Universiteit Leiden
The Netherlands

Opleiding Informatica & Economie

Critical Success Factors and Hosting Cost Reduction for a Cloud Based Platform as a Service

Willem-Pieter van Vlokhoven

CONFIDENTIAL
Supervisors:
Dr. G.J. Ramackers
Dr. F.W. Takes
A.J. Gerrits, Nationale Nederlanden

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
www.liacs.leidenuniv.nl

August 6, 2019

Acknowledgements

First of all, I thoroughly want to thank Alain Gerrits, Product Owner & Manager at Nationale Nederlanden, for his support, guidance and feedback throughout the project. Secondly, I want to thank the whole Digital Foundation team of Nationale Nederlanden for their support and answers. Thirdly I want to thank Dr. G. J. Ramackers for his support, guidance and feedback. I also want to thank Dr. F.W. Takes for his final feedback. Next I want to thank Simon Houmes, Product Owner & Manager at Nationale Nederlanden, for his support and for bringing me in contact with Alain. Last but not least I want to thank everyone I was in contact with during my internship at Nationale Nederlanden.

Abstract

To speed up the time-to-market of front end applications at Nationale Nederlanden, a new application platform is needed. The current platform is inert due to the way it is set up. The new platform that is currently in development will change this with self-service automatic deployment and uses Amazon Web Services (AWS) Elastic Compute Cloud (EC2) instances on which these applications are deployed. In this thesis we provide two contributions. First, we interview more then a dozen different stakeholders about these platforms and enumerate the factors that are in their opinion needed to make the new platform a success. Then every factor is give a score based on their importance. With this score we can determine important or overlooked factors. Second, from these factors the cost management factor is chosen. After evaluating different methods, we will propose a method for the reduction of the operating costs using AWS spot instances and AWS Auto Scaler. As part of this approach is a bidding strategy instead of using the default settings. This approach maximises the chance that the instance is acquired and minimises the chance that the instance is terminated. This is monitored by an algorithm for which a pseudo code is presented.

Contents

Abbreviations

- API, Application Programming Interface
- AWS, Amazon Web Services
- BU, Business Unit
- C&C, Customer & Commerce (NN department)
- CIAM, Customer Identity Access Management
- CMS, Content Management System
- CSF, Critical Success Factor
- DF, Digital Foundation Team (part of Digital Stack and C&C)
- EC2, Elastic Compute Cloud (AWS)
- FAQ, Frequently Asked Questions
- IaaS, Infrastructure as a Service
- IaC, Infrastructure as Code
- I(C)T, Information (& Communications) Technologies
- NN, Nationale Nederlanden
- OS, Operating System
- PaaS, Platform as a Service
- S3, Simple Storage Service (AWS)
- Saas, Software as a Service
- SLA, Service Level Agreement
- XC, XperienCentral (CMS)

1 Introduction

This research is done in collaboration with Nationale Nederlanden (NN). The goal of this research is to enumerate and do a in-depth research on one of the Critical Success Factors (CSF's) that the stakeholders find important in the creation of a new digital application platform. Critical Success Factors are a limited number of characteristics, conditions, or variables that have a direct and serious impact on the effectiveness, efficiency, and viability of an organisation, program, or project. Activities associated with CSF must be performed at the highest possible level of excellence to achieve the intended overall objectives. Also called key success factors (KSF) or key result areas (KRA) [DEF]. So in other words these factors are characteristic for the success of the project.

This platform is being developed by the NN Customer & Commerce Digital Foundation team. It is developed for the use by other internal IT teams. So the platform is provided as a service to the other departments and teams. As such the platform can be described as an internal PaaS. PaaS is the layer that lies between the underlying system infrastructure and the overlaying application software, containing all application infrastructure services including application containers (servers), application development tools, database management systems, integration middleware, portal products, business process management suites, etc. known as "middleware" [Cha09]. The mentioned underlying infrastructure is an IaaS provided by Amazon Web Service part of Amazon. So in short the Digital Foundation team converts the purchased AWS IaaS to an PaaS for internal use.

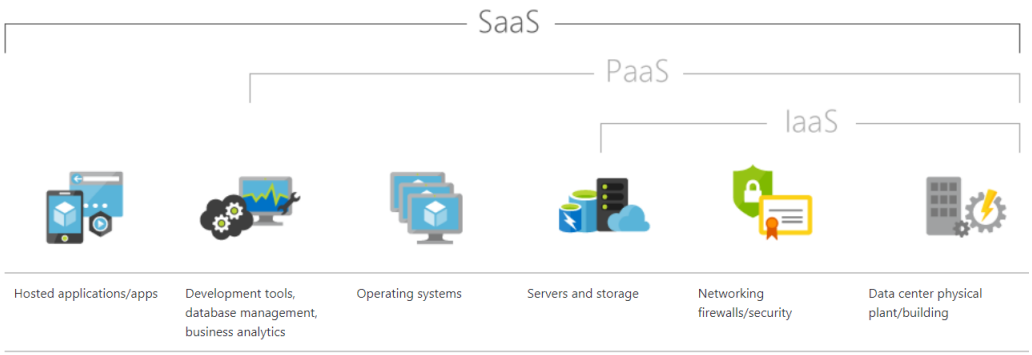

Figure 1: Differences between IaaS, PaaS and SaaS
https://azure.microsoft.com/en-us/overview/what-is-SaaS/

1.1 Thesis overview

This bachelor thesis is for the bachelor study Computer Science and Economics at the Leiden Institute of Advanced Computer Science part of Leiden University in collaboration with Nationale Nederlanden. It is supervised by Dr. G.J. Ramackers, Dr. F.W. Takes and A.J. Gerrits. In Section 3, a description of the current situation and the preferred future situation will be sketched. In Section 4 an enumeration and an explanation will be given about the Critical Success Factors of the platform based on the interviews with stakeholder, both on the providing and consumer side. These interviews can be found in Appendix A. In Section 5 first a functionality overview will be given on EC2 and in particular on spot EC2, then a research will be presented on how to reduce costs of the platform within the constraints imposed by NN.

2 Approach

2.1 Overview

The research will be split into two parts. Part one will be solely to aggregate important CSF's from the stakeholders. All these CSF's will be categorised on their operating type and requirement type. The operating types being people, process and technology. The requirement types being function requirement, quality requirement and constraint. The definitions of these types can be found [MWHK17] and [RR14] respectively.
For part two one of the CSF's enumerated in part one is thoroughly investigated. The chosen factor is "cost management", more specifically cost reduction.

2.2 CSF identification

This part will result in an enumeration of Critical Success Factors (CSF's) that are relevant for the creation and the maintenance of a successful internal PaaS.

Part 1 will consist of requirements of the stakeholders based on an applied "market" research. The market being all the teams that develop and developed applications that eventually are supposed to be deployed on the new platform with the addition of the teams that are involved in the development of the platform. The research will consist of interviews with a number of different roles within of the customers of the platform, consisting of a couple stakeholders that are happy to migrate their applications to the new platform, but also a couple of teams that are sceptical about the possibilities of the new platform. In this way we can get a clear picture of their expectations. The stakeholders will be asked to rank the most important factors. A score will be calculated based on their answers and their ranking.

Face-to-face interviews are preferred, this gives the opportunity to abandon the precomposed questions if an interesting opinion or an interesting factor is risen. If this is not an option due to scheduling problems, then the stakeholder in question will be interviewed via an online meeting.

Besides the "market" research, the Digital Foundation team will be interviewed. This interview gives a clear picture of the ideas the development team has about the factors that they think are crucial to the success of the platform. This interview will take place with as many members of the Digital Foundation team as possible, with this approach a discussion will be facilitated. The tendency to discuss a matter in more depth is higher with a discussion style interview rather than a one on one interview.

So in short the following people will be asked for their opinion:

- "Consumers", containing a mix of enthusiastic and sceptical teams
- Digital Foundation team
- Other important stakeholders, for instance the Central IT Organisation at NN

2.3 Cost reduction

The "cost management" factor is subsequently analysed. This is due to a couple of reasons. The first one being that this factor is mentioned the least amount of times by the different stakeholders, we will see this in Section 4.2. This could mean that it is not important, but it could also mean that it is important but overlooked. The second reason is that the factor is mentioned as a point of concern by the Digital Foundation team. In combination with the first reason, one could say that this factor is been explored insufficiently at this moment.

Cost management can mean different things. It is a given that most companies would save money if they can, given the situation and the constraints. One of these constraints could be that it is easy to accomplish without needing to redesign for instance processes or systems. Another could be that other factors do not need to suffer from cutbacks, for instance up-time does not need to decrease due to inferior but cheaper infrastructure. Both are applicable in the NN case. That is why there is chosen to try to reduce costs of computing resources. Given the future situation, a method will be presented to save on the buying of computing power. To use this method there is no need to redesign the architecture and infrastructure that is already planned. The architecture and infrastructure in the NN case will be elaborated in Section 3.2.

So cost reduction with the use of Spot instances and Auto Scaler is specifically chosen because in the NN case nothing under-laying has to be changed and with minimal impact on security, load time and up-time. Other cost saving methods will be discussed in Section 8.

3 Current and future situation

The description of the current and future situation are based on `20190118 IT Digital Foundation Cloud Strategy v1.pptx` [DF19].

3.1 Current situation

NN currently uses a platform that they developed in house. A brief description of the workflow for the deployment of a new application on the current platform is as follows, for the sake of simplicity we represent the whole development team with all of her different roles and functions by just one developer:

First the application is developed by the developer of a certain business unit (BU) the app in either Blueriq or Vue.js / Angular, which are front end development languages or tools. The application needs resources, for example a database and a connection to log analyser Splunk. The developer has to list all these dependencies that are needed. He has to puzzle together who or which team is responsible for every dependency on his list. He has to contact those teams in order to get the dependencies that are needed, this is depicted in Figure 2. So in other words for our example the developer has to contact the database-team and also the Splunk-team and ask them for the resources. Later on in this thesis we will see that real world projects are far more complicated than our example.
Finally, when all team are contacted and all dependencies are met then the application is done and released.

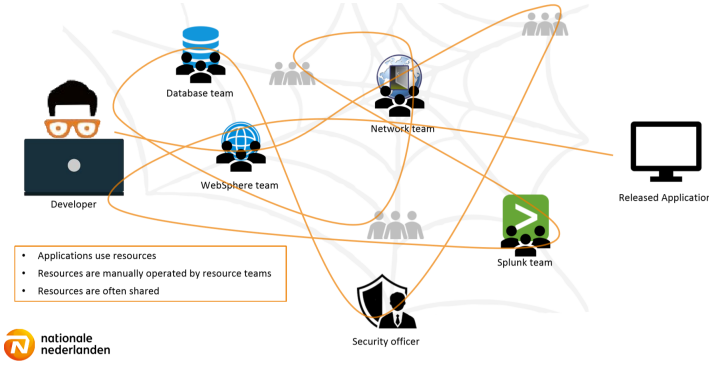

Figure 2: Current situation
2019 01 18 IT Digital Foundation Cloud Strategy v1

This "cobweb" of teams needed for the deployment of new applications is also used when releasing a new version of an existing application. Not every team is needed for every new version, only the teams that are affected by the new version. Nevertheless, for every small modification there is always another team involved.

In Section 4 we will go in depth on the problems that arise with the current platform, such as time-to-market, scalability, robustness and performance.

3.2 Future situation

3.2.1 Overview

In the future the platform will become a PaaS. The keyword is self-serviceable, meaning every customer can deploy his own application himself via a one click deployment. This is depicted in Figure 3. The context in this figure is the same as the context in Figure 2. There is a developer, that is part of given BU. The developer wants to deploy the application made by the BU. Instead of having to deal with all the different resource teams, the developer only has to deal with the interface of the pipeline. This pipeline validates the application after which the resources are automatically assigned via Infrastructure as Code (IaC). Finally, the application is automatically deployed.

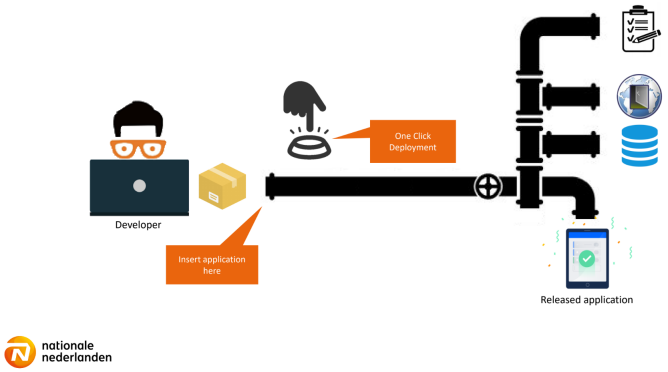

Figure 3: Future situation, deployment
2019 01 18 IT Digital Foundation Cloud Strategy v1

The one click deployment approach provides a whole range of automated checks and auto-generation. The auto-generation is done via a process called Infrastructure as Code. This process is a way to control hardware via a code based configuration. This technique is based on AWS building blocks. Every application is hosted on his own stack inside his own AWS EC2 instance with as example an Simple Storage Service (S3) bucket for storage and a database. The precise layout of the stack and of all of its additional block depends on the exact resources needed by the application. The standardised layout of the stack inside an EC2 instance is depicted in Figure 4. So every applications has at least one instance, depending on use it could be more.

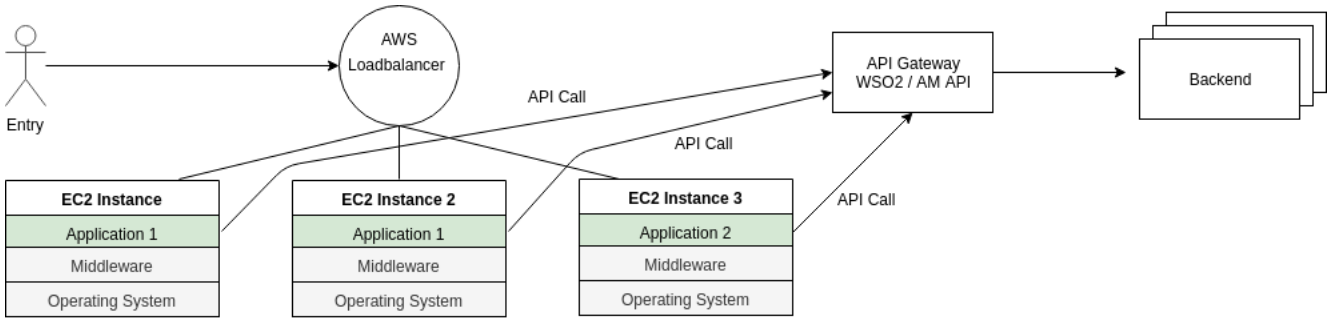

Figure 4: Future situation, simplified application stack layout

In Figure 5 is depicted which types of resources are available for the application during runtime. The pillars in this figure stand for the types of resources the PaaS can deliver.

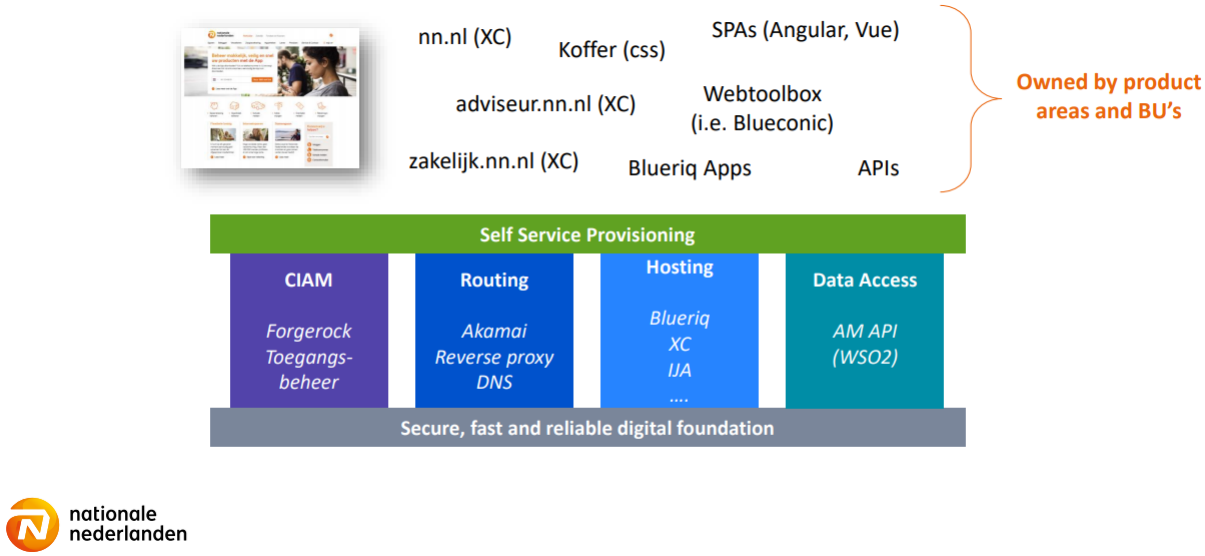

Figure 5: Future situation, provided resources
2019 01 18 IT Digital Foundation Cloud Strategy v1

3.2.2 CIAM

CIAM is short for Customer Identity Access Management. The CIAM pillar represents all the software that controls the authentication and access of the customers. In other words this collection of software manages the login of all sorts of the customers. ForgeRock is selected for the task of authentication and WSO2 is selected for the protection of Application Programming Interfaces (API's).

3.2.3 Routing

Routing is the art of getting the customer from point A to point B. What this means is that there are a lot of different applications, but for the customer the NN website looks like one single application. So routing is the collecting and stitching together of all the different applications the customer requests.

3.2.4 Hosting

Hosting is the serving of application to the customer. Hosting on this platform specifically means the serving of Java application hosting. This Java platform consists of Tomcat application server inside an EC2 instance. Tomcat will primarily serve the Content Management System (CMS) XperienCentral (CMS), Blueriq and Ibis applications, which are all front end applications.

3.2.5 Data Access

Data Access means the ability for the front end applications on the platform to make calls to back end systems. These API calls will be routed through either AM API or WSO2, to provide a standardised way of making calls.

3.2.6 Other resources

There are some supplementary resources that are provided by the platform but are not resources for the applications. These resources are available for the developers of the applications. These resources are at the moment of writing not finalised, additions can be made in the future.

- **Documentation & Guidelines**, are documents for the development teams that state how to use the platform and all of its resources.
- **Reusable components**, are functions or snippets of code that are regularly used. In this way these snippets only have to be made once. So in other words this is a library that can be used by the applications.
- **Tests**, are automated checklists that test if the application is in line with the guidelines and if the application works as supposed to.
- **Customer support**, is additional support on top of Documentation & Guidelines on a question and answer basis. This resource contain elements as a Frequently Asked Questions (FAQ) and a Slack channel.

4 Critical Success Factors

Part one of the research contains interviews with a lot of different stakeholders as mentioned. A transcription of every interview with summarised answers can be found in Appendix A. These transcriptions are in the language that the interview was conducted in, the interviews were mainly in Dutch.

For clarity: Some people stated in their interview that the current platform can not be seen as a platform, but more as a workflow. The reason behind this is that a platform is centralised. The scattering of the different teams is essentially decentral. "A digital platform is a foundation of self-service APIs, tools, services, knowledge and support which are arranged as a compelling internal product. Autonomous delivery teams can make use of the platform to deliver product features at a higher pace, with reduced co-ordination." [Bot]. Following the given definition we can say that the current platform is not a platform. The current platform does deliver support, knowledge, services and tools as an internal product. The current platform does not deliver self-service API's also the users are not autonomous in any way because they depend on these teams that create resources for the development team.

4.1 Current problems

This section covers the results from the interview regarding the current platform. It mainly focuses on the problems this platform creates.

1. In the interview with Luc hartering it became clear that for some projects or applications there are a lot of teams needed to get every resource they need. He stated that for a project that he was working on they needed 35 teams to achieve this. These 35 teams all have their own planning regarding the tasks that they have. So if they are working on a complex task, which takes a lot if time. Then other tasks have to wait. If another team depends on the team that has the task low on their list then the depended team also has to wait. With 35 teams involved chances are that a couple of teams are dealing with this. So time-to-market is heavily dependent on the backlog of these teams. For complex applications this time-to-market can be months. All of the interviewees that come across these time-to-markets agree that it takes to long. They say that development with a long time-to-market is frustrating for both the developer and the user.

2. This decentral chain of teams also complicates the contact with the teams. The workflow in Section 3.1 does not depicts the contact with these teams for the sake of simplicity. But for a lot of teams it is difficult to find and contact the team responsible for a component that is needed. There is some asymmetry in the opinion if this process is only tedious or also complex. But it is clear that the traversal of this chain is difficult and unnecessary.

3. Due to this decentral chain of teams their are a lot of handovers. This means that when a team, called A, depends on the result of another team, called B. Then when B finishes their task there will be a transfer of the results to team A. This is called the handover. With the example of the first bullet point in mind, it is likely that in large project there are multiple handovers. These handovers consist of communication between the teams. Communication between teams

or individuals is known to be easily distorted [LLSO13]. This distortion introduces problems that never existed before. For instance there is an update in a portal needed. This update contains a change in the colour, from green to orange, of a certain button. For this example there is a given amount of teams, at least more then one, needed to deploy this update. So when the first team is done their will be a handover. The handover meeting is chaotic and last minute, so the second team thinks that instead of orange the button has to be red. After a couple of weeks and multiple teams, the button that had to be orange is red. In order to have the button changed to the correct colour the chain of teams has to be traversed again. Again with the chance that something is distorted.

4. The underlying software that serves the applications is IBM WebSphere. This Java-based application server is proprietary software [Bai08]. Besides it being proprietary, WebSphere is not the industry standard. WebSphere is one of the least used Java Application Servers available in [JAS] it is enumerated under "Other". This disuse has some significant drawbacks. Vendors obviously develop their software in such a way that as many companies can easily implement and use their software. So they will develop their software based on industry standards. In [JAS] we saw that the market share of WebSphere was marginal, in contrast to Tomcat being the absolute standard for a couple of years. A consequence for NN is that the CMS that they use, XC, has to be modified to run on WebSphere. This has consequences for the time-to-market of new versions of the CMS and thus for the life cycle management. Which can result in less flexibility, stability, security, availability, reliability etc.

5. The current platform consists of servers that have not the possibility to auto-scale the applications that are hosted on that particular server. If an application endures sudden heavy traffic then a Denial of Service is possible. This DoS is an indication of the robustness and reliability of the current infrastructure.

6. There are certain guidelines for the development of applications. For instance all the applications have to comply with the corporate identity. This identity contains elements as fonts, colours and other style elements. Due to the decentral approach there is currently little supervision. So as mentioned by a few interviewees there is no supervisor that enforces the compliance with these guidelines.

4.2 Discovered CSF's

4.2.1 Overview

This section will cover the results of the interviews in combination with the operating model and requirements types. The operating model consists of 3 types, People, Process and Technology based on [MWHK17]. The requirement types are Functional requirements, Quality (non-functional) requirements and constraints based on [RR14]. The column "Mentioned by" is the percentage of all interviews in which the factor is mentioned. The column "Score" depicts a score based on the importance of the factor based on opinions of the stakeholders. In every interview every factor can get 5 different values, 0 if the factor is not mentioned or if the stakeholder explicitly said that the factor is not important. 0.5 if the factor is mentioned but not prioritised, 1 if the factor is mentioned ranking third in the ranking question. 1.5 if the factor is mentioned ranking second in the same question. And 2 if the factor is mentioned ranking first in the same question. If this ranking question is not asked the the score is based on frequency of mentioning the factors in the interviews, more about the factor scoring system can be found in Section 6 and the score per CSF per interviewee can be found in Appendix B.
Formal representation of the Score:

$$Score_k = \sum_{i=0}^{n-1}(S_i) \tag{1}$$

Where k is the index number of the factor, n is the amount of interviews ($n = 13$), S is the score as mentioned with $\{0,0.5,1,1.5,2\}$. The minimum total Score for a factor is obviously 0. The maximum total Score is 26, this is calculated by the maximum score per interview times the amount of interviews.

Name (Factor k)	**Operating type**	**Requirement**	**Mentioned by (%)**	**Score**
Automatic self deployment	Technology	Functional	92.31	14.5
Customer support	People	Functional	76.92	10.5
Compatibility	Technology	Quality	69.23	9.5
Load time	Technology	Quality	53.85	9
Security	Technology	Quality	46.15	7.5
Up-time	Technology	Quality	69.23	7.5
Feature alignment	People	Quality	61.54	6.5
Usability	Technology	Quality	76.92	6.5
Innovativeness	Technology	Quality	53.85	4
Tangible guidelines	People	Quality	46.15	3.5
Flexibility	Technology	Quality	38.46	3
Automatic testing	Technology	Functional	30.77	2.5
Cost management	Process	Constraint	23.08	2.5

Table 1: Shortlist Critical Success factors discovered in interviews, sorted by Score

4.2.2 Descriptions

- **Automatic self deployment**, is the ability to use self-service to deploy new applications and update existing. This function automatically handles the majority of the technical difficulties

of deploying applications via pipelines. Because it is handled automatically the user, in this case the developer or the development team, does not have to have the exact knowledge of the infrastructure.

- **Customer support**, is the degree of support that a "customer" can receive if necessary. This has multiple sub factors containing documentation, general support and training. Documentation meaning the documents needed to understand how the platform works and how the developer has to use it. But also why certain design decisions were made. Documentation is not a replacement for training. Documentation is the possibility to review. General support meaning that when a developer has problems that he can ask them. This can contain elements as a FAQ and a Slack channel. Training meaning that there is the opportunity for developers to learn on how to use the platform and see demo's of applications.

- **Compatibility**, is the degree of compatibility with the newest and older versions of applications and other components. This is currently especially important for the CMS. As mentioned before the current version of the CMS is outdated because of the shortcomings of the current platform. So to benefit from the changes that new versions bring the new platform has to accommodate these new versions. Also in order to accommodate an easy transition from the current to the future platform, the platform has to be able to compatible with "legacy" applications. This ensures that every application can migrate without the need for a new version of that application.

- **Load time**, is the ability of an application with all of its under laying infrastructure to serve the right content to the end user in a certain amount of time. If the page takes to long the end user will quickly lose interest and will leave the page. It does not matter if the content of the page meets the demands of the end user, if the customer already left because the page took too long to load.

- **Security**, is the ability to protect all their digital resources, not limited to but including user data, infrastructures and applications, from use in a way it was not intended to. This has to be taken as broad as possible. Especially for NN which is under constant monitoring of authorities.

- **Up-time**, is the ability to serve content to the end user on a determined percentage of the time. So in other words this means that the possibility that the end user will receive the content is equal to these percentages. The percentage is part of the Service Level Agreement (SLA).

- **Feature alignment**, is the alignment of the expectations of the new platform with the features and changes that the platform actually brings. This factor depends heavily on the information and presentation that the "consumers", the application developers, have received.

- **Usability**, is the ability to satisfy the needs of the developer in order to effectively use the platform. Usability has multiple sub factors, being ease-of-use, design components and speed of the deployment. Ease-of-use meaning to which degree the developer can use the given dashboard or pipeline without being stuck on something trivial. Design components meaning the design of the individual components that make up the platform, these components have

to be flexible to work with all the applications that will be placed on the platform. Speed of deployment meaning the overall speed of the process, from reading the documentation to the automated deployment. This speed indicates also how easy to use the platform is but also the speed of the underlying code and infrastructure.

- **Innovativeness**, is the degree to which legacy structures or applications are replaced with structures that offer a certain advantage, for instance efficiency, security or ease of use. Advantages that were not possible with the legacy setup, or with every other legacy setup.
- **Tangible guidelines**, is the ability to formulate a set of guidelines and best-practises that are easily understandable. These guidelines and best-practises are meant for the applications that will be deployed on the platform.
- **Flexibility**, is the ability to transform the platform when necessary. The platform that is being built is based on current industry standards. Standards change overtime when innovative software is developed. The flexibility is the degree to which the platform can follow those new standards without the need to build it from scratch.
- **Automatic testing**, is an extra ability inside the automatic self deployment. This function ensures that the application operates as it is supposed to and that the application follows the guidelines. This function tests the application inside self deployment but prior to the assignment of resources.
- **Cost management**, is the constraint of the set budget. Obviously every project need to operate in a certain budget to become a success. In this case there is not a hard numerical limit, but it is soft "within reason" budget.

4.3 CSF's and future situation: recommendations

To get a better overview about the new platform we will make a comparison between the mentioned factors in Section 4.2 and the vision of the future situation described in Section 3.2. This will be done for each mentioned CSF and is based on the interviews, the description of the future situation, the (potential) problems with the current platform and relevant documents.

- **Automatic deployment**, is the base idea of the future platform. As mentioned in Section 3.2.1 the keyword is self-service. The under laying technology to achieve a platform that is usable in real-time is Automatic deployment.
- **Customer support**, is represented in the future situation by "Documentation & Guidelines" and "Customer support". Both of these resources will enable the developers and development teams to develop applications, that are fully compliant with the guidelines and that integrate all the needed components of the platform. The exact elaboration of these documents and channels will determine the satisfaction the developers, because the given setup corresponds with the opinions of the interviewees.
- **Compatibility**, is primarily connected to the transformation from WebSphere to Tomcat. As mentioned in one of the problems in Section 4.1. The use of the industry standards as Tomcat will increase the compatibility. The vendors are more likely to develop their products

based on these standards. But could also mean that "legacy" application that are optimised for WebSphere are not compatible with standard Tomcat.

- **Load time**, is only partly effected. The new platform only effects the front end and its underlying stack. With the transition to the cloud and the description of its infrastructure, we can conclude that the performance is more scalable than an on-premise solution. The load time is heavily dependent on the performance of all the systems involved, so if only the performance of the front end is improved then the total load time will be reduced.

- **Security**, is according to interviewees already up to par. This remains a critical factor in the future integrating a new Identity Access Management, replacing proprietary WebSphere solutions with industry standards.

- **Up-time**, has with the featured design, containing cloud solutions, auto-scaling and an application per instance, a higher chance of being high-available.

- **Feature alignment**, the overall connection between the expectations between both parties are well aligned. In the interviews it became clear that every interviewed stakeholder knew what the purpose is of the new platform and what the technological advances compared to the current platform.

- **Usability**, contains multiple sub factors. The general plan and design in Section 3.2.1 is aligned with the needs of the interviewed stakeholders. The usability will depend on the elaboration of the sub factors presented Section 4.2.2. Use of a general library, in Section 3.2 "Other resources" called "Reusable components", will assist the developers on top of the mentioned sub factors.

- **Innovativeness**, in the interview with Leon Kortekaas, who is among other things responsible for the AWS for the central ICT department, it became clear that, in his opinion, the future platform is not state-of-the-art. There are components that can be replaced with other more innovative counterparts. In his opinion this is not a problem, the future platform is already a big step forward compared to the current platform. He also said that with the current legacy it is not possible to transition to the most state-of-the-art systems, and that the future platform is the basis to catch up with the trends. Recommendation: continue the current development of the new platform and iterate to new components and industry standards.

- **Tangible guidelines**, depends heavily on the elaboration of these guidelines and therefore can not be compared with the plan to make them. Recommendation: to study existing guidelines that are known for their usability, use that knowledge to elaborate existing guidelines.

- **Flexibility**, in [DF19] there is mentioned that the amount of custom code is minimised. This means that the code that is needed to connect all the components that make up the platform is minimal. So these components are on the platform side easily interchangeable with other solutions. On the side of the application this depends on the technology that is in use and what the new industry standard is. If the new standard performs the same high level tasks, for instance serving Java web applications, these components can be changed with fairly little complexity. Recommendation: the new platform has to keep up with industry standards.

- **Automatic testing**, is represented and is one-on-one with the tests in Section 3.2 in section "Other resources".
- **Cost management**, in the future situation there is not an explicit strategy formulated for cost management. Although, it is a consensus that a cloud solution is generally more cost-effective than an on-premise solution [BKB12]. The comments from the DF team that this factor needs more attention, in combination with the lowest "mentioned by" percentage and the lowest score, makes clear that this factor is overlooked. That is why this factor is chosen to be analysed further.

5 Hosting cost reduction with Spot Instances

5.1 Overview

Cost management is a broad term. This term incorporates in this context everything that is associated with the costs to build and maintain this project. Cost reduction is a subset of cost management and is chosen because in the interview with the Digital Foundation (DF) team it became clear that cost reduction was the initial factor to transition to the cloud. In the current state of development it is unclear if this transition is still believed to reduce costs. It is clear that the costs do not matter in the current state, this can become a problem in the future when the platform is to expensive to maintain or develop further. This in combination with the general consensus that a lot of ICT projects fail due to their costs is the foundation for the second part of this research. Spot instances are chosen to try to reduce the costs because the use of them does not require a redesign of the stack or infrastructure. Other possible reduction methods discussed in Section 8 require some sort of redesign.

5.2 NN constraints

The future situation outlined in Section 3.2 in combination with the results in Section 4.2 gives us some insight into the constraints. The platform will mainly serve front end applications. If we go back to the interview with Leon Kortekaas, we will see that a couple of CSF's are self-evident. These are self-evident for the stakeholders because the end-user, the person that pays NN for a certain product, in explicitly demands them. The factors are up-time or reliability, load time or performance and security. These factors will be the constraints for the cost saving methods in this Section. On top of these constraints we will only focus on methods in a single region. Although, using spot instances across different regions can give more favourable prices we are bound to the NN case. NN has selected Frankfurt (eu-central-1) as their region, this is based on latency but also on legal restraints, so we will stick with this region for this Section.

5.3 EC2 functionality

5.3.1 Overview

In this Section we will give a overview about AWS EC2 functionality, after some general information we will go in-depth about AWS EC2 spot instances. In Section 5 we will use this knowledge to reduce costs of the future platform. AWS offers compute capacity in the cloud called EC2. This compute capacity is available in multiple different hardware configurations called instance types. These instances can be rented in different geographical locations called regions. Every region has one or multiple data centers, every data center is a so called availability zone (AZ) in a particular region. For instance, region Frankfurt (eu-central-1) has three availibility zones eu-central-1a, eu-central-1b and eu-central-1c. AWS offers four different buying options for instances, being on-demand, spot, reserved instances and dedicated. Buying options means that the hardware that is rented is the same it only differs in duration and costs. The reserved option is comparable with a standard Virtual Private Server, being renting hardware for a longer period of time being months or years. On-demand instances are rented per hour. Dedicated is the same as traditional dedicated servers, being a whole server only to be used by the renter. Spot instances are the same as on-demand,

being rented per hour. The renting of spot instances is done via a bidding system, this can mean a discount up to 90 %. If somebody wants to buy a spot instance, they place a bid on the instance. If this bid is above the current spot price, then the buyer will receive the instance. Else the buyer will get the instance when the spot price drops below the bid. The right of use differs because when the price of the spot instance rises above the bid from the buyer after receiving the instance, then the instance is terminated [AWSd]. So in other words the buyer bid on a spot instance, the bid was accepted because the market price was below the bid. Then the buyer received the instance, but the market price rose above the bid on the spot instance that is currently in use by the buyer. When this happens the instance is terminated in order to sell the instance to a buyer that has a bid higher than the first buyer. Since 2015 the buyer will get a notification 2 minutes prior to the termination. This notification enables applications to save their current status and prepare them self to be shut down [Bar15].

5.3.2 Spot instances without bidding

As of November 2017 it is possible to buy spot instances without a bid. This is currently the default setting in a spot request, if somebody wants to place a bid via the dashboard they have to uncheck "Apply default" in "Additional request details" and thereafter has to check "Set your max price". The buyer still pays the market price, but the instance will be terminated if the market price exceeds the on-demand price. This seems rather strange, because if every bidder on the market is rational then there would be a maximum on the spot price exactly at the on-demand price. Why would anyone buy a spot price with the uncertainty that spot instances bring, more specifically the possibility of termination at every given moment, for a higher price than an on-demand instance without that uncertainty.

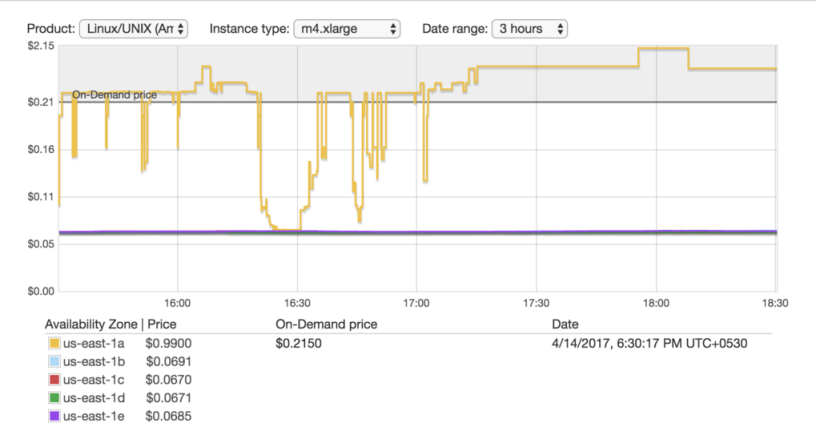

Figure 6: Spot prices reach `10X` the price of on-demand instances (14-04-2017)[AWSf]

There are multiple explanations for this phenomenon:

- The first explanation is that the market price is not related to the bids from the buyers but rely on an algorithm. For instance the authors of [ABYBYST13] proposed that the market price is not correlated with the bidding but rather based on an `AR(1)` function with at least a lower bound and the on-demand price as normalising factor. This basically means that the price is random and not based on any bids. As of March 2018 Amazon introduced a new price

model which is based on long term supply and demand [AWSb]. So we can safely say that an `AR(1)` algorithm is not responsible for this phenomenon.

- Another option, which given the previous option, is more likely to cause this kind of behaviour. This behaviour is caused by the bidding strategy of other buyers. A spot price above the on-demand price suggests that there are bidders above this on-demand price [AWSf]. This maximises the chance that those bidders will have the highest bids, because they are well above on-demand prices, so the chance that they will lose their instance is minimal. This is actually one of the strategies that we will discuss in Section 5.4.

- The last option is that the total amount of available spot instance in that region is equal to zero and Amazon needs to terminate all spot instances to accommodate the demand for on-demand instances and raises the spot price above all bids.

5.3.3 Current situation

Figure 6 is an incident in 2017, this means that this happened before the new pricing model mentioned in Section 5.3.2. It is unknown if this new pricing model has effect on this phenomenon. In a little research done on the data on all the instance types in AWS region Frankfurt (EU-Central-1), with date range set to the max being 3 months, the spot price never exceeded the on-demand price. On a couple of instance types it reached the on-demand price (Figure 7) but it did not exceed it. This only suggests that the default setting, being the max price set to on-demand, is widely used. There is no proof for this theory.

5.4 Plain overbidding

5.4.1 Overview

This method relies on the information given in Section 5.3.2. So to summarise, it is unlikely for somebody without the proper knowledge to bid close to or far above the on-demand price of a certain spot instance. This "irrational" bidding strategy can actually save money. The opportunity lies in the fact that the bidder does not pays its bidding price for a spot instance but the market value of this instance [Tai13]. The instance is terminated when the bid rises above the bid. So if a bidder places a trivially high bid, then the chance that an instance is lost due to rising market prices is close to zero, the savings depend on the market price across the whole period. There are a couple of limiting factors that need mentioning:

- **Spot request limit**, there is a limit on the amount of requests one can make. It is unclear what the exact number of requests are, because this differs per Amazon customer. The number depends on their AWS activity. In the NN case we can assume that the maximum number is rather high or at least has the potential to become rather high, because NN is a rather large company. If in the exceptional case that the limit is too low, a limit increase can be requested [AWSe].

- **Capacity not available**, this error is given by AWS when the bid is above the market price but there is no or not enough resources available to launch this spot instance [AWSc]. There is indication beforehand that an instance cannot be launched due to capacity shortages. This

error is also the only possibility that a spot instance is suddenly terminated. The market price is not always an indication that there is a resource shortage, because we saw in Section 5.3.2 that these prices are based on long term supply and demand. The shortage can be temporarily and therefore not visible in the pricing. The way to mitigate this factor is to evade instance types with a lot of visible fluctuations and also to spread spot requests and spot instances across multiple availability zones and instance types.

5.4.2 Possible uses

Plain overbidding could be used as base for other strategies or as bidding strategy for a spot fleet or single instances. The only downside to this method is long term above on-demand prices. If one places a bid that is a lot higher than the on-demand price there is a chance that something close to this bid has to be paid at some point. Given that the AWS EC2 spot price model in Section 5.3.2 is based on long term demand and supply this is unlikely but still possible.

5.5 Bid management system

5.5.1 Overview

This method relies an on-demand instance as core. This core divides the work that has to be done and handles spot instances. The core places bids and buys spot instances when the price is correct. The definition of the correct price depends on the used algorithm. Some algorithms will buy on-demand instances if the price is of spot instances is not correct, this is especially true for tasks that have to be performed right away or tasks that always have to be online. This ensures that the SLA is met for those applications. [PRB14], [CLT14], [LLZ15], [Tai13], [MHL+14] and[TYWL13] propose a framework varying from a complete system with code for the core node to only bidding strategies. We will evaluate them to find the best system given the constraints in the case of NN. Note that all of these papers are made before all of the changes mentioned in Section 5.3. So with the 2 minute termination notice, some of these algorithms can be improved and so downtime can be reduced.

MaxMe [CLT14], is an heuristic price-aware and auto-scaling algorithm with build-in migration algorithm. This algorithm is designed to maintain an Hadoop cluster in the cloud. This algorithm consist of 3 types of nodes: master nodes, essential slaves and non-essential slaves. The master nodes are the core as mentioned above, the essential slaves are the compute and data nodes exclusively running on on-demand instances and non-essential nodes also serve as non-essential data and compute nodes but run on either on-demand or spot instances. The general idea behind this algorithm is that these non-essential slaves can be terminated at any given time, even without the 2 minute notice, and this would not endanger the integrity of the Hadoop cluster.

Translated to the NN case: This algorithm of a core and essential and non essential slaves can be used to provide a platform for front end applications. The core would run the scheduler and data transfer. The essential slaves would serve the application to the end-user and would scale according to the load. The non-essential nodes would perform background tasks. These background tasks differ per application. A possible task is provide a caching layer for the essential slaves. Another option is that scaling is done via non-essential nodes. So when more resources are needed, then

spot instances are acquired to cope with the higher load.

RAMP [LLZ15], is an algorithm that uses a reliability function to determine the probability that an instance is available for the duration that the instance is needed based on historical data. The algorithm is designed to buy spot instances to resell them for a percentage of the on-demand price to other clients. In this way the maintainer of the system is driven to buy in the spot instance that maximises his profit while maximising the reliability for his customer.

Translated to the NN case: This algorithm seems rather trivial compared to the proposed strategy in Section 5.4. RAMP does not require maximum reliability, where the NN case does require that. We saw that to achieve maximum reliability one can bid a trivially high number, this does not maximise the savings but will still save significantly compared to on-demand instances.

ODB [PRB14], is an algorithm that incorporates a scheduler. This scheduler calculates the critical path for the workflow and tries to minimise the execution cost. It uses a combination of spot and on-demand instances. To accomplish this the algorithm employs a minimising bidding strategy and just in-time scheduling. This algorithm is designed for the processing of workflows. To mitigate the chance that a spot instance can be terminated the algorithm uses checkpointing. Checkpointing is a mechanism that makes periodical snapshots, in order to have a fallback in case of a termination.

Translated to the NN case: This algorithm can be used for background tasks. For a front end application there is no critical path to calculate because this application is not a workflow but a continuous process. So there is no end node in the critical path. Some background tasks have an end node, for instance the generation of a report. The paper does not mention the time it takes to make checkpoint. If this is possible between the 2 minute notice and the termination then the checkpointing does not have to be periodically.

AMAZING [TYWL13], is an algorithm that is aimed towards divisible applications under an SLA constraint. This means that the workload is divided between nodes and has to be performed with a maximum execution delay. The algorithm uses a predictive bidding system based on spot price models constructed on historical data. These models consist of two rules, being bidding 0 or bidding the maximal amount.

Translated to the NN case: This algorithm uses a bidding strategy that is a subset of the strategy discussed in Section 5.4. In the paper the assumption is made that the market price of spot instances will never exceed the on-demand price. We saw in Section 5.3.2 that the market price actually can exceed this barrier. This system can still be used, possible with a minor adjustment in the bidding strategy, for the processing of background tasks with time constraints.

5.5.2 Possible uses

The possible uses of these algorithms vary between the systems. The part of the systems where scaling is constructed with on-demand or spot instances, this part is largely replaced by AWS Auto Scaling. The method to use spot instances with Auto Scaling will be elaborated in Section 5.6. The

use of a scheduler that processes workloads with the use of spot instances can still be relevant for background tasks including but not limited to periodic tasks and onetime tasks.

5.6 Autoscaling

5.6.1 Overview

The new Auto Scaling Group Launch Templates have the option to launch not only multiple instance types but also the option of multiple purchasing types. So with these Launch Templates an Auto Scaling group can be created that consists of both on-demand instances as well as spot instances. In combination with the option to use multiple instance types, with obviously different spot prices, this function can be a method for cost reduction. If we go back to the constraints the reliability is not trivial to accomplish with spot instances due to their volatile behaviour in terms of termination of instances.

The constraint of up-time is in the base case for this method is fairly easy to accomplish. The base case being the minimum size of the Auto Scaling group, in other words the base case is that the group endures a small load. This base case is always present because it is the minimum size of instances in the group. If we would use spot instances for this case, then there is a chance that these instances can get terminated at some point. To maximise the reliability of the base case the instances there have to be purchased as reserved instances. This purchasing type provides a capacity reservation and can also offer up to 70% discount. This does not mean that this type provides better reliability than on-demand instances, but the base case is always present so the reserved instances provide a better value and the reserved instances guarantee computing power at all time [AWS18a].

Auto Scaling provides a solution to scale the resources when the load on the instances changes. So in the base case, a couple (or one) of on-demand or reserved instances are running, when traffic or the load is high and the resources are nearly drained the Auto Scaler will upscale the resources automatically in the form of more instances. With Dynamic Scaling these instance will run until they are unnecessary. This basically means that their run time is limit, it depends on the application and the end-users how limited it exactly is. In the announcement of the new spot pricing model Amazon promises that the prices are more predictable and update less. Even in the most extreme example we could find (Figure 7) the climbing is across multiple days. So in the circumstances that spot instances are only needed to cope with heavy traffic for certain hours of certain days, then this climbing is only an issue when new spot instance need to be acquired and the maximum price is lower than the on-demand prices. If this is the case then a diversification strategy can be chosen to launch other spot instance types, types that are not that expensive with comparable resources.

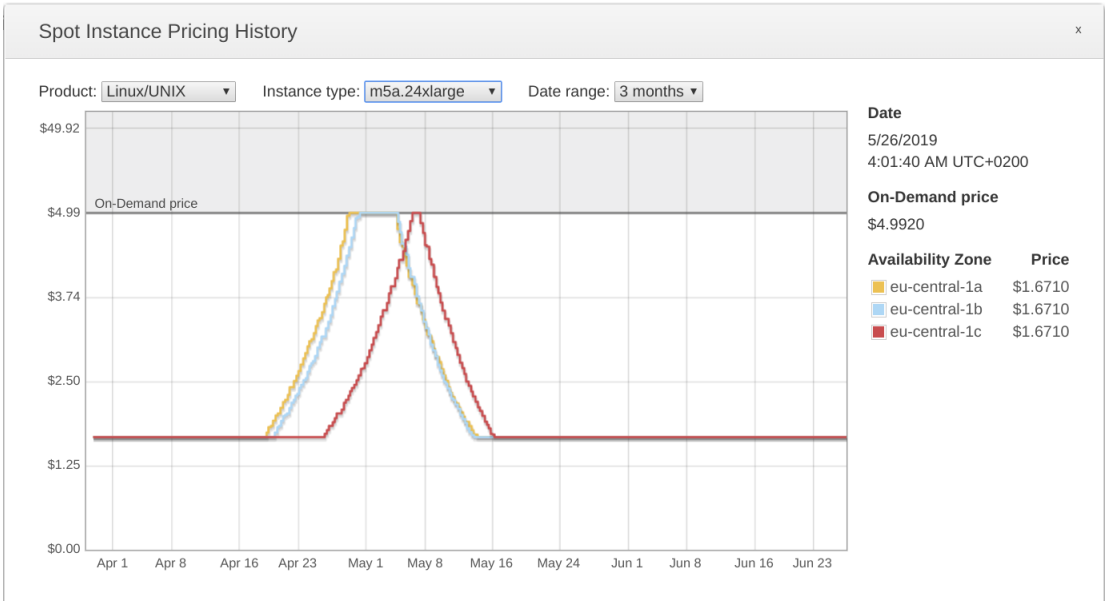

Figure 7: Spot pricing m5a.24xlarge 30-03-2019 till 26-06-2019, `eu-central-1`
AWS Spot Pricing History

5.6.2 Possible uses

This method is for front end applications that have certain peaks in their usage. These applications need to be reliable but also scale in and out when needed to cope with these peaks. This method does not save money on the base case, being on-demand or reserved instances, but on the processing of the peaks using spot instances. This method can save up to 90% of computing costs for peak hours [AWS18b]. The chance that the spot instances are acquired depends heavily on the bidding strategy. Also the reliability of the spot instance depend on said bidding strategy as well as the strategy with unavailable capacity.

5.7 Proposed method

The mentioned methods all have the potential to save significantly on the computing costs. But all solutions have problems in the NN case. We propose a combination of the three mentioned methods. The basis for this method is the Auto Scaler mentioned in Section 5.6. We saw that this method is easily usable in the case. This method can be improved if the other methods are used in coöperation with the Auto Scaler.

So to summarise the Auto Scaler method, the idea is that there is a base case that depends on on-demand or reserved instances. When the load is draining the resources of this base then the Auto Scaler will acquire spot instances. A potential problem is that spot instances can not be acquired due to capacity unavailability. The first mitigation for this problem is that when a spot instance is needed, the Availability zone with the least amount of running instances in the group is selected. If this zone has not adequate resources, the request will fail and Auto Scaler will select another Availability Zone [AWSa]. As an extra safety measure the “ValidUntil” parameter is set to a very low value, this assures that the request is cancelled as soon as the capacity unavailability

error is given [Ava]. The second mitigation of this problem is the diversification of instances used. If an instance is not available or more expensive than other comparable instances, it is a best practise to select as many instance types as possible in the Fleet Selection over as many Spot Pools. This practise maximises the chance that an spot instance is acquired. AWS automatically selects the cheapest instance first. This automatically maximises the total discount. The third option will be discussed in the paragraph about the added agent

We combine this Auto Scaling with the overbidding method as bidding strategy mentioned in Section 5.4 instead of using the default settings. The default settings meaning that the bid is equal to the on-demand price. So we place a trivially high bid on every spot instance. This minimises the chance that the instance is terminated due to a rising market price and maximises the chance of acquiring the spot instance.

For extra security we combine this setup with an algorithm. This algorithm acts as a controlling agent. There are two possible versions of this agent.
In the first version the agent only monitors for the situation that all instance type that are selected are more expensive than the on-demand buying option. This option suggests that the overbidding method is widely used and exhausting all the spot instances available, therefore the long-term supply and demand on which AWS bases its prices can exceed the on-demand price. The agent will notice this and update the Auto Scaling Group settings to use on-demand instances.
The second version is the first version with an added functionality. This functionality is the third option to mitigate capacity unavailability. The capacity unavailability suggests that this particular spot instance type is popular. Popularity in the long run given the current spot price model means that the price fluctuates or is always high. The latter is not an issue due to the already mentioned fact that Auto Scaler picks the cheapest options. The agent can periodically check all instances types that are suitable for the application and select the N instance types that least fluctuate over time. These instances are set in the Auto Scaling Group settings to be used by Auto Scaler. A pseudo-code for the second version of the agent is given.

Algorithm 1 Controlling agent second version

```
1: procedure MONITORING
2:     instanceTypes[] ← all suitable types
3:     fluctuation['instanceTypes']
4:     for All types in instanceTypes[] do
5:         for All Availability Zones do
6:             for All historical dates + 1 do
7:                 temp ← spotPrice(type, AZ, currentDate) - spotPrice(type, AZ, previousDate)
8:                 fluctuation['type']+ = |temp|
9:     *Update Group settings with N lowest value of fluctuations['type']*
10:    //the following lines also represent the first version of the agent
11:    seclectedTypes[] ← all instance types selected in Auto Scaling Group
12:    exceedingTypes[]
13:    for All types in selectedTypes[] do
14:        if ondemandPrice(type, AZ) < spotPrice(type, AZ, currentDate) then
15:            Add type to exceedingTypes
16:    if selectedTypes[] == exceedingTypes[] then
17:        *Update Auto Scaling group to use on-demand instances*
18:    else
19:        *Update Auto Scaling group to use Spot instances*
```

6 Discussion

6.1 CSF identification

The score presented in Section 4.2.1 is a way to make something abstract as interviews tangible. In this translation there are some things to keep in mind.

The allocation of the score per factor per interviewee is in some cases influenced by subjectivity. Sometimes the ranking question, being "Can you order the CSF's in a top 3?", is not asked. Then the score is based on the frequency of the mentioning of the factors throughout the interview. Sometimes the interviewee did not answer with a complete ranking. In this case the scoring is based on both the answer on the ranking question as far as possible but also based on frequency of the mentioning of the factors for the rest of the ranking.

The answers are subject to the comparison with the current platform. This means that for some factors the score is relatively low because stakeholders find these factors non-negotiable or self-evident. These factors are security, up-time, performance / load time. Some of these factors are already "accomplished" on the current platform and therefore the stakeholders already assume that these will also be accomplished on the future platform.

6.2 Hosting cost reduction

The effectiveness of the proposed method in Section 5.7 if used long-term is a point of discussion. It is well known that the AWS platform is volatile and in current development. It is possible that if AWS innovates or changes some aspects that the proposed method becomes unusable.

7 Related Work

7.1 CSF identification

With regards to the critical success factor enumeration there is no public related work that researches CSF's in this kind of corporate platforms. It is likely that this kind of research is done but not not published due to competitive advantages.

7.2 Hosting cost reduction

With regards to the cost reduction via the use of spot instances there is a lot of research done on this topic, although none of these papers cover the NN case with their constraints. The algorithms or methods proposed in [MHL+14], [Tai13], [LLZ15], [CLT14], [PRB14] and [TYWL13] are optimised for workloads that are in a way distributive over multiple instances. These workflows are for example Hadoop clusters. Some of these papers only work with fault-tolerant workflows while some use redundancy and snapshots to mitigate the possible loss of data.

8 Further Research

8.1 CSF identification

The CSF's found in this thesis could be researched further. For every factor found there could be researched how to achieve this factor in the most effective way with the use of literature, prototyping, interviews etc.

8.2 Hosting cost reduction

Further research can be done on the possibility of saving costs with reserved instances. Reserved instances is the only buying option where instances are bought and paid upfront. This option can save up to 70% of the costs. The exact saving depends on the type of instance that is bought, it depends on the term (one or three years) and it depends on the part of paid upfront (none, partial or all upfront). This buying option trades the flexibility of on-demand for reduced costs. Without the chance that an instance is suddenly terminated.

Further Research can be done on cost saving strategies for companies with a comparable IT landscape as NN. For instance in the future situation every application gets his own EC2 instance. Meaning every instance has his own stack containing an Operating System (OS) and middleware. The application is placed on top of this stack in his own EC2 instance. If multiple applications can be placed in the same instance, this will save the resources that are used by the stack. The number of application per instance has to be balanced with the type of instance. So in other words, is there a cost saving possible if an instance with more resources with multiple applications is used regarding multiple instances with less resources, but with one application per instance. This method is known as containerisation.

Finally, further research can be done on replacing EC2 with some other form of said containerisation for example with EKS, which incorporates Kubernetes. The level on which virtualisation is introduced is higher than with EC2. With EC2 the virtualisation is between the hardware and the virtual machines (instances) with each having its own operating system. With EKS the virtualisation is between the operating system which incorporates a container runtime and the containers. These containers do not have the overhead of having their own operating system but make use of the under laying container runtime. Without this overhead each server can dedicate more resources to the applications and therefore have the potential to save significantly on costs.

9 Conclusions

9.1 CSF identification

In this thesis we first interviewed stakeholders for their opinion on what the Critical Success Factors are in creation of an internal Platform as a Service. We can conclude that the vast majority of these stakeholders are content with the sketched future situation. We evaluated the CSF's that where obtained from the stakeholders and made a comparison with the sketched future situation. We saw that the future situation is aligned with the CSF's and thus with the expectations of the different stakeholders.

9.2 Hosting cost reduction

We saw that cost management was not mentioned extensively, while the Digital Foundation team themselves saw the factor overlooked. This factor was chosen and there was chosen to try to reduce the costs. In the current state of development of the new platform it is easier to reduce costs without the need to redesign the architecture and infrastructure. So we evaluated a couple methods to reduce costs with the use of spot instances. All methods had their drawbacks when applied to the NN case, some of them where even outdated. A method was proposed that combines the three other methods. This combination can mitigate all the individual drawbacks, in order to achieve a stable platform for front end applications with a significant cost reduction. The exact percentage depends on the use of the application, the instance types used, the amount of spot instances used, the duration of the usage of the instances and the on-demand price of all instance types involved.

The only thing is to build this method and test it alongside the planned setup. Since this thesis is only a research on the PaaS and how to improve one of the mentioned CSF's, it is not possible to give certainty on the stability and the realised discount in comparison to the planned system.

References

[ABYBYST13] Orna Agmon Ben-Yehuda, Muli Ben-Yehuda, Assaf Schuster, and Dan Tsafrir. Deconstructing amazon ec2 spot instance pricing. *ACM Transactions on Economics and Computation*, 1(3):16, 2013.

[Ava] Autoscaling makes a bad zone choice (zone with unavailable capacity). `https://forums.aws.amazon.com/thread.jspa?threadID=117650`. Accessed: 27-06-2019.

[AWSa] Benefits of auto scaling. `https://docs.aws.amazon.com/autoscaling/ec2/userguide/auto-scaling-benefits.html`. Accessed: 27-06-2019.

[AWSb] New amazon ec2 spot pricing model: Simplified purchasing without bidding and fewer interruptions. `https://aws.amazon.com/blogs/compute/new-amazon-ec2-spot-pricing/`. Accessed: 22-06-2019.

[AWSc] Spot instance interruptions. `https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/spot-interruptions.html`. Accessed: 27-06-2019.

[AWSd] Spot instances. `https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-spot-instances.html`. Accessed: 22-06-2019.

[AWSe] Spot request limits. `https://docs.aws.amazon.com/whitepapers/latest/cost-optimization-leveraging-ec2-spot-instances/spot-request-limits.html`. Accessed: 27-06-2019.

[AWSf] Why is the aws ec2's spot price greater than the on-demand price? `https://devops.stackexchange.com/questions/893/why-is-the-aws-ec2s-spot-price-greater-than-the-on-demand-price`. Accessed: 23-06-2019.

[AWS18a] Cost optimization pillar. `https://d1.awsstatic.com/whitepapers/architecture/AWS-Cost-Optimization-Pillar.pdf`, 2018. Accessed: 24-06-2019.

[AWS18b] New – ec2 auto scaling groups with multiple instance types purchase options. `https://aws.amazon.com/blogs/aws/new-ec2-auto-scaling-groups-with-multiple-instance-types-purchase-options/`, 2018. Accessed: 24-06-2019.

[Bai08] Stacy Avery Baird. The heterogeneous world of proprietary and open-source software. In *Proceedings of the 2Nd International Conference on Theory and Practice of Electronic Governance*, ICEGOV '08, pages 232–238, New York, NY, USA, 2008. ACM. URL: `http://doi.acm.org/10.1145/1509096.1509143`, `doi:10.1145/1509096.1509143`.

[Bar15] Jeff Bar. `https://aws.amazon.com/blogs/aws/new-ec2-spot-instance-termination-notices/`, 2015. Accessed: 22-06-2019.

[BKB12] S. Bibi, D. Katsaros, and P. Bozanis. Business application acquisition: On-premise or saas-based solutions? *IEEE Software*, 29(03):86–93, may 2012. doi:10.1109/MS.2011.119.

[Bot] Evan Bottcher. What i talk about when i talk about platforms. https://martinfowler.com/articles/talk-about-platforms.html. Accessed: 05-08-2019.

[Cha09] David Chappell. Windows azure and isvs. *A Guide for Decision Makers*, 2009.

[CLT14] Changbing Chen, Bu Sung Lee, and Xueyan Tang. Improving hadoop monetary efficiency in the cloud using spot instances. In *2014 IEEE 6th International Conference on Cloud Computing Technology and Science*, pages 312–319. IEEE, 2014.

[DEF] What is critical success factors (csf)? http://www.businessdictionary.com/definition/critical-success-factors-CSF.html. Accessed: 04-06-2019.

[DF19] Team Digital Foundation. *20190118 IT Digital Foundation Cloud Strategy v1*. Nationale-Nederlanden, Internal Communication, 2019.

[JAS] Most popular java application servers: 2017 edition. https://plumbr.io/blog/java/most-popular-java-application-servers-2017-edition. Accessed: 16-06-2019.

[LLSO13] Mark E. Laidre, Alex Lamb, Susanne Shultz, and Megan Olsen. Making sense of information in noisy networks: Human communication, gossip, and distortion.(report). *Journal of Theoretical Biology*, 317, 2013.

[LLZ15] Luke M Leslie, Young Choon Lee, and Albert Y Zomaya. Ramp: reliability-aware elastic instance provisioning for profit maximization. *The Journal of Supercomputing*, 71(12):4529–4554, 2015.

[MHL+14] Aniruddha Marathe, Rachel Harris, David Lowenthal, Bronis R De Supinski, Barry Rountree, and Martin Schulz. Exploiting redundancy for cost-effective, time-constrained execution of hpc applications on amazon ec2. In *Proceedings of the 23rd international symposium on High-performance parallel and distributed computing*, pages 279–290. ACM, 2014.

[MWHK17] Firas Masri, Trevor Wood-Harper, and Peter Kawalek. Systems thinking: Analysis of electronic patient records implementation and knowledge transfer practice. bp trust in the uk, nhs. *International Journal Of Management and Applied Research*, 4:105–121, 03 2017. doi:10.18646/2056.42.17-009.

[PRB14] Deepak Poola, Kotagiri Ramamohanarao, and Rajkumar Buyya. Fault-tolerant workflow scheduling using spot instances on clouds. *Procedia Computer Science*, 29:523–533, 2014.

[RR14] Suzanne Robertson and James Robertson. *Mastering the requirements process*. Addison-Wesley, 2014.

[Tai13] Moussa Taifi. Banking on decoupling: budget-driven sustainability for hpc applications on auction-based clouds. *ACM SIGOPS Operating Systems Review*, 47(2):41–50, 2013.

[TYWL13] Shaojie Tang, Jing Yuan, Cheng Wang, and Xiang-Yang Li. A framework for amazon ec2 bidding strategy under sla constraints. *IEEE Transactions on Parallel and Distributed Systems*, 25(1):2–11, 2013.

Appendices

A Interviews

DF-team (09-04-2019)

Wat vinden jullie van het huidige platform, de huidige manier van werken?
Route niet complex, het zijn procedures die veel tijd kosten maar toenemende mate van controle op deze procedures Doorlooptijd is hoog

De toenemende mate van controle is dat bij het nieuwe platform of ook al bij het huidige platform?
Beide, niet bij iedereen maar er is sprake van een toenemende mate van controle.

Wat vinden jullie van de procedures in vergelijking met Infrastructure As Code (IAC)?
IAC had ook on premise kunnen draaien, er is echter door hogerhand besloten dat IAC in combinatie met de cloud de manier van werken wordt. Ofwel deze transitie had ook in kleine stappen. Websphere is nu de manier, daar is veel kennis van. Dit kan gebruikt worden om te starten met IAC. Het zou dus een begin zijn van het afbouwen van de procedures en het automatiseren van deployments.

Want een deployment duurt ongeveer 4 weken, heb ik begrepen.
Dat ligt erg aan de afhankelijkheden en dus de hoeveelheid teams er doorlopen moeten worden. Daarnaast is het afhankelijk van de tijd die deze teams nodig hebben om hun taak te vervullen. Serveraanvraag is nu eenmalig

Hoe zit het dan met de performance van die servers?
Dat is geen probleem, de on premise servers zijn blades en afhankelijk van de applicatie en het gebruik ervan kan uitgebreid worden doordat er in die blades bijgestoken kan worden. Dat inprikken kan op dit moment niet als Digital Foundation team, iets wat met het nieuwe platform wel kan.

Dus in plaats van een AWS style serverpark wordt er nu geïnvesteerd in de cloud?
Ja, de redenatie is dat dat de core business is van Amazon waardoor we ervan uit gaan dat ze het beter voor mekaar hebben dan we het ooit zonder veel investeringen kunnen hebben. Dat automatisering kan ook stap voor stap op eigen hardware, je begint klein en werkt het langzaam uit.

Zou het hebben van een geautomatiseerd on premise serverpark jullie voorkeur hebben als de keuze opnieuw gemaakt zou worden?
De keuze is meer dan de cloud, ook de verschuiving van Websphere naar Tomcat is daar een belangrijk onderdeel van. Dit is omdat Tomcat de standaard is bij leveranciers. Maar ook omdat het algemene model veranderd is, ook ten goed van open source projecten zoals bv. Tomcat.
(Language changed on the arrival of a non-dutch speaking team member)
Open source software has the benefit that it can be easily downloaded and deployed quickly to test

the full potential. Traditional software requires a lot of contact with the vendor for a purchase of a license or a kind of trial. Downside is the lack of support, or you need to buy the support. With Tomcat the free version is chosen.

New platform is more based on open source like Tomcat does that have other benefits than cost?
Becoming more Agile and time to market are the main drivers, in the early stages costs was the driving factor. That idea was abandoned when the project was started. The cost still matter but it is a constraint.

The main goal at the moment is to be quicker (time to market)
The tendency at the moment is that the DF-Team plays a big role when a business unit wants to deploy something. So the DF-Team keeps its role but as a self-service. So everything is still centralized, but the business unit has the opportunity to deploy on their own, without the technical knowledge. Focus that teams need to deploy their own things, but it is that the services that are available will become self serviceable.

Is that a mismatch with what the team thinks?
The story that is presented is mainly focused on autonomy and adjusted to the audience. But the team thinks the same, it all are benefits that count. There is a shift in responsibility from the DF-Team to the developer. But it is not necessary for the developer to have the technical knowledge, this is hidden by the dashboards that will be created, but also the support and documentation that is being provided by the DF-Team.

Is there a factor that needs more attention?
Inside into the cost for the DF-Team. We have attention for a lot of factors but less for cost management. Cost are exploding, which makes sense because we just started. But someone will get the bill and think the project is too expensive. This person will complain and maybe eventually kill the project. A system of budget warnings is needed.

Nick Permanand (aanvulling op DF) (19-4-2019)

Wat vind je van de verschuiving van de verantwoordelijkheid van deployment van alle verschillende teams (huidige manier van werken) naar de developer zelf?
Goed, zelf ervaring bij een Business Unit, deze manier van werken geeft veel afhankelijkheden. Daar worden alle verschillende teams en mensen mee bedoeld. Het is goed dat BU's nu zelf die deployment kunnen uitvoeren zonder dat het DF-Team controle verliest over de security en de content.

Vind je dat de huidige manier van werken niet complex is maar lang duurt (net zoals de rest van het DF-Team)?
De route is niet complex als je de mensen erachter kent en ervaring hebt met de techniek die achter de schermen nodig is. Voor de meeste mensen is het dus wel complex aangezien die niet de

kennis hebben die hiervoor nodig is. Voor de anderen die de kennis wel hebben is het een afvinklijstje.

Wat vind je van de overgang van Websphere naar het open source Tomcat?
Goed om twee redenen. Ten eerste ben je door de manier van het tokensysteem leverancier gebonden aan IBM, bij Tomcat wordt overgegaan naar een meer industriestandaard, dit zorgt voor flexibiliteit in de toekomst. Ten tweede duurt de deployment van XC (CMS) te lang op Websphere, aangezien dat niet native wordt ondersteund en op Tomcat dit wel vanuit de leverancier wordt ondersteund. Bij open source zijn de kosten een pluspunt, het team kan wel inschatten of deze producten voldoen aan de standaarden.

Vind je het een nadeel dat bij open source producten geen ondersteuning is?
Dat is geen probleem.

Denk je dat de eerste insteek om te migreren naar de cloud kosten was of het zijn van een agile omgeving?
Het kostenplaatje was een factor met eerste migratie naar de cloud. De kosten zijn nu meer een constraint in plaats van een hoofdfactor, tenminste binnen het DF-Team. Binnen het DF-Team is kwaliteit en robuustheid belangrijker. Het nieuwe platform biedt wel kansen om efficiënter om met die kosten om te gaan.

Heb je de angst dat iemand van hogerhand beslist dat het project te veel geld kost en dan de stekker eruit trekt.
Nee, geen angst maar die kans bestaat er wel degelijk. Ik kan daar wel wat van vinden maar dat is niet iets waar je wat aan kan doen. Daarnaast is er wel toewijding ook hoger in de organisatie om dit project te laten slagen. Forgerock is nu ingekocht daar kun je aan zien dat er toewijding is voor dit project.

Zou je toch een indicatie willen of er binnen het budget geopereerd wordt (indicatie op een van de monitors)?
Nee, de prioriteit is nu vooral kwaliteit. Als de kosten te hoog worden kan er altijd wel geoptimaliseerd worden, echter is dit een later zorg. Besparing kan later nog bekeken worden als de kwaliteit en stabiliteit al gegarandeerd is.

Hoe vind je de connectie met jullie interne klanten?
Die connectie is nog niet nodig, nu wordt gewerkt aan de randvoorwaarden. Zoals een proxy en identity provider. Er wordt wel getoetst of dingen voldoen aan de verwachtingen van de klanten. Later in het proces zal dit gaan en moeten toenemen maar dan is er ook meer terugkoppeling met de klanten.

Is er op dit moment CSF waarvan je denkt dat die op dit moment minder aandacht krijgt dan het verdient?
Er zijn factoren die ik kan noemen, echter is dat moment logisch gezien de fase naar de ontwikkeling nu is. Echter, vind ik over het algemeen dat er binnenkort al begonnen moet worden met het uitleggen van wat het DF-Team mee bezig is. Het overgrote deel van de mensen op de vloer hebben geen idee. De managers en product owners wel maar de rest nog niet. Met andere woorden dat iedereen weet dat binnenkort al het nieuwe platform eraan komt en wat dat precies inhoudt.

Simon Houmes (15-04-2019

Wat is je specifieke functie binnen NN?
Manager Digital Commerce & manager Digital Application, concreet betekent dit het motiveren en ondersteunen van mensen in deze Area. Verder systeemeigenaar van ECTF (Electronic Crimes Taskforce), dit heeft met security te maken.

Heb je een idee van de doorlooptijd binnen het huidige platform, de huidige manier van werken?
Bij niet alleen de app maar ook de portals duurt dit weken, dit komt door het doorlopen van veel verschillende disciplines.

Wat verwacht je van het nieuwe platform vergeleken met die huidige manier van werken?
Life cycle management is op dit moment slecht, er zijn op dit moment versies van software in gebruik die niet meer ondersteunt worden door de fabrikant maar ook te weinig functionaliteit hebben. Daar komt nog bij dat de huidige infrastructuur niet toereikend is.

Wat vind je dan van het concept self service?
Helemaal voorstander omdat dit afhankelijkheden van logge bureaucratische procedures wegneemt. Daarnaast eigen invloed in performance is belangrijk.

Wat verwacht je van de verschuiving van on premise naar in de cloud?
Resources zijn op dit moment ondermaats. Hierdoor ontstaan veel prio één en prio twee meldingen, dit houdt in dat delen of de gehele functionaliteit niet op orde is. Daarnaast is performance ondermaats waardoor de app en sommige webonderdelen te traag zijn. Nieuwe functionaliteit pushen duurt op dit moment te lang en de cloud zal hier uitkomst moeten bieden.

Wat verwacht je van de mogelijkheid van autoscaling?
Het scalable maken van applicaties is belangrijk voor de performance, die op dit moment ondermaats. Dit is vooral te merken in de laadtijd van onderdelen van de app.

Wat vind je van de verschuiving van verantwoordelijkheid van de teams in de keten naar de developer?
Dit is een goede ontwikkeling, zeker ook voor de developer zelf. Die moet nu allerlei teams af en vragen om de juiste resources terwijl in de toekomst de developer dit zelf kan doen. Dit is dus goed voor het moreel van de developer.

Is de hoeveelheid kennis die een developer nodig heeft om het platform te gebruiken een indicatie van het succes van het platform?
Als er door de eenvoud van het platform geen kennis nodig is van infrastructuur lijkt me dat wel een Key Performance Indicator. Daardoor wordt het werkpakket van de developer alleen maar interessanter.

Is er op dit moment CSF waarvan je denkt dat die op dit moment minder aandacht

krijgt dan het verdient?
Daar heb ik dit moment geen beeld van. Eerst moeten deze goede initiatieven goed worden uitgebracht. Zoals het aanhouden van industriestandaarden is op dit moment belangrijk zodat er wanneer nodig ook support mogelijk is vanuit de leverancier.

Hoe vind je dan het gebruik van meer open source software, waar over het algemeen geen support voor is?
Geen probleem als het maar veilig is eventuele problemen kunnen alsnog worden opgevangen door eigen mensen.

Wat zou je top drie zijn, dus in volgorde van prioriteit, van het rijtje met CSF's?
Security boven alles, dan beschikbaarheid en nummer drie is performance.

Content Team (16-04-2019)

Wat is jullie specifieke functie binnen NN?
Content manager, we plaatsen de content op verschillende websites van NN. Na de komende transitie naar de area's wordt dat anders, een deel blijft in de huidige situatie als overall web manager. De anderen worden webredacteurs.

Wat is de verbinding tussen jullie team en het nieuwe digitale platform?
Niet direct verbonden, we zijn afhankelijk van het Content Management System GX / XC. Daar wordt alle content in geplaatst. Dus we zijn gebruikers van het CMS. We zijn dus gebruiker van de gebruiker (CMS) van het platform.

Waar lopen jullie nu tegenaan met de huidige manier van werken, het huidige platform?
Het gaat voornamelijk om de versies van het CMS. Bij de verschillende portals wordt een verschillende (verouderde) versie van GX / XC gebruikt. We willen graag dat die versies gelijk getrokken wordt dat kan op dit moment niet door het huidige platform. Daarnaast is de performance ondermaats. Deze problemen worden op dit moment niet aangepakt vanwege de investering in het nieuwe platform. Een nieuwe versie van het CMS wordt verlangd aangezien deze nieuwe mogelijkheden en minder bugs met zich meebrengt.

Zijn jullie sceptisch over de haalbaarheid van de tijdsplanning?
Ja, er wordt al een slag om de arm gehouden. Er moet echter een ambitie gemaakt worden met bijbehorende planning.

Wat zijn jullie verwachtingen van het nieuwe platform?
We verwachten dat met de snellere deployment in de cloud dat we straks maximaal 2 versies achterlopen met het CMS. We werken nu met een versie die 4 jaar oud is en niet meer ondersteund wordt door de fabrikant. Het nieuwe platform met bijbehorende nieuwe versie is voor onze functie alleen maar gunstig.

Jullie zijn dus over het algemeen enthousiast over het nieuwe platform?

We zijn sceptisch over de haalbaarheid van de planning, het maakt ons niet uit als we maar naar een nieuwe versie van het CMS gaan.

Hebben jullie naast performance en nieuwe versies ook te maken met andere beperkende factoren zoals downtime?
Downtime komt soms voor, soms bewust want dan moeten er zaken opnieuw gestart worden.

Wat zou je top drie zijn, dus in volgorde van prioriteit, van het rijtje met CSF's?
Eerst is die nieuwe versie, dan performance. Over de rest van de factoren zijn we tevreden, dus downtime komt voor maar is niet ondermaats. Soms ligt wel de redactie kant van het CMS eruit.

Zijn er factoren die meer aandacht verdienen dan het nu krijgt?
Het is voor ons een ver van ons bed show aangezien we niet direct te maken hebben met dit platform.

Dick van Es (19-04-2019)

Wat is je specifieke functie binnen NN?
Platform Architect voor CIO. Alles wat onder een applicatie zit, dus bv. networking, databases etc. Dat ontwerp ik voor de gehele groep.

Wat voor connectie heb je dan met het Digital Foundation team?
Dat is meestal case gedreven, als er een bepaald raakvlak is. Daarnaast is er elke twee weken een meeting waarbij alle lead architecten samen ervaringen en cases delen, om zo van elkaar op de hoogte te blijven. Dus als het DF-Team bezig is met de implementatie van een bepaald relevant onderdeel, dan benaderen ze ons.

Wat vind je van de verschuiving van de verantwoordelijkheid van de deployment van verschillende teams naar de developer?
Dat past helemaal binnen de huidige strategie van Agile en DevOps. DevOps moeten zo min mogelijk afhankelijk zijn van anderen aangezien voorspelbaar kunnen werken, deze verschuiving waarborgt de vereiste onafhankelijkheid. De grote factor hierin is de selfservice eigenschap van het nieuwe platform. Daarnaast is de inperking van de deployment een groot voordeel.

Wanneer is het nieuwe platform een succes vanuit eigen referentiekader?
Het is succesvol als alle product teams hun werk kunnen doen zonder al te veel hindering door de foundation om snel en betrouwbaar en snel producten neer te kunnen zetten. Cruciaal is de identificatie van die diensten, ofwel omschrijving van de functionaliteiten en dat inpakken in een dienst die makkelijk te gebruiken en te integreren zijn door die teams. Het ontwerp van de bouwblokken van het platform zijn belangrijk zodat alle teams met dezelfde bouwstenen toch verschillende applicaties kunnen bouwen. Daarnaast is de financiële kant ook belangrijk hoe belast je welk team voor het gebruik. Het wordt een wat meer klant leverancier relatie, om het generiek te houden kunnen er geen aanpassingen gemaakt worden voor een specifiek team.

Ben je enthousiast of sceptisch over het nieuwe platform?

Over het algemeen enthousiast, echter zijn de voorgaande genoemde factoren zaken die snel over het hoofd gezien worden. Communicatie is dus belangrijk, dit komt volledig door de andere relatie tussen leverancier en klant. Het model zelf en de strategie kan ik alleen enthousiast over zijn. Documentatie is van belang, de klant moet zelf in staat zijn het product volledig te gebruiken zonder al teveel tussenkomst van het DF-Team.

Is de term User Experience een goede term om dat samen te vatten?
De term dekt een deel goed, het helpen met een goede UX voor een BU komt ze alleen maar ten goede. Dat maakt het prettig voor ze met het afnemen van de dienst. Het gevoel van een klant dat hij geholpen is, is sterk afhankelijk van deze UX.

Wat zou je top drie zijn, dus in volgorde van prioriteit, van het rijtje met CSF's?
Prioriteit een is het definiëren van de services, twee is goede levering, derde is dat de klant weet hoe hij de services moet gebruiken.

Zijn er factoren die meer aandacht verdienen dan het nu krijgt?
Nee, kan ik niets van zeggen.

Peter Schlingmann (19-04-2019)

Wat is je specifieke functie binnen NN?
Product owner van retail customer service, dus voor particuliere klanten.

Wat is de verbinding tussen jouw area en het nieuwe digitale platform?
We hebben aantal kanalen voor het bieden van service naar de klanten. Denk hierbij aan een Mijn omgeving en de App. Hier kunnen we binnen de area aan werken, maar we kunnen aantal zaken, vooral in de ICT hoek, niet uitvoeren. Deze zaken zitten vooral in de Foundation, zaken als toegangsbeheer en hosting.

Dus jouw area is gebruiker van de DF / PaaS of toch meer onderdeel van?
Dat is maar net hoe je het bekijkt om alles te kunnen bieden aan de eindgebruiker, nemen wij onderdelen af van de DF. Als je het bekijkt als interne PaaS dan zijn wij inderdaad een gebruiker.

Wat vind je van het huidige platform, heeft deze tekortkomingen?
Er zijn nu een aantal component teams, er zijn dus meerder teams met afzonderlijke backlogs en eigen taken. Om iets nieuws uit te brengen voor een eindgebruiker moet je over deze teams heen. Het is process heeft daardoor een lange doorlooptijd. Daarnaast lig er een uitdaging om sneller het product te ontwikkelen. Er is nu behoefte om een stuk sneller nieuwe versies uit te kunnen brengen. Ik verwacht van het nieuwe platform dat dit een beter life cycle management / time-to-market met zich meebrengt.

Nemen jullie ook (net als het CMS) ook software af die door het huidige platform op een verouderde versie draait?
Ik heb de indruk dat we inderdaad vaak met verouderde versies werken. Voorbeelden daarvan zijn het CMS en onderdelen van de app. Het is jammer als er nieuwe onderdelen beschikbaar zijn maar

niet toegepast kunnen worden, zeker als je kijkt naar concurrenten.

Wat vind je van de verschuiving van de verantwoordelijkheid van de deployment van verschillende teams naar de developer?
Er is dan maar 1 backlog binnen het multidisciplinaire development team, hierdoor kan je kort cyclisch zaken uitvoeren, dat is een pluspunt. Het nadeel is dan dat het testen ook werk voor dat team wordt. Het betekent dus dat er binnen de area meer werk is.

Welke facoren maken het nieuwe platform in jouw ogen een succes?
Het moet platform moet snel zijn, online zijn, veilig zijn en wendbaar zijn. Het moet kunnen aansluiten op andere componenten.

Zijn er factoren die meer aandacht verdienen dan het nu krijgt?
Als ik gezien wordt als klant, dan verwacht ik dat de klant zo goed mogelijk bediend wordt. De leverancier moet blijven nadenken over hoe ze het de klant zo makkelijk mogelijk kunnen maken. Er moet dus een balans zijn tussen wat geboden wordt en het gebruiksgemak voor de klant.

Hoe is op dit moment de communicatie over technische aspecten?
Op dit moment is die communicatie er niet, in de zin dat je dingen aangevraagd en die worden opgeleverd. Daarvoor is geen documentatie of training nodig.

Ben je enthousiast of sceptisch over het nieuwe platform?
Op het gebied van architectuur denk ik dat het een grote stap is, dus in dat opzicht enthousiast. Ik maak me wel zorgen over de focus op dit project terwijl er op andere gebieden ook aandacht nodig is. Over die prioritisering ben ik ietwat sceptisch. Echter over het algemeen is het de goede richting.

Ben je dan sceptisch over de planning?
Ik heb veel vertrouwen in Alain en zijn team, echter is het met meerjarige ICT projecten vaak zo dat ze niet de eindstreep halen als van tevoren is bedacht. Ik heb ook niet het idee dat we in een keer alles kunnen wat we zouden willen, als we al precies weten wat we willen.

Tim van Ommen (19-04-2019)

Wat is je specifieke functie binnen NN?
Manager van enkele teams die werken aan generieke onderdelen van de digitale stack. De teams zijn het App team, het CMS-team, Webcomponents team, Access en Overview team.

Wat is de verbinding tussen je functie en het nieuwe digitale platform?
Het CMS-team en het Webcomponents team dragen bij aan dit platform. We denken er nu zelf aan om het Webcomponents team onderdeel te laten zijn van het platform. Daarnaast zijn de andere teams, app en Access en Overview, gebruikers van het platform.

Wat vind je van het huidige platform, heeft deze tekortkomingen?
Ja, dat er teveel afhankelijkheid is van C&C als je van buiten C&C op het platform iets wilt ontwikkelen. Je hebt bij elke applicatie zeker nieuwe altijd wel iemand nodig om het te deployen.

Er is op dit moment veel handwerk. Het huidige platform dwingt ook enkele afspraken niet af, bijvoorbeeld styling. Er wordt nu nergens technisch afgedwongen dat elke applicatie dezelfde NN-styling gebruikt. Er is niet voor niets een huisstijl, het gebruik ervan zou technisch afgedwongen kunnen worden. Het nieuwe fundament zou hier een rol in kunnen spelen. Daarnaast is het traag en ligt het er vaak uit. Ook is de complexiteit van de route van de deployment een minpunt. (later in gesprek genoemd) Ook een belangrijk punt is het life cycle management.

Wanneer is het nieuwe platform een succes vanuit eigen referentiekader?
Aangezien ik aan de ontwikkel kant van dit platform sta, vind ik het moeilijk om dat in te schatten. In mijn ogen is dit een succes als de mensen die dit moeten gebruiken, dit platform liever gebruiken.

Zou je dit met de term User Experience kunnen samenvatten?
Ja, de gebruiker moet daar tevreden over zijn. Ze moeten er eigenlijk niet eens over na hoeven te denken, of proberen het zelf te fabriceren.

Wat zou je top drie zijn, dus in volgorde van prioriteit, van het rijtje met CSF's?
Performance is het belangrijkst, dit komt zowel de interne gebruiker als de echte eindgebruiker ten goede. Minder handwerk, met andere woorden de selfservice mogelijkheden. Alle anderen staan op een derde plaats.

Wat vind je van de verschuiving van de verantwoordelijkheid van de deployment van verschillende teams naar de developer?
Erg goed mits de juiste controles ingebouwd zijn in de pipelines. Echter, zijn die controles er nu ook nog niet.

Maar die teams die afgelopen moeten worden, controleren toch?
Nee als een developer weet wat hij wil dan voeren die team alleen maar uit. Er wordt gezegd dat er geen tijd voor is maar niet dat het geen goed idee is.

Zijn er factoren die meer aandacht verdienen dan het nu krijgt?
Testen, testen van applicaties voor de deployment.

Testen aan de kant van het platform of aan de kant van de developer?
Het helpen van developers om het testen van de applicatie makkelijk te maken. Bijvoorbeeld: Blueriq Model Analyser is aangeschaft, daarmee kan door vooraf opgestelde regels een applicatie gecontroleerd worden.

Bart van der Poel (23-04-2019)

Wat is je specifieke functie binnen NN?
Verantwoordelijk voor het goed functioneren van Blueriq, zowel vanuit C&C als vanuit Blueriq. Met andere woorden het tevreden houden van de klant en de Blueriq gebruikers.

Wat bedoel je met de klant, is dat de eindgebruiker?
Eigenlijk alle gebruikers dus zowel de klanten binnen NN als de eindgebruikers buiten NN. Als de

eindgebruiker niet tevreden is, dan is NN dat ook niet.

Wat vind je van het huidige platform, de huidige manier van werken?
Het landschap is te complex, de keten werkt niet goed. De individuele blokjes werken goed, het is echter door de omvang te complex geworden om goed te kunnen doorlopen.

Er zijn meerdere stakeholders die het juist niet complex noemen, maar vooral langdradig. Wat vind je daar van?
Qua doorlooptijden ben ik het daar mee eens, maar het is beide. Dat komt doordat de teams zo goed mogelijk willen werken. Hiervoor zijn teams overgegaan op Scrum. Als zo'n team nodig is maar nog bezig met een sprint dat creëert dit onbegrip. Als dit beter gecommuniceerd zou worden zou er meer begrip zijn. Dus ik denk dat het beter is om dat te verbeteren dan alles weg te gooien en opnieuw te beginnen.

Vind je het nieuwe platform dan een nieuw begin i.p.v. het verbeteren wat er al is?
Het platform is wel nieuw, de huidige manier van werken is niet handig en duurt lang. Doorlooptijden van meerdere maanden is onacceptabel. Van de nieuwe manier van werken wordt iedereen blij.

Zijn er naast de complexiteit en doorlooptijd nog andere tekortkomingen?
Afhankelijkheden, als er problemen zijn dan wil je dat er zo snel mogelijk een oplossing voor komt. Nu ben je afhankelijk van anderen door hun functie, waardoor dit lang kan duren.

Met het nieuwe platform verschuift de verantwoordelijkheid van die teams naar alleen de developer (het ontwikkelteam), wat vind je daar van?
Ondersteuning is hier cruciaal, ik geloof niet dat iedereen alles kan. De problemen worden niet automatisch opgelost als de developer het zelf kan oplossen. Het huidige handhaven met een nieuw fundament, oftewel one click deployment, zal al veel oplossen. Maar als er problemen worden ondervonden moet er een goede ondersteuning zijn. De praktijk is vaak anders.

Het is dus afhankelijk van die ondersteuning, zou je dat kunnen vatten in user experience? Daar zit dan ondersteuning en de gebruiksvriendelijkheid van het gebruik in.
Als je kijkt naar het kopen van een product in een webshop, dan bestel je niet alleen omdat de webshop overzichtelijk is en makkelijk werkt maar ook omdat als het product niet is wat er van verwacht werd, het terug gestuurd kan worden. Dat zou met het nieuwe platform ook het geval moeten zijn. Met andere woorden, de klant moet een goed gevoel hebben over het afnemen van een product, dit goede gevoel komt dan doordat er goed gecommuniceerd wordt over wat de klant nodig heeft en dat als de klant iets mis doet dat er dan vangnetten zijn die hem helpen.

Zou er dan naast documentatie en ondersteuning ook training nodig zijn voor alle teams die dit platform gaan gebruiken?
Dat contact is belangrijk, het liefst face to face. C&C zou continue bezig moeten zijn met het ontzorgen van haar klanten. Dus niet alleen een product leveren dat werkt maar ook een product leveren voor het geval iets niet werkt.

Wanneer is het nieuwe platform een succes vanuit eigen referentiekader?
De genoemde support is belangrijk. Daarnaast moet het duidelijk zijn wat geconfigureerd moeten worden, d.m.v. wizards of stappenplannen etc. Ook moet tijdens die configuratie duidelijk gemaakt worden dat bepaalde instelling zoals bv. availability ook gevolgen hebben voor bv de kosten.

Moet die bewustwording van de klant zelf komen of van een andere kant?
Dat is moeilijk te zeggen. Het is alleen belangrijk dat iemand daarvoor verantwoordelijk is.

Zou voor het bepalen van sommige configuratie onderdelen een automatisch stappenplan een oplossing zijn?
De kosten ofwel de gevolgen moeten inzichtelijk zijn voor het team dat de applicatie heeft gemaakt en online zet. Als team kun je dan sturen op basis van die gegevens. Daar komt dan automatisch die bewustwording vandaan.

Nog andere factoren die je belangrijk vindt?
Nee op dit moment niet meer.

Wat vind je van de overgang van Websphere naar het open source Tomcat?
Technisch gezien heb ik daar geen mening over, ik heb alleen het idee dat Websphere een bottleneck is.

Is die bottleneck technische of heeft het een andere oorzaak?
Alles kan, maar er is wel degelijk een bottleneck. Het is echter nog maar de vraag of iets nieuws als Tomcat die beperkingen gaat verhelpen. Nieuw is, lost niet altijd alle beperkingen op, misschien ontstaan er wel nieuw beperkingen.

Wat zou je top drie zijn, dus in volgorde van prioriteit, van het rijtje met CSF's?
Vanuit Blueriq is doorlooptijd het belangrijkst. Met Blueriq kan snel een applicatie in elkaar zetten. Maar als de landing ervan lang duurt, heeft dat wel invloed.

Zijn er factoren die meer aandacht verdienen dan het nu krijgt?
De looptijd van hun eigen project. Het gevoel is er dat het af is wanneer het af is. De BU's willen dat hun problemen nu worden opgelost. De kans is er dat bepaalde bedrijfsdelen het te lang vinden duren, vervolgens C&C aan de kant schuiven en dan zelf een oplossing in elkaar zetten. Er zou meer bewustwording gecreëerd moeten worden bij de verschillende bedrijfsonderdelen voor wat C&C doet.

Application managers (23-04-2019)

Wat is jullie specifieke functie binnen NN?
Onderdeel Area Retail & Commerce, het inrichten van het CMS, onderdeel daarvan is bv. het maken van interactieve formulieren en ook performance advies. Onderdeel Area Retail & Commerce, voorheen het DAM-team dat is onderhoud van de websites via GX.

Wat is de verbinding tussen je functie en het digitale platform dat nu gebouwd wordt?

Voornamelijk op het gebied van performance verbetering, de laadtijd is nu erg lang omdat er onderdelen geladen worden die niet nodig zijn. Ik verwacht dat dat met het nieuwe fundament verbeterd gaat worden, niet direct van het platform maar meer van de gehele vernieuwing.

Is die laadtijd afhankelijk van de versie van de software die gebruikt wordt?
Een onderdeel daarvan is de oude versie van GX die nu in gebruik is, er worden in de huidige versie bv. meerdere versies geladen van plaatjes terwijl er maar een gebruikt wordt.

Voor een nieuwe versie is het wachten op de cloud?
Dat is op dit moment nog niet helder, het kan zijn dat er al een nieuwe versie wordt geïnstalleerd op het huidige platform. Maar het kan ook zijn dat er pas een nieuwe versie wordt geïnstalleerd als het nieuwe platform in gebruik wordt genomen. Of er dan opnieuw begonnen wordt met de website of dat de componenten gemigreerd worden is ook nog de vraag.

Die nieuwe versie van GX is op dit moment het belangrijkst?
Ja die grote stap willen we maken, we liggen op dit moment stil met het verwerken van dingen op de backlog.

Die items komen die dan voort uit gebreken?
Dat is een onderdeel, er zijn performance wensen die niet voldaan kunnen worden. Maar er zijn ook veel features die niet ingebouwd kunnen worden. Er zijn ook veel elementen waarvan niemand weet of die nog gebruikt worden. Het is nu nog niet mogelijk om die eruit te filteren.

Vind je dat er nu te veel aandacht gaat naar andere zaken?
Ik kan dat erg moeilijk beoordelen, ik vind mijn stuk belangrijk maar ik heb geen zicht op de prioriteiten van andere zaken. Het hangt ook af van besluiten die genomen worden op zowel het gebied van prioriteiten maar ook ICT structuur. In de toekomst wordt de leverancier gevolgd

Wat vind je van die keuze?
Ik vind dat erg lastig, ik zit niet diep in die materie. Het lijkt me wel dat als je de leverancier volgt dat er dan minder maatwerk is. Een eigen structuur die leidend is niet het makkelijkst, aangezien je zelf dan de aanpassingen moet maken. Misschien moeten ze zelf niet de leverancier volgen maar de cloud.

Kan je een voorbeeld noemen waardoor dat beter zou zijn?
Ja je hebt Lambda, daar kan je veel meer mee doen in mijn ogen. Daar kan je binnen korte tijd alles opnieuw inrichten. Het lijkt me beter om in een keer alles om te gooien dan stap je voor stap te doen. Dus i.p.v. PaaS gebruiken AWS als SaaS gebruiken.

Ben je niet bang dat als er nu begonnen wordt met alles vernieuwen, dat tegen de tijd dat alles af is dat je dan weer achter loopt? Ik denk dat als zaken vernieuwd zijn dat het makkelijk wordt om die zaken te vernieuwen doordat het de zaak flexibeleren maakt, en ondertussen kunenn andere zaken worden vernieuwd. Nu blijf je altijd achterlopen.

Leon Kortekaas (26-04-2019)

Wat is je specifieke functie binnen NN?
Manager Cloud Integration, dit is de centrale IT-dienst. We leveren diensten die verschillende BU's afnemen.

Wat is de verbinding tussen je functie en het digitale platform dat nu gebouwd wordt?
Het Digital Foundation team bouwt het platform op de infrastructuur die we leveren. Voor het nieuwe platform maken ze gebruik van onze AWS-diensten.

Een vorige stakeholder zei dat het huidige platform geen platform is, vind je dat ook?
Het is wel een platform, het is alleen niet consumeerbaar. Het is veel handmatig, het ligt aan de definitie en de eisen die je stelt, maar ik vind het wel een platform.

Wat vind je van het huidige platform, de huidige manier van werken?
Ik ben geen gebruiker, maar ik hoor er wel klachten over. Op dit moment duurt het veel te lang om simpele wijzigingen door te voeren. Er moeten meer teams afgegaan worden om tot een verandering te komen. Daarnaast is kwaliteit ook een probleem.

Waar moet ik dan bij kwaliteit aan denken, bijvoorbeeld up-time?
Ook maar ook first time right. Bij verandering komt het nog wel is voor dat de bedoelde verandering verkeerd wordt geïnterpreteerd. Dus een knopje moet groen worden maar wordt eerst oranje.

Ligt dat aan de communicatie?
Ja, er zijn teveel handovers. Je kunt zelf geen verandering doorvoeren, dus het moet door een keten van mensen voordat een verandering doorgevoerd is. Dat zou je ook zelf kunnen doen, als je het maar simpel genoeg maakt.

Dus minder handovers, zijn er nog andere factoren wat het nieuwe platform een succes zou maken?
Het moet simpel te gebruiken zijn, dus selfservice / automatisering maar ook de documentatie moet duidelijk zijn en goed beschikbaar zijn. Het moet makkelijk te consumeren zijn.

Wat zou je top drie zijn, dus in volgorde van prioriteit, van het rijtje met CSF's?
Security, reliability en laadtijd zijn niet onderhandelbaar. De gebruiker en de markt eisen dat. Bijna alle eisen zijn niet onderhandelbaar.

Een van de CSF's is de flexibiliteit voor toekomstige veranderingen, is dit platform in je ogen flexibel genoeg?
Het platform is niet state of the art. Er kunnen nu andere en modernere dingen, maar je moet het opbouwen. Het DF-Team moet de kans krijgen om hiermee lessen uit te leren om in de toekomst mee te kunnen lopen met de nieuwste technieken.

Maar als ze die ervaring hebben kan de markt weer ergens anders mee bezig zijn.

Ze moeten sneller ervaring opdoen dan de markt zich ontwikkeld. Dat is moeilijke taak maar het gebied waar ze in opereren is ook niet zo groot.

Zou je dit platform dan zien als tussenstap?
Het is nooit een eindpunt maar het is inderdaad een tussenstap?

Is het platform een nieuw iets of een nieuwe versie van?
Het is een nieuwe versie, je kunt als bedrijf niet alle elementen van vroeger achterlaten puur en alleen om over te stappen op het nieuwste van het nieuwst. Bij NN kunnen we niet zomaar alle Blueriqs weggooien, je kunt niet verwachten dat al die programma's binnen 2 dagen opnieuw gebouwd zijn. Er moet een balans zijn tussen innovatie en programma's die bewezen hebben dat ze werken.

Vind je dit nieuwe platform een goede balans daarin?
Dat kan ik moeilijk zeggen, ik weet niet precies waar ze naartoe gaan. Ik kan alleen zeggen dat we met het toendertijd AWS-platform de sprong te snel gemaakt hebben. Het was voor onze gebruikers toen geen makkelijke transitie.

Zou je jezelf indelen als sceptisch of enthousiast?
Meer enthousiast, de richting die ze ingaan is goed en de enige manier om te leren hoe het werkt. Ze moeten door blijven itereren en leren, ook gaat het waarschijnlijk de eerste keer mis.

Zijn er factoren die meer aandacht verdienen dan het nu krijgt?
De bruikbaarheid van het platform is een belangrijke factor. Ik vind dat de BU's die gebruik gaan maken van het platform niet genoeg bij de ontwikkeling betrokken worden.

En die betrokkenheid moet in deze fase al?
Ja dat is nodig. Ik kan het niet goed inschatten maar dat zou ik het team adviseren.

Kees Kool (26-04-2019)

Wat is je specifieke functie binnen NN?
Digital design team strategie digitale proposities (UX) Na de komende transitie: PO zakelijk: Area, bestaand uit een verzameling scrum teams. Wij zijn verantwoordlijk voor C&C zakelijke klanten.

Wat is de verbinding tussen je functie en het digitale platform dat nu gebouwd wordt?
Na de komende transitie gebruiker van het platform

Wat vind je van het huidige platform, de huidige manier van werken?
Huidige platform: nuttig maar onhoudbaar. Veel te veel binnen een paar teams. Langdradig doordat keten van teams.

Wat verwacht ten opzicht van het huidige platform van het nieuwe self-service platform?
Minder overdrachtsmomenten daardoor minder coördinatie nodig dus makkelijker en sneller.

Wat zou je top drie zijn, dus in volgorde van prioriteit, van het rijtje met CSF's?
Instant deployment nummer 1, overige factoren minimaal net als het huidige platform Backend calls duren het langst, het nieuwe platform gaat daar niks aan veranderen, dus response tijd als CSF is niet aan de orde.

Zijn er op dit moment CSF waarvan jij denkt dat die op dit moment minder aandacht krijgt dan het verdient?
Wie zijn de gebruikers? en hoe ga je ze met deze transities helpen? dus documentatie training etc.

Hans Berlijn (01-05-2019)

Wat is je specifieke functie binnen NN?
Security Architect bij C&C, ik bepaal hoe we omgaan met security welke maatregelen we nemen. De hoge richtlijnen vanuit centraal worden omgezet naar meer toegespitste regels. Daarnaast het beschrijven van de algemene ontwerpen met als focus de veiligheid is een ander element van mijn functie.

Wat vind je van het huidige platform, de huidige manier van werken?
Het is complex, er is weinig overzicht. Het DF-Team heeft veel kennis, maar de verschillende BU's hebben die kennis niet. Ze hebben het gevoel dat ze naar een loket moeten. Dat zie je ook met security, mijn collega gaat langs allerlei teams om er zeker van te zijn dat ze allen voldoen aan de eisen.

Er zijn veel stakeholders die het langdradig vinden en niet complex, vind je dat ook?
Dat weet ik niet zo goed, het duurt sowieso weken voordat iets is geland.

Zijn er in je ogen nog anderen plus of minpunten aan de nieuwe manier van werken?
Ik hoop dat er een hoop fouten niet meer gemaakt kunnen worden. Daarnaast hoop ik dat als dit project af is dat er dan genoeg mensen met kennis ervan aanwezig blijven.

Acht je die kans dat de mensen weggaan groot?
Ik zie het als mogelijkheid, nu ziet er groot team op met de juiste kennis vergeleken met een tijd terug dat er maar een enkeling mee bezig was.

Is er wel een einde aan dit project, zou dit platform niet door moeten blijven ontwikkelen?
Dat ligt aan hoeveel mensen daarvoor nodig zijn, misschien heb je nu 20 mensen nodig maar is om verder te innoveren minder nodig. De beweging zoals het nu is, is goed.

Zou je jezelf indelen als sceptisch of enthousiast?
Erg enthousiast, ook geen delen waar ik sceptisch over ben.

Wanneer is het nieuwe platform een succes vanuit eigen referentiekader?
Ik zou graag zien dat als alles veilig, snel en niet stuk te krijgen is dat dan het een kant en klaar product wordt. Dus dat generieke onderdelen als container aangeboden worden. Afgezien van custom onderdelen als het CMS.

Er zijn enkele mensen die vinden dat het CMS sowieso al verouderd is en dat er gekozen moet worden voor iets anders.
Als er nu opnieuw begonnen zou worden, dan zou ik wel kiezen voor een ander CMS. Je hebt wel een CMS nodig, de huidige oplossing is zeker niet optimaal. Maar het is niet iets waar nu miljoenen tegenaan gegooid moet worden.

Zijn er nog andere factoren die het een succes zouden maken?
Het zou fijn als nieuwe dingen ook echt sneller zou zijn. Dus als er ook met nieuwe API's en backend calls de klant.

Zijn die prestaties dan meer performance of ook laadtijd?
Echt meer performance, het opstarten van de applicatie tot hoe snel calls worden verwerkt.

Wat zou je top drie zijn, dus in volgorde van prioriteit, van het rijtje met CSF's?
Security vanuit mijn rol op één, voor de eindgebruiker maakt het niet uit hoe snel een service is ontwikkeld of het nou een week is of een maand, als het maar stabiel is.

Zou je kunnen zeggen dat die factoren samen met availability een gegeven zijn?
Ja, die worden vanuit de markt bepaald, we kunnen niet zonder.

Zou je die dan een op een gedeelde eerste plaats willen zetten?
Ja eigenlijk wel, het zou dan vooral eenvoudig moeten zijn voor de ontwikkelteams. Daarnaast zouden de security instellingen standaard al goed moeten zijn. Zodat in ieder geval al een basis is voor een goede security.

Zijn er op dit moment CSF waarvan jij denkt dat die op dit moment minder aandacht krijgt dan het verdient?
Nee eigenlijk niet. Er zijn prioriteiten die nu voorgaan en later komen er nog andere zaken aan de orde.

Gabriel Lommers (06-05-2019)

Wat is je specifieke functie binnen NN?
Product Owner voor de Area Retail & Commerce, dit houdt onder andere in dat ik ga over email marketing, Next Best Action strategieën & content marketing. Voornamelijk marketing.

Wat is de verbinding tussen je functie en het digitale platform dat nu gebouwd wordt?
Dat zit vooral in de campagne machine, daarmee zorgen we ervoor dat de verschillende campagnes logisch in elkaar over lopen voor de klanten. De digitale touchpoints zijn daarin belangrijk, dus bijvoorbeeld het CMS. We gebruiken dus tools die verzorgt worden door het DF-Team.

Wat vind je van het huidige platform, de huidige manier van werken?
Ik heb daar zelf niet vaak mee te maken, maar als we nu iets willen dan blijkt dat veel verschillende mensen iets moeten uitvoeren. Daardoor is dit geen geoliede machine met allemaal ook

nog een eigen backlog en eigen doelstellingen. Die alignment van backlogs moet in de toekomst verbeterd worden, daarnaast zou je in de toekomst graag willen dat teams zaken zelf kunnen regelen.

Zijn die twee factoren de voornaamste factoren die verbeterd moeten worden?
Ja, die twee kun je samen onderbrengen in het verlangen om flexibeler te zijn. Naast meer flexibiliteit kun je dus ook een stuk autonomer zijn met veranderingen.

Zijn er nog andere factoren die het nieuwe platform een succes zou maken?
Ik heb het gevoel nu dat de up-time niet specifiek bij het DF-Team ligt maar meer afhankelijk is van de BU waar de applicatie onder valt. Er gaat wel veel fout bij infra, ik ga er dus vanuit dat het nieuwe platform door de cloudservices daar verbetering in brengt. We gebruiken optimalisatie tools die we moeten verbinden met onze eigen services, ik hoop dat dat in de toekomst makkelijker kan met API's. Veiligheid is ook een belangrijk punt voor het nieuwe platform.

Veiligheid is een goed punt, er zijn stakeholders die vinden dat veiligheid samen met up-time en performance niet onderhandelbaar is. Vind je dat ook?
Dat is afhankelijk van hoe je ernaar kijkt. Als er nu een bepaald onderdeel niet werkt dan kan de impact daarvan klein zijn. Als je kijkt naar onze ambities dan denk ik wel dat dat inderdaad niet onderhandelbaar. Het is in mijn ogen niet van groot belang, echter verandert die mentaliteit als je kijkt naar de toekomst.

Wat zou je top drie zijn, dus in volgorde van prioriteit, van het rijtje met CSF's?
Dan zou ik zeggen support van het DF-Team is het meest belangrijk, compatibiliteit met de nieuwe versies van applicaties die we gebruiken zowel op het platform als via API's met applicatie die we gebruiken buiten het platform. Verder de innovativiteit van het platform op drie.

Zijn of is er op dit moment CSF waarvan je denkt dat die op dit moment minder aandacht krijgt dan het verdient?
Nee dat durf dat ik nu niet te zeggen, de belangrijkste zaken worden behandeld.

Luc Hartering (07-05-2019)

Wat is je specifieke functie binnen NN?
Manager Digital bij Pensioenen. Verantwoordelijk voor de strategie binnen dat domein.

Wat is de verbinding tussen je functie en het digitale platform dat nu gebouwd wordt?
We nemen de diensten af, dus zowel de basisdiensten als enkele toevoegingen zoals bijvoorbeeld autorisatie. Daar boven zetten we pensioen specifieke applicaties neer.

Wat vind je van het huidige platform, de huidige manier van werken?
Ik heb geen idee, ik kan meer zeggen over wat we in de toekomst nodig gaan hebben. Je hebt nu een groot aantal teams nodig om iets in productie te zetten. Ik geloof dat je met huidige technologieën dat eenvoudiger kan maken. Dat betekent dat teams autonomie hebben en zelf applicaties kunnen aanpassen en deployen. Voor een bepaald project nu hebben we tussen de dertig en 35 teams nodig.

Is dat doorlopen van die teams complex of langdradig?
Dat is afhankelijk van de competenties van de mensen in je team. Maar het kan snel een stuk complexer worden. Dus beide, waardoor er ook veel regie nodig is, het zou dus makkelijk zijn als we dat zelf kunnen doen.

Zijn er nog andere factoren die het nieuwe platform een succes zou maken?
Ik denk dat het belangrijk is het makkelijk naar productie brengen van applicaties, maar ook snel een fallback kan doen als dat nodig is. Maar ook monitoring is belangrijk, er moet snel ingegrepen kan worden wanneer er een probleem is.

Wat vind je van de term User Experience als overkoepelende term voor usability en support?
Het belangrijkste is dat duidelijk is wat de regels zijn tussen het DF-Team en de afnemers. Het is een samenspel tussen demand en supply. Er moeten afspraken gemaakt worden over wat er verwacht wordt en verwacht kan worden. Concreet voorbeeld is een stand-by. Is er een stand-by buiten kantooruren, hoe bereik je die, wat kun je er van verwachten etc. Maar ook dat de portalen tussen bepaalde tijden gegarandeerd beschikbaar zijn. Ook moeten er eisen gesteld worden aan de applicatie van de afnemer.

Wat vind je van de verandering in contact tussen het DF-Team en de afnemers, die was eerst samenwerkend en dat wordt een klant leverancier relatie?
Voor mij verandert de relatie niet zo, het is wel anders maar het grote punt is dat het er minder worden.

Zijn of is er op dit moment CSF waarvan je denkt dat die op dit moment minder aandacht krijgt dan het verdient?
Ik zit er niet bovenop, maar wat ik zie, is dat C&C standaarden levert. Ik vraag me vooral af hoe sterk die community onder die technologieën is. Bijvoorbeeld er komt binnenkort een nieuwe versie van Vue.js uit, Vue is een van de standaarden voor frontend. Ik denk dat we binnen de ontwikkeling met Vue nog wel stappen kunnen maken om de life cycle management te verbeteren. Waardoor nieuwe versies minder impact hebben. Ik verwacht dat daar richtlijnen voor uitgezet worden.

B Score Table

Name	Automatic self deployment	Automatic testing	Compatibility	Feature alignment	Cost management	Customer support	Flexibility	Innovativeness	Load time	Security	Tangible guidelines	Up-time	Usability
DF-team	2	0	0	0.5	1.5	1	0.5	0.5	0	0.5	0	0	0.5
Simon Houmes	0.5	0	0	0	0	0	0	0.5	1	2	0	1.5	0
Content Team	0.5	0	2	0	0	0	0	0	1.5	0	0	1	0
Dick van Es	0.5	0	0	1.5	0.5	1	0.5	0.5	0	0	0.5	0	2
Peter Schlingmann	2	0.5	1.5	1	0	0.5	0	0	0.5	0.5	0	0.5	0.5
Tim van Ommen	1.5	1	0.5	0.5	0	0.5	0	0	2	0	0.5	0	0.5
Bart van der Poel	2	0.5	0.5	1	0.5	1.5	0	0	0	0	0.5	0.5	0.5
Application managers	0	0	2	0	0	0	1	0.5	1.5	0	0	0.5	0.5
Leon Kortekaas	0.5	0	0.5	0.5	0	0.5	0.5	0.5	1.5	2	0.5	1	0.5
Kees Kool	2	0	0	1	0	1.5	0	0	0	0	0	0	0.5
Hans Berlijn	0.5	0	0.5	0	0	0.5	0	0.5	1	2	0.5	1.5	0.5
Gabriel Lommers	0.5	0	1.5	0	0	2	0.5	1	0	0.5	0	0.5	0
Luc Hartering	2	0.5	0.5	0.5	0	1.5	0	0	0	0	1	0.5	0.5
Totals	**14.5**	**2.5**	**9.5**	**6.5**	**2.5**	**10.5**	**3**	**4**	**9**	**7.5**	**3.5**	**7.5**	**6.5**