Advances in Streamlining Software Delivery on the Web and its Relations to Embedded Systems

Kasper Hirvikoski

Master's thesis
UNIVERSITY OF HELSINKI
Department of Computer Science

Helsinki, February 4, 2015

HELSINGIN YLIOPISTO — HELSINGFORS UNIVERSITET — UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty	Laitos — Institution — Department
Faculty of Science	Department of Computer Science

Tekijä — Författare — Author
Kasper Hirvikoski

Työn nimi — Arbetets titel — Title

Advances in Streamlining Software Delivery on the Web and its Relations to Embedded Systems

Oppiaine — Läroämne — Subject
Computer Science

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
Master's thesis	February 4, 2015	5

Tiivistelmä — Referat — Abstract

Abstract.

ACM Computing Classification System (CCS):

Avainsanat — Nyckelord — Keywords
keyword

Säilytyspaikka — Förvaringsställe — Where deposited

Muita tietoja — Övriga uppgifter — Additional information

Contents

1 Introduction **1**

2 Software Delivery **2**
2.1 Adapting to Requirements 2
2.2 Ensuring Quality 2
2.3 Processes and Practises 2
2.4 From Agile to Lean 2

3 Deployment Pipeline **3**
3.1 Development 3
3.2 Staging 3
3.3 Production 3

4 Using Web as a Platform **3**
4.1 Continuous Integration 3
4.2 Continuous Deployment 3
4.3 Continuous Experimentation 3

5 Towards Embedded Systems **3**
5.1 Using Hardware as a Platform 3
5.2 Adapting for Deployment Pipeline 3

6 Example Case **4**

7 Conclusions **4**

References **4**

1 Introduction

Software delivery on the web has over the years evolved into a rather established process. A software is developed iteratively through multiple stages, which ensure the user's requirements and the quality of the product or service. These stages form what is called the deployment pipeline [Fow06, HF11, Fow13a, Fow13b].

The deployment pipeline nowadays usually consists of at least three stages: development, staging and production. Organisations alter these stages depending on their size and needs. Using modern iterative and incremental processes, a software is developed feature by feature by iterating through these stages. Development starts in the development stage where developers build the feature requested by the customer or user. The feature is then tested in the staging phase, which represents the production setting. When the feature has been validated, it is then deployed to production. If necessary, each stage can be repeated until the feature is accepted. The stages are short and features are deployed frequently — in some cases even multiple times a day [O'R11, Sny13, Rub14].

Software engineering consists of various different processes and practises for ensuring the quality of the product or service — nowadays more or less based on Agile and Lean ideologies and practises [BBvB+01, Fow05, Mon12]. At the low level, developers use source code management to keep track of changes to the software and to collaborate with other team members. To reinforce that the features work as intended, developers write automated test cases. Teams can also use more social methods — such as reviewing each other's code — to validate the implementations. These practises form the basis for Continuous Integration and Continuous Deployment [Fow06, HF11, Fow13a, Fow13b]. Software changes are frequently integrated, tested and deployed — automatically in each stage. The first two form Continuous Integration and the latter Continuous Deployment. If any stage fails, the process starts from the beginning.

The web enables the use of the deployment pipeline and its practises in an unprecedented way [KLSH09]. Due to the distributed nature of the web, software can be deployed as needed and the user always sees the newest version without the need of any interaction. This eases the use of many cutting-edge methods [KLSH09].

Not all software can be developed easily this way. Many embedded systems, which have a dedicated function within a larger mechanical or electrical system, require hardware to accompany the software. This presents a variety of challenges to overcome. Hardware can require thorough planning and iterating can take time. Contexts such as cross-platform support, robotics, aerospace and other embedded systems pose interesting cases. Many of these contexts can at a glance seem regarded as models for more traditional sequential software engineering processes with heavy planning, documentation

and long development phases. However, even NASA's earlier missions have iterated on the successes and failures of previous ones. Even though it can be more difficult, hardware can be build and tested iteratively with new approaches such as prototyping and 3D-printing.

This raises an interesting research topic — *presenting the advances in streamlining software delivery on the web and relating its practises and their advantages and challenges to embedded systems.* Using case studies to identify which practises are used, how they could be improved and how new practises could be incorporated to these settings.

The approach of my thesis is to identify how modern software development methods, such as Agile and Lean, suit embedded settings and if and how they have been adapted to these environments. Moreover, the aim is to identify which modern Continuous Integration, Delivery and Experimentation practises are used, how they could be improved and how new practises could be incorporated to more embedded settings. Can we determine how they compare to web practises?

The hypothesis is that there should be no reason why these practises could not be successfully used and cleverly adapted to hardware settings.

My research method for this thesis was reviewing the current practises in literature and industry. I also conducted several interviews with the industry working on leading embedded systems to get a view on if and how the deployment pipeline has changed the development of hardware related products.

2 Software Delivery

Software delivery has changed notably in the past few decades.

2.1 Adapting to Requirements

Adapting to requirements.

2.2 Ensuring Quality

Ensuring quality.

2.3 Processes and Practises

Processes and practises.

2.4 From Agile to Lean

From Agile to Lean.

3 Deployment Pipeline

Deployment pipeline.

3.1 Development

Development.

3.2 Staging

Staging.

3.3 Production

Production.

4 Using Web as a Platform

Using Web as a platform.

4.1 Continuous Integration

Continuous Integration.

4.2 Continuous Deployment

Continuous Deployment.

4.3 Continuous Experimentation

Continuous Experimentation.

5 Towards Embedded Systems

Towards embedded systems.

5.1 Using Hardware as a Platform

Using hardware as a platform.

5.2 Adapting for Deployment Pipeline

Adapting for deployment pipeline.

6 Example Case

Example case.

7 Conclusions

Conclusions.

References

[BBvB+01] Beck, Kent, Beedle, Mike, Bennekum, Arie van, Cockburn, Alistair, Cunningham, Ward, Fowler, Martin, Grenning, James, Highsmith, Jim, Hunt, Andrew, Jeffries, Ron, Kern, Jon, Marick, Brian, Martin, Robert C., Mellor, Steve, Schwaber, Ken, Sutherland, Jeff, and Thomas, Dave: *Manifesto for agile software development*, 2001. `http://agilemanifesto.org`, (accessed 13 January 2015).

[Fow05] Fowler, Martin: *The new methodology*, 2005. `http://martinfowler.com/articles/newMethodology.html`, (accessed 13 January 2015).

[Fow06] Fowler, Martin: *Continuous integration*, 2006. `http://martinfowler.com/articles/continuousIntegration.html`, (accessed 13 January 2015).

[Fow13a] Fowler, Martin: *ContinuousDelivery*, 2013. `http://martinfowler.com/bliki/ContinuousDelivery.html`, (accessed 13 January 2015).

[Fow13b] Fowler, Martin: *DeploymentPipeline*, 2013. `http://martinfowler.com/bliki/DeploymentPipeline.html`, (accessed 13 January 2015).

[HF11] Humble, Jez and Farley, David: *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation*. The Addison-Wesley Signature Series (Fowler). Addison-Wesley, 2011, ISBN 9780321601919.

[KLSH09] Kohavi, Ron, Longbotham, Roger, Sommerfield, Dan, and Henne, Randal M.: *Controlled experiments on the web: survey and practical guide*. Data Mining and Knowledge Discovery, 18(1):140–181, 2009, ISSN 13845810. `http://dx.doi.org/10.1007/s10618-008-0114-1`.

[Mon12] Monden, Yasuhiro: *Toyota Production System: An Integrated Approach to Just-In-Time, 4th edn.* CRC Press, 2012, ISBN 9781439820971.

[O'R11] O'Reilly: *Velocity 2011: Jon Jenkins, "velocity culture"*. YouTube, 2011. `https://youtube.com/watch?v=dxk8b9rSKOo`, (accessed 13 January 2015).

[Rub14] RubyKaigi: *Continuous delivery at GitHub - RubyKaigi 2014*. YouTube, 2014. `https://youtube.com/watch?v=Rhvri5cozTc`, (accessed 13 January 2015).

[Sny13] Snyder, Ross: *Continuous deployment at Etsy: A tale of two approaches*, 2013. `http://slideshare.net/beamrider9/continuous-deployment-at-etsy-a-tale-of-two-approaches/`, (accessed 13 January 2015).