecture, Architectural Documentation.**

I. Introduction

Nowadays more and more software-driven businesses feel the pressure of responding dynamic market requirements and satisfying customers' needs. These challenges push software industry to embrace agile software methodologies rapidly [1].

Since the publication of *Agile Manifesto* over a decade ago the use agile practices in the software industry has increased significantly as a mechanism for increasing adaptation to changing market requirements. In today's software business, usage of agile methodologies comes to the point that number of surveys shows that agile is now the dominant approach amongst software companies [1]. However, there are still questions and concerns regarding software architecture and documentation practices.

Software architecture describes the key components and their interactions with each other or different parts of the system; therefore it provides the skeleton of the system to all stakeholders [1, 2]. Regardless of the software development methodology, architectural information should be kept under documentation in order to build common understanding between stakeholders. According to Clements et al. documentation is important for building, maintaining and reusing software architecture [3].

According to the "*Working software over comprehensive documentation*" principle of agile, documentation effort mostly seen by practitioners as it is not required and time-consuming. In fact the underlying meaning of this principle is that comprehensive documentation is less important due to high cost of maintenance effort [4]. Considering the importance of the software architecture and agile approaches, finding the middle ground between these concepts could be an ideal way for the software industry and that's the aim of this article.

In the next sections, we will detail agile approaches and architectural related processes, and the tension between them. After that, we will discuss documentation effort of these two cultures. Finally, we will introduce a lightweight architecture documentation template that can be used in agile practices.

II. Agile vs Architecture

According to many agile practitioners, software architecture practices seem to be in the context of plan-driven development paradigm [5]. They feel big upfront design and maintenance of software architecture requires too much work and have little benefit to the customer, additionally all these efforts contains high ceremony processes [6]. Thapparambil [7] writes that "no agile methods discuss Architecture at any length." Architectural design is thought to be unimportant activity amongst agile approaches; therefore there is not enough detailed information about architectural related activities [8]. Thapparambil [7] also sees refactoring as the main activity to replace architecture in the agile world.

On another side, companies working with well-established architectural practices often tend to see agile methodologies as amateurish and more suitable for relatively small applications [9]. According to them, architectural related software development processes cannot be followed using the agile methodologies [6]. Satashi Basaki [9] criticizes agile practitioners with these words "It seems that many agile method users misunderstand what agile methods are, just ignore architecture, and jump onto refactoring". He believes threating "refactoring" as the only remedy for the architectural challenges is a big problem.

Considering both sides opinions it's obvious that the tension between two cultures lies under adaptation versus anticipation. Agile practitioners want to decide architectural decisions at the "last responsible moment". Because of the changing requirements and uncertainties, they have right to think so [9]. On the other hand, some projects need to be planned carefully when the software quality expectations are in at a high level and changes in requirements do not expected to happen very often. Rick Kazman [1] noted this issue as: "No one would want to fly in an aircraft with flight control software that had not been rigorously planned and thoroughly analyzed. Similarly, no one would want to spend 18 months planning an e-commerce website for their latest cell phone model, or video game, or women's shoe style (all of which are guaranteed to be badly out of fashion in 18 months)"

However, paying more attention to architectural disciplines increased significantly amongst agile approaches [5, 10, 11]. Architects have also seemed happy with agile practices. Kati Vilkki surveyed more than 2,400 developers, testers, architects, and managers at Nokia Siemens Networks. The result of the survey showed that more than 70 percent of the architects were satisfied with agile development approaches in their work [12].

Most of the development teams frequently find themselves into this dilemma "should I do agile or architecture" but the real question should be "how much architecture should I need up front or how long should I delay this process until requirements become solid enough" [1]. A short answer to this question is "it depends". Rick Kazman details this answer with the following criteria:

- If you are working on a large complex system with well-defined requirements doing a large amount of architectural work at the beginning is beneficial.
- Be ready to change and maintain architecture when quality attribute requirements emerge and solidify.
- On smaller projects with uncertain requirements, try to find optimal architectural concepts with major functionalities are included. Do not spend lots of time with upfront work.

Both of the two cultures carry their own good values inside. Except for some specific domains, it should not become a choice to pick one of them. Mixing and choosing the right time and effort to combine these cultures might be the key to the success in the software industry.

III. ARCHITECTURAL DOCUMENTATION

Because of the "*Working software over comprehensive documentation*" [13] principle, documentation gets a bad reputation amongst agile approaches. In fact, the underlying meaning of this principle is, eliminate creation and maintenance of the unnecessary and comprehensive documentation due to its high cost [4]. But many agile practitioners forget the fact that documentation has two valuable assets "communication" and "collective memory" [1, 14]. Architectural documentation considered as one of the most important document due to containing the key information about the product which every stakeholder might need it for different reasons. Another principle of agile development "*the most efficient and effective method of conveying information to and within a development team is face-to-face conversation*" describes the way of knowledge sharing, but Clements at al [15] refuses this idea and points the importance of the architectural documentation with these words: "…think of a maintainer, who has inherited the system years after all the original developers have left, trying to understand where to begin. On a project involving hundreds of developers, do you really want your architects to spend all of their time answering the same questions over and over? Or would you rather let documentation serve that purpose, while also making sure the developers get the same answer every time?"

Considering the philosophy of the agile the amount of architectural documentation should be "just enough" [16]. If no one is going to read it, there is no point to generate it but if there is anyone, the document should be lean and simple enough. According to Cockburn, the correct amount of architectural documentation is exactly as needed to accomplish the next step. Any other effort to exceed this limit is a waste of time and money [17]. Finding the middle ground between agile and architecture documentation process brings the following benefits to the product:

- Achievement of critical quality attributes.
- Building a communication channel for the future generations.
- Enabling general system analysis.

IV. PROPOSAL

The main challenges related to architectural documentation could be listed as:

- Understandability of the document by each stakeholder (architects, reviewer, analyst, developer etc.), the complexity of the document.
- Setting the correct information to the right place or finding required information easily [18, 19].
- Keeping the document up to date. Stakeholders eventually will lose the confidence of quality for outdated document [19, 20].

The purpose of the lightweight architectural documentation template is to help software development teams to eliminate or minimize the challenges stated above; another goal of this proposal is, decreasing the creation and the maintenance effort of the architectural document.

The contents of proposed template inspired from one of the most common architectural document templates of "arch42" [21]. As a natural behavior of agile, the contents of this template can be changed from team to team. The main motivation when editing the document should be "Describe what you can't get from code". Here are the contents of the proposed lightweight architecture documentation template:

A. Context

A short set of explanations, context diagram (or both) that answer these questions:

- What is the system all about?
- Who are the users of this product?
- What is the main motivation to build this product?

B. Functional Overview

Summarization for the key functionalities of the system (use cases, user stories)

C. Quality Attributes/Tree

A small set of the most critical quality attributes like performance, security, flexibility etc. Short explanations could be filled for each attribute.

D. Constraints

A small set of the most important constraints that need to be aware of.

E. Software Architecture

This section is the big picture of the system and can contain different types of view like logical, development, physical views. It is preferred to put only diagrams into this section.

F. Decision Log

A short paragraph or set of architectural decision records which explains the most important decisions.

V. CONCLUSION

Agile and architectural processes seem like totally different concepts but both of the cultures carry great artifacts for the software industry. Although agile philosophy advice less documentation, creating a simple and lean software architecture documentation brings great benefits to working product. Documentation creates an alternative communication channel and builds historical memory. These benefits would likely to pay off after the team member changes or project size grows up after few years.

REFERENCES

[1] M.A. Babar, A.W. Brown, I. Mistrik, "Agile Software Architecture" Elsevier, pp. 192, 2014.

[2] R.C. de Boer and H. van Vliet, "On the similarity between requirements and architecture," Mar. 2009.

[3] P. Clements, D. Garlan, R Little, R. Nord, and J. Stafford. (2003). Documenting Software Architectures: Views and Beyond. Proceedings of the 25th International Conference on Software Engineering, pp. 740 741.

[4] S. Sherman and I. Hadar, "Identifying the Need for a Sustainable Architecture Maintenance Process", 5th Int. Workshop on Cooperative and Human Aspects of Software Engineering (CHASE), ICSE, June 2012.

[5] R. Nord,.L, and J.E Tomayko, Software Architecture-Centric Methods and Agile Development, IEEE Software,2006. 23(2): pp. 47-53.

[6] M.A. Babar, 2009. An exploratory study of architectural practices and challenges in using agile software development approaches. Proceeding of the 2009 Joint Working IEEE/IFIP Conference on Software Architecture and European Conference on Software Architecture, WICSA/ECSA 2009, Cambridge, pp: 81-90

[7] P. Thapparambil, Agile architecture: pattern or oxymoron?, Agile Times, 2005. 6(1): pp. 43-48.

[8] C. Hofmeister, et al., A general model of software architecture design derived from five industrial approaches, Journal of System and Software, 2007. 80(1): pp. 106-126.

[9] P. Abrahamsson, M. A. Babar, and P. Kruchten, "Agility and architecture: Can they coexist?" IEEE Software, vol. 27, no. 2, pp. 16–22, Mar.–Apr. 2010.

[10] T. Ihme, and P. Abrahamsson, Agile Architecting: The Use of Architectural Patterns in Mobile Java Applications, International Journal of Agile Manufacturing, 2005. 8(2): pp. 1-16.

[11] R. Parsons, Architecture and Agile Methodologies - How to Get Along,
in WICSA. 2008.

[12] K. Vilkki, "Impact of Agile Transformation," Flexi Newsletter, vol. 2, no. 1, 2008, pp. 5–6.

[13] K. Beck et al., "Agile Manifesto," [Online] Available: http://agilemanifesto.org/, 2001.

[14] A. Cockburn, Agile software development: The cooperative game, 2nd ed. Reading, MA: Addison-Wesley 2007.

[15] P. Clements, J. Ivers, R. Little, R. Nord, and J. Stafford, (2003), "Documenting Software Architectures in an Agile World", Carnegie Mellon University, Software Engineering Institute, CMU/SEI2003-TN-023, Pittsburgh, PA.

[16] R.Hoda, J. Noble & S. Marshall, "Documentation strategies on agile software development projects." International Journal of Agile and Extreme Software Development, 1(1), pp. 23-37, 2012.

[17] A. Cockburn, Agile Software Development. Boston, MA: Addison-Wesley, 2002.

[18] R. C. de Boer and H. van Vliet, "Architectural Knowledge Discovery With Latent Semantic Analysis: Constructing a Reading Guide for Software Product Audits", Journal of Systems and Software, vol. 81, no. 9, pp. 1456-1469, 2008.

[19] S. Sherman, I. Hadar, and M. Levy, "Enhancing Software Architecture Review Process via Knowledge Management", The Sixteenth Americas Conference on Information Systems, AMCIS, Lima, Peru, August 2010.

[20] A. Jansen, P. Avgeriou, and J.S. van der Ven, "Enriching Software Architecture Documentation," The Journal of System and Software. vol. 82, pp. 1232-1248, 2009.

[21] Dr. G. Starke, [Online] Available: http://arc42.org/