A Conceptual Framework for Lean Regulated Software Development

Cawley, Oisin; Richardson, Ita; Wang, Xiaofeng; Kuhrmann, Marco

A Conceptual Framework for Lean Regulated Software Development

Oisín Cawley, Ita Richardson
University of Limerick
Limerick, Ireland
{name.surename}@lero.ie

Xiaofeng Wang
Free University of Bolzano
Bolzano, Italy
xiaofeng.wang@unibz.it

Marco Kuhrmann
University of Southern Denmark
Odense, Denmark
kuhrmann@mmmi.sdu.dk

ABSTRACT

A growing number of companies are discovering that their software development processes must be in compliance with some form of regulation. This is particularly so when it comes to safety-critical or business-critical systems such as Automotive Software, Robotics, Medical Devices or Financial Management systems. These regulations affect the software development process itself in various forms. Furthermore, much attention is being given to ways of improving the efficiency of businesses, for example, by adopting lean principles. This raises the question for how to adopt lean principles for software development within a regulated environment? This poster presents the results of our empirical research into lean and regulated software development. Built from a combination of data sources, we have developed a conceptual framework comprising five primary components. In addition the relationships they have with both the central focus of the framework (the situated software development practices) and with each other are indicated.

Categories and Subject Descriptors

D.2.9 [**Software Engineering Management**]: Software process models

General Terms

Management, Theory

Keywords

conceptual framework, regulations, software development, software Engineering, lean, agile

1. INTRODUCTION

Because software has become so pervasive in society, and forecasts are for this trend to continue, we have come to rely on it more and more in our daily lives [6, 7]. Consequently, when it fails or is misused, the effects can be quite devastating. To counteract this risk, various authorities have introduced regulations that aim to govern how software is developed, how it is secured and how it interacts with other systems. These diverse sources of regulations are increasing as software continues to push boundaries, be misused and get embedded in ways, which were not envisioned before. For example, the Enron scandal 2001, which resulted in the loss of over $11 Billion of investors' and employees stocks and pensions, was due to fraudulent financial reporting [1]. In Panama, 21 patients died from overdoses of radiation during cancer treatment as a result of software failure combined with software misuse [2]—and many other risks to public as collected by Neumann et al. [10]. As a consequence, regulations have been imposed to help reduce the possibility of such events recurring. Therefore, software development teams have to demonstrate that they have a clearly defined and compliant development process, supportive of a risk identification and reduction approach, and backed up by clear and objective evidence of adherence to those processes.

At the same time, software development is also asked to be "lean". Lean, with roots in the manufacturing world [5], is about achieving more with less [5], or producing in one-third the time, at one-third the cost, and with one-third the defect rate [4] through process refinement/change. To become lean, companies eliminate process waste to focus on creating customer value. Looking specifically at lean from a Software Development perspective, Raman [11] concluded that with practices, like rapid prototyping, quality function deployment, continuous integration, object oriented and component-based development: *"The question whether Lean Software Development is Feasible can easily be answered with 'yes'."* The question, however, becomes [12]: What exactly is lean software development? Combining these two concepts (regulated and lean software development) leads us to an even more focused and interesting area, namely, how do the regulations, combined with assuming a lean mind-set, affect the software development process companies adopt.

2. OUR APPROACH

None of the existing literature discussing software development within domains that are subject to regulatory compliance [3, 8, 9], had synthesized the specific effects of regulations in relation to the software development process into a concise framework. We therefore propose a framework for this and further extend it to include the concept of lean software development.

1. A systematic mapping study of the academic literature
2. The experiences of the SD function within a large manufacturing multinational subject to SOX regulations

3. A case study of a multinational medical device manufacturer subject to both U.S. FDA and EU medical device regulations.

3. THE CONCEPTUAL FRAMEWORK

The central component of the framework in Figure 1 reflects the activities contributing to the full software lifecycle, and therefore includes the broader range of activities involved. The cloud shape used to depict this component reflects the somewhat undefined nature of its contents. Since these are what are used to design, develop and deliver the software they are given the central focus within our framework. The situated practices at the centre represent the collection of specific software development practices, which get implemented within any particular development context.
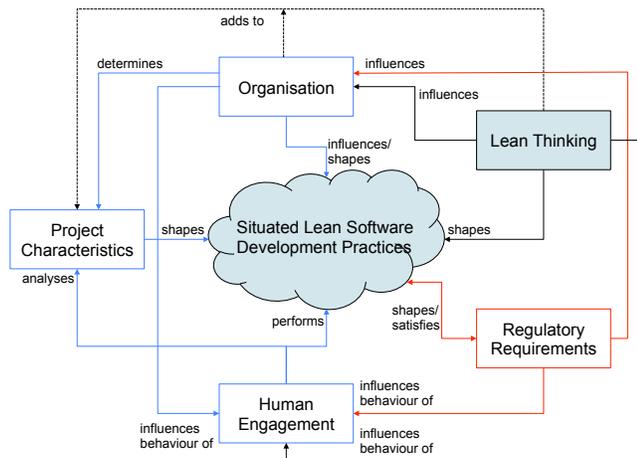

Figure 1: Conceptual framework for lean software development (simplified).

A collection of software practices exist which may or may not be used by the developers depending on a number of factors. These factors are depicted within the framework as the outer five components that encircle the central cloud, indicating their varying influence on the actual software process implementation.

Regulations. Given the increase in regulations, a key component *Regulatory Requirements* aims to reflect the necessity of compliance when performing the software development activities. The framework highlights the relationships between regulatory requirements and the other framework components by means of the red lines. The growing amount of regulation governing software development, for example the expanding definition of a medical device, is forcing an increasing number of organizations to ensure they have compliant development processes.

Lean Thinking. Given the aim of the framework to describe a lean software development context, this component has a prominent position within the framework. At a high level we might view lean and regulatory compliant as contradictory, however we suggest this is not necessarily the case. From an organizational perspective lean suggests a mind-set that should be adopted in all business-related processes in-

cluding software development. In this sense the influence is seen by the manner in which the organization guides and supports the development staff. For example, is decision-making centralized or is the formation of self-managing and autonomous development teams supported? To what extent is process improvement promoted? The seven principles of the *Lean Thinking* component aim to install a lean mind-set in how people carry out their daily activities.

4. REFERENCES

[1] BBC. Enron scandal. Available from: `http://news.bbc.co.uk/2/hi/business/1780075.stm`, 2002.

[2] C. Borrás. Overexposure of radiation therapy patients in panama: problem recognition and follow-up measures. *Revista Panamericana de Salud Pública*, 20:173 – 187, Sep. 2006.

[3] O. Cawley, I. Richardson, and X. Wang. Medical device software development – a perspective from a lean manufacturing plant. In *Software Process Improvement and Capability Determination*, volume 155 of *CCIS*, pages 84–96. Springer, 2011.

[4] R. N. Charette. Challenging the fundamental notions of software development, 2003.

[5] M. Christopher and D. R. Towill. Supply chain migration from lean and functional to agile and customised. *Supply Chain Management: An International Journal*, 5(4):206–213, 2000.

[6] M. Duranton, D. Black-Schaffer, K. De Bosschere, and J. Maebe. The HIPEAC vision for advanced computing in horizon 2020, 2013.

[7] L. D. Eggermont. Embedded systems roadmap 2002, March 2002.

[8] R. Feldmann, F. Shull, C. Denger, M. Host, and C. Lindholm. A survey of software engineering techniques in medical device development. In *Joint Workshop on High Confidence Medical Devices, Software, and Systems and Medical Device Plug-and-Play Interoperability*, pages 46–54, 2007.

[9] B. Fitzgerald, K.-J. Stol, R. O'Sullivan, and D. O'Brien. Scaling agile methods to regulated environments: An industry case study. In *International Conference on Software Engineering*, pages 863–872, May 2013.

[10] P. G. Neumann. Risks to the public. *SIGSOFT Software Engineering Notes*, 15(5):3–17, Oct. 1990.

[11] S. Raman. Lean software development: is it feasible? In *The AIAA/IEEE/SAE Digital Avionics Systems Conference*, pages C13/1–C13/8, 1998.

[12] X. Wang, K. Conboy, and O. Cawley. "Leagile" software development: An experience report analysis of the application of lean approaches in agile software development. *Journal of Systems and Software*, 85(6):1287 – 1299, 2012.