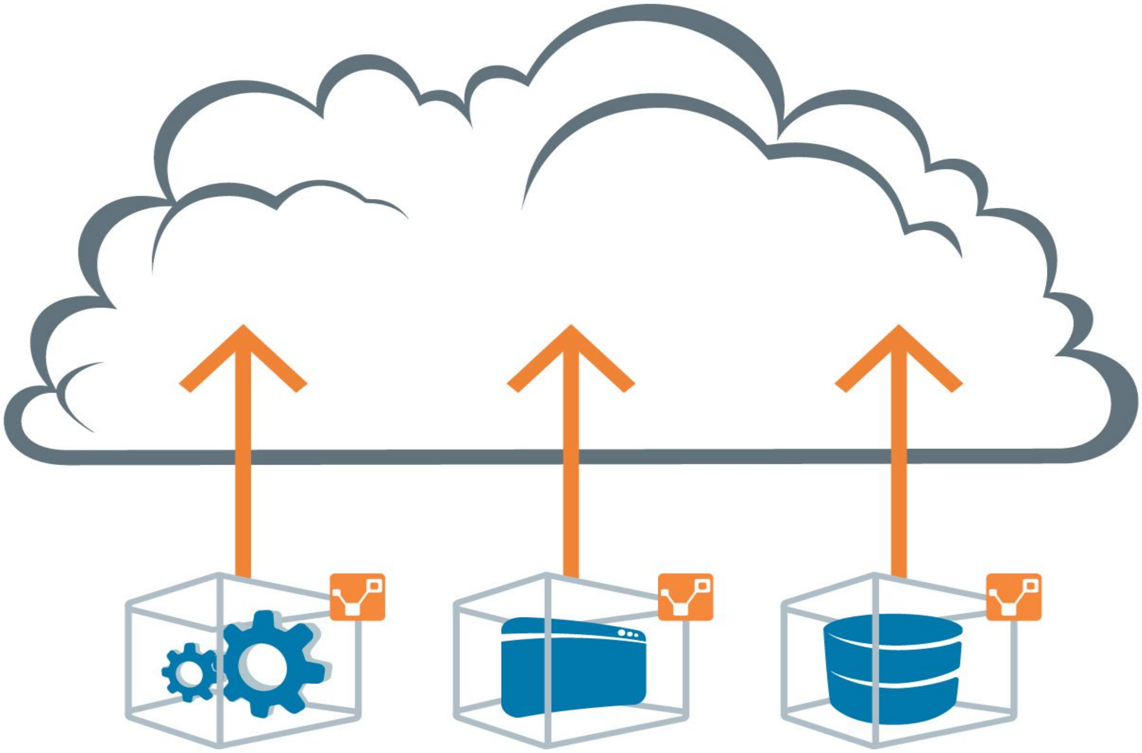


WirelessCar — A Requirements Engineering Case Study in a Legacy-to-Cloud Migration

Bachelor of Science Thesis in Software Engineering and Management

ALEXANDER ANDREASSON
DENNIS NIELSEN

Department of Computer Science and Engineering
UNIVERSITY OF GOTHENBURG
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2017

The Author grants to University of Gothenburg and Chalmers University of Technology the non-exclusive right to publish the Work electronically and in a non-commercial purpose make it accessible on the Internet.
The Author warrants that he/she is the author to the Work, and warrants that the Work does not contain text, pictures or other material that violates copyright law.

The Author shall, when transferring the rights of the Work to a third party (for example a publisher or a company), acknowledge the third party about this agreement. If the Author has signed a copyright agreement with a third party regarding the Work, the Author warrants hereby that he/she has obtained any necessary permission from this third party to let University of Gothenburg and Chalmers University of Technology store the Work electronically and make it accessible on the Internet.

WirelessCar — A Requirements Engineering Case Study in a Legacy-to-Cloud Migration
Dennis Nielsen
Alexander Andreasson

© Dennis Nielsen, June 2017.
© Alexander Andreasson, June 2017.

Supervisor: Jan-Philipp Steghöfer
Examiner: Imed Hammouda

University of Gothenburg
Chalmers University of Technology
Department of Computer Science and Engineering
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Cover:
The image on the cover depicts a visualization of different components being migrated to a cloud service platform, i.e. the cloud.

Department of Computer Science and Engineering
UNIVERSITY OF GOTHENBURG
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2017

WirelessCar - A Requirements Engineering Case Study in a Legacy-to-Cloud Migration

Alexander Andreasson
Department of Computer Science and Engineering
University of Gothenburg
Gothenburg, Sweden
gusandalad@student.gu.se

Dennis Nielsen
Department of Computer Science and Engineering
University of Gothenburg
Gothenburg, Sweden
gusnielde@student.gu.se

***Abstract*—This paper discusses the trade-offs between requirements, technical limitations and development efforts during a legacy-to-cloud migration at a software company that provides vehicle telematics services for over two million users in the commercial industry. To tackle the trade-offs that arose during the migration, a case study was conducted with stakeholders at the company who were interviewed and the resulting trade-offs analyzed. An engineering proof-of-concept was also developed in the form of a business reporting tool on Amazon Web Services, to further validate the data gathered throughout the case study. Our results aim to highlight the fact that a cloud migration process is complex and that sacrifices have to be made during development to keep certain business goals intact.**

***Keywords*-requirements engineering; legacy-to-cloud; trade-offs; migration; technical limit; development effort; AWS; amazon**

I. Introduction

With the emergence of new software technologies that offer more computing power, cheaper and scalable data-capacities, and high speed bandwidth, cloud computing and its services have become increasingly more common and attractive from small to large-scale companies, world-wide [1] [2] [3] [4] [5] [6]. Cloud computing is defined as on-demand computing services that can serve applications and databases with fast provisioning and minimal deployment effort, using the “Pay-As-You-Go” or “Pay-Per-Usage” payment model whereas the customer is only charged for computation time or storage used [7]. As more organizations are adopting such trends in services to dynamically adjust their vision into matching their customer’s requirements, on-the-premises to cloud platform migrations have become more prominent, not only entailing the technical transition, but also the organizational changes and improvements which reflect upon the business goals and visions. Jamshidi and Phal point out that some of the main drivers behind cloud migrations are cost-saving, scalability and flexibility of services [8] whilst Hosseini and Sommerville, in their case study [9], add that one has to consider potential risks like uncertainty with new technology or deterioration of a service’s quality. Consequently, three major aspects are considered: requirements engineering, its technical implications and the resulting development efforts – how teams perform and work. These ensures that stakeholder requirements are properly elicited, that technical challenges such as software or hardware limitations are taken into account and mitigated beforehand, as they have the potential to drastically affect the system and its customers, and that development teams are able to effectively accommodate changes. This research paper focuses on a requirements engineering case study and an engineering task that entails a legacy-to-cloud platform migration, within a company-specific context. Three major key aspects are researched: *Requirements*, *Technical Limitations* and *Development Efforts*. Their relationship and trade-offs will be studied, closely monitoring how they change throughout the migration scenario regarding the engineering solution, and requirements elicitation via stakeholders. The scope of the study regards the time-frame from when the company initiated the switch between the two platforms on a technical and organizational level, considering its effects on the mentioned three key aspects.

It is important to note the difference between the engineering task of the researchers, a proof-of-concept representing a small portion of the entire migration – done in a couple of months, and the company’s main legacy-to-cloud migration – done over the course of a few years. Despite this, the prototype that was developed and communication with various stakeholders at the company helped the researchers understand the multiple reasons behind wanting to undertake such a migration. This knowledge and experience allowed the researchers to thoroughly study the trade-offs between the requirements, technical limitations and development efforts, and effectively relate and connect the findings to research conducted in this thesis. The main focus will therefore be on the research methods implemented, however the engineering task offers a unique opportunity to add further validity to the data captured as it was done keeping in mind such concepts and how they related back to the entirety of the migration and the research goals. The research questions are as follows and are elaborated in the *Research Purpose* and subsequent sections.

- **RQ1** What are the trade-offs between existing requirements, technical limitations and development efforts in a cloud-based migration project from one system to another?
- **RQ2** How do business requirements evolve based on

technical limitations that may arise?

II. Case Description

WirelessCar, the company in focus, is a software company part of Volvo Group – that *"connects 2 million vehicles in more than 50 countries"* by providing *"manufacturers of cars and commercial vehicles with customized telematics services to end-customers"* [10]. The platform provides a multitude of services such as remote diagnostics, vehicle & user service provisioning, vehicle monitoring, geo-fencing, infotainment and remote control of vehicles services such as climate control, journey logs and remote dashboards [11]. These services can be provided and have various use cases for several types of users in the platform, however only two types of users will be mentioned as they are directly relevant to this study and the engineering task worked on by the researchers.

- **Drivers:** The drivers of vehicles where the telematics system is installed and connected to the WirelessCar system servers. All drivers are subscribed to the WirelessCar telematics service upon purchase of a vehicle connected to the platform.
- **OEMs:** The vehicle manufacturers or car dealerships that purchase the telematics platform and use it to monitor their car fleets. Car fleets are all the vehicles owned or leased by the manufacturers or dealers.

For the past three to four years WirelessCar have been migrating their previous technical solution from an on-the-premises data center to the cloud on Amazon Web Services (AWS), using a *"Pay-Per-Usage"* payment model that delivers "...secure cloud services, computing power, database storage, content delivery..." [12]. The main reasons being scalability, low downtime, avoidance of incurring costs for overcapacity as well as higher abstraction in terms of development effort. As infrastructure in the legacy system had to be planned and designed ahead of time, it often lead to cases of overcapacity in terms of processing power. Thus, a major driver behind adopting AWS was that the server instances that handled the business logic could easily be scaled dynamically based on the current business needs, instead of having to be provisioned for, also leading to cheaper run-time costs.

The company originally sought after students to implement a business analysis reporting prototype proof-of-concept to verify and ensure that their undergoing migration would succeed in meeting their business requirements.

A. Proof-of-concept

The engineering task revolved around a main requirement derived from the previous WirelessCar legacy platform, the ability to generate Key Performance Indicator (KPI) reports that would help the *OEM* track their progress towards their established business goals, by providing insights to driver behaviour and service subscriptions via easy-to-use dashboards and interactive visualizations [13]. These reports were intended for business analysis and played a major role in the company's business case. In the legacy platform the business reporting tool and data warehouse were the only components deployed as cloud services on the Microsoft Azure platform. The proof-of-concept aimed to prove whether an Amazon service called *QuickSight*, their proprietary business analytics service, could generate KPI reports as per the business requirements of the legacy system. One of these requirements dealt with measuring the telematics service invocations over time and aggregating those numbers into charts. The *OEMs* whose cars utilized this service, had a limited amount of invocations per month in the form of credits. Credit limits were measured against the recorded amount of invocations used during that period and WirelessCar would see if an *OEM* went above or below their credit limit and bill them accordingly – this was planned to be the future payment model for the telematics service after the migration.

B. Engineering Task

There are a number of cloud AWS services and a runtime system mentioned in the description of this engineering task and they are elaborated as follows:

- **AWS Kinesis** [14], is a service that functions as a unified pipeline for streaming data to by data producers such as social media feeds or Internet of Things (IoT) application data.
- **AWS Lambda** [15], is a service that enables running code in a cloud environment without any prior configuration or server management. The user that runs the code only pays for the actual computation time.
- **AWS QuickSight** [16], is a business analytics and visualization tool that supports a business user with intelligence to make informed decisions about their business goals.
- **Node.js**, according to the homepage of Node.js, it is a JavaScript runtime built on Chrome's V8 JavaScript engine [17].
- **NPM** Node has a developer community and a package manager called **NPM** that contains the largest selection of JavaScript modules and their corresponding documentation in the world [18].

In order to generate KPI reports, the data had to be randomly generated with timestamps within an adjustable time frame in order to mimic the collection of data over time and how the platform would function in reality, once properly implemented. *Figure* 1 illustrates an overview of the process of generating this data and its resulting extraction and persistence to the database. The content of the data can't be mentioned due to confidentiality agreements, however the protocols and process flows are explained further. The requirements stated that the data could be stored in different sources, both a comma-separated-value (CSV) file and in an SQL database. The generated CSV file was read and JSON objects were generated from each row and encoded in base64 format. The base64 encoded string was injected in an *AWS Kinesis* event, a JSON object with predefined fields.

A *AWS Lambda* handler was invoked, an entry point for executing any **Lambda** function, which parsed the decoded

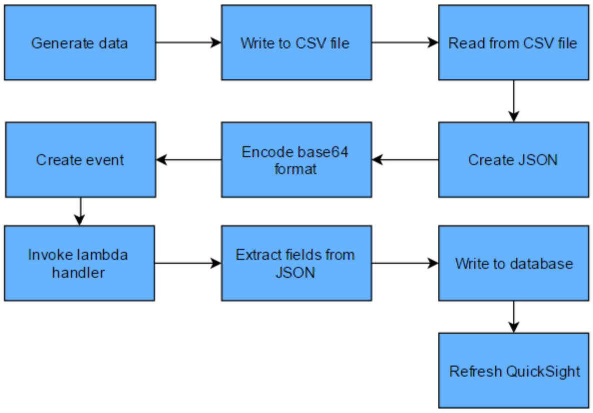


Fig. 1. *Process flowchart of data generation and extraction*

resulting JSON object, writing its data to the database.
QuickSight was refreshed to display the newly generated data in diagrams and charts. The design, development and testing of the functionality of the prototype were enacted solely by the researchers using JavaScript. All of the development was done locally on laptops using *Node.js* as a runtime. Any libraries that were needed to implement the prototype were downloaded and added to the runtime via **npm**. The only actual service that was used in the cloud was *QuickSight*.
The mention of the other AWS services and the usage of their templates and entry points were to try and match the constraints and process flows of the AWS platform as closely as possibly without actually using them which would result in spending too much time with configuration. This was a requirement in order to prove that the data could successfully flow through the entire platform and minimize the difference between the proof-of-concept and the actual future implementations of this task by the developers of the company.
Once the data was loaded into database, the data sources had to be imported via the *QuickSight* dashboard and credentials had to be provided. In order to generate a business analysis with diagrams, correct fields had to be chosen from the imported database tables and combined to fulfill the necessary business requirement in order to easily visualize the data.

C. Research Purpose

The purpose of this research paper revolves around the company's proof-of-concept engineering task in tandem with the derived requirement engineering implications that change throughout the migration. These implications reflect upon the aforementioned research questions which serve as guidelines in understanding which research aspects play an important role and what conclusions can be drawn.

The main research question, ***RQ1***, depicts the relationship between the three key aspects related to the migration process alongside the respective stakeholders ranging from developers, business owners and delivery managers. These components are: *Requirements*, *Technical Limitations* and *Development Efforts*.

- *Requirements* are the functional needs of a program or service in order to perform their intended purpose [19]. They form the backbone of a company's business goals and thus create the appropriate solution for its customers. These include statements regarding the services which the system should provide, how the system should react to particular inputs and how the system should behave in particular situations [20].
- *Technical Limitations* outline the inability of the hardware or software to perform its intended function or the constraints that are imposed on a system due to such a limit that will prevent it from working at its intended capacity. These constraints may involve the limitations of architectural designs or flows for implementing certain requirements and incompatibilities between different communication protocols.
- *Development Efforts* is a broad term. Software development refers to the process of programming, designing, documenting, testing, bug fixing and maintaining a software product. In the scope of this research it additionally involves the management and workflow of software developers, testers and architects in terms of the actual effort expedited. The efforts concern mainly the productivity and ease of development.

The sub research question, ***RQ2***, serves as a follow-up question based on the technical limitations and their effects on the business requirements rather than requirements in general which the main research question focuses on. This question is considered because one of the company's main focal points, just like any business, is to successfully present customers with an affordable business proposition and solution – consequently we look at how the technical implementation may affect the business outcomes, if any. Development efforts were not seen as relevant in this research question due to the fact that business requirements are high level profit driven goals that can't be directly decomposed in user stories and their corresponding functional requirements. Due to their general and abstract nature one can't enable traceability from such requirements to other phases of software development such as testing, design, development and maintenance. However technical limits may be small or large, such as lack of processing power or limitations in third party applications and can thus be related to business drivers.

Figure 2 outlines the tri-point relationship between the three key aspects via dotted lines. Each component, as described in the following research questions, is susceptible to compromises as a result of the company's and its customer's relationship and consequent requirements. That is to say, each aspect is prone to facing certain trade-offs in the event of another requiring more or less focus depending on company's situation. A concrete example of a trade-off would be if there is a requirement to view data in real time but the architecture that processes the data executes a batch process once a day, hence the architecture would interfere with the requirement. A trade-off in this case would possibly involve modifying the requirement to allow semi-real time data viewing (generate data more than once per day, but not in real time) or spending additional resources on architects and developers to implement such a real time system. Tying this example and previous points together, the triangular relationship is therefore not static but rather dynamic with respect to the company's direction. Thus the interest to the research community is a study in how much leeway there is in compromising between these aspects and to what extent such an aspect can be assigned a higher priority than another. The results can highlight crucial trade-offs and mitigations that will serve as a frame of reference for other researchers when studying cloud migration or possibly, system architects that are evaluating the risks and benefits of such a migration.

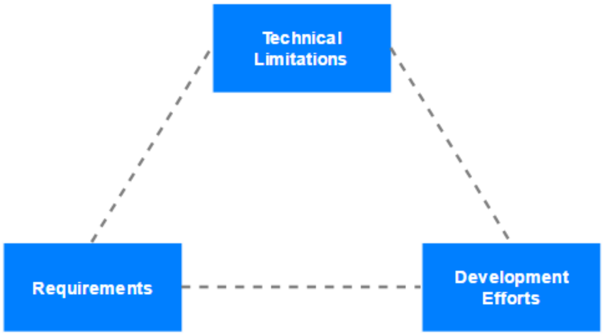


Fig. 2. *Requirements, Technical Limitations, Development Efforts*

III. Background and Related Work

A. Background

Due to the relatively young nature of cloud computing services, few studies exist that comprehensively focus on cloud migrations in an requirements engineering context [21]. The existing ones, however, tie into this research by touching upon three themes that intertwine and are brought up in the research questions - requirements, technical implications and development efforts. When companies adopt cloud services and researchers conduct studies about these migrations, these themes are important to consider according to the numerous pieces of literature. Consequently, being able to conduct studies and collect results about these themes allow both parties (companies and researchers) to better understand the entire process of migrations and its implications to reach better solutions. In terms of research, the researchers are able to determine to what extent these themes affected the company throughout the migration and relate the results back to the community and literature that discuss these aspects, strengthening the bridge between the reality of migration scenarios and what current research talks about.

B. Related Work

The related work will be focus on the three key aspects regarding requirements, technical limitations and development effort in order to comprehensively connect literature with this study.

From a requirements perspective, Zardari and Rami point out that one should *"understand customer requirements and implement them into appropriate requirements for the solution to be realized*" [22], much like what WirelessCar is doing to determine which cloud solution works best for its customers' requirements and how this research looks into the trade-offs between these requirements in order to create a comprehensive proof-of-concept and answer the research questions. Westfall additionally states that companies should consider the *"architectural, legal and pricing implications"* before moving into development [23] [21] which in turn, result in certain requirements being focused on more or less to accommodate for what the company deems important and relevant, thus adding to the research's trade-offs question.

From a technical perspective, the literature heavily emphasizes, as Ramachandran and Mahmood discuss, the importance of security in terms of encryption *"for protection and confidentiality"* [20] of user data. As more users adopt cloud services, more data is distributed facing risks such as *"loss of sensitive data, theft and data integrity"* [24]. Consequently, service providers and companies using those services have to add security requirements based off of these implications as the lack of insight and control over the operations and functions of third-party cloud systems can raise privacy concerns regarding user data integrity and security since services and data are stored in third-party locations instead of on-the-premises, removing customer control and leading to doubts about the security of provider's applications. Additionally, Zalazar and Ballejos touch upon the *"availability, performance, scalability and usability"* of a service, during and prior implementation [20]. Such technical implications are important to look into because they relate to the company's requirements and affect their team's development efforts. Availability because the service needs to have a high percentage of uptime, performance because the service needs to handle dynamic quantities of data and perform computations in the least amount of time possible, scalability because data storage needs to increase or decrease dynamically, and usability because customers should be able to easily access their profiles and view KPI reports. From a development efforts perspective, one should consider how requirements *"change over time"* as they urge organizations to *"elicit, analyze, specify, verify and manage them"* thus affecting the development effort, as teams needs to accommodate for various requirements and implement them effectively*"in a systematic way"* [20]. A study showed that *"almost 48% of the problems in software development could be traced back to poor requirements engineering"* [25], which highlights the importance of considering such mentioned factors as to mitigate shortcomings and organize the development effort in an efficient way to tackle important requirements whilst considering their technical implications. The fact that cloud services encompass a growing variety of users, makes the use of traditional requirements elicitation frameworks quite challenging [26] as the scalability, flexibility and wide geographical distribution of such services can often lead customers changing their needs, rendering requirements imprecise or unstable. Thus, by analyzing the trade-offs and the importance of requirements, and their consequent development effort effects, conclusions can be drawn regarding how such a migration process is formed, what aspects are taken into account and to what extent. The literature's combination of the three themes adhere to this case study's focus on their trade-offs with respect to the company. This prompts for an interesting study opportunity to build upon what is already known about these themes from a general context and how a company considers these aspects and prioritizes them based on its vision, needs and customers.

IV. Methodology

A. Research Strategy

As trade-offs between the three broadly defined software engineering aspects (requirements, technical limitations and development efforts) with respect to a specific technical task were looked into, it was concluded that a qualitative approach was needed because trade-offs could only be determined via capturing the underlying motivation and reasons the stakeholders had when prioritizing certain areas of the migration, be they technical or managerial. Consequently, semi-structured interviews were chosen as the appropriate investigation method because both specific and open-ended questions and answers could be elicited, providing optimal amounts objective and subjective data by stimulating natural discussion whilst keeping a fixed scope on certain questions. The interviewing process took place after the majority of the technical proof-of-concept was done, due to the researchers' time constraints. The original subject for the first interview was the company supervisor of the researchers and the delivery manager of one of the AWS software teams. Due to his managerial role and knowledge of all members in the project, the supervisor was asked to provide additional contact information about other stakeholders that could help in further answering the research aims of this study, such as developers or architects.

The data gathered aimed to give comprehensive insight to possible mitigations and compromises that were enacted during the migration in relation to the mentioned key aspects. The qualitative nature of the data and the lack of a given hypothesis supported the use of an inductive approach, which included observations and analysis of patterns of data in order to form theories or explore new phenomenons [27]. The forthcoming sections highlight the team's organization in the company, the design of the interviews and subject sampling.

B. Company Team Organization

WirelessCar aims to deliver a telematics platform solution to several different automotive vehicle manufacturers and distributors, whose names have been left out due to confidentiality. Each software team is responsible for delivering a customized solution to each automotive brand and has a certain structure that consists of: a delivery manager, a solutions architect, developers and testers. In this study, subjects were chosen from a single team and interviewed. The only participant that was not chosen from a team was the architect for the external business analysis tool that was developed in the legacy system.

C. Subjects Description

The five participants of this case study include:

- **Delivery Manager**
- **Customer Solution Architect**
- **Two AWS General Solution Architects & Developers**
- **Azure Solution Architect**

The *Delivery Manager* is responsible for delivering a tailored customer solution of the telematics service. The role entails planning resources for maintenance, development as well as capacity planning. Other responsibilities involve being a Quality Assurance lead and the manager of the architects, business analysts and team leaders.

The *Customer Solution Architect* is responsible for providing and customizing an architectural solution fit for delivery to one of WirelessCar's major customers whose identity cannot be disclosed. The role entails both architectural and technical knowledge regarding the current and previous solution, and the customer's requirements.

The *AWS General Solution Architects & Developer* were two employees 1) primarily a developer for the AWS platform, with limited architectural responsibilities and 2) primarily an architect with some limited developer responsibilities.

The *Azure Solution Architect* is responsible for the original design and development of the legacy WirelessCar business analysis reporting service.

D. Subject Choice Motivation

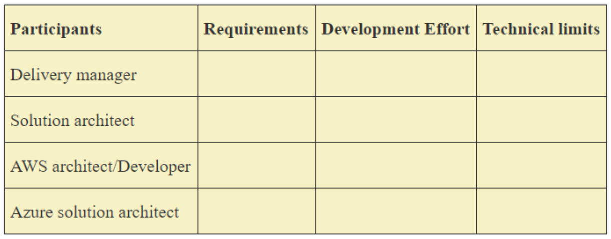
These five stakeholders were chosen because their technical and managerial roles at the company related to the engineering proof-of-concept regarding AWS. Moreover, the Delivery Manager and Customer Solution Architect were automatically assigned to the researcher's thesis via contract from the beginning and thus had a better understanding and connection with the study and its meaning. The interviewees represented different parts of the system and relevant competence or experience to answer the research questions. The *Delivery Manager* of the new AWS architecture was interviewed to answer more managerial and administrative questions regarding requirements and development resources. The *Customer Solution Architect* was interviewed to gain further insight into the technical difficulties and architectural differences between the new and old system. The *Azure Solution Architect* was interviewed in order to gain a more fundamental understanding of the architecture of the previous business reporting platform so that the evolution between the two platforms can be better understood. Additionally, the *AWS General Solution Architects & Developer* were interviewed to gain insight into the complexity of the migration process and development effort. Their unique position offers a perspective that entails both a high-level and low-level view of the system.

E. Interview Questions

The appendix contains all the interview sheets. Questions were formed based on the research aim and modified between each interview depending on the role in the organization of the interviewee. The questions were evaluated by both researchers before they could be approved. Each interview led to the discovery of new themes that either had to be expanded or confirmed from a different interviewee. This led to the revision and addition of new questions during the interview process, to better answer and define unclear themes that were previously extracted. The interviews took place after the proof-of-concept, across a period of weeks with roughly one candidate per week. There was no particular preference of the order the subjects were interviewed in. Each interview's results provided the researchers with more knowledge in certain areas and the next set of questions were formulated with a better grasp.

F. Interview Design

The interviews were designed in such a way to accommodate each interviewee independently with separate interview sessions lasting around an estimated 20 to 30 minutes as not to take up too much of their time and allow the researchers to ask and receive concise questions and answers, respectively. Since all the interviewed subjects were Swedish, the questions were first written in English and then translated into Swedish to avoid a potential language barrier. Consequently the interviews were all held in Swedish. Any potential notes taken during the interviews were in English. A hand-held audio recorder was used for each interview in order to record it in full length and used as reference to refine transcription notes. This removed possible bias by not relying entirely on notes during the interview. Before the interview started, the subject was asked to consent to be recorded and informed of the ethical guidelines that we as researchers would follow when handling the audio recordings. One researcher conducted the interview by explaining the content and asking the questions, while the other researcher took notes on a computer. Before the actual interview started, the purpose of the research and the research questions were iterated. The subject was asked if they understood the goal of the research and if there were no misunderstanding, the interview could proceed.



Participants	Requirements	Development Effort	Technical limits
Delivery manager			
Solution architect			
AWS architect/Developer			
Azure solution architect			

Fig. 3. Tabulation for data categorization

G. Data Collection and Categorization

The first step in collecting the data was to cross reference the field notes with the audio recordings of the interviews. A table was used to categorize common themes from the notes in separated columns as seen in *Figure* 3. Three columns were defined according to the different aspects of the main research question: *Requirements*, *Development Effort* and *Technical Limitations*. These categories encompassed the vision of the research and compared the different stakeholder's views. Four rows represented each interviewee by their company role, allowing the researchers to independently analyze and compile field notes using the mentioned format – summarized in separate tables. Afterwards, the resulting tables were compared and a final table was created containing results that both researchers agreed upon, backed up by the audio recordings. As both researchers were present during all the interviews and listened to the recordings several times afterwards, the validity of the tabulation data was ensured as well as eliminating the need for data harmonization. If something was too ambiguous, the candidates were contacted again via email or at the locale of the offices to clarify. After the first interview, it was realized that the interviewee's role had not been elicited and that the researcher's descriptions were based upon their own interpretation and not the subject's own definition of his/her role. A mail was sent to the subject and they was asked to clarify their role at the company in terms of responsibility and objectives. This question was added in the beginning of all subsequent interviews. Another point was that there was confusion about the scope and content of the business requirements of one of the customer solutions of the WirelessCar platform. The delivery manager was mailed and asked to define these business objectives.

H. Data Synthesis

The three different key aspects of the main research question – requirements, development effort and technical limitations – were combined in pairs yielding all such possible combinations. Additionally to answer the sub research question, business requirements and technical limitations were combined as a single pair. In the end, this yielded a total of four tables, where each table displayed a pair from the combined set. Each table contained two columns as a result of the combined pair and the table highlighted the trade-offs between these two aspects. *Regulations* were also included as an additional key aspect since the findings from the interviews yielded data that could not be categorized in the other aspects. Due to this change in scope, one more table was added.

Regulations generally encompassed governmental, international or judiciary concerns and were found to be of different types and scopes. These are elaborated in the results section. Data from the interviews that could not be regarded as trade-offs between two elements were processed in a separate table referred to as *Causes and Effects*. These had to do with obstacles that were encountered and their mitigations or negative or positive effects in the migration. Thus a total of six tables were compiled and presented in the results. The process of finding the causation to a trade-off between one element and the other were discussed in length by the researchers until a mutual conclusion could be made. For each trade-off, notes from several interviews were analyzed and cross referenced to triangulate the data from different sources and confirm its existence. This occurred when the subject didn't explicitly state that a trade-off had been made between two elements and therefore the researchers had to infer by themselves from the tabulations.

V. Results

This section details the results from the four interviews via tables, explained in the methodology section. The results section is divided into three subsections; one that compares trade-offs between different aspects, a second that presents comparisons and outcomes between other aspects that were unable to be covered under the first subsection, and a third that presents the trade-offs that were encountered in the software prototype developed by the researchers. Every row of each table will be elaborated in greater detail in enumeration.

A. Trade-offs

This section presents results that aim to answer **RQ1** and briefly **RQ2**. *Tables 1, 2, 3* compare the three major aspects. *Table 4* compares *Business Requirements* and *Technical Limitations*.

Requirements	**Technical Limitations**
Less database load and real-time data extraction	Previous batch-driven platform
Streamlined deployment and development	Difference between test, development and production environment

TABLE 1: *Requirements vs. Technical Limitations*

Table 1 compares Requirements and Technical Limitations. The findings were as follows:

1) One of the requirements stated that the databases of the system should encounter less loads and ensure real-time data extraction whilst the previous on-the-premises platform provided a batch-driven solution, that the data and subsequent KPI reports were generated in *batches* or bulks at daily time periods and not in real-time, ranging from *minutes* to *hours*. As a result of the technical limitations on the previous system, a requirements trade-off arose as the company had to accept the technical state of the platform and prioritize that over the mentioned requirement. As of now, however, the migration introduced the aforementioned AWS system which met the this pre-existing requirement, by technically extracting and generating KPI reports in real time and thus removing the need to consider trade-offs between the two aspects.
2) A second requirement stated that the platform should be streamlined in terms of deployment and development, meaning that the entire process became more homogeneous with all the system's components residing in the same environment - ranging from how the developers interacted with the system and how the system functioned across the platform. In contrast, the previous system technically limited the ability for the company to ensure such a streamlined process and thus led to a differentiation between deployment and development. As the system's operations regarding testing, programming, and environment creation became more isolated, the requirements had to adapt and thus a trade-off occurred by prioritizing the technical limitations.

Requirements	**Development Effort**
Multi-tenant Platform (user sharing/access)	Creating automation tools and standardized protocols
Share data with end-user data warehouse	Business analysis solution more simple to maintain
Packaged Service	Outsourcing services to other stakeholders
Encrypt all vehicle telemetry data	Evaluate the scope of all sensitive data

TABLE 2: *Requirements vs. Development Efforts*

Table 2 compares Requirements and Development Efforts. The findings were as follows:

1) A multi-tenant platform is defined as an architecture where a single instance of software application is shared between multiple customers and customers have dedicated access roles so that they cannot affect or manipulate each other's data accidentally or intentionally [28]. The positive benefit is that only one deployment platform is needed to be maintained and updated which cuts down costs and configuration considerably. However as WirelessCar aims towards such an architecture, the scalability of AWS poses challenges as a large quantity of instances and services are created. These instances have to communicate in one single flow and in order to fulfill the demand for multi-tenancy, developers had to face the challenge of creating automated tools and standardized protocols that would enable a smooth and consistent flow of data without manual configuration. This can be seen as a trade-off as the requirement is one of their main business drivers.
2) With the introduction of AWS, multiple customers required that their data be shared with them as opposed to WirelessCar keeping all the data. This requirements change affected the development effort in such a way that the analysis and maintenance of that user's data became much simpler as the customer had the ability to handle certain portions of the services such as data analysis and report generation. This trade-off had a positive outcome as a new requirement led to shared access, satisfying business interests, shared data handling and minimizing development effort in terms of having teams spend less time worrying and focusing on such tasks.
3) One of the key requirements involved being able to run all the needed services for the solution on one platform.

However, the previous solution forced a trade-off as the the development effort had to change in such a way that the in-house developers could not work on certain services on the Azure platform. External stakeholders, independent of the company, had to be hired to work on certain tasks that were vital to the delivery of the solution. This increased the development complexity, as the effort was divided, external and third-party reliant. The switch to AWS offered "packaged" services so that all the components the development team needed to fulfill the company requirements could be worked on (technically) and managed under one system, as opposed being externalized. This positively affected the development effort as third parties were no longer needed, thus greatly reducing development complexity and dependency with other entities.

4) Due to the fact that the data was sensitive and particular to a customer, the company couldn't afford any leaks or violations of any potential upcoming regulations regarding data security and privacy. Consequently, this became an important requirement and forced a trade-off that affected the development effort in such a way that the handling of all the data on an architectural, technical and organizational level had to redefined in terms of scope. Thus, the development effort increased in order to accommodate for such a change. Moreover, since the data logically passed through the entire system from the raw data retrieval coming from vehicles all the way to data presentation in the form of reports, the development teams had to look at the entire process and ensure that this requirement was met at every stage.

Development Efforts	**Technical Limitations**
Hindered the dev process	Long downtime in case of failure
Caused a redesign of data flows and modeling	Lower DB read capacity (AWS)
Issues when doing minimal migration, but not with new components	Multi-cast communication network
Large scale refactoring and redesign of components	Decomposition of components into microservices
Stalls and conflicts in development time	Dependencies among components
Learning and applying AWS services	Event-driven architecture

TABLE 3: *Development Efforts vs. Technical Limitations*

Table 3 compares Development Efforts and Technical Limitations. The findings were as follows:

1) The previous on-the-premises system had a technical limitation that forced development to a halt if the servers encountered down-time or failure. As a result, the entire development effort across all services of the platform was hindered until the issue was resolved.
2) The previous system was able to support large database capacities as well as subsequent read and writes made to the entire system. This boded well with the previous development, as one did not have to consider the fact that their could exist a potential over-flow of data read and writes, even in a worst-case scenario. The introduction of the AWS system, however, lowered the database read capacity by a factor of 30. Consequently, this technical limitation had to be considered and the development effort was negatively affected in such a way that the development data flows had to be redesigned and remodelled. That is to say, the previous development effort's design and way of working could not match the new system's technical limitation and thus had to face a trade-off, and undergo overhaul.
3) The previous solution technically communicated between all the components in a multicast fashion, meaning that any component in the system could communicate with several others - ensuring a one-to-many or many-to-many relationship. Although the new cloud-based system did contain the multicast functionality needed, it's technical properties differentiated from a on-the-premises system in terms of how data was handled and dispersed. This led to a trade-off with respect to the development effort, as certain parts of the migration faced complications when migrating previously existing components, especially when old components from the previous system had to match the new system's technical multicast requirements. However new components had no such issues. Thus, the development time spent on ensuring compatibility of previous forms of data and considering the new technical properties, increase.
4) In order to fully utilize the scalability and modularity of AWS instances, a design choice was made to decompose the existing monolithic system in smaller components with defined sets of responsibility. However due to the complexity of the legacy system, a costly refactoring process was enacted that would stretch both development time and costs.
5) Dependencies between certain components and data flows stalled the software teams at some points, since a team could not begin development of one component until its dependency was resolved.
6) Due to the system change to an event-driven architecture that processes data events, the developers had to educate themselves about the AWS in order to apply the design patterns in their component development.

Business Requirements	**Technical Limitations**
Share data with end-user data warehouse	User data was private and stored internally on-the-premises

TABLE 4: *Business Requirements vs. Technical Limitations*

Table 4 compares Business Requirements and Technical Limitations. The findings were as follows:

1) A new business requirement was added to allow some of the data from the platform service to be shared with an OEM. Since the OEM had their own data warehouse centres, it could generate their own KPI reports and thus alleviate some of the load and requirements on WirelessCar's business analysis tools. This requirement was unable to be implemented previously, due to the fact that all data was stored internally in WirelessCar's data warehouses which was private and inaccessible to an external actor.

B. Comparisons and Outcomes

This section presents results that couldn't be regarded as trade-offs and are instead comparisons between different aspects as well as obstacles that lead to certain outcomes. Some themes that were discovered in the data synthesis could not be regarded as trade-offs and were of differing nature. Hence it was decided that a general table consisting of comparisons between different outcomes would fully encompass all the relevant research results. *Table 5* deals with *Regulations* and the effects they had on the development of the platform and the business model. *Table 6* displays *Causes and Effects* that were either obstacles and their mitigations or certain causes and their resulting outcomes.

Regulations	**Effects**
EU data privacy directive	Large scale refactoring and redesign of components
Azure licensing model	Complex user sharing
China AWS regulation	Some Amazon services unavailable

TABLE 5: *Regulations and their effects*

Table 5 compares Regulations and their effect on the migration. The findings were as follows:

1) A new EU data privacy protection directive dictates that individuals in EU should have control over their personal data and is meant to be enacted May 2018. This had led to new requirements for all of WirelessCar's customer solutions to involve the ability to trace all records and meta data for a subscribed user. Therefore if the user discontinues their subscription by for example selling their car, they have the right to require that all their data gets wiped out. This obviously requires a cumbersome redesign and analysis of data models to facilitate this policy. However it was recorded that this policy was known for a period of time, and was therefore not a result of the migration.
2) The previous business analysis service developed in Azure was more complex and had the capability to theoretically generate any type of KPI report from the data warehouse. However due to an expensive licensing model and difficulties sharing the reports as a Software-As-A-Service, which even involved a judicial meeting between different company representative lawyers, the decision was made to create a new and simpler solution that could still be offered as a service. Hence AWS QuickSight was evaluated as a possible solution as the rest of their system was already in a migration process to the AWS cloud.
3) WirelessCar offers their platform service in China, but due to governmental regulations in China, some AWS products are not available and cause a severe difficulty in delivering a viable solution.

Cause	**Effect**
Azure business reports hard to sell due to licensing	Major driver behind switching reporting tool to AWS
Microsoft a competitor in vehicle telematics	Major driver behind choosing AWS over Azure
Azure had ability to generate any type of KPI report	Multitude of requirements which added to complexity
Product development progresses time-wise	Developers becomes more familiar, experienced and comfortable with the platform

TABLE 6: *Comparisons with Causes and Effects*

Table 6 is a collection of causes and effects or *comparisons*, represented by the columns. These could not be placed under any of the other sections because they depicted events the company had considered prior to any potential upcoming trade-offs or simply did not fit in elsewhere.

1) A primary reason in switching to a platform was the fact that the Azure business reports became hard to sell to various customers due to licensing issues which cannot be disclosed. Thus, a new platform that could generate such reports without licensing hindrance was highly favoured and evaluated in the scope of this thesis.
2) Microsoft proved to be a competitor to WirelessCar as they also got in the business of providing vehicle telematics solutions. Therefore, this was a major factor behind choosing AWS as a cloud service provider for the entirety of their migration, regardless of the fact that WirelessCar had the external business reporting tool already implemented in the Azure platform, which could otherwise have become the chosen provider.
3) All the interviewees pointed out that Azure was an advanced platform in terms of what it offered and how one could use the services. With respect to KPI reports, Azure had the ability to generate any type of report at a

more complex level which increased the number of requirements the company faced both on an organizational and development plane. The increased complexity led to further analysis of the Azure functionality and how one could properly create an acceptable requirement.

4) As the development throughout the migration progressed, the development teams and developers themselves became more familiar, comfortable and knowledgeable with the platforms, specially the new one. Consequently, the company realized that a migration to a new system which was more “packaged”, efficient and scalable helped the development effort become more effective and in turn, mitigate potential risks or trade-offs.

C. Engineering Task Trade-Offs

This section portrays the trade-offs between aspects that were encountered as the result of developing the prototype.

Requirements	**Technical Limitations**
Multiple data sources	N/A
Mimic real scenario data flow	N/A
Ability to share reports	Can’t share reports outside existing QuickSight account
Limit user access to analysis	N/A
Instant QuickSight data source refresh	Can’t update QuickSight based on database refresh
Customer credit comparison	Difficulty viewing credits and invocations in same diagram
Scroll zoom scale on X-axis	Can’t use adjustable zoom on X-axis

TABLE 7: *Requirements and technical impediments*

Table 7 highlights the requirements that were captured by the researchers’ company supervisors and the resulting technical limitations that occurred due to the inherent limitations of *QuickSight*. **N/A** indicates that no technical limit was found for that particular requirement. The following list details the requirements and the trade-offs between them, and the technical limits as well as the development efforts (each enumeration relates to the number of the row):

1) Multiple type of data sources were required to be imported in *QuickSight*. This posed no problems as the service was designed to handle multitude of data types and sources such as CSV, Excel and CLF (common log format) files, and SQL databases as well as PostgreSQL. The development effort was the only factor impacted, as additional data sources had to be generated in the prototype.
2) The researchers attempted to mimic the actual data process flow in the Amazon cloud platform the way it would be enacted according to the architecture designed by WirelessCar. Since all the templates for the different type of data events were well-documented and easily available, there was no greater effort spent on implementing these.
3) Due to the inherent security constraints in the service, business analyses and dashboards were unable to be shared with users in a different AWS account. In such a scenario, a mitigation was found that involved exporting the diagrams as CSV files from one account and importing it into another.
4) The service has various user access levels that can be assigned to users easily from the dashboard and this requirement can therefore be easily implemented.
5) One of the major drivers behind the migration was to enable instant generation of KPI reports whenever a new event triggered an update in the database. Data was originally imported into QuickSight’s own proprietary calculation engine called *SPICE* that aims to optimize data aggregation for visualization in order to increase performance . However *SPICE* couldn’t update its database sources in real time. Currently it can only refresh them manually or by scheduling the refresh rate at a minimal interval period of once every 24 hours. In order to fulfill this vital requirement, database sources were queried directly instead of being imported. However this will lead to performance degradation in time, which is an unfortunate trade-off that has to be made due to the inherent technical limitations of *SPICE*.
6) The future business model of WirelessCar involves assigning customers a credit limit each month that is measured against the invocations of the telematics service their car fleet has made during the period of a month. The diagrams in *QuickSight* were thus supposed to visualize the current service usage this month per customer and their credit limit for the same month, in order to see if it exceeds the limit or not and bill them accordingly. This was a challenging requirement. A separate database table was made that held information about fictional customer’s credit limits and was joined with the service invocations table in the data set in *QuickSight*. However the design of the user interface for data sets prevented the tables to be used separately and thus no visualization could be made as the diagrams additionally had a limited amount of fields that could be utilized. This has not yet been resolved and is considered a requirement of utmost importance which can’t be traded off.
7) No adjustable scroll zoom scale exists on the X-axis in the diagrams. However it was mitigated by using a filter instead that can set the range of data between two different dates, the minimal interval being a day. This requirement was associated with usability preferences and thus considered marginally important and therefore the trade-off was to change the requirement in order to overcome the technical limit.

VI. Discussion and Future Work

Due to the extensive scope of this research revolving around a large-scale system migration, there were difficulties in identifying trade-offs between the three key aspects. Eliciting clear details about specific requirements and their consequent trade-offs proved challenging partly due to the fact that the researchers only had a couple of months to both create a proof-of-concept, conduct research and write a thesis. Moreover, considering that the company had already spent a couple of years undertaking such a migration which logically involved many changes with respect to all the three aspects, it was difficult to compile a list of ”all” the aspects that were relevant to the research, as changes could not have been documented or even forgotten by stakeholders at the company. As a result, the researchers tried to utilize their time as efficiently as possible by talking to the most relevant stakeholders mentioned in the interview section and asking questions that would specifically answer the research questions.

The results showed that more trade-offs arose when comparing the *Development Efforts* and *Technical Limitations* shown in *Table* 3 where the development efforts suffered as a result of the technical limits from the legacy and Azure systems. The requirements mentioned in *Table* 1 highlight the importance of real time data extraction and streamlined deployment, as such, the technical limits forced the company and its the teams to adhere to those limits and continue working with those setbacks – that real time data computing was not possible nor development and deployment in a streamlined fashion. Thus, the development effort had to be redesigned several times and distributed to third-party actors outside the company. Likewise, the requirements that changed based off of the technical limitations affected the development effort as shown in *Table* 2. The company prioritized resources towards the development effort all the way through migrating from the legacy to cloud system. Rather than compromising on any requirements, the development efforts were increased to match the requirements that defined the company’s vision and business objectives, vital in sustaining their profit model.

The literature in the *Background and Related Work* section described the three themes that have been mentioned throughout this paper: requirements, technical limitations and development efforts. Each theme reflected upon the findings of this study as seen throughout the *Results* section with each table. The emphasis on security and data privacy corresponded to the concerns that are often faced about security in the cloud [20] [29]. The most probable (the company has not yet implemented it) solution in this case study was to evaluate the scope of all data to be encrypted and make sure it was end-to-end encrypted for maximum protection. As for the choice of cloud providers, the results showed that scalability and cost saving were the most prominent, as highlighted in the literature [30] [20]. However, the main reason for migrating to AWS was simply due to the fact that Microsoft was a competitor in the same business as WirelessCar, which could lead to a severe conflict of interest over time - adding an additional perspective to how cloud providers might be chosen.

The vast array of global customers that WirelessCar is serving caused difficulties in capturing requirements while at the same time building a multi-tenant platform. This is due to the different needs and expectations of stakeholders that makes the requirements engineering process unstable and imprecise, as mentioned previously in the background section [26].

When the prototype was developed, the trade-offs became increasingly more apparent and more easily visualized. This was due to the very clearly defined requirements that were received from the company supervisor and by developing the prototype hands-on, a more intimate understanding was achieved of its technical limitations. Since the prototype involved a vital component in the business model of WirelessCar there were more trade-offs done in favor of the core business requirements that were mentioned in the case study description. Requirements that were of minimal importance to the business objectives were considered less critical and more open to compromise. Having a more complex case study with a larger proof-of-concept or several separate smaller ones could add validity to the data collected in the research in the future.

As this study was company-specific, within a short time frame and distributed in terms of a technical and research task, not enough data was gathered to substantially conclude that the results were specific to a legacy-to-cloud migration nor to automotive industry in general. However, the combination of the study’s results with respect to the interviews and general discussions with the researcher’s company supervisors, and literature analysis, provided insight into important factors that a company with an automotive-industry focus consider whilst undertaking such a large-scale migration. These factors, as a result, were largely specific to this company and a more thorough study would have to be done to concretely determine if such factors apply to other companies with the exactly or roughly the same industry-focus and migration task.

A. Limitation of data

Since the migration project is unfinished as of yet, certain requirements haven’t been fulfilled and additional requirements might be added at a later date. The migration is meant to be a proof-of-concept in its current state and therefore there is a possibility that not all technical limits have been discovered yet or that future requirements can’t be assessed in this research study.

B. Validity Threats

External validity – This research focused on a single case of a migration from a legacy system to a single cloud service provider. Thus, it was hard to generalize and deduce whether the trade-offs and difficulties encountered were due to the limits of the cloud services provided or the actual complexity of the migration. More case studies of migrations to AWS with companies that have similar objectives as WirelessCar might be able to consolidate or disprove the findings in this thesis. However it was also noticed that themes mentioned in the background appeared throughout the interviews, especially the emphasis on scalability, security and difficulty in capturing requirements. This may prove that companies set on performing a legacy-to-cloud migration all face similar issues.

Construct validity – posed an issue during this study as there was confusion among the researchers in the initial interviews regarding the nature of the migration. It was thought that the entire system migration was cloud-to-cloud instead of legacy-to-cloud. This confusion was due to excessive focus on the technical prototype, which was actually being migrated from cloud-to-cloud (Azure to AWS). Additionally no pilot study of the interviews was conducted due to time constraints. Both of these factors could have led to formulating interview questions that didn't fully capture the intended data. These threats were however mitigated by constantly revising the questions in order to extract the most relevant information from the interviewees and by asking follow-up questions and seek further clarification from concepts or points that were missed or misunderstood in previous interviews via emailing or consulting the stakeholders at their offices. The large scope of this study made it quite difficult to be able to record detailed requirements and their corresponding technical limits. There might have been no observable technical limitations on a higher level, but on a lower level they might have been more frequent and obstructive in their nature. The engineering task proved the ease of capturing the trade-offs when they were limited to a very specific component, compared to the challenge in doing the same in the interviews. A future improvement might consider decomposing the case study in several smaller parts that are all researched in parallel and combined in the end.

VII. Conclusion

Few technical limits were recorded from the interviews and the most prominent cause was that the architects evaluated the limitations beforehand through feasibility studies and exhaustive comparisons between different cloud service providers. Most limitations were probably due to environmental changes such as government policies and business objectives. One can draw the conclusion that as a cloud platform's hardware and software API specification remains relatively stable and backwards-compatible over time, the environment in which they operate is quite unstable and prone to drastic changes. As it can be quite challenging to predict environmental changes due the unpredictability and complexity of the world, it is recommended to look at previous similar cases to gain insight and to have a business model or architecture that easily enables traceability and change requests.

VIII. Acknowledgements

We would like to thank our supervisor, Jan-Philipp Steghöfer, at the University of Gothenburg and WirelessCar for their invaluable knowledge and help.

References

[1] S. Marston, Z. Li, S. Bandyopsjwadhyay, J. Zhang, and A. Ghalsasi, "Cloud computing - The business perspective," *Decision Support Systems*, vol. 51, no. 1, pp. 176–189, 2011.

[2] K. Weins, "Cloud Computing Trends: 2016 State of the Cloud Survey," 2016.

[3] Fredric Paul, "The cloud continues to rise—and fast," 2017, tECH-WATCH, Network World.

[4] D. Zissis and D. Lekkas, "Addressing cloud computing security issues," *Future Generation Computer Systems*, vol. 28, no. 3, pp. 583–592, 2012.

[5] R. Buyya, R. Buyya, C. S. Yeo, C. S. Yeo, S. Venugopal, S. Venugopal, J. Broberg, J. Broberg, I. Brandic, and I. Brandic, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Future Generation Computer Systems*, vol. 25, no. June 2009, p. 17, 2009.

[6] Q. Zhang, L. Cheng, and R. Boutaba, "Cloud computing: State-of-the-art and research challenges," *Journal of Internet Services and Applications*, vol. 1, no. 1, pp. 7–18, 2010.

[7] https://www.ibm.com/cloud-computing/learn-more/what-is-cloud-computing/, accessed: 2017-05-22.

[8] P. Jamshidi, A. Ahmad, and C. Pahl, "Cloud migration research: A systematic review," *IEEE Transactions on Cloud Computing*, vol. 1, no. 2, pp. 142–157, 2013.

[9] A. Khajeh-Hosseini, D. Greenwood, and I. Sommerville, "Cloud migration: A case study of migrating an enterprise it system to iaas," 2010, pp. 450–457.

[10] http://wirelesscar.com/, accessed: 2017-05-09.

[11] http://wirelesscar.com/our-services/, accessed: 2017-05-23.

[12] https://aws.amazon.com/what-is-aws/, accessed: 2017-05-02.

[13] https://www.clearpointstrategy.com/what-is-a-kpi-report-how-do-i-create-one/, accessed: 2017-05-09.

[14] https://aws.amazon.com/kinesis/streams/details/, accessed: 2017-05-02.

[15] https://aws.amazon.com/lambda/, accessed: 2017-05-02.

[16] http://docs.aws.amazon.com/quicksight/latest/user/welcome.html, accessed: 2017-05-02.

[17] https://nodejs.org/en/, accessed: 2017-05-26.

[18] https://www.npmjs.com/, accessed: 2017-05-26.

[19] G. Kotonya and I. Sommerville, *Requirements Engineering: Processes and Techniques*, 1st ed. Wiley Publishing, 1998.

[20] Z. M. Muthu Ramachandran, "Requirements Engineering for Service and Cloud Computing," 2017.

[21] H. Schrödl and S. Wind, "Requirements Engineering for Cloud Computing," *Journal of Communication and Computer 8*, no. January 2011, pp. 404–415, 2011.

[22] S. Zardari and R. Bahsoon, "Cloud adoption: A goal-oriented requirements engineering approach," *Proceedings - International Conference on Software Engineering*, pp. 29–35, 2011.

[23] L. Westfall, "Software Requirements Engineering: What, Why, Who, When, and How," *ASQs Software Quality Professional Journal*, pp. 9–15, 2006.

[24] B. Joshi, B. Joshi, and K. Rani, "Proceedings of the 6th International Conference on Communication Systems and Networks," *Proceedings of the 6th International Conference on Communication Systems and Networks*, pp. 175–182, 2014.

[25] The Standish Group, "The Standish group: the chaos report," *Project Smart*, p. 16, 2014.

[26] M. E. Rana, J. Dauren, and S. Kumaran, "An improved requirements engineering framework for cloud based application development," 2015, pp. 702–709.

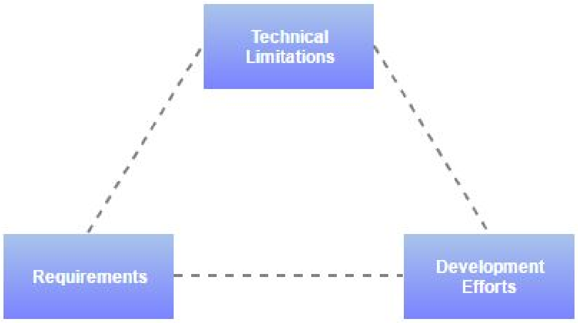
[27] W. Goddard and S. Melville, *Research Methodology: An Introduction*, 2nd ed. Blackwell Publishing, 2004.

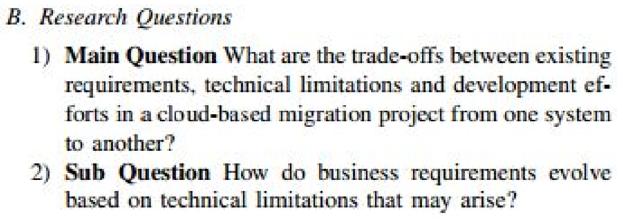
[28] https://msdn.microsoft.com/en-us/library/aa479086.aspx, accessed: 2017-05-23.

[29] C. Kalloniatis, H. Mouratidis, and S. Islam, "Evaluating cloud deployment scenarios based on security and privacy requirements," *Requirements Engineering*, vol. 18, pp. 299–319, 2013.

[30] S. Zardari and R. Bahsoon, "Cloud adoption: A goal-oriented requirements engineering approach." New York, NY, USA: ACM, 2011, pp. 29–35.

Requirements Engineering Interview
Delivery Manager
2017-05-08





B. Research Questions

1) **Main Question** What are the trade-offs between existing requirements, technical limitations and development efforts in a cloud-based migration project from one system to another?
2) **Sub Question** How do business requirements evolve based on technical limitations that may arise?

1. Business perspective:

- What were important aspects to consider when choosing a different cloud platform?
  - FQ: Why Amazon Web Services and Quicksight?
- What customer or business requirements led to a new system being desired?

2. Requirements and technical limits:

- Was there a business requirement(s) that was limited by a technical limitation?
  - FQ:Which technical limitations added or changed what business requirements?
- What requirements remained the same from the previous system? Consequently, what requirement was too important to be compromised with?

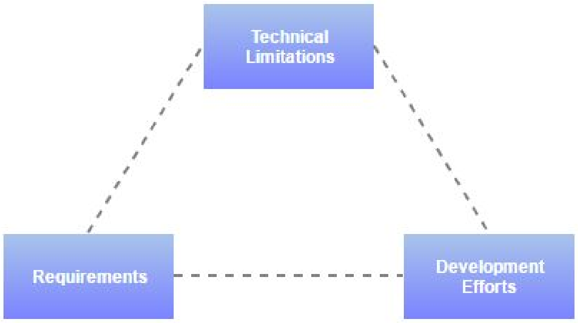
3. Development efforts:

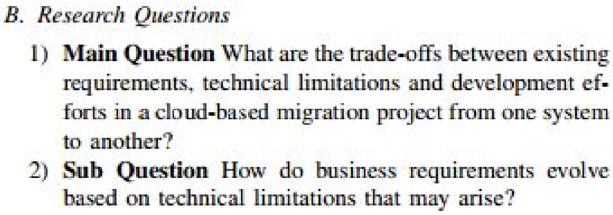
- How was the development effort affected based on the requirements that were added or changed?
- What are the greatest challenges the development team has faced in migrating the system?
- What part of the development plan took most time/effort to produce?
- What mitigation plans did you use to minimize development effort in the migration? For example, reuse of components, using same consultants as previous project or maybe similar architectural patterns?

4. Considering existing requirements, technical limitations and development efforts:

- When planning for the migration, what compromises had to be made between development resources/effort/time and requirements?
  - FQ: Did requirements have to be revised in order to accommodate existing development resources?
  - FQ: Were additional development resources added to realize the new/changed business requirements?

Requirements Engineering Interview
Customer Solution Architect
2017-05-10





B. Research Questions

1) **Main Question** What are the trade-offs between existing requirements, technical limitations and development efforts in a cloud-based migration project from one system to another?
2) **Sub Question** How do business requirements evolve based on technical limitations that may arise?

1. Architecture perspective:

- What are the main differences between the previous Azure architecture and the new AWS one?
- What are the reasons for a different architecture?

2. Requirements and technical limits:

- Have there been any technical limitations that have impeded the migration process?
- (Follow up question if he says there are technical limits) What requirements can't be implemented due to current or upcoming technical limitations?
- How do you ensure technical limitations do not arise or are mitigated?

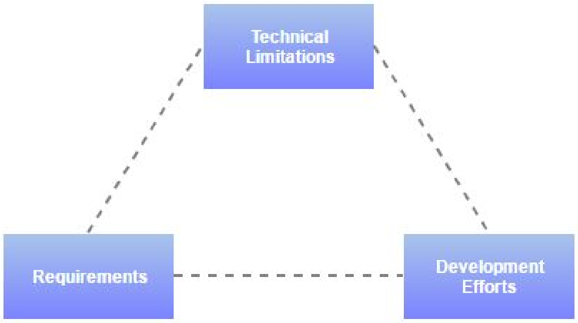
3. Development efforts:

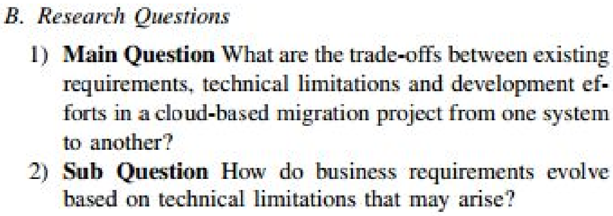
- How have the development resources been affected by new or modified requirements?
- What challenges has the development team faced when migrating to AWS?
- How have you minimized the development effort, avoided stalls or obstacles during the migration process?

4. Considering existing requirements, technical limitations and development efforts:

- Have compromises been between development resources and requirements?
  - FQ: Have requirements faced the need to be revised in order to accommodate existing development resources or vice versa?

Requirements Engineering Interview
AWS General Architects & Developers
2017-05-12





B. Research Questions

1) **Main Question** What are the trade-offs between existing requirements, technical limitations and development efforts in a cloud-based migration project from one system to another?
2) **Sub Question** How do business requirements evolve based on technical limitations that may arise?

1. Architectural perspective:

- What are the main differences between the previous Azure architecture and the new AWS one?
- What are the reasons for a different architecture?

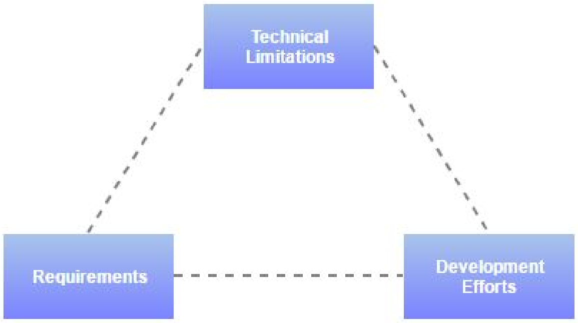
2. Requirements and technical limits:

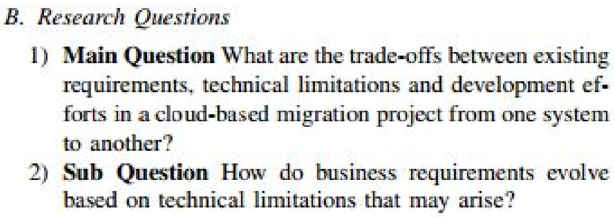
- What are the key requirements of the new AWS system?
- Have the requirements changed over time or remained static?
    - FQ (if changed): How have they changed or what lead to a change?
- Have there been any technical impediments in the migration?
    - FQ: Have previously established technical limitations forced you to develop components according to a certain standard?
    - FQ: What requirements can't be implemented due to current or upcoming technical limitations?
- How do you ensure technical limitations do not arise or are mitigated?

3. Development efforts:

- How have the development resources been affected by new or modified requirements?
    - FQ (if there has been an effect): Have requirements faced the need to be revised in order to accommodate existing development resources or vice versa?
- How have you streamlined the development process?
- What have been the greatest challenges during the migration?
    - FQ: Learning the new platform?
    - FQ: Testing or deploying components?
    - FQ: Reverse engineering the previous one?
    - FQ: Designing the new architecture?

Requirements Engineering Interview
Azure Solutions Architect
2017-05-17





B. Research Questions

1) **Main Question** What are the trade-offs between existing requirements, technical limitations and development efforts in a cloud-based migration project from one system to another?
2) **Sub Question** How do business requirements evolve based on technical limitations that may arise?

1. **Architectural perspective:**
    - Why was Microsoft Azure originally chosen as the cloud solution?
2. **Requirements and technical limits:**
    - What was the scope of Azure during the previous solution?
    - What were the requirements for the Azure solution in the form of KPI report generation?
    - The delivery manager said that the Azure service was outsourced. Were you part of that team and how did it change the general development process?
    - During the Azure period:
        - Did requirements change or remain static?
        - FQ (if changed): How have they changed or what lead to a change?
    - Have there been any technical impediments with Azure?