

# **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 3, 2015

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 3, 2015	4	
Tiivistelmä — Referat — Abstract			
<p>Abstract.</p> <p>ACM Computing Classification System (CCS):</p>			
Avainsanat — Nyckelord — Keywords			
keyword			
Säilytyspaikka — Förvaringsställe — Where deposited			
Muita tietoja — Övriga uppgifter — Additional information			

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Approach</b>	<b>2</b>
2.1	Research Method . . . . .	2
2.2	Case Studies . . . . .	2
<b>3</b>	<b>Software Delivery</b>	<b>2</b>
3.1	Adapting to Requirements . . . . .	2
3.2	Ensuring Quality . . . . .	2
3.3	Processes and Practises . . . . .	2
3.4	From Agile to Lean . . . . .	2
<b>4</b>	<b>Deployment Pipeline</b>	<b>2</b>
4.1	Development . . . . .	3
4.2	Staging . . . . .	3
4.3	Production . . . . .	3
<b>5</b>	<b>Using Web as a Platform</b>	<b>3</b>
5.1	Continuous Integration . . . . .	3
5.2	Continuous Deployment . . . . .	3
5.3	Continuous Experimentation . . . . .	3
<b>6</b>	<b>Towards Embedded Systems</b>	<b>3</b>
6.1	Using Hardware as a Platform . . . . .	3
6.2	Adapting for Deployment Pipeline . . . . .	3
<b>7</b>	<b>Case</b>	<b>3</b>
<b>8</b>	<b>Conclusions</b>	<b>4</b>
	<b>References</b>	<b>4</b>

# 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.

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.

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. 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. 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. 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.

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.

## **2 Approach**

Approach.

### **2.1 Research Method**

Research method.

### **2.2 Case Studies**

Case studies.

## **3 Software Delivery**

Software delivery.

### **3.1 Adapting to Requirements**

Adapting to requirements.

### **3.2 Ensuring Quality**

Ensuring quality.

### **3.3 Processes and Practises**

Processes and practises.

### **3.4 From Agile to Lean**

From Agile to Lean.

## **4 Deployment Pipeline**

Deployment pipeline.

## **4.1 Development**

Development.

## **4.2 Staging**

Staging.

## **4.3 Production**

Production.

# **5 Using Web as a Platform**

Using Web as a platform.

## **5.1 Continuous Integration**

Continuous Integration.

## **5.2 Continuous Deployment**

Continuous Deployment.

## **5.3 Continuous Experimentation**

Continuous Experimentation.

# **6 Towards Embedded Systems**

Towards embedded systems.

## **6.1 Using Hardware as a Platform**

Using hardware as a platform.

## **6.2 Adapting for Deployment Pipeline**

Adapting for deployment pipeline.

# **7 Case**

Case.

## 8 Conclusions

Conclusions.

## References