MASTER'S THESIS | LUND UNIVERSITY 2018

In-depth study of the potentials of web-based deployment in product development

Patrik Danielsson, Tom Postema

Department of Computer Science
Faculty of Engineering LTH

ISSN 1650-2884
LU-CS-EX 2018-34

In-depth study of the potentials of web-based deployment in product development

Patrik Danielsson
dat13pda
patrik.danielsson.041@student.lu.se

Tom Postema
dat13tpo
tom.postema.463@student.lu.se

June 20, 2018

Master's thesis work carried out at Axis Communications AB.

Supervisors: Theis Hasselgaard, theish@axis.com,
Hussan Munir, hussan.munir@cs.lth.se

Examiner: Per Runeson, per.runeson@cs.lth.se

Abstract

Context: The Cloud has significantly reduced the time it takes to get feedback from customers in software development. However, running an application in an Infrastructure as a Service such as Amazon AWS still requires significant skills and resources.

Aim: To investigate the drivers and associated values with an innovation platform for Axis Communications AB.

Method: With the help of interviews we investigated why an innovation platform could be attractive to Axis and what the drivers are for Axis to create it. A prototype in the form of an innovation platform is developed, which can be used by developers both at Axis and Axis' partners to develop applications and quickly get feedback. With the help of a focus group, the platform is evaluated on several metrics. The response from the focus group is then evaluated and analyzed to measure its relevance for Axis.

Results: The participants of the focus group were very positive to the easy deployment enabled by the use of a Platform as a Service. A majority of the focus group agreed to the statement that the development of the platform should continue, but were neutral to the idea of having it open source.

Conclusion: The developed prototype illustrated the concept of an innovation platform well and the participants of both the interviews and the focus group agreed that an innovation platform could bring value to Axis.

Keywords: web-based deployment, cloud services, innovation platform, product development, open innovation

Acknowledgments

We would like to thank our supervisor and manager at Axis Communications AB. We would also like to thank our academic supervisor Hussan Munir for your guidance during our master's thesis and our examiner Per Runeson. Furthermore, we would like to thank everyone who in some way has helped us with completing our master's thesis, either by being interviewed, by participating in our focus group or by contributing with knowledge or help in some other way.

Contents

Contribution Statement

The initial idea of this thesis came from Axis but was then refined by us. Most of the time during this master's thesis, we were working together. When developing the prototype and the proof of concept, we were pair programming. While working on the interviews however, we divided the workload so that one of us was transcribing and writing a summary of an interview while the other was either writing things in the report or working on the implementation. We took turns transcribing. All parts of the report were written by both authors. Whenever one of us had written parts of a section, the other one reviewed those parts. We were both conducting the interviews and organizing and taking part in the focus group. When emails were to be written, we always made sure that we both agreed on the contents of the email. Any decisions made during the thesis work were made after a discussion between the two of us.

Chapter 1
Introduction

Technology is accelerating and in an ever-changing industry there is a need to develop applications quicker and get feedback faster. Cloud computing can simplify the deployment process but common cloud solutions such as Infrastructure as a Service (IaaS) can come with a great amount of complexity. Platform as a Service (PaaS) helps reduce some of the cumbersome complexity whilst still providing many of the features necessary. In this project we evaluate how an innovation platform could be beneficial to Axis Communications AB by interviewing experts, we develop a prototype platform and evaluate the platform through the use of a focus group.

This chapter includes the objectives of this project, the overall structure, and the most commonly used abbreviations. Open innovation and cloud computing are two major areas related to this project. They are therefore introduced in the beginning of this chapter.

1.1 Open Innovation

Rose and Furneaux describe innovation as the creativity that creates inventions which reaches a wider use. It includes changes to the practices of individuals or groups which are brought on by new ideas [78]. The concept of open innovation was defined by Chesbrough in 2003 in his book *Open Innovation: The New Imperative for Creating and Profiting from Technology* [15]. In his book *Open Innovation: Researching a New Paradigm*, he describes it in one sentence as *"the use of purposive inflows and outflows of knowledge to accelerate internal innovation, and expand the markets for external use of innovation, respectively"* [14]. Chesbrough explains that outside-in and inside-out are two types of open innovation that are significant. Outside-in is about getting input from external contributors. Inside-out, on the other hand, is about letting other businesses take over ideas [16]. According to Munir et al., an example of outside-in is acquiring a start-up company while an example of inside-out is selling intellectual property [62].

Inside-out and outside-in can through *the coupled process* be linked together among

organizations. The process works by companies forming interrelationships and joint ventures where they are both giving and taking when collaborating [27, 62].

1.2 Cloud Computing

Cloud computing enables the hosting and providing of services over the internet. It works by allowing clients to lease resources, for example CPU utilization and storage, based on the current needs [83]. There are some major advantages to cloud computing as mentioned in a paper by Zhang et al., including: No up-front investment, lower operating costs, high scalability, easy access, and reduced business risk and maintenance expenses [83].

There are three different services of cloud computing:

- Infrastructure as a Service (IaaS)
- Platform as a Service (PaaS)
- Software as a Service (SaaS)

According to a paper by Goyal, the difference between these services is what is up to the customers to implement and what is up to the provider to implement. IaaS provides all necessary infrastructure to the clients, which allows for almost full control over the software. An example of IaaS is Amazon AWS.

PaaS works in such a way that it offers developers a development stack with ready-to-use libraries, databases, networks and more. PaaS providers therefore provide less control for its customers, with the benefit of less configuration [32]. Microsoft Azure is one example of PaaS.

SaaS takes this concept even further. SaaS allows deployment of specific already developed applications in a cloud infrastructure. This gives clients access through a simple interface, for instance a web browser [56]. An example of SaaS is Google Docs.

1.3 Objectives

This section includes the purpose of this project and the overall goals which we aim to achieve.

1.3.1 Problem Statement

The Cloud has significantly reduced the time it takes to get feedback from customers in software development [33]. This enables developers to solve customer needs faster and with better quality.

Running an application in an IaaS [56] such as Amazon AWS still requires significant skills and resources. This thesis will investigate the benefits and drawbacks of using a commercial PaaS [56] as a complement or a replacement.

The main purpose of this master's thesis is to examine both the technical and business potentials a PaaS built platform offers to a more traditional company, in this case Axis Communications AB. We are going to investigate what could be beneficial for Axis' developers or developers at one of Axis' partners when building applications using a PaaS

in conjunction with our PaaS platform and determine if it could impact their workflow. Furthermore, we are going to explore whether a PaaS platform can reduce complexity and make it possible to quickly create prototypes, thus enabling quicker feedback on new features.

1.3.2 Goals

The goals of this master's thesis are the following:

- Map the potential benefits for Axis, from a technological but also from a business perspective.
- Build a prototype based on the drivers and recommendations from Axis.
- Evaluate the prototype and compare it to Axis' current solution.

1.4 Structure of the Report

The report has been structured in the following way. First, chapter 2 discusses related work after which chapter 3 describes the research method, including the research questions, research methodology and validity threats. Chapter 3 also includes a small section about Axis. Next, chapter 4 includes the steps taken when implementing the prototype but also the steps taken for the interview and the focus group while chapter 5 contains results and an analysis. Chapter 6 then contains a discussion. Finally, the report is concluded in chapter 7.

1.5 Abbreviations

Table 1.1: Abbreviations used in the report

Abbreviation	Expression
3LPM	Three Layer Product model
ACC	Axis Camera Companion
API	Application Programming Interface
AWS	Amazon Web Services
CORS	Cross-Origin Resource Sharing
IaaS	Infrastructure as a Service
OI	Open Innovation
OIDC	OpenID Connect
OSS	Open Source Software
PaaS	Platform as a Service
SaaS	Software as a Service
SVM	Software Value Map

Chapter 2
Related Work

There are many aspects of research to consider in this thesis, as building a prototype and evaluating it results in a large amount of possible aspects to focus on. This chapter includes related work about PaaS platforms and also related work from the areas of continuous delivery, open innovation, security and the value of software. The related works we have chosen are chosen either based on the fact that they focus on the what the benefits could be for the developer or the company from a research perspective, or they are chosen based on the viability of the project from a software development perspective.

2.1 PaaS

As mentioned in section 1.2, there are several differences between the cloud computing alternatives IaaS, PaaS and SaaS. A study by Goyal describes different cloud services and their benefits. In this study, one major benefit that is mentioned with PaaS is the flexibility to choose features and tools as needed while the PaaS provider handles security, backups and recovery [32]. According to a study by Costache et al., the increased speed of the application deployment is also a characteristic of PaaS [17].

2.2 Continuous Delivery

Continuous delivery is about *"ensuring the software* can *be released and/or deployed to production at anytime"* [82]. Cloud computing enables continuous delivery and a study by Austel et al. suggests ways to improve continuous delivery for cloud systems and one of the ways to do this is to use the features already present in many PaaS platforms [1].

One of the most important parts of continuous delivery is the shorter release cycles. Instead of having fixed cycles, continuous delivery enables automated deployment to production. According to a study by Neely and Stolt, there are several benefits to this, among

others: No planning for a release, requiring tighter discipline and an incremental delivery of value [64].

2.3 The Three-Layer Product Model

According to a paper by Bosch, product architectures can be organized in three different layers. These are the 'Innovation and experimentation layer', the 'Differentiating functionality layer' and the 'Commoditized functionality layer' [11]. At Axis, our prototype will fit into the experimentation layer. We will in this report investigate what potential benefits this entails.

2.4 Levels of Openness

Open source software (OSS) means that the source code of the software is available to be freely examined, copied and augmented for any purpose.

Making the prototype and/or proof of concept open source could have many benefits, such as broader adoption and a higher degree of innovation [62]. Openness should be seen as a continuum and there are various levels of openness varying from completely open, to partly transparent, to completely closed [60].

In our case we are using OSS as an example of open innovation. There are also considerations to make when choosing what should and should not be open-source. This has been discussed in multiple studies. One of these studies is a study by Linåker et al., in which a Contribution Acceptance Process (CAP) model is proposed which aims to help in making that decision [60]. The model provides a classification with respect to business impact and control complexity after which it creates contribution strategies which firms can adopt. Another study by Munir et al. discusses the openness when it comes to software engineering tools [63]. In this study, the result is *"a theory of openness"*, discussing both strategy, triggers, outcomes and level of openness. Both of these studies can be used to determine the level of openness when it comes to our prototype and/or proof of concept.

2.5 Security

When implementing the prototype, the security of cloud computing should be considered. A number of studies discuss the challenges regarding cloud security. One of these studies is a study by Dhaivat et al. that categorizes different cloud security issues in several layers [20].

Hussain et al. list the major security concerns associated with IaaS and PaaS cloud services as being hardware and software virtualization. They also mention possible attacks with potential risks varying from low to high [57]. Security threats are important to take into consideration when choosing the PaaS platform.

Based on the findings of this project, we are going to discuss the security of cloud providers in the analysis section of our report.

In the comparison in section 4.1 we evaluate the PaaS platforms fixes to the Spectre and Meltdown vulnerabilities.

2.6 Value

Calculating or estimating the value of software is not an easy task. In a paper by Khurum et al., a software value map (SVM) is proposed. The SVM categorizes software value into four 'perspectives' and a number of sub categories [59]. This study will be used to map the value our prototype has with regard to a number of different aspects. Whenever an aspect is lacking in the software value map, we define our own.

2.7 Summary

In our research, which is based on design science [55], we evaluate research aspects of the prototype and reason about the drivers and values. There is one study (Brad et al.) that is similar to our project, which is a study about a tool used specifically for open innovation [12]. This tool is however a supportive tool which can aid open innovation, and not a tool used for the deployment and running of applications. This study evaluates how an innovation platform can be used by developers to build their own applications, and what benefits and challenges this may entail.

Chapter 3
Research Method

This chapter starts with brief descriptions of Axis and our chosen platform Heroku. Furthermore, it describes the research methods used, including the research questions and the research design. It also discusses validity threats and confidentiality.

3.1 Case Company

Axis Communications AB, or Axis for short, is a company founded in 1984 that is a pacesetter within network video, launching the first network camera in the world in 1996. Nowadays they have partners in 179 countries all around the world with a vision of *"Innovating for a smarter, safer world"*. In December 2017, Axis had 2865 employees [67].

Axis offers a service called Axis Connect which is described by Axis as a *"cloud based platform to connect devices to the cloud"*. It provides communication between a device and the cloud and currently it includes functionality such as health monitoring and firmware management.

One of the reasons Axis cites for its quick growth is that they do not sell cameras directly to end consumers. Instead, they sell the cameras to third party distributors who then sell to resellers and system integrators [9].

Axis is an avid user and contributor of open source software (OSS). They use tools like Jenkins, frameworks such as GStreamer and they are also discussing, together with other companies, strategies for how to use open source. If they have the possibility to choose an open source alternative they most of the time do, and the reason is, according to an expert, that they would not be able to do many of the things that they were doing if it were not for OSS.

3.2 Selected Platform

Based on a comparison between a number of different PaaS platforms as described in section 4.1, we decided to build our platform on Heroku. Heroku is a container based PaaS [54]. The applications deployed to Heroku gets packaged together with its dependencies into containers. The containers run on a shared host in which it is able to run and scale all applications. More specifically, a container on Heroku is known as a 'dyno' and the application can be scaled to a certain number of dynos based on the needs of the developer [47].

3.3 Research Questions

There is a study similar to ours (Brad et al.) which introduces a platform for enabling open innovation. The platform was successful in bringing together many small and medium enterprises which contributed information [12]. This platform is however used as a supportive tool in software development and cannot be used in a standalone manner. Our platform is used to deploy and run applications which introduces different benefits and challenges which have to be mapped.

To be able to find the requirements and to understand how the implementation should be constructed we have to identify what Axis values and why they value it. Poor requirements engineering can often lead to unmet requirements [66] and this is something we would like to avoid. Initially, the most important aspect is to identify what potential benefits Axis sees with using a PaaS for cloud deployment. This introduces the first research question:

> **RQ1** What are the drivers for Axis to create an innovation platform?

In order to investigate the possibilities that web based deployment has to offer but also to facilitate an ease of comparison towards the current solution, a prototype will be built. The second research question is therefore:

> **RQ2** What are the potentials of web-based deployment?

This involves exploring which platform should be used, how the prototype should be implemented and how an evaluation of the prototype can be done.

We have to map the benefits and values of this project, and after we have built the prototype it will be easier to evaluate the concept as we will have something real to compare the current solution to. This leads to the final research question:

> **RQ3** What value does the innovation platform create and why is it attractive to Axis?

This includes reasoning about (technical) limitations and benefits when using PaaS versus IaaS, including, but not limited to, the time to market, the workflow and the support for testing and methodologies such as continuous development. Khurum et al. [59] will be used to map and categorize the perceived values in order to identify how we measure the values.

3.4 Research Design and Operation

This section contains a description of the research method used. It is based on Hevner's seven guidelines for design-science research [55]. A brief overview of these seven guidelines can be found in table 3.1. In the following subsections, (Hevner X) refers to this table. According to Hevner, a design-science research is not complete unless all guidelines have been considered [55]. Figure 3.1 shows an overview of the research methods, including objectives.

Table 3.1: A summary of Hevner's guidelines [55]

Nr	Guideline	Description
1	Design as an Artifact	When doing a design-science study, the result is a useful IT artifact.
2	Problem Relevance	A community and the relevance with respect to this community should be defined. Furthermore, the requirements on the artifact should be investigated.
3	Design Evaluation	Appropriate metrics should be defined for the evaluation of the artifact and feedback should be provided to the construction phase. The artifact is complete if it satisfies the requirements and constraints defined in the artifact design.
4	Research Contributions	The contributions should be specified. This can either be the artifact itself, a design foundation or a methodology.
5	Research Rigor	Good methods are required when constructing and evaluating the artifact. Furthermore, the artifact should be well-defined.
6	Design as a Search Process	Designing an artifact is an iterative process.
7	Communication of Results	Results should be presented in an effective way, either adjusted for developers (technological) or managers (managerial).

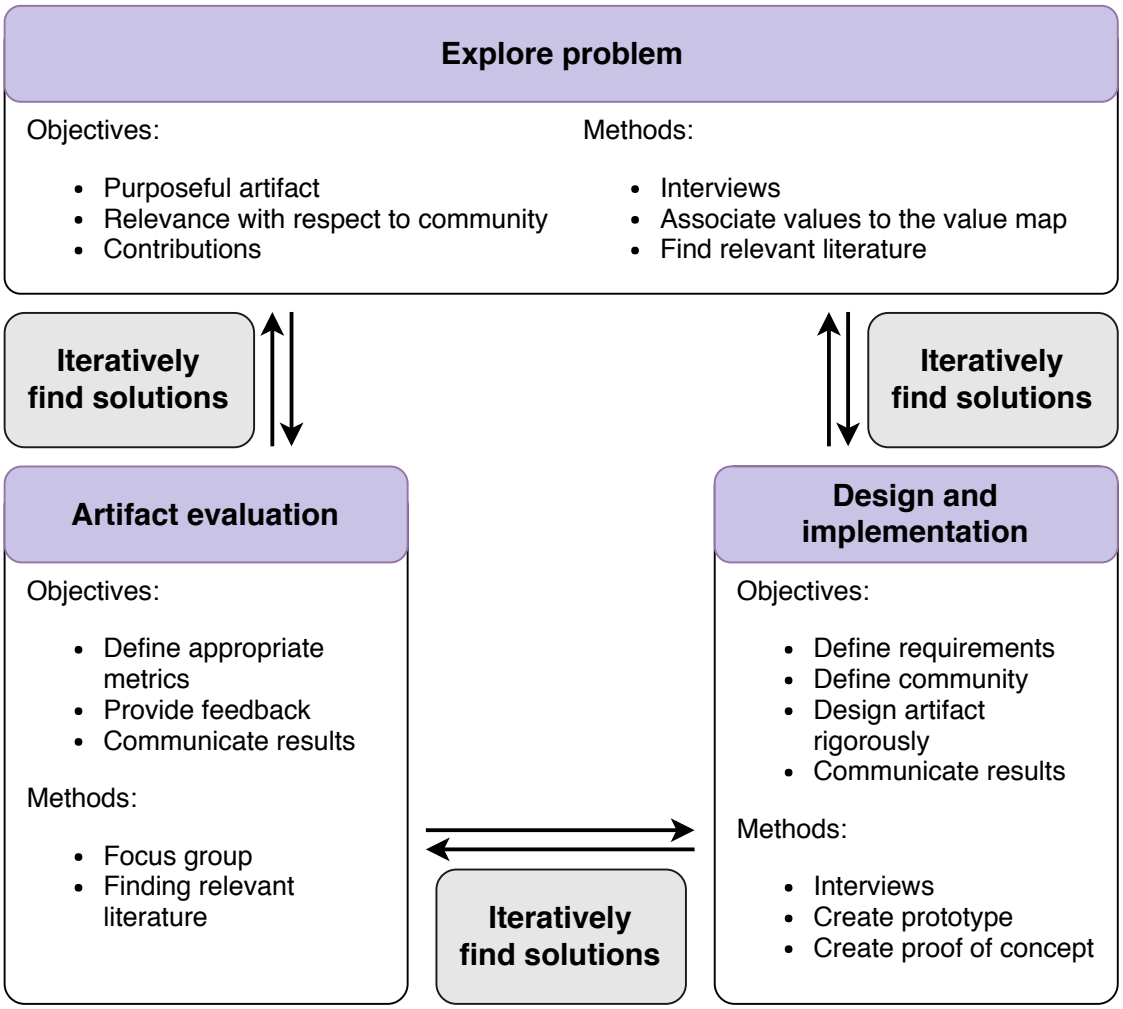


Figure 3.1: Overview of the research method

3.4.1 Explore Problem

When investigating a problem, the objective is to fully understand the problem and confirm its relevance. One of the measures that will be taken to get more insight into why web-based deployment might be beneficial in product development is to conduct five interviews. This gives us the opportunity to investigate the drivers and the value for Axis, thereby addressing **RQ1** and **RQ3**. By conducting the interviews, we will also find out the relevance of our prototype with respect to a community (Hevner 2).

Selecting the Interviewees

Two interviewees were selected by us, two other interviewees were suggestions by our supervisor at Axis and a fifth interviewee volunteered to be interviewed after we sent out an email stating that we were looking for people to interview. The demographics of the interviewees can be found in table 3.2.

Table 3.2: Demographics of the interviewees

Interviewee (anonymized)	Title	Years of experience
A	Software developer ACC	5 years with Axis
B	Experienced Web Engineer	11-12 (3 years with Axis)
C	Director	6 years with Axis
D	Engineering Manager	10 years with Axis
E	Experienced Engineer	4 years with Axis

Interview setup
An initial set of interview questions, which can be found in Appendix A.1, was formulated. After every interview, these steps were performed:

1. Conduct the interview.
2. Transcribe.
3. Make a summary.
4. Validate the summary.
5. Look at the interview questions and add questions and possibly improve the questions before the next interview. Search available literature based on the answers given in order to find comparable cases which might improve the understanding of the research area and its relevance.

Thematic analysis
After conducting three interviews, a thematic analysis was done following this process:

1. Identify themes (Table 5.1) and define them.
2. Label statements with the themes found in step 1.
3. Answer the research questions based on the labeled statements.

The results from the thematic analysis can be found in section 5.1.1.

Focus group setup
An initial set of survey questions was formulated. After two rounds of feedback and revisions of the survey, we sent the updated survey to someone to try it out. The survey can be found in appendix B. Since there were no additional comments, we invited a number of people to participate in our focus group. When it was time for the focus group, six people were in attendance. The demographics of the participants of the focus group can be found in table 3.3.

Table 3.3: Demographics of the focus group participants

Participant (anonymized)	Title	Years of experience
1	Software Engineer	10
2	Experienced Software Engineer	9
3	Engineer	2
4	Developer	10
5	Experienced Engineer	10
6	Senior Engineer	15

During the focus group meeting we started with a short presentation. Next, everyone attending was given a link to a survey that they filled in. Finally, we had a discussion based on the answers given in the survey.

After the focus group, the following steps were performed:

1. Transcribe.
2. Make a summary.
3. Validate the summary.
4. Label statements according to the themes as defined in table 5.1.

The results from the focus group can be found in section 5.3.2.

According to Hevner, the result of a design study is a purposeful IT artifact (Hevner 1) [55]. In our case, this artifact is a prototype that makes it possible to implement applications using the prototype, thereby quickly trying out new features and get feedback from customers within hours. Our contribution (Hevner 4) is a new and effective way to quickly try out things in the 'innovation layer', as described by Bosch [11]. By choosing an appropriate platform and building the prototype, we address **RQ2**.

3.4.2 Designing an Artifact

Hevner states that, in order to design the artifact, we need to investigate which requirements are to be met (Hevner 2) and we need to define the community (Hevner 2) [55]. The interviews mentioned above are also used to understand these aspects. Hevner's fifth guideline refers to the methods used when constructing the artifact, addressing the way in which research is conducted [55].

The following are the initial requirements and constraints for the implementation based on our perception of the thesis description and initial discussions with experts at Axis:

- Easy to scale
- Faster time to market
- Easy to use
- Easy to receive payment
- Portable
- Secure

By conducting interviews (Hevner 5), we clarify these requirements and find additional constraints and requirements on the system. The analysis from the results of the interviews can be found in section 5.1.1.

To build the prototype mentioned in section 3.4.1, an evaluation of different PaaS solutions needs to be done at first, based on a set of criteria as defined in section 4.1. After selecting a PaaS solution, the prototype is implemented. When the prototype is implemented, an iterative process is used (Hevner 6). By working like this, we can make sure that we at all times have a working prototype. Results are presented continuously to our supervisors and our manager (Hevner 7).

3.4.3 Artifact Evaluation

Evaluation is an important part of the research and it is important to define the metrics of which to evaluate the artifact properly (Hevner 3), and to define which data might be appropriate to use [55].

We evaluate which PaaS platform should be used for the prototype. Each of the possible platforms on which we can build our prototype is evaluated by an initial set of requirements. The discussions we had with experts at Axis gave us insight into the selection criteria that could be used to determine which platform as a service should be selected for implementing the prototype. The PaaS platform should:

- Enable faster time to market.
- Decrease overall complexity related to managing a cloud service.
- Ensure ease of extendability for developers, i.e. make the prototype provide base functionality for developers to use for their add-ons which enables them to realize their objectives.
- Be open-source and it should be possible to extend and enhance it through collaboration.
- Make it easy to get some kind of payment for using the add-on.

There are initial aspects that are deemed important to the project. Following the software value map (SVM) these aspects can be divided into four different perspectives, namely: 'Financial', 'customer', 'internal business process', and 'innovation and learning' [59].

A short description of each value and how they are categorized:

- Functionality: The major perspective here is for the internal business, how the functionality could bring value as a tool mostly directed towards improving the internal development.
- Completeness: Here we are talking about the overall packaging of all perceived values, that the artifact itself feels complete and whole.
- Performance: This is a measurable value specifically related to time. Examples of this are how long it takes to deploy, time to set up and time to learn the intricate details.
- Reliability: Related to the rigidity of the final artifact, if the user (customer) is able to trust the artifact to achieve its objective now and in the future.
- Usability: If the user (customer) is able to intuitively or through prior experience understand the final artifact.
- Fit with organization: Relates to if the artifact is suitable to the way the organization works.
- Economical aspects: The business model of the artifact and how and why it is able to generate revenue and/or value.

The values and their respective categorizations can be found in table 3.4.

Table 3.4: Value aspects and its perspectives

Value	Perspective	Subcategory
Functionality[1]	Internal business	Production value: Physical value wrt. Quality: product architecture value
Completeness[2]	Internal business	Production value: Market requirements value
Performance[2]	Customer	Perceived value: Pragmatic value
Reliability[1]	Customer	Perceived value: Intrinsic value
Usability[1]	Customer	Perceived value: Pragmatic value
Fit with organization[1]	Internal business	Production value: Physical value wrt. Quality
Economical aspects[1]	Financial	Shareholder value

[1] Value present in SVM [59].
[2] Value not present in SVM [59].

As shown in table 3.4, two of the value aspects are not present in the SVM. The values were still fitted into the SVM as there are perspectives and corresponding subcategories which, in our opinion, fit the values.

The interviews and the focus group will possibly point out additional values that are important to the project. These additional values will, together with the initial values, be discussed in section 5.1.1 in order to measure their relevance and benefit.

To provide feedback to the implementation (Hevner 3) and to evaluate the values, dynamic validation is used. More specifically a focus group is used for evaluation (Hevner 5), this also addresses **RQ3**. The artifact is deemed complete if it satisfies both the initial but also the subsequent constraints and requirements. As the proof of concept uses the artifact, it is also used to validate the prototype. To identify how well the artifact behaves in relation to the aspects previously listed, a Likert scale is used when conducting the focus group (Hevner 5). The scale has five levels, ranging from 'strongly disagree' to 'neutral' to 'strongly agree'. With the focus group, we gain insight into the pros and cons of using the prototype versus Axis' Amazon AWS solution.

Whenever the extended requirements and constraints as defined after conducting all five interviews are satisfied, we can consider the artifact to be complete (Hevner 3) [55]. The results are then clearly presented, not only with regard to technical aspects but also with respect to business aspects (Hevner 7), in which it is determined whether it is worth continuing with our project or not.

3.5 Validity Threats

According to Runeson and Höst, there are four classes of validity threats. This includes construct validity, external validity, internal validity and reliability [79].

3.5.1 Construct Validity

Construct validity is a validity aspect that for instance refers to the importance of making questions that cannot be interpreted differently than how the researcher intended it to be interpreted [79].

Selecting the interviewees
The interviewees were selected through triangulation which is to take data from multiple sources in order to achieve a broader picture of the studied object [79]. We made sure that we had interviewees with different titles within the organization to make sure that we got different views from both technical and business aspects.

Design of interviews
The initial set of interview questions was sent to our manager, our supervisors and our examiner for feedback. When we felt that a question was not clear enough, the question was rephrased and updated for the next interview. To avoid that the interviewees would prepare answers that they thought we were looking for, we only sent them a general email about our project and the purpose of the interview instead of giving them the complete set of interview questions. To make sure that the interviewees spoke freely, we assured them that they would remain anonymous. Every interview was recorded in order for us to completely focus on the interview at the time of the interview and then analyze the responses at a later time.

3.5.2 External Validity

External validity refers to the possibility to generalize findings [79]. The results from our study may be applied to companies in similar situations to Axis, specifically companies wanting innovation tools used for prototyping. The findings from the interviews and from the focus group can be used to identify the potentials, but also the drawbacks of having an innovation tool.

3.5.3 Internal Validity

Internal validity deals with factors that changes the outcome of a study unbeknownst to the researcher. [79].

To be able to identify the drivers and the benefits of our thesis correctly we have had multiple interviews with different subjects before and during the implementation and then a focus group composed of multiple persons after the implementation of the prototype was finished. This allows for a multivariate analysis which is less dependent on ourselves. We have also sent our questions, summaries and thematic analysis to an independent third party in order to get constructive feedback.

3.5.4 Reliability

Reliability is an aspect of validity threats that deals with whether or not it is possible for other researchers to do the same study and obtain the same results [79].

In order to increase the reliability with respect to the results from the interview and the focus group, we both conducted the interview and the focus group. Then we made sure that one of us transcribed and summarized an interview or the recording from the focus group, after which the other one reviewed the findings, thereby validating the summary. A similar approach was taken when doing the thematic analysis.

Chapter 4
Implementation of an Artifact

This chapter includes each step in the implementation of both the prototype and the proof of concept.

The initial requirements and constraints for the implementation of the prototype can be found in section 3.4.2. A thorough analysis of what each of these requirements mean in terms of technical and business specifications can be seen in section 5.2.2.

4.1 Comparison

Currently, there are a number of PaaS solutions. An incomplete list [25] already has 71 entries. That is too many to make a complete comparison of all available solutions. Therefore, we chose four different PaaS solutions for our comparison, making sure that they were different types of solutions to get a broad range of solutions. The solutions chosen were AWS Elastic Beanstalk [2], Dokku [21], Heroku [35] and Red Hat OpenShift [69], based on the criteria below.

The most important reason for selecting these PaaS solutions in our initial selection is that they either are (1) stand-alone PaaS solutions not bound to any IaaS (Dokku, Heroku and OpenShift) or (2) they are bound to Amazon AWS (Elastic Beanstalk). For each specific platform, there was a different reason for choosing it. We chose Amazon Elastic Beanstalk due to the integration with Amazon AWS which Axis already uses. The reason for choosing Dokku was that it is an open source, make it yourself, alternative. Heroku was chosen because it was suggested by Axis. When it comes to OpenShift, it was selected due to it being a popular open source platform.

4.1.1 Selection Criteria

The PaaS platform has many requirements as to what it should enable. An initial set of requirements posed on the PaaS platform can be seen in section 3.4.3. The criteria

Ease to integrate, Security, Portability, Capabilities, Costs and Users on a per application basis, arose from discussions about the goal document and while reading literature about platform as a service [18, 32, 56, 76].

Enabling faster time to market

Faster time to market is enabled by all PaaS platforms due to the nature of how PaaS platforms work. As the underlying layer of the cloud infrastructure is controlled by the provider [56], the developer can focus on building and deploying the applications.

Decrease overall complexity

The complexity for the user of a PaaS platform is highly dependent on how the cloud deployment is implemented. According to the Practical Guide to Platform-as-a-Service written by the Cloud Standards Customer Council (CSCC), a PaaS solution can either have public, private or hybrid cloud deployment. In public cloud deployment, resources of a cloud service are used by several customers. If a customer instead can use the service exclusively, it is private cloud deployment. This is divided into either a solution on-premise or at a data center which makes sure that resources are isolated from each other. Hybrid cloud deployment is a combination of these [18].

Public cloud deployment reduces complexity the most. There is also virtual private which is public but it enables separate networks and more control for the price of complexity. The primary easy alternative with private cloud deployment is when it is used in conjunction with an IaaS as Amazon AWS, the most complex solution is to use private cloud deployment in house. The public solutions require the least complexity and are therefore preferred.

- AWS Elastic Beanstalk: public or virtual private (using AWS Elastic Beanstalk with Amazon Virtual Private Cloud) [7].
- Dokku: private cloud deployment [21].
- Heroku: public or virtual private cloud deployment [58].
- OpenShift: public cloud deployment (PaaS cloud service) or private (PaaS software) [69].

Ensure ease of extendability

The original criterion on extendability was how well developers could create add-ons, that is, applications which extend the functionality already present in applications. This turned out to be incorrect, as there is no need for other developers to build add-ons. Instead, it is preferable if an add-on is developed and other developers may create applications which uses the add-on for its base functionality.

- AWS Elastic Beanstalk: does not include add-on functionality.
- Dokku: does not include add-on functionality.
- Heroku: has add-on functionality [39].
- OpenShift: has add-on functionality [70].

Create a prototype which is open-source

Open-source prototypes can be created with all PaaS platforms. The add-on that may be created should perhaps be closed source as with certain platforms (such as Heroku) you can charge for the add-on within the PaaS platform. It is however important that the proof of concept application that uses the add-on should be open source as to provide an example of how to use the add-on but also in order to allow users to share their implementations.

Ease of receiving payment through add-on

For receiving payment, Heroku is the only alternative according to Axis' general business plan as discussed in section 3.1, that is, using third parties to sell.

- AWS Elastic Beanstalk: does not have add-ons.
- Dokku: does not have add-ons.
- Heroku: has add-ons in which you can specify a specific price [41]. Heroku pays the creator 70% of the revenue it receives from subscribers of the add-on [49].
- OpenShift: requires that you have a separate payment solution in which you identify paying customers. In OpenShift you can then limit the add-on users to only include those subscribed to your payment solution, an example is given here [72].

Ease to integrate with the current system

The PaaS platforms are all easy to integrate with the current system, but for different reasons:

- AWS Elastic Beanstalk: Easy to integrate because Axis already uses Amazon AWS.
- Dokku: Easy to integrate because Axis already has a required server (Amazon AWS).
- Heroku: It is very easy to sign up and start using it, with the benefit of not having to manage it locally.
- OpenShift: Same as Heroku.

Security

In light of recent revelations of the Meltdown and Spectre bugs, the security threats posed by these bugs are severe and threatens the mere existence of cloud services with virtual machines [30]. Therefore, a thorough rundown of the PaaS' reaction to the bugs have been included.

- AWS Elastic Beanstalk: kernel has been updated and all instances of Amazon EC2 are protected [8].
- Dokku: runs privately and therefore fixes are dependent on the underlying infrastructure.
- Heroku: released a statement of its fixes to the bugs, the first is that Heroku uses AWS which has fixed hardware vulnerability and the second is that Heroku has applied a kernel patch to fix host OS vulnerability [26].
- OpenShift: all OpenShift v3 Online and Dedicated systems have been updated [75].

Portability

Is it easy to move to a different PaaS solution if for instance the terms in the license agreement were to change or is there a risk for vendor lock-in? One solution to prevent vendor lock-in when using a PaaS is to use Docker to be able to move to a different PaaS easier. All four alternatives support Docker [4, 23, 77, 81].

Capabilities

This criterion includes supported languages and support for developer tools. More supported languages gives a higher score.

- AWS Elastic Beanstalk (7): .NET, Go, Java, Node.js, PHP, Python and Ruby [5].
- Dokku (8): As it uses Heroku's buildpacks by default, the same languages are supported [22].
- Heroku (8): Clojure, Go, Java, Node.js, PHP, Python, Ruby and Scala with other languages available through buildpacks [43].
- OpenShift (6-7): Java, Node.js, Perl, PHP, Python, Ruby, (.NET) [74].

Costs

The costs criterion include the cost for the different PaaS solutions and how these costs scale.

- AWS Elastic Beanstalk: no additional charges, pay for AWS resources used. You can create up to 75 applications and 1,000 application versions [6].
- Dokku is free but requires your own server. Comparable to AWS Elastic Beanstalk as you could run it on Amazon AWS and then pay for the AWS resources [21].
- Heroku: Price based on time running each Dyno (Heroku's application container). One Dyno always on is billed at $7/month for the hobby price plan and $25/month for the standard plan. Included RAM is 512 MiB [38].
- OpenShift: Pro includes possibility for 10 projects and 2 GiB RAM by default for $50/month [73].

Allows specific users on a per-application basis

This criterion investigates whether the PaaS service allows for more granular control over the users of an application.

- AWS Elastic Beanstalk: Possible to create custom policies allowing/denying access for specific users and groups [3].
- Dokku: Not supported [24].
- Heroku: Supports it [46].
- OpenShift: Based on the information on their site it seems not to be supported (at least not trivially) [71].

4.1.2 Selection Overview

Table 4.1 shows an overview of the criteria and the results. Here, the categorization is as follows:

- +: the best alternative(s). 1 point is added to the total score.
- -: the worst alternative. 1 point is deducted from the total score.
- 0: neither the best nor the worst. The total score is unchanged.

Looking at these results, Heroku seems to be the best alternative. Therefore, we will choose Heroku as our PaaS solution.

Table 4.1: Comparison

	Amazon Elastic Beanstalk	**Dokku**	**Heroku**	**OpenShift**
Faster time to market	**+**	**+**	**+**	**+**
Decrease complexity	**+**	**-**	**+**	**+**
Extendability	**-**	**-**	**+**	**+**
Open Innovation	**+**	**+**	**+**	**+**
Payment	**-**	**-**	**+**	**0**
Ease to integrate	**+**	**+**	**+**	**+**
Security	**+**	**+**	**+**	**+**
Portability	**+**	**+**	**+**	**+**
Capabilities	**0**	**+**	**+**	**-**
Costs	**+**	**+**	**-**	**0**
Per-application users	**0**	**-**	**+**	**-**
Total	**5**	**3**	**9**	**5**

4.2 Implementation of the Prototype

This section includes the steps taken while implementing the prototype, including the selection of the programming language, a discussion about the portability and a discussion about what needs to be done when setting up the system.

4.2.1 Initial Architectural Overview

When producing the goal document of the thesis, an initial model was sketched to illustrate the overall architecture of the system, as can be seen in figure4.1. The idea is that the prototype which we build, in the figure marked as 'PaaS' communicates directly with Axis Connect (Connect in the figure) through the 'PaaS agent', this agent provides authentication and allows the use of APIs to access functionality on the camera(s).

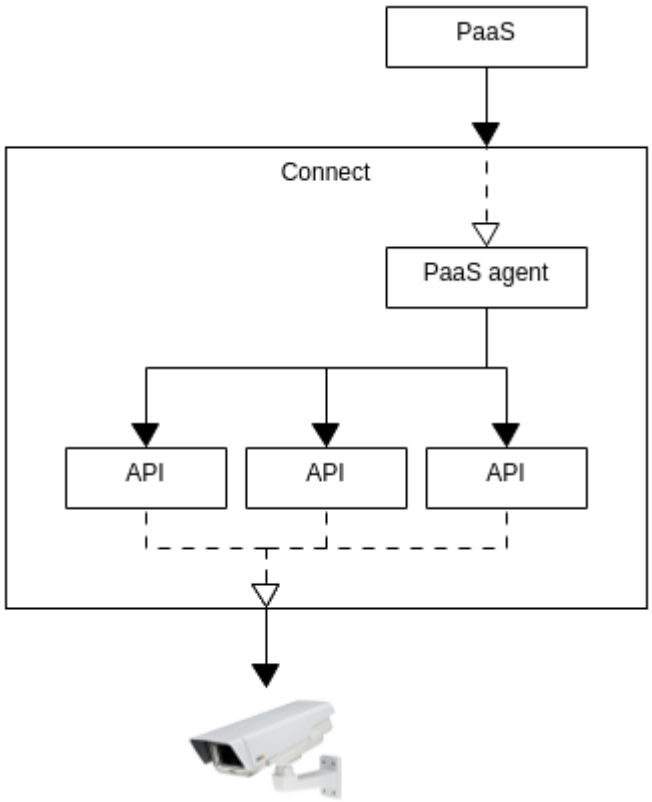


Figure 4.1: Initial architectural overview

4.2.2 Language Selection

The preferred language for the prototype which is a Heroku add-on was chosen to be Go. The reason for this is that Go is one of the main languages that Axis uses for cloud development. Go is also one of the languages officially supported by Heroku [43]. Both of us had experience with a number of programming languages but no previous experience with Go so in order to gain some experience and get an understanding for the potentials of Go, we went through a few tutorials. Among other things, we tried to implement a simple HTML form to get a username and a password. We also tried redirecting between HTML sites. Next, a small 'Hello world'-program was implemented for testing purposes.

4.2.3 Portability

One problem that PaaS solutions have is the issue of vendor lock-in. Simply deploying a Go application to Heroku will result in a dependency of Heroku's operating system image, known as Stack [52]. To avoid vendor lock-in, applications can be deployed within a Docker image. This allows the specification of the needed packages inside the Docker image and it is guaranteed to run the same, independent of the host system. To try this, we created a Docker image that included the simple 'Hello World'-program implemented earlier. To do this, a Dockerfile is required. After signing up and installing Heroku, we successfully deployed everything to Heroku.

A problem with this solution was that the size of the Docker image was around 700 MB even though the program itself was small. This was due to that, in order for it to work, the complete Go language needed to be included in the Docker image. Heroku does not have a limitation on the size of a Docker image [44]. However, it does put limitations on the boot time of the Dyno (Heroku's application container) [40]. The solution to the large image file was found in a blog post [28]. The Go application is in this case compiled first and then only the binary is included in the Docker image without the need to include the complete Go language. When using this solution, the size of the Docker image was reduced to about 7 MB.

In our efforts to avoid vendor lock-in, we instead introduced another dependency. The compilation of the application is now dependent on the local development environment with a specific version of Go and specific Go packages. Still, this approach was chosen as the most preferable due to the reduced size and its independence from Heroku's stack.

4.2.4 Setting up the System

Building an add-on skeleton

The next step in our implementation process was to start building an add-on that could actually be used by our proof of concept application. In order to do this, we first needed to register as an add-on partner. Then we needed to install Kensa, a tool that helps with the integration of an add-on to Heroku [50]. Following the steps in [42], we ran:

```
$ kensa init
```

to create a skeleton for a manifest, that is, a document describing the interface between Heroku and our add-on. We then changed the manifest to include the correct values for the required parameters.

When this was done, we needed to implement different endpoints that handle Add-on Partner API requests. For instance, Heroku requires an endpoint that takes care of when a user wants to start using our add-on. Another endpoint that is required is an endpoint for when a user stops using the add-on or when the user wants to change their add-on plan, the difference being that it is either a DELETE or a PUT request that is being sent to that particular endpoint [42]. We also needed to implement a basic authentication.

To be able to test our implementation locally, Kensa included the command:

```
$ kensa test
```

After a process of iteratively testing and implementing, we deployed the add-on to Heroku again and modified the base URL of the manifest to point to the correct URL. Finally, the manifest was pushed to Heroku using:

```
$ kensa push
```

Connecting to Connect

In order to get some understanding on how to connect to Connect, we had a meeting with a person from the Connect team. We then learned that, to be able to use the APIs that Axis Connect provides, we need to get an ID token that should then be sent to Connect.

Connect should then in turn return a session ID that can be used to do things in Connect. We were given some options to explore on how to get the necessary token while our contact at Connect was also doing some investigations on his own. When he got back to us, we were told that it would be best if we could integrate against Axis' own OpenID Connect (OIDC) provider. For this, we needed a client ID and client secret from another team at Axis. At first, we received these for the stage environment, but it turned out that it was not possible to access the stage environment externally from Heroku. Therefore, we later received a client ID and client secret for the production environment instead. Now we managed to receive the ID token that we needed to send to an endpoint in Connect. The endpoint does not exist at this point but with the endpoint in place, we could send the token to Connect and receive the session ID back. Figure 4.2 shows the authentication sequence required to make API calls from our add-on to Axis Connect.

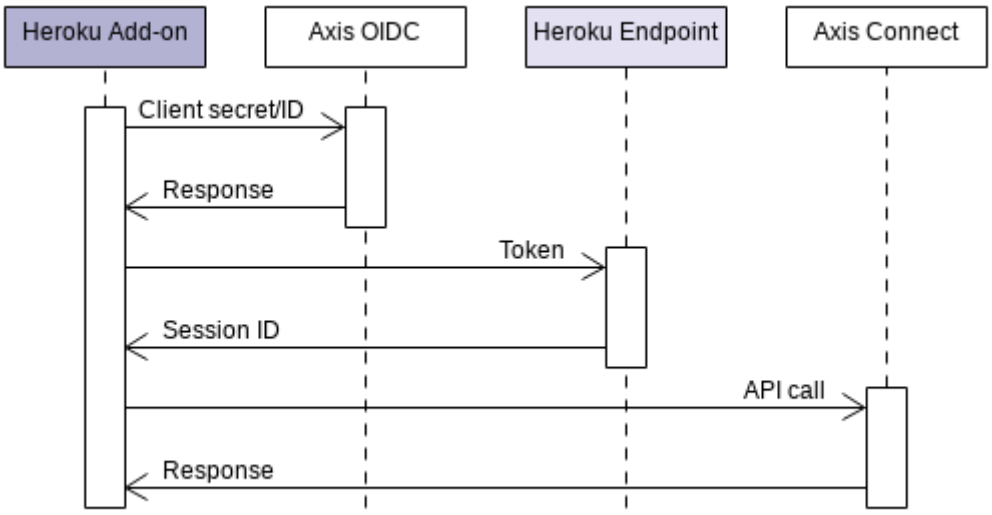


Figure 4.2: Sequence diagram

Our contact at Connect would make sure that the endpoint we needed was to be implemented. To be able to continue in the meantime, we were told how to retrieve a session ID manually. The temporary solution was then to only allow users who have been authenticated through Axis OIDC to access our API. This was achieved by checking whether the login had been successful when a redirect from OIDC came back to our add-on, and in that case put an HTTP cookie in the user's browser. The cookie then enables us to authenticate users for future calls. This is not ideal since it will always be our cameras that will show up, no matter who logs in.

Connected to Connect

Now that we had a way to get a session, we had a look at the available API calls by checking the documentation and took notes on what API calls might be interesting to implement. We successfully implemented two calls, but since we did not have any cameras to test things on, we only received empty responses back. We were then given access to 15 cameras that

had already been setup and connected to Connect which allowed us to start making API calls.

Add-on API implementation

When we could confirm that we managed to successfully connect to Connect, we took some time to reason about how our add-on and Connect should interact with each other. We came up with three different alternatives. The first alternative was to implement each API call in our add-on, simply doing a direct translation of Connect's APIs.

A second alternative was to implement a handler in our add-on which takes a POST request. This request should then include the type of HTTP request to be sent (GET, DELETE, POST) and then include the URL for the request. The major benefit of this was that we did not have to implement each API call since it is possible that these will change.

A third option was to just implement that all API calls happen on /connect/url. The URL is then extracted and a new request is sent to Connect with the same type of HTTP request. In this scenario we only needed to implement a handler for each type of HTTP request. In the end this was the solution that we chose because it allows the use of current but also future API requests. The limitation presented by this solution is that Connect uses different types of content in its response but also when used as a parameter for sending POST requests. Most calls uses JSON but a few uses plain text and zip. As of right now we accept and send in all content types (JSON, plain text and zip). A diagram that illustrates what happens when a Heroku application makes an API call to the add-on can be seen in Figure 4.3.

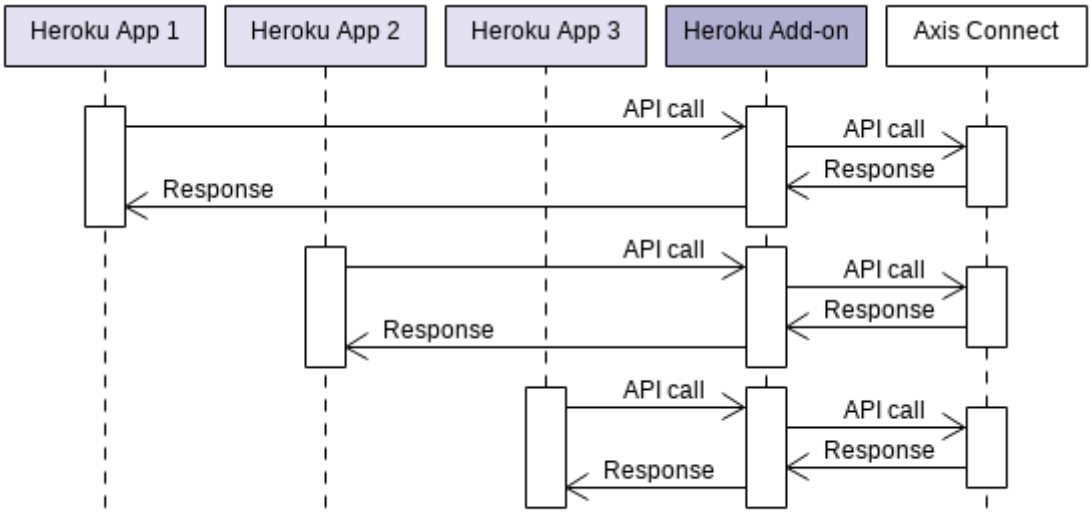


Figure 4.3: API calls diagram

4.2.5 Prototype Implementation

Upgrade to Hobby plan

At some point, the prototype was upgraded from Heroku's Free plan to Heroku's Hobby plan. There are clear benefits with the Hobby plan over the Free plan, such as that the Hobby plan's Dynos (see section 3.2) are always active while the Free plan's Dynos go to sleep after 30 minutes of inactivity [38].

HTTPS

In order to make our prototype more secure, we had to make sure that our add-on used HTTPS, which makes communication private through the use of encryption. The default Heroku website accepts both HTTP and HTTPS. To ensure that both the intro page of our add-on but also the API calls were made through HTTPS and not HTTP we permanently redirected all HTTP requests to their HTTPS counterpart.

Axis user authentication

We implemented a login page on the add-on on `/oidc` which redirects to Axis OIDC and then back to our add-on. As mentioned previously, we then utilized cookies to identify which user has been authenticated. Whenever someone that was not logged in (does not have the cookie) tries to make an API call on the add-on, they would be redirected to Axis OIDC's login page. After a login has been successful, the user gets redirected to the add-on. In order to then redirect the user correctly back to the original API call we used the state parameter of OIDC [68], which allows us to pass the origin as a parameter.

Intro page

An intro page written in HTML for the add-on was created which is displayed when the base url[1] is accessed.

Authentication of applications

There is a need to authenticate which application is making calls to the add-on. The need first and foremost stems from a security aspect, to be able to ensure that only applications which have installed the add-on through Heroku are able to make calls to the add-on. To be able to determine how much of the load a certain application uses is also an attractive aspect of the authentication.

When an add-on is installed through Heroku, a provisioning request in the form of a POST request is sent to the url `/heroku/resources`. Our handler then sets the Heroku environmental variable (ENV) `AXISWDT_KEY` to a long random string. This `AXISWDT_KEY` will then be sent by the application to the add-on when making calls.

In order to store the generated `AXISWDT_KEY` we installed the Heroku add-on Heroku Postgres [51]. The add-on allows us to connect to a database through a URL and create

[1] `https://axiswdt.herokuapp.com`

tables. We created a table `USERS` where we store the `AXISWDT_KEY` generated by us and a `UUID` that Heroku sends with the provisioning requests which is unique for each application.

To authenticate an application on the add-on we then extract the `AXISWDT_KEY` sent by the application and assert that it is present in the database.

Chaining API calls

One way to create value with our add-on is to create new API calls not present in Connect by chaining calls that do exist. Unfortunately it is currently not possible to get video streams using Connect. We then decided to make use of the fact that a snapshot is created every time that a camera is connected to Connect. These snapshots could be shown instead of a video to give an idea of how it would look like. Our idea was to have a button which would refresh the snapshot of all cameras, to get an authentic feel. This was implemented by using three API calls to Connect: *get all devices in a folder*, *remove a device* and *add a device*. After getting all devices in a folder, we looped through them and removed them after which we added all devices again. This functionality was implemented under the AxisWDT API on the add-on as an API call to `api/refreshcameras/{folderid}`. One problem with this API call is that when a camera gets added to Connect it sometimes takes a very long time to receive a snapshot. To be able to see the snapshots on the proof of concept the user has to be authenticated.

VAPIX calls

Through the use of the VAPIX part on Connect, one could implement video stream. This requires some work however as it is not possible through the regular Connect API. Due to time constraints, we did not have time to implement this functionality.

Updated architectural overview

The implementation of the prototype added many new aspects not included in the initial architecture. Discussions with the people from Connect led to an improved view of how the authentication should be implemented and the authentication of users created the need to store user related information in a database. An updated architectural overview can be seen in Figure 4.4. As shown, the Heroku add-on has a database that stores user data. The figure also includes the authentication through Axis OIDC and the interaction with Axis Connect. The Heroku apps have to install the Heroku add-on to interact with the prototype.

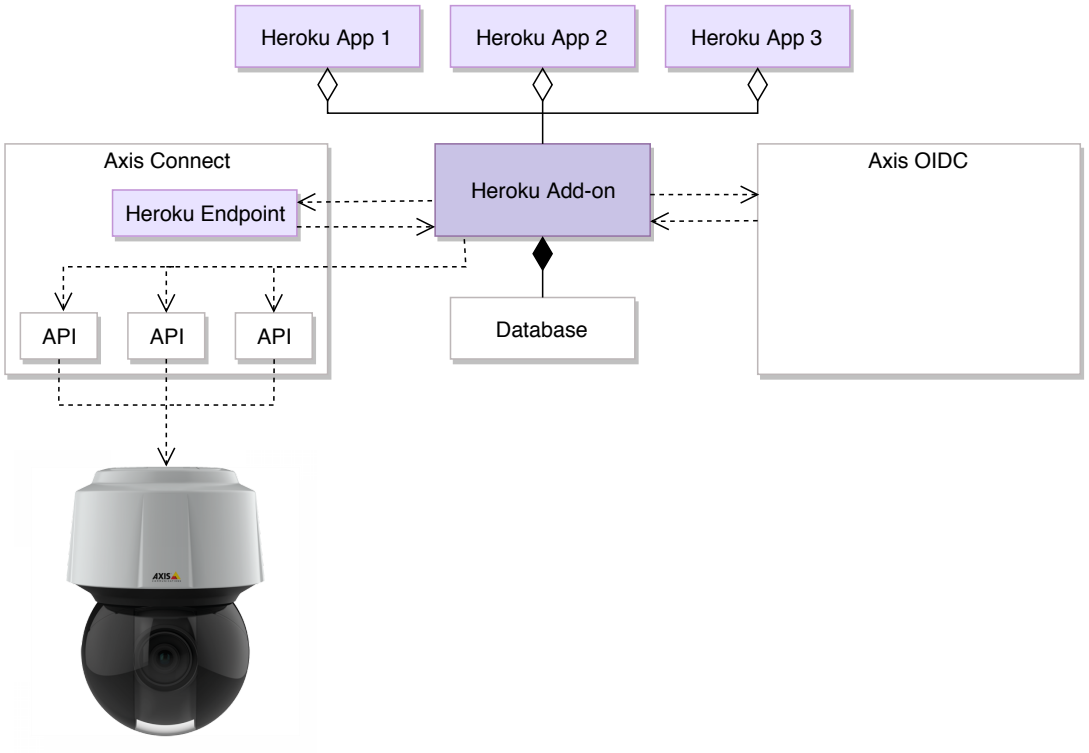


Figure 4.4: Updated architectural overview

4.3 The Proof of Concept

To illustrate how the prototype could be used, we created a proof of concept. This section includes the steps taken when implementing the proof of concept.

4.3.1 Language Selection

To be able to test whether applications can still be implemented in other languages, we decided to implement a proof of concept in JavaScript. Initially we had created a simple proof of concept in Go to test what happened when our add-on was installed on another application on Heroku. The Go proof of concept was kept as is and tested to ensure we did not break anything in our efforts to make things work in our proof of concept written in JavaScript.

There are several ways to implement web pages in JavaScript but one of the most popular libraries to use is React [34]. There are other options available, such as Vue.js, but after reading several comparisons [65, 31] and considering its popularity, React was chosen.

In terms of JavaScript, Heroku only supports the Node run-time environment. To be able to make a React application on Heroku, a specific BuildPack was needed [36]. To create the React application we ran the following commands:

```
$ create-react-app jswdt

$ cd jswdt && git init
```

```
$ heroku create jswdt --buildpack https://github.com/mars/
    create-react-app-buildpack.git
```

4.3.2 Initial Implementation

The initial implementation of the proof of concept was mostly getting used to the combination of JavaScript, React, HTML and CSS. A difficulty that did occur had to do with Cross-Origin Resource Sharing (CORS), because Cross-Origin HTTP requests are limited by the web browser when started by scripts. CORS allows a user agent to access a different server on a different domain than what the site is using [19]. This is needed as we send API calls from our proof of concept to our add-on which is on a different domain. To ensure CORS is still enabled while allowing API calls from any origin, the HTTP header `Access-Control-Allow-Origin` with the value `request.Header.Get("Origin")` had to be added to our add-on to allow any origin to make API calls.

A difficulty that presented itself when writing the initial Go proof of concept was displaying JSON images, this was easily implemented in JavaScript with just a few lines of code.

4.3.3 Theme of the Proof of Concept

Now that we had a basis for a proof of concept, we wanted to do something more elaborate. We developed personas with user stories to start getting some ideas of what might be implemented with the use of our add-on and why our add-on could be useful.

After some brainstorming about applications that we might be able to implement, we came up with 'Axis Grocery Store'. The idea was to include several features that Connect offered and show the possibilities that our add-on offers. This would be our demo, in which we could show how our product can be combined with other products, in our case a revenue counter and information about the next scheduled delivery to the store. The demo included the possibility to check that the cameras had the latest firmware and that the cameras were connected. We also included snapshots from a number of cameras and the possibility to update the snapshots.

The application was created with an Axis logo and Axis for instance in the title of the page but after discussions with our manager, we made sure that there was no mention of Axis anywhere on the page unless a user is logged in.

Chapter 5
Results and Analysis

In this chapter, we present the results and analysis of project outcomes from the three main parts of the project, that is, the interviews, the prototype and the focus group.

5.1 Qualitative Results and Analysis

We conducted five interviews to explore values and clarify requirements of the innovation platform. Next, we analyzed the interviews qualitatively using a thematic analysis. This section includes the results and an analysis of the results from the interviews.

5.1.1 Interviews: Thematic Analysis

Table 5.1: Themes found during the thematic analysis

Theme	Definition	Research Question
Benefits/values	The outcomes that Axis imagines are possible. These might be considered as requirements on the system	RQ2, RQ3
Business	Business aspects that Axis considers to be relevant	RQ2, RQ3
Challenges/ limitations	The obstacles that Axis sees with this project. These might be considered as constraints on the system	RQ2, RQ3
Drivers	Reasons for Axis to invest in this project	RQ1
Level of openness	The degree to which Axis is willing to contribute to open source	RQ3

Benefits/values

When it comes to the outcomes that Axis imagines are possible, the interviewees imagine a short release cycle, fast prototyping and a way to make it easy to build services. Our platform is seen as an innovation platform where you can, as interviewee C puts it, *"quickly experiment with something"* and get features out to the users in a faster way. In interviewee B's words, it is a way to *"get it out, get some experience, refactor and continuously improve"*. It should be simple to add new functionality to Axis' products. A small start-up should be able to connect to Axis' cameras and Axis' system and build basic solutions and quickly and easily obtain customer value without putting too much effort into it (D). A PaaS reduces the burden on the developer when it comes to setting up, for instance, virtual machines (B) and gets rid of things that are hard to do (D), making time to market quicker (B).

One of the benefits that several interviewees mention is the payment model. Interviewee C points out that if Axis starts to charge for software themselves it would be a huge challenge. Using Heroku, you can still sell using third parties. However, there are some disagreements on how this payment model would be realized.

Another benefit or value that Axis imagines is possible is to enable backwards compatibility using cloud solutions. *"When building products it can happen that you have to break your old APIs and then the cloud service can be a way to still have support for older products. [...] if it works on this product, you can be sure that this functionality works on all products"*.

One of the interviewees also sees the platform as a manual on how things can be done and our project as a way to learn more about Connect (D). Interviewee A mentioned that if our solution *"simplifies the credentials aspects, maybe it's a good idea"* as a response to giving reasons for why Amazon AWS could be replaced or complemented.

In section 3.4.3 we mention the values important for this project, we will analyze the responses from the interviews in accordance to these values.

- Functionality: The functionality of Heroku, especially the deployment process was seen as positive, interviewee B put it: *"One is to get features out to the users in a faster way. Which means that we should be able to develop things much faster..."* To offer functionality that was not present in Axis Camera Companion (ACC) was mentioned as something important as well.
- Completeness: Not mentioned in the interviews.
- Performance: Not mentioned in the interviews.
- Reliability: Several interviewees state the need for the prototype to be reliable, interviewee C said: *"It would need to work seamlessly it would need to be... I mean no hassle, it would need to install itself."*
- Usability: Many interviewees were positive to the usability factor of Heroku and how simple it was, interviewee A stated: *"If it is cheaper and easy to develop it would be worth it."*
- Fit with organization: Many interviewees were positive to the prototype being a part of Axis' portfolio, interviewee C stated: *"[...] I mean if it's something that could easily be utilized by many to try things. I mean building prototypes or proof of concepts first and then you can say: Hey we have something, let's do it in a proper way that scales and things and absolutely, it could be an innovation tool I guess."*
- Economical aspects: Many things were stated about the business side of the prototype, interviewee C put it as *"[...] low cost for Axis, high value for our partners, if you could demonstrate that and demonstrate a pull effect from our partners then that would probably... that is probably what is needed to sell to Axis that this is something we should continue using."*

No additional values from the SVM were found during the interviews.

Business

Regarding business aspects that Axis considers to be relevant, almost all interviewees mention that we have to make sure that we have something to offer that, for instance, Axis Camera Companion (ACC) does not offer. According to interviewee E, the add-on needs more functionality to be viable. Multiple interviewees also mention that Axis is moving towards a solution based service company (B, C, E).

Interviewee D states that Axis does not have many partners *"developing things for Axis' cloud services, this could be a way to reach small start-ups or hobby projects that will build things on Axis' products"*. Later, interviewee D compares our platform with the App-store/Google Play community and envisions a similar community for our platform. As interviewee D puts it, *"it's no problem that there are a number of other [applications] in App Store that do the same thing. As long as we get the best one. [...]. One makes it possible to build many services that do similar things and in the end you have the key to satisfying customer needs in the best way"*.

Interviewee E thinks that it is positive that you can get payment through the add-on, but it conflicts with Connect if Connect begins charging for services. However, the solution, as we presented it, could not be sold to a product owner according to the interviewee.

Analyzing the responses from the interviewees, they were mostly positive to the business aspects of the project. The possibility of receiving payment through Heroku but also the idea of having the prototype be an innovation platform was seen as important business aspects by the interviewees. Specifically increasing the speed of innovation was an important business factor as mentioned by interviewee C: *"If you are successful you could potentially be something that increases speed of innovation for others, potentially, I think that every entrepreneur in the world knows that means money"*. This is a very interesting aspect of the prototype, as it can allow developers to try out new ideas quickly due to the speed. Bosch mentions the 'New-product transition interface' in his paper *"Achieving Simplicity with the Three-Layer Product Model"*. The interface allows products to move from the innovation layer into the differentiation layer [11]. If a product is built on our innovation platform it can then, if it has been proved to be successful and it has been tried and tested, be moved into the differentiation layer.

Challenges/Limitations

This theme reasons about the obstacles that Axis sees with our project. One challenge that interviewee C mentions is that Axis is a *"company with very limited experience with cloud to begin with"*. Later, interviewee C states that another challenge is *"[...] actually learning how to do things in a correct way, in a cyber secure way"*.

Several interviewees mention that it takes some time to set up Amazon AWS, but once it is there, it is quite easy. According to interviewee B, it is indeed the case that you have to think about a lot from an architectural point of view with an IaaS that you do not have to think about with a PaaS, but a lot of documentation and best practices exist. Since Axis has already invested time and money in developing tools for AWS and everything is already set up, interviewee E argues that our project is not the way to go.

Something that might limit the innovation platform is the fact that it is very dependent on Connect prioritizing our issues and that Connect has support for everything that we want to do.

Risks mentioned during the interviews are that other projects might be delayed if a decision is made to continue with our project (A). Also, the costs for Heroku could be considerably larger than expected (D). Another risk that is mentioned is that we might risk opening a backdoor into Connect if not everything is done well (D). Furthermore, interviewee D states that when abstracting and choosing for someone else, you also take away possibilities

The challenge or limitation that was most frequently mentioned was the security risks introduced by Heroku, as Axis has not worked with the platform previously. A thorough analysis of different aspects of the security of our platform can be found in section 5.2.1.

Drivers

Reasons for Axis to invest in this project are all related to business. For instance, interviewee C states that *"large customers or large customer sections can have issues with using Amazon [since they] view Amazon as a competitor"*. Interviewee C later states that a reason to investigate moving from Amazon AWS would mainly be from a lock-in perspective, *"we would not want to be totally reliant on one supplier, especially another supplier as*

big and fast moving as one of the giants within cloud". One reason that interviewee E mentions as a driver is that the issue with Amazon AWS is the cost.

One reason for not continuing with the project that interviewee E mentions is that Axis already has invested time and money in writing their own tools for Amazon AWS.

The business aspect is the most important driver for Axis, as mentioned previously the possibility of an innovation platform but also the monetary aspect of decreased costs. A paper by Rose and Furneaux has identified many different drivers for spurring software innovation. There are specifically three drivers that resonate with our project and why Axis is interested in it, those are:

Innovation leadership. Leaders can be spearheading innovation and are responsible for creating a work environment that stimulates innovation [78]. Two of the interviewed candidates were managers and this could be applied to their reasoning as to why this product is needed. Interviewee C stated that: *"I think absolutely if we have specific use cases if we can enable speed of innovation that could absolutely be a use case"*. Interviewee D stated that: *"[...] build simple solutions and quickly and easily obtain customer value without putting too much effort into it"*.

Innovation evaluation. The possibility to evaluate software is an important driver for innovation. Our platform could be used to develop prototypes and understand their feasibility. Many interviewees were positive to the idea of the platform being used for prototyping and for evaluating ideas. One interviewee put it *"get it out, get some experience, refactor and continuously improve"*.

Innovation tools and techniques. Tools and techniques can be designed to enable creative work, and specialized toolkits have been created to aid users in the innovation process. This is spot on as one of the purposed uses for our platform is as an innovation platform made to enable prototyping and creative ideas. Interviewee C put it *"we're building a lot of proof of concepts, we're running a lot of demos and we're throwing things away and doing something else"*, this further validates the value of an innovation tool and that there are areas in which it could be applied.

Level of openness

When it comes to open sourcing, there are some differences in opinions. One of the interviewees stated that the project should not be open sourced at all (E). Other interviewees were positive towards open sourcing as much as possible (B, C) or positive to only open sourcing the example applications (A, D). This depends on whether or not we want to charge for our prototype or not. Interviewee D was of the opinion that the applications should be open sourced, giving contributors the possibility to see what can be done. Contributors can then simply download the demo application and add functionality that they want to have and remove functionality that they don't want to have. *"All of a sudden you have your own application and you haven't spent that many hours getting it working. You don't need to understand everything. Copy-paste programming"*.

When asked why people would contribute if the project was open sourced, interviewee C states that in his experience, if people use an open source project, they will contribute to it. Interviewee B remarks that applications can be seen as advanced documentation and that people get annoyed if they are not up-to-date. He also thinks that if partners feel that there is something "lacking", they might contribute. But according to the interviewee, the

question is then if there is really a need for the add-on to be open source in that case or if it is better to keep everything within the partner network.

An analysis of the results from the interviews gave very mixed results when it came to the idea of having some parts of the project open source. One stated that nothing should be open source while another said it should be open sourced as much as possible. A paper by Munir et al. discusses that the largest concern with open source among companies was the idea of giving away intellectual property rights that could be of competitive advantage [62]. This concern could be a factor for the skepticism some interviewees feel towards open sourcing the prototype. Interviewee C stated: *"But if you intend to provide your platform as an innovation tool that you would like to charge money for maybe we shouldn't open source it..."*

5.1.2 Interviews: Updated Requirements

In section 3.4.2 a number of initial requirements and constraints posed on the prototype were defined. These can be found in table 5.2 for convenience. Analyzing the results from the interviews, a number of solutions on how to fulfill the requirements came up. These solutions are also included in the same table.

Table 5.2: Requirements on the PaaS platform and solutions found through the interviews

Requirement	Solutions
Easy to scale	Solution was not found from the interviews.
Faster time to market	PaaS enables this since there is no need to spend time thinking about the architecture, such as setting up your own virtual machines
Easy to use	Proof of concepts as (advanced) documentation
Easy to receive payment	Third party takes care of payment
Portable	Docker
Secure	Heroku and authentication through OIDC makes it more secure

After conducting the interviews, the following additional constraints and requirements were found:

- Enable backwards compatibility
- Reliable
- Offer new functionality
- Cost

These additional requirements are discussed in section 5.2.2.

5.2 Implementation Results and Analysis

Chapter 4 discusses the steps taken towards implementing the final prototype and proof of concept. Appendix C includes screenshots of the finalized proof of concept. The proto-

type's intro page can be found on URL `http://axiswdt.herokuapp.com/intro/` and the proof of concept is accessible on `https://jswdt.herokuapp.com/`.

5.2.1 Prototype: Security

Something that is very important to discuss regarding our prototype is security. There are several aspects in this project which requires special attention in terms of cyber security.

CORS

As mentioned in section 4.3 we made modifications to the add-on to not get any CORS errors on the proof of concept. These modifications circumvent CORS by allowing all origins. This creates the odd case that the add-on creates a security risk for the proof of concept. The risk here is that the add-on could have malicious code which would then be executed without the knowledge of the user of the proof of concept.

Authenticating applications

To authenticate applications we set the config variable `AXISWDT_KEY` on the application. This API key is visible to the user of the proof of concept as it is part of the HTTP requests, this poses a potential security risk of the user sharing the contents of the key.

Authenticating users

While the Heroku endpoint was being built we used a cookie to determine whether a user had been authenticated or not. The session id `sessionid` is hardcoded into the add-on so therefore, no matter which users log in to Axis OIDC, they still access our Connect account. However, when a Heroku endpoint is in place, as mentioned in section 4.2.4, this will no longer be the case.

Using OIDC's State to pass information

To be able to send the users back to the original site after a successful authentication we needed some way to pass information from the original request to the authentication and then back to the add-on. To achieve this there is a parameter in OIDC called state, which we set to the origin of the original request. This is not how the state parameter should be used, it is used for security reasons [68].

Cloud providers

As mentioned in the comparison in section 4.1, Heroku is primarily a public cloud provider. There are security threats present when dealing with cloud providers and Heroku is not immune to these threats. Gholami and Laure discuss the security issues present in cloud computing.

The first issue is about how, in the cloud, physical and virtual resources are shared between independent users and how this created a security risk as the attacker could be on the same physical machine as the target.

Loss of control is also a security threat, as everything is hosted at the cloud provider's premises. This created the potential of cloud providers acting maliciously by for instance mining its customers' data [29].

Trust is also a major aspect as the customer has no control over certain aspects of cloud security, they have to trust the provider in implementing the security measures needed [29]. There are many aspects to consider when choosing to work with a PaaS [80]. One benefit of using a public PaaS is that the security implementations which usually is an overhead cost is up to the provider, which often spends much more money to secure it correctly compared to if you managed the security yourself [10].

Heroku

There have previously been security related issues with Heroku. In 2012 a security vulnerability was revealed which allowed attackers to gain access to user accounts through the use of malicious HTTP requests. Heroku stores the users passwords in hashed form, so they were never at risk [61]. Heroku was told about the issue on 19 December 2012 and went public with the issue after a preliminary patch had been produced on 20 December 2012 [61].

5.2.2 Prototype: Clarifying the Requirements

The initial requirements for the add-on are described in section 3.4.2. Section 5.1.2 analyzes the response to the interviews and the solutions for the requirements. The interviews also added four additional requirements. All requirements and what they mean for the prototype have become better understood during the implementation. The following list discusses in greater detail the meaning of each of these requirements.

- Easy to scale: This is first and foremost a requirement posed upon the platform. That applications built can handle more clients and processing power in a scalable manner. This is achieved in Heroku through the ease of deploying more dynos as your application grows [47].
- Faster time to market: The setup of the application is the most important aspect when reducing the time to market.
- Easy to use: Ease of deploying the application, ease of extending and using its features.
- Easy to receive payment: This one is self-explanatory, if it is possible through the PaaS platform to receive payment and how easy it is.
- Portable: If it is possible to migrate the application from the PaaS platform to another platform. As mentioned in table 5.2, Docker helps mitigate this problem as a container can easily be moved.
- Secure: That the platform is inherently safe and secure to use. This is difficult to prove.
- Enable backwards compatibility: Ensuring that old functionality is not lost with future implementations but still enabled and in working condition.
- Reliable: That the platform works as expected and does not have bugs and/or unexpected behavior.

- Offer new functionality: That the platform offers functionality that is not present in another Axis product.
- Cost: That the add-on application is cost competitive with other alternatives.

Based on the clarification of these requirements we analyze the finished add-on and see how well it enables the requirements.

Table 5.3: Requirements fulfilled by the add-on implementation

Requirement	The add-on implementation	Fulfilled
Easy to scale	Enabled by Heroku	Yes
Faster time to market	Enabled by Heroku	Yes
Easy to use	Partially enabled by Heroku and our implementation	Partially
Easy to receive payment	Enabled by Heroku	Yes
Portable	The use of Docker to be independent from Heroku's stack	Yes
Secure	Measures taken to ensure security	Yes
Enable backwards compatibility	Enabled by Axis Connect	Yes
Reliable	Partially enabled by Heroku and our implementation	Partially
Offer new functionality	Several features, see section 5.2.3 and 5.2.4.	Yes
Cost	Cost of the add-on application can be covered by pricing plans imposed on the user.	Yes

As shown in table 5.3 above, many requirements of the add-on implementation are fulfilled. Through the selection of the right PaaS platform and the functionality of Axis Connect many requirements are fulfilled already. The aspects 'Easy to use' and 'Reliable' are marked as partially fulfilled and the explanation as to why is explained in the subsequent sections.

Easy to use

As mentioned in section 5.3.2 participants where positive to the ease of deployment to Heroku and agreed that the add-on could be used to develop prototypes which suggests the add-on implementation is easy to use. To mark the requirement as 'Fulfilled' however would require multiple test subjects to try out the add-on and analyze their response to it.

Reliable

Heroku itself is a very reliable platform, especially in its uptime. Looking at statistics from May 14, 2018, Heroku had an uptime of 99.999929 in the EU region in the past 60 days [53]. The focus group where also very positive to the idea that when an app is deployed to Heroku there is never any downtime. To mark the requirement as 'Fulfilled' would require actual users to test the add-on and for us to analyze the findings.

5.2.3 Alternative or Complement to AWS

Axis Connect is built on Amazon AWS, and to have Heroku replace AWS was viewed as impossible from at least one interviewee. It could possibly be used as an alternative to AWS when building applications that use Axis Connect.

Having our prototype as a complement to AWS seems the most fitting, as we can use the three layer product model [11] to fit Amazon AWS to the two upper layers and our prototype in the innovation layer.

As interviewee B puts it: *"If we get back to the thing with Heroku maybe as the innovation layer and then a higher layer that is more mature, then I don't see a problem with having a mature Amazon AWS-based implementation and then below on the innovation layer you have something in Heroku which allows you to quickly push out things but then when you know that this is something you want to bet on, then you can move it up and in Amazon AWS"*.

This quote is in line with how Bosch describes the innovation layer. Specifically, Bosch mentions the possibility of creating innovations not relevant for a specific product but suitable for deployment within a software system as a way of showing potentials of actual results [11].

5.2.4 Complement to Connect's API

One thing that can be discussed is the reason to use our add-on instead of Connect's API. As we have implemented the ability to take any API call that Connect has and redirect it to Connect, that functionality might validate the question whether our add-on is actually needed, why not simply use Connect's API? However, there is some functionality that we have implemented that Connect does not offer:

First is authentication through Axis OIDC which allows us to authenticate every user so they are able to make API calls.

Second is the possibility to join API calls to extend the functionality of Connect without the need to change Connect. This can enable the creation of new, more specific, calls which reduces the users need to connect multiple calls in order to get the specific response that they are looking for. The creation of additional API calls is especially exciting if the add-on would be open source, which has many benefits as mentioned in section 2.4. If the add-on is open source, it means that users can do pull requests with the functionality they may need which furthers the possibilities.

Third is the ability to join functionality already present in another Heroku add-on with the API calls. As an example, one idea is that there is a Heroku add-on called CameraTag [48]. This add-on can be combined with the video stream from Axis Connect to create new easy to use functionality.

5.2.5 Continuous Delivery

One of the major benefits of the automated deployment process on Heroku is the possibility of adding more stages to the process. Heroku has a continuous integration setup which allows you to add stages to the deployment chain [37].

Pipelines may be added to the continuous integration setup for categorizing the apps in four stages: Review, Development, Staging and Production. This allows you to review, test and then deploy the apps in a simple fashion [45].

Continuous integration can be used to enable continuous delivery [82]. There are six major benefits to continuous delivery according to a paper by Chen, those are:

- Accelerated time to market
- Building the right product
- Improved productivity and efficiency
- Reliable releases
- Improved product quality
- Improved customer satisfaction [13]

Many of these benefits are very useful when discussing the prototype as an innovation and experimentation functionality as mentioned in the 3LPM from *"Achieving Simplicity with the Three-Layer Product Model"*. As the purpose for the experimentation layer is to create new products or to conduct trials [11], benefits such as an accelerated time to market and a higher efficiency are very valuable.

5.2.6 Business Aspects

When this project was initially presented, there was a discussion about charging for the add-on or not. As mentioned in section 3.1, Axis has always been using third parties to sell. If Axis wants to continue using third parties to sell, Heroku would make this possible. The other option would be that Axis starts charging for it themselves. However, according to interviewee C, this is challenging. *"I think there is a huge challenge for Axis in transforming from a client oriented company [...] Selling services is something else, I do not believe that we have a sales organization that can handle selling services, not on a global scale"*. Later, interviewee C also states that *"we are interfacing a certain number of distributors worldwide [...] they interface partners who interface end customers [...] selling directly to an end customer would mean it's an entirely different ballgame"*.

5.2.7 Licensing

An important business aspect of our prototype that needs to be considered is licensing. When agreeing to the license, what do we give away to Heroku? According to the general terms in the agreement with salesforce.com, inc. (SFDC) for Heroku Services, *"SFDC claims no ownership or control over any Content or Application. You retain copyright and any other rights you already hold in the Content and/or Application, and you are responsible for protecting those rights, as appropriate"*. In another section, SFDC states that. *"SFDC acknowledges and agrees that it obtains no right, title or interest from you (or your licensors) under these Terms in or to any Content or Applications that you create, submit, post, transmit or display on, or through, the Heroku Services, including any intellectual property rights which subsist in that Content and the Application"*. However, it is also worth noting that SFDC later states that *"You acknowledge and agree that the form and nature of the Heroku Services which SFDC provides may change from time to time without prior notice to you"* and *"You agree to comply with the Heroku Acceptable Use*

Policy available at acceptable use policy (the "Acceptable Use Policy") which is incorporated herein by this reference and which may be updated from time to time". Therefore, when deciding whether to proceed with this project, it is important to really consider these terms of agreement again.

5.3 Dynamic Validation

We launched a focus group meeting to dynamically validate our results. The results from the focus group survey can be found in section 5.3.1, these are followed by a thematic analysis in section 5.3.2.

5.3.1 Focus Group: Survey Results

The results from the focus group survey can be found in figure 5.1. The figure includes shortened versions of the questions, for the full questionnaire see appendix B. A discussion of the results can be found in section 5.3.2.

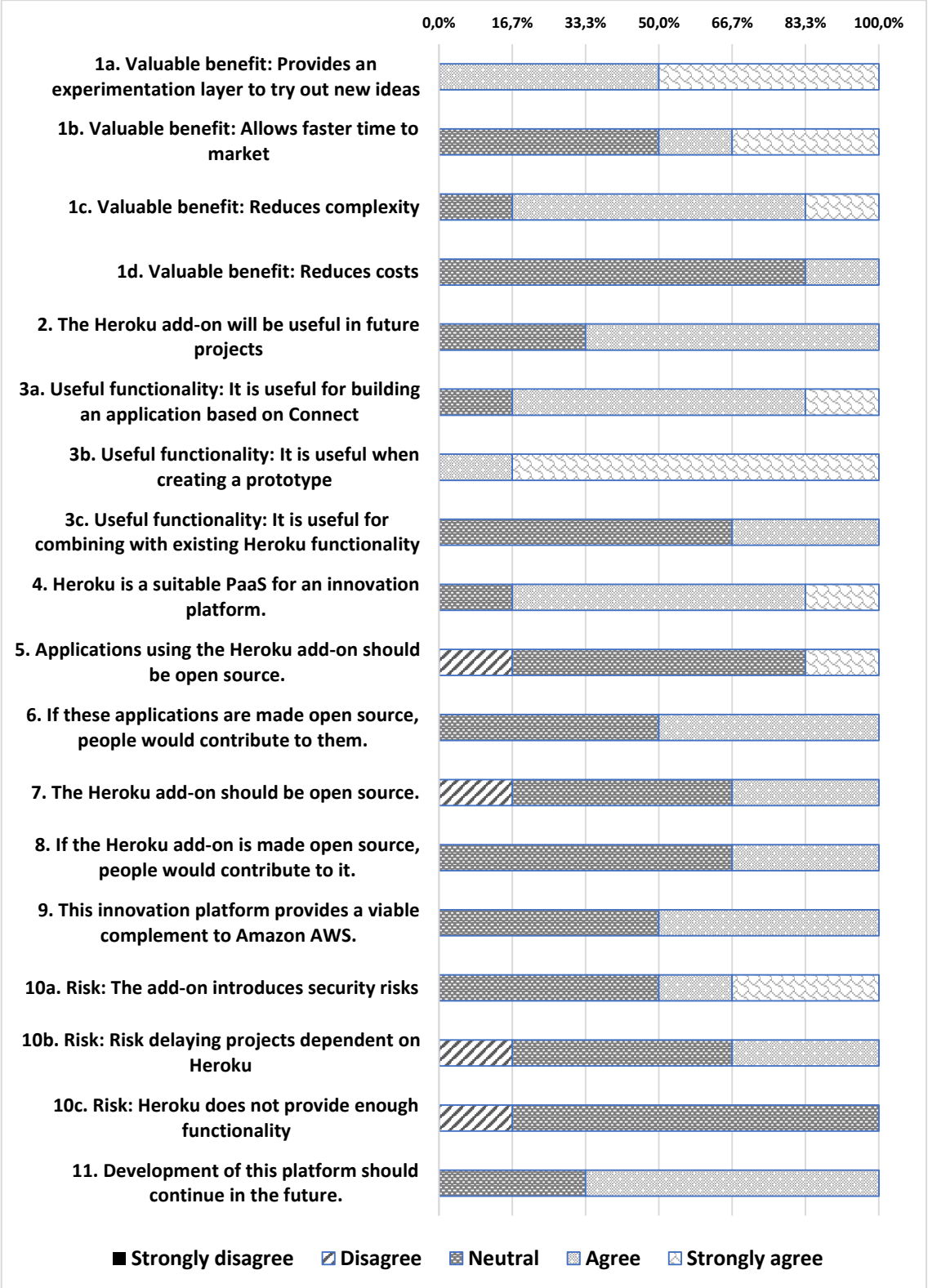


Figure 5.1: Focus group survey results

5.3.2 Focus Group: Thematic Analysis

This section includes a discussion with regard to every theme found in the thematic analysis where it relates to findings from the interviews, survey results and the focus group discussion. The themes are defined in table 5.1.

Benefits/values

Regarding the outcomes that Axis imagines are possible, the platform is seen as an innovation platform. As one of the participants of the focus group mentions, you can *"just push your thing and it's... built and released."*. Another participant stated that *"it's really good for trying out concepts and so on. But if you want to use it in production then that's a completely different thing. [...] for prototyping it seems only positive. Easy."*. Looking at the results from the survey, everyone agrees or strongly agrees to that the platform provides an experimentation layer to try out new ideas and that it is useful when creating a prototype. Furthermore, almost everyone thinks that it reduces complexity. Another thing that is mentioned as a value is that the platform *"is a lot faster in prototyping"*, compared to the current solution.

During the interviews, our platform was seen as an innovation platform. This is confirmed by the focus group, where 50% of the participants agree and 50% strongly agree on this point as shown in figure 5.1, question 1a. According to one of the participants, the platform is a lot faster in prototyping compared to the current solution, reducing complexity (Question 1c; 83% agree or strongly agree) and allowing faster time to market (Question 1b; 50% agree or strongly agree). Overall, 67% agree on that development of our platform should continue in the future while 33% were neutral (Question 11).

Business

Discussing business aspects that Axis considers to be relevant, 67% agree that the Heroku add-on will be useful in future projects as shown in figure 5.1, question 2. One potential future project mentioned during the discussion is a web based VMS, *"where you can see your cameras on AxisConnect and not have to have any fixed client. Windows or..."*

One recurring subject during the interviews was the payment model — should Axis sell using third parties or charge themselves? This topic was not mentioned at all during the focus group. This might be because only developers were present.

Challenges/limitations

When it comes to the obstacles with our project discussed during the focus group, the main concern is security. *"For me it's an unknown. So it's something that we need to.... Either contain really strictly and test or understand perfectly and then maintain our own branch of it and so on. So... That's how I see the risk."*. The same participant also states that *"for actual production [...] you really have to know... How secure is it and so on, that's really important because all the data is... that we handle can be really sensitive. So it's a big concern for us."*. Later, another participant remarks that *"A big problem is that it authenticates towards Axis Connect. And then you have to make sure that the Heroku app is properly maintained and secure so it doesn't give anyone access into Connect calls that*

they shouldn't be able to." A third participant points out that *"we add another platform that needs to be reached, one more failure point."*. This is something that does indeed need to be considered, as one of our demos had issues due to changes made by Heroku.

A limitation mentioned during the discussion is that we might want some functionality that isn't present in Heroku and that Axis might have to wait for Heroku to develop this functionality since Axis doesn't own Heroku's development.

Looking at the thematic analysis of the interviews and the thematic analysis of the focus group, both are concerned with security, especially regarding Axis Connect. As one of the participants of the focus group states, we do need to test and understand this perfectly before really launching our innovation platform.

Another common denominator is that our project is dependent on other projects. According to one interviewee, our project is very dependent on Connect prioritizing our issues while participants of the focus group argue that we might have to wait for Heroku to implement functionality.

Drivers

Reasoning about why Axis should invest in this project, the participants of the focus group think that our innovation platform is a viable complement to AWS. *"It's viable because it's easier. That's mostly I think where it's strength lies."*

Reasons for Axis to invest in this project was not really discussed during the focus group, perhaps since only developers participated. However, one of the questions in the survey was whether the prototype provides a viable complement to Amazon AWS and then 50% agree and 50% are neutral as seen in question 9, figure 5.1.

Level of openness

During the discussion on whether or not the platform and the proof of concept should be open sourced or not, one participant stated that *"it depends on the application. If it's an application that is very close to Axis' interests, I think it would not be open source because we want to profit from it."*. A reason to not make the add-on open source according to one of the participants is that *"the add-on integrates a lot with Connect and Connect isn't open source. So it might be an example of just... We don't want open source to encroach too much of Connect."*

When discussing why people would contribute if it was open source, one remark is that *"[contributors] want some kind of specific functionality for themselves. Maybe they would do it then. If they like it but it's lacking something they might..."*.

While discussing whether or not the prototype should be open source or whether or not the applications using the prototype should be open source during the interviews, there were some mixed opinions. This also holds after conducting the focus group. When it came to the applications, the majority of the participants (67%) were neutral, while people were more positive towards open sourcing the prototype as seen in figure 5.1, questions 5 and 7, respectively. This might be because it is hard to grip what open innovation is and what it offers. We gave a short presentation and a demo, but the developers need more time to discover the potentials of our platform.

Chapter 6
Discussion

In the following sections, we discuss the research questions that were defined in section 3.3 based on the results and the analysis from chapter 5 and relate it to existing literature.

6.1 RQ1: What are the drivers for Axis to create an innovation platform?

To answer **RQ1**: There are many drivers that have been mentioned both in the interviews and in the focus group. Different people see different potentials with the project.

The most mentioned drivers were:

- Enable faster speed of innovation
- Achieve a higher customer satisfaction
- Deal with less complexity

Depending on which role a person had in the company, the drivers were sometimes different, as discussed in section 5.1.1.

Section 5.1.1 shows that many of the drivers that were mentioned in the interviews were in line with the innovation drivers according to Rose and Furneaux such as Innovation leadership, Innovation evaluation and Innovation tools and techniques [78]. We think that the reasoning as to why Axis, or any other company for that matter, is so firmly committed to innovation is a number of factors and not an easy question to answer. Rose and Furneaux claim that globalization, standardization and industrialization are major factors [78] and we agree with those claims. In Axis' case there could be more factors at play however, as Axis is a very innovation driven company with a large product line. They seem to always be trying to challenge themselves and come up with new ideas. Axis has hosted around 1000 master thesis students over the years and according to us, this reinforces the idea

that they highly value new ways of doing things, and can therefore be driven by a need to always reinvent themselves.

The business reasons for the drivers seem mainly to be gaining a competitive edge by spending less time on things that are deemed “non-essential”. The focus group’s reaction to us deploying the proof-of-concept reinforced this theory as they were very positive to the idea of having such a simple deployment process.

The drivers speak to a need of developing in a quicker fashion and getting faster results. If we were to guess how the drivers would change in the future, they would largely stay the same but they would be even more relevant as technology keeps accelerating and tools are needed to keep up with the pace.

6.2 RQ2: What are the potentials of web-based deployment?

In this thesis we have developed a prototype (**RQ2**) which is a Heroku add-on that can be installed by developers who want to develop an application. The add-on takes care of authentication, redirects calls to Axis Connect and combines Connect API calls into new API calls.

As mentioned in chapter 2, there is a study by Brad et al. about a supportive tool that can aid open innovation [12]. Our tool is not a supportive tool but rather a standalone tool which allows developers to deploy and run applications and we feel that there are many benefits to this. The main benefit is that of a scientific standpoint. That is, if developers try our platform, it is easy to measure and get feedback from their perceived benefits and drawbacks in comparison to their normal workflow.

The prototype and the proof of concept was demoed during the presentation part of the focus group. Both the people from the interviews and the people from the focus group were positive about the concept of the platform. By developing the prototype, we have provided an experimentation layer to try out new things and we have shown that time to market and complexity is reduced, benefits that have also been discussed in the paper by Costache et al. [17]. With the help of our prototype and available Heroku add-ons, a developer can try out things without the need to involve other developers. The application of our platform is therefore mainly quick prototyping.

There are several features that could be implemented if there was more time, the major one being video streaming. The proof of concept was however enough to illustrate the potentials of our project and to give an idea of how the platform could be used and what benefits it might lead to.

6.3 RQ3: What value does the innovation platform create and why is it attractive to Axis?

Regarding **RQ3**, Axis mentions that they see our platform as an innovation platform which enables fast prototypes, giving customers a demo quickly. With the help of it, the release cycle can be reduced and feedback can be obtained within hours instead of after a full release cycle. Relating it to the Three Layer Product Model by Bosch, our platform is in the innovation layer [11].

Another value with the innovation platform is that it makes it easy to build services. The innovation platform is attractive to Axis because it is easier than Amazon AWS when it comes to quickly setting up a prototype. It is also important for Axis to not be reliant on Amazon for all their web-service needs.

During the interviews, we talked about the importance of creating something that another Axis product does not offer. The functionality that our platform offers and the possibility to combine our add-on with other Heroku add-ons to create new functionality might be another reason why our platform is attractive to Axis.

The idea of having the prototype and/or the proof of concept open source received mixed reviews both during the interviews and in the focus group survey. We have previously mentioned aspects as to why companies can be concerned about open source in section 5.1.1 where the largest concern was giving away intellectual property right that instead could have given a competitive advantage according to Munir et al. [62]. Many of the respondents to the focus group survey were neutral to the idea of having the prototype and/or proof of concept open source. Since the participants are not negative, there may also be other factors at play than just a concern about open source. Looking at another paper by Munir et al. [63], specifically at the model of openness for tools, it seems that Axis could be categorized somewhere between the leverage category and the lucrativeness category as they seem to show traits of both strategies. They are an avid user of OSS and collaborate regarding open source strategies and in that sense they are open. However, when they feel that they have a competitive edge, they are reluctant to share.

The values deemed important to the proof of concept and prototype as mentioned in 3.4.3 were in line with what the interviewees valued in the platform. Two aspects, Completeness and Performance, were not mentioned in the interviews. The reason Completeness was not mentioned was probably due to it being rather diffuse. Performance is harder to analyze why it was not mentioned. One possibility could be due to the prototype is a web platform, it will have a similar magnitude of performance to alternatives (such as AWS) independent on the implementation as there are more limiting factors in web development such as latency and running it in a web browser.

The paper by Khurum et al. discusses the move a company can make from cost-based reasoning to making value-based decisions [59]. As our platform would be mainly for developers at Axis it would generate a cost, but the results discovered in this paper show that there are clear and tangible values associated with an innovation platform. These aspects can be used for value-based decisions regarding an innovation platform in the future.

6.4 Future Work

Reasoning about work that still needs to be done regarding the platform, thorough validation of the platform is on top of this list. As of now, the platform has not really been used by other developers at Axis or at any partners of Axis. Another thing that needs to be tested is how the platform works with multiple users.

The innovation platform has much to gain from being investigated further. To do this one could have testers try out the platform, either in a control group or to invite people at Axis who are planning to create a prototype. This would bring further results into whether the platform is usable in a real world scenario and if the theoretical benefits and values actually exist.

There are several more features of the prototype that could be implemented. The first and major one is implementing video stream from the VAPIX part of Axis Connect, which would add much to the overall functionality of the prototype. Authenticating which application is using the add-on and having different pricing plans and corresponding functionality would also be a great addition. Cloud storage is being discussed at Axis Connect and this could be a part of the prototype, especially when you have the possibility to charge extra for it by the use of a specific pricing plan.

Chapter 7
Conclusions

Cloud services can aid developers by enabling faster feedback from customers and easier deployment.

In this thesis we have developed a prototype which is a Heroku add-on that can be installed by developers who want to develop an application (**RQ2**). The add-on takes care of authentication, redirects calls to Axis Connect and combines Connect API calls into new API calls. To show the viability of the platform, a proof of concept was developed.

With the help of interviews, we concluded why such a platform is attractive to Axis and what the drivers are for Axis to create an innovation platform, thereby addressing **RQ1**. Here, the main driver was to increase the speed of innovation.

A focus group was held to present the prototype, demonstrate the proof of concept and evaluate the values associated with the innovation platform (**RQ3**). The focus group were positive to the idea of having the platform be an innovation platform used for prototyping, but there were some disagreements whether the platform should be open source or not.

Bibliography

[1] Paula Austel, Han Chen, Thomas Mikalsen, Isabelle Rouvellou, Upendra Sharma, Ignacio Silva-Lepe, and Revathi Subramanian. "Continuous delivery of composite solutions: A case for collaborative software defined paas environments". In: *Proceedings of the 2nd International Workshop on Software-Defined Ecosystems*. ACM, 2015, pp. 3–6.

[2] *AWS Elastic Beanstalk*. URL: https://aws.amazon.com/elasticbeanstalk/. (accessed: 23.02.2018).

[3] *AWS Elastic Beanstalk: Controlling Access to Elastic Beanstalk*. URL: https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/AWSHowTo.iam.managed-policies.html. (accessed: 23.02.2018).

[4] *AWS Elastic Beanstalk: Deploying Elastic Beanstalk Applications from Docker Containers*. URL: https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/create%5C_deploy%5C_docker.html. (accessed: 23.02.2018).

[5] *AWS Elastic Beanstalk: Elastic Beanstalk Supported Platforms*. URL: https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/concepts.platforms.html. (accessed: 07.03.2018).

[6] *AWS Elastic Beanstalk Pricing*. URL: https://aws.amazon.com/elasticbeanstalk/pricing/. (accessed: 14.03.2018).

[7] *AWS Elastic Beanstalk: Using Elastic Beanstalk with Amazon Virtual Private Cloud*. URL: https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/vpc.html. (accessed: 14.03.2018).

[8] *AWS: Processor Speculative Execution Research Disclosure*. URL: https://aws.amazon.com/security/security-bulletins/AWS-2018-013/. (accessed: 14.03.2018).

[9] Axis. *Annual & Sustainability Report 2016*. Lund, 2016.

[10] Jay Barbour. *Masergy - How To Secure IaaS/PaaS Environments Effectively With Cloud Workload Protection*. Dec. 2017. URL: https://www.masergy.com/blog/secure-iaas-paas-environments-effectively-cloud-workload-protection/. (accessed: 23.02.2018).

[11] Jan Bosch. "Achieving simplicity with the three-layer product model." In: *Computer 46.11* (2013), pp. 34–39.

[12] Stelian Brad, Mircea Fulea, Bogdan Mocan, Adina Duca, and Emilia Brad. "Software platform for supporting open innovation". In: *Automation, Quality and Testing, Robotics, 2008. AQTR 2008. IEEE International Conference on. Vol. 3. IEEE* (2008).

[13] Lianping Chen. "Continuous Delivery: Huge Benefits, but Challenges Too." In: *IEEE Software 32* (2015), pp. 50–54.

[14] Henry Chesbrough. "Open innovation: A new paradigm for understanding industrial innovation". In: *Open Innovation. Researching a New Paradigm*. Ed. by H. Chesbrough W. Vanhaverbeke and J. West. New York, NY: Oxford University Press, 2006. Chap. 1, pp. 1–12.

[15] Henry Chesbrough. *Open Innovation: The New Imperative for Creating and Profiting from Technology*. Boston, MA: Harvard Business School Press, 2003.

[16] Henry Chesbrough. "Open innovation: Where we've been and where we're going." In: *Research-Technology Management 55.4* (2012), pp. 20–27.

[17] Stefania Costache, Djawida Dib, Nikos Parlavantzas, and Christine Morin. "Resource management in cloud platform as a service systems: Analysis and opportunities." In: *Journal of Systems and Software 132* (2016), pp. 98–118.

[18] Cloud Standards Customer Councils. "Practical Guide to Platform-as-a-Service Version 1.0". In: (2015).

[19] *Cross-Origin Resource Sharing (CORS)*. May 2018. URL: https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS. (accessed: 29.05.2018).

[20] Dhaivat Dave, Nayana Meruliya, Tirth D. Gajjar, Grishma T. Ghoda, Disha H. Parekh, and R. Sridaran. "Cloud Security Issues and Challenges". In: *Big Data Analytics* (2018), pp. 499–514.

[21] *Dokku*. URL: http://dokku.viewdocs.io/dokku/. (accessed: 23.02.2018).

[22] *Dokku: Buildpack Deployment*. URL: http://dokku.viewdocs.io/dokku/deployment/methods/buildpacks/. (accessed: 07.03.2018).

[23] *Dokku: Dockerfile Deployment*. URL: http://dokku.viewdocs.io/dokku/deployment/methods/dockerfiles/. (accessed: 23.02.2018).

[24] *Dokku: Per App Permission*. URL: https://github.com/dokku/dokku/issues/1532. (accessed: 23.02.2018).

[25] *Find your Platform as a Service!* URL: https://paasfinder.org/filter. (accessed: 23.02.2018).

[26] Trey Ford. *Heroku: Meltdown and Spectre Security Update*. Jan. 2018. URL: https://blog.heroku.com/meltdown-and-spectre-security-update. (accessed: 23.02.2018).

[27] Oliver Gassmann and Ellen Enkel. "Towards a Theory of Open Innovation: Three Core Process Archetypes". In: *R&D Management Conference (RADMA)*. 2004.

[28] Nick Gauthier. *Building Minimal Docker Containers for Go Applications*. Feb. 2018. URL: https://blog.codeship.com/building-minimal-docker-containers-for-go-applications/.

[29] Ali Gholami and Erwin Laure. "Security and Privacy of Sensitive Data in Cloud Computing: A Survey of Recent Developments". In: *arXiv preprint arXiv:1601.01498* (2016).

[30] *Google - Project Zero*. URL: https://googleprojectzero.blogspot.se/. (accessed: 23.02.2018).

[31] Anthony Gore. *React or Vue: Which Javascript UI Library Should You Be Using?* Dec. 2016. URL: https://medium.com/js-dojo/react-or-vue-which-javascript-ui-library-should-you-be-using-543a383608d. (accessed: 29.05.2018).

[32] Sumit Goyal. "Software as a Service, Platform as a Service, Infrastructure as a Service - A Review". In: *International Journal of Computer Science & Network Solutions 1.3* (2013), pp. 53–67.

[33] Radha Guha and David Al-Dabass. "Impact of web 2.0 and cloud computing platform on software engineering." In: *International Symposium on Electronic System Design (ISED)*. IEEE, 2010, pp. 213–218.

[34] John Hannah. *Why Is React More Popular Than Angular?* Jan. 2018. URL: https://javascriptreport.com/why-is-react-more-popular-than-angular/. (accessed: 16.04.2018).

[35] *Heroku*. URL: https://www.heroku.com/. (accessed: 23.02.2018).

[36] *Heroku Buildpack for create-react-app*. URL: https://elements.heroku.com/buildpacks/nhutphuongit/create-react-app-buildpack. (accessed: 16.04.2018).

[37] *Heroku CI*. URL: https://www.heroku.com/continuous-integration. (accessed: 03.05.2018).

[38] *Heroku: Detailed Comparison*. URL: https://www.heroku.com/pricing#dyno-comparison. (accessed: 14.03.2018).

[39] *Heroku Dev Center: Add-ons Overview*. May 2018. URL: https://devcenter.heroku.com/articles/add-ons. (accessed: 29.05.2018).

[40] *Heroku Dev Center: Boot timeout*. Apr. 2018. URL: https://devcenter.heroku.com/articles/limits#boot-timeout. (accessed: 29.05.2018).

[41] *Heroku Dev Center: Bringing an Add-on to Market - Add-on plans and pricing*. May 2018. URL: https://devcenter.heroku.com/articles/bringing-an-add-on-to-market#add-on-plans-and-pricing. (accessed: 29.05.2018).

[42] *Heroku Dev Center: Building an Add-on*. May 2018. URL: https://devcenter.heroku.com/articles/building-an-add-on. (accessed 29.05.2018).

[43] *Heroku Dev Center: Buildpacks*. Apr. 2018. URL: https://devcenter.heroku.com/articles/buildpacks. (accessed: 29.05.2018).

[44] *Heroku Dev Center: Container Registry & Runtime (Docker Deploys)*. May 2018. URL: https://devcenter.heroku.com/articles/container-registry-and-runtime. (accessed: 29.05.2018).

[45] *Heroku Dev Center: Pipelines*. May 2018. URL: https://devcenter.heroku.com/articles/pipelines. (accessed: 29.05.2018).

[46] *Heroku Dev Center: Using App Permissions in Enterprise Teams*. Jan. 2018. URL: https://devcenter.heroku.com/articles/app-privileges-in-heroku-organizations. (accessed: 23.02.2018).

[47] *Heroku Dynos*. URL: https://www.heroku.com/dynos. (accessed: 23.04.2018).

[48] *Heroku Elements: CameraTag*. URL: https://elements.heroku.com/addons/cameratag. (accessed: 26.04.2018).

[49] *HEROKU, INC. ADD-ONS LICENSE AND DISTRIBUTION AGREEMENT*. URL: https://addons.heroku.com/provider/resources/HerokuAddonsLicenseAgreement.pdf. (accessed: 23.02.2018).

[50] *Heroku: Kensa*. URL: https://github.com/heroku/kensa. (accessed: 08.03.2018).

[51] *Heroku Postgres*. URL: https://elements.heroku.com/addons/heroku-postgresql. (accessed: 16.04.2018).

[52] *Heroku: Stacks*. May 2018. URL: https://devcenter.heroku.com/articles/stack. (accessed: 29.05.2018).

[53] *Heroku Status*. URL: https://status.heroku.com/. (accessed: 14.05.2018).

[54] *Heroku: What is Heroku?* URL: https://www.heroku.com/about. (accessed: 23.04.2018).

[55] Alan Hevner, Salvatore March, Jinsoo Park, and Sudha Ram. "Design Science in Information Systems Research". In: *MIS Quarterly 28.1* (2004), pp. 75–105.

[56] C. N. Höfer and Georgios Karagiannis. "Cloud computing services: taxonomy and comparison". In: *Journal of internet services and applications 2.2* (2011), pp. 81–94.

[57] Syed Asad Hussain, Mehwish Fatima, Atif Saeed, Imran Raza, and Raja Khurram Shahzad. "Multilevel classification of security concerns in cloud computing". In: *Applied Computing and Informatics 13.1* (2017), pp. 57–65.

[58] Jesper Joergensen. *Heroku: Introducing Heroku Private Spaces: Private PaaS, delivered as-a-Service*. Sept. 2015. URL: https://blog.heroku.com/heroku_private_spaces_private_paas_delivered_as_a_service. (accessed 14.03.2018).

[59] Mahvish Khurum, Tony Gorschek, and Magnus Wilson. "The software value map - an exhaustive collection of value aspects for the development of software intensive products". In: *Journal of Software: Evolution and Process 25.7* (2013), pp. 711–741.

[60] Johan Linåker, Hussan Munir, Krzysztof Wnuk, and Carl Eric Mols. "Motivating the contributions: An Open Innovation perspective on what to share as Open Source Software". In: *The Journal of Systems and Software 135* (2018), pp. 17–36.

[61] Jane McCallion. *Heroku plugs password security hole*. Jan. 2013. URL: http://www.itpro.co.uk/645013/heroku-plugs-password-security-hole. (accessed: 02.05.2018).

[62] Hussan Munir, Johan Linåker, Krzysztof Wnuk, Per Runeson, and Björn Regnell. "Open innovation using open source tools: a case study at Sony Mobile." In: *Empirical Software Engineering 23.1* (2018), pp. 186–223.

[63] Hussan Munir, Per Runeson, and Krzysztof Wnuk. "A Theory of Openness for Software Engineering Tools in Software Organizations." In: *Information and Software Technology* (2017), pp. 26–45.

[64] Steve Neely and Steve Stolt. "Continuous Delivery? Easy! Just Change Everything (well, maybe it is not that easy)". In: *2013 Agile Conference (AGILE)*. IEEE, 2013, pp. 121–128.

[65] Jens Neuhaus. *Angular vs. React vs. Vue: A 2017 comparison*. Sept. 2017. URL: https://medium.com/unicorn-supplies/angular-vs-react-vs-vue-a-2017-comparison-c5c52d620176. (accessed: 29.05.2018).

[66] Bashar Nuseibeh and Steve Easterbrook. "Requirements Engineering: A Roadmap". In: *Proceedings of the Conference on the Future of Software Engineering*. ACM, 2000, pp. 35–46.

[67] *Om Axis Communications*. URL: https://www.axis.com/se/sv/about-axis. (accessed: 26.02.2018).

[68] *OpenID Connect explained*. URL: https://connect2id.com/learn/openid-connect. (accessed: 16.04.2018).

[69] *OpenShift*. URL: https://www.openshift.com/. (accessed: 23.02.2018).

[70] *OpenShift: All Add-ons*. URL: https://hub.openshift.com/addons. (accessed: 23.02.2018).

[71] *OpenShift Documentation*. URL: https://docs.openshift.com/. (accessed: 23.02.2018).

[72] *OpenShift: Load Focus*. URL: https://hub.openshift.com/addons/41-load-focus. (accessed: 14.03.2018).

[73] *OpenShift: Plans & Pricing*. URL: https://www.openshift.com/pricing/index.html. (accessed: 14.03.2018).

[74] *OpenShift: Supported Container Images*. URL: https://www.openshift.com/features/containers.html. (accessed: 07.03.2018).

[75] *OpenShift: What OpenShift Online and Dedicated customers should know about Meltdown and Spectre*. Jan. 2018. URL: https://blog.openshift.com/openshift-online-dedicated-meltdown-spectre/. (accessed: 23.02.2018).

[76] *Platform as a Service, Increasing Cloud Adoption by Giving Developers the Key to Cloud-Aware Development*. Aug. 2013. URL: https://www.intel.com/content/dam/www/public/us/en/documents/white-papers/cloud-computing-paas-cloud-demand-paper.pdf. (accessed: 16.04.2018).

[77] Travis Reeder. *How to Run Dockerized Apps on Heroku… and it's pretty sweet*. June 2017. URL: https://medium.com/travis-on-docker/how-to-run-dockerized-apps-on-heroku-and-its-pretty-great-76e07e610e22. (accessed: 23.02.2018).

[78] Jeremy Rose and Brent Furneaux. "Innovation Drivers and Outputs for Software Firms: Literature Review and Concept Development," in: *Advances in Software Engineering 2016* (2016), 25 pages.

[79] Per Runeson and Martin Höst. "Guidelines for conducting and reporting case study research in software engineering". In: *Empirical Software Engineering 14.2* (2009), pp. 131–164.

[80] Char Sample. *An examination of PaaS security challenges*. Sept. 2012. URL: http://searchcloudsecurity.techtarget.com/tip/An-examination-of-PaaS-security-challenges. (accessed: 23.02.2018).

[81] Alex Soto. *Deploying Docker Images to OpenShift*. May 2017. URL: https://dzone.com/articles/deploying-docker-images-to-openshift. (accessed: 23.02.2018).

[82] Daniel Stahl, Torvald Martensson, and Jan Bosch. "Continuous practices and devops: beyond the buzz, what does it all mean?" In: *2017 43rd Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. IEEE, 2017, pp. 440–448.

[83] Qi Zhang, Lu Cheng, and Raouf Boutaba. "Cloud computing: state-of-the-art and research challenges". In: *Journal of internet services and applications 1.1* (2010), pp. 7–18.

Appendices

Appendix A
Interview Questions

A.1 Initial Interview Questions

Demographics/Personal

- What is your role in the organization (Job title?)
- Which department do you work in? For how long?
- What is your daily work and responsibilities?
- How many years of experience do you have?
- Do you have any experience with OSS? Open platforms?
- Have you worked with or do you know about AWS?

Drivers

- What was the trigger for finding a replacement or complement for AWS?
- What is the reason to switch/investigate moving from Amazon AWS?

Challenges

- What are the challenges when using Amazon AWS?
- What are the major potential limitations that Axis sees with this project?
- Are there any risks associated with this thesis?
- What problems do you face in the deployment process with AWS? (Talk/ask about solutions, for instance Workflow, Productivity)

Benefits/values

- What are the major potential benefits that Axis imagines are possible?
- How should Axis continue with this project?
- Can this project be made open? (If yes: Why? If no: Why not?)
- Why would people contribute to this project?

A.2 Final Set of Interview Questions

Demographics/Personal

- What is your role in the organization (Job title?)
- Which department do you work in? For how long?
- What is your daily work and responsibilities?
- How many years of experience do you have?
- Have you contributed to any OSS? Or used OSS? Open platforms?
- Have you worked with or do you know about AWS?

Drivers

- What could be a reason to move to our platform?
- What could be a reason to switch/investigate moving from Amazon AWS?

Challenges

- What are the challenges when using Amazon AWS?
- What problems do you face in the deployment process with AWS? (Talk/ask about solutions, for instance Workflow, Productivity)
- What are the major potential limitations that Axis sees with this project?
- Are there any risks associated with our project?

Benefits/values

- What are the major potential benefits that Axis imagines are possible?
- How should Axis continue with this project?
- Do you have any suggestions of a project that our add-on can be used for?
- Can this project be made open? (If yes: Why? If no: Why not?)
- Why would people contribute to this project?

Appendix B
Focus Group

This section includes the focus group questions.

Focus Group

* Required

1. **Job Title** *

2. **Years of Experience** *

3. **Have you contributed to open source?** *
   Mark only one oval.
   - Yes
   - No

4. **Have you worked with Amazon AWS?** *
   Mark only one oval.
   - Yes
   - No

5. **If no, do you know about Amazon AWS?**
   Mark only one oval.
   - Yes
   - No

6. **1. Heroku provides a valuable benefit to the deployment process.** *
   Mark only one oval per row.

	Strongly disagree	Disagree	Neutral	Agree	Strongly agree
Provides an experimentation layer to try out new ideas					
Allows faster time to market					
Reduces complexity					
Reduces costs					

7. **Additional benefits not mentioned**

8. **2. The Heroku add-on will be useful in future projects.** *
Mark only one oval.

- ◯ Strongly disagree
- ◯ Disagree
- ◯ Neutral
- ◯ Agree
- ◯ Strongly agree

9. **3. The Heroku add-on provides useful functionality.** *
Mark only one oval per row.

	Strongly disagree	Disagree	Neutral	Agree	Strongly agree
It is useful for building an application based on Connect.	◯	◯	◯	◯	◯
It is useful when creating a prototype.	◯	◯	◯	◯	◯
It is useful for combining with existing Heroku functionality.	◯	◯	◯	◯	◯

10. **4. Heroku is a suitable PaaS for an innovation platform.** *
Mark only one oval.

- ◯ Strongly disagree
- ◯ Disagree
- ◯ Neutral
- ◯ Agree
- ◯ Strongly agree

11. **5. Applications using the Heroku add-on should be open source.** *
Mark only one oval.

- ◯ Strongly disagree
- ◯ Disagree
- ◯ Neutral
- ◯ Agree
- ◯ Strongly agree

12. **6. If these applications are made open source, people would contribute to them.** *
Mark only one oval.

- ◯ Strongly disagree
- ◯ Disagree
- ◯ Neutral
- ◯ Agree
- ◯ Strongly agree

13. **7. The Heroku add-on should be open source.** *
Mark only one oval.

- ◯ Strongly disagree
- ◯ Disagree
- ◯ Neutral
- ◯ Agree
- ◯ Strongly agree

14. **8. If the Heroku add-on is made open source, people would contribute to it.** *
Mark only one oval.

- ◯ Strongly disagree
- ◯ Disagree
- ◯ Neutral
- ◯ Agree
- ◯ Strongly agree

15. **9. This innovation platform provides a viable complement to Amazon AWS.** *
Mark only one oval.

- ◯ Strongly disagree
- ◯ Disagree
- ◯ Neutral
- ◯ Agree
- ◯ Strongly agree

16. **10. There are several risks associated with creating an application based on this Heroku add-on.** *
Mark only one oval per row.

	Strongly disagree	Disagree	Neutral	Agree	Strongly agree
The add-on introduces security risks.	◯	◯	◯	◯	◯
Risk delaying projects dependent on Heroku	◯	◯	◯	◯	◯
Heroku does not provide enough functionality	◯	◯	◯	◯	◯

17. **Additional risks not mentioned**

18. **11. Development of this platform should continue in the future.** *
Mark only one oval.

- Strongly disagree
- Disagree
- Neutral
- Agree
- Strongly agree

19. **Additional comments**

Powered by
Google Forms

Appendix C
Screenshots

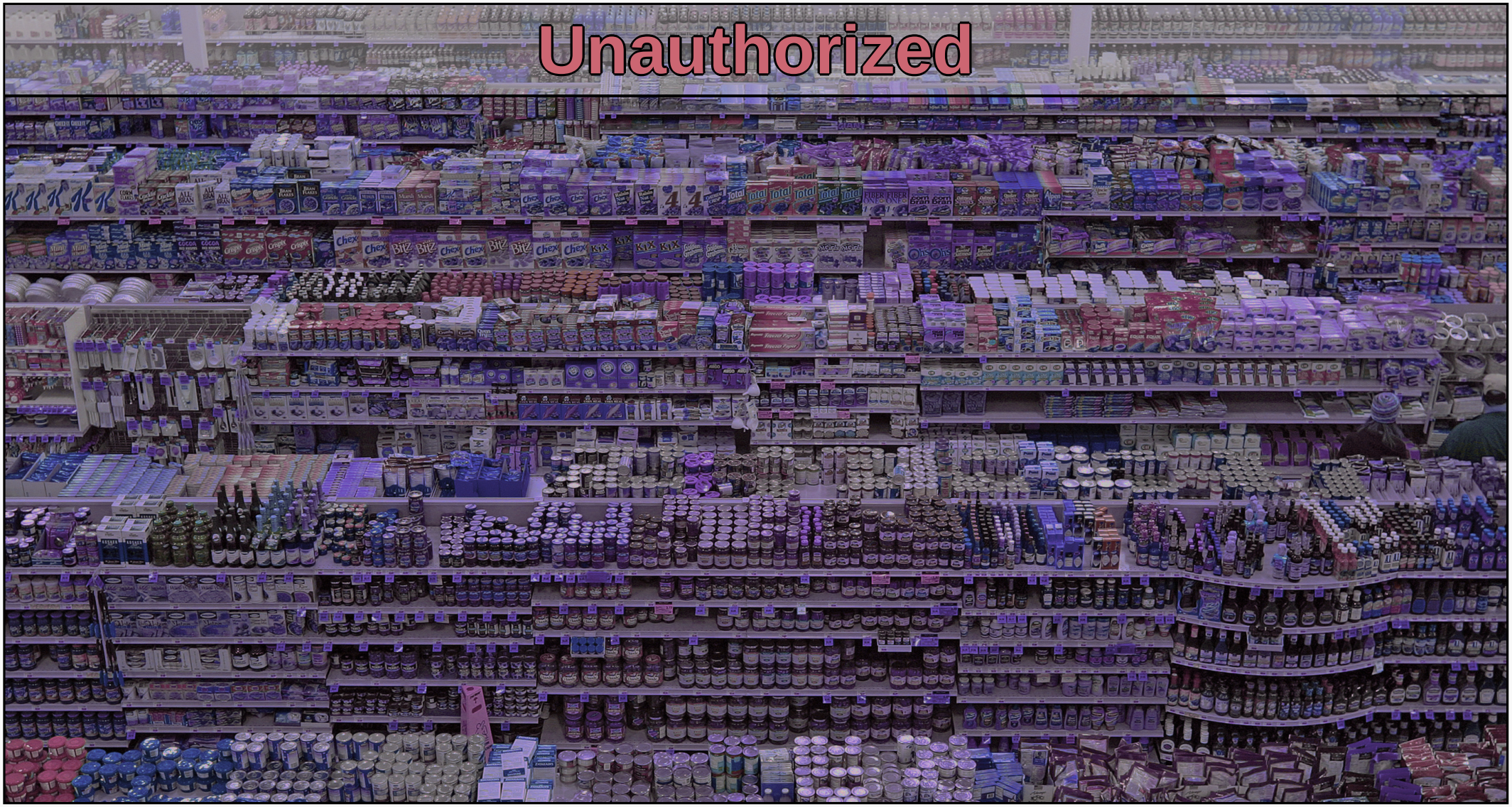
This appendix includes screen shots of the proof of concept.


Authorized
Unread messages: 1
Revenue this year: $1525335610352
Next delivery: Friday

WDT Demo - Grocery store

BUILT ON HEROKU

Intro Analytics Live Front Back Register History Login



EXAMENSARBETE In-depth study of the potentials of web-based deployment in product development
STUDENTER Patrik Danielsson, Tom Postema
HANDLEDARE Hussan Munir (LTH), Theis Hasselgaard (Axis) och Petter Johansson (Axis)
EXAMINATOR Per Runeson (LTH)

Möjligheter med molnbaserad mjukvaruutveckling

POPULÄRVETENSKAPLIG SAMMANFATTNING **Patrik Danielsson, Tom Postema**

Teknikens framfart är påtaglig i många branscher. Mjukvaruutvecklare strävar efter att utveckla sina program snabbare, få återkoppling från kunder lättare och de vill vara mer innovativa. Nya verktyg, såsom webbplattformen Heroku har skapats för att reducera komplexiteten, men fungerar det verkligen?

Molntjänster, vilket innebär att du jobbar mot en server över internet istället för din lokala dator, hjälper utvecklare att påskynda mjukvaruutveckling. Mjukvaruutveckling på molntjänster kan däremot ibland vara krångligt, detta eftersom det krävs mycket erfarenhet och tid att sätta sig in i hur traditionella molntjänster fungerar.

Inom mjukvaruutveckling finns ett begrepp som heter "open source". Detta betyder att alla byggstenar för programmet är tillgängliga och kan undersökas, kopieras och förändras. Det finns många fördelar med open source, ett exempel är ökad innovation.

För att undersöka om det finns möjliga förbättringar för hur Axis arbetar med mjukvaruutveckling byggde vi en plattform för att visa möjligheterna som kan finnas med molntjänster. Denna plattform är en prototyp som framför allt är tänkt att fungera som en innovationsplattform för mjukvaruutvecklare där de har möjlighet att testa nya idéer på ett snabbt och smidigt sätt. Vi undersökte om det fanns intresse hos Axis för att vårt projekt skulle vara open source.

För att förstå varför Axis vill ha en sådan plattform och vad drivkrafterna bakom plattformen är så intervjuade vi ingenjörer på Axis som arbetade med eller nära molntjänster. Många av de intervjuade svarade att de kände att den nuvarande utvecklingsprocessen med molntjänster kunde vara krånglig och att det kan finnas ett värde i att göra det på ett annat sätt. Deltagarna hade väldigt blandad respons till frågan på om vår prototyp borde vara open source eller inte.

Prototypen som vi byggde kompletterades med ett användningskoncept som utgjordes av ett exempel på hur en matbutiks hemsida hade sett ut. Detta koncept skapades helt med hjälp av plattformen.

För att utvärdera om vår prototyp tillförde värde för Axis så hölls ett fokusgruppmöte där en grupp ingenjörer fick svara på vad de tyckte om plattformen. Många var då positiva till hur program kunde skapas med hjälp utav prototypen och att det finns framtida projekt som fördelaktigt hade kunnat byggas med hjälp av plattformen. Det framkom även att de såg många potentiella risker med vår prototyp, den risken som nämndes som starkast var säkerhetsrisken. Som exempel kan det vara risken att någon stjäl information eller intellektuell egendom. Många av de intervjuade var neutrala till om plattformen i framtiden skulle vara open source och en risk som de såg med plattformen var att den var beroende av externa aktörer.