Securing business agility in Application Portfolio Migrations

a Case in the Dutch Insurance Industry

Nils de Bruin and Maarten van der Vlugt
VU University Amsterdam, 2011

Abstract

Organizations periodically update their desktop operating system (OS) for various critical reasons. To ensure business continuity, any software application incompatibility threat should be anticipated. Therefore all applications need to be tested and re-deployed to the new desktop OS. In large organizations application portfolio migration projects could span for far more than one year. However the reality is that organizations use many different COTS applications, which adhere to different standards and require different techniques to function properly within an organization. This makes predicting the required time and budget even in the best cases a mere informed guess.

The uncertainty on the time needed to migrate a software application introduces many difficulties in project planning and it clouds the possibilities for project optimization. Moreover it could make managers uncertain, hence delay or even cancel critical updates, resulting in a loss of business opportunities.

In our study we analyzed application portfolio migrations, we identified and ranked more than a dozen key factors that contribute to the complexity of application portfolio migrations and investigated their impact at a large Dutch insurance company. Based on these insights we devised an instrument to configure application portfolio migration projects for efficiency. Our method provides a generative basis that can be easily extended with new strategies for factors that were uncovered so far.

Preface

This study is a cooperation between Nils de Bruin and Maarten van der Vlugt to shed some academic light on IT infrastructure projects. We have a professional interest for continuously improving these kinds of projects. We believe our exploratory research can serve as a stepping stone for broader academic investigation of the IT infrastructure domain.

We would like to thank Marco Hagen, Sophie Tidman and Ment van der Plas for sharing their expert domain knowledge with us. Special thanks also to the application portfolio migration team in the case study project, for taking the effort of registering their engineering times during the project. And the case study organizations, for letting us use this data for academic purposes.

For all academic support we would like to thank Dr. Pieter de Leenheer - our mentor from the VU University - for leading and motivating us throughout this study. And also Prof. Dr. Jaap Gordijn for his role as second reader. Furthermore our gratitude goes to Drs. Martin Keesenberg for always being ready to give us academic insights and motivational support.

For all assistance on the Statistics front we would like to thank Jorg de Bruin, Cees van Rees and David van der Boon, who were able to tame our data into SPSS and help us process the results.

Last but not least we would like to name our organization: BVI ict solutions, hyperactive in application portfolio migration projects, for bringing us the opportunity to perform this research in the first place.

Index

1. Introduction

IT infrastructure should support business strategic agility [1], but how agile is an IT infrastructure? Factors that hinder IT agility can be a very important disabler in achieving more business agility [2]. In this study we try to explore the factors of agility in a specific kind of IT infrastructure change project: an application portfolio migration project.

1.1 Problem statement

Operating System upgrade

An IT infrastructure is the system of hardware, software, facilities and service components that support the delivery of business systems and IT-enabled processes. These systems are constantly subject to change. A specific IT infrastructure change that organizations face periodically is a desktop operating system (OS) upgrade. Operating system vendors support a certain OS version for a limited time before it goes “end-of-life” and will be unsupported. As organizations need to be compliant to rules and regulations (e.g. Sarbanes-Oxley or HIPAA) as well as keeping up to security standards, unsupported operating system versions should be avoided.

Desktop OS migration projects

In this thesis we take a closer look at Microsoft Windows desktop OS migration projects. With 96% of corporate PC's running Microsoft Windows [3] and advisory companies suggesting migration to new Windows versions in time [4], there is a substantial interest in Windows desktop OS migration projects.

According to Gartner, organizations should plan to spend between 12 and 18 months on an organization-wide Microsoft Windows desktop OS migration project. [5] But with so many organizations needing such long running projects, shortages of IT migration personnel occur periodically. For example: in the present day Gartner predicts there will be a shortage of IT migration personnel to actually perform the migrations in 2011 and 2012. This will lead to higher service rates, increasing the migration costs. But it also might set back migration plannings for some organizations, increasing the risk of being non-compliant for running unsupported operating systems and thus increasing security and regulatory risks.

So apart from business agility motivations there is a stressing need to reduce the amount of effort required for these migration projects. Gartner claims “complex environments will require even more time”, but what contributes to this complexity? How can we assess this complexity beforehand, solve it before it becomes a hurdle, or maybe even prevent potentially complexing factors?

Application migration

Applications are defined as a special class of Information Systems that support business processes directly. [6] Chief Information Officers regard applications as main ‘'products'' delivered to the business. Therefore one of the most important and major actions in a desktop OS migration project is the migration of the software application portfolio to the new OS platform. Reducing the effort needed to migrate the application portfolio thus leads to a reduction in the total effort needed for the desktop OS migration project.

To have an opportunity to reduce the effort needed to migrate the software application portfolio, we need to have insight in the factors that might have an effect on the time needed to migrate a software application.

Stakeholders

The results of our research are targeted for the internal IT and business decision makers, but the results of optimizing the application portfolio migration projects can have a positive effect on various different stakeholders:

- *End-users of the software applications*

  An end-user of a software application is the person that actually uses the software application in an IT-enabled process or business system. His or her main concern is the ability to continue work without interruption. Thus the correct functioning of the required software applications is a necessity.

- *Project initiator*

  The initiator of the desktop OS migration project. Typically this is the Chief Information Officer of the organization the end-users work for. The initiator is concerned with the business continuity and budgeting in short and long term on one hand, and the budget, deadline, quality and gains of the project on the other hand.

- *Project manager of the application portfolio migration*

  The project manager is concerned with accomplishing the stated project objectives and with the satisfaction of the project initiator on the constraints: budget, deadline and quality.

- *Application migration engineers*

  The application migration engineers are concerned with the technical migration of the software applications. They are managed by the project manager.

- *Application owners*

  The application owners are responsible for a single application or a group of applications within an organization. The application owners provide information about the applications to the engineers of the migration project. Furthermore they test the applications for production suitability after the technical migration.

1.2 State of the art

In a 1945 report by the American government the basic functional specifications of computers were set out. Two types of innovation were defined. The first is the improvement of hardware components, bringing faster computing speeds, larger capacities, better reliability and improved price-performance ratios. This innovation was driven predominantly by manufacturing companies. The second type of innovation is in the mode of operation, and according to Campbell-Kelly and Aspray the agent of change for this type of innovation was most often the academic sector. [7]

So lets see where science has brought us. The field of application portfolio migration projects seems to be in an academic niche between a number of research fields. In the following section we describe the academic foundations surrounding application portfolio migration projects.

The final purpose of computing systems is the completion of certain (business) tasks. But to get to application portfolio migration projects, we found several layers of abstraction:

Level 0: Complete a task
Level 1: Develop a tool to complete a task
Level 2: Manage a tool
Level 3: Manage a set of tools

1.2.1 Level 1: The science of developing a tool to complete a task

Since prehistoric times, mankind has sought for tools to make tasks easier. Anthropologists believe that the use of tools was an important step in the evolution of mankind [8]. Modern archeology found that the first computing tools were invented as early as 35.000 BC [9]. The “Lebombo bones” were found in the Lebombo Mountains between South Africa and Swaziland and are deemed to be one of the oldest mathematical artifacts.

Mechanical computers

Computer Science studies have come a long way since then. After the invention by the Italian mathematician Blaise Pascal in 1914, a Differential analyzer was built ten years later by Harold Locke Hazen and Vannevar Bush at the Massachusetts Institute of Technology [10]. It was used as a tool to solve previously time-consuming calculations. Unfortunately this device had to be programmed manually for every calculation; “with a wrench in one hand, and a gear in the other” [11]. This made the use of the machine very labour-intensive and too slow to meet the demands of the military [7].

Electronic computers

When Presper Echert and John Mauchly proposed a computer ten times as fast as the Differential analyzer and at least 100 times as fast as a human computer with a desk calculator the “Electronic Numerical Integrator And Computer” - or ENIAC - was built. [11] However it still needed to be programmed manually for each calculation: instead of shafts and gears, wires and set switches were manipulated. As Burks states: “an electromechanical method of programming would have been superior”.

Stored program computers

Based on new electromechanical discoveries and logical designs [12] [13], the Stored Program Computer was designed, and built in the form of the EDVAC. [11]. Now programs could be stored in memory and the concept of “Software” was born, a milestone in Computer Science [14] [11].

Programming

Unfortunately the state of art back then was that “Programming is one of the most difficult branches of applied mathematics; the poorer mathematicians had better remain pure mathematicians.” (prof. dr. E.W. Dijkstra). So high-level programming languages like FORTRAN [15], LISP [16], COBOL and BASIC were developed to be easier in operation than the low-level programming language “Assembly”. Soon the limitations of these programming languages were met [17] [18], and more structured programming languages were developed, like Pascal [19] and C [20]. Evolving further, programming languages started getting new possibilities for data structures [21], and starting the 1990s the main academic innovations in the mode of operation were on the principle of programming productivity. Object Oriented programming languages with features like Garbage collection emerged, like Java [22], Python, PHP and Ruby.

Software architecture

New programming dilemma's emerged, which could not all be solved in a solely technical manner. The research field of Software architecture was born. The Component-Oriented Software Development [23] theory solved software integration problems in a methodological and cultural way. And the Model-View-Controller paradigm [24] poses a way of thinking about programming that can help solve software design complexity issues.

Managing software development

Still, with all the academic innovations in the field of programming and software architecture, software application development projects still had different degrees of success with respect to finishing on-time and within costs [25]. The research field of software application development management developed various methods of managing software application development projects. The most prominent are the iterative methods, like the Waterfall model [25], and incremental methods like Agile programming [26].

Requirement Engineering

But the primary measure of success of a software system is the degree to which it meets the purpose for which it was intended [27]. The academic field of Requirements Engineering investigates and innovates on the ways to shorten the distance between the purpose and the development of the tool. And the newest methods of requirements engineering even recognize the value of systems as the purpose for building them [28].

1.2.2 Level 2: The science of managing a tool

Science has brought us a long way since the Lebombo bones. The academic fields developed tools to get the job done for us, made the development of these software tools faster and improved the success ratio of software application development. So now there are more and more software applications in the field, how can we “get a grip on” them?

Software complexity

The field of software complexity research provide metrics for assessing the complexity of software artifacts. Software metrics are focused on the internal structure of the software, such as functions, classes, programs or design documents [29]. Also the complexity of the application-domain knowledge can be regarded as an artifact of complexity [30]. This internal software complexity seems to have a direct relation with the costs of Software maintenance [31].

Software maintenance

The field of Software maintenance describes the problems and solutions to keep software matching to evolving requirements. This research focuses on several broad areas [32] :

- System dynamics, model the software as it changes over time, in order better to understand the underlying mechanisms.
- Maintenance processes; defining, measuring, improving, risk analysis, quality assurance.
- Studies of software change; impact analysis, change propagation.
- Products, linking software attributes to maintainability (from architecture to identifier naming conventions); higher levels of software abstraction.
- Program comprehension methods and tools, to link attributes to better cognitive understanding.

- High level management, business and people issues; business models such as outsourcing and applications management.
- Legacy and reverse engineering, to recover a software asset that has become very hard (expensive) to maintain.
- Validation, ensuring that the software changes work as required, and the unchanged parts have not become less dependable.

The subject of all these area's is the maintenance of one software application.

Application deployment

To better understand the complexity involved in applications and the deployment of these applications, we first investigate several concepts. This is based on previous work [33], which laid the groundwork for a framework of deployment concepts.

Concepts

Application
An application is a group of software components which together perform a certain action [34]. Furthermore, applications are defined as a special class of Information System that support business processes directly [6]. The application is visible to a user in the form of an entry point , usually a link, to the application.

Software deployment
According to [34] software deployment falls between the acquisition and the actual use of the application within a certain environment. This is further strengthened by the notion of [33], which states that this is the process of activities that takes place between the end of development and "the actual installation and maintenance of the application on consumer computers"

Release
To bridge the gap between the end of the development of an application and the deployment of the application, the term release is used. During the release activity, an application includes packaging which consists of combining software components and configuration settings into a deployable application [33].

Deployer site
The deployer site is the location from where an application will be deployed. In most cases this is a software vendor or, when an application is developed in-house, the department that develops the application.

Consumer site
Consumer sites are the locations which receive a certain application. This is, for example, a single PC [33], but this could also be an enterprise [35].

Application management systems
To be able to deploy a certain release of an application, application management systems are used. These systems are used to deliver applications to consumers sites, usually within corporate environments [33]. These systems can control the deployment of applications to specific targets.

Enterprise software deployment

As application complexity has increased due to a number of reasons [35] management of application deployment within an organization has also become more complex. Furthermore the fact that software vendors do not have any control in the environment their application will execute in [33], creates challenges in the enterprise environment. To be able to control and maintain applications within the organization, a model to overcome this challenge has been developed [35].

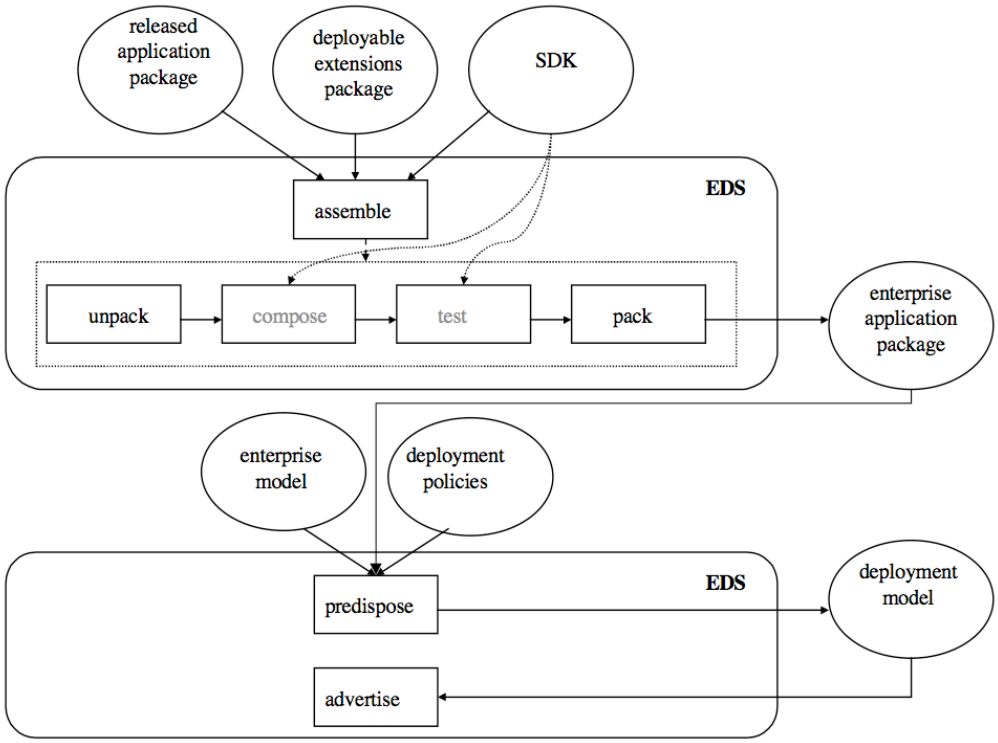
When an application or a new version of an application is released to the enterprise consumer, the application will first go through an *assemble* process [35]. During the assembly process, the application This assemble phase consists of the following steps:

- Unpack - The released application package is unpacked so its configuration can be modified
- Compose - The application configuration is modified, including corporate standards, for use within the enterprise environment,
- Test - The application is tested according to the requirements of the enterprise environment
- Pack - The result of these actions is packed into a new package which can be deployed

If we put the effort required in perspective [35] states: “Then starts the compose activity. This is really the core of the assemble activity. It takes a significant amount of time and human resources and cannot be fully automated. It consists in putting together components coming from different sources. This is mainly a “manual" programming activity”

After the assembly process is complete, the application is ready for deployment. During the deployment preparation, which is called the predispose activity, the deployment strategy is determined. This includes the time, location and the manner in which the application will be deployed. Finally the deployment is offered to the end-users in the form of an advertisement.

These activities are depicted in the following figure [35]:



Although this model is sufficiënt for a conceptual view of an application deployment within an enterprise, it doest not answer questions regarding the impact and effort required for a mass migration of applications.

The model does give us insight in the steps needed to deploy an application into an enterprise, so we can use this model as a basis to further explain the complexities involved in a mass migration of applications. With our study we focus on the effort required during the assemble process as this is the most intensive task in terms of manual human intervention [35]. As it is human-executed tasks that define the project costs, this is the most interesting part to investigate.

1.2.3 Level 3: The science of managing a set of tools

On this level, the "set of tools" we are talking about is known as the "Information Services". Managing the Information Services has posed new challenges for science to solve. The solutions come operationally and strategically, usually in the form of methods, systems, standard or norms [36].

According to Derksen et al., the development of these Information Services quality models can be triggered by several factors:

- Enforced by law (e.g. Sarbanes-Oxley, IFRS or HIPAA)
- Projects structurization (e.g. PRINCE2, PMBoK or IPMA Competence Baseline)
- Improving revenue (e.g. Six Sigma or Balanced Scorecard)
- New organizational structures (e.g. eSCM for Service Providers)
- Customer demands (e.g. capability models like eSCM, Capability Maturity Model or the Growth Phases model)

The Sarbanes-Oxley (SOx), IFRS and HIPAA models are designed to regulate specific business processes to make sure organizations are compliant to certain rules and regulations. PRINCE2, PMBoK and IPMA Competence Baseline provide a structure and a common glossary for projects. The aim is to provide the fundamentals of project management as they apply to a wide range of projects [37] [38].

The quality models that are aimed to improving revenue or living up to customer demands seem to have some consensus on the need to analyze the factors that have an impact on the process or product, and to learn from them to reach a higher quality level in terms of time, budget and product.

Six Sigma has a focus on the need for "intentional or explicit learning" to make improvements [39]. And the Balanced Scorecard model describes a focus on "problem solving and general organizational improvements" [40]. It also describes the need to measure the impacting factors before an organization is able to address them [41].

The Capability Maturity Model [42] is one of the models driven by Customer demands. The development of CMM was triggered by a need to have Software development "delivered on time and within budget". It describes five levels of Maturity of an organization that ultimately result in an improved control over time and budget. Level 5 (the highest level: "Optimizing") can only be reached when organizations analyze factors that have a negative impact on the software development process or product. With that, organizations can "establish a basis for learning, for taking action in a timely and well-informed way, and for improvement over time." [43]

The eSourcing Capability Model for Service Providers (eSCM) describes the methods an organization can develop "a repeated set of management processes that will ensure high-quality results, each and every time" [44]. These maturity

definitions and standards emphasize the need to analyze failures and learn from them to significantly improve the service process [45].

So the lesson what we can learn from peer research in the field of managing a set of tools is: to get to a higher level of maturity and efficiency, we need to learn from practice and identify factors that have a negative impact.

Application portfolio migrations

Unfortunately, the migration of a set of software tools looks like an academic greenfield. We think this might be because this is a problem that can only be investigated from the perspective of a migration project in an organization making use of all these software tools. The metaphor: “investigating a running engine” might be very appropriate for this.

However, many organizations face the application portfolio migration challenge. In 2010: “Forrester's analysis of more than 85,000 enterprise clients found that Windows XP, while still king, is finally beginning its long-anticipated decline in the corporate PC market.”. And in their lessons learned of 40 early adopter customers of Windows 7 they warn: “Don't underestimate the application compatibility challenge.”. [3]

Apparently the application portfolio migration is an important issue: “a fairly consistent lesson we heard from early adopter customers is to ensure that you have enough human resources tackling these [application migration] tasks initially, particularly for multinational organizations that support thousands of applications worldwide — many of which were likely developed in-house.”

Gartner states the situation: “The tepid corporate response to Windows Vista indicates that, for most organizations, Windows 7 will be the first major OS migration project in more than six or seven years.”. They propose three phases to prepare for the Operating System Migration: [4] [5]

1. Information gathering (estimated lead time: 3 months)
2. Engineering & Testing (estimated lead time: 6 - 9 months)
3. Pilot testing (estimated lead time: 6 months)

Gartner stresses the focus on the application portfolio in each of these phases. The goal is clear: “All business- critical applications must be determined to work on Windows 7, while all user-critical applications must operate or have reasonable workarounds.”.

Based on this we think study in this field is relevant and necessary and we are ready to accept the challenge of leading the way to a broadening of the research field.

1.3 Research questions

In this study we investigate application portfolio migrations. We ask the following research questions:

1. What is an application portfolio migration?
2. Who are the main actors in these projects?
3. What do experts recognize as possible factors that might effect the time needed for an application migration?
4. How does this compare to a real case study project?
5. What are the opportunities for optimization?

1.4 Expected outcomes

We expect that there are certain factors to be found that have an impact on the time needed to migrate an application to a new OS. We expect these factors can be measured as technical aspects or surrounding factors of the specific applications, and we are able to validate the factors in a case study. We also expect to find opportunities to mitigate the negatively impacting factors.

The results of our study will describe:

- A description and analysis of an application portfolio migration project.
- A set of factors, believed to have an impact on application migration time.
- Validation results on a subset of these factors, gained by data from an actual project.
- An instrument to optimize application portfolio migration projects.

2. Research strategy and methods

2.1 Strategy in detail

Strategy for research question 1. What is an application portfolio migration?

Based on the context found in the “State of the art” chapter an application portfolio migration project is described and analyzed. This case study description is a qualitative research method and enlightens the field and the way work is performed within these projects.

Strategy for research question 2. Who are the main actors in these projects?

In the case study project, we take the opportunity to find the main actors. These are described and visualized in relation to the project.

Strategy for research question 3. What do experts recognize as possible factors that might effect the time needed for an application migration?

To answer this research question, we use interviews to harvest knowledge from experts. To elevate the quality of the knowledge, we use expert consensus methods to qualify the importance of these factors.

Strategy for research question 4. How does this compare to a real case study project?

We try to confirm the findings of the previous research questions in a case study project. As described in [46], Qualitative methods can be used for exploratory purposes, and then quantitative methods for confirmatory purposes. Thus measurements are done in the case study project to get quantitative data which can be generalized and validated using statistical methods.

Strategy for research question 5. What are the opportunities for optimization?

Based on the results found we search for the scientific foundations and look for methods to mitigate the results of the impacting factors.

2.2 Strategy in practice

To optimize the external validity of our study we minimize to possibilities of interference by separating the case study research from the expert research. We divide the study into two phases. In the first phase of our study we use qualitative research methods to gather information from experts about the possible factors which might have an impact on the migration time of a software application. This answers research question 3, and serves as a qualitative basis for the quantitative case study research in phase 2.

During the second phase we perform a case study. We use this case study in a qualitative manner to answer research question 1 and 2. We then use quantitative research strategies to test if the factors found in phase 1 really do have an impact in the practice of the case study, as a form of methodological triangulation. This answers research question 4.

Finally we put our findings back in a scientific context and pose strategies for mitigation to answer research question 5.

2.3 Phase 1: Expert knowledge

2.3.1 Phase 1a: Expert interviews

We start with eliciting information from technical experts who are specialized in the migration of applications in application portfolio migration projects. In the interviews we ask the interviewees to identify possible factors which might have an impact on the time needed to migrate a software application. This results in a list of possible factors which is used in the next research phases.

We chose to perform one-on-one semi-structured interviews for this phase. One-on-one interviews, instead of physical group techniques, give us the insurance our experts are not biased by consented factors. This is important because that bias might increase the risk of missing out on important factors.

A structured interview technique would not suffice due to the fact that there is little scientific background in this field of research. On the other side, an open interview would possibly give us too little grip on the subject to form conclusions. Therefore we use a semi structured interview method and provide a framework in which interviewees can freely transfer their knowledge and which also gives us the possibility to standardize results.

Interviewees

We included experts in the field of application portfolio migration projects on the following inclusion criteria:

- More than five years of experience in the field of application portfolio migration projects, in more than five different projects.
- Multi-disciplinary expertise in these projects: technical as well as organizational.

We found three experts to be qualified and available for our research. Two experts are from The Netherlands and one expert is from the United Kingdom:

Marco Hagen (Application migration engineer, Hagen IT)
Abbreviated as “MH” in the results. Marco is an independent ICT professional specialized in application portfolio migration projects. As an application migration engineer he is responsible for the technical migration of applications. He has worked in many application portfolio migration projects over the years. His experience includes application portfolio migrations on many different platforms in many different organizations.

Sophie Tidman (Technical consultant, ChangeBASE Limited)
Abbreviated as “ST” in the results. Sophie has worked at InstallShield - a software company internationally known for it's application migration tools - as an application migration engineer, responsible for the technical migration of applications. Currently she works at ChangeBASE, a UK based company “specialized in automated application compatibility testing and remediation software”. At ChangeBASE she focuses on all the technical application challenges during desktop OS migration projects. In her new role as a technical application migration consultant she assists organizations all over the world with application portfolio migration projects.

Ment van der Plas (Technical consultant, Log In Consultants)
Abbreviated as “MP” in the results. Ment worked as an application migration engineer for a large Dutch governmental organization. Afterwards he became a technical application migration consultant and trainer at Log In Consultants, an “international IT service provider specialized in migration, desktop deployment and application delivery”. He has been rewarded with notable awards, like the Microsoft Most Valuable Professional Award (MVP) on Microsoft Application Virtualization (Softgrid), hence his expert reputation in the field of application portfolio migrations.

Interview strategy

The interviews are led by the researchers. All interviews are conducted in a similar structured manner:

1. Introduction.

   In the introduction we describe the goals and usage of our research, and the role of the interviewee in this context.

2. Alignment of definitions.

   Here we define the main definitions, to make sure we all talk about the same concepts.

3. Knowledge elicitation.

   Iterative and circular knowledge elicitation until the satisfaction point has been reached. All identified factors are analyzed and labeled ad-hoc by two interviewers independently.

4. Validation

   A validation of all elicited factors by method of teach-back, to confirm understanding.

Expected results

The validated lists of factors are combined, and this is represented in a matrix of factors on one axis, and the experts that identified the factors on the other axis. A first sign of expert consensus is the overlap of factors between experts.

We expect our experts to already show some overlap in the factors they address. The representation of the results in a matrix makes this overlap visible. To make it possible to get an intuitive interpretation of the results, the matrix is ordered so that the most-overlapping factors are at the top, and the least-overlapping factors at the bottom. The ordering strategy is also a preparation for the next phase.

2.3.2 Phase 1b: Getting expert consensus

After performing and processing the semi structured interviews, we try to reach expert consensus. By reaching expert consensus we get the opportunity to focus on the most important set of possible impacting factors during the Case study in the next phase.

We want to take every opinion into account but we are not able to get the experts in the same physical location. Therefore we perform a combination of the nominal group technique and the Delphi method to reach expert consensus.

Nominal group technique

The nominal group technique is a decision making method for use among groups of many sizes, who want to make their decision quickly, as by a vote, but want every opinion taken into account. This is as opposed to traditional voting, where only the largest group is considered [47].

The expert will start with writing down their opinions, and then as a group all opinions are ranked according to their relative importance according to the expert. The method of group-ranking is that each expert will give their own ranking. These results are scored so that the number 1 option gets the highest score, and the bottom option the lowest score, and the scores for each opinion are added to each other. The opinion that has the highest score is treated as the most important consented factor.

As it is geographically and plan-wise difficult to join all the experts together at the same time, we are unable to use the nominal group technique in our research. We do, however, use the group-ranking method in our research.

Delphi method

The Delphi method is a systematic, interactive forecasting method which relies on a panel of experts. The experts answer questionnaires in two or more rounds. After each round, a facilitator provides an anonymous summary of the experts' forecasts from the previous round as well as the reasons they provided for their judgments. Thus, experts are encouraged to revise their earlier answers in light of the replies of other members of their panel. It is believed that during this process the range of the answers will decrease and the group will converge towards the "correct" answer. Finally, the process is stopped after a pre-defined stop criterion (e.g. number of rounds, achievement of consensus, stability of results) and the mean or median scores of the final rounds determine the results [48]. The Delphi method is usually used when experts are geographically separated as is the case in our research.

As we already perform one-on-one interviews with our experts, we have a head-start with the questionnaire rounds. We start with the results of the previous phase and perform the inter-expert influence from there.

Consensus strategy

First round
During the first round of the Delphi technique we send each interviewee the list of his or her identified factors. The interviewee is asked to validate the formalized factors to be sure no misunderstanding corrupts our results.

In the lists, similar factors from the other interviews are combined with matching overlapping factors. Those overlapping factors are anonymously summarized to common identified factors. This summarizing of factors will result in expert interaction when the experts validate these common factors as their own.

Second round
During the second round of the Delphi technique we send the complete combined list of factors to the interviewees for them to rank all factors by presumed order of impact. As an extra method of interaction between the experts, we now also list the number of experts that identified that factor. The ordering of the factors is by amount of overlap - most overlapping factor on top - for this will make the results of the overlap more visible, and thus the interaction between the experts more intuitive.

Expected results

The ranking is summed up for every factor and the factor with the highest amount of points is regarded as the most consented - and thus deemed the most important - factor.

We expect our experts to have a clear preference on some of the factors, so that a strong top segment of factors will be seen. We expect other factors to be clearly ranked at the bottom, being clearly regarded as low-impact factors.

The result of this phase is a more reliable ordering of the results from the previous phase. Based on this ordering we can select the most important factors for further investigation in the next phase.

2.4 Phase 2: Case study

We perform a case study at one of the top 3 large Dutch insurance companies where we use the top 15 most impacting deemed factors of the first phase of our research study as input.

The organization is an internationally operating insurance, pension and investment products company with the number of employees ranging in the 5-digit scale, and the total number of customers in the 8-digit scale. The corporate headquarters is based in the Netherlands.

2.4.1 Project set-up

The organization in this case study migrates from a Microsoft Windows 2000 desktop OS to a Microsoft Windows Vista desktop OS. As the expertise and manpower is not present within the organization itself, the project is executed by a large Dutch system integrator, which specializes in transition and management of IT infrastructures.

The project is divided into two project teams; a migration team and an application migration team. The migration team is responsible for the end-to-end migration of all computers within the organization and the application migration team is responsible for the end-to-end migration of the application portfolio based on the new platform of the insurance company.

The application migration team consists of engineers from different organizations which all specialize in application migrations.

2.4.2 Application portfolio

The application portfolio of the organization in the case study consists of around 360 applications which are diverse and consists of software applications licensed for use from third party software vendors (known as “Common Of The Shelf” software applications), and applications that have been developed in-house specifically for the organization.

2.4.3 Application portfolio migration project

The application portfolio migration project consists of many sub-projects, consisting of the migration of a single application.

The goal of the application portfolio migration project is to migrate all software applications that are in-use by an organization to the new target platform without losing application functionality but also without adding new functionality. Furthermore, this has to be performed without disrupting business continuity.

To be able to understand the application migration case study, we provide a brief description of the migration process of a single application. This is based on the project documentation from the case study project.

Main actors

- Application owners

  The application owners are responsible for a single application or a group of applications within an organization. The application owners provide information about the applications to the engineers of the migration project. Furthermore they test the applications for production suitability after the technical migration.

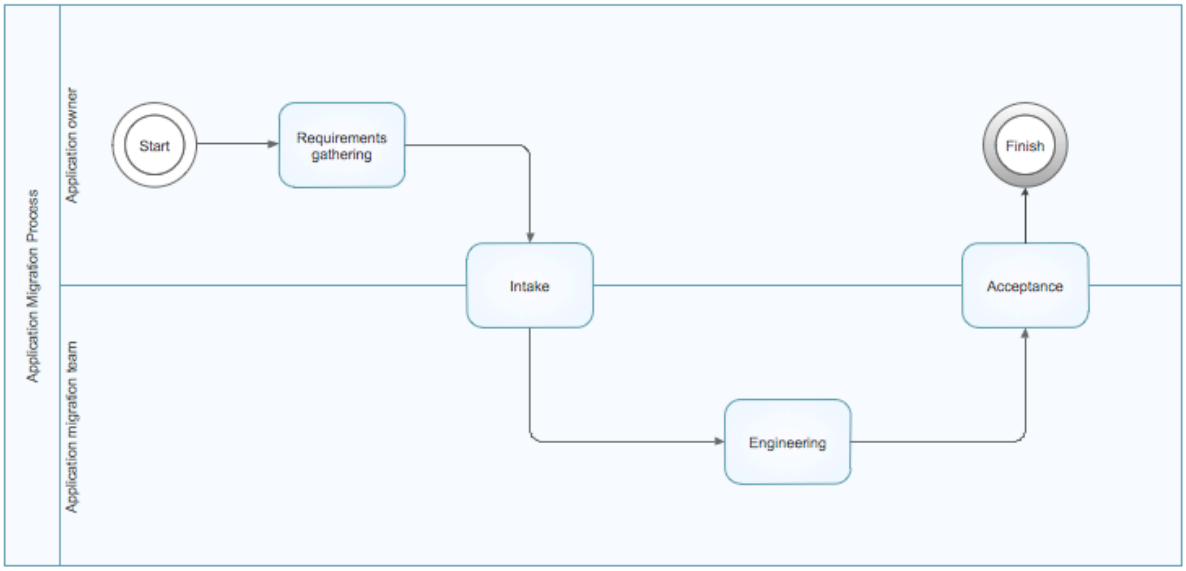
- Application migration engineers

  The application migration engineers are concerned with the technical migration of the software applications. Furthermore, the application migration engineers validate the output of an application owner and assist with the testing of the finished product.

In the application migration case study, the total amount of application owners is 63. The total number of application migration engineers fluctuates between 3 and 11 during the project. This is determined by the workload of the project.

Phases

The main actors and phases are described in the following picture:



1. Requirements gathering

During this phase, an application owner gathers all information needed about an application so the application can be migrated. This includes an installation manual and the source files of the application.

Task	*Responsibility*
Gathering information on application	Application owner

2. Intake

This is the official moment of information transfer between an application owner and the application migration team. The application source files and documentation are checked. The test plan is carried out to make sure all functionality required is also available on the target platform.

Task	*Responsibilty*
Transfer validated application information from application owner to application migration team	Application owner / Application migration team

3. Engineering

In the engineering phase an application migration engineer uses the information gathered in the intake phase to perform a technical migration of the application to the new platform.

Task	*Responsibility*
Technical migration of the application to the new platform	Application migration team

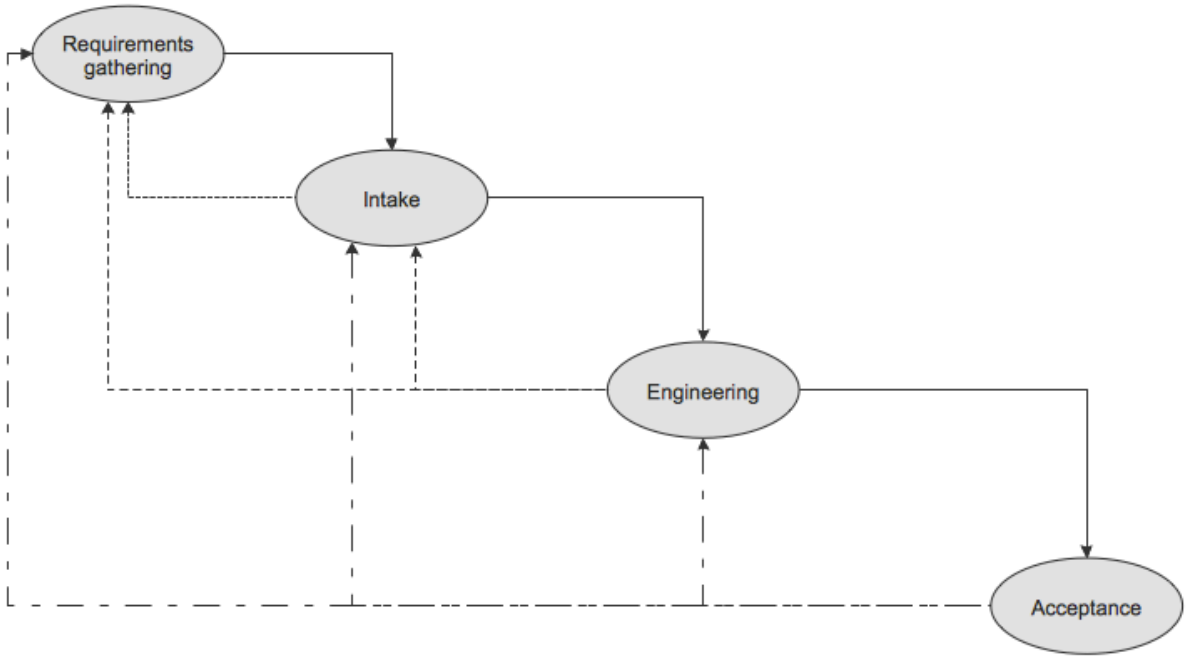
4. Acceptance

During this phase the application owner tests the application for complete functionality and completeness. When an application owner accepts the application, it is ready for use in the production environment.

Task	*Responsibility*
Validate application for correctness on the new platform	Application owner / Application migration team

Process flow

The application portfolio migration process used by the company in the case study is sequential and is divided into several consecutive phases. This process flow resembles a Waterfall model [25] - as used in software development projects. The following picture describes the flow of the process:



In the case when a fault or error is detected during one of the phases, the application returns to an earlier phase and then all subsequent phases are executed again. The earlier phase in the process the application returns to does not have to be the first phase; this depends on the fault that has been detected.

Waterfall like model

The waterfall model is one of the oldest software development methods. Its steps resemble a waterfall like movement towards a finished product, with each consecutive step adding value to the finished product.

One of the major flaws of the waterfall model, is the fact that it is not flexible when requirements change. When requirements change, the complete process needs to be repeated from the first steps onward [49]. As can be expected, this can lead to several ineffective development cycles which requires extra time and thus costs.

One of the characteristics of the application portfolio migration is the fact that application requirements rarely change; the migration of existing functionality to the new platform is the main purpose. Therefore, one of the biggest disadvantages of the Waterfall model can be minimized; when the requirements for a single application are complete, the application can be migrated.

The completeness of the requirements can be validated when an application owner tests the application before it will be migrated to the new platform. This step is performed after the requirements gathering phase during the intake phase.

To minimize the risk of faults in a later stage of the migration process, each process phase has a test suitable for the phase performed. Before information is processed to the next phase, a validation test is performed. This ensures quality especially in the final stage during acceptance testing and minimizes application errors during the application migration process.

Case study information sources

The information sources which are used by the researchers during the case study are divided in the following categories:

- Time registration tooling
- Application dossiers
- Project management tooling
- Deployment tooling

Time registration tooling

During the migration project, the application migration engineers have to track their time spent on the engineering of a particular application. A time registration tool is set up to facilitate this process for the engineers. All unique application numbers are pre-loaded into the tool to prevent registration errors as much as possible. Also, during the project, the times entered by engineers is monitored by the researchers, and deviations in quality (e.g. missing hours) are flagged to be corrected.

Application dossiers

Application documentation in application portfolio migration projects are divided in several sections. These sections provide different kinds of information for different stakeholders.
The first section of application documentation consists of meta information. Typical information of this section is:

- Application name
- Application owner
- Relations with other applications

The next section is the installation manual. This section consists of a step by step installation flow which results in a functional application.
The following section is the section containing Package notes. Package notes give insight in the engineering process of the application. An application migration engineer writes information which he/she deems important about the application.
The final section is the test plan. The test plan is a step by step plan which covers the complete functionality of an application. This is used to determine if an application has been migrated according to specification.

Project management tooling

To be able to track progress and issues, management tooling is used during the migration project. Major steps are recorded to be able to track progress of the project. The status of an application, which will be used to track errors after an acceptance test has been performed, will be used as an input for our data gathering.

Deployment tooling

To be able to deploy applications in a large managed corporate environment, deployment tools are used. This is also the case at the company of the case study. The deployment tool has a database of application dependencies with one another. We use the information of this database to further enrich the data of our case study results.

Case study strategy

1. Measure migration time needed for each application

The time needed for the technical migration of the applications - measured in hours - is recorded in the case study project, by the application migration engineers. At the end of the project an export of all time sheets from the time registration tooling is performed, and all time spent on an application are added together. This results in a list of total engineering time per application.

2. Assess factor measurability

We examine the possibilities for measurement of the possible impacting factors in the case study. Factors that can not be measured in the case study are filtered out.

3. Assess factors for each application

A strict quantification model is constructed for each remaining factor and these factors are quantified for a random subset of 99 migrated applications in the case study.

4. Statistical analysis

The information about the application migrations gathered in step 1 and 3 are combined in a statistical analysis to asses the possible interactions in the case study. We investigate the interactions between the potentially impacting factors and the measured engineering time per application. Factors that occurred in less than 5% of the cases are deemed to be insufficiently represented in the case study, and are left out of the analysis.

Step 1. Individual factor investigation
All factors are individually investigated by using several statistical tests. First we describe the difference between two groups, by using boxplots. After this, we perform the Mann-Whitney, Kaplan-Meier and Mantel-Cox tests.

Boxplots
Boxplots are used to visually describe distribution of values [50]. It also shows outliers which give abnormal results.

Mann-Whitney test
With the Mann-Whitney test it is possible to analyze two groups of cases and report if there is a statistical difference between the two groups [51]. As we can divide each factor in being true or false, this test can give statistical results about the impact of a factor.

Kaplan-Meier
The Kaplan-Meier test is used to visually show the difference between two groups, in the case a factor is true or false. It takes time into account from start of development, till the application has been migrated [52].

Mantel-Cox
By analyzing the areas under the curve of both lines in the Kaplan-Meier figure, we can determine if there is a significant difference between a factor being true or false. This difference can be calculated by using a logrank test (Mantel-Cox). As it takes time into account, this gives a more refined statistical results [53].

Overall
With these tests we want to find correlations between time (dependent variable) and the investigated factor (independent variable).

Step 2. Complete regression analysis
We perform a regression analysis by using the Cox Regression test. With this test we want to find the factors, when all factors are combined, with the highest impact and significance.

After this test a step-wise regression analysis is performed with a forward selection in order of predictive power, starting with the highest predictive power. This step-wise analysis gives us insight in the proportions of predictive power of the impacting factors.

Expected results

The top 15 factors are presented in an overview with their statistical significance in the case study, together with the statistical impact. This overview shows the top factors elicited from the experts in phase 1 in the light of the actual case study project.

We expect to see significant correlations between the factors our experts address, and the development times we measure in the case study. This correlation proves the impact of the factors on the case study project. As the expert consensus phase was conducted without a link to the case study project, we can assess the case study project as a random a-select sample, and this might provide some basis for generalization.

3. Data analysis and Results

3.1 Phase 1: Expert knowledge

3.1.1 Phase 1a: Expert interviews

In Addendum A we present the validated interview results per interviewee. Each factor is normalized in the way that when the factor is “true” or “large”, the migration of the software application will supposedly take more time.

These results are grouped by common factor and presented in the following matrix, in order of overlap. The right columns indicate the interviewees bringing up that factor and the code refers to their interview results in Addendum A.

		MH	*ST*	*MP*
F01	Application has a connection to specific hardware	MH23	ST2	MP15
F02	Application has not been tested on target platform	MH4	ST18	MP1
F03	Application has pre-requisites	MH2	ST15	MP5
F04	Installation time needed for installation	MH6	ST14	MP22
F05	Application contains legacy device driver installation	MH3	ST10	
F06	Application does not have an application owner available ad-hoc		ST17	MP18
F07	Application functions in combination with other applications	MH20		MP6
F08	Application has a machine or user specific license	MH10		MP13
F09	Application has organization-specific customization		ST6	MP10
F10	Application installation source is not supplied in the MSI format	MH1	ST5	
F11	Application makes use of a backend system which is unavailable from within the development environment	MH17	ST12	
F12	Application needs extra local permissions to function	MH21		MP9

F13	Application test manual is not available or incomplete	MH14		MP3
F14	Number of files in shared locations	MH8	ST7	
F15	Number of major steps in installation manual	MH7		MP8
F16	Size of installation source	MH5		MP21
F17	Size of test plan		ST13	MP24
F18	Application contains self registering com files		ST9	
F19	Application has been developed in-house		ST1	
F20	Application has files or registry keys already used by another application			MP19
F21	Application has organization-specific machine-based customization	MH11		
F22	Application has organization-specific user-based customization	MH12		
F23	Application includes a form of authorization			MP16
F24	Application includes a form of authorization but credentials are not supplied in installation manual	MH18		
F25	Application includes Session 0 Services.		ST11	
F26	Application installation causes DLL conflicts	MH9		
F27	Application installation describes non-generic customizations			MP11
F28	Application installation information is not available online, e.g. communities	MH16		
F29	Application installation makes changes to shared network data	MH22		
F30	Application installation manual contains steps that cannot be automated			MP14
F31	Application installation manual is incomplete	MH13		
F32	Application installation needs reboot	MH25		
F33	Application installation source is supplied in the MSI format			MP23

F34	Application installs a server component on the client	MH26		
F35	Application is an upgrade to a previously deployed version			MP7
F36	Application is externally configured	MH19		
F37	Application is not the newest available version of this application		ST16	
F38	Application makes use of a backend system		ST3	
F39	Application makes use of a backend system, which is different for DTA			MP17
F40	Application makes use of deprecated functions not available on new platform		ST4	
F41	Application needs changes to computer BIOS configuration	MH24		
F42	Application vendor does not supply a customization tool for the application installation			MP20
F43	Application vendor does not supply installation documentation	MH15		
F44	Application will be used by multiple users on one machine			MP4
F45	Application will be used on multiple platforms			MP12
F46	Different geographical location of application migration engineer and application owner			MP2
F47	Number of registry items in system locations		ST8	

3.1.2 Phase 1b: Expert consensus

In Addendum B we present the full list of anonymized elicited factors, as presented to the interviewees for ranking. In Addendum C we present the ranking-results from our experts.

Based on the nominal group technique, each factor gets a score based on its ranking so that the number 1 option gets the highest score, and the bottom option the lowest score. Then the scores for each factor are summed up and the factor that has the highest score is treated as the most important consented factor.

This research phase resulted in the following ranked list of possible impacting factors, in order of rank:

Overall Rank	*Factor*	*Description*	*Total Score*
1	F11	Application makes use of a backend system which is unavailable from within the development environment	130
2	F24	Application includes a form of authorization but credentials are not supplied in installation manual	126
3	F30	Application installation manual contains steps that cannot be automated	124
4	F40	Application makes use of deprecated functions not available on new platform	123
5	F06	Application does not have an application owner available ad-hoc	110
6	F07	Application functions in combination with other applications	109
7	F08	Application has a machine or user specific license	107
8	F01	Application has a connection to specific hardware	96
9	F15	Number of major steps in installation manual	95
10	F22	Application has organization-specific user-based customization	89
11	F13	Application test manual is not available or incomplete	87
12	F31	Application installation manual is incomplete	86
13	F39	Application makes use of a backend system, which is different for DTA	85
14	F41	Application needs changes to computer BIOS configuration	85
15	F05	Application contains legacy device driver installation	83
16	F38	Application makes use of a backend system	82
17	F36	Application is externally configured	79
18	F10	Application installation source is not supplied in the MSI format	78
19	F04	Installation time needed for installation	77
20	F03	Application has pre-requisites	76

21	F25	Application includes Session 0 Services.	74
22	F12	Application needs extra local permissions to function	73
23	F21	Application has organization-specific machine-based customization	73
24	F45	Application will be used on multiple platforms	72
25	F23	Application includes a form of authorization	70
26	F09	Application has organization-specific customization	69
27	F43	Application vendor does not supply installation documentation	68
28	F20	Application has files or registry keys already used by another application	68
29	F35	Application is an upgrade to a previously deployed version	65
30	F29	Application installation makes changes to shared network data	64
31	F16	Size of installation source	64
32	F02	Application has not been tested on target platform	64
33	F27	Application installation describes non-generic customizations	60
34	F26	Application installation causes DLL conflicts	60
35	F44	Application will be used by multiple users on one machine	58
36	F14	Number of files in shared locations	50
37	F32	Application installation needs reboot	46
38	F37	Application is not the newest available version of this application	45
39	F17	Size of test plan	42
40	F46	Different geographical location of application migration engineer and application owner	42
41	F34	Application installs a server component on the client	42
42	F19	Application has been developed in-house	38
43	F47	Number of registry items in system locations	34

44	F18	Application contains self registering com files	33
45	F28	Application installation information is not available online, e.g. communities	31
46	F33	Application installation source is supplied in the MSI format	29
47	F42	Application vendor does not supply a customization tool for the application installation	23

3.2 Phase 2: Case study

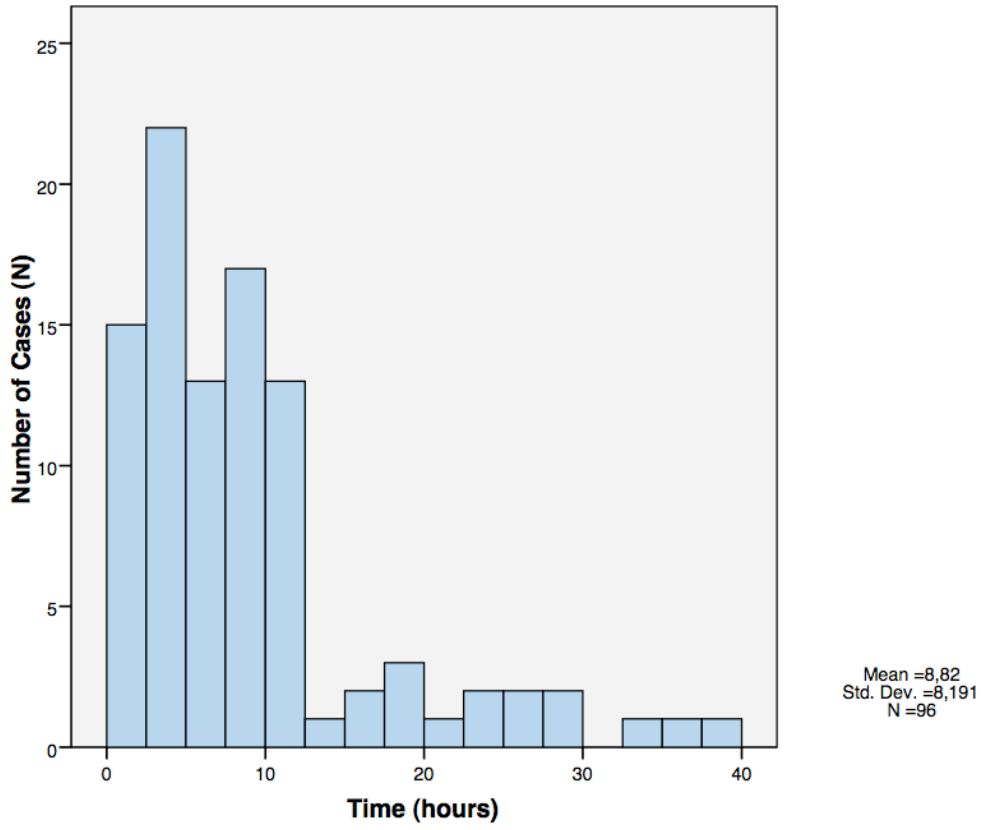
With the quantitative data from the case study we perform a statistical analysis to confirm the findings of phase 1. We describe our observations from several viewpoints. First we determine the overall characteristics of the development times in the case study. After this we analyze the statistical relationship between the development times and each factor individually. Finally we analyze all factors together and determine their effect on engineering time.

3.2.1 Case description

Of the total amount of 360 applications which were migrated, we examined 96 applications. These 96 applications were chosen random a-select and consisted off many different COTS applications. When an application did not require any technical migration, the application was eliminated from the examined group of applications as no engineering was needed (migration engineering time: zero hours).

For each application the total engineering time was calculated to be used for analysis. Because each engineering hour needs to be paid for, this total engineering time is directly proportional to the costs involved to technically migrate the application. With this engineering time per application we are able to make a histogram in which we can see how the engineering times are distributed. For example, it can show us if there are many applications with an engineering time in a certain time range or if engineering times are distributed evenly.

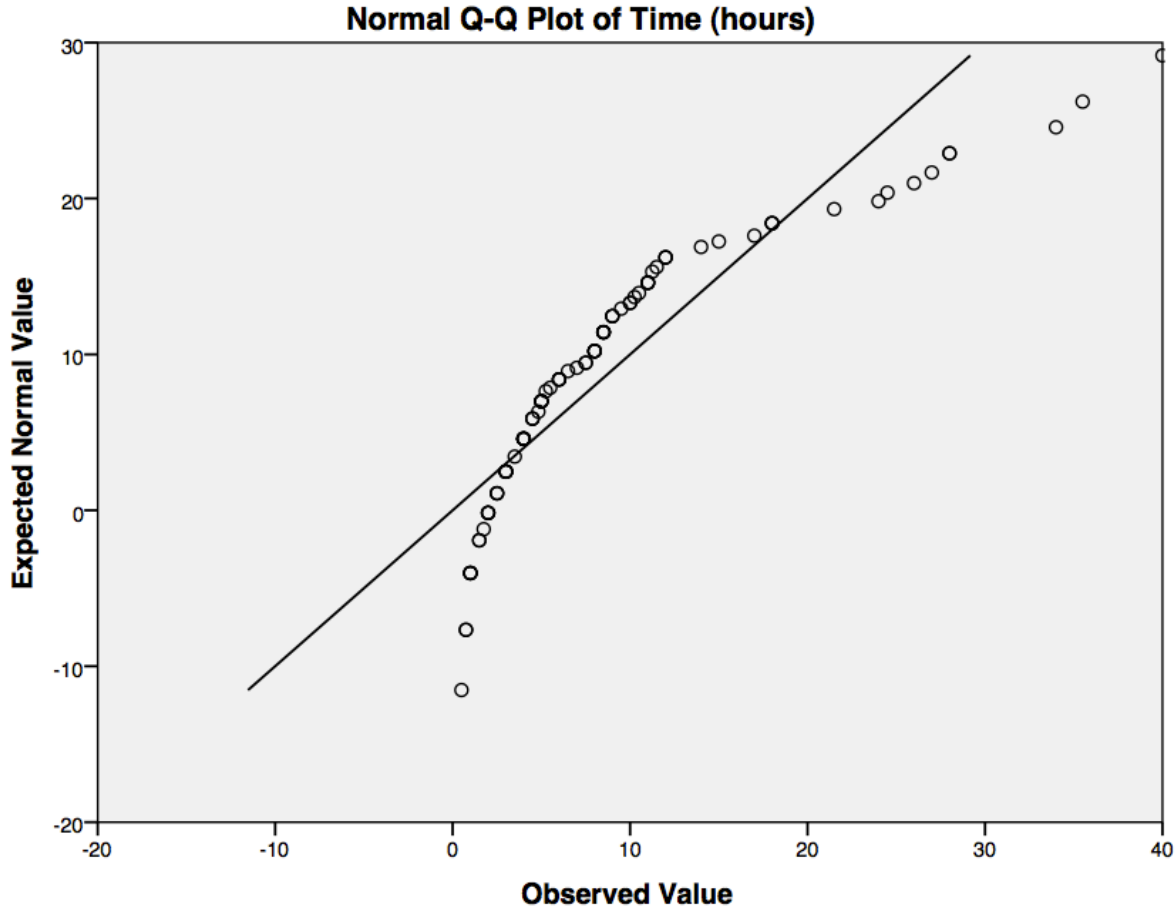
The histogram for the recorded Engineering times:



The figure shows us that the engineering times are not distributed normally and that the distribution is therefore non-gaussian. A normal distribution would have a little group of applications in the first range of time, followed by a larger group in the middle and again a little group of applications in the end, like a bell shape.

The histogram has a right skewness; this means that most applications are situated on the left of the histogram with a few applications in the 'tail' of the histogram. This can be clearly seen visually, with the largest group of applications having an individual engineering time under 12,5 hours. The mean of all the engineering times is 8,82 hours; this means that on average it takes 8,82 hours for an engineer to migrate an application.

To have more insight in the distribution of the cases, we created a Q-Q plot. Q-Q plots are used to determine if observed values adhere to the expected normal values.



As can be seen in the figure, the observed values (circles) are not aligned to the expected normal values. This further shows that the distribution of this population of cases is non-gaussian. With a non-gaussian distribution, we can use specific non-parametric statistical tests like a Mann-Whitney test, Kaplan-Meier and Mantel-Cox tests.

3.2.2 Factors

Factors prevented by project design

Several factors that the experts recognized did not occur within the case study project. Upon further inspection, these factors apparently were prevented by the project design. This prevention was prior to the start of the project and independent of our expert research study (phase 1 of this thesis). These prevented factors are described below:

Factor F11: Application makes use of a backend system which is unavailable from within the development environment (Ranked Nr. 1) and **Factor F39**: Application makes use of a backend system, which is different for DTA (Ranked Nr. 13)
In the case study, there was no special “development environment” to hinder the application migration. All applications were tested in the operational “production environment” on the production backend systems. The application portfolio migration project was considered a special kind of development project for all applications were already used in the production environment. The risks of performing the migration project in the production environment was thus deemed small.

Factor F40: Application makes use of deprecated functions not available on new platform (Ranked Nr. 4) and **Factor F05**: Application contains legacy device driver installation (Ranked Nr. 15)
In the case study the project was set-up so that not one incompatible application entered the technical migration project. Before an application got through the “intake” checks, the application was tested technically and functionally on the target platform. These functional checks are a part of the “Tortola Application Migration Method”. Any application that could not pass these tests was intercepted before it could create problems in the technical migration. The intercepted applications went into a remediation phase, where a compatible version was found to be used on the new platform. So while this factor did occur in the application portfolio migration project, steps were taken in the project design to ensure this factor was fixed as early in the project as possible. According to Paulk [54]: "developers should plan testing early and develop test cases in parallel with requirements analysis". According to Steve McConnell [55], errors found as early as possible in the process hugely benefit the total effort which is needed to fix the problem.

Factor F31: Application installation manual is incomplete (Ranked Nr. 12)
A positive side-effect of testing all applications technically and functionally before letting it enter the technical migration phase is that all application installation manuals need to be complete before the tests are possible. Thus preventing this factor from occurring during the technical migration phase, this aides in reduced effort to migrate applications. Again this is due to the effect Steve McConnell [55] describes in his book. This prevention method is also part of the “Tortola Application Migration Method”, thus this factor had no effect in the case study.

Not measured in case study

Factor F06*: Application does not have an application owner available ad-hoc (Ranked Nr. 5)* This factor was not measured in the case study for the technical

migration engineers have had contact with the application owners during engineering, but due to the distributed and ad-hoc nature of this communication, we could not get reliable enough data on this factor in hindsight.

Finally, three factors (F08: “Application has a machine or user specific license”, F30: “Application installation manual contains steps that cannot be automated” and F41: “Application needs changes to computer BIOS configuration”) were insufficiently represented in the case study (below 5% representation) and were left out of the final analysis.

Factors measured in case study

The remaining factors are assessed using the methods described in addendum D. These factors are (in order of expert-deemed importance):

- **Factor F24:** Application includes a form of authorization but credentials are not supplied in the installation manual
- **Factor F07:** Application functions in combination with other applications
- **Factor F01:** Application has a connection to specific hardware
- **Factor F15:** Number of major steps in installation manual
- **Factor F22:** Application has organization-specific user-based customization
- **Factor F13:** Application test manual is not available or incomplete

3.2.3 Statistical analysis

Univariate testing

Univariate tests asses each factor one at a time, and gives more insight in the characteristics of the specific factor. All factors have been examined on the cases. This has been performed with two approaches as this gives a better statistical result:

1. Boxplots and Mann-Whitney

   For each factor we created boxplots, to visualize how the values are distributed compared to each other. With these boxplots we are able to find the outliers (cases with irregular values). These outliers are analyzed in a later chapter to find out what has possibly caused the application to have an abnormal value.
   The factors were then statistically analyzed using the Mann-Whitney test. This test shows the statistical significance, which tells us if the factor hypothesis is true.

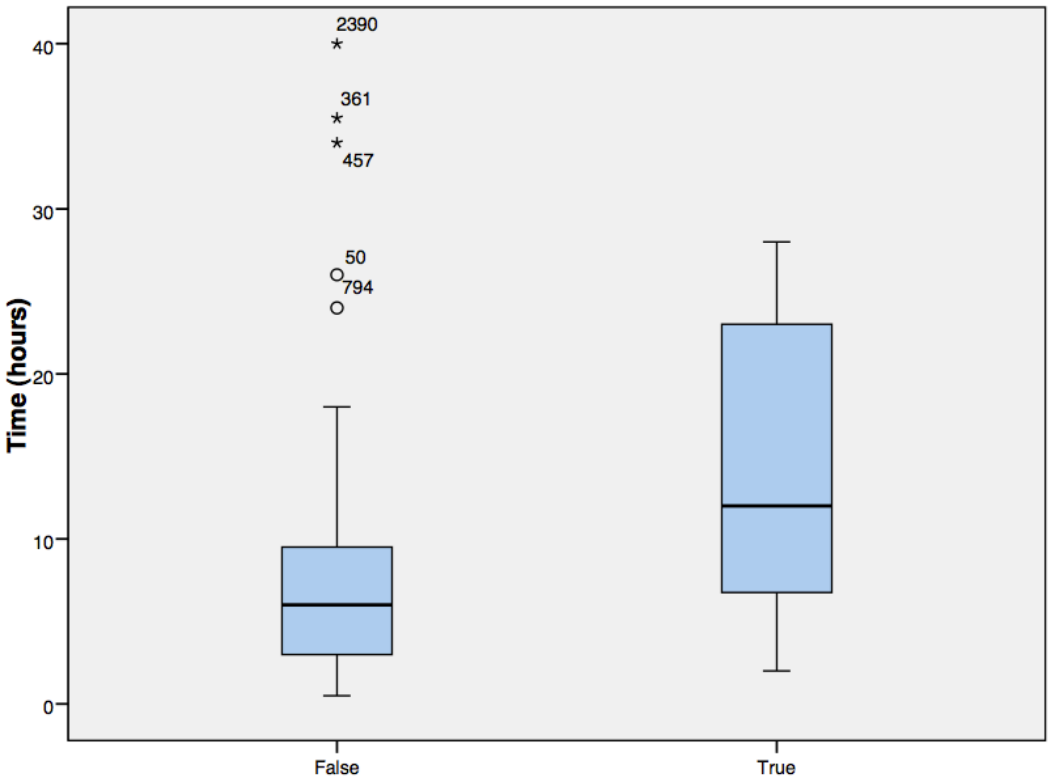
2. Kaplan-Meier and Mantel-Cox

   To further analyze the factors we used the statistical analysis of Kaplan-Meier and the Mantel-Cox tests. The results of the Kaplan-Meier test is a figure in which two lines representing the factor - being true or false - can be seen. It is then possible to compare the area under the curve of these lines and see if there is a statistical difference. This is analyzed using the Mantel-Cox test.

To be able to confirm the assumption a factor has an effect, we need a statistical significance of at most 0,05. This means we are at least 95% certain that the effect is not coincidental.

Factor F24: Application includes a form of authorization but credentials are not supplied in the installation manual

1. Boxplot and Mann-Whitney test



F24: Application includes a form of authorization but credentials are not supplied in installation manual

In this boxplot we see that there is a difference between the two groups of applications; one group consisting of applications where Factor F24 is true, the other being the group of applications where Factor F24 is false. The group of applications for which the factor value is false seems to have less engineering time then the group of applications for which the factor value is true.

Mann-Whitney Test

Ranks

	F24: Application includes	N	Mean Rank	Sum of Ranks
Time (hours)	False	81	45,04	3648,50
	True	15	67,17	1007,50
	Total	96		

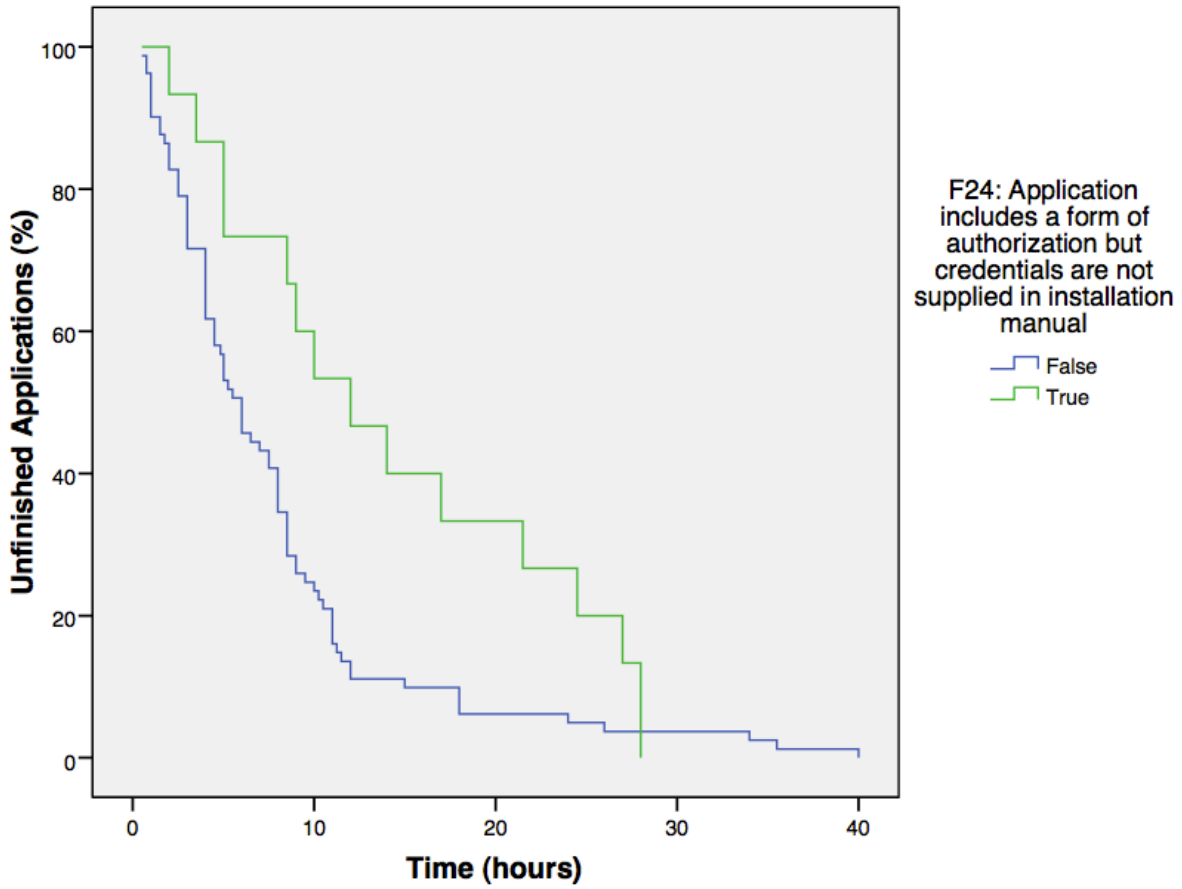
Test Statistics[a]

	Time (hours)
Mann-Whitney U	327,500
Wilcoxon W	3648,500
Z	-2,828
Asymp. Sig. (2-tailed)	,005

a. Grouping Variable: F24: Application includes a form of authorization but credentials are not supplied in installation manual

The Mann-Whitney test gives statistical significance to this claim. The tables above give some extra information on the number of application (N) for which F24 was true (15) or false (81), but the value we are looking for is the Asymp. Sig. (2-tailed) label in the Test Statistics table which gives us the significance value. In this case, this is 0,005 which means we can state the hypothesis for this factor is true, and we can confirm Factor F24 has had an effect on the development times in the case study.

2. Kaplan-Meier and Mantel-Cox test



Overall Comparisons

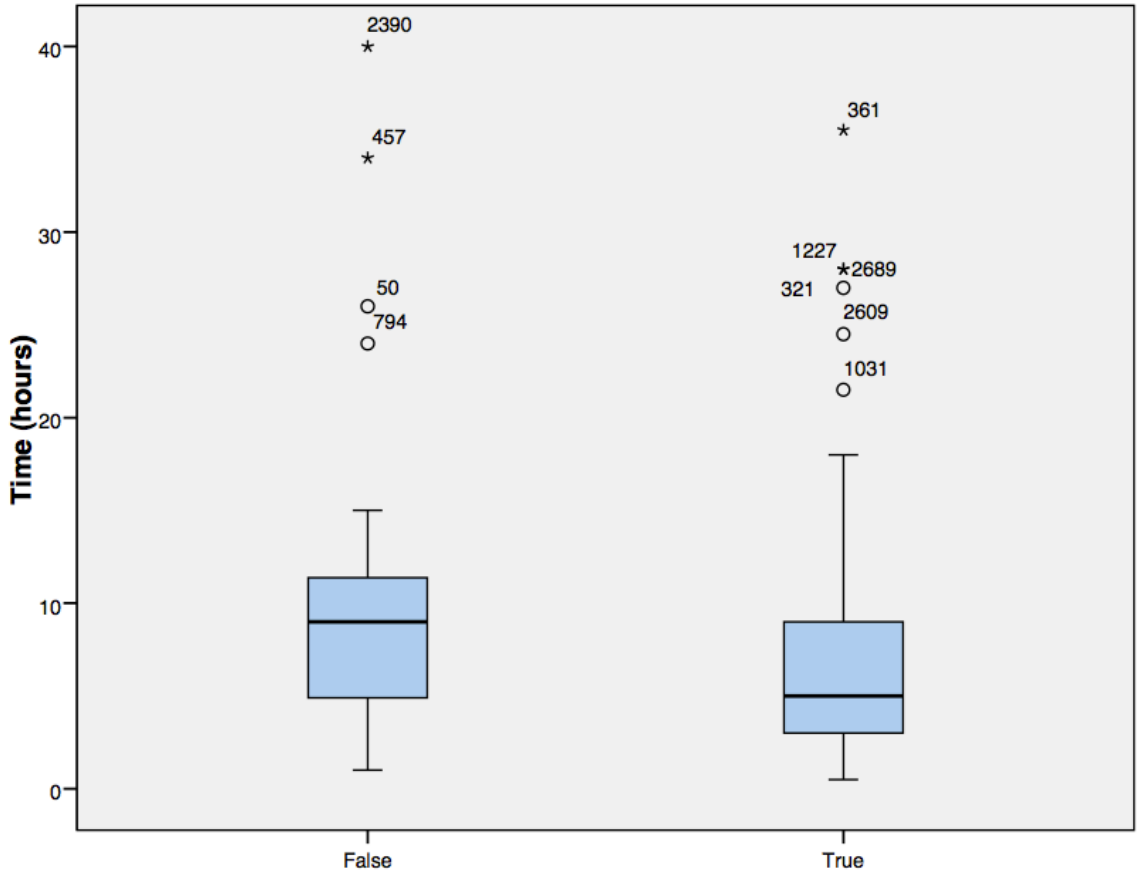
	Chi-Square	df	Sig.
Log Rank (Mantel-Cox)	5,281	1	,022

Test of equality of survival distributions for the different levels of F24: Application includes a form of authorization but credentials are not supplied in installation manual.

The Kaplan-Meier test displays two lines representing the groups of applications divided by the fact if the factor is true or false. In this figure there is a difference between the two lines. To be able to determine if it also has a statistical significance, we performed a Mantel-Cox test, which involves comparing the areas under both curves. We can see that the area under the curve of the true line is larger then under the curve of the false line. This means that applications, when the factor is assessed as true, take more time to complete then when the factor is assessed as false. This is confirmed by Mantel-Cox in the table Overall Comparisons in which the value (Log Rank) of this test is 0,022. This means that the hypothesis for this factor is true as well, and we can confirm that the Factor F24 has had an effect on the engineering times in the case study.

Factor F07: Applications function in combination of other applications

1. Boxplot and Mann-Whitney test



F07: Application functions in combination with other applications

Mann-Whitney Test

Ranks

	F07: Application functions	N	Mean Rank	Sum of Ranks
Time (hours)	False	31	58,42	1811,00
	True	65	43,77	2845,00
	Total	96		

Test Statistics[a]

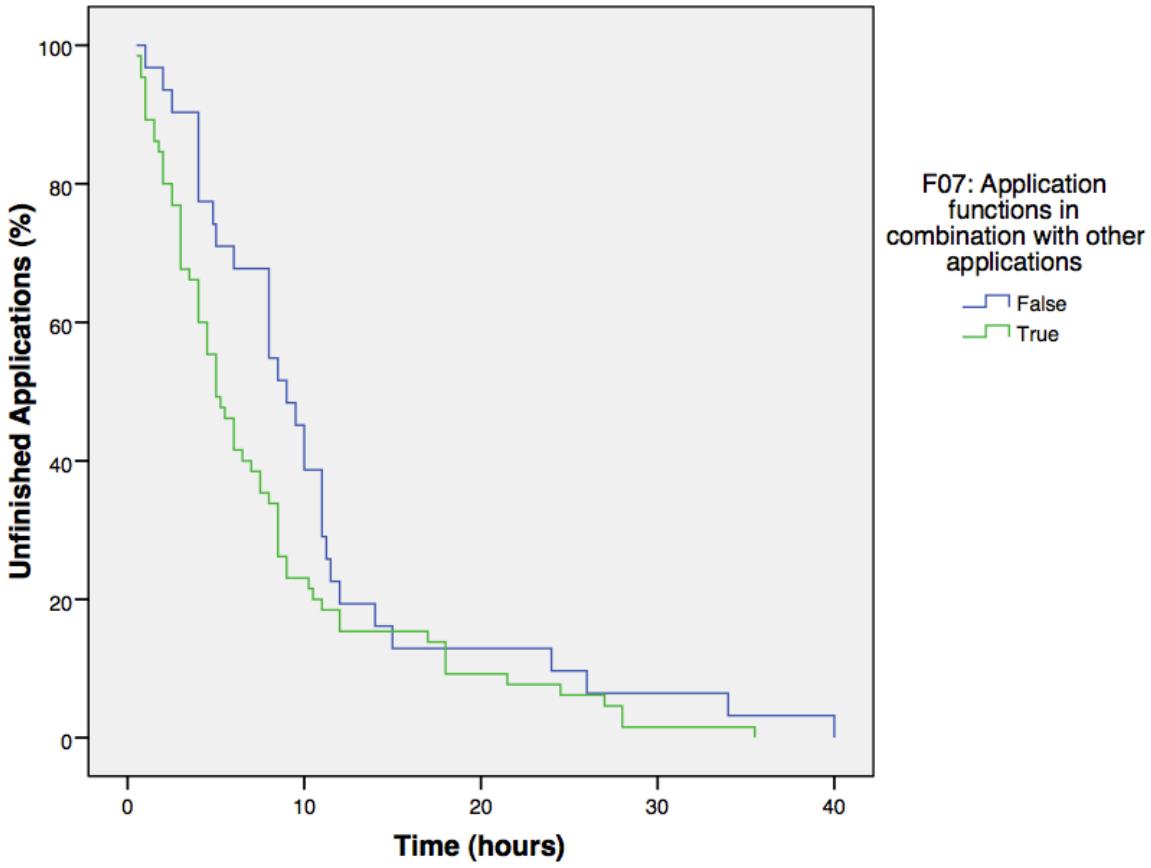
	Time (hours)
Mann-Whitney U	700,000
Wilcoxon W	2845,000
Z	-2,412
Asymp. Sig. (2-tailed)	,016

a. Grouping Variable: F07: Application functions in combination with other applications

With Factor F07 we see a situation the experts did not anticipate; the impact of an application functioning in combination with other applications is actually reversed. According to this result, this means that when applications function in combination

with each other, it actually takes less engineering time to migrate them instead of standalone applications.
If we now check the table Test Statistics from the Mann-Whitney test, we see that the significance of this factor (Asymp. Sig.) is 0,016 which shows us that this assumption is significant.

2. Kaplan-Meier and Mantel-Cox test



Overall Comparisons

	Chi-Square	df	Sig.
Log Rank (Mantel-Cox)	3,187	1	,074

Test of equality of survival distributions for the different levels of F07: Application functions in combination with other applications.

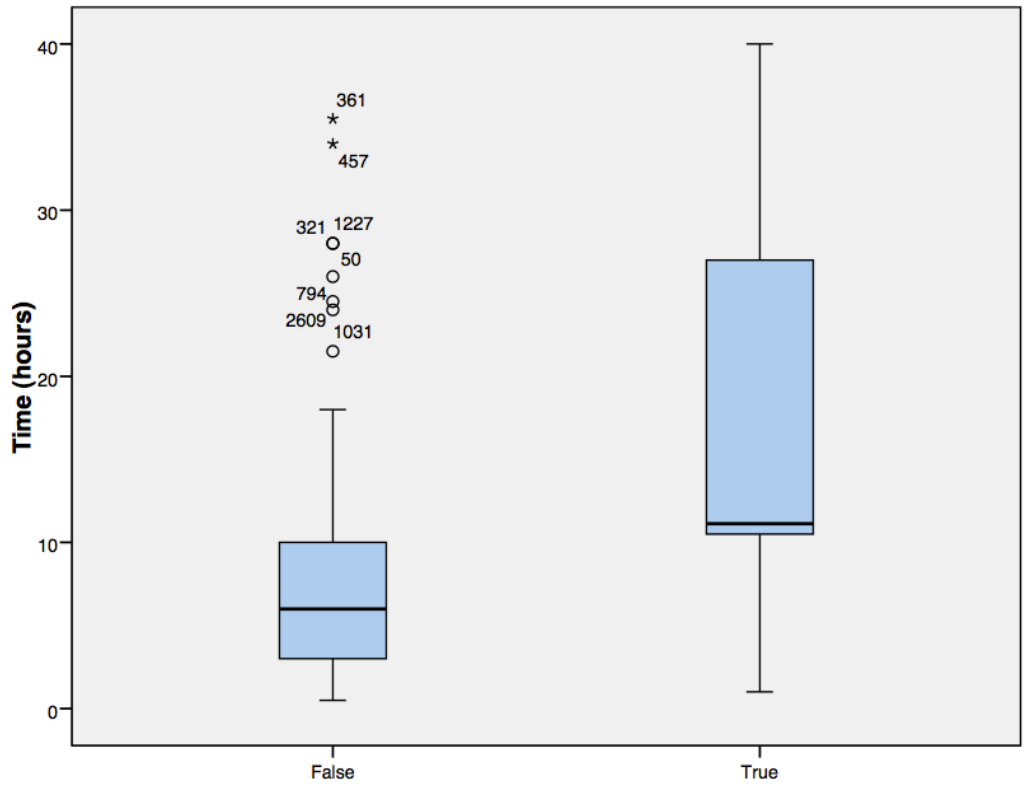
If we take a look at the Kaplan-Meier figure, we can see a difference as well, although it is not statistical significant anymore; according to the Mantel-Cox test, shown in the table Overall Comparisons, it is 0,074. We can see a trend, but more cases need to be examined to prove the assumption.

Possible reasons for this factor having the opposite effect then anticipated could be explained by the fact that the application portfolio consisted of many applications which required a client application to function. This is for example the case with applications which depend on the Oracle client software. In most cases, the dependent applications only make use of a shortcut and/or a small set of files,

therefore the engineering time of these applications is very low compared to standalone applications. As this is an assumption, this should be investigated in future research.

Factor F01: Application has a connection to specific hardware

1. Boxplot and Mann-Whitney test



Mann-Whitney Test

Ranks

	F01: Application has	N	Mean Rank	Sum of Ranks
Time (hours)	False	90	47,14	4242,50
	True	6	68,92	413,50
	Total	96		

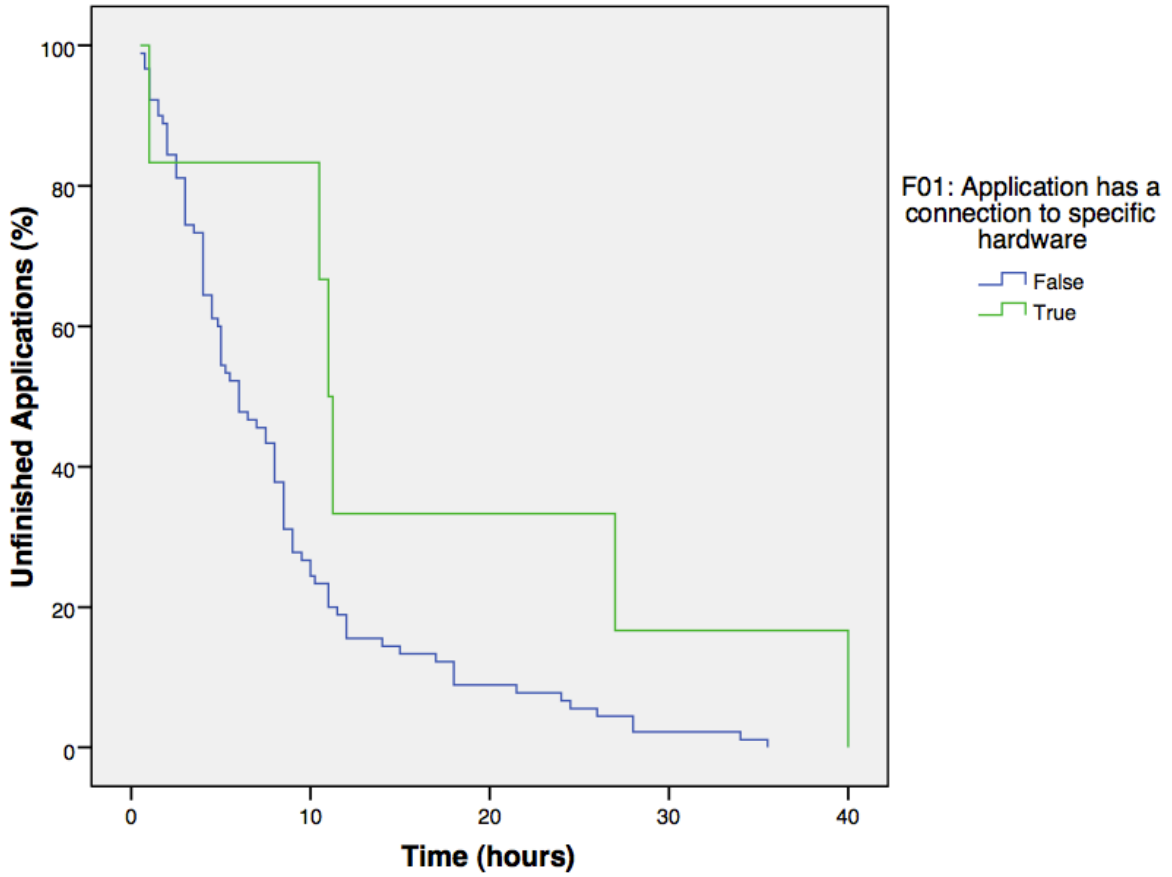
Test Statistics[a]

	Time (hours)
Mann-Whitney U	147,500
Wilcoxon W	4242,500
Z	-1,856
Asymp. Sig. (2-tailed)	,063

a. Grouping Variable: F01: Application has a connection to specific hardware

When we look at the boxplot of Factor F01, we see there is a large spread when applications are assessed as true. The large spread is caused by the fact that there are not many values assessed as true. This can be seen in the Rank table of the Mann-Whitney test in the column N; there are only 6 applications for which this factor was true.
This factor is almost significant with a value of 0,063 according to the table Test Statistics of the Mann-Whitney test. With this value we can see that there is a trend, but more samples are needed to prove the assumption.

2. Kaplan-Meier and Mantel-Cox test



Overall Comparisons

	Chi-Square	df	Sig.
Log Rank (Mantel-Cox)	4,278	1	,039

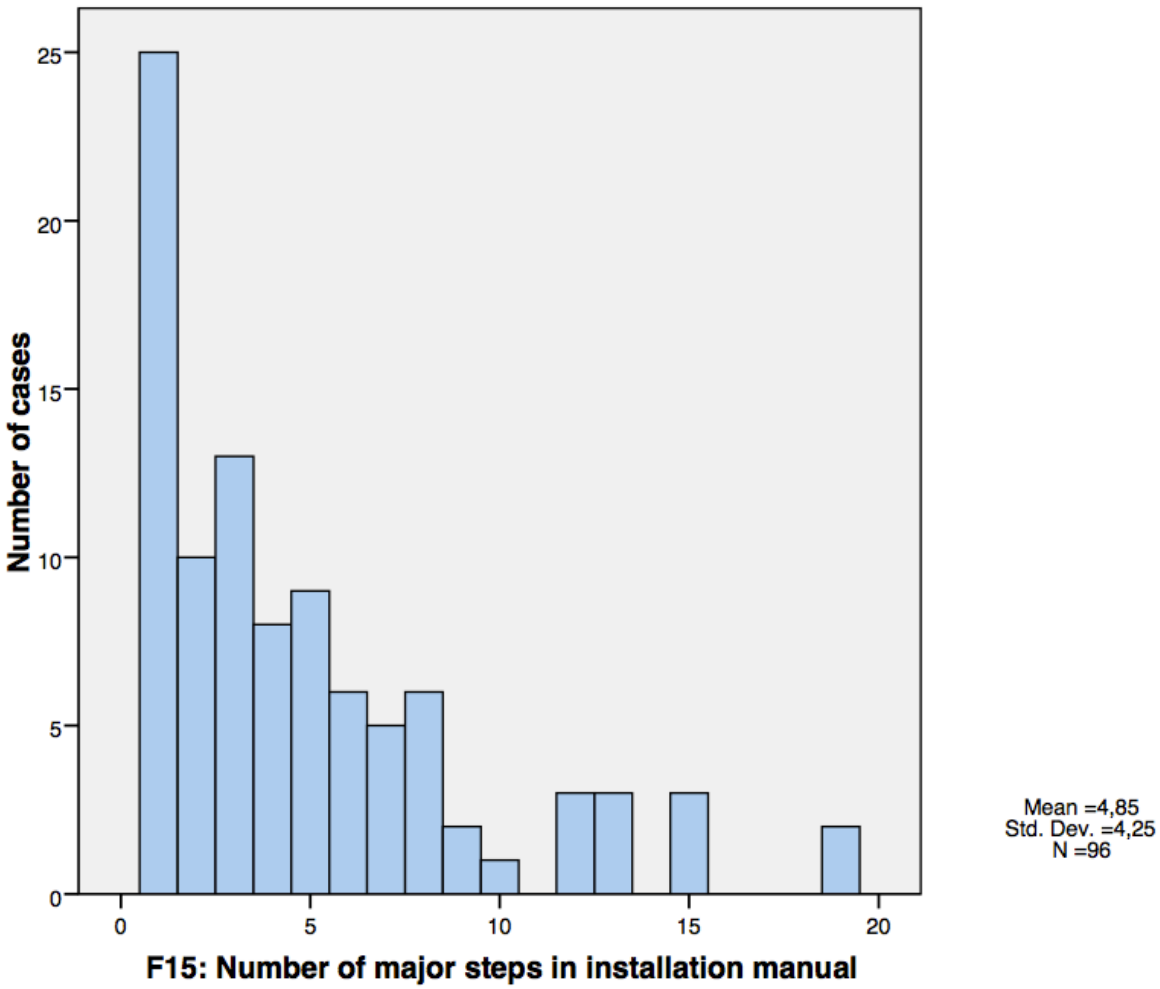
Test of equality of survival distributions for the different levels of F01: Application has a connection to specific hardware.

If we take a look at the Kaplan-Meier figure we can see that the area under the curve for true is larger then for false. This is also proved by the table Overall Comparisons of the Mantel-Cox test, which gives a significance of 0,039 for this factor. However, because the Mann-Whitney test is not significant more research needs to be performed to prove this hypothesis.

Factor F15: Number of steps in installation manual

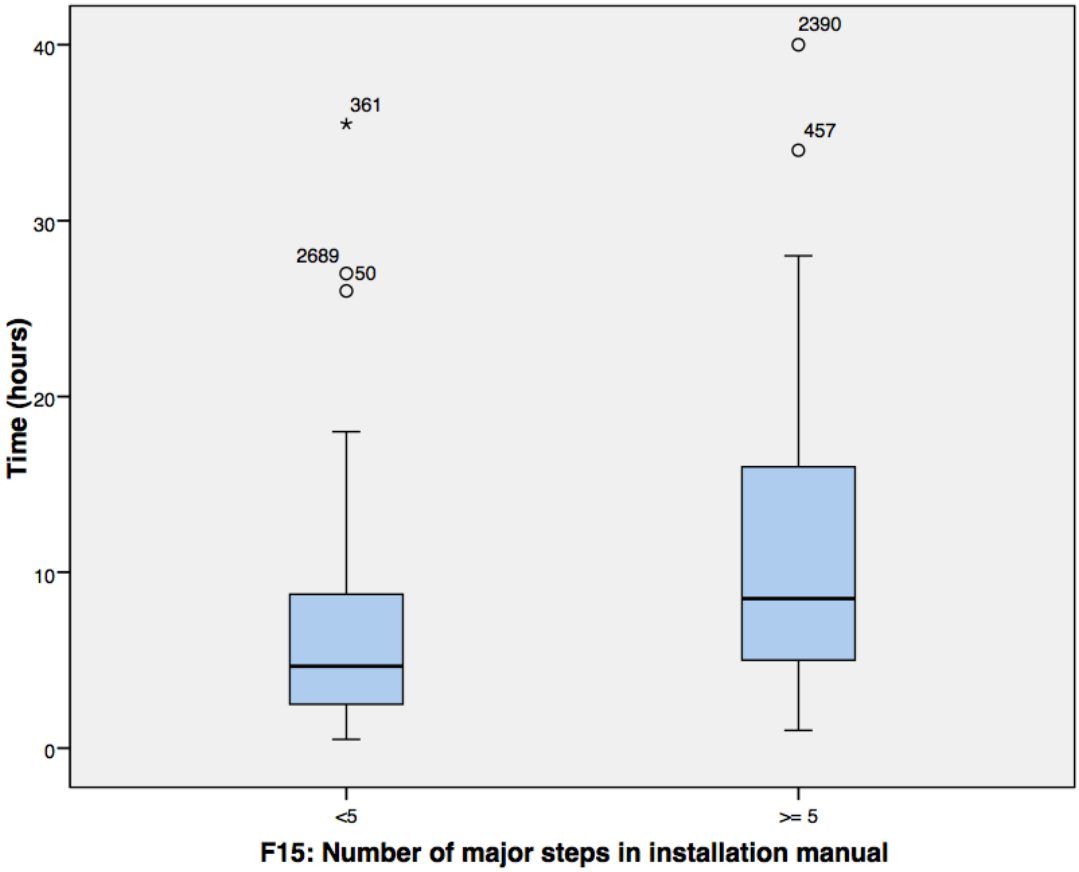
With Factor F15 we have a different case compared to the other factors. The factor does not describe a situation being true or false, but it describes the number of steps which is required to install a certain application. To be able to perform statistical analysis and compare this factor to the other factors, we examined this factor further.

1. Histogram



As we can see, the histogram has a right skewness, so the distribution is again non-Gaussian. The mean of the number of steps is 4,85, which means that on average an application takes almost 5 steps to install completely.
To be able to continue the statistical analysis, we created two groups and divided the applications in groups with less or more then 5 steps. This gave us the best results with a good distribution between the two groups.

2. Boxplot and Mann-Whitney test



Mann-Whitney Test

Ranks

	F15: Number of	N	Mean Rank	Sum of Ranks
Time (hours)	<5	56	40,42	2263,50
	>= 5	40	59,81	2392,50
	Total	96		

Test Statistics[a]

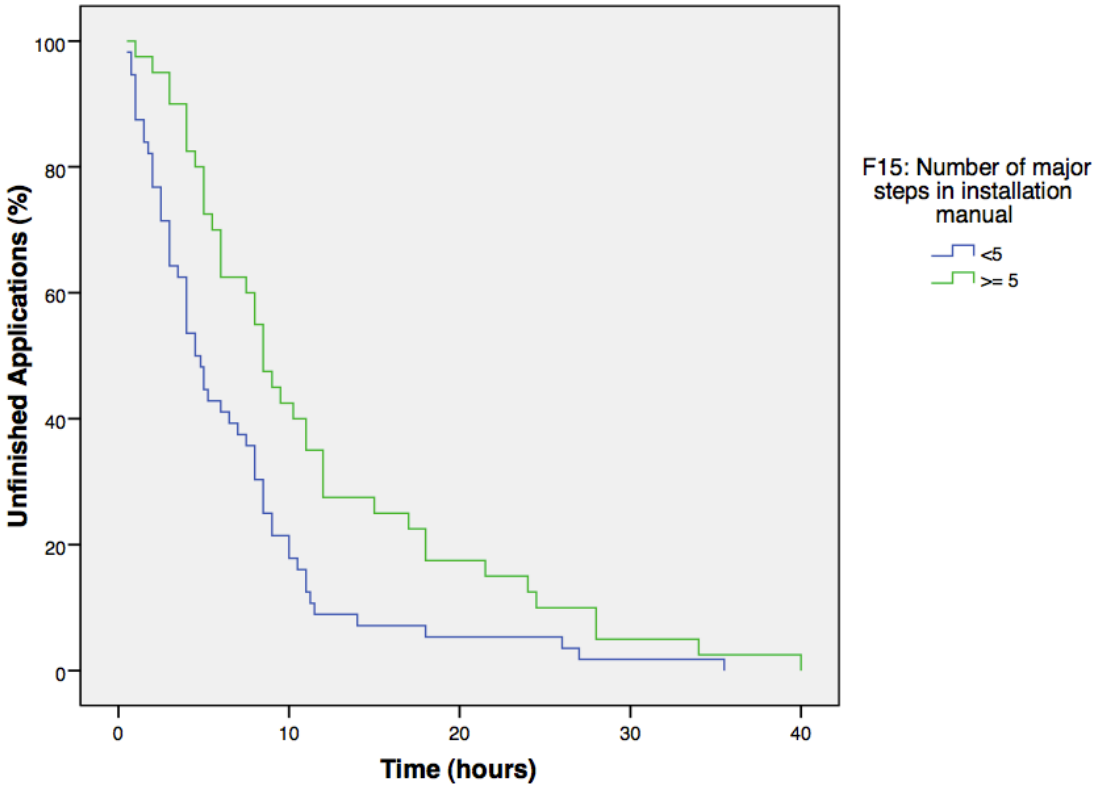
	Time (hours)
Mann-Whitney U	667,500
Wilcoxon W	2263,500
Z	-3,366
Asymp. Sig. (2-tailed)	,001

a. Grouping Variable: F15: Number of major steps in installation manual

As we can clearly see, the significance of this assumption is that the engineering time of an application increases with the number of steps. More precisely: When an

application has more then 5 installation steps it takes significantly more time to migrate the application.

3. Kaplan-Meier and Mantel-Cox test



Overall Comparisons

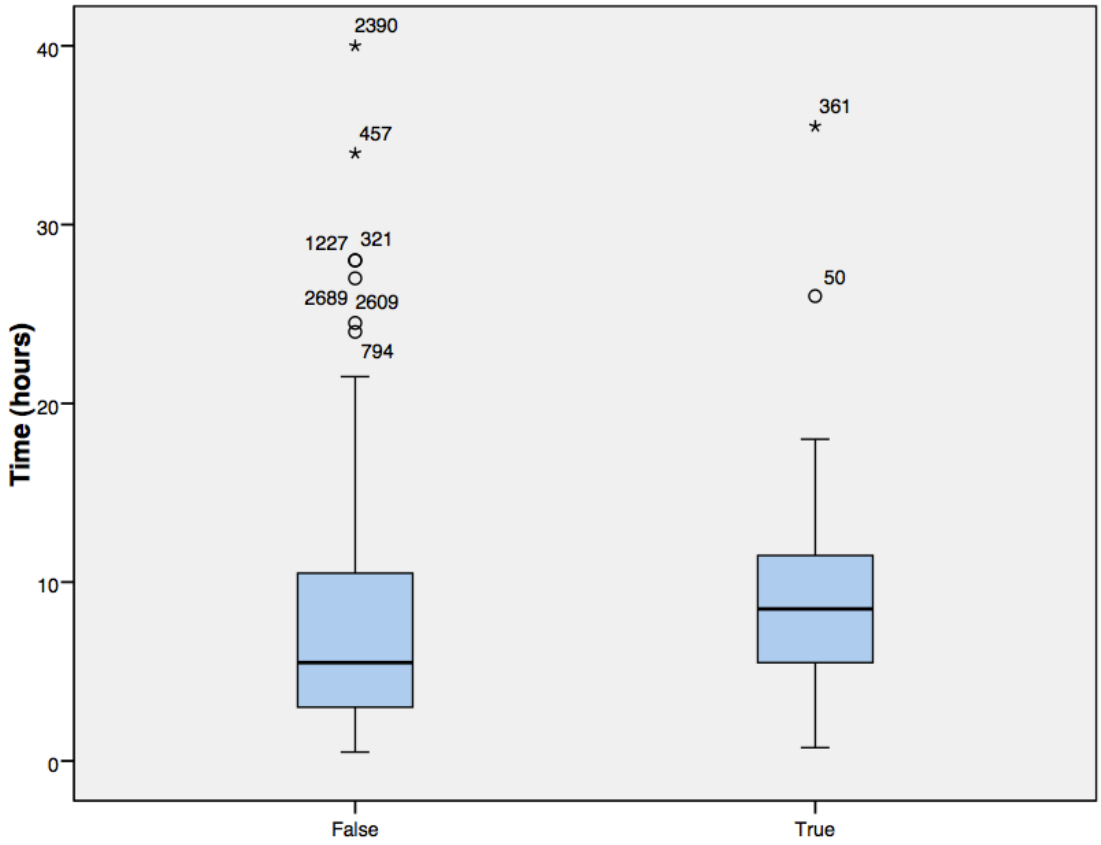
	Chi-Square	df	Sig.
Log Rank (Mantel-Cox)	9,015	1	,003

Test of equality of survival distributions for the different levels of F15: Number of major steps in installation manual.

If we look at the Kaplan-Meier and the significance of the assumption, we can see that the two lines are clearly apart from each other. The table Overall Comparisons of the Mantel-Cox test shows this significance of 0,003. This proves that the hypothesis is valid and that the number of steps negatively impacts engineering time.

Factor F22: Application has organization-specific user-base customization

1. Boxplot and Mann-Whitney test



F22: Application has organization-specific user-based customization

Mann-Whitney Test

Ranks

	F22: Application has	N	Mean Rank	Sum of Ranks
Time (hours)	False	77	46,58	3586,50
	True	19	56,29	1069,50
	Total	96		

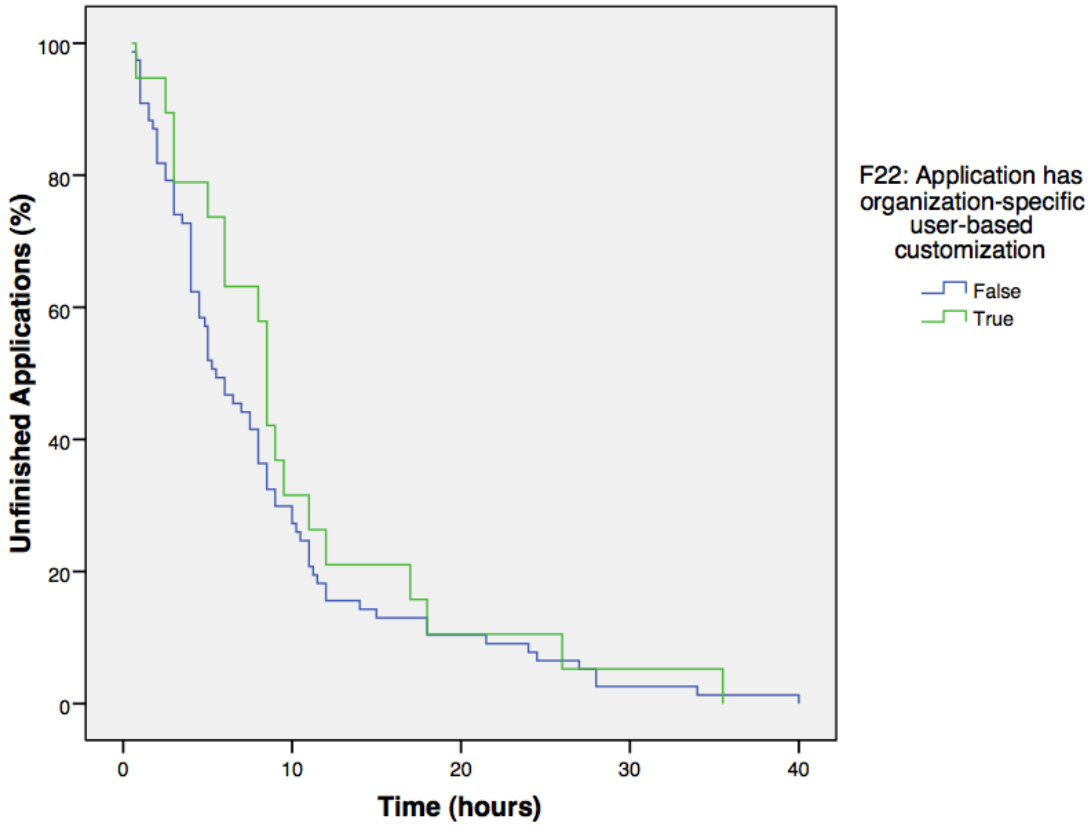
Test Statistics[a]

	Time (hours)
Mann-Whitney U	583,500
Wilcoxon W	3586,500
Z	-1,362
Asymp. Sig. (2-tailed)	,173

a. Grouping Variable: F22: Application has organization-specific user-based customization

With Factor F22 we see a different result. The boxplots are not very different from each other, which suggests that the two groups of applications are very similar. The Mann-Whitney test proves this as there is no significant difference in engineering time whether this factor is true or false. This can be seen in the table of Test Statistics which shows a significance of 0,173.

2. Kaplan-Meier and Mantel-Cox test



Overall Comparisons

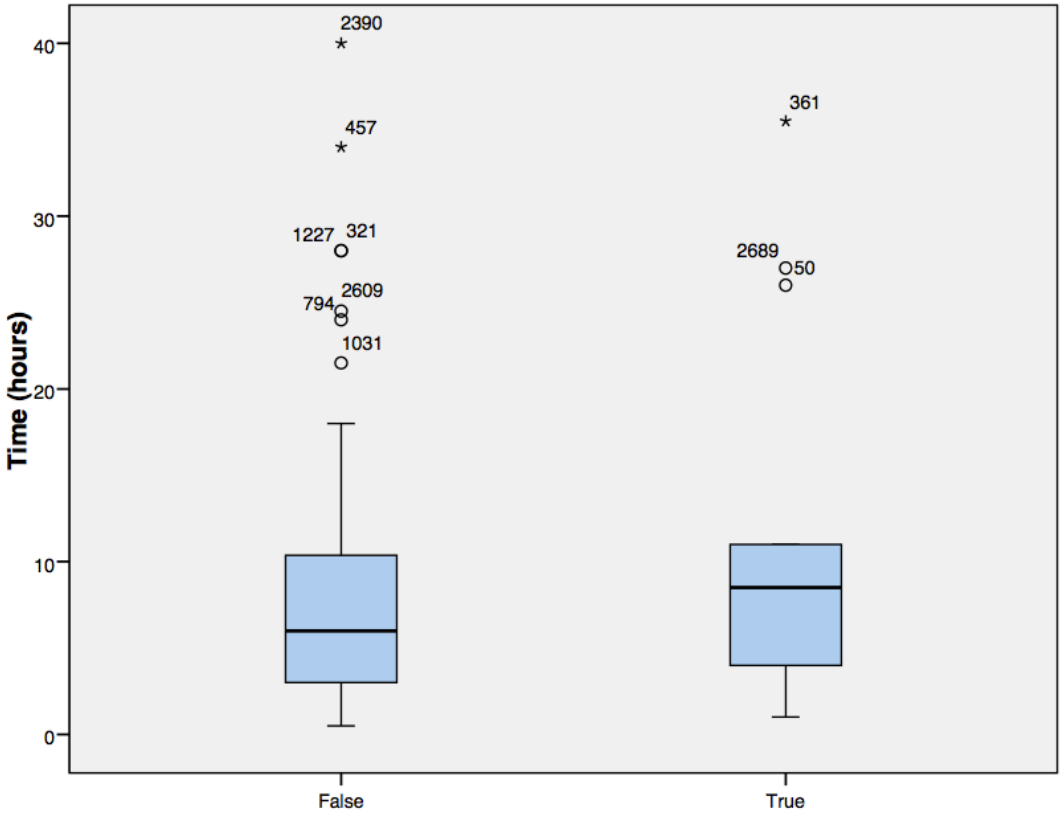
	Chi-Square	df	Sig.
Log Rank (Mantel-Cox)	,869	1	,351

Test of equality of survival distributions for the different levels of F22: Application has organization-specific user-based customization.

In the Kaplan-Meier figure, we can see the same results as could be seen in the Mann-Whitney test for this factor: the lines are very close to each other which suggests that the development times within these groups of applications are very similar to each other.
The Mantel-Cox test shows that we can disprove this hypothesis as we can see in the table Overall Comparisons, which gives us a statistical significance of 0,351.Therefore we can disregard the impact of Factor 22 on engineering time.

Factor F13: Application test manual is not available or incomplete

1. Boxplot and Mann-Whitney test



F13: Application test manual is not available or incomplete

Mann-Whitney Test

Ranks

	F13: Application test	N	Mean Rank	Sum of Ranks
Time (hours)	False	83	47,36	3931,00
	True	13	55,77	725,00
	Total	96		

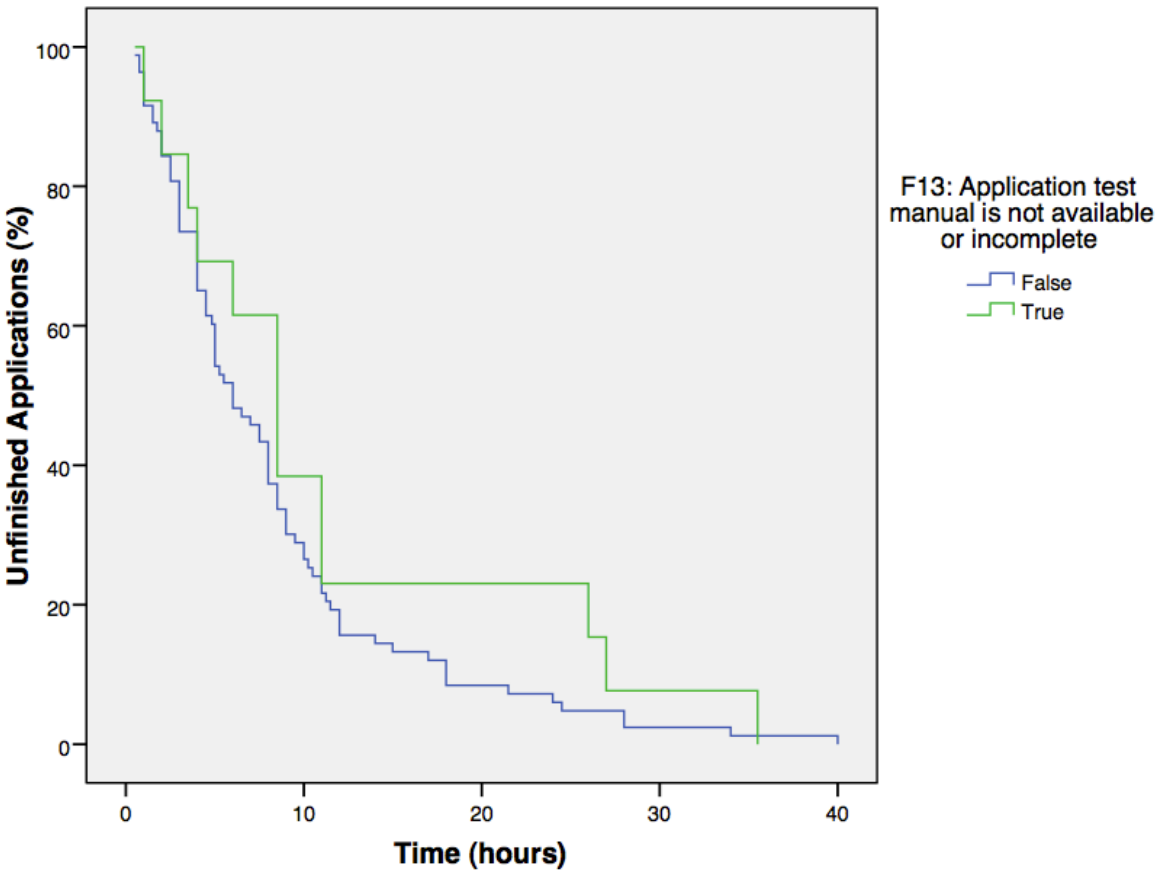
Test Statistics[a]

	Time (hours)
Mann-Whitney U	445,000
Wilcoxon W	3931,000
Z	-1,013
Asymp. Sig. (2-tailed)	,311

a. Grouping Variable: F13: Application test manual is not available or incomplete

With Factor F13 we again see a same result in the boxplots; the boxplots look very similar which suggests that the factor does not have an impact on engineering time. When we look at the found significance in the Test Statistics table of the Mann-Whitney test, we see that the found significance is 0,311. Thus the impact of this factor is not significant.

2. Kaplan-Meier and Mantel-Cox test



Overall Comparisons

	Chi-Square	df	Sig.
Log Rank (Mantel-Cox)	1,265	1	,261

Test of equality of survival distributions for the different levels of F13: Application test manual is not available or incomplete.

The Kaplan-Meier figure shows that there is a difference between the two lines, but it does not look significant. The Mantel-Cox test proves this; the significance value found in the Overall Comparisons table is 0,261 which means we the impact of this factor is again not significant.

Outliers

We created a list of all outliers for all factors and examined why these applications required irregularly more engineering time compared to other applications.

Application 0361

Occurrence: 6 times
Outlier in factor: F24, F07, F01, F15, F22, F13
Possible explanation: This application was incorrectly submitted as validated to the migration process. One of the problems was the fact that the source files were not correct. Furthermore, the requirements regarding the functionality of the application were not complete. As these faults were detected during the last phase of the migration process, the acceptance phase, the application had to be re-engineered. This took a considerable extra amount of time.

Application 0457

Occurrence: 6 times
Outlier in factor: F24, F07, F01, F15, F22, F13
Possible explanation: This application has been engineered several times, due to several inconsistencies in the test plan which should not have resulted in the application validated for migration. Secondly, errors were made by the engineering team which caused extra development time.

Application 2390

Occurrence: 5 times
Outlier in factor: F24, F07, F15, F22, F13
Possible explanation: This application depended on hardware to function correctly. As the hardware was not present during engineering, the engineers had difficulty migrating the application.

Application 0050

Occurrence: 5 times
Outlier in factor: F24, F07, F15, F22, F13
Possible explanation: Not all requirements regarding the application were present during development. This caused the application to not be accepted in the acceptance phase. Also, this application had technical complexities, resulting in more engineering hours.

Application 0794

Occurrence: 5 times
Outlier in factor: F24, F07, F01, F22, F13
Possible explanation: The installation manual and test plan of this application were not correct for the version which was migrated. After development, a new installation manual and test plan were supplied, which were used to create a new version of the application.

Application 1227

Occurrence: 4 times
Outlier in factor: F07, F15, F22, F13
Possible explanation: This application is a client application for other applications. Due to technical issues, this application was created in two specific versions, which took more engineering time. Furthermore, an error was found just before the acceptance phase, so the application had to be slightly re-engineered to fix the error.

Application 2689

Occurrence: 4 times
Outlier in factor: F07, F15, F22, F13
Possible explanation: The application 2689 made use of hardware (a dongle), but during the intake phase the application had only been tested with a new version of the hardware. During the acceptance test the old dongle was used and this did not work correctly. Therefore the application had to be engineered again to accommodate the old dongle.

Application 0321

Occurrence: 4 times
Outlier in factor: F07, F01, F22, F13
Possible explanation: During the engineering phase of this application, there were many technical difficulties migrating the application to the new platform. This was due to the fact that the application could not be deployed according to the project deployment guidelines. A fallback scenario was followed, but this also required extra engineering time.

Application 2609

Occurrence: 4 times
Outlier in factor: F07, F01, F22, F13
Possible explanation: No real explanation can be found for the extra engineering time of this application.

Application 1031

Occurrence: 3 times
Outlier in factor: F07, F01, F13
Possible explanation: The application could only be tested by the application owner. This created difficulties during engineering, as the engineered result could not be tested. This eventually resulted in extra engineering time as the application had to be re-engineered after a failed acceptance test.

Multiple factor impact analysis

After the individual factor analysis, we analyze all factors together to see if they interact. We start the analysis with the Cox-Regression test and then do a stepwise regression analysis to analyze the effects of the results.

Cox regression test

To evaluate the impact of the factors on the development time of the examined applications, we use the Cox regression test. This test shows which factors show, and which do not show a significant correlation with the development times of the applications. For any factor that has a significance under 0,05, we can be at least 95% certain the variance is not coincidental.

Variables in the Equation

	B	SE	Wald	df	Sig.	Exp(B)
F24	-,759	,298	6,489	1	,011	,468
F07	,433	,229	3,584	1	,058	1,542
F01	-1,055	,477	4,893	1	,027	,348
F22	-,278	,321	,750	1	,386	,757
F13	-,550	,397	1,915	1	,166	,577
F15_5_Steps	-,770	,228	11,385	1	,001	,463

The following factors show a significant impact on engineering time (Sig. < 0,05):

- F15: Number of major steps in installation manual (Significance: 0,001)
- F24: Application includes a form of authorization but credentials are not supplied in installation manual (Significance: 0,011)
- F01: Application has a connection to specific hardware (Significance: 0,027)

This also indicates that the partially found significance of factors F07 and F01 have different outcomes from the independent analysis. Factor F07 was partially significant in the independent analysis, but when it is compared with all factors, the effect seems to fade away.

For Factor F01 the opposite is true; the factor was partially significant in the independent analysis but compared with all factors the effect is significant.
Direction and predictive power (Step-wise regression analysis)
To put these results in perspective we use a step-wise regression analysis. In this test, the direction and predictive power is calculated, by calculating the change in effect between different models. The first model consists of just the factor that has the highest effect (in terms of explained variance). The second model adds the factor with the second-highest effect, and the third model adds the factor with the third-highest effect to that. This statistical test shows the effect of the factors on the whole, and in relation to each-other.

		$\boldsymbol{\beta}$	p	R^2	R^2 *(Diff)*
Model 1:	F15	359	0	129	
Model 2:	F15 F01	.379 .280	.000 .003	207	78
Model 3:	F15 F01 F24	.333 .275 .221	.000 .003 .019	253	46

The Step-wise regression analysis indicates that the 3 factors: F01, F15 and F24 all have a positively predicting power ($\boldsymbol{\beta}$ is positive) on the engineering time. This means that when these factors are true, the engineering time will be longer. The coefficient of determination (R^2) of the significantly correlating factors combined is 0,253, which indicates the three significant factors combined can explain 25,3% of the variance in Time.

The order of explained variance in the Step-wise analysis is:

1. F15: Number of major steps in installation manual (R^2: 12,9%)
2. F01: Application has a connection to specific hardware (R^2 difference: 7,8%)
3. F24: Application includes a form of authorization but credentials are not supplied in installation manual (R^2 difference: 4,6%)

This means that Factor F15 can explain 12,9% of the variance in the measure development time. And on top of that F01 and F24, 7,8% and 4,6% respectively, bringing the total explained variance with these three factors to 25,3%.

4. Conclusion

At the start of our research, we asked the following research questions:

1. What is an application portfolio migration?
2. Who are the main actors in these projects?
3. What do experts recognize as possible factors that might effect the time needed for an application migration?
4. How does this compare to a real case study project?
5. What are the opportunities for optimization?

4.1 Analyzing application portfolio migrations

In our analysis of the scientific domain we identified application portfolio migration projects on the meta level: “science of managing a set of tools”. Unfortunately we found no prior studies on these projects so we set-up our thesis to perform exploratory research, with a quantitative component for confirmatory purposes.

To answer research question number one: “What is an application portfolio migration?”, we have described and analyzed an application portfolio migration in the case study and found a parallel of the individual application migrations to traditional software application development strategies. During this application portfolio migration, a form closely resembling the Waterfall model was used to migrate individual applications. As the main purpose of the migration process is to migrate the current functionality to the new platform, requirements are set and requirements rarely change for individual applications. This makes a sequential model suitable for the task. Nevertheless, functional testing before starting the migration process of an application is vital for an efficient application migration.

For research question number two: “Who are the main actors in these projects?” we have described and visualized the interaction of the application portfolio migration with the actors involved. We have established that there are two main actors involved in the application migration process:

- The application owner
- The application migration engineer

Although each application migration only has two actors, the number of actors and stakeholders of the whole application portfolio migration can increase with the total number of applications in the portfolio.

Based on expert interviews we have generated a list of 47 different factors that experts deemed to have an impact on the engineering time needed to migrate an application. Using a consensus method we have ordered this list by importance. This list answered research question three: “What do experts recognize as possible factors that might effect the time needed for an application migration?”.

In Phase 2, we analyzed the top 15 factors in a case study application portfolio migration project to answer research question number 4: “How does this compare to a real case study project?”. By measuring the engineering times of the individual application migrations and analyzing the factors in a research of 96 individual applications, we have positively triangulated several of the findings of phase 1.

Based on the case study research results we can conclude the following:

1. In the case study, three factors had a positively predicting power on the engineering time, explaining around 25% of the variance in time:

   Factor F01: Application has a connection to specific hardware
   Factor F15: Number of major steps in installation manual
   Factor F24: Application includes a form of authorization but credentials are not supplied in the installation manual

2. We found five factors that were prevented by project design. This suggests it is possible to actively reduce possible negatively impacting factors in the application migrations and thus configure application portfolio migration projects for efficiency.

   Factor F05: Application contains legacy device driver installation
   Factor F11: Application makes use of a backend system which is unavailable from within the development environment
   Factor F31: Application installation manual is incomplete
   Factor F39: Application makes use of a backend system, which is different for DTA
   Factor F40: Application makes use of deprecated functions not available on new platform

3. For three factors we could not find any significant correlations in the case study project between the factors and the engineering time of application migrations:

   Factor F07: Application functions in combination with other applications
   Factor F13: Application test manual is not available or incomplete
   Factor F22: Application has organization-specific user-based customization

Looking back on phase 1 with these case study results in mind, we can conclude that our experts were close on target. Independent of the case study project, they have predicted several significantly correlating factors within their top 10:

1. (Highest explained variance) F15 was ranked nr. 9 by experts.
2. F01 was ranked nr. 8 by experts
3. F24 was ranked nr. 2 by experts

If we look at the average engineering time of an application in the case study (8,8 hours), 25% of this could be reduced. This leads to a engineering time reduction of more than 2 hours per application. This can be multiplied with the number of applications within an organization, which can run into the 100s [6]. In our case study, the number of applications was around 350. Therefore, the proposed optimizations could have lead to a reduction of more than 750 engineering hours.

And with larger application portfolio's the total engineering time - and directly with that: engineering costs - can show even larger improvements!

4.1.1 Scientific interpretation of the results

From the top 15 factors identified by the expert we have found positive results for 8 factors in the case study project: 3 positive correlations and 5 prevented factors. We found academic evidence of the possible negative impact of each of these factors.

Signalling and fixing errors as early as possible

Factor F24: “Application includes a form of authorization but credentials are not supplied in installation manual” seems to focus on the need of testing the product. Two other factors describe possibilities for errors: Factor F40: “Application makes use of deprecated functions not available on new platform” and Factor F05: “Application contains legacy device driver installation”. The assumption these factors have an impact seems to be in line with science.

In [55] McConnell states that signaling and fixing errors as early as possible reduces the total fixing effort required. If Factor F24 is true, the signaling of errors during the project is more arduous, impeding the fixing of errors as early as possible, and thus possibly increasing the total fixing effort required. And if errors like Factor F40 and Factor F05 could be prevented beforehand, this would presumably lead to a direct reduction of the fixing effort required, and thus a reduction of the migration engineering time.

Reducing the complexity of the task

Factors F15: “Number of major steps in installation manual”, F01: “Application has a connection to specific hardware”, F11: “Application makes use of a backend system which is unavailable from within the development environment” and F39: “Application makes use of a backend system, which is different for DTA” seem to represent the complexity of the migration task. Wood [56] identifies three sources and three classes of task complexity. In Wood's model, the sources of complexity are:

1. Products (deliverables)
2. Acts (behaviors required to create products)
3. Information cues (knowledge that permits actors to make judgments)

The classes of complexity are the kinds of complexity that can manifest in any of these elements. They are:

1. Component complexity (number of and interdependency among acts and information cues needed to create products
2. Coordination complexity (the frequency, timing, intensity, and interdependencies of sequencing interactions required to produce products)
3. Dynamic Complexity (the degree to which required products, acts, and information cues and the interdependencies among them change during the task).

In [57] the researchers propose the following figure to illustrate Wood's model:

		Classes of Task Complexity		
		Component	Coordination	Dynamic
Sources of Task Complexity	Products	Goals Use Cases Requirements	Derivative Dependencies	Change in vision
	Acts	Life Cycle Activities	Critical Path Dependencies	Changes in Process
	Information Cues	Identify and Resolve Con-flicts Among Requirements	Model Clash	Changes in Meaning

Factor F15 describes the: “Number of major steps in installation manual”. The source that seems to weigh in Factor F15 is the notion of “acts”. The class that seems to weigh in Factor F15 is Component complexity. This links F15 directly to Wood's model as a measure of task complexity.

Reduce dependencies

Factors F01: “Application has a connection to specific hardware“, F11: “Application makes use of a backend system which is unavailable from within the development environment” and F39: “Application makes use of a backend system, which is different for DTA” seem to describe certain dependencies. Following Wood's model, the source of complexity would be “acts”, and the class would be “coordination”. This links F01, F11 and F39 to Wood's model as measures of task complexity.

Validate requirements for completeness

Factor F31: “Application installation manual is incomplete” proposes the incompleteness of requirements as a negatively impacting factor. This corresponds to the findings of [58]. In a study of 1,027 IT projects, he found the number one cause of project failure to be “Unclear objectives and requirements”. This is exactly what Factor F31 seems to cover.

In an application portfolio migration project failure of an individual application migration is not an option. When one application migration has failed, it must be performed again and again until the application is successfully migrated. Any failure due to unclear requirements (read: Factor F31) might lead to the repetition and thus possible blowup of the engineering efforts for that application.

4.2 Optimizing application portfolio migrations

To answer research question number five: “What are the opportunities for optimization?” we propose a model for factor mitigation. For classification purposes, we first define the mitigation of the factors in three different categories:

- Prevention of factor

  The factor can be prevented by project design.

- Reduction of factor

  The negative impact of the factor can be reduced.

- Awareness of factor

  This category consists of factors for which the effects are difficult to prevent or reduce. Decision makers should be aware of the effects of these factors to prevent surprises.

Using the classification we propose a model. The model starts with the positively correlating factors on top, in order of explained variance, for these the effects are the most strongly proved. Following are the other findings, in order of expert-deemed importance. When this model is expanded with more factors correlating in case studies, the bottom factors can be promoted to the top, strengthening the model over time. We categorize the factors and propose a mitigation strategy.

Factor	*Description*	*Mitigation Category*	*Mitigation*
*F15**	Number of major steps in installation manual	Reduction	Reduce the amount of installation steps needed to come to a fully functional installation. For example: in a dialog with the software supplier.
*F01**	Application has a connection to specific hardware	Awareness	As it might not be possible to prevent or solve connection to specific hardware, applications which make use of specific hardware should be avoided as much as possible.
*F24**	Application includes a form of authorization but credentials are not supplied in installation manual	Prevention	Availability of test-credentials should be a standard requirement for all applications.

F11	Application makes use of a backend system which is unavailable from within the development environment	Prevention	Application portfolio migration projects should be performed in an environment which exactly resembles the to-be situation.
F40	Application makes use of deprecated functions not available on new platform	Prevention	Applications should be tested on the target platform, before the technical migration of the application starts.
F31	Application installation manual is incomplete	Prevention	During the intake of an application, installation manual should be tested for validity and completeness.
F39	Application makes use of a backend system, which is different for DTA	Prevention	Application portfolio migration projects should be performed in an environment which exactly resembles the to-be situation.
F05	Application contains legacy device driver installation	Prevention	Applications should be tested on the target platform, before the technical migration of the application starts.

* Proved to have a significant impact in the case study.

4.3 Final thoughts

We have shown that it is possible to provide decision makers with a tool to make a better forecast of the work needed to migrate an application portfolio, and to mitigate possible negatively impacting factors.

The application portfolio is a joint responsibility between business- and IT-departments. Our study proves that choices regarding the applications in an application portfolio can have an impact on the actual effort needed to managing the set of software tools. Our proposed model can help organizations align their operations to a higher level of application portfolio optimization.

4.4 Other uses of our results

The insights of this study can also be used to make better decisions on buying COTS applications. When an application has many installation steps (Factor F15) and has a connection to specific hardware (Factor F01), the application could require more effort to migrate when an application portfolio migration project is initiated, make the change more effort-full and thus lead to a lower level of IT agility. When COTS applications are compared with one another, this insight can be taken into

account to make a better assessment of the Total Costs of Operation of the applications. When our model is further extended even more factors could be taken into account, optimizing the possibilities for deciding which applications to buy.

Software application vendors could make use of our model to further improve their applications and making them more aligned to IT agility. This could give software application vendors a competitive advantage over other software application vendors.

5. Discussion

Based on this study there are vantage points for further research. Some to mitigate possible bias or weaknesses of our study, some to build on and expand the results of our study.

5.1 General discussion

This paper might be the first academic paper in this field of expertise. For there was very limited prior research to base our research on, it is difficult to verify or generalize our results. Further research can focus on verifying our groundwork, and testing the generalizations of our conclusions to other application portfolio migration projects.

The authors of this study are professionally involved with application portfolio migration projects. Though we have made an effort to act as objective as possible, we can not exclude our possible bias in this field as an effect on this study.

Our study focuses on application portfolio migrations in Microsoft Windows environments. Although certain factors could also be valid in other Operating System migrations, further research could focus on investigating the effects in other OS migration projects.

There are probably many more factors that might have an influence on the measured engineering time. These are factors we can not fathom at the moment but can be uncovered with further research.

5.2 Phase 1a: Expert interviews

The inclusion of expert in our research was limited. However, adding more interviewees might have damaged the overall quality of the information thus we chose to stick to the three included experts. Further research could focus on expanding the expert group to get a broader set of possible factors.

The three interviews were structured similarly, but they were still semi-structured. For there was no prior research for us to base a more structured interview technique on, the method used was the most optimal in this research phase. However, the different conversations we had with the interviewees might have biased the factors we elicited. Furthermore, in the teach-back step of the interviews, the knowledge of the authors in the field of expertise might have biased the experts on the nuances of the factors. Further research could build on this study and use our results to build more structured interview techniques.

We continued the interviews to the satisfaction point, when no new factors emerged, but this was just during one interview with each interviewee. In subsequent interviews with the same experts extra factors might have emerged. Multiple interviews per expert could be a strategy in further research to get an even more complete elicitation of factors.

5.3 Phase 1b: Expert consensus

In choosing the method of getting expert consensus we were restricted by the geographical and temporal spread of our experts. Therefore we chose a technique that did not require the experts to be at the same place at the same time. Different expert consensus techniques - for instance where experts are placed together - might have had different interaction results and thus different study results.

We performed two rounds of the Delphi method. More rounds might have induced more expert interaction, and thus a more reliable expert consensus.

In the first step of the Delphi method the authors summarized and consolidated all the elicited factors for the experts to review and confirm them as their own. In this conversion the knowledge of the authors might have biased some nuances of the factors. Although the confirmation of the experts on the list of their own factors mitigates this risk, further research could focus on different consensus techniques that do not require the possibly biased conversion by researchers.

5.4 Phase 2: Case study

In the case study we used the top 15 factors elicited from the experts. Further research could focus on including the other 32 factors too. This might result a larger total explained variance, and thus result in a more complete prediction model. And if it does not result in a larger total explained variance, it might prove that application migrations are not majorly impacted by application specific factors; thus that the time needed for the migration of an application is largely independent on the specifics of the application itself.

The interpretation of the factors into the quantification model could be prone to bias of the authors. Also the interpretation of this quantification model in the documentation research could be prone to errors and bias. Further research could focus to get a more reliable measurement of the factors. An option could be to have the experts themselves quantify all factors on all cases.

The engineering times in the case study were measured manually by the project engineers. This method could be error prone and a subject to confounding factors. Certain engineers could for instance be more efficient in their work, influencing the engineering time. Although effort has been put in to ensure a high quality of this data by monitoring the hours, further research could focus on introducing a more objective measurement of the engineering time.

We measured the factors and engineering times in a case study project at a large Dutch insurance company. The results of this one case study can not be generalized to be representative of other projects in other organizations, with a different project set-up and a different technical context. Further research could focus on reproducing our case study in different projects to find possible generality in the results.

Addendi

Addendum A: Interviews

Interview MH

Factor MH1: Application installation source is not supplied in the MSI format
The installation source supplied in the "Intake" phase is not in the MSI format. It could for instance be a proprietary vendor installation routine, or a manual installation routine.
Other factor occurrences: ST5

Factor MH2: Application has pre-requisites
To set up a fully functional instance of the application, a different application also needs to be installed. Examples of a pre-requisite could be: an application framework, a database client or a host application for a plug-in application.
Other factor occurrences: ST15, MP5

Factor MH3: Application contains legacy device driver installation
The application needs a device driver installation which is based on outdated technology. This driver is needed for the application to function correctly.
Other factor occurrences: ST10

Factor MH4: Application has not been tested on target platform
The installation of the application has not yet been tested on the new platform, or there are uncertainties on the functioning of the application on the target platform.
Other factor occurrences: ST19, MP1

Factor MH5: Size of installation source
The size of the application installation source on disk, excluding unnecessary files.
Other factor occurrences: MP21

Factor MH6: Installation time needed for installation
The time it takes to complete the application installation, when one follows the installation manual step by step.
Other factor occurrences: ST14, MP22

Factor MH7: Number of major steps in installation manual
The number of "major" actions that comprise the installation manual. A major action is an action that has an effect on the application installation, for instance: “Start the installation source”, “Make a configuration change to the application”, “Copy a file”, “Make a Windows dll file registration”. Non-major actions would for instance be: “Open folder X”, “Click the Start-button”, “Wait for the installation to finish”.
Other factor occurrences: MP8

Factor MH8: Number of files in shared locations

The number of files the installation copies to shared locations on the system, like system folders. These shared locations are folders whereto other application installations also install files, thus increasing the risk of conflicts.
Other factor occurrences: ST7

Factor MH9: Application installation causes DLL conflicts
The application installation installs certain DLL files to the target computer, but these are already present in a different version.
Other factor occurrences: None

Factor MH10: Application has a machine or user specific license
The application uses a license compliance enforcement system that ties the application to a specific computer or user. Running the application on another computer or as another user is technically restricted.
Other factor occurrences: MP13

Factor MH11: Application has organization-specific machine-based customization
In the installation manual there are actions to alter the default application machine-based configuration settings that are set after running the installation source. These could be settings that are saved to a file which is not in a user-specific location. The settings are organization-specific and the same for all users of the application.
Other factor occurrences: ST6, MP10

Factor MH12: Application has organization-specific user-based customization
In the installation manual there are actions to alter the default application user-based configuration settings that are set after running the installation source. These could be settings that are saved to a file which is in a user-specific location. The settings are organization-specific and the same for all users of the application.
Other factor occurrences: ST6, MP10

Factor MH13: Application installation manual is incomplete
Certain steps in the installation manual are missing. These missing steps are essential to the correct installation of the application.
Other factor occurrences: None

Factor MH14: Application test manual is not available or incomplete
A step-by-step manual for a functional test of the application is not available to the application migration engineer. For this a functional test of the application is not possible during the application installation development.
Other factor occurrences: MP3

Factor MH15: Application vendor does not supply installation documentation
The application vendor does not supply in-depth information regarding the installation and configuration of their application.
Other factor occurrences: None

Factor MH16: Application installation information is not available online, e.g. communities

There are a number of well known online communities for sharing information regarding application installations. However, for the specific application there is no information present in the online communities, or to be found using an internet search engine.
Other factor occurrences: None

Factor MH17: Application makes use of a backend system which is unavailable from within the development environment
The application needs a connection to certain database or server to function correctly, which cannot be reached during engineering. This could have an effect on the installation or the possibility to execute the test plan.
Other factor occurrences: ST12

Factor MH18: Application includes a form of authorization but credentials are not supplied in installation manual
To fully start and test the application, one must provide valid login credentials, however these valid credentials are not available for use by the technical project team. For this reason the application migration engineer is not able to fully test the application during engineering, without the help of a third party.
Other factor occurrences: None

Factor MH19: Application is externally configured
In addition to the application installation, other systems have been set-up to manage the configuration or other elements of this application. These systems are not installed on the desktop computer itself but are applied using networked systems. Examples include: “Group policies” and “Log-in scripts”.
Other factor occurrences: None

Factor MH20: Application functions in combination with other applications
The application integrates with one or more other applications which make engineering more complex. To be able to validate correct integration of applications, the applications need to be tested with one another.
Other factor occurrences: MP6

Factor MH21: Application needs extra local permissions to function
The application requires extra permissions on the target platform to be able to function properly. This can be due to not complying to application engineering guidelines by the vendor of the application or by the nature of the application.
Other factor occurrences: MP9

Factor MH22: Application installation makes changes to shared network data
The application installation is performed on the network for it to be centrally available. As shared network data is already available, installing the application on the network could hamper correct functionality of the application and disrupt current use of the application. Thorough testing of the installation and functionality of the application is important.
Other factor occurrences: None

Factor MH23: Application has a connection to specific hardware

To fully test the functionality of the application specific hardware needs to be connected to the desktop computer the application is tested on.
Other factor occurrences: ST2, MP15

Factor MH24: Application needs changes to computer BIOS configuration
One of the actions in the installation manual concerns changing a specific BIOS configuration setting. A characteristic of such a configuration change is the necessity to reboot the desktop computer and enter the “BIOS-mode” to make the configuration.
Other factor occurrences: None

Factor MH25: Application installation needs reboot
One of the actions in the installation routine is a reboot of the desktop computer. This reboot is necessary for a complete installation of the application software.
Other factor occurrences: None

Factor MH26: Application installs a server component on the client
The developer of the application has divided the application in a client and server component. In certain cases this server component could be reached by other users on the network. This increases the complexity of the application installation and its availability on the client.
Other factor occurrences: None

Interview ST

Factor ST1: Application has been developed in-house
The application is developed and maintained specifically for use in one organization. Therefore it is possibly only tested for use on the old organization platform.
Other factor occurrences: None

Factor ST2: Application has a connection to specific hardware
To fully test the functionality of the application specific hardware needs to be connected to the desktop computer the application is tested on.
Other factor occurrences: MH23, MP15

Factor ST3: Application makes use of a backend system
To be able to function properly, the application has a connection with different systems. These systems can be application or database servers. As there are more requirements for the application to function properly, this results in more complexity and therefore more testing is needed.
Other factor occurrences: None

Factor ST4: Application makes use of deprecated functions not available on new platform
The application is not fully compatible with the new platform, for it uses deprecated functions. Examples include: use of an unsupported networking stack or dependency on the old “GINA” stack.
Other factor occurrences: None

Factor ST5: Application installation source is not supplied in the MSI format

The installation source supplied in the "Intake" phase is not in the MSI format. It could for instance be a proprietary vendor installation routine, or a manual installation routine.
Other factor occurrences: MH1

Factor ST6: Application has organization-specific customization
In the installation manual there are actions to alter the default application configuration settings that are set after running the installation source. These could be settings that are saved to a file or to the registry. The settings are organization-specific and the same for all users of the application.
Other factor occurrences: MH11 / MH12, MP10

Factor ST7: Number of files in shared locations
The number of files the installation copies to shared locations on the system, like system folders. These shared locations are folders whereto other application installations also install files, thus increasing the risk of conflicts.
Other factor occurrences: MH8

Factor ST8: Number of registry items in system locations
The number of registry items in the system location of the registry could increase the complexity of the installation of the application. This is due to the fact that these items can conflict with other settings or applications.
Other factor occurrences: None

Factor ST9: Application contains self registering com files
Self registering com files are files which are registered by invoking a manual registration command. This action is not recommended, because the registration is not stored centrally which could cause conflicts. Applications which are dependent on these files need to be tested more thoroughly.
Other factor occurrences: None

Factor ST10: Application contains legacy device driver installation
The application needs a device driver installation which is based on outdated technology. This driver is needed for the application to function correctly.
Other factor occurrences: MH3

Factor ST11: Application includes Session 0 Services.
The application installation includes a Session 0 Service. These Services integrate into Windows deeply, and thus might be prone to conflicts and/or incompatibility issues.
Other factor occurrences: None

Factor ST12: Application makes use of a backend system which is unavailable from within the development environment
The application needs a connection to certain database or server to function correctly, which cannot be reached during engineering. This could have an effect on the installation or the possibility to execute the test plan.
Other factor occurrences: MH17

Factor ST13: Size of test plan
The number of major steps in the test plan. A major action is an action that has an effect on the application, for instance: “Start the application”, “Open file X”, “Process file”, “Export file”. Non-major actions would for instance be: “Open folder X”, “Click the Start-button”, “Wait for the processing to finish”.
Other factor occurrences: MP24

Factor ST14: Installation time needed for installation
The time it takes to complete the application installation, when one follows the installation manual step by step.
Other factor occurrences: MH6, MP22

Factor ST15: Application has pre-requisites
To set up a fully functional instance of the application, a different application also needs to be installed. Examples of a pre-requisite could be: an application framework, a database client or a host application for a plug-in application.
Other factor occurrences: MH2, MP5

Factor ST16: Application is not the newest available version of this application
An application which will be installed on the new target platform, is not the latest version currently available. As migrations are typically performed to the newest target platforms, an older version of an application could be incompatible or difficult to install on the new target platform.
Other factor occurrences: None

Factor ST17: Application does not have an application owner available ad-hoc
During the technical migration of this application, the application owner and his replacement are not ad-hoc available for extra information inquiries. This includes unavailability by e-mail and telephone.
Other factor occurrences: MP18

Factor ST18: Application has not been tested on target platform
The installation of the application has not yet been tested on the new platform, or there are uncertainties on the functioning of the application on the target platform.
Other factor occurrences: MH4, MP1

Interview MP

Factor MP1: Application has not been tested on target platform
The installation of the application has not yet been tested on the new platform, or there are uncertainties on the functioning of the application on the target platform.
Other factor occurrences: ST19, MH4

Factor MP2: Different geographical location of application migration engineer and application owner
The application migration engineer who is responsible for the technical migration of the application works on a different geographical location than the application owner. This could be in a different city or different country.
Other factor occurrences: None

Factor MP3: Application test manual is not available or incomplete
A step-by-step manual for a functional test of the application is not available to the application migration engineer. For this a functional test of the application is not possible during the application installation development.
Other factor occurrences: MH14

Factor MP4: Application will be used by multiple users on one machine
The application will be installed on machines, which will be used by more than one user. This can mean multiple users on one machine at the same time, but also multiple users on one machine at different times.
Other factor occurrences: None

Factor MP5: Application has pre-requisites
To set up a fully functional instance of the application, a different application also needs to be installed. Examples of a pre-requisite could be: an application framework, a database client or a host application for a plug-in application.
Other factor occurrences: MH2, ST15

Factor MP6: Application functions in combination with other applications
The application integrates with one or more other applications which make engineering more complex. To be able to validate correct integration of applications, the applications need to be tested with one another.
Other factor occurrences: MH20

Factor MP7: Application is an upgrade to a previously deployed version
The application has already been deployed within the organization. Research needs to be done to find out if and how the application should be upgraded or replaced.
Other factor occurrences: None

Factor MP8: Number of major steps in installation manual
The number of "major" actions that comprise the installation manual. A major action is an action that has an effect on the application installation, for instance: “Start the installation source”, “Make a configuration change to the application”, “Copy a file”, “Make a Windows dll file registration”. Non-major actions would for instance be: “Open folder X”, “Click the Start-button”, “Wait for the installation to finish”.
Other factor occurrences: MH7

Factor MP9: Application needs extra local permissions to function
The application requires extra permissions on the target platform to be able to function properly. This can be due to not complying to application engineering guidelines by the vendor of the application or by the nature of the application.
Other factor occurrences: MH21

Factor MP10: Application has organization-specific customization
In the installation manual there are actions to alter the default application configuration settings that are set after running the installation source. These could be settings that are saved to a file or to the registry. The settings are organization-specific and the same for all users of the application.

Other factor occurrences: MH11 / MH12, ST6

Factor MP11: Application installation describes non-generic customizations
In the installation manual there are actions to alter the default application configuration settings that are set after running the installation source. These could be settings that are saved to a file or to the registry. The settings are not the same for all users of the application, they should differ depending on the user who runs the application or on what specific machine the application is run.
Other factor occurrences: None

Factor MP12: Application will be used on multiple platforms
The application needs to be made available on multiple platforms, for example on a desktop and on a SBC environment.
Other factor occurrences: None

Factor MP13: Application has a machine or user specific license
The application uses a license compliance enforcement system that ties the application to a specific computer or user. Running the application on another computer or as another user is technically restricted.
Other factor occurrences: MH10

Factor MP14: Application installation manual contains steps that cannot be automated
The application requires installation or configuration steps which cannot be automated. This is the case if an application needs user input to finish installation. For example, a user needs to call a phone number for activation.
Other factor occurrences: None

Factor MP15: Application has a connection to specific hardware
To fully test the functionality of the application specific hardware needs to be connected to the desktop computer the application is tested on.
Other factor occurrences: MH23, ST2

Factor MP16: Application includes a form of authorization
The application performs a login or password check at some time during the functional test. This could be an authorization check in a database or other backend system.
Other factor occurrences: None

Factor MP17: Application makes use of a backend system, which is different for DTA
The application makes use of a backend system, but there are different backend systems for “Development, Testing and Acceptance”. The client settings need to be configured manually to use the correct systems in specific situations.
Other factor occurrences: None

Factor MP18: Application does not have an application owner available ad-hoc
During the technical migration of this application, the application owner and his replacement are not ad-hoc available for extra information inquiries. This includes unavailability by e-mail and telephone.

Other factor occurrences: ST17

Factor MP19: Application has files or registry keys already used by another application
The application installation installs files or registry keys that are already installed by another application in the organization.
Other factor occurrences: None

Factor MP20: Application vendor does not supply a customization tool for the application installation
The application vendor does not supply customization software which can be used to create specific customizations to the application installation. These tools are specifically developed for customizing this application installation.
Other factor occurrences: None

Factor MP21: Size of installation source
The size of the application installation source on disk, excluding unnecessary files.
Other factor occurrences: MH5

Factor MP22: Installation time needed for installation
The time it takes to complete the application installation, when one follows the installation manual step by step.
Other factor occurrences: MH6, ST14

Factor MP23: Application installation source is supplied in the MSI format
The installation source supplied in the "Intake" phase is in the Microsoft Installer (MSI) format. This is possibly combined with an MST file.
Other factor occurrences: None

Factor MP24: Size of test plan
The number of major steps in the test plan. A major action is an action that has an effect on the application, for instance: “Start the application”, “Open file X”, “Process file”, “Export file”. Non-major actions would for instance be: “Open folder X”, “Click the Start-button”, “Wait for the processing to finish”.
Other factor occurrences: ST13

Addendum B: Full list of elicited factors

Factor F01: Application has a connection to specific hardware
To fully test the functionality of the application specific hardware needs to be connected to the desktop computer the application is tested on.
(Identified by 3 experts)

Factor F02: Application has not been tested on target platform
The installation of the application has not yet been tested on the new platform, or there are uncertainties on the functioning of the application on the target platform.
(Identified by 3 experts)

Factor F03: Application has pre-requisites
To set up a fully functional instance of the application, a different application also needs to be installed. Examples of a pre-requisite could be an application framework, a database client or a host application for a plug-in application.
(Identified by 3 experts)

Factor F04: Installation time needed for installation
The time it takes to complete the application installation, when one follows the installation manual step by step.
(Identified by 3 experts)

Factor F05: Application contains legacy device driver installation
The application needs a device driver installation which is based on outdated technology. This driver is needed for the application to function correctly.
(Identified by 2 experts)

Factor F06: Application does not have an application owner available ad-hoc
During the technical migration of this application, the application owner and his replacement are not ad-hoc available for extra information inquiries. This includes unavailability by e-mail and telephone.
(Identified by 2 experts)

Factor F07: Application functions in combination with other applications
The application integrates with one or more other applications which make engineering more complex. To be able to validate correct integration of applications, the applications need to be tested with one another.
(Identified by 2 experts)

Factor F08: Application has a machine or user specific license
The application uses a license compliance enforcement system that ties the application to a specific computer or user. Running the application on another computer or as another user is technically restricted.
(Identified by 2 experts)

Factor F09: Application has organization-specific customization
In the installation manual there are actions to alter the default application configuration settings that are set after running the installation source. These could

be settings that are saved to a file or to the registry. The settings are organization-specific and the same for all users of the application.
(Identified by 2 experts)

Factor F10: Application installation source is not supplied in the MSI format
The installation source supplied in the ‘Intake' phase is not in the MSI format. It could for instance be a proprietary vendor installation routine, or a manual installation routine.
(Identified by 2 experts)

Factor F11: Application makes use of a backend system which is unavailable from within the development environment
The application needs a connection to certain database or server to function correctly, which cannot be reached during engineering. This could have an effect on the installation or the possibility to execute the test plan.
(Identified by 2 experts)

Factor F12: Application needs extra local permissions to function
The application requires extra permissions on the target platform to be able to function properly. This can be due to not complying to application engineering guidelines by the vendor of the application or by the nature of the application.
(Identified by 2 experts)

Factor F13: Application test manual is not available or incomplete
A step-by-step manual for a functional test of the application is not available to the application migration engineer. For this a functional test of the application is not possible during the application installation development.
(Identified by 2 experts)

Factor F14: Number of files in shared locations
The number of files the installation copies to shared locations on the system, like system folders. These shared locations are folders whereto other application installations also install files, thus increasing the risk of conflicts.
(Identified by 2 experts)

Factor F15: Number of major steps in installation manual
The number of ‘major' actions that comprise the installation manual. A major action is an action that has an effect on the application installation, for instance ‘Start the installation source', ‘Make a configuration change to the application', ‘Copy a file', ‘Make a Windows dll file registration'. Non-major actions would for instance be ‘Open folder X', ‘Click the Start-button', ‘Wait for the installation to finish'.
(Identified by 2 experts)

Factor F16: Size of installation source
The size of the application installation source on disk, excluding unnecessary files.
(Identified by 2 experts)

Factor F17: Size of test plan

The number of major steps in the test plan. A major action is an action that has an effect on the application, for instance ‘Start the application', ‘Open file X', ‘Process file', ‘Export file'. Non-major actions would for instance be ‘Open folder X', ‘Click the Start-button', ‘Wait for the processing to finish'.
(Identified by 2 experts)

Factor F18: Application contains self registering com files
Self registering com files are files which are registered by invoking a manual registration command. This action is not recommended, because the registration is not stored centrally which could cause conflicts. Applications which are dependent on these files need to be tested more thoroughly.
(Identified by 1 experts)

Factor F19: Application has been developed in-house
The application is developed and maintained specifically for use in one organization. Therefore it is possibly only tested for use on the old organization platform.
(Identified by 1 experts)

Factor F20: Application has files or registry keys already used by another application
The application installation installs files or registry keys that are already installed by another application in the organization.
(Identified by 1 experts)

Factor F21: Application has organization-specific machine-based customization
In the installation manual there are actions to alter the default application machine-based configuration settings that are set after running the installation source. These could be settings that are saved to a file which is not in a user-specific location. The settings are organization-specific and the same for all users of the application.
(Identified by 1 experts)

Factor F22: Application has organization-specific user-based customization
In the installation manual there are actions to alter the default application user-based configuration settings that are set after running the installation source. These could be settings that are saved to a file which is in a user-specific location. The settings are organization-specific and the same for all users of the application.
(Identified by 1 experts)

Factor F23: Application includes a form of authorization
The application performs a login or password check at some time during the functional test. This could be an authorization check in a database or other backend system.
(Identified by 1 experts)

Factor F24: Application includes a form of authorization but credentials are not supplied in installation manual
To fully start and test the application, one must provide valid login credentials, however these valid credentials are not available for use by the technical project team. For this reason the application migration engineer is not able to fully test the application during engineering, without the help of a third party.

(Identified by 1 experts)

Factor F25: Application includes Session 0 Services.
The application installation includes a Session 0 Service. These Services integrate into Windows deeply, and thus might be prone to conflicts and/or incompatibility issues.
(Identified by 1 experts)

Factor F26: Application installation causes DLL conflicts
The application installation installs certain DLL files to the target computer, but these are already present in a different version.
(Identified by 1 experts)

Factor F27: Application installation describes non-generic customizations
In the installation manual there are actions to alter the default application configuration settings that are set after running the installation source. These could be settings that are saved to a file or to the registry. The settings are not the same for all users of the application, they should differ depending on the user who runs the application or on what specific machine the application is run.
(Identified by 1 experts)

Factor F28: Application installation information is not available online, e.g. communities
There are a number of well known online communities for sharing information regarding application installations. However, for the specific application there is no information present in the online communities, or to be found using an internet search engine.
(Identified by 1 experts)

Factor F29: Application installation makes changes to shared network data
The application installation is performed on the network for it to be centrally available. As shared network data is already available, installing the application on the network could hamper correct functionality of the application and disrupt current use of the application. Thorough testing of the installation and functionality of the application is important.
(Identified by 1 experts)

Factor F30: Application installation manual contains steps that cannot be automated
The application requires installation or configuration steps which cannot be automated. This is the case if an application needs user input to finish installation. For example, a user needs to call a phone number for activation.
(Identified by 1 experts)

Factor F31: Application installation manual is incomplete
Certain steps in the installation manual are missing. These missing steps are essential to the correct installation of the application.
(Identified by 1 experts)

Factor F32: Application installation needs reboot

One of the actions in the installation routine is a reboot of the desktop computer. This reboot is necessary for a complete installation of the application software.
(Identified by 1 experts)

Factor F33: Application installation source is supplied in the MSI format
The installation source supplied in the ‘Intake' phase is in the Microsoft Installer (MSI) format. This is possibly combined with an MST file.
(Identified by 1 experts)

Factor F34: Application installs a server component on the client
The developer of the application has divided the application in a client and server component. In certain cases this server component could be reached by other users on the network. This increases the complexity of the application installation and its availability on the client.
(Identified by 1 experts)

Factor F35: Application is an upgrade to a previously deployed version
The application has already been deployed within the organization. Research needs to be done to find out if and how the application should be upgraded or replaced.
(Identified by 1 experts)

Factor F36: Application is externally configured
In addition to the application installation, other systems have been set-up to manage the configuration or other elements of this application. These systems are not installed on the desktop computer itself but are applied using networked systems. Examples include ‘Group policies' and ‘Log-in scripts'.
(Identified by 1 experts)

Factor F37: Application is not the newest available version of this application
An application which will be installed on the new target platform, is not the latest version currently available. As migrations are typically performed to the newest target platforms, an older version of an application could be incompatible or difficult to install on the new target platform.
(Identified by 1 experts)

Factor F38: Application makes use of a backend system
To be able to function properly, the application has a connection with different systems. These systems can be application or database servers. As there are more requirements for the application to function properly, this results in more complexity and therefore more testing is needed.
(Identified by 1 experts)

Factor F39: Application makes use of a backend system, which is different for DTA
The application makes use of a backend system, but there are different backend systems for ‘Development, Testing and Acceptance'. The client settings need to be configured manually to use the correct systems in specific situations.
(Identified by 1 experts)

Factor F40: Application makes use of deprecated functions not available on new platform
The application is not fully compatible with the new platform, for it uses deprecated functions. Examples include use of an unsupported networking stack or dependency on the old ‘GINA' stack.
(Identified by 1 experts)

Factor F41: Application needs changes to computer BIOS configuration
One of the actions in the installation manual concerns changing a specific BIOS configuration setting. A characteristic of such a configuration change is the necessity to reboot the desktop computer and enter the ‘BIOS-mode' to make the configuration.
(Identified by 1 experts)

Factor F42: Application vendor does not supply a customization tool for the application installation
The application vendor does not supply customization software which can be used to create specific customizations to the application installation. These tools are specifically developed for customizing this application installation.
(Identified by 1 experts)

Factor F43: Application vendor does not supply installation documentation
The application vendor does not supply in-depth information regarding the installation and configuration of their application.
(Identified by 1 experts)

Factor F44: Application will be used by multiple users on one machine
The application will be installed on machines, which will be used by more than one user. This can mean multiple users on one machine at the same time, but also multiple users on one machine at different times.
(Identified by 1 experts)

Factor F45: Application will be used on multiple platforms
The application needs to be made available on multiple platforms, for example on a desktop and on a SBC environment.
(Identified by 1 experts)

Factor F46: Different geographical location of application migration engineer and application owner
The application migration engineer who is responsible for the technical migration of the application works on a different geographical location than the application owner. This could be in a different city or different country.
(Identified by 1 experts)

Factor F47: Number of registry items in system locations
The number of registry items in the system location of the registry could increase the complexity of the installation of the application. This is due to the fact that these items can conflict with other settings or applications.
(Identified by 1 experts)

Addendum C: Delphi results

Result from MH

According to you, what is the order of impact of the following possible factors of application migration complexity?

1. Application has a machine or user specific license (Factor F08)
2. Application makes use of a backend system which is unavailable from within the development environment (Factor F11)
3. Application includes a form of authorization but credentials are not supplied in installation manual (Factor F24)
4. Application makes use of deprecated functions not available on new platform (Factor F40)
5. Application installation manual is incomplete (Factor F31)
6. Application installation manual contains steps that cannot be automated (Factor F30)
7. Application makes use of a backend system, which is different for DTA (Factor F39)
8. Application needs changes to computer BIOS configuration (Factor F41)
9. Application installation source is not supplied in the MSI format (Factor F10)
10. Application has a connection to specific hardware (Factor F01)
11. Application will be used on multiple platforms (Factor F45)
12. Application will be used by multiple users on one machine (Factor F44)
13. Application has organization-specific user-based customization (Factor F22)
14. Application does not have an application owner available ad-hoc (Factor F06)
15. Application has organization-specific machine-based customization (Factor F21)
16. Application has organization-specific customization (Factor F09)
17. Application installation describes non-generic customizations (Factor F27)
18. Application functions in combination with other applications (Factor F07)
19. Application includes Session 0 Services. (Factor F25)
20. Application contains legacy device driver installation (Factor F05)
21. Application has files or registry keys already used by another application (Factor F20)
22. Installation time needed for installation (Factor F04)
23. Size of installation source (Factor F16)
24. Application installation needs reboot (Factor F32)
25. Application is an upgrade to a previously deployed version (Factor F35)
26. Application is externally configured (Factor F36)
27. Number of major steps in installation manual (Factor F15)
28. Application installation causes DLL conflicts (Factor F26)
29. Application makes use of a backend system (Factor F38)
30. Application contains self registering com files (Factor F18)
31. Number of files in shared locations (Factor F14)
32. Application installation makes changes to shared network data (Factor F29)
33. Application includes a form of authorization (Factor F23)
34. Application installs a server component on the client (Factor F34)
35. Application needs extra local permissions to function (Factor F12)
36. Number of registry items in system locations (Factor F47)

37. Application has pre-requisites (Factor F03)
38. Application has been developed in-house (Factor F19)
39. Application vendor does not supply installation documentation (Factor F43)
40. Application vendor does not supply a customization tool for the application installation (Factor F42)
41. Application has not been tested on target platform (Factor F02)
42. Application test manual is not available or incomplete (Factor F13)
43. Size of test plan (Factor F17)
44. Application is not the newest available version of this application (Factor F37)
45. Different geographical location of application migration engineer and application owner (Factor F46)
46. Application installation information is not available online, e.g. communities (Factor F28)
47. Application installation source is supplied in the MSI format (Factor F33)

Result from ST

According to you, what is the order of impact of the following possible factors of application migration complexity?

1. Application does not have an application owner available ad-hoc (Factor F06)
2. Application makes use of a backend system which is unavailable from within the development environment (Factor F11)
3. Application vendor does not supply installation documentation (Factor F43)
4. Application includes a form of authorization but credentials are not supplied in installation manual (Factor F24)
5. Application installation manual contains steps that cannot be automated (Factor F30)
6. Application makes use of a backend system, which is different for DTA (Factor F39)
7. Application test manual is not available or incomplete (Factor F13)
8. Application installation manual is incomplete (Factor F31)
9. Application needs changes to computer BIOS configuration (Factor F41)
10. Application makes use of deprecated functions not available on new platform (Factor F40)
11. Application functions in combination with other applications (Factor F07)
12. Application installation makes changes to shared network data (Factor F29)
13. Application contains legacy device driver installation (Factor F05)
14. Application includes Session 0 Services. (Factor F25)
15. Application has a connection to specific hardware (Factor F01)
16. Application has organization-specific user-based customization (Factor F22)
17. Application has organization-specific machine-based customization (Factor F21)
18. Application makes use of a backend system (Factor F38)
19. Application is externally configured (Factor F36)
20. Application is not the newest available version of this application (Factor F37)
21. Number of major steps in installation manual (Factor F15)
22. Application needs extra local permissions to function (Factor F12)
23. Application installation describes non-generic customizations (Factor F27)
24. Application installation causes DLL conflicts (Factor F26)

25. Application has files or registry keys already used by another application (Factor F20)
26. Application has pre-requisites (Factor F03)
27. Application installs a server component on the client (Factor F34)
28. Application has organization-specific customization (Factor F09)
29. Application includes a form of authorization (Factor F23)
30. Number of files in shared locations (Factor F14)
31. Number of registry items in system locations (Factor F47)
32. Application installation source is not supplied in the MSI format (Factor F10)
33. Application has a machine or user specific license (Factor F08)
34. Application contains self registering com files (Factor F18)
35. Application has not been tested on target platform (Factor F02)
36. Application will be used by multiple users on one machine (Factor F44)
37. Application will be used on multiple platforms (Factor F45)
38. Application is an upgrade to a previously deployed version (Factor F35)
39. Different geographical location of application migration engineer and application owner (Factor F46)
40. Application installation information is not available online, e.g. communities (Factor F28)
41. Application vendor does not supply a customization tool for the application installation (Factor F42)
42. Size of test plan (Factor F17)
43. Installation time needed for installation (Factor F04)
44. Size of installation source (Factor F16)
45. Application installation needs reboot (Factor F32)
46. Application has been developed in-house (Factor F19)
47. Application installation source is supplied in the MSI format (Factor F33)

Result from MP

According to you, what is the order of impact of the following possible factors of application migration complexity?

1. Number of major steps in installation manual (Factor F15)
2. Installation time needed for installation (Factor F04)
3. Application has a machine or user specific license (Factor F08)
4. Application has not been tested on target platform (Factor F02)
5. Application has pre-requisites (Factor F03)
6. Application functions in combination with other applications (Factor F07)
7. Application makes use of deprecated functions not available on new platform (Factor F40)
8. Application test manual is not available or incomplete (Factor F13)
9. Application installation manual contains steps that cannot be automated (Factor F30)
10. Application makes use of a backend system which is unavailable from within the development environment (Factor F11)
11. Application includes a form of authorization but credentials are not supplied in installation manual (Factor F24)
12. Application includes a form of authorization (Factor F23)
13. Size of installation source (Factor F16)

14. Application needs extra local permissions to function (Factor F12)
15. Application makes use of a backend system (Factor F38)
16. Application is an upgrade to a previously deployed version (Factor F35)
17. Size of test plan (Factor F17)
18. Different geographical location of application migration engineer and application owner (Factor F46)
19. Application does not have an application owner available ad-hoc (Factor F06)
20. Application is externally configured (Factor F36)
21. Application installation source is supplied in the MSI format (Factor F33)
22. Application has been developed in-house (Factor F19)
23. Application has a connection to specific hardware (Factor F01)
24. Application will be used on multiple platforms (Factor F45)
25. Application installation source is not supplied in the MSI format (Factor F10)
26. Application has organization-specific user-based customization (Factor F22)
27. Application installation information is not available online, e.g. communities (Factor F28)
28. Application contains legacy device driver installation (Factor F05)
29. Application installation needs reboot (Factor F32)
30. Application has files or registry keys already used by another application (Factor F20)
31. Application has organization-specific customization (Factor F09)
32. Application installation causes DLL conflicts (Factor F26)
33. Number of files in shared locations (Factor F14)
34. Application vendor does not supply installation documentation (Factor F43)
35. Application is not the newest available version of this application (Factor F37)
36. Application installation makes changes to shared network data (Factor F29)
37. Application includes Session 0 Services. (Factor F25)
38. Application will be used by multiple users on one machine (Factor F44)
39. Application has organization-specific machine-based customization (Factor F21)
40. Application vendor does not supply a customization tool for the application installation (Factor F42)
41. Application installs a server component on the client (Factor F34)
42. Application needs changes to computer BIOS configuration (Factor F41)
43. Number of registry items in system locations (Factor F47)
44. Application installation describes non-generic customizations (Factor F27)
45. Application installation manual is incomplete (Factor F31)
46. Application makes use of a backend system, which is different for DTA (Factor F39)
47. Application contains self registering com files (Factor F18)

Addendum D: Factor measurement methods

A summary of the Top 15 most impacting deemed factors and their measurements is provided here:

Rank	*Factor*	*Description*	*Method of Observation*
1	F11	Application makes use of a backend system which is unavailable from within the development environment	(Factor prevented by project-design)
2	F24	Application includes a form of authorization but credentials are not supplied in installation manual	Application dossier: *Test plan*
3	F30	Application installation manual contains steps that cannot be automated	Application dossier: Installation Manual / Package notes / Test plan
4	F40	Application makes use of deprecated functions not available on new platform	(Factor prevented by project-design)
5	F06	Application does not have an application owner available ad-hoc	(Factor not measured in case study)
6	F07	Application functions in combination with other applications	Application dossier: Installation Manual / Test plan Deployment tool database
7	F08	Application has a machine or user specific license	Application dossier: Installation Manual / Package notes
8	F01	Application has a connection to specific hardware	Application dossier: Installation Manual / Test plan
9	F15	Number of major steps in installation manual	Application dossier: *Installation Manual*
10	F22	Application has organization-specific user-based customization	Application dossier: *Installation Manual*
11	F13	Application test manual is not available or incomplete	Application dossier: *Test plan* Project Management Tooling

12	F31	Application installation manual is incomplete	(Factor prevented by project-design)
13	F39	Application makes use of a backend system, which is different for DTA	(Factor prevented by project-design)
14	F41	Application needs changes to computer BIOS configuration	Application dossier: Installation Manual / Test plan
15	F05	Application contains legacy device driver installation	(Factor prevented by project-design)

Factor measurement methods

Factor F01: Application has a connection to specific hardware
Source: Application dossier (Installation manual, Test plan)
Measurement explanation: The installation manual and test plan were reviewed to reveal references to hardware devices, like dongles, printers etc. When this is found, this factor is considered TRUE.

Factor F07: Application functions in combination with other applications
Source: Application dossier (Installation manual, Test plan), Deployment tool database
Measurement explanation: This factor can be measured by checking information regarding other applications in the installation manual and the test plan. If found, this factor is considered TRUE. Furthermore, the deployment database is used for further reference. When an application has a dependency or the application itself is a dependency of another application, TRUE is filled in. In other cases, this factor is considered FALSE.

Factor F08: Application has a machine or user specific license
Source: Application dossier (Installation manual, Package notes)
Measurement explanation: When, according to the documentation, license information needs to be filled in and the license is machine or user specific, TRUE is filled in. Otherwise the factor is FALSE.

Factor F13: Application test manual is not available or incomplete
Source: Application dossier (Test plan), Project Management Tooling
Measurement explanation: Test plans of applications are reviewed with the following minimal requirements of a decent test plan, based on the functionality of the application:

- Client applications are considered sufficiently tested if it is possible to make a connection to a server backend.
- Middleware applications are sufficiently tested if an application that depends on this middleware application can be tested.
- For all other applications, testing of starting up the application and testing of the applications core functionality, should be minimally included in the

test plan. The test plan should assess any application specific functionality.

When a test plan is not present or above cases are not met, the value TRUE is filled in. Furthermore when an application was not accepted during acceptance test we will consult the project management tool for the exact reason of the fault. When the cause of the rejection could have been prevented with a more complete test plan, we count this factor as TRUE. Otherwise, FALSE is filled in.

Factor F15: Number of major steps in installation manual
Source: Application dossier (Installation manual)
Measurement explanation: All major steps of an installation manual are counted. All actions where a choice needs to be made, are included as a separate step. Steps that do not add value to the total installation, like clicking on a next or finish button, are disregarded.

Factor F22: Application has organization-specific user-based customization
Source: Application dossier (Installation manual)
Measurement explanation: The installation manual is reviewed to provide information on customization of the application. When an application contains customizations and these customizations are saved in a location which is user specific, TRUE is filled in. In the situation that an application has certain customizations which are not stored in a user specific location, FALSE is filled in. When no customizations are performed FALSE is filled in.

Factor F24: Application includes a form of authorization but credentials are not supplied in installation manual
Source: Application dossier (Test plan)
Measurement explanation: The application test plan is used to determine if an application includes a form of authorization. If the application does not require any kind of authorization, or it does require authorization but the necessary credentials are provided, this factor is considered FALSE. In the case authorization is required but the credentials are not supplied, TRUE is filled in.

Factor F30: Application installation manual contains steps that cannot be automated
Source: Application dossier (Installation manual, Package Notes, Test plan)
Measurement explanation: This factor describes application configuration that cannot be configured in a general way and requires external intervention to work correctly. An application migration engineer is unable to perform this step. This includes, for example, a workstation ID which is specific for a certain workstation, a telephone extension coupled to a certain workstation/user or the need for a manual workstation/user registration in a database for the application to work correctly. When one or more of these kind of situations are found, TRUE is filled in. For the other cases, FALSE is filled in.

Factor F41: Application needs changes to computer BIOS configuration
Source: Application dossier (Installation manual, Test plan)
Measurement Explanation: Some applications require modification to the computer BIOS configuration. When BIOS modification steps in the installation manual or test plan are found, TRUE is filled in.

References

[1] P. Weil, M. Subramani, and M. Broadbent, *IT infrastructure for strategic agility*, Center for Information Systems Research, Sloan School of Management, Massachusetts Institute of Technology, 2002.

[2] M. van Oosterhout, E. Waarts, and J. van Hillegersberg, "Change factors requiring agility and implications for IT", *European Journal of Information Systems*, vol. 15, apr. 2006, p. 132–145.

[3] B. Gray, S. Yates, and C. Kane, Lessons Learned From Windows 7 Early Adopters, Forrester Research, 2010.

[4] M.A. Silver and S. Kleynhans, Creating a Timeline for Deploying Windows 7 and Eliminating Windows XP, Gartner Information Technology Research, 2010.

[5] S. Kleynhans, *Prepare for Windows 7 in Three Phases*, Gartner Information Technology Research, 2009.

[6] G. Riempp and S. Gieffers-Ankel, "Application portfolio management: a decision-oriented view of enterprise architecture", *Information Systems and e-Business Management*, vol. 5, jul. 2007, p. 359–378.

[7] M. Campbell-Kelly and W. Aspray, "Computer: A History of the Information Machine (The Sloan Technology Series)", jun. 2004.

[8] S. Lilley, Men, machines and history: a short history of tools and machines in relation to social progress, Cobbett Press, 1948.

[9] J. Bogoshi, K. Naidoo, and J. Webb, "The oldest mathematical artefact", *The Mathematical Gazette*, vol. 71, 1987.

[10] K.L. Wildes and N.A. Lindgren, A century of electrical engineering and computer science at MIT, 1882-1982, MIT Press, 1985.

[11] A.W. (Arthur W. Burks, "From ENIAC to the stored program computer : two revolutions in computers", 1978.

[12] J. von Neumann, "First draft of a report on the EDVAC", *IEEE Annals of the History of Computing*, vol. 15, okt. 1993, p. 27–75.

[13] A.M. Turing, Scientific Commons: Systems of Logic Based on Ordinals, Oxford University Press, 1939.

[14] A. Newell and H.A. Simon, "Computer science as empirical inquiry: symbols and search", *Communications of the ACM*, vol. 19, mrt. 1976, p. 113–126.

[15] J.W. Backus, H. Stern, I. Ziller, R.A. Hughes, R. Nutt, R.J. Beeber, S. Best, R. Goldberg, L.M. Haibt, H.L. Herrick, R.A. Nelson, D. Sayre, and P.B. Sheridan, "The FORTRAN automatic coding system", feb. 1957.

[16] J. McCarthy, "Recursive functions symbolic expressions and their computation by machine, Part I", *Communications of the ACM*, vol. 3, apr. 1960, p. 184–195.

[17] E.W. Dijkstra, *Notes on structured programming*, Technological University Eindhoven, 1970.

[18] E.W. Dijkstra, *A Discipline of Programming*, Technological University Eindhoven, 1976.

[19] N. Wirth, "The programming language pascal", *Acta Informatica*, vol. 1, 1971, p. 35–63.

[20] B.W. Kernighan and D.M. Ritchie, "The C programming language", *Prentice-Hall Software Series*, 1988.

[21] B. Stroustrup, “Adding classes to the C language: An exercise in language evolution”, *Software: Practice and Experience*, vol. 13, feb. 1983, p. 139–161.
[22] J. Gosling and H. McGilton, “The Java language environment”, *Sun Microsystems Computer Company*, 1995.
[23] O. Nierstrasz, S. Gibbs, and D. Tsichritzis, “Component-oriented software development”, *Communications of the ACM*, vol. 35, sep. 1992, p. 160–165.
[24] T. Reenskaug, “Thing-model-view-editor-an example from a planningsystem”, *Xerox PARC technical note*, vol. 12, 1979, p. 1–11.
[25] W.W. Royce, “Managing the development of large software systems”, *proceedings of IEEE WESCON*, Los Angeles, 1970, p. 1–9.
[26] C. Larman and V.R. Basili, “Iterative and incremental developments. a brief history”, *Computer*, vol. 36, jun. 2003, p. 47–56.
[27] B. Nuseibeh, “Requirements engineering: a roadmap”, *on the Future of Software Engineering*, vol. 1, 2000, p. 35–46.
[28] J. Gordijn and J.M. Akkermans, “Value-based requirements engineering: Exploring innovative e-commerce ideas”, *Requirements engineering*, vol. 8, jul. 2003, p. 114–134.
[29] D. Card and W. Agresti, “Measuring software design complexity”, *Journal of Systems and Software*, vol. 8, jun. 1988, p. 185–197.
[30] H. Delugach and W. Li, “Software metrics and application domain complexity”, Proceedings of Joint 4th International Computer Science Conference and 4th Asia Pacific Software Engineering Conference, 1997, p. 513–514.
[31] R.D. Banker, “Software complexity and maintenance costs”, *Communications of the ACM*, vol. 36, nov. 1993, p. 81–94.
[32] K.H. Bennett and V.T. Rajlich, Software maintenance and evolution: A Roadmap, New York, USA: ACM Press, 2000.
[33] A. Carzaniga, “A Characterization Framework for Software Deployment Technologies”, apr. 1998.
[34] A. Dearle, Software Deployment, Past, Present and Future, IEEE, 2007.
[35] T. Coupaye and J. Estublier, “Enterprise Software Deployment : Foundations & Related Technologies”, *Management*, 2000.
[36] B. Derksen and P. Noordam, “Modellen die werken”, 2007, p. 1–17.
[37] S. Charbonneau, “Software Project Management-A Mapping between RUP and the PMBOK”, IBM-Rational White Paper., 2004.
[38] R.M. Wideman, Comparing PRINCE2® with PMBoK®, APM Group, 2002.
[39] K. Linderman, “Six Sigma: a goal-theoretic perspective”, *Journal of Operations Management*, vol. 21, mrt. 2003, p. 193–203.
[40] R.S. Kaplan and D.P. Norton, Putting the balanced scorecard to work, Harvard Business Review, 1995.
[41] C.E. Schneier, D.G. Shaw, R.W. Beatty, and L.S. Baird, Performance measurement, management, and appraisal sourcebook, Human Resource Development, 1995.
[42] M.C. Paulk, C.V. Weber, S.M. Garcia, M.B. Chrissis, and M. Bush, “Key practices of the capability maturity model SM, version 1.1”, *Office*, 1993.
[43] J. Bamberger, “Essence of the capability maturity model”, *Computer*, vol. 30, jun. 1997, p. 112–114.

[44] E.B. Hyder, K.M. Heston, B. Hefley, and M.C. Paulk, eSourcing Capability Model for Service Providers (eSCM-SP), Van Haren Publishing, 2010.
[45] M. Biro, G. Deak, and J. Ivanyos, “Using the eSourcing Capability Model to improve IT enabled business process outsourcing services”, *informatika.uni-corvinus.hu*, 2003.
[46] B. Johnson and L. Christensen, Educational Research: Quantitative, Qualitative, and Mixed Approaches, SAGE, 2010.
[47] M.D. Dunnette, J. Campbell, and K. Jaastad, “The effect of group participation on brainstorming effectiveness for 2 industrial samples.”, *Journal of Applied Psychology*, vol. 47, 1963, p. 30–37.
[48] G. Rowe and G. Wright, “The Delphi technique as a forecasting tool: issues and analysis”, *International Journal of Forecasting*, vol. 15, 1999, p. 353 – 375.
[49] K. Petersen and C. Wohlin, “The waterfall model in large-scale development”, *Product-Focused Software Process*, 2009, p. 386–400.
[50] F. Hartwig, *Exploratory data analysis*, Beverly Hills: Sage Publications, 1979.
[51] W. Conover, Practical nonparametric statistics, New York: Wiley, 1980.
[52] E.L. Kaplan and P. Meier, “Nonparametric Estimation from Incomplete Observations”, *Journal of the American statistical association*, vol. 53, 1958, p. 457–481.
[53] D.L. Demets, “Tests and Group Sequential Methods at Fixed Use of Logrank Calendar Times”, *Society*, vol. 41, 1985, p. 1039–1044.
[54] M. Paulk, Extreme Programming from a CMM Perspective, 2001.
[55] S. McConnell, Code Complete: A Practical Handbook of Software Construction, Microsoft Press, 2004.
[56] R. Wood, “Task complexity: Definition of the construct”, *Organizational Behavior and Human Decision Processes*, vol. 37, feb. 1986, p. 60–82.
[57] R.O. Briggs and P. Gruenbacher, EasyWinWin: managing complexity in requirements negotiation with GSS, IEEE Comput. Soc, 2002.
[58] A. Taylor, “IT projects: sink or swim”, *The Computer Bulletin*, vol. 42, 2000, p. 24–26.