Facilitating Tap-to-Phone adoption

Towards a portable architecture decision flow for designing a server-based payment architecture

Vince Vissers

Faculty of Technology, Policy and Management

Facilitating Tap-to-Phone adoption
Towards a portable architecture decision flow for designing a server-based payment architecture

Master thesis submitted to Delft University of Technology

in partial fulfilment of the requirements for the degree of

MASTER OF SCIENCE

in **Complex Systems Engineering and Management**

Faculty of Technology, Policy and Management

by

Vince Vissers

Student number: 5170370

To be defended in public on August 19th 2021

Graduation committee

Chairperson	: Dr. A.F. (Aad) Correljé , { EoTI }
First Supervisor	: Dr. A.Y. (Aaron) Ding , {ICT }
Second Supervisor	: Dr. Ir. R (Rutger) van Bergem, {EoTI}
External Supervisor	: Mr. Niels de Vries (Adyen)

This research was conducted as part of the Complex Systems Engineering & Management Master program at:

Faculty of Technology, Policy, and Management
Delft University of Technology

As a graduate intern at:

Point of Sale, Retail and Hospitality
Adyen The Netherlands

Supervisors:	Chairperson	: Dr. A.F. (Aad) Correljé , { EoTi}
	First Supervisor	: Dr. A.Y. (Aaron) Ding , {ICT }
	Second Supervisor	: Dr. Ir. R (Rutger) van Bergem, {EoTI}
	External Supervisor	: Mr. Niels de Vries (Adyen)

Executive Summary

Increased by the global shift away from cash, lifted by the rise of electronic transactions, and amplified by big tech and fintech innovations, global payments revenue is on its way to grow by 5,9% annually until 2028. Opportunities rise in the payment industry and one of the most exciting innovations in the payment chain for retailers today is the potential of utilizing smartphones for mobile Point-of-Sale (mPOS) terminals, providing convenience and ease-of-use. These off-the-shelf devices have already opened the industry, giving consumers a new way of processing payments and giving merchants an easy way to meet demand. However, the fast-changing nature of these innovations and the grey area between hardware and software makes it complex for the payment industry to provide suited payment architecture. Furthermore, it is expensive to redesign payment architecture and therefore the payment industry has difficulties adopting these new solutions. With the prospect of saving money and increasing customer satisfaction on the line, the payment industry has benefits by the simplification of mPOS innovation adoption.

Therefore, to simplify mPOS adoption, the objective of this research is to design a decision flow that provides an overview of the architectural changes possible for both the current as well as the future mPOS solutions. This will be done so by answering the main research question:
"How can one single payment architecture provide a secure payment solution that can be used for the current as well as the future (mobile) Point of Sale (POS) solutions?"

No academic literature can be found on the new mPOS innovations or on a payment architecture that will fit these innovations. Based on the problem highlighted and the knowledge gap, the answers to this main research question are derived by designing a portable payment architecture decision flow. The design decisions enabled by the move towards software terminals and identified by experts are analyzed by determining their effect on stakeholder quality criteria. The architecture decision flow is visualized in Figure 8-5.

The architecture decision flow illustrates a sequence of decisions, indicated by color, that are necessary to adopt the various mPOS solutions. These decisions can be made gradually by iterating an existing architecture and adopting mPOS innovations successively. The decisions can also be made at once, building one architecture that is portable enough to fit all current and future mPOS innovations without the need for expensive architecture iterations.

Designing a portable architecture has been made feasible by the identification of three architecture tradeoffs:

1. Application based Controller versus a modular server-based Controller

By rebuilding the Controller in a modularized manner and using rule-based coding, it becomes possible to efficiently move parts of the Controller to the server. To decrease latency a continuous Websocket connection should provide the communication between the server and the application. This results in improved portability, maintainability and security of the mPOS application but dependent on network variables it could increase communication latency.

2. Using hardware-based security versus using software-based and operating system security

By using software-based security the application no longer is reliable on the hardware provided by the smart device. Combining the software-based security with security provided by the operating system will result in an almost similar level of security. This will increase the maintainability and portability of the application.

3. Application based Kernel versus a server-based Kernel

Because of the adoption of Websocket communication, 5G internet and contactless payments it becomes feasible to move the transaction Kernel to the server. This will improve the portability, security and maintainability of the application but will likely also increase latency.

Furthermore, moving the various components to the server simultaneously has proven to compensate for the increase in communication. A significant part of the communication between components can now happen directly in the server. Therefore, communication between the device and the server can be combined. This reduces the added latency and makes the solution feasible in more scenarios.

In conclusion, the overview presented by the decision flow creates awareness and manages expectations for organizations that want to keep innovating their POS solution. The demonstrated combination of design decisions resulted in an architecture that can be used for all above mentioned mPOS devices. This allows PSP's to be flexible in their decisions and therefore simplifies the adoption of new mPOS innovations. Furthermore, the combination of technological innovations determined in this research that allow for the relocation of the various components result in several advantages. Moving components away from the user and into internal servers gives direct control to the solution builder. This will significantly increase the efficiency in testing, updating, certification, configuration and data monitoring of the mPOS solution.

The following paragraphs will sequentially discuss the evaluation, academic and societal contribution, recommendation and reflection.

In the evaluation phase, an expert panel found common ground on the societal and scientific contribution of this research. They agreed that the output of this research is generalizable to the full payment industry in Western-Europe and acknowledged its significance outside of the Western-European scope.

By illustrating the necessary steps for building a flexible architecture the current literature on payment solutions is made more recent. This results in providing an academic base for the implementation and testing of a server-based payment architecture. Furthermore, this research creates awareness of the potentials of the upcoming mPOS innovations and provides a direction of new academic research.

Moreover, this research provides significant benefits to society from a number of perspectives. By simplifying the adoption of new mPOS technologies shoppers will experience improved check out experiences. Furthermore, the output of this research decreases the need for expensive payment architecture iterations for PSP's. Additionally, the implementation of the proposed architectural changes will cause the number of devices capable of accepting payments to increase significantly. This will decrease the need for cash as well as for additional hardware. This improves sustainability and in the long run decreases the barrier for unevolved areas to be included in the payment infrastructure.

With the growth of new technologies such as 5G, Edge Computing and contactless transactions, a consistent global network comes within reach. This provides the opportunity for not only the industry to use these mPOS solutions but also the consumers. By enabling consumers to both accept and conduct payments using a smartphone, there will be no need for cash anymore. This will result in the inclusion of the global society in the payment infrastructure and excluding the need for cash payments by simply downloading an application. Although this seems like an ambitious goal, this research has proven the theoretical feasibility of this goal.

Following this research, the researcher will proceed with further study on the quantification of the effects on quality criteria derived out of this research. This will be done by implementing the proposed architecture decisions and measuring their effects in different scenarios. This research will be conducted at the payment service provider Adyen in collaboration with the TU Delft. By measuring and quantifying the effects of the proposed architecture changes, it becomes possible to determine the environmental and infrastructural variables necessary for successful implementation.

Keywords: mPOS, MobilePOS, SoftPOS, Cloud, Software architecture, payments.

Preface

The past six months have enabled me to make use of the full set of skills that the CoSEM program has provided me with by diving into the upcoming world of software architecture and cloud payments. During the process of this research my enthusiasm for the subject of cloud-based payment grew and was supported by the growing interest I received by the payment industry. This helped me in convincing my surrounding of the importance of this subject and resulted in the opportunity of implementing this research in the coming years. I am very grateful to all the people that have supported me on this journey and therefore helped me in reaching the next big step, a job at a successful payment service provider.

For this and many other reasons, I would like to thank and acknowledge my first supervisor, Aaron Ding. Aaron has been involved in the project from the start and was able to drive me to my limits. Aaron allowed me the autonomy and freedom necessary to conduct this research while also providing guidance when needed. Especially during Covid period it was nice to be able to have someone that fully understood the pain points of this research subject. Aaron helped me to look further but maybe even more important, to create boundaries and to work towards an end goal. It was a very enjoyable collaboration. Moreover, I would like to thank my second supervisor Rutger van Bergem and Chairman of the committee Aad Correlje. Rutger and Aad have provided me with a critical and essential view. Aad especially for sharing his experience with large projects and pushing for a smaller scope at the beginning. Without this insight the project would probably still be in its start. Rutger, especially for his viewpoint resulting in the addition of business opportunities and competition. Rutger's comments have broadened my view and brought my attention to the significance of the opportunity that this research can bring. This helped me to also look beyond the academical opportunities.

I would also like to thank my external supervisor Niels de Vries. Niels has been a source of inspiration and knowledge for this research and has kept pushing me during the research process. Niels has had a very good sense of what was necessary to succeed my project and supported me in every way. Furthermore, I would like to thank Derk Busser, for providing me with the right subject and for, together with Niels, providing me with the access to more expertise than I could have hoped for. Lastly, I would like to emphasize how important all participants and experts have been for this research and thank them for their help. The outcome of this research has been fully dependent on their expertise and could not have been successful without them. I am very much looking forward to continuing our collaboration by successfully implementing this research. I hope this will just be the start of a journey towards a fully portable payment solution.

Vince Vissers
Delft, August 1, 2021

Table of Contents

EXECUTIVE SUMMARY **4**

PREFACE **7**

LIST OF FIGURES **13**

LIST OF TABLES **15**

GLOSSARY **17**

1 INTRODUCTION **20**

1.1 BACKGROUND INFORMATION AND SIGNIFICANCE 21
1.1.1 Societal significance and business opportunity 23
1.2 PROBLEM STATEMENT AND KNOWLEDGE GAP 24
1.3 MAIN RESEARCH QUESTION AND SUB-QUESTIONS 25
1.4 THE SCOPE OF THIS RESEARCH 26
1.5 DELIVERABLES 28
1.6 SUMMARY OF THE INTRODUCTION 29

2 RESEARCH DESIGN AND METHODOLOGY **30**

2.1 RESEARCH OBJECTIVES 30
2.2 RESEARCH DESIGN, A DSRM PROCESS MODEL 30
2.3 RESEARCH PHASES AND RESEARCH FLOW 31
Phase 1: Motivation and identification of the problem 31
Phase 2: The objectives of a solution 32
Phase 3: The design and development phase 33
Phase 4: Demonstration of the design 33
Phase 5: Evaluation of the design 34
Phase 6: Communication 34
2.4 DESIGN OBJECTIVES 34
2.5 DATA GATHERING METHODOLOGY 35
2.5.1 Continues architecture methodology 35
2.5.2 Methodology for gathering quality criteria 36
2.5.3 Literature research 37
2.5.4 Interviews with industry experts 38
2.5.5 Brainstorm sessions and expert panels 38
2.5.6 Qualitative data analysis 38
2.6 SUMMARY OF THE RESEARCH DESIGN AND METHODOLOGY 39

3 (M)POS SOLUTIONS AND ARCHITECTURES **40**

3.1 LITERATURE REVIEW 40
3.1.1 Literature review methodology 40
3.1.2 Literature on payment solutions 42
3.2 RESEARCH GAP 44
3.3 CLASSICAL MOBILE POS SOLUTIONS 44
3.3.1 From hardware to software 44
3.4 (M)POS INNOVATIONS 45
3.4.1 (M)POS in components 46
3.4.2 PIN on COTS flow explained 48
3.4.3 Path to offering a full solution 49
3.5 BACKGROUND INFORMATION ON UTILIZED ARCHITECTURE TECHNOLOGY 50
3.5.1 Latency 50
3.5.2 HTTP versus Websocket connections 51
3.5.3 Terminal Driver and Terminal Controller 52
3.5.4 Rule Based Engine 52
3.5.5 The Transaction Kernel 54
3.6 SUMMARY OF (M)POS SOLUTIONS AND ARCHITECTURES 58

4 DATA GATHERING AND METHOD DEVELOPMENT 59
4.1 GATHERING DATA 60
4.1.1 Joint Requirements Development sessions 60
4.1.2 Merchant Interviews 60
4.1.3 Choosing participants 61
4.1.4 Interview process. 64
4.1.5 Qualitative data analysis 64
4.2 SUMMARY OF DATA GATHERING AND METHOD DEVELOPMENT 65
5 TOWARDS A PORTABLE MPOS ARCHITECTURE 66
5.1 COMBINING THE STAKEHOLDER'S ATTRIBUTES 66
5.1.1 Interview and expert panel analysis 67
5.1.2 Security and certification requirements 67
5.1.3 Visualizing quality attributes 68
5.1.4 Quality attributes utility tree 68
5.2 GATHERING EXPERTISE AND CONTINUES FEEDBACK 71
5.2.1 Expert interviews 71
5.2.2 Interview process. 72
5.2.3 Interview guideline 73
5.3 THE PROCESS OF IDENTIFYING ARCHITECTURE TRADE-OFFS 74
5.4 DRAFTING THE ARCHITECTURE OUTLINE 74
5.5 ARCHITECTURE DESIGN OPTIONS 76
5.6 SECURITY 76
5.6.1 Architecture design options 77
5.6.2 Hardware based security 77
5.6.3 Software security 78
5.6.4 Security reliant on COTS Platform 78
5.6.5 Tradeoff analysis 78
5.6.6 Summary of the security decisions and effect on quality criteria 80
5.7 APPLICATION 81
5.7.1 Controller and Drivers 82
5.7.2 Option 1: Controller in the application 83
5.7.3 Option 2: Moving Controller to the server 84
5.7.4 Tradeoff 85
5.7.5 Summary of the Controller design options and their effect on quality criteria 86
5.8 KERNEL 87
5.8.1 CPOC versus SPOC 89
5.8.2 The cloud Kernels 90
5.8.3 The effects of moving the Kernels to the server 91
5.8.4 Tradeoff 92
5.8.1 Summary of the Kernels design options and their effect on quality criteria 94
5.9 COMBINING THE KERNEL AND CONTROLLER ARCHITECTURE 95
5.10 COMBINING TRADEOFFS INTO AN ARCHITECTURE DECISION FLOW 97
5.10.1 Architecture decision sequence 97
5.10.2 Design decision flow diagram 99
5.10.3 Classical terminal → SCR with a COTS device 100
5.10.4 SCR with a COTS device → SPOC 101
5.10.5 SPOC → CPOC 102
5.11 SUMMARY OF CHAPTER 5 103
6 DEMONSTRATION AND EVALUATION 105
6.1 DEMONSTRATION OF A FULLY PORTABLE ARCHITECTURE 105
6.1.1 Demonstration scenario 105
6.1.2 Demonstrated architecture 106
6.2 EVALUATION PROCESS 109
6.3 EXPERT PANEL 109
6.4 FULFILLMENT OF OBJECTIVES 110

6.4.1 Evaluation checklist *111*
6.5 FINDINGS AND RECOMMENDATIONS 111
6.5.1 Determining fulfillment of objectives *111*
6.5.2 Strengths *112*
6.5.3 Recommendations and limitations *112*
6.5.4 Evaluation method *113*
6.6 SUMMARY OF THE DEMONSTRATION AND EVALUATION 113

7 DISCUSSION 114

7.1 GENERALIZATION 114
7.2 CONNECTION TO THE EXISTING LITERATURE 114
7.3 LIMITATIONS 116
7.4 ADDED VALUE 117
7.5 OPEN RESEARCH PROBLEMS 118

8 CONCLUSION 120

8.1 A SUMMARY OF THE ANSWERS TO THE SUB-RESEARCH QUESTIONS 121
8.2 SCIENTIFIC CONTRIBUTION 124
8.3 SOCIETAL CONTRIBUTION 125
8.4 REFLECTION AND FUTURE RESEARCH 126

REFERENCES 128

APPENDICES 133

8.5 APPENDIX A: FUNCTIONAL REQUIREMENTS FROM PSP EXPERTS. 133
8.6 APPENDIX B ATLAS TI MERCHANT FINDINGS 136
8.7 APPENDIX D APPLICATION VERSUS SDK 140
8.8 APPENDIX E PROJECT TIMELINE 143
8.9 APPENDIX F TRANSACTION FLOW DIFFERENCES BETWEEN SDK AND LIGHTWEIGHT APPLICATION 145

List of Figures

Figure 1-1 (m)POS innovations....................22
Figure 1-2 mPOS enabling trends....................23
Figure 1-3 Thesis flow diagram....................28
Figure 2-1 DSRM process model (Peffers et al., 2007)....................30
Figure 2-2 Research Flow Diagram....................31
Figure 2-3 research phases....................35
Figure 3-1 NFC application: Pourghomi et al.....................43
Figure 3-2 from software to hardware....................45
Figure 3-3 The move towards fully software (SPOC & CPOC)....................46
Figure 3-4 SPOC with PIN flow....................48
Figure 3-5 Terminal Drivers and Terminal Controller....................52
Figure 3-6 Rule Based Engine....................53
Figure 3-7 Kernels detailed architecture....................55
Figure 3-8 Process P....................56
Figure 3-9 Process D....................57
Figure 3-10 Process S....................57
Figure 3-11 Process K....................58
Figure 5-1 Quality code groups....................67
Figure 5-2 Quality Attributes Utility Tree....................69
Figure 5-3 Movable POS components....................75
Figure 5-4 Security components....................77
Figure 5-5 Modularized Controller design choice....................82
Figure 5-6 Controller – driver transaction flow classical terminal....................84
Figure 5-7 Controller – driver transaction flow option 2....................85
Figure 5-8 Kernel design options....................88
Figure 5-9 Kernels detailed architecture....................89
Figure 5-10 Cloud Kernel....................90
Figure 5-11 Cloud Kernel transaction flow....................92
Figure 5-12 Controller and Kernel synergy....................95
Figure 5-13 Controller and Kernel synergy transaction flow....................96
Figure 5-14 Design Choices Flow diagram....................99
Figure 6-1 Flexible Architecture....................108
Figure 8-1 Answering sub-questions....................121
Figure 8-2 Significant mPOS innovations....................122
Figure 8-3 Quality attributes utility tree....................122
Figure 8-4 Portable architecture....................123
Figure 8-5 Architecture decision flow....................124

List of Tables

Table 3-1 Literature review 42
Table 3-2 the relevant future mPOS solutions 49
Table 3-3 Kernel processes 55
Table 3-4 Kernel responses (EMVCO Book C-2, 2021) 56
Table 4-1 Merchant expert interviewees 62
Table 4-2 Subject matter experts 63
Table 4-3 Interview process 64
Table 5-1 Expert interview guideline 73
Table 5-2 Security experts 76
Table 5-3 Security design options and quality criteria 80
Table 5-4 Application experts 81
Table 5-5 Application design options 87
Table 5-6 Kernel experts 88
Table 5-7 Kernel Design Choices 94
Table 5-8 Sequence of mPOS innovations and requirements 98
Table 5-9 Effects of moving from the classical terminal to the SCR 100
Table 5-10 Effects of moving from SCR to SPOC 101
Table 5-11 Quality criteria effects Kernel 102
Table 6-1 Quality criteria effects Demonstration 107
Table 6-2 Evaluation expert panel 110
Table 0-1 Merchant requirements 139

Glossary

Term	*Definition*
Attestation	"The act of attestation is the interaction between a verifier (possibly server-based) and a prover (possibly client-based) to determine the current security state/behavior of the prover based on predefined measurements and thresholds provided by the prover" (PCI, 2019)
Commercial off-the-shelf (COTS) device	A mobile device (i.e. tablet or smartphone) that is designed for mass-market distribution
Contactless payments on COTS (CPoC) application	"All parts of the code, regardless of the execution environment, that is installed and executed on the merchant COTS device for the purposes of accepting and processing account data associated with a contactless transaction. The CPoC API, attestation component, and/or a payment application may be incorporated into the CPoC application or may be separate."(PCI, 2019)
Contactless Payments on COTS (CPoC) solution	"The set of components and processes that supports the contactless read and protection of account data into a COTS device. At a minimum, the solution includes the CPoC application, attestation system, and the back-end systems and environments that perform attestation, monitoring, and payment processing. "(PCI, 2019)
NFC Interface	"The subsystem in the COTS device that is used by the COTS platform to access data, including account data, read from contactless cards or devices. The main physical components are the NFC antenna and the NFC Controller. "(PCI, 2019)
Point of Sale (POS) solution	Place where the transaction takes place. Often a payment terminal with a cash register.
Payment service provider (PSP)	Company that provides payment services (i.e. Adyen, Wirecard, Paypal)
Tap-on-phone	Accepting payments on a COTS device (i.e. by tapping your card on a smartphone)
Payment Card Industry (PCI) security standards council	The general authority on conducting transactions. This organization sets guidelines on how transactions should happen on a secure basis.

Merchant	Any company that sells services or products using the solutions provided by a payment service provider.
SCR(P)	Abbreviation for Secure Card Reader. A physical card reader that has been assessed compliant to the PCI PTS POI device in SCR Approval Class.
SoftPOS	The technology that allows people to accept payment on a COTS device without any additional hardware.
Secure Reading and Exchange of Data (SRED)	The SRED ensures that cardholder account data is protected at the point of acceptance, which will assist in meeting the required security considerations of the wider point-to-point security process.
Whitebox cryptography	A form of cryptography that allows for cipher keys to not be revealed.
CVM	Cardholder Verification Method; a process used to confirm that the person presenting the Card is an authorized Cardholder. In certain Card-present environments, a Merchant may complete the Transaction without a CVM ("no CVM" as the CVM), such as Contactless Transactions whose amount is below the Contactless CVM limit
API	Application programming interface, is a software interface that makes it possible for two separate applications to communicate.
Kernel	The Kernel contains logic to manage a set of commands and responses to retrieve the necessary data from the card. The Kernel processing covers the interaction with the Card between the selection of the card application and processing of the outcome of the transaction.
mPOS	Category of POS terminal, where merchants use a COTS device (mobile phone or tablet). Considering their use cases (for example contactless only and/or intended use in micro merchants), MPOS terminals may be subject to specific authorization or deployment requirements
NFC	Near Field Communication: the contactless communication standard between the COTS device and the contactless Card or Access device

1 Introduction

Increased by the global shift away from cash, lifted by the rise of electronic transactions, and amplified by big tech and fintech innovations, global payments revenue is on its way to grow by 5,9% annually through 2028 (Sushil et al., 2019). This creates a market for innovating fintech companies and accelerates payment innovations rapidly.

In this radically changing world of payment service providers, various innovations for retailers in the payment chain today increase the potential of utilizing a commercial of the shelf (COTS) device. For example, a smartphone for Point-of-Sale (POS) payment solutions. Instead of accepting payments using classical terminal devices, a shift is upcoming enabling a mobile point of sale solution that allows you to accept payments using your COTS device (Hartog & Moreno, 2019).

In the last few years, a number of new Payment Card Industry (PCI) certification requirements have been introduced. These allow a shift of responsibility and security for in-store card-based transactions from hardware based to (partially) software-based solutions. These solutions can roughly be divided into three types of solutions/products: The Secure Card-Reader for PIN (SCRP), the Software-Based PIN entry on COTS (SPOC) and Contactless Payments on COTS (CPOC). At this point it is not yet allowed to combine PIN Entry and Card Reading on the same COTS device due to security reasons (PCI, 2019). For the proceeding of this research these innovations combined will be referred to as a mobile Point-of-Sale (mPOS) solution.

The move to software based, off the shelf technology (CPOC with PIN Entry) is seen as the (long-term) future of in-store payment transactions for both enterprise and micro-merchants. However, most known Payment Service Providers (PSP) such as Adyen or Stripe do not support this feature yet (Bruno et al., 2019) (Hartog, Moreno, 2019). These PSPs see the slow adoption of mPOS solutions as a problem because many merchants are already familiar with the new innovations and are looking for cheaper and more attractive looking solutions than the currently used terminals (interview, Chapter 5.4).

A reason for the slow adoption of these solutions is their complexity. The transition from accepting payments on a closed POS system towards accepting payments on a commercial off-the-shelf (COTS) device involves many different architectural requirements. Various informal discussions with payment system vendors have indicated that the current variety in possibilities leads to a chaotic process of adoption by the bigger PSP's. The CEO of one of the biggest payment software companies in the world clearly summarized the issue in a quote:

"*The current SoftPOS industry can currently best be compared with the wild, wild west. Everybody claims to have a different solution and nobody knows which way to go anymore*" *(Anonymous,2021)*

Various payment architecture layouts are possible in which different types of hardware and software solutions are necessary to fulfil security requirements (PCI, 2019). Moreover, investing in future proof payment architecture can be complex due to the fast-changing security requirements for the different upcoming COTS payment solutions. Due to the move from hardware to software it has become unclear where in the overall architecture certain POS functionality should live and whether that is part of the scope of your company. Therefore, it is difficult for PSP’s to start investing in the development of a new payment infrastructure without knowing if it will be sufficient for the coming innovations.

Due to the complex nature of these innovations various different issues arise. It is difficult for the payment security authority to provide a single set of requirements for the various solutions (Hartog, Moreno, 2019). Furthermore, it is hard for PSP’s to decide on how to innovate. Which functionality should be bought and which functionality can be built in the company’s server? There is no clear software solution that is portable enough to be used for the current mPOS innovations. Therefore, the POS users or merchants cannot use the available, more attractive and cheaper mPOS solutions.

Complex systems engineering is necessary to provide an overview of the current and upcoming solutions and design an architecture decision flow that can support PSP’s in making design decisions and gain an oversight in the architectural possibilities for mPOS. The Design Science Research Methodology (DSRM) provides the necessary support for designing a flexible and portable mPOS architecture.

Not all PSP’s require the full mPOS architecture and various PSP’s have different use-cases for their mPOS solution. Therefore, this research will not focus solely on developing a flexible mPOS architecture but rather on developing the process towards it. The objective of this research is to design a decision flow that provides an overview of the architectural changes possible for both the current as well as the future mPOS solutions. This will be done so by answering the question *“How can one single payment architecture provide a secure payment solution that can be used for the current as well as the future (mobile) Point of Sale (POS) solutions?”*

The output of this research will simplify and therefore increase mPOS adoption by PSP’s. It will do so by assisting the payment industry in making complex ‘buy or build’ decisions and reduce costs on iterating payment innovations to fit new standards. Furthermore, by combining new but existing technologies for architectural purposes this design research will identify multiple research gaps that can provide more insight in the disruptive potential of a server-based POS system.

1.1 Background information and significance

Using smartphones for payments is not a new innovation. With the rise of the NFC chip in smartphones most people in the Netherlands are able to pay using their smartphone (Hartog & Moreno, 2018). However, this innovation only allows people to pay with their smartphone. What has changed is that new innovations now also allow people to accept payments on their smartphone.

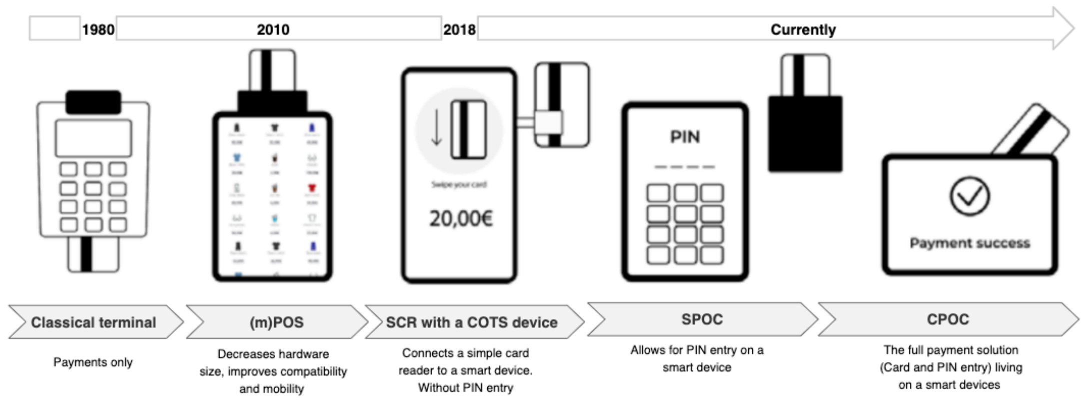

1980
2010
2018
Currently
20,00€
PIN
Payment success
Classical terminal
Payments only
(m)POS
Decreases hardware size, improves compatibility and mobility
SCR with a COTS device
Connects a simple card reader to a smart device. Without PIN entry
SPOC
Allows for PIN entry on a smart device
CPOC
The full payment solution (Card and PIN entry) living on a smart devices

Figure 1-1 (m)POS innovations

The new innovations that allow for this feature are still in development and can be divided into different types of solution phases. The most well-known and fully developed POS solution is the classical POS terminal. This terminal relies on hardware-based security and has proven to be secure. Due to new innovations the wearable mPOS terminal made it possible to have a more lightweight and cheaper POS device. These wearable devices often have less features and a lower battery life. Although the outcome of this research can be applied to all POS systems, the scope of this research is on the innovations that follow after the wearable mPOS device.

Furthermore, there is the Software-based PIN entry on COTS (SPOC) and SCRP. This is an intermediate solution combining the SCRP (i.e. a dongle) for card data entry and a COTS device for processing and for PIN entry. The combination of software and hardware maintains security while decreasing the hardware costs. Furthermore, there is the Tap-on-Phone or also called CPOC or CPOC with PIN. These mPOS solutions do not use additional hardware and therefore provide an environment that enables customers to enter their PIN and card data in the same COTS device. These innovations and their chronological appearance are shown in image 1. Currently, as an intermediate step most solutions are implemented without PIN due to the complexity of providing security for PIN on a COTS device.

The presently developed mPOS solutions require additional on device software or hardware to allow for payments to be processed in a secure manner. Using on device software on the different COTS devices decreases flexibility and is less scalable. A COTS device is not under the control of the PSP's. For PSP's it is therefore beneficial to build a payment architecture in which they can control the majority of the software and are less dependent on the users.

Furthermore, the upcoming payment solutions introducing software development kits (SDK's), Libraries and Bluetooth will architecturally set payment service architecture back a couple of years. Modern payment service providers want to use cloud payment processing to remain flexible and increase scalability. Future payments will most likely be processed using COTS devices due to their simplicity and availability. As is indicated in Chapter 2, there is currently no literature on server-based COTS payment solutions that is compliant with security standards and therefore payment service providers are less likely to adopt this new innovation.

These various solutions bring the payment industry in a difficult position. PSP's want to offer these solutions but do not like to invest in building different payment architectures (Bruno et al., 2019). Merchants using these COTS solutions will have to continuously update their software, hardware and firmware. Furthermore, security authorities need to certify various protocols which makes it difficult to keep up with the innovations.

All these reasons address the need for a payment solution that lives outside of the various hardware solutions while still fulfilling the requirements of the various mPOS innovations. This research is conducted to provide software architects and design science researchers the necessary requirements and background information to simplify implementing the new mPOS solutions. In this way, they can provide a secure solution that fulfils their customers demand for cheaper and more attractive POS solutions. This will be done so by validating the possibility of a single payment architecture that will fit all current and future mPOS solutions. This will be validated by designing a software-based payment architecture aimed to fit both CPOC and SPOC solutions that is compliant with security standards and PSP requirements.

1.1.1 Societal significance and business opportunity

To stress the opportunity and the societal benefits of this research the business case will be discussed in this paragraph. Various upcoming trends enable the move from hardware devices to software (Shafique et al., 2020). The increase of internet speed and coverage and new technologies such as cloud servers and edge computing enable the move towards solely software-based products such as IoT, Netflix for streaming movies or Spotify for streaming processing power. The payment industry has long withheld itself from this movement due to technological limitations but with the rise of contactless payments and the improvements of the smartphone a similar disruption in the payment industry is no longer far away.

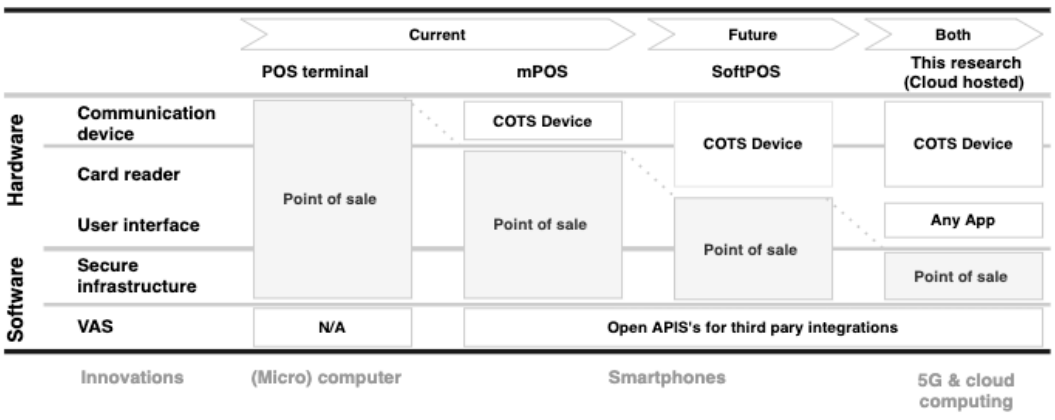


Figure 1-2 mPOS enabling trends

As shown in Figure 1-2 the transition towards software-based payments is in full speed enabling various new opportunities both for society and the industry. According to research done by Mastercard and Juniper reducing the barriers to terminal ownership would enable an estimated 15 million additional card accepting merchants globally in the next five years resulting in an acceptance of almost 500 billion dollars in transaction value. This increase of contactless payments will benefit both merchants and consumers by declining the need for cash payments.

Therefore, building architecture to simplify the move towards mPOS will benefit society in an economical manner but it will also improve the benefits of technology maintenance (Bulman, Garraghan 2020). Moving the software based mPOS application to a server will result in:

- *Simplified certification process* - All devices will be certified at once by certifying the server solution.
- *Availability on lower end devices* - The server will lower performance demands on COTS devices making mPOS available to lower end devices with low processing power
- *Minimize maintenance* - All changes to Kernel and security functions will be effective immediately on all connected mPOS devices
- *Increased data Security and richer transaction data* - Information is stored and synched with the server, following all the security guidelines and standards, assuring encryption, backup and protection at all times Furthermore, the cloud offers unlimited memory space and processing power to store richer transaction data
- *Better Distribution* - Terminal vendors can easily port the Kernel and security functions across hardware and operating systems. Furthermore, Single Cloud POS instances can support thousands of merchants
- *Unlocked Potential for value added services (VAS) and IoT* – Server-based POS could serve as a platform to distribute VAS instantly. Furthermore, Server based POS can integrate with multiple APIs to deliver IoT use cases.

1.2 Problem statement and knowledge gap

Literature and conversations with various payment system vendors pointed out that the increased complexity caused by the variety of mPOS innovations slows down the adoption of new mPOS innovation by the payment industry (Hartog, Moreno, 2019). The literature research provided in Chapter 3 shows that the existing literature can be considered insufficient on the subject of mPOS innovations.

Terminal payments have been researched as well as payments on a COTS device in the form of P2P NFC payment acceptance. However, this does not allow for card payments. Therefore, some research has been done on designing an NFC based payment card (Ghosh et al., 2017). Furthermore, to facilitate a solely software-based solution cloud-based payment architectures have been investigated as well (Pourghomi et al., 2013). However, all this research is aimed at providing solutions for single problems without considering the stakeholder requirements from an overall payment industry perspective. Furthermore, NFC based payments don't allow for the widely used payment cards. Changing the payment cards is out of the scope of payment service providers, cards are the responsibility of the issuer and not the PSP. Cloud based payments exist, but they are not yet combined with the current P2P and NFC based payment acceptance structures. Although much research has been done on payment transactions, few or no research can be found on mPOS payment architectures flows.

Likely, the innovative nature of this subject has surpassed the academic literature. Conducting qualitative research to provide literature on this subject will therefore be of great value for academic and economical purposes.

Providing research on the architectural decisions and tradeoffs necessary for new mPOS innovations will simplify and therefore increase mPOS adoption by PSP's. It will do so by assisting the payment industry in making complex 'buy or build' decisions and reduce costs on iterating payment innovations to fit new standards. Furthermore, by combining new but existing technologies for architectural purposes this design research will identify multiple research gaps that can provide more insight in the disruptive potential of a server-based POS system.

1.3 Main research question and sub-questions

To provide an overview of the architectural changes necessary to allow for the upcoming mPOS innovations a design should be made that fits all mPOS innovations. Designing such an architecture will allow the researcher to showcase the decision process and tradeoffs necessary. Therefore, designing such a flexible architecture will provide an architecture decision flow that can be used for PSP's with various use cases. This research will therefore answer the following research question:

"How can one single payment architecture provide a secure payment solution that can be used for the current as well as the future (mobile) Point of Sale (POS) solutions?"

In order to find an answer to this research question some aspects are identified. First of all, it is distinguished what is meant by a secure payment. Secondly, it is established what the current and the future point of sale solutions are, and which fall into the scope of this research. Lastly, the design research is be conducted. Due to limited resources it is conducted considering the various stakeholders in the Western European POS payment industry. This payment architecture facilitates and improves the acceptance and adoption of the new POS innovations by PSP's. It is therefore in line with the expectations of the merchants using it, the PSP experts building it and the authorities guaranteeing its security. To cater for the broad range of stakeholders an architecture decision flow derived from a flexible architecture is the output of this research. Furthermore, the new architecture decision flow should be technically and legally feasible.

Moreover, this research will partly consist of expert market interviews. Therefore, it is necessary to establish which market will be considered as a case study to answer this research question. Furthermore, conducting a case study will decrease the complexity of the research and therefore enable the research to be more focused on the actual design. Therefore, the scope of this research question is limited to Western Europe.

The research approach will be further explained in the coming chapter guided by the following sub- questions:

1. *What are the most significant current and future mobile POS solutions?*

Firstly, a thorough analysis will be performed on the various mPOS solutions. Only significant innovations will be considered. With significant is meant innovations that will have an impact on the payment industry because they have additional benefits to the current solutions and are technically viable. The answer to this research question is elaborated on in Chapter 3.

2. *What are the stakeholder's quality attributes for the most significant current and future mobile POS solutions?*

The answer to this question can be found in Chapter 4 and 5. The answer provides the security requirements from the payment acceptance authorities, the merchant and the PSP requirements. These are gathered by in depth literature research and qualitative expert interviews with various merchants, PSP's and a payment authority laboratory (Riscure). Eventually only the requirements that have an effect on architectural decisions will be considered. Therefore, most functional requirements are out of scope and the main focus will lie on quality attributes.

Most of the security requirements are known and can be taken from literature. It is important to identify which rules apply to this solution and if they are the same in each country in Western Europe. Therefore, a descriptive theoretical approach is necessary.

The literature review in Chapter 3 shows that no research has been done on the feasibility of a (cloud-based) mobile POS payment solution for COTS devices. Information on this subject can be gathered by conducting qualitative research by interviewing experts as well as the users of the products. Several Experts (i.e., developers and PCI laboratories) were interviewed to structure the requirements and feasibility of a cloud based mobile POS Payment solution.

3. *What payment architecture fits the requirements derived from sub question one and two?*

The requirements will form the basis to design the payment architecture decision flow. The design will be iterated upon after technical and legal evaluation by a feedback panel. This sub-question will be answered by using a design approach. This will be further elaborated on in the following chapter on research methods and design.

1.4 The scope of this research

Because of resource limitations this research will be limited to Western-Europe. The data gathered and the conclusions drawn from this data will only be considered applicable to the Western-European payments industry. The entire Western-European POS payment industry will be considered. Therefore, due to the variety of PSP's it is not possible to contemplate functional requirements. Consequently, a holistic architecture is considered using quality criteria and quality criteria refinements. In addition to this, only scenarios and quality criteria that have an effect on architectural performance will be regarded.

Furthermore, the to be designed architecture in this research only considers software architecture. Although hardware components are mentioned in this research, no research will be done on making changes in hardware architecture. In the process of moving from hardware to software-based payment solutions considering new hardware is irrelevant to new innovations.

Moreover, the mPOS solutions that are considered in this research should be certifiable by the PCI Security Standards Council or likely to be certified in the future. mPOS solutions are considered likely to be certified by PCI in the future when one of the global schemes, Mastercard or Visa, has publicized security requirements for those solutions. This assumption has been made after consultation with representatives from Visa, Mastercard and PCI. Additionally, only the two major COTS device operating systems, Android and IOS will be considered in this thesis. According to the most recent (2021) number from statistics platform Statistica, these OS have a combined market cab of 99.99% of the global market ("Statista", 2021).

Finally, as a consequence of resource limitations, the implementation of a software architecture is not within the scope of this research. However, implementation of this research design will be regarded as a follow up to this research.

1.5 Deliverables

This research will provide multiple deliverables which are discussed throughout the research paper and summarized in the conclusion. In Figure 1-3 an overview of the methodology and output of this thesis report is given.

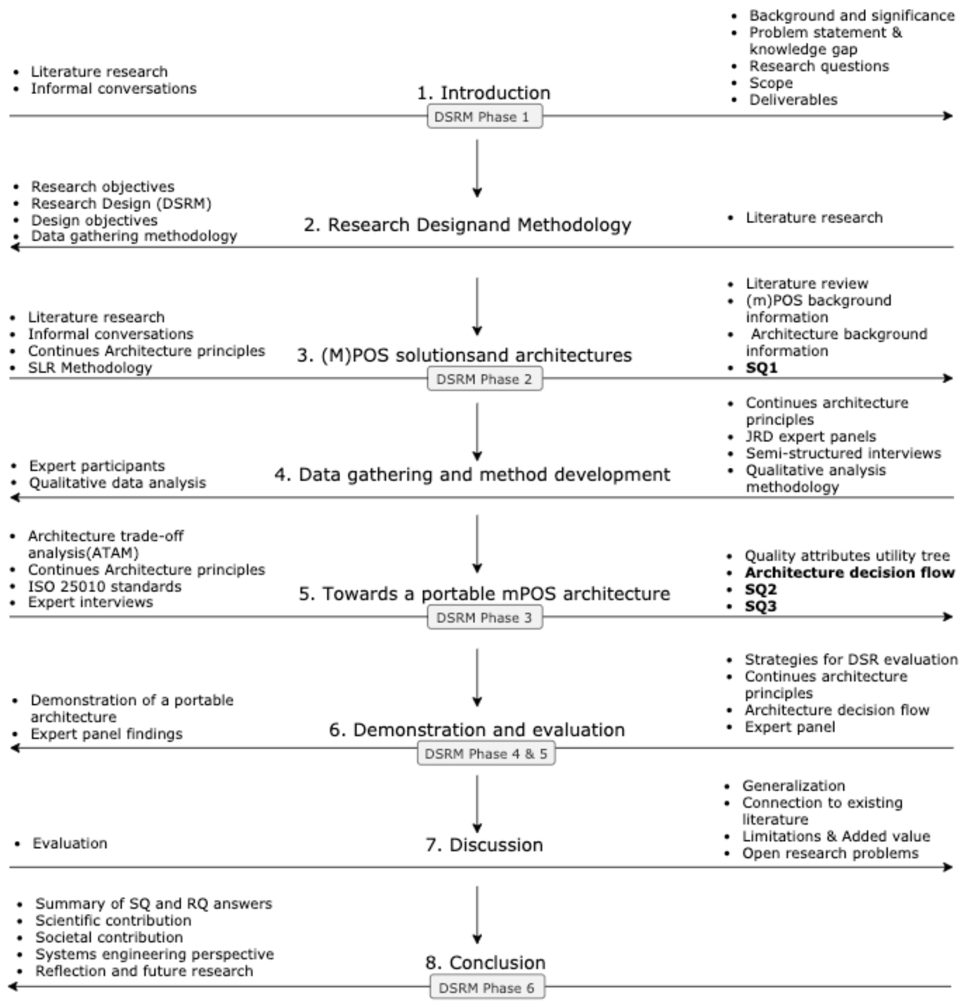

• Literature research
• Informal conversations
1. Introduction
DSRM Phase 1
• Background and significance
• Problem statement & knowledge gap
• Research questions
• Scope
• Deliverables
• Research objectives
• Research Design (DSRM)
• Design objectives
• Data gathering methodology
2. Research Designand Methodology
• Literature research
• Literature research
• Informal conversations
• Continues Architecture principles
• SLR Methodology
3. (M)POS solutionsand architectures
DSRM Phase 2
• Literature review
• (m)POS background information
• Architecture background information
• SQ1
• Expert participants
• Qualitative data analysis
4. Data gathering and method development
• Continues architecture principles
• JRD expert panels
• Semi-structured interviews
• Qualitative analysis methodology
• Architecture trade-off analysis(ATAM)
• Continues Architecture principles
• ISO 25010 standards
• Expert interviews
5. Towards a portable mPOS architecture
DSRM Phase 3
• Quality attributes utility tree
• Architecture decision flow
• SQ2
• SQ3
• Demonstration of a portable architecture
• Expert panel findings
6. Demonstration and evaluation
DSRM Phase 4 & 5
• Strategies for DSR evaluation
• Continues architecture principles
• Architecture decision flow
• Expert panel
• Evaluation
7. Discussion
• Generalization
• Connection to existing literature
• Limitations & Added value
• Open research problems
• Summary of SQ and RQ answers
• Scientific contribution
• Societal contribution
• Systems engineering perspective
• Reflection and future research
8. Conclusion
DSRM Phase 6

Figure 1-3 Thesis flow diagram

The main deliverable consists of a portable architecture decision flow. The deliverable will show which options can be considered in building an architecture and proposes newly developed architecture options that allow for the architecture to be cloud-based and fit all mPOS innovations. This enables payment service providers (PSP) to build an architecture fitting their quality scenarios while maintaining visibility on the changes necessary for future mPOS solutions. It also allows new entrants to design an architecture that allows for all mPOS innovations without have to do expensive iterations upon the architecture.

In the process of this design a quality attributes utility tree will be designed that is specifically made for payment architecture analysis. This provides a better design tool for architects to analyze their artifacts.

1.6 Summary of the introduction

The payment industry suffers from a slow adoption of the various mPOS innovations. A variety of solutions results in an unclear and unstructured environment in which it is difficult to determine how to develop your architecture to be ready for the current as well as the future mPOS solutions. For both academical and societal purposes it is of importance to provide structure and overview in the architectural design decisions possible and necessary for new developments. The objective of this research is to design a decision flow that provides an overview of the architectural changes possible for both the current as well as the future mPOS solutions. This will be done so by answering the question *"How can one single payment architecture provide a secure payment solution that can be used for the current as well as the future (mobile) Point of Sale (POS) solutions?"*

2 Research Design and Methodology

2.1 Research objectives

The overall objective of this research is to provide PSP's the necessary requirements and background information to simplify implementing the new mPOS solutions such that they have an easy way of providing a secure solution that fulfils their customers demand for cheaper and more attractive POS solutions. Exploration of the mPOS solutions and the architectural changes necessary will contribute to the body of knowledge by technology adoption of disrupting technologies. Society will benefit from this research by the improvements it will bring to mPOS technology adoption by simplifying mPOS adoption through designing a flexible payment architecture.

2.2 Research design, a DSRM process model

This research will use the Design Science Research methodology (DSRM) process model (Peffers et al., 2007). DSRM is specifically developed for information systems. It involves the structure, principles, practices and procedures necessary to carry out design research on information systems such as software architectures.

As shown in image 3, the DSRM model requires six steps: the motivation and identification of the problem, the objectives of a solution, the design and development phase, demonstration of the design, evaluation of the design and lastly, communication. These steps are synchronous but also enable and require process iteration and feedback loops. This research will follow these steps to find the solution to the main research question.

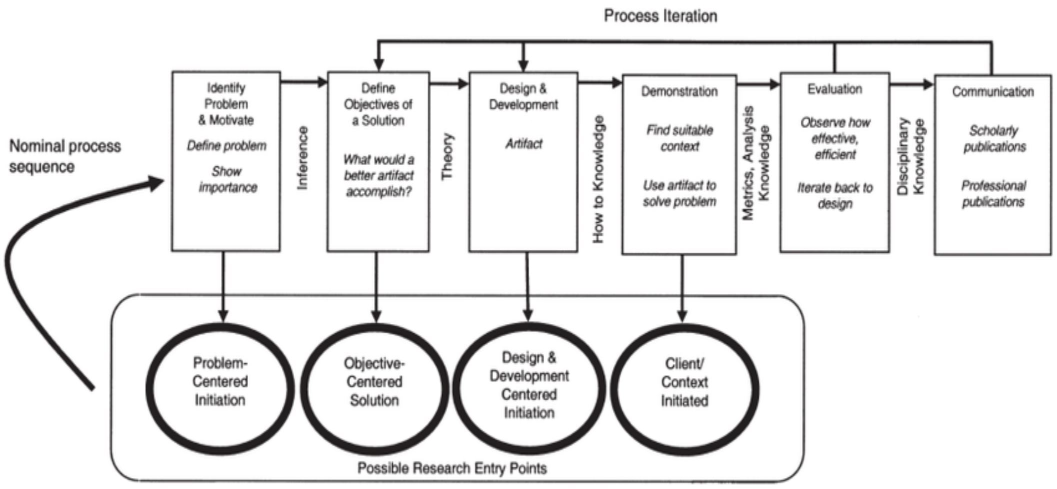

Process Iteration
Nominal process sequence
Identify Problem & Motivate
Define problem
Show importance
Inference
Define Objectives of a Solution
What would a better artifact accomplish?
Theory
Design & Development
Artifact
How to Knowledge
Demonstration
Find suitable context
Use artifact to solve problem
Metrics, Analysis Knowledge
Evaluation
Observe how effective, efficient
Iterate back to design
Disciplinary Knowledge
Communication
Scholarly publications
Professional publications
Problem-Centered Initiation
Objective-Centered Solution
Design & Development Centered Initiation
Client/ Context Initiated
Possible Research Entry Points

Figure 2-1 DSRM process model (Peffers et al., 2007)

The phase-wise approach suits the complex character of the objective of this research and will assist in simplifying the research. This methodology is chosen above other research methodologies because it provides the tools for designing a software architecture. Furthermore, the methodology is developed for research in information systems which results in a good fit for this research, because it is aimed at designing a complex information system. Moreover, the iterative nature of this methodology allows for continues feedback which is necessary to come to an optimal solution.

2.3 Research phases and research flow

The research flow is indicated in image 4. Further explanation on image 4 and the various phases derived from the DSRM process model will be given in this chapter.

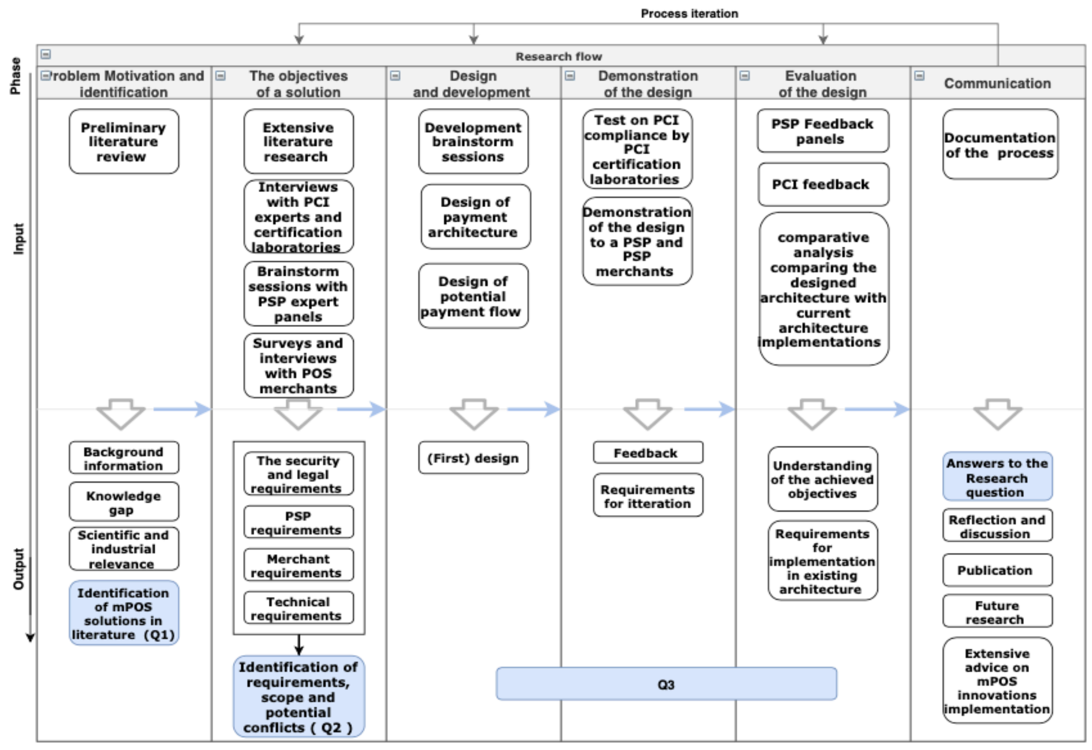

Process iteration
Research flow
Phase
Input
Output
Problem Motivation and identification
The objectives of a solution
Design and development
Demonstration of the design
Evaluation of the design
Communication
Preliminary literature review
Extensive literature research
Interviews with PCI experts and certification laboratories
Brainstorm sessions with PSP expert panels
Surveys and interviews with POS merchants
Development brainstorm sessions
Design of payment architecture
Design of potential payment flow
Test on PCI compliance by PCI certification laboratories
Demonstration of the design to a PSP and PSP merchants
PSP Feedback panels
PCI feedback
comparative analysis comparing the designed architecture with current architecture implementations
Documentation of the process
Background information
Knowledge gap
Scientific and industrial relevance
Identification of mPOS solutions in literature (Q1)
The security and legal requirements
PSP requirements
Merchant requirements
Technical requirements
Identification of requirements, scope and potential conflicts (Q2)
(First) design
Feedback
Requirements for itteration
Understanding of the achieved objectives
Requirements for implementation in existing architecture
Answers to the Research question
Reflection and discussion
Publication
Future research
Extensive advice on mPOS innovations implementation
Q3

Figure 2-2 Research Flow Diagram

Phase 1: Motivation and identification of the problem

In this phase the research problem and the justification of the value of a potential solution will have to be found (Peffers et al., 2007). The first stage of this research has already been conducted partly and will identify the problem using literature research. Furthermore, the literature research will be used to identify a research gap and therefore motivate the importance of the solution presented by this research paper. Furthermore, in this first phase the first sub question will be answered:

SQ1 – "What are the most significant current and future mobile POS solutions?"

SQ1 will be answered by preliminary literature research on the current mPOS solutions. The Literature on these mPOS solutions will most likely indicate the complexity of the problem and expose the need for a flexible architecture that enables integration of the various mPOS solutions. An important aspect of this phase is the identification of the architectural requirements for the various mPOS solutions.

Phase 2: The objectives of a solution

In the second phase 'Define the objectives of a solution' the problem identified in phase 1 can be transformed into system objectives or meta-requirements. The problem does not necessarily identify all objectives. For that reason, there remains a step of determining the performance objectives of the, to be designed, artifact (Peffers et al., 2007). These objectives will mainly be qualitative to describe how the new architecture should form a solution to the problem addressed in phase 1. Knowledge on the current solutions as well as on the requirements for the, to be developed, solution should be derived in this phase. Therefore, the second research question will be answered in this phase.

SQ2 – "*What are the stakeholder quality attributes for the most significant current and future mobile POS solutions?*"

The objective of the solution will be further scoped in the answer of SQ2. The quality criteria and therefore the boundaries of the design will be identified. To answer this question the requirements can be divided in three aspects. Data has to be gathered from the three main parties, the PSP experts the merchants and the security laboratories safeguarding PCI requirements.

Data should be found in literature on the current and future requirements regarding accepting payments in general and for mPOS solutions (PCI, 2019). Furthermore, data on this subject will be gotten through interviews with the experts from Riscure, a company that can provide expertise on payment security requirements.

Furthermore, expert interviews will be conducted with PSP Adyen and current mPOS vendors to get more insight in the PSP requirements.
Lastly, expert interviews with western European merchants will be conducted to establish the merchant requirements. The merchant requirements will partly indicate the market demand and therefore this sub phase will also contribute to phase 1, identification of the problem and will therefore have to be conducted in an early stage.

By comparing the outcomes of these expert interviews, the actual objectives for a solution can be determined (Peffers et al., 2007). It has to be considered that requirements and objectives can be conflicting between stakeholders. Therefore, this is an iterative process in which objectives are gathered and adjusted. It is therefore important in this phase to distinguish between hard requirements and soft requirements. In which the hard requirements are considered to be obligations for the stakeholder and the soft requirement a 'nice-to-have' requirement that can be interpreted more flexibly. The data and experts necessary for this phase will be provided by PSP's as well as merchants that will use the mobile POS solution. Further information on these interviews can be found in the appendix.

Phase 3: The design and development phase

The third stage 'Design and development' will demonstrate the actual creation of the artifact or in this research, the architecture. This phase will demonstrate the outcome of the second phase by combining the gathered requirements and objectives with the information obtained from qualitative interviews with development experts, feedback sessions and tests with identified certification offices. Although the feedback and testing are part of the following phases these phases will be considered due to the iterative character of this methodology. A payment architecture will be designed in this stage that will attempt to combine the software solutions and the stakeholder requirements for a flexible one-size-fits-all mPOS payment solution. Phase three will roughly consists of four sub-phases.

- *Scoping the design* - The objectives acquired in phase 2 will help identify the boundaries of the solution that is requested. In phase 3 various objectives will likely turn out to be conflicting and will have to be adjusted or removed.

- *Brainstorming and development of potential solutions* - In this sub-phase a design will be made using the requirements and the current mPOS solution architectures. This will be verified during brainstorm sessions with multidisciplinary development experts. It is not clear yet whether only one architecture will fit the objectives acquired in phase 1. Therefore, it is possible that multiple potential architectures will be designed conceptually.

- *Identification of the optimal solution* - When multiple architectures fit the requirements an optimal solution should be chosen using a performance matrix such as the best in class or (weighted) priority checkmark method (Lucero, 2016).

- *Iteration* - The following phases will provide feedback on the designed solution which will result in an iterative process in which the design will be improved.

Within this phase, after iterations the SQ3 will be answered.

> *SQ3 – "What payment architecture fits the requirements derived from sub-question one and two?"*

As explained earlier the requirements will also involve the technical and legal feasibility of the design. Furthermore, SQ3 will not directly be answered solely out of phase 3 due to the feedback loops that are necessary and that will evolve out of the following phases.

Phase 4: Demonstration of the design

In phase 4 a demonstration of the design should provide proof of the capabilities of the design to solve the problem. (Peffers et al., 2007). In this stage the presented architecture will be simulated in a payment flow diagram and verified by PSP experts at PSP Adyen. Furthermore, the documentation of the architecture will be demonstrated to a PCI laboratory to test the design on the security requirements. This feedback will be used for evaluation in the fifth phase.

Phase 5: Evaluation of the design

In phase 5 it will be observed and measured how well the architecture actually fits the solution to the problem (Peffers et al., 2007). Therefore, there will be evaluated whether the architecture is flexible enough to incorporate all identified mPOS solutions without significant iterations. This will involve comparing the defined objectives with the actual observed result from the demonstration of the architecture. This phase will therefore first require the identification of the relevant metrics necessary for evaluation and comparison. At the end of this phase a decision has to be made on whether to iterate back to step three to improve the architecture or to continue to phase 6.

Phase 6: Communication

Finally, the ‘Communication” phase will result in publications for the scientific community and the payment service providers and its users. The problem and the importance or utility of its solution design will be communicated. The communication should clearly state the added value to the stakeholders of the problem as well as the academic relevance of it.

2.4 Design objectives

This research is conducted to provide PSP’s the necessary requirements and background information to simplify implementing the new mPOS solutions such that they have an easy way of providing a secure solution that fulfils their customers’ demands.

It is not possible to fulfill all requirements of different payment service providers. Furthermore, not all payment service providers will make use of the entire architecture. Competition between vendors offering only parts of the architecture will likely occur. Therefore, not all PSP’s require the full mPOS architecture and various PSP’s have different use-cases for their mPOS solution. Therefore, this research will not focus solely on developing a flexible mPOS architecture but rather on developing the process towards it. The objective of this research is to design a decision flow that provides an overview of the architectural changes possible for both the current as well as the future mPOS solutions. This decision flow should:

- Enable architects to identify a one-size-fits all architecture that is flexible enough to provide for the current as well as the future (m)POS solutions
- Comply with security and PCI standards
- Be technically feasibly
- Support researchers and companies in making architectural decisions.
- Help identify the effect that changing the architecture will have on the stakeholder quality criteria.
- Identify what scenarios need additional research to monetize the effect of a flexible architecture.

2.5 Data gathering methodology

Phase 1 to 5 require gathering of data. Various types of data need to be gathered and therefore this sub-chapter will explain which elaborate on the methodologies that have been used to receive the required information. First, this chapter will elaborate on the use of qualitative date. Then, it will explain about the continues methodology principles used in this research. Thirdly, it will elaborate on the qualitative methodology that has been used and lastly the data analysis will be briefly explained.

The methodology explained in this chapter will be used as a guideline of collecting, analyzing and interpreting architecturally significant data in order to understand and give answers to the sub-research questions. The three most common approaches to conducting research are quantitative, qualitative and mixed methods (Marvasti, 2018). An assessment has to be made on what type of data (i.e., numerical or textual) is necessary, to conduct this research. Typically, the qualitative approach is used for research questions requiring textual data such as the main research question in this research (Marvasti, 2018).

Therefore, to gather information or data that can serve as the theoretical foundation of the design a qualitative approach will be used. Figure 2-3 shows the research methods used and the order in which they were applied. Information will be gathered using literature research, semi-structured interviews, unstructured interviews, brainstorm sessions and expert panels. Qualitative research methods are chosen to accommodate for the first five phases of this research. Literature research and qualitative interviews will form the theoretical foundation and provide data for the input of the design phase. Furthermore, brainstorm sessions provide guidance during the design phase and expert panels in the form of feedback sessions will accommodate for the final phases 4 and 5.

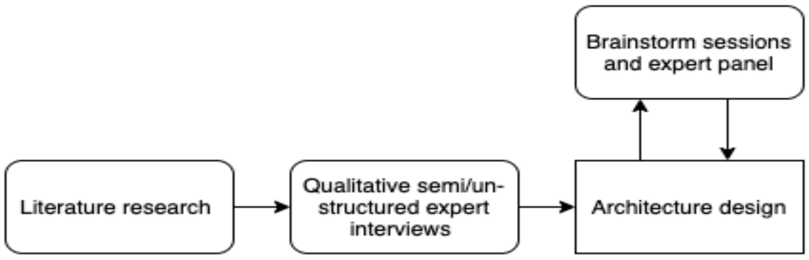

Brainstorm sessions and expert panel
Literature research
Qualitative semi/un-structured expert interviews
Architecture design

Figure 2-3 research phases

2.5.1 Continues architecture methodology

The design of this software architecture was conducted while following the Continues Architecture principles provided by the research of Murat Erder and Pierre Pureur (Erder & Pureur, 2016). The continuous architecture principles are specifically suitable for building this software architecture because it allows for flexibility and it allows for rapid delivery. This methodology is therefore more suitable in comparison with more traditional delivery models because of the rapid advances in software engineering and the increase in new innovations in the payment industry. The Continues architecture methodology follows these simple principles:

The architecture should be built for products and not project solutions - Projects are considered temporary and are meant to deliver a well-defined result. An architecture can best be built more strategically in nature and therefore needs a longer-term product-level approach.

Focus on quality attributes and not on functional requirements - Quality attribute requirements have a more significant impact on a software architecture. This is mainly because the architecture of a system determines how well nonfunctional requirements are implemented by the system. This principle will be further elaborated on in this chapter.

Delay design decisions until they are absolutely necessary – This principle is to be followed to withhold the architect from making design decisions before requirements are known. Functional requirements often change frequently and therefore it is best to first focus on quality requirements.

Architect for change – The third principle of continuous architecture implies that requirements might change in the process of building architecture. Change is unavoidable and therefore the challenge is to achieve consistency while evolving at the same time.

Architect for build test and deploy – The architect should be built for continues delivery. The focus for this research is not on implementation of the architecture. Therefore, this principle should be considered but will not yet fall into the scope of this research.

During the process of gathering requirements and designing the architecture, these principles will be considered.

2.5.2 Methodology for gathering quality criteria

For gathering the quality criteria, the second principle of the Continues architecture methodology: "*Focus on quality attributes and not on functional requirements*" (Erder & Pureur, 2016) will be elaborated on.

Any IT system requirement can be classified in two categories: functional requirements and non-functional requirements. Functional requirements describe the actual functionalities that the system must provide. These have been stated by the merchants and experts and apply to the eventual mPOS terminal. Non-functional requirements describe the so-called quality attributes. These are the attributes that must be met by the system in order to deliver the previously mentioned functional requirements. These attributes also consist of constraints which are design decisions with no degrees of freedom.

Focusing on these quality attributes might lead to an abstraction of the most important requirements or also called functional requirements. According to the architecture research by Philippe Kruchten it would be best in an architecture design stage to "focus on a small subset of important scenarios - instances of use cases to show that the elements of the four views work together seamlessly" (Kruchten, 1995). Quality attribute requirements have a more significant impact on a software architecture. This is mainly because the architecture of a system determines how well non-functional requirements are implemented by the system.

To achieve a global quality attribute approach rather than a functional approach it is important to define the architecture in terms of addressing problems to be solved. This is greatly summarized by architect Kurt Bittner in his forester research:

"*An outcome is either satisfied or it is not; outcomes are the appropriate units of scope for a release, not requirements. A requirement that does not contribute to satisfying an in-scope outcome is superfluous. When requirements become disconnected from outcomes, scoping discussions become complex and contentious, subject to mere whim and opinion. The result is usually wasted time, effort and money (Brittner, 2014).*"

In this research architectural and design decisions will be showcased that allow for the implementation of those quality attributes. Those decisions are often compromising since deciding for one quality attribute can often result in a negative impact on the implementation of another quality attribute. The effects on these quality attributes will be shown in the next chapters.

In general, functional requirements define the work that the system must do but do not define how it does it. Functional requirements have quality attributes associated with them and it is therefore possible to derive functional requirements from non-functional requirements and vice versa. When only functional requirements are being used it is possible to end up with a large number of candidate architectures. Designing for quality attribute requirements will enable the architect to limit candidate architectures. This will allow this research to showcase certain architectural design choices within the scope of the quality criteria. An eventual design decisional can be made dependent on the stakeholders more functional requirements.

Designing a software architecture requires a flexible approach in gathering requirements due to the evolving nature of architectural requirements. The Design science research methodology allows for such an approach due to its iterative nature (Peffers et al., 2007). Other requirements gathering approaches such as the waterfall approach determine and document every requirement precisely at the beginning of the process. This leads to bias and potentially narrows down the potential of a design in an early stage (McCormick, (2012)).

2.5.3 Literature research

Literature will serve as the theoretical foundation for the development of the payment architecture. Furthermore, it will help with creating an understanding of the current solutions and the academic research that has already been conducted on those solutions. Further explanation on the literature study will be given in Chapter 3.

2.5.4 Interviews with industry experts

A significant part of the data used in this research will be collected by the use of expert interviews. The complexity of the problem requires a less directed approach and a more qualitative interviewing strategy. Therefore, the data collection in this research will mainly focus on unstructured and semi-structured formats because more structured interviews often produce quantitative data (DiCicco-Bloom & Crabtree, 2006). How to conduct the interview is also dependent on the type of stakeholder that will be interviewed and the data that is expected. Due to the more simplistic data necessary from merchant experts on their position on mPOS, a more structured approach can be used. The outcome from the expert interviews with developers on the actual architecture is more abstract and therefore a less structured approach will be used in these interviews.

2.5.5 Brainstorm sessions and expert panels

The brainstorm sessions and expert panels will help in the designing phase as well as the demonstration and evaluation phase. The brainstorm sessions will be conducted in an unstructured manner to cater for a broad scope and input of ideas for development. The expert panels will provide feedback with a more structured approach in order to structurally test the design on lacking features.

2.5.6 Qualitative data analysis

A software architecture has to deliver the functionality that is needed by its stakeholders while satisfying both the safety and quality concerns. What challenges to consider is dependent on the type of structure that has to be dealt with (Chen, et al,. 2014; Chung, 2000; Glinz, 2008). While many challenges can be considered in later stages of software development, the core concerns that can be impacted by the architecture should be identified before building an architecture. These concerns or criteria are also called 'architecturally significant' (Erder & Pureur, 2016). Only actually architecturally significant data will be considered.

The objective of data analysis is to derive conclusions from the data gathered with the previously mentioned research methods and to keep a clearly identified chain of evidence (Runeson, & Höst, 2008). The data that is gathered through expert surveys has multiple purposes:

- *Scoping:* The data from both the interviews as well as the literature review should provide the restrictions and boundaries in which the design can be developed. The design should only facilitate for the determined mPOS solutions and should only enable the solutions to fulfil the full length of the retrieved requirements.
- *Requirements:* After analysis of the interview data the requirements and restrictions should become clear. The data analysis should eventually provide a list with all merchant, PSP and payment authority requirements.
- *Substantiation of the design:* Designing a payment architecture requires multidisciplinary development expertise. The expertise of an interviewee should therefore be documented together with the interview data in order to provide insight in the design decisions that were made and why decisions were

made. The feasibility of the design is dependent on the data obtained from experts. This data is therefore part of the substantiation of the feasibility of the designed architecture.

- *Testing and validation:* After designing the design should fit the actual requirements gotten from the data. Furthermore, expert panels should provide information on the feasibility of the design.

2.6 Summary of the research design and methodology

This research aims to provide PSP's the necessary requirements and background information to simplify implementing the new mPOS solutions. This should allow the payment industry to provide a secure solution that fulfils their customers demand for cheaper and more attractive POS solutions.

Therefore, the main objective of this research is to design a decision flow that provides an overview of the architectural changes possible for both the current as well as the future mPOS solutions. This will be derived from the design of a software payment architecture that fits all current and future mPOS solutions.

This will be done by conducting a design research using the DSRM process model. This model fits the objective of designing an information system infrastructure. The design needs to fit requirements from various stakeholders with potentially conflicting objectives. These requirements will be gathered following the continues architecture principles and by using various qualitative research methodology.

3 (m)POS solutions and architectures

This chapter will provide the general background information necessary for this research. Firstly, a literature review is described that is conducted in an early stage to determine the research gap and to provide argumentation for the lack of literature on mPOS architectures.

Secondly, the first sub-question "*What are the current and future mPOS solutions?*" Will be answered by conducting a literature review and informal interviews on the current and future mPOS solutions and their architectures. This information is necessary to gain insight in the solutions that should be driven by the designed architecture. This information is provided in text and illustration in the middle part of this chapter.

Lastly, the technologies that will influence architecture design decisions according to the experts will be further explained in the final part of this chapter. This is explained to allow the user of this research to easily interpret the results. Additional background information can be found in the Appendix and in the provided sources.

3.1 Literature review

This literature review has been conducted to identify academic research on the subject of server-based payment architectures to establish the research gap and gain additional information on the subject. The literature review process will be briefly described in this sub-chapter.

3.1.1 Literature review methodology

To do a review on the existing literature a systematic literature review (SLR) process described by Kitchenham, (2014) has been utilized as a tool of guidance. This section will elaborate on this method and on the selection-criteria used to get the correct literature. The literature has been conducted solely using computer research. The goal of using this method is to be able to evaluate and interpret all available and useful research for this subject. (Kitchenham, 2004).

The following strings containing keywords were used for this literature research:

- Smartphone OR Mobile And "cloud based" AND Payment AND "Point of sale"
- "Cloud based" AND "transaction processing systems" AND Payment
- Tap to phone AND transaction OR Payment
- "Cloud applications" AND payment
- Transaction OR Payment AND Smartphone OR Mobile OR COTS

According to the SLR method multiple electronic sources have to be used (Kitchenham, 2014). The following databases have been used for this literature review:

- Scopus
- Web of Science
- Science Direct
- Google Scholar

The literature that has been selected for reviewing had to comply with some selection criteria. These criteria were established to determine whether the literature is relevant or not. Firstly, the literature should be in English. Secondly the literature should consist of articles or reviews. Thirdly the literature should be considered relevant for the payment industry. Furthermore, literature will only be considered when the innovations described have a significant impact on the payment industry because they have additional benefits to the current solutions and are technically viable. In order to identify more relevant literature other studies that were found in citations via the use of so called “Snowballing” have been considered. Table 3-1 elaborates on the literature that was found during this literature review. It indicates the methodology used for the research and shows which technology was subject of the paper. This should enable the reader to directly see that no literature could be found that identifies an architecture design flow that allows for multiple mPOS solutions.

Table 3-1 Literature review

Citation	*Methodology*	*Technology*	*Main objective*
(Liao et al., 2018)	Problem analysis and design	Cloud verification	Explores and improves a mobile payment protocol with outsourced verification in the cloud server.
(Ruiz-Martínez, 2015)	Framework design and analysis	Web-based payments	By developing a web payment framework based on a layered approach.
(de Reuver & Ondrus, 2017)	Framework design	Secure element, cloud or handset	The relocation of the secure element from SIM card to cloud.
(Rieger & Majchrzak, 2019)	Comparative analysis	Cross platform apps	Cross-platform development has seen much progress but the challenges are ever growing.
(Guille, 2018)	Design	Cloud-based transactions	Sending a token instead of a real account identifier to enable cloud-based transactions
(Al-Sayed et al., 2020)	Comparative analysis	Cloud services	Cloud computing and its non-functional and functional features. Divided in layers.
(Hartog and Moreno, 2019)	Explorative analysis	Tap-to-phone or CPOC	Exploration of the rise of smartphones as mPOS terminals.
(Hartog and Moreno, 2018)	Explorative analysis	Cloud based payment	Analysis of the security issues derived from cloud-based payments.
(Bruno et al., 2019)	Explorative analysis	Cloud based payment	An analysis on the rise of payment as a service through cloud-based payments.
(Bojjagani & Sastry, 2019)	Problem analysis and design	NFC-based payment	The design of an NFC-based payment protocol
(Kumar Royyuru et al., 2017)	Descriptive analysis	Peer-to-Peer payments	Describes patented systems and methods that can facilitate payments via a peer-to-peer protocol.
(Ghosh et al., 2017)	Design	Digital NFC payment card	A digital card module that uses NFC and biometric authentication for peer-to-peer payment and identity.
(Pourghomi et al., 2013)	Model design	Cloud, wallet, NFC	Development of an NFC payment application that uses cloud wallet models.
(Levialdi Ghiron et al., 2009	Design case study	NFC ticketing on COTS	NFC ticketing in which tickets can be bought for public transportation using COTS devices.
(Turk & Cosar, 2016).	Framework design	Containerization, NFC	Containerization of NFC application for payment security improvements.

3.1.2 Literature on payment solutions

The transition from accepting payments on a closed POS system towards accepting payments on a commercial off-the-shelf (COTS) device involves many different architectural requirements. Various scenarios are possible in which different type of hardware, and software solutions are necessary to fulfil security requirements (Shakeel Ahmad et al., 2016), (Turk and Cosar, 2016). Investing in future proof payment architecture can be complex due to the fast-changing security requirements for the different upcoming COTS payment solutions (Hartog & Moreno, 2018). For payment service providers it is therefore beneficial to build a payment architecture that is able to fulfil all current security requirements and adapt fast to the requirements requested in the future.

Various different solutions can be found on how to best design a payment architecture that allows for payment on COTS devices. The paper written by Bojjagani and Sastry in 2019 proposes an NFC based payment solution allowing customers to pay at check out by tapping their phone with a payment wallet installed on it on a phone receiving the payment (Bojjagani & Sastry, 2019). They describe this as an NFC-enabled payment method that can be used for peer-to-peer (P2P) payments and payer-to-merchant (P2M) payments.

A different, more cloud based, NFC enabled approach is described by Pourghomi et al. These researchers developed an NFC payment application that uses a cloud wallet model to support the entire transaction. The login credentials are gotten from the cloud in a secure way and are being used to make the payment in the COTS payment application (Pourghomi et al., 2013). The state diagram representation of Pourghomi et al. scheme is shown in image 3-1. The six steps indicate the flow in which a cloud provider can conduct the payment in its servers.

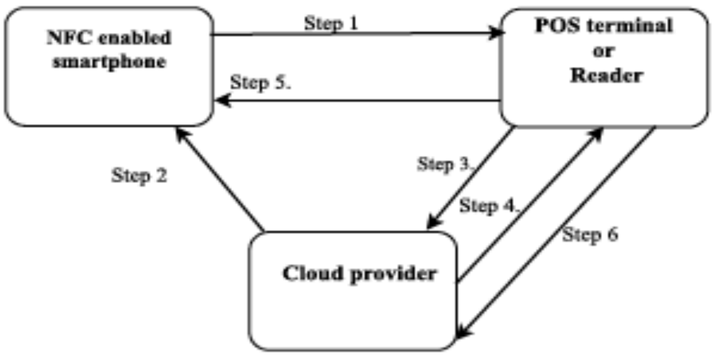


Figure 3-1 NFC application: Pourghomi et al.

Much research has been done on similar approaches using NFC and wallet payment acceptance. Ghiron et al. proposed NFC ticketing in which tickets can be bought for public transportation using COTS devices (Levialdi Ghiron et al., 2009). Ahamed et al. Proposed a secure NFC framework based on biometrics and a wireless public security key framework and Turk et al. containerized an NFC application to improve payment security (Shakeel Ahmad et al., 2016), (Turk and Cosar, 2016).

The former mentioned NFC based solutions can all directly be implemented in a secure manner. The downside of this solution is that it does not allow for card-based payments. The research done by Ghosh et al., in 2017 has a solution for the former mentioned problem regarding not being able to pay by card on a COTS device. They suggest not only changing the COTS software but also improving the payment card in order to fulfil security requirements. A digital card module that uses NFC and biometric authentication should provide for peer-to-peer payments or secure tap-to-phone solutions (Ghosh et al., 2017).

The various proposed solutions imply that there is academic research on improving the payment flow and even on cloud-based solutions. However, no research could be identified that considered the new software-based technologies.

3.2 Research gap

Although there is no academic research on the subject of conducting cloud-based payments on a COTS device the cloud innovation itself is not new and academic research has been done on this subject (Guille, 2018). Companies are developing faster server-based systems and are widely innovating on this subject (Liao et al., 2018) (de Reuver and Ondrus, 2017) (Guille, 2018). However, there is almost no literature on transaction acceptance innovations. Many of the cloud-based solutions already developed for other segments could be implemented in the payment sector but are not yet considered secure enough. Accepting payments on a server instead of on the device itself would create a more flexible architecture that allows for various devices to work on the same architecture. Determining the necessities for a solely cloud based CPOC payment system would open the way to more research on secure cloud-based payments on commercial off the shelf devices (Pourghomi et al., 2013), (PCI, 2019). More academic research on mPOS solutions could provide guidance and simplicity for the direction in which the payment industry is moving.

3.3 Classical mobile POS solutions

The well-known terminals that consumers use in stores are traditional hardware-based PIN entry models. These have evolved into a more hybrid form of hardware- and software-based models. The hardware-based model is the most used model well known from the counter top at most retailers (Raqib, Rizwan, 2020). In such a hardware-based security model the payment terminal hardware together with its firmware are primarily responsible for protecting the PIN entry. The most traditional devices have all protections within the device itself and are considered very secure (Hartog, 2018). Monitoring or detection mechanisms living in the terminal are responsible for identifying security controls that are not in an approved state to operate within the terminal. The hardware security model can then reply by disabling the terminal from processing the transaction and by clearing all sensitive data. Furthermore, all protection is being handled by the terminal itself.

3.3.1 From hardware to software

The security of the data entered was guaranteed by the hardware in the traditional payment terminal. This has now been evolving into a more software-based PIN entry security model. The move from hardware to software has resulted in a relocation of terminal components from the hardware towards a software device and a server. This process is visualized in Figure 3-2. In Figure 3-2 it is clearly shown that the classical terminal can be divided into software components that partly fit in terminal software and in a back-end server. The use of the various components will be explained in paragraph 3.3.3. In the software-based solution, the security part in the device where the PIN is entered is an important component. The component should support attestation to validate whether the security mechanisms are not tampered with. It should also support detection, to notify abnormalities. Furthermore, it should support response to control and act for an alert. A fully online solution will enable the possibility to extend these functionalities (PCI,2019).

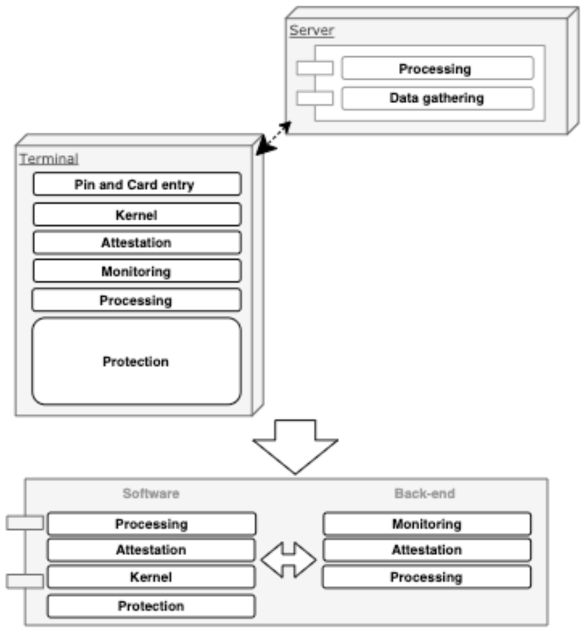


Figure 3-2 from software to hardware

However, over some components is limited control. The OS and the COTS device itself should be regarded compromised and untrusted. The assumption is therefore made that an attacker can get full access to the hardware device. For that reason, it is important that the software provides enough protection to complicate tampering and reverse engineering of the code (PCI, 2019).

3.4 (M)POS innovations

In 2018, the PCI Security Standards Council published the specifications for PIN entry on commercial off- the-shelf devices (PIN on COTS). Instead of using a dedicated terminal at the point of sale, merchants are using smartphones or tablets to accept in-store card payments.

PIN on COTS refers to cases where PIN entry is applied on a consumer touchscreen device, such as a smartphone or a tablet. For this solution, the merchant uses a PCI-certified card reader, without any PIN entry capability, in order to communicate with the payment card. The mPOS dedicated app on the smartphone or tablet shows a virtual PIN Pad for PIN entry when needed. While the external card reader needs to be PCI-approved, the consumer-grade mobile device is not subject to any PCI certification. The PIN entry takes place on a non-certified consumer hardware. The software-based PIN entry process is used to securely authenticate the user. The software managing this PIN entry must be PCI certified.

In Figure 3-3 the SPOC and CPOC solutions are shown. The PIN entry processing component is shown grey because it is not officially part of PCI certification yet. Due to the likelihood of it to be added to PCI certification it will be considered in the scope of this research. Mastercard and Visa have already published specifications for this solution. These specifications will be used to identify the additional requirements that can be expected from PCI (Visa, 2021).

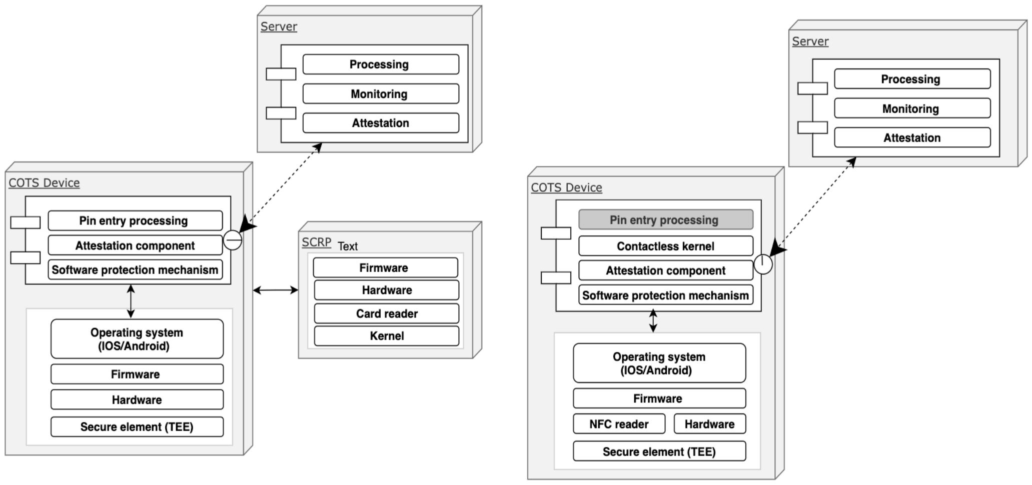


Figure 3-3 The move towards fully software (SPOC & CPOC)

3.4.1 (M)POS in components

With the help of informal conversations with multiple PSP vendors the various components in CPOC and SPOC devices have been identified. These will be used to identify the components that can be moved out of the COTS devices. The various components will briefly be explained in this paragraph and are visualized in Figure 3-3.

The secure element - is often integrated in a COTS device or in its mobile System-on-Chip (SoC). A secure element can be used to provide a robust security environment in which transactions and payment data can be processed. This provides a high level of security equivalent to modern Smart Cards. The Secure element is integrated in the COTS device and therefore it introduces a dependency on the hardware device it is coupled to. This means that just like with the TEE using a secure element can cause additional complications when the PSP wants to collaborate with multiple vendors that use different hardware.

The Trusted Executed Environment (TEE) - or for IOS called the Secure Enclave is part of the smartphone platform and integrated in most modern smartphones. The TEE offers hardware support for software security and therefore can be seen as a hybrid form of security. The TEE can be used as a secure isolated environment that can host isolated trusted applications guarded from the 'normal' operating systems environment. This is directly the main advantage of the TEE, it enables app builders to allow secure entries such as a PIN to happen while locking down the touch screen for other appliances in the OS. Similar as to the secure element this is a platform feature and therefore there is a dependency on the underlying platform for the developer. This potentially limits the number of devices and models on which the solution can be deployed. Furthermore, there will be another dependency on the update cycle of the underlying platform. This will be out of the control of the solution developer which can cause additional dependencies.

The Secure Card Reader (with PIN) (SCR (P)) device - can be used to obtain the card data via Contactless, Magstripe or Chip entry. The SCRP provides additional security because the PIN and the card data are provided on two different systems. The device is SRED enabled, which means that it should fulfill the PCI PTS POI standard of detailing the security controls for devices that protects account data (PCI - PTS POI specifications, 2020). According to the PCI standards the SCRP should provide:

- The protection of card data retrieved from the card reader
- Decryption of the encrypted PIN retrieved from the payment application
- Translation of PIN data into the required format
- Re-encryption of the PIN data

These requirements potentially limit an architecture that enables flexibility and allows for multiple devices. Therefore, this research will also consider a SCR(P) with the single function of gathering 'raw' card data. This allows the solution to exclude payment processing from the SCRP while still being able to use the SCRP for gathering card data.

Attestation - The general act of attestation is the interaction between a verifier, which can be server based, and a prover. This interaction is used to determine the status of the prover with regards to security. The attestation consists of two components. One on the COTS device and one somewhere outside of this device in a backend server. This allows the attestation component in the backend to process attestation health check data from the application on the COTS device. This process provides pre-established security policies and can determine whether the COTS device is safe. The attestation component is obligatory by both PCI requirements as well as the requirements from MasterCard and Visa. (PCI - CPOC standards, 2020)

The Monitoring component - is connected to the COTS Device and provides security controls to detect, alert and mitigate attacks and threats against the entire solution. For this, the monitoring component makes use of the attestation component.

The Processing environment - receives the encrypted card and PIN data and then processes the payment. This can take place on the terminal as well as in a server.

The Kernel - is a system provided by the various card schemes that contains interface routines, control and security functions and additional logic to manage a set of commands and responses to retrieve the necessary data from the card to complete a transaction. Each scheme has its own Kernel such that the cards of the various schemes can communicate with the terminal system via the Kernel. The Kernel processing enables the interaction with the card between the selection of the card application and processing the outcome of the transaction. This component is originally located in the terminal. This research will try to identify how this component can be split in order to relocate certain features of the terminal outside of the terminal.

3.4.2 PIN on COTS flow explained

Figure 5 gives a general overview of the payment process as explained by PCI in which PIN is entered on the COTS device (Software-based PIN Entry on COTS (SPoC) Security Requirements, 1.1, 2020).

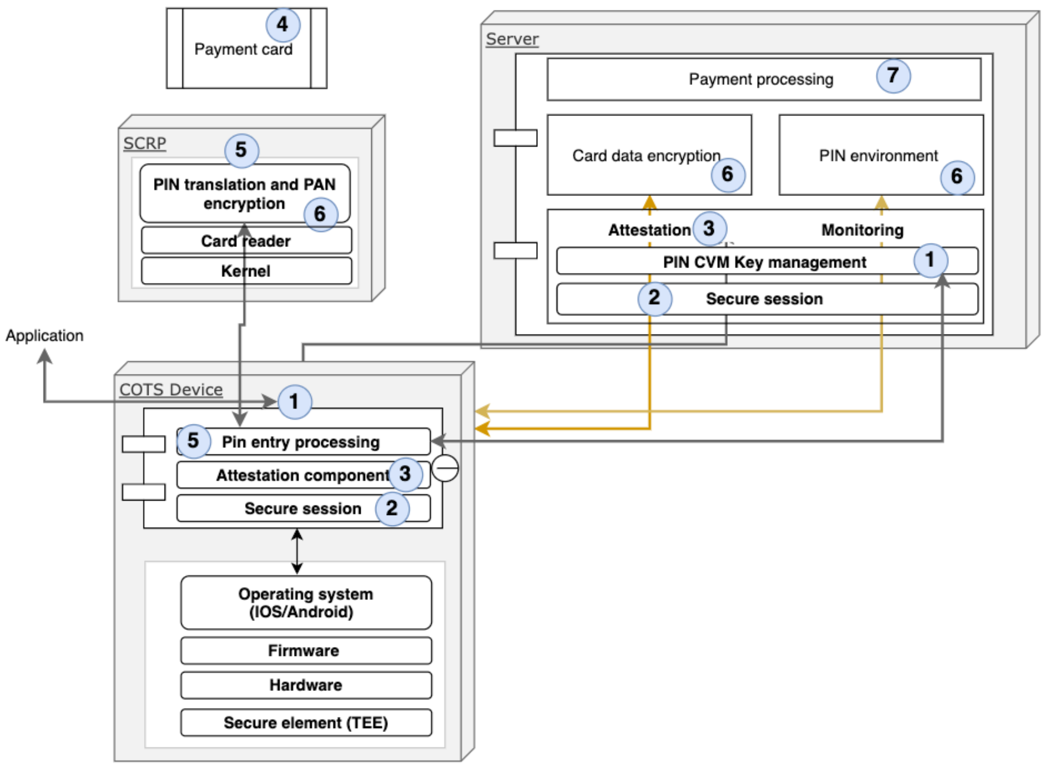


Figure 3-4 SPOC with PIN flow

The following steps illustrate a payment flow including PIN:

1. The card reader or SCRP and the PIN entry application are initialized with their financial keys.
2. Between the PIN entry application and the server-based monitoring system a secure communication channel is formed.
3. The server-based monitoring and attestation system determines the status of the payment application and its platform.
4. An EMV card or a payment device is presented for card entry
5. The application renders a PIN entry screen on the COTS device and the PIN is entered. This data is enciphered and sent to the card reader.
6. The Security or SRED component of the card reader enciphers the data using encryption keys.
7. The payment transaction is processed.

3.4.3 Path to offering a full solution

The future payment systems that can be considered relevant for this research should either have the possibility to be certified or should be likely to be certified in the future. Whether it is likely that a solution will be certified in the future is determined by conducting informal conversations with various well-known PSP's. After the classical terminal and the wearable mPOS terminal the new innovations can be divided in four solutions. These can be considered as a sequence of mPOS devices in which the SCR with no PIN is the simplest implementation and the CPOC the most complex solution due to the difficulty of protection. The different functionalities of these devices have been determined by literature research and informal conversations with the payment solution certification experts mentioned in Chapter 4 and are shown in table 3-2. True indicates that the specification is required or available. False means that it is not.

Table 3-2 the relevant future mPOS solutions

	OPTIONS			
	SCRP No Pin	SCRP PIN (SPOC)	CPOC No Pin	CPOC PIN
PCI Certification	SCRP	SCRP + SPOC	CPOC	No certification yet
Works on Apple iOS	TRUE	TRUE	FALSE	FALSE
Works on Android	TRUE	TRUE	TRUE	TRUE
Library/Application Needed	TRUE	TRUE	TRUE	TRUE
Kernels on COTS	FALSE	FALSE	TRUE	TRUE
Monitoring & Attestation Requirement	TRUE	TRUE	TRUE	TRUE
Allows EMV	TRUE	TRUE	FALSE	FALSE
Allows MSR	TRUE	FALSE	FALSE	FALSE
Allows EMV Ctls	TRUE	TRUE	TRUE	TRUE
Allows for PIN Entry	FALSE	TRUE	FALSE	TRUE
SCRP Certified Device	TRUE	TRUE	FALSE	FALSE
Use of Hardware protection (TEE)	TRUE	TRUE	TRUE	TRUE

Depending on the region, technical architecture and business models these functionality limitations can be blocking for a merchant.

Technically the most logical ordering of a SPOC/CPOC results from a combination of Shared architecture, the opportunity for merchants and technical capabilities. The following flow indicates the sequence in which the innovations are expected to be adopted with the general necessities for adoption.

1. SCR with NO PIN with IOS based Cash registers
    a. Less relevant for the Western-European market due to the lack of PIN
    b. IOS is easier to secure
    c. SCRP certification

2. SCRP with NO PIN with Android based Cash Registers
    a. Android SDK
    b. SCRP Certification
3. SCRP with PIN Entry for global Platforms/Enterprise markets with iOS-based Cash registers
    a. IOS CVM Application
    b. SPOC Certification
    c. Monitoring and Attestation framework
4. SCRP with PIN Entry for global Platforms/Enterprise markets with Android based Cash registers
    a. Android CVM Application
    b. SPOC Certification
5. CPOC for global Platforms/Enterprise markets with Android based Cash registers
    a. CPOC Android SDK / Application
    b. Payment Kernels
    c. CPOC Certification
6. CPOC with PIN
    a. Needs a waiver certification from Mastercard and Visa

The distinction is being made between Android and IOS because the IOS hardware does not yet allow external applications to use the NFC reader. Due to the innovative nature of this subject the assumption will be made that IOS will allow NFC reading once the CPOC solution is fully accepted.

3.5 Background information on utilized architecture technology

To substantiate the design decisions that are visualized in Chapter 5 some background information will be indicated in this paragraph. This will include information on technologies that enable certain design decisions and information that explain the terminology used to identify or group POS components. This sub-chapter aims to provide readers with little technical knowledge on payment architectures the necessary background to understand the output of this research. Furthermore, this sub chapter explains some expert output and substantiates design decisions.

3.5.1 Latency

Latency is an important factor influencing the performance efficiency of a software solution. The design decisions made in Chapter 5 will almost all have an effect on latency.

Latency describes delays in communication over a network. Latency meaning in networking is best thought of as the amount of time it takes for a packet of data to be captured, transmitted, processed through multiple devices, then received at its destination and decoded. The latency can be dependent on many variables such as coding structure, software architecture, distance or hardware. Therefore, it is not possible to theoretically define the amount of latency caused by design decisions in a monetary manner. This applies to the holistic architecture level that this research is being conducted on. In further more detailer research, building an example solution could help in monetizing latency.

3.5.2 HTTP versus Websocket connections

Hypertext Transfer Protocol (HTTP) is the protocol for communication between a web client (i.e. an application) and a server. This protocol is widely used on both the worldwide web as well as on local networks.

The downside of this protocol in comparison with the newer Websocket protocol is that it is not persistent. This means that with HTTP, subsequent request has to establish a new connection and security handshake for each signal. This causes additional latency in communication (Stenberg. *2014)*.

Websocket is a network protocol that allows for full-duplex communication over a single TCP-connection. Just like HTTP it is designed to provide communication between a web client and a server. It has similarities with HTTP because it uses a similar handshake. The important difference for this research is that it allows bidirectional data traffic without the necessity of additional requests. By eliminating the need for a new connection with every request, WebSockets greatly reduce the data size of each message, drastically decreasing latency. After the initial handshake, which includes standard HTTP header information, all subsequent messages include only relevant information. This reduction in latency enables fast AV transport and communication. The Websocket protocol is therefore a widely used technological innovation for IoT purposes (Silva, et al., 2018).

3.5.3 Terminal Driver and Terminal Controller

This paragraph includes background information specifically for the design shown in paragraph 5.7.

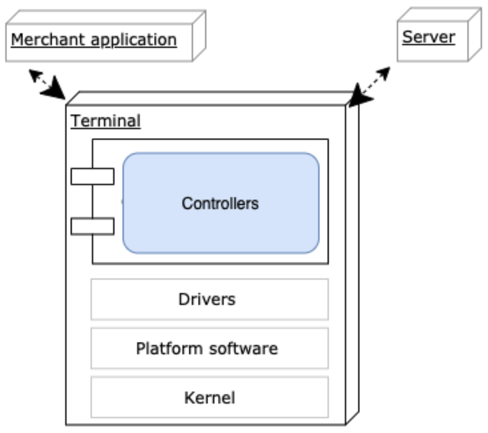


Figure 3-5 Terminal Drivers and Terminal Controller

In an overly simplified flow, a component of the payment terminal called the driver communicates with the hardware to obtain data from the card. Another part of the payment terminal called the Controller decides what happens with this data. The location of these components is illustrated in Figure 3-5. The Controller lives in the application and includes the business logic. The device driver is a software program that runs as a part of the operating system and is directly communicating with the hardware. This allows the system to be empowered with the specific commands needed to operate the device. The OS does so by combining the hardware and software techniques. The operating system communicates with the I/O hardware via the device driver software. This software belongs to the device and cannot be replaced or moved.

The term Controller is derived from the Model-view-Controller (MVC) design pattern. In this pattern the Controller represents the application logic or in other words the business logic of the application (Leff & Rayfiel, 2001). The business logic implies the decisions that will be made in the device that are dependent on the input. Examples of these decisions are the selection of one of the payment applications available on the card or whether PIN should be asked or not. Since these decisions are depended on the data that has been inserted a lot of communication takes place between the Controller and the drivers.

3.5.4 Rule Based Engine

This paragraph includes background information specifically for the design shown in paragraph 5.7.

The business logic living in the Controller encompasses what the application can or cannot do. Data is gathered via the driver and by evaluating this data decisions are being made by the Controller dependent on the logic that lives inside of it. This process goes back and forth between the Controller and driver, dependent on the amount of decisions that have to be made. Moving the Controller to a server would therefore increase the time it takes for a signal consisting of data (for the Controller) or instructions (back to the driver) to get to its destination. This can be compensated by decreasing the number of signals that will be send between the Controller and the drivers. Expert interviews and informal discussions under guidance of architecture expert Mayke Ploeger have resulted in an architectural change introducing a rule-based engine to replace the current Controller's structure.

Using a rule-based engine decreases the complexity of the Controller's application by encapsulating business logic and separating it from the rest of the applications code. This would decouple the storage and management of the business logic in the Controller by maintaining business rules in a repository isolated from the rest of the code. This has some advantages:

- The business logic is no longer integrated or scattered among other parts of the application allowing for modularization of the Controller
- The complexity of the application will be reduced
- The separation of business logic allows for the rules to be reused for other drivers
- The rules can be updated separate from the application on the device
- The rules can be sent in a decision tree

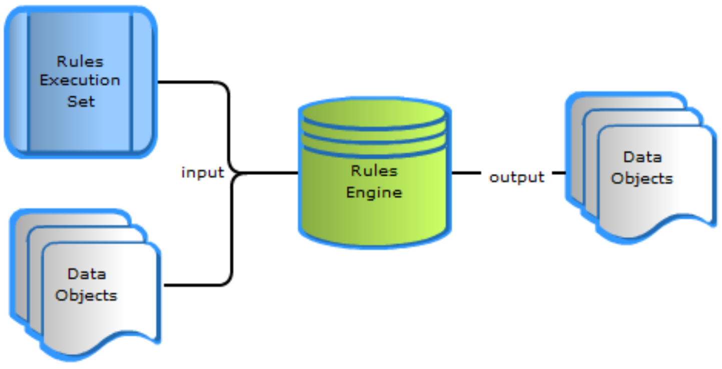


Figure 3-6 Rule Based Engine

Using a rule-based engine allows the application to send a decision tree including the various rules. The decision tree will consist of all possible scenarios and enables the application to make decisions based on the data objects gotten from the driver. A simple overview is shown in Figure 3-5.

1. The rules execution set or also called decision tree is send to the rules engine living in the application
2. All data gotten from the drivers is send to the rules engine
3. The data goes through the decision tree formatting it into the Controller output

This decreases the number of signals send between the server and the application because the entire rules set is sent in one or two signals. Furthermore, the business logic can be changed and altered without updating the application because the business logic is being send from the server for each transaction.

3.5.5 The Transaction Kernel

This paragraph includes background information that can specifically be utilized for Chapter 5.8

The Kernel shown in Figure 3-7 is a partly software component that contains interface routines, security, control functions, and logic to manage a set of commands and responses to retrieve the data from the Card for transaction completion. The Kernel processing covers the interaction with the Card between the selection of the card application (excluded) and processing of the outcome of the transaction (excluded).

The regular POS terminal as described in the EMV protocol book C-2, Contactless specifications for payment systems (EMVCO, 2021) has a general architecture that roughly consists of an Application and a Reader. The distinction between an Application and a Reader refers to a separation in functionality and responsibility between two logical entities.

The Application is the component that connects to the authorization and/or clearing network and that together with the Reader makes up the POS System. The Application and the Reader may exist in a single integrated device, but are considered separate logical entities in this research. In the classical terminal the application would include the Controller with the business logic, transaction logging, data storage and additional services.

The reader consists of the driver combined with the payment Kernels. The Reader is the device that supports the Kernel(s) and provides the contactless interface used by the Card. Although this can be an integral part of the POS System, it is considered in this research as a separate logical entity. This allows for the communication with the card, message sending, application selection, management of communication between the Kernel and the terminal application and the processing of the Kernel outcome.

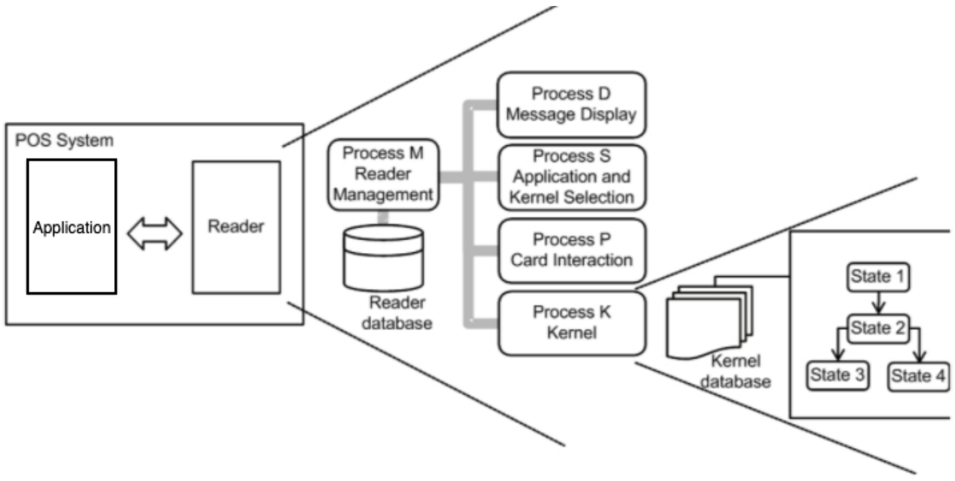


Figure 3-7 Kernels detailed architecture

Figure 3-7 gives insight in the component of the terminal Reader. On the left side of the image the terminal Application is shown that communicates with the Reader. The Reader involves various processes which can mainly be divided in the Main process, The Display process, The selection process, the PCD process and the Kernel process. The Kernel process can then again be divided in the Kernel software and the Kernel database. Some explanation on Figure 3-7 and the Kernel responsibility is provided in Table 3-3.

Table 3-3 Kernel processes

Process	Responsibility
Process P(CD)	Management of the contactless interface
Process D(isplay)	Management of the user interface
Process S(election)	Selection of the Card Application and Kernel
Process K(ernel)	Interaction with the Card once the application has been selected, covering the payment and data storage transaction flow specific to Kernel 2
Process M(ain)	Overall control and sequencing of the different processes. As part of this role, it is also responsible for the configuration and activation of the Kernel and the processing of its outcome. Process M is also responsible for initiating the housekeeping within the Kernel.

In paragraph 5.8.2 a new Kernel structure is proposed, the cloud Kernel. In this structure the Kernel processes will be divided and partially moved to a server. To give some insight in the consequences and opportunities of moving these processes to the server, some simplified background information on the processes is given in the following section.

There are three different Kernel levels. Kernel level 1 indicates the native software directly connected to the hardware (the bits and bytes). Level 3 considers the payment application and level 2 contains the logic to retrieve card data. In this and in most other literature, when talked about the Kernel this means level 2 Kernels.

Table 3-4 Kernel responses (EMVCO Book C-2, 2021)

Services	Signal
Return authorization or clearing record	ACT
Stop processing	STOP
Remove the logs	CLEAN
Return data from the Kernel or the card	DET/DEK

Responses as noted by the EMV contactless protocol Book C-2 are as shown in table 5-7 (EMVCO Book C-2, 2021).

Process P shown in Figure 5-10 implements functionality described in the EMV book Kernel L1 (EMVCO Book C-1, 2021) and ISO 7816-4 protocol and manages the access to the card as illustrated.

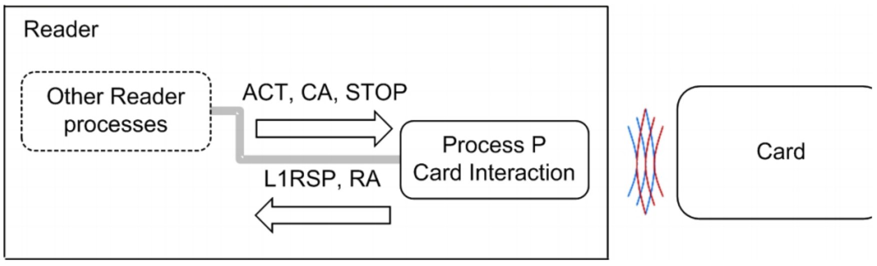


Figure 3-8 Process P

Process P initiates the interaction with the card. This component is therefore closely attached to the Level 1 hardware part of the device and cannot be fully removed from any device. For the SPOC solution process P takes place in the SCRP, the CPOC solution in the NFC reader and for the classical terminal it uses the driver.

Process D manages the User interface requests as defined in the EMV book A protocols. As illustrated in Figure 5-11, process D allows signals to stop the display and to send messages regarding status, amounts, currencies or languages (EMVCO Book A-1, 2021).

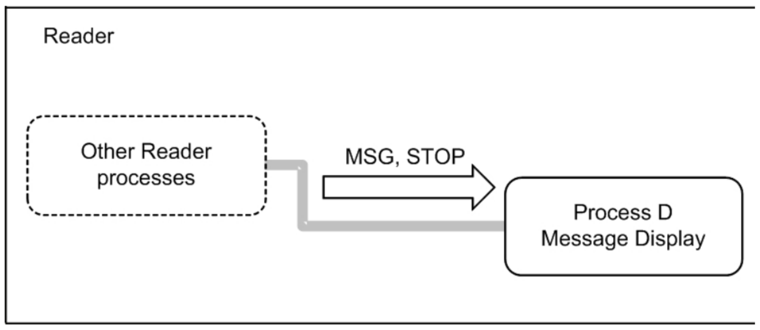


Figure 3-9 Process D

Process S shown in Figure 5-12 manages the application and Kernel selection. When one application (i.e. Mastercard) is selected the applicable Mastercard Kernel will be identified and its Kernel ID will be sent via process S. Process M initializes process S by sending the list of Application Identifiers (AID) and Kernel ID combinations.

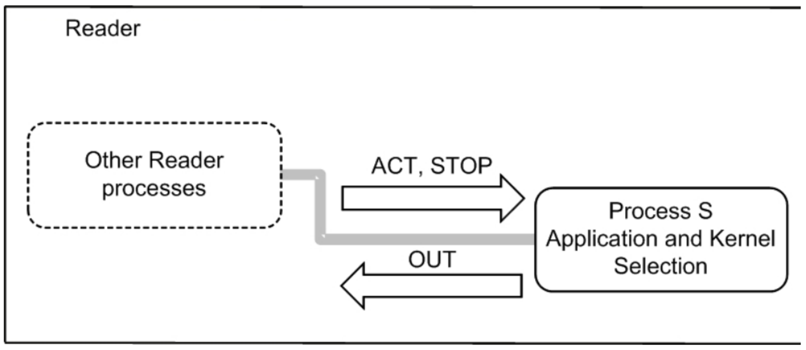


Figure 3-10 Process S

The Reader may support multiple Kernels but only one Kernel will execute at a time. The Kernel that is activated depends on the information returned by Process S, which may in turn depend on data retrieved from the Card. For each transaction, Process K is initialized with a Kernel-specific dataset. Within the different available datasets, the value of the data objects may vary depending on the selected AID and the transaction type.

Once the Kernel is selected and configured, it executes as Process K which is shown in Figure 5-13. Using the services of Process P as an intermediary, Process K manages the interaction with the Card application beyond application selection. Upon completion, Process K sends its results to Process M in an OUT Signal and then terminates (EMVCO Book C-2, 2021).

As a part of the interaction with the card the Kernel 2:

- Verifies the compatibility between the Kernel settings and the card settings
- Reads and writes payment data
- Determines the need for PIN or CVM

- Authenticates data
- Informs process M of the transaction outcome (OUT)
- Requests messages to be displayed dependent on the transaction.

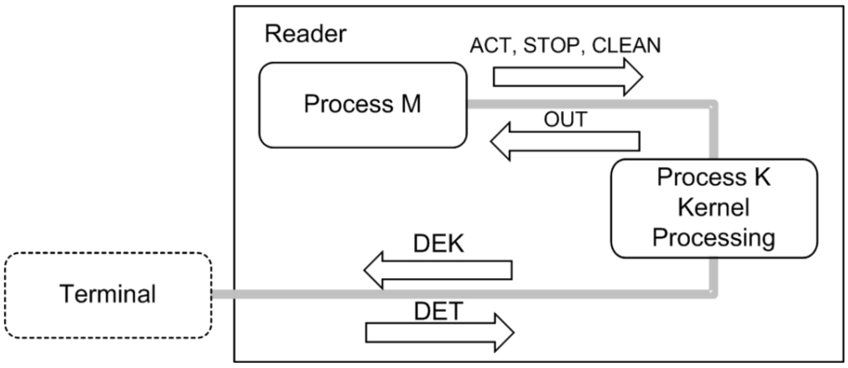


Figure 3-11 Process K

3.6 Summary of (m)POS solutions and architectures

In Chapter 3 a literature gap has been presented that indicated that no literature is available on software based mPOS solutions as well as cloud-based payments. Furthermore, sub-question one: "*What are the current and future mPOS solutions?*" has been answered by providing the chronological list of expected mPOS solutions and their specifications. The current and future mPOS solutions consists of the classical terminal, the wearable MPOS, the SCRP with SPOC and the CPOC solution. The current as well as the future mPOS solutions are visualized in Figure 1-1 and table 3-2 and have been explained in detail. mPOS solutions that cannot be certified or that are not expected to be certifiable in the future have not been considered for this research.

The background information necessary to conduct this research and to explain the design has been conducted using informal conversations with PSP's terminal vendors and literature. The second part Chapter 3 provided thorough information on the architectural components of current and future mPOS solutions. Furthermore, the relevant technologies necessary for architecture iterations have been elaborated on. These technologies, once combined, will show to have a significant impact on architectural possibilities. The implementation of these technologies in combination with the explained terminal components and mPOS requirements will be elaborated on in Chapter 5.

4 Data gathering and Method development

In this chapter phase 2 of DSRM, the objective of a solution will be further specified by gathering requirements. The objective of this research earlier mentioned in paragraph 2.4 is to design a decision flow that provides an overview of the architectural changes possible for both the current as well as the future mPOS solutions. This decision flow should:

- Enable architects to identify a one-size-fits all architecture that is flexible enough to provide for the current as well as the future (m)POS solutions
- Comply with security and PCI standards
- Be technically feasibly
- Support researchers and companies in making architectural decisions.
- Help identify the effect that changing the architecture will have on the stakeholder quality criteria.
- Identify what scenarios need additional research to monetize the effect of a flexible architecture.

This is still a general objective and therefore in the second phase ‘Define the objectives of a solution’ the problem identified by the systems stakeholders mentioned in Chapter 1 will be transformed into system objectives or meta-requirements. The problem does not necessarily identify all objectives. For that reason, there remains a step of determining the performance objectives of the, to be designed, artifact (Peffers et al., 2007). These objectives will mainly be qualitative to describe how the new architecture should form a solution to the problem addressed in phase 1. Knowledge on the current solutions as well as on the quality criteria that can form a base for the, to be developed, solution should be derived in this phase. Therefore, the second research question will be answered. “*What are the stakeholder quality attributes for the most significant current and future mobile POS solutions?”*

To establish the quality criteria the continues architecture principles explained in Chapter 2 will be followed. These guiding principles have been of assistance in phase 2 and 3 (Defining objectives and architecture design) of the DSRM methodology.

Secondly, the strategy for gathering data by conducting qualitative research as explained in Chapter 2 will be elaborated on and applied. Furthermore, using judgement sampling the process of choosing participants and conducting interviews is represented.

Lastly the quality criteria are obtained from the gathered data using qualitative analysis. The qualitative data analysis as explained in Chapter 2 will be elaborated on in this chapter. The coding using Atlas.Ti will be summarized in Appendix A/B. Furthermore, its results will be illustrated in Chapter 5.

4.1 Gathering data

In the Continues architecture methodology as well as many other qualitative research methodologies (Curran et al., 2014 ; Brittner, 2014), three basic techniques for gathering and managing requirements are defined. Stakeholder interviews, Joint Requirements Development (JRD) sessions and conversations with the business partners or product owners. Although these techniques can be used independently a combined approach will give a more holistic view of the stakeholder requirements and quality criteria. This subchapter will show the process of gathering data using the methodology explained in Chapter 2.

The majority of the experts have been provided by the PSP Adyen. Furthermore, expertise was gotten from informal interviews with anonymous sources among whom security certification experts, payment scheme providers and several terminal, security and payment service vendors. Furthermore, market experts were provided by 7 well known market platforms representing multiple companies.

4.1.1 Joint Requirements Development sessions

The expert panel is a commonly used method to elicit expert knowledge. Four expert panels have been conducted. One for validation of the quality criteria, two for gathering the quality requirements and one for the design evaluation.

As explained in the methodology chapter the expert panels have been conducted in an unstructured process to cater for a broad scope and input of ideas for development. It consisted out of multiple two-hour open discussions. Thirteen experts were invited with various areas of expertise. The goal of the orientating expert panel was to gather the architectural requirements for an mPOS architecture and determine its technological feasibility.

The expert panel had the following functions:

- Retrieving relevant information and knowledge
- Synthesis of the gathered information
- Stimulating new insights and views
- Validation of proposed concepts.

4.1.2 Merchant Interviews

A significant part of the data used in this research has been collected by the use of Subject Matter Expert (SME) interviews. The complexity of the problem requires a less directed approach and a more qualitative interviewing strategy. Therefore, the data collection in this research is mainly focused on unstructured and semi-structured formats because more structured interviews often produce quantitative data (DiCicco-Bloom & Crabtree, 2006). How to conduct the interview is also dependent on the type of stakeholder that will be interviewed and the data that is expected. The main goal of the merchant interviews is: “Gather the merchants use cases and requirements for their required mPOS solution.”.

4.1.3 Choosing participants

The subject of this research, POS terminals requires very specific expertise on developing a payment architecture. The subject of cloud computing is an area that is still in its infancy just as the subject of mobile application development. This requires a broad range of expertise. There is only a limited group of people that can provide the required expert information and it can be difficult to find the right knowledge (Turner, 2010).

In order to find the right interviewees to acquire the required knowledge, interviewees have been sampled based on judgement sampling. Judgement sampling, or also called authoritative sampling, is a non-probability sampling technique where the researcher or an expert selects units to be sampled based on expert knowledge or professional judgement (Hamed, 2020). This sampling method is most effective when only a limited number of individuals possess the trait that a researcher is interested in. Therefore, the complex nature of this research subject in addition to the limited timeframe in which it is conducted make authoritative sampling the most suitable sampling method. Furthermore, authoritative sampling refers to the process where subjects are selected based on the basis of their expertise in the subject under investigation.

The aim of this research is not about generalizing the findings to the entire society. Therefore, interviewing specialists allows the research to go faster while maintaining the right expert data input. Therefore, authoritative sampling is a better suited method in comparison with other probability sampling methods such as systematic sampling or cluster sampling. Furthermore, authoritative sampling is cost and time effective, which makes it more interesting for this research due to its short timespan in comparison with a more time-consuming snowball sampling method (Hamed, 2020).

A distinction can be made between the stakeholders and the technical experts. The stakeholders will be concerned with the outcome of this architecture and can therefore provide input on the architecture requirements. As mentioned in Chapter 2, these stakeholders consist of users (or also called merchants), payment service providers and security/certification authorities. The users are companies that will likely be using the new mPOS terminals. They will be focused on the feature requirements that the architecture will provide. The payment service providers can provide technical expertise on the technical requirements. Lastly, the security/certification authorities will provide the boundaries in which the solution should live to be secure. The technical experts might also be stakeholders. Due to the rapid changes and the closed nature of the financial technical environment, little literature is available on new mPOS innovations and architectural possibilities. Therefore, the data on designing the architecture has been gotten from technical experts.

To prevent bias the SME's were not only chosen by the researcher but also by other experts. The technical experts needed to comply with the following prerequisites:

- The expert should at least have conducted one year of working on a project within the PSP system.

- The expert should be representative for the discussed area of expertise. For example, gathering information on applications should be done not only with an app developer but with app developers from all operating systems within the scope of this research.
- Each architectural component mentioned in Chapter 5 should be validated by at least two specialists that have expertise on that particular subject.

The stakeholder experts needed to comply with the following prerequisites:

- The relevant merchant markets should be represented by at least one expert.
- The users should be represented. This entails retail, hospitality and Micro merchants
- The expert should have more than 1 year of experience in their field of expertise.
- The expert should be willing to provide their expertise.

The architecture users are shown in Table 5-1. To get a broad range of data the experts that were used for data gathering are experts responsible for big companies or platforms. These platforms represent multiple businesses of the markets that are included in this research. Due to privacy reasons the names of the companies are not shared.

Table 4-1 Merchant expert interviewees

COMPANY/PLATFORM	MARKET	REACH
PLATFORM A	Hospitality	1000+ businesses
PLATFORM B	Micro merchants	300.000 merchants
COMPANY A	Retail	5000+ stores
COMPANY B	Retail	90+ stores
PLATFORM C	Retail	10+ grand businesses
PLATFORM D	Hospitality	12.000+ businesses
PLATFORM E	Retail (healthcare)	-

The SME's selected for this research are shown in Table 4-2. All SME's complied with the expert prerequisites. Table 4-2 also indicates the methodologies used for the different experts. The experts that collaborated in the expert panel where utilized for gathering quality criteria. Which experts where used for gathering data on technological feasibility of the architecture will be further elaborated on in Chapter 5. For privacy reasons the names of the experts are not mentioned in this document.

Table 4-2 Subject matter experts

EXPERTISE	EXPERT	ROLE	EXPERIENCE	METHOD	EXPERT PANEL
POS PAYMENTS / ANDROID	A	Vice president Product POS	8+ years	Multiple Informal interviews	Yes
POS INTERFACE	B	Product manager	2+ years	Multiple informal interviews	Yes
POS DEVELOPMENT	C	SVP innovation & Development	10+ years	Informal interview	Yes
IOS & Java DEVELOPMENT	D	IOS app developer/ tech lead	10+ years	Semi-structured interview	Yes
POS PAYMENTS	E	Product manager / Team lead	10+ years	Multiple informal interviews	Yes
PAYMENT SECURITY	F	Certification manager	10+ years	Semi-structured interview	NO
C DEVELOPMENT / KERNELS	G	Senior C developer	5+ years	Semi-structured interview	Yes
ANDROID DEVELOPMENT	H	Senior developer/ Tech lead	5+ years	Semi-structured interview	Yes
JAVA DEVELOPMENT	I	Senior Java developer/ Tech lead	5+ years	Semi-structured interview	Yes
C DEVELOPMENT	J	Senior C developer / Tech lead	5+ years		Yes
C DEVELOPMENT	K	Senior C developer	10+ years		Yes
IOS DEVELOPMENT	L	Senior developer IOS	5+ years	Informal conversations	Yes
C DEVELOPMENT	M	Senior C developer / Tech lead	10 + years	Informal conversations	Yes
PAYMENT SECURITY	N	Security Senior	10+ years	Semi-structured interview	Yes

4.1.4 Interview process

The process of interviewing is summarized in table 5-3. This includes the objectives and the general outcome of the SME interviews.

Table 4-3 Interview process

Interview Type	Semi-structured expert interviews
Interview Strategy	Confirmatory and exploratory
Method	● Semi structured interview ● Confirmatory interview
Number of interviewees	7
Duration	1 hour
Objectives	Objective 1: Gathering merchant requirements Objective 2: Gathering input on potential architectural components that have not been considered yet. Objective 3: Gathering more technical requirements and their weight for the identified components Objective 4: Gathering merchant use cases to define a potential business case
Input	- mPOS examples - Semi-structured questions
Output	- Functional requirements - Non-Functional requirements - Use cases - Information on the importance and feasibility of the requirements
Interviewees	SME's mentioned in table 4-2

4.1.5 Qualitative data analysis

The qualitative data analysis has followed the objectives explained in Chapter 2. The objective of the analysis was to scope, gather requirements, substantiate the design and test the outcome. A well-known qualitative analysis methodology is called framework analysis. Framework analysis is a variety of content analysis (Ritchie & Spencer, 1994), (Ritchie et al., 2014). Like content analysis it is using a matrix to provide structure in the data. It provides clear and small steps in the qualitative analysis process. The analysis provides a hierarchical case and theme-based approach. It uses summaries to reduce the amount of data used. Furthermore, framework analysis allows the researcher to retain a link to the original data (Ritchie et al, 2014). The framework analysis was applied in five phases:

Familiarization - This involved getting more involved in the data by reading and re-reading the data gathered. The data gathered using the various qualitative methods has been analyzed to get familiar with the overall subjects and criteria.

Identifying a thematic framework - This phase assisted in identifying the key and sub themes in the data. The sub themes could be derived from the data and combined with the sub themes gotten from literature research. The subthemes for the merchant interviews are shown in Appendix A and B and its outcome will further be elaborated on in Chapter 6.

Indexing - In this phase the most interesting fragments in which the identified themes and sub themes come forward are being indexed and coded. Processing and analyzing the data has been done by transcription of the interviews, coding and categorizing the data using ATLAS TI 9 software. Using the ATLAS TI 9 software enabled the researcher to generate networks and identify unnoticed connections in the data.

Charting summarizing - To get a better insight in the indexed fragments, their understanding is summarized. These are visualized in the code groups shown in the Appendix.

Interpretation - The different themes and categories derived out of the data have been described in Chapter 6 and in Appendix A, B and E. This paragraph shows the result in descriptions, typologies, categories, linkages and eventually explanations of the quality criteria.

4.2 Summary of data gathering and method development

Chapter 4 has explained the process of gathering data using the methodology explained in Chapter 2. A variety of experts and market representatives have been utilized using several data gathering techniques. This resulted in a significant amount of data that has been used to both design and analyze an architecture structure that can fit the stakeholder needs.

The findings from the SME research and the expert panel have been combined and will be presented in Chapter 6. To limit the length of the report the raw data will be shown in a compact way and can be found in the Appendix. As mentioned in the previous chapter on method development functional requirements are not applicable for architecture design. The data gathered will be used to identify the architectural quality criteria and their importance. The criteria gotten from the expert panel and interviews have been grouped in code groups to determine the quality criteria relevant for this architecture. These criteria will be completed by adding security and certification criteria gotten from qualitative literature research. The quality criteria will be explained with the use of the widely used ISO 25010 software architecture quality characteristics (Haoues et al., 2017). This model will be altered according to the findings in this research. Chapter 6 will show the combination of the various requirements which will result in a utility tree. This will be used for the analysis of architectural decisions.

5 Towards a portable mPOS architecture

The third stage 'Design and development' will demonstrate the actual creation of the artifact, or in this research, the architecture decision flow diagram. This phase will demonstrate the outcome of the second phase by combining the gathered requirements and objectives with the information obtained from qualitative interviews with development experts. The feedback and testing phase are also considered in this chapter due to the iterative character of the DSRM methodology. A payment architecture decision flow will be designed in this stage that will attempt to combine the software solutions and the stakeholder requirements for a flexible one-size-fits-all mPOS payment solution. By documenting the decisions that need to be made to design this architecture an architecture decision flow diagram will be made that can be utilized by a variety of users with various requirements.

Firstly, the output of Chapter 4, which contained merchant criteria, security criteria and payment expert criteria will be combined in a quality attribute utility tree. Secondly, the design process will be explained. This is being done by elaborating on how additional expertise has been utilized during the design process. After that the design process as explained in Chapter 2 is followed. This entails scoping the design by drafting the architecture and identifying potential design decisions. Using this first draft the various design decisions will be outlined and analyzed using the quality attribute utility tree. This will be followed by a tradeoff analysis. The combination of tradeoffs and their effects on quality criteria will be illustrated in an architecture decision flow diagram and are the output of this design phase. The final artifact will be providing an answer for sub-question three: *"What payment architecture fits the requirements derived from sub-question one and two?"*

The methodologies used in the process of this design phase are explained in Chapter 2. All background information necessary to substantiate the architecture design choices are elaborated on in Chapter 3.

5.1 Combining the stakeholder's attributes

As mentioned earlier in this chapter, a set of functional requirements can often be delivered by various different architectures. Each functional requirement or set of functional requirements can be associated with quality attributes. According to the architecture literature found, the optimal architecture is one that best satisfies the quality attribute requirements (Cai et al., 2018). Not only the attributes gotten from the merchant and PSP SME interviews will be considered but also the security requirements should be added. The expert interviews have resulted in various requirements and constraints. These will be combined with security requirements to precisely determine and describe the quality attribute requirements in order to design the architecture.

5.1.1 Interview and expert panel analysis

The high-level view of the various code groups can be found in image 5-1. The criteria found in this phase will be used to provide insight and background knowledge on the quality criteria necessary for the architecture design.

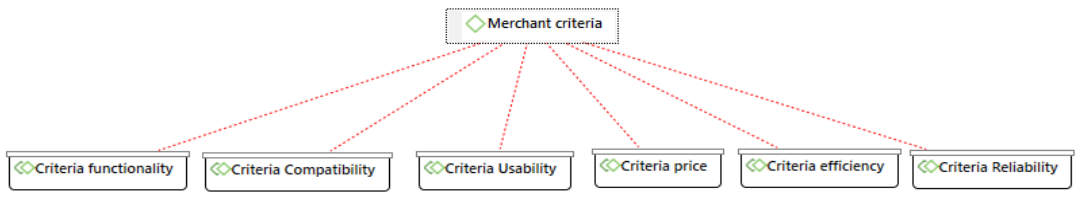


Figure 5-1 Quality code groups

The high-level code groups have been made to structure the merchant data and provide a clear overview. Further elaboration of these code groups can be found in Appendix B. Each criteria, idea or notion in the data has been given a specific code to investigate the trends, co-occurrences and dependencies in the data. The letter "G" on the images in the appendix stands for grounded and indicates the amount of quotations that are linked to that code. This does not necessarily indicate importance but it does indicate the occurrence in the data.

The technical functional requirements gathered by the expert panel can be found in Appendix A. The expert panels have resulted in some additional criteria that should be considered as boundaries in the design of an architecture:

- *Portability* - the solution should provide for the long term and should be flexible enough for future innovations.
- *Compatibility* - it should be possible to integrate with multiple different merchant applications.
- *Security* - the solution should allow for PCI certification
- *Portability* - the solution should support both Android and IOS
- *Functional suitability* **-** the solution should provide only its core payment functions including the major payment schemes (Mastercard, Visa and AMEX) but should be *modifiable* enough to extend to full functionality.

5.1.2 Security and certification requirements

Not only the expert and merchant requirements should be considered. The scope of this research is to design an architecture within the boundaries of the PCI security standards. The security objectives as stated by the PCI security standards are developed to provide a standard to which the architecture should comply. The objective of these security requirements is to "ensure the integrity of the NFC interface and contactless Kernel on the COTS device, and to reasonably ensure that the solutions provide adequate security mechanisms, controls, and mitigations to protect the consumer's account data and other assets, such as cryptographic keys. These requirements ensure protection from unauthorized disclosure, modification, or misuse by restricting the available attack surface and make it cost prohibitive to attack.” (PCI Security standard council, 2021)).

The security experts that have been consulted for this research did therefore not supply any expertise on security requirements. The experts agreed upon the assumption that a solution that fits within the boundaries of PCI requirements will be a secure solution. The expertise of the security experts has therefore only been used to gather information on the consequences of design decisions within the PCI and or scheme requirements boundaries.

For the COTS platform components, the main security objective is to provide a reasonable assurance that the components are kept up-to-date and have not been tampered. More detailed information on the security objectives can be found in the PCI security standards program (PCI Security standard council, 2021)) and in Appendix D.

5.1.3 Visualizing quality attributes

To manage and visualize the criteria, the quality attributes utility tree mentioned in Architecture Tradeoff Analysis Method (ATAM) will be used. This is a software engineering technique developed by the Software Engineering Institute (SEI) at the Carnegie Mellon University. It enables an architect to decide upon a suitable architecture for a software system by focusing on delicate points and trade-offs (Kazman et al., 2003). ATAM is suitable for this research because it allows its user to identify risks early in the life cycle, increase stakeholder communication and it clarifies quality attribute requirements. Furthermore, the ATAM methodology aligns with the Continues architecture principles.

The quality attribute utility tree is a prioritization of specific quality attribute requirements, realized as scenarios. The utility tree allows the quality attributes to move from and abstract attribute towards a more concrete utility. The goal of building the utility tree is to identify, refine and prioritize the most important quality attribute requirements derived from the qualitative data. The utility tree is determined in the following phases:

Quality attributes - The quality attribute requirements are recorded from the data as the highest-level nodes of the three.

Quality attribute refinements - These quality attributes are refined by using the gathered data and functionalities in combination with the ISO architecture standards (Haoues et al., 2017).

Quality scenarios - As mentioned earlier in this chapter, due to the general nature of the architecture functional requirements are outside of the scope of this research.

5.1.4 Quality attributes utility tree

To allow an architect to evaluate the different architectural options shown in Chapter 3 and 5 the quality attribute tree is shown in Image 5-2. The quality attributes are gathered by grouping the requirements gotten from the qualitative analysis and using them to determine which aspects of the ISO 25010 quality model to consider for this research.

The quality attributes utility tree is only determined until the second level. This allows us to validate the architecture components generally. Identifying the third layer is outside of the scope of this research. The third layer includes quality scenarios that identify the measurable objects. These scenarios are dependent on the type of company or vendor that wants to use the architecture decision tree and are therefore variable. These scenarios are also not dependent on the proposed architecture decisions. For example: A third level scenario could be: “When the application is loaded, the system should process a maximum of 20mb.” Such a scenario is not dependent on the proposed architecture since the architecture does not consider hardware components and therefore these are outside of the scope of this research. Exceptions, such as the effect on offline functionality will be mentioned during the design phase and will be considered.

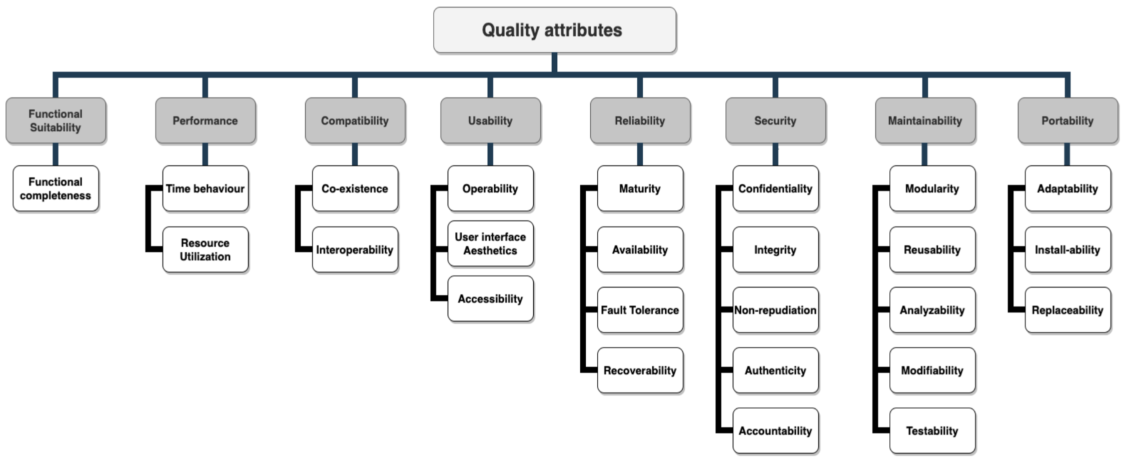


Figure 5-2 Quality Attributes Utility Tree

In the combination of data and literature the following characteristics could be defined within the boundaries of the ISO 25010 protocol:

Functional Suitability - The characteristic functional suitability constitutes the degree to which the designed architecture provides functionalities that meet the stated requirements. This characteristic can be explained by the following sub characteristic:
Functional completeness: The extent to which the designed functions provide for all the specified user objectives. Is the software application in line with the required specifications and does it provide the functions that are shown in Appendix C?
The functional correctness and appropriateness will not be considered since this is not measurable on architectural level.

Performance efficiency - This characteristic represents the performance relative to the amount of resources used under stated conditions. This characteristic is composed of the following sub-characteristics:

- *Time behavior*: The degree to which the response and processing times and throughput rates of a product or system, when performing its functions, meet requirements. Although the measurement of transaction speed is not part of the scope of this thesis it is an important criterion to consider while designing the architecture. The data showed that merchants require fast transactions as well as the certification authority which requires contactless card reading within 500 milliseconds according to the EMVCO book C-2.

- *Resource utilization* - Degree to which the resources such as data storage and network speed is being used by the solution.

The performance measure capacity is considered outside of the scope of this research since it is not dependent on the software architecture.

Compatibility - This is the degree to which the given architectural components can exchange information with other components, systems or products. Compatibility will be divided in the following sub characteristics:

- *Co-existence* - To what extent is the architecture capable of sharing a common environment and resources with other COTS devices?
- *Interoperability* - Is it able to exchange data and use the data that has been exchanged?

Usability **-** Degree to which a product or system can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in the use cases mentioned in Chapter 3. This quality criteria also takes the look and feel of the application into account. This criterion is composed out of the following sub-criteria:

- *Operability* - is the solution easy to operate and control?
- *User interface* – Is the solution considered well designed?
- *Accessibility – Degree to which the app can be used by people of all capabilities and characteristics.*
- *Appropriateness recognizability, learnability and user error protection are* not affected by a holistic architecture and are therefore outside of the scope of this architecture.

Reliability - Degree to which the architecture allows the system to perform functions well. The following sub-criteria will be considered:

- *Maturity* – The degree of stability during normal use
- *Availability* – The degree of availability of the solution under normal circumstances
- *Fault tolerance* – The degree of working of the architecture when the hardware around it fails
- *Recoverability* – When an interruption or failure occurs. Can the architecture communication still recover or proceed?

Security - The degree to which the architecture protects the system and confidential information. All sub-characteristics will be considered by the PCI authorities. As mentioned before these are: Confidentiality, integrity, non-repudiation, accountability and authenticity. All architecture options will be within the PCI and/or VISA/Mastercard protocol.

Maintainability - This characteristic represents the degree in which the solution can be modified or maintained in the future. This includes the following characteristics:

- *Modularity:* The degree in which the architecture is built in components. Components are easily interchangeable and therefore increases maintainability.
- *Reusability:* Whether the asset can be used in more than one system
- *Analyzability:* The degree of complexity in which it is possible to monitor and analyze the system.
- *Modifiability:* Degree of efficiency in which the architecture components can be modified after the design.
- *Testability* – How easy it will be to identify whether the testing criteria have been met.

Portability – The degree of suppleness to which the architecture can be coupled to a different hardware or software system. This allows the architecture to be flexible in use. This consists of the following sub-characteristics:

- *Adaptability:* How easy the architecture can be adapted for different or evolving hardware or software.
- *Install ability:* How easy it is to install the application logic
- *Replaceability:* How easy it is to replace (parts of) the architecture with different software or hardware.

5.2 Gathering expertise and continues feedback

The boundaries set by the expert panel, the information gotten from the literature and informal conversations with both vendors and experts have resulted in a general architectural approach shown in Chapter 3 and in the drafting phase in 5.5. Due to the confidentiality and novelty of payment systems there is little literature available to validate the proposed restructuring of the architecture and to help determine the technical consequences on the quality criteria. Therefore, the experts mentioned in table 5-2 were interviewed to gather the necessary information on design options. The data retrieved from these interviews is elaborated on in the remaining of this chapter. The background information gotten from these data is shown in Chapter 3.

5.2.1 Expert interviews

In extend to the interview methodology stated in Chapter 2, the expert data was gathered using four different qualitative research methods Turners (2010):

- Semi-structured interviews were used to identify the possibility of new design architectures and to gather more information on the technical feasibility of moving architecture components. These interviews have been recorded and transcribed.

- Informal conversations. Due to the Continues architecture approach the design is continue sly validated and altered to provide for changing requirements. These conversations have not been recorded.
- Expert panels. For validation of the proposed methodology and designs expert panels have been used to gather feedback in an efficient manner. These expert panels have been recorded.
- Informal conversations with vendors. 11 different companies with knowledge on one of the aspects of this architecture have been contacted to discuss and identify business cases and solutions. This information is sometimes shared in a confidential manner in will therefore not be used in this research. The information that can be shared and that is relevant for this research will be shared anonymously. These conversations have not been recorded.

5.2.2 Interview process

The interview consisted out of 5 elements:

1. Validating if the general POS components (Identified from literature review and informal talks) are identified and located correctly.
2. Gathering input on potential architectural components that have not been considered yet.
3. Gathering additional technical requirements and their weight for the identified components
4. Gathering design options (on the POS component of which the interviewee has expertise) and identify the effect they have on reaching the identified quality criteria.
5. Evaluation and validation of design decisions and trade-offs.

5.2.3 Interview guideline

In table 5-4 the guideline that was used for the semi structured expert interviews is shown.

Table 5-1 Expert interview guideline

Interview Type	Semi-structured expert interviews
Interview Strategy	Confirmatory and exploratory
Method	• Semi structured questions guided by visual aid of the conceptual POS components • Confirmatory interview
Number of interviewees	12
Duration	1 hour
Objectives	Objective 1: Validating if the general POS components (Identified from literature review and informal talks) are identified and located correctly. Objective 2: Gathering input on potential architectural components that have not been considered yet. Objective 3: Gathering more technical requirements and their weight for the identified components Objective 4: Gathering design options (on the POS component of which the interviewee has expertise) and identify the effect they have on reaching the identified quality criteria. Objective 5: Validation of design decisions and trade-offs
Input	• The general Point of Sale architectural components • Semi-structured questions
Output	• Confirmation on the general POS components that where drafted in the first design phase following the informal interviews and expert panel • New architectural components • Information on the importance and feasibility of the requirements • The different options possible in which the discussed architectural component can be build and their effects
Interviewees	C Developers, Java developers, IOS and Android experts and PSP product managers who are involved in the Point of Sale ecosystem

5.3 The process of identifying architecture trade-offs

A part of the design process has already been indicated by the visualization the Quality attributes tree in paragraph 5.1.3 and by establishing the relevant mPOS components in Chapter 3. The design phase will roughly consist of four sub-phases. Scoping, analyzing and development, identification of the solution and continues iteration. As followed by the continues architecture principles the design process was conducted by many iterations.

First the design was scoped following the objectives mentioned in phase 2 which helped to identify the boundaries of the solution. In phase 3 all quality criteria have been considered. And are illustrated in paragraph 5.1.3. As explained in the first chapters no final decisions have to be made, the tradeoffs itself need to be determined and final decisions will be made for demonstration purposes.

In the second sub-phase called *brainstorming and development of potential solutions* a design has been made using the requirements and the current mPOS solution architectures. These have been verified during brainstorm sessions with multidisciplinary development experts. The indicated design options are the outcome of this phase.

The third sub-phase *Identification of the optimal solution* is derived out of the scope of this research. Due to the holistic approach of this research and therefore the lack of functional requirements the optimal solution does not include a final architecture. The optimal solution is highly dependent on the architect's preferences and situation. Therefore, the optimal solution consists of a decision flow that enables architects to build an architecture portable enough for all mPOS innovations. The research presents the identified tradeoffs. The optimal solution can be made dependent on the quality criteria of the specific user.

The fourth sub-phase *Iteration,* is very much in line with the DSRM as well as with the continues architecture approach. A continues line of feedback has been managed by continues verification by informal conversations with experts. This has led to revisiting and to revising the former process phases.

5.4 Drafting the architecture outline

In the first phase of de design process the different software components that are subject to relocation have been identified. All components that are not directly connected to hardware can theoretically be moved to a server. This excludes the card and pin entry components but includes the components that process and command that entry. Furthermore, due to certification compliance the monitoring and attestation components cannot be relocated. The components and the risks that come with relocating them will briefly be discussed in this paragraph.

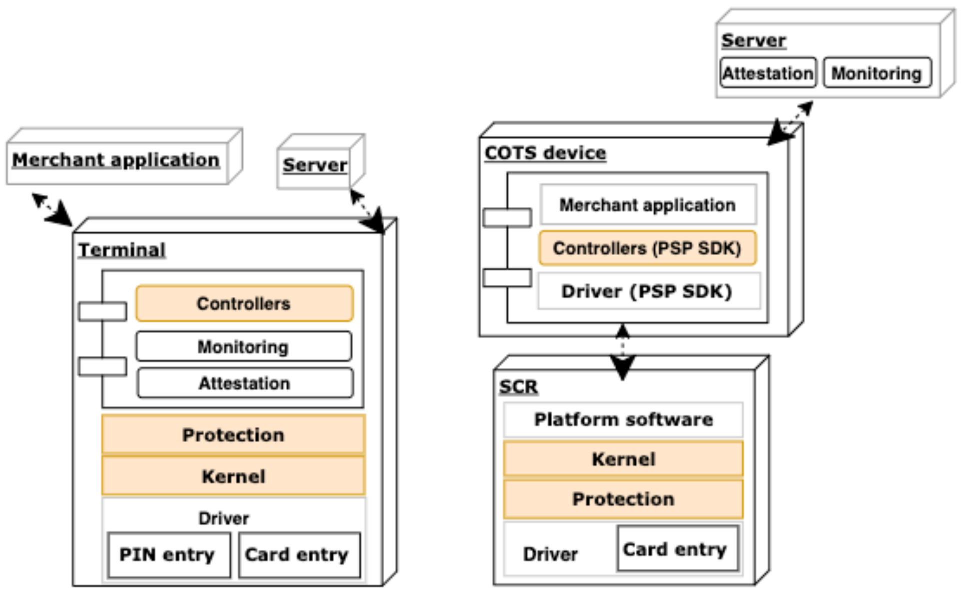


Figure 5-3 Movable POS components

The term Controller is derived from the Model-view-Controller (MVC) design pattern. In this pattern the Controller represents the application logic or in other words the business logic of the application (Leff & Rayfiel, 2001). The business logic implies the decisions that will be made in the device that are dependent on the input. Examples of these decisions are the selection of one of the payment applications available on the card or whether PIN should be asked or not. Since these decisions are depended on the data that has been inserted a lot of communication takes place between the Controller and the drivers. This makes it complicated to move the Controller to a more distanced location.

The Kernel contains interface routines, security and control functions, and logic to manage a set of commands and responses to retrieve the necessary data from the Card to complete a transaction. The Kernel processing covers the interaction with the Card between the selection of the card application (excluded) and processing of the outcome of the transaction (excluded). The Kernel is a complex computational component that sends and receives a high number of signals. During the transaction the Kernel is giving commands and keeping track of the status. A contactless transaction should process the card data within 500 milliseconds (EMV – Book -C Kernel L2 specifications, 2020). The Kernel is an important component that is partly responsible for the speed of that communication. Conversations with terminal vendors have indicated that moving the Kernel to the server is a highly discussed idea that is likely to be more successive due to current technological innovations. These will be further discussed in paragraph 5.8.

The protection components can live in the application as well as in the hardware of the COTS device. Moving the protection component to the server is not an option according to security experts due to the necessity of native checks.

The tradeoff between using an SDK or an Application or a hybrid variant of this tradeoff will not be discussed in this research. It is considered outside of the scope of the research because it does not have an effect on the possibility of moving the terminal solution to the cloud and can be considered an implementation detail. However, it is an important decision that will have to be taken while building the payment architecture. Therefore, this information is added in appendix E.

5.5 Architecture design options

The design decisions regarding the components identified in paragraph 5.4 will be shown in the coming paragraphs. First the expertise that has been used is indicated by showing a summary of the experts that were questioned and by providing the obtained and necessary background information. Secondly, the options are visualized and explained. By combining the technical capabilities of the options with the quality criteria an analysis can be made on the effects that a design decision will have.

5.6 Security

Some security background information leading to the security design options will be shown in this paragraph. Security constitutes the components in the terminal that are responsible for data protection. The security decisions and design options are based on and verified by the input of the experts shown in table 5-2.

Table 5-2 Security experts

EXPERTISE	EXPERT	ROLE	EXPERIENCE	METHOD
IOS DEVELOPMENT	1	IOS app developer/ team lead	10+ years of work on IOS SDK's, JAVA development and Tech lead	Semi-structured interview and expert panel
PAYMENT SECURITY	2	Certification manager	10+ years	Semi-structured interview
PAYMENT SECURITY	3	Security Senior	10+ years of work on terminal security	Semi-structured interview and expert panel
IOS DEVELOPMENT	4	Senior developer IOS	5+ years	Informal interviews and expert panel
ANDROID DEVELOPMENT	5	Senior developer/ Tech lead	5+ years of work on app development, Android SDK's and the full Android terminal solution	Semi-structured interview and expert panel

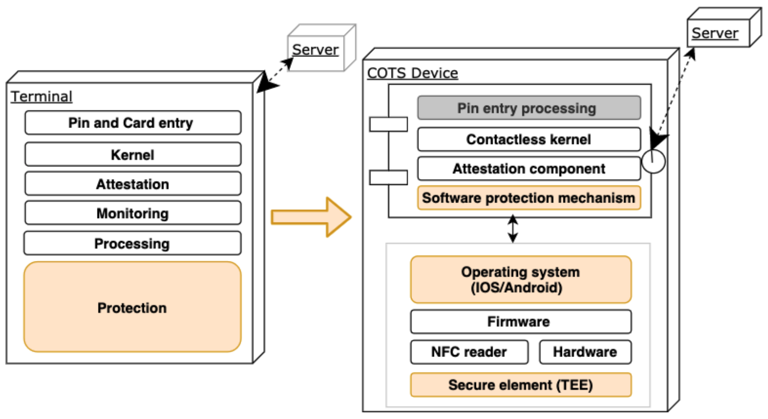


Figure 5-4 Security components

“Software” in Software POS does not mean that simple software sitting on top of the OS of a mobile device is going to replace all the features implemented in a classical POS terminal. The fact that all the transactions are going to be authorized online and that Magstripe transactions are prohibited, allows SoftPOS solution vendors to mitigate the risk of a security breach by reducing the functional surface of their software sitting on the device (Assadi, 2020).

The main challenge that faces a Software POS solution provider is ultimately to protect the cryptographic keys managed by the application and, additionally, the application code sitting on the mobile device. The biggest challenge is to find the balance between the security of the implementation and the usability of the service. Implementing the maximum level of security that the current Android OS could offer becomes a hurdle when it comes to guarantee the transaction performance (processing time on the device), the maintenance and the scalability of such a solution.

5.6.1 Architecture design options

The architectural security options consist of hardware and software-based security. Within hardware and software-based security some additional distinctions can be made in what type of security is used. All provided options have been proposed or verified by the experts mentioned in table 5-2.

5.6.2 Hardware based security

Hardware based security uses the COTS hardware to provide an additional layer of security. For IOS this is called the Secure enclave and for Android the Secure element and Trusted execution environment. Both systems allow the application to run secure elements of the process such as keys in its own CPU and storage. The hardware bases security makes sure that it is not possible to temper with the data.

These hardware components are used by Android and IOS OS to improve the security of libraries, such as the KeyStore, proposed to app developers to secure the sensitive data and the operations made by their apps. The Android APIs accessible to developers can be used to ensure that a keyset has been generated, or not, using a security mechanism provided by the OS and backed by a hardware component on the device. This is guaranteed by the latest versions of the OS. These new APIs make the development of secure applications possible and the full dependency on software security mechanisms, such as White box cryptography, less critical (Assadi, 2020).

The Secure element is a hardware form of security that comes embedded in some smartphone types. It provides a secure environment outside of the normal processor to process secure elements. Because of its embeddedness in smartphones is creates a dependency on specific devices. It is therefore a less required solution form. Out of the thirty analyzed mobile security or mobile POS vendors only 2 vendors supported hardware security (Mastercard, 2021).

5.6.3 Software security

The security system will always assume that the underlying hardware system can be compromised and therefore distrusts the OS. Even in this scenario it should be possible to protect the application of manipulation. Various complementary software security systems are available. A variety of layered security systems can will increase the security and the time required to tamper the system. Examples of these security mechanisms include Anti-cloning, Anti-rooting, Anti-key-recovery, Anti-analysis and Anti-instrumentation. The advantage of fully software-based security is its portability. It can fit to any device and is not depended on hardware. Furthermore, it increases control because it can be easily updated remotely.

5.6.4 Security reliant on COTS Platform

The OS system environment can be used to encrypt and stare secure data. However, a rooted device cannot guarantee a sufficient protection for these keys. A White box cryptography component helps achieving a better protection when properly integrated into the product. It allows the use and the manipulation of keys and ensures that these cryptographic elements remain protected during runtime. Nevertheless, an attacker who is able to take the control of the application and understand the implementation that is being made can recover the keys and get access to protected data. An additional layer of code protection is necessary to harden the application.

A clear distinction between Android and IOS can be made in this architecture choice. According to the interviews with both android and IOS experts the IOS operating system can be considered sufficient and within the boundaries of PCI requirements. The Android OS is due to its open source nature less reliable and will need additional security layers to comply with PCI requirements. Therefore, in the architecture design tradeoff a distinction needs to be made between IOS and Android operating systems.

5.6.5 Tradeoff analysis

Hardware-based security – The classical terminal is a secure device due to its hardware-based security. If the hardware would be broken the keys would get lost and could not be retrieved. Moving to software-based security compromises this layer of security. By using hardware components of the COTS devices this will be compensated.

Whether hardware-based security is more secure in comparison with software-based security is up for debate. In the qualitative interviews Alejandro Ferrer Delgado stated that software-based security could be just as good as hardware-based security. This is obviously dependent on the amount of security layers and the type of software-based security added. However, Tim Hartog a security certification expert mentions that there is no better security than hardware-based security due to the fact that hardware cannot be hacked. Both experts agree that from a security perspective, combining both hardware and software security would be most secure. This does however come with some complications.

The downside of using hardware-based security is that it creates a dependency on the hardware vendor and the COTS devices that support that hardware. A collaboration is needed with a hardware vendor which might complicate the process of developing a payment service.

Software-based security – Software based security can exist out of a software library including bought or build security software or the use of the security functionalities that already exist in the operating systems software.
Using software security will result in a software system living on the COTS device that can be fully controlled by the PSP. It can be used on multiple devices and can be made independent of the device it runs on. This makes the application more flexible.

Having such a library living on the COTS does however have an impact on the systems performance because it requires additional coding, memory and updates. This could be unnecessary when the required functions are already available by the operating system.

IOS OS Security - According to the IOS security experts a big part of the responsibility of the application security lies within the operating system of the COTS device. Apple is known to be good in this area. When the application is downloaded from the app store it can be expected that the application is secure and that nobody can get their hands on the data it processes. The communication to the backend should be encrypted on multiple levels. How this is actually established are implementation details up to the developer.

It is possible to either use the cryptographic functions of the operating system or build them in a library. Since the cryptographic functions of the IOS OS are sufficient there are no real reasons to enlarge the application or SDK with additional self-made cryptography. Most cryptography algorithms are already supported by the current IOS devices.

Android OS Security – Similar to the IOS OS security the android OS offers cryptography and security. The expert interviews indicated that android has a less secure environment and therefore needs additional security measures. Especially when critical data such as PIN is entered additional software security such as White box encryption should be part of the solution.

5.6.6 Summary of the security decisions and effect on quality criteria

Due to the limitations of OS security, security experts state that solely using the OS security is likely not secure enough. Especially the Android security is not sufficient. The tradeoff is therefore not between hardware based, software based and OS based security but between hardware based and software based including OS system security. A solely software-based solution is also still an option but it would decrease the performance efficiency. A summary of the different design decisions and their effect on the determined quality criteria is shown in Table 5-3. Furthermore, this trade-off is not between the two architecture options but between their quality criteria. It is not necessary to only choose one. The security experts interviewed for this research state that the combination of all security measures would result in the highest possible level of security. This would however also result in a decrease in portability, performance efficiency and modifiability as can be seen in Table 5-3.

Table 5-3 Security design options and quality criteria

Architecture	Security	Performance efficiency	Portability	Maintainability
Hardware based	**+ Confidentiality** Hardware based security increases the confidentiality of key protection	**+ resource utilization** Resources that are used are already existent	**-- Adaptability** The app will only be able to run on a limited number of supporting devices	**- Modifiability and testability** Hardware based security is outside of the control of the PSP
Software based		**- Resource utilization.** The use of software-based security will increase the need for storage and the use of libraries.	**++ Adaptability** This security can be used on every COTS device	**+ Modifiability and testability** A fully software-based solution can be in full control of its owner
Software based OS IOS		**+ resource utilization** Resources that are used are already existent	**+- Adaptability** The security will only work on the most used OS versions.	**- Modifiability and testability** There is a dependency on the OS for modifying security
Software based OS Android	**- Integrity** Androids open nature is not secure enough for PCI compliance	**+ resource utilization** Resources that are used are already existent	**+- Adaptability** The security will only work on the most used OS versions.	**- Modifiability and testability** There is a dependency on the OS for modifying security

5.7 Application

The application of the mPOS terminal is the entire software solution that processes all the data input and provides a User Interface (UI). The application can either be a full application or a Software Development Kit (SDK) which can be used by the merchant to build a full application. This chapter will focus mainly on the Controller part of the application. The distinction between using an SDK or an application is not considered part of the scope of this thesis. Therefore, the relevant application components that will be considered in this chapter are the Controller and Driver. More background information on these components can be found in paragraph 3.5.

Table 5-4 Application experts

EXPERTISE	EXPERT	ROLE	EXPERIENCE	METHOD
IOS DEVELOPMENT	1	IOS app developer/ team lead	10+ years of work on IOS SDK's, JAVA development and Tech lead	Semi-structured interview and expert panel
POS PAYMENTS	2	Product manager / Team lead	10+ years of work on payment flows	Multiple informal interviews
C DEVELOPMENT / KERNELS	3	Senior C developer	10+ years of work on C development and payment architectures	Semi-structured interview, expert panels and multiple informal interviews
IOS DEVELOPMENT	4	Senior developer IOS	5+ years	Informal interviews and expert panel
ANDROID DEVELOPMENT	5	Senior developer/ Tech lead	5+ years of work on app development, Android SDK's and the full Android terminal solution	Semi-structured interview and expert panel
JAVA DEVELOPMENT	6	Senior Java developer/ Tech lead	8+ years of work on Java development and responsible for SCRP architectures	Semi-structured interview and expert panel
POS INTERFACES	7	Product manager/ Team lead	3+ years	Multiple informal interviews

5.7.1 Controller and Drivers

To argument for changing the architectural location of the Controller and the driver some background information is shown in paragraph 3.5. Because of the complex nature of these components only the information of the technical components relevant for this research have been explained. Additional information on these technologies can be found in the provided sources.

Moving from a terminal towards a COTS device means that the software used for communication with the hardware (Drivers) and the software for the business logic (Controller) will be moved to an application. For the SPOC solution the driver will then be part of both the PSP SDK/App and the SCRP software. For the CPOC solution the driver will be living in the PSP SDK/App. Moving the business logic from a classical hardware terminal to a software solution leaves two options. Moving the Controller into the application or moving it to the server. Various experts pointed out that having the business logic living in the cloud will increase latency significantly and therefore would not be feasible. Therefore, this architectural option will not be considered. An exception can be made by moving the business logic to the server in a modular and rule-based approach. Therefore, the modular approach will be further discussed in this paragraph. This approach is visualized in Figure 5-5. The Modular rule-based approach is further explained in Chapter 3.5.

The distance that the signals have to cover inside of the application is neglectable and therefore communication within the application has a low effect on the latency. Moving the Controller to a location other than the application would result in an increase in latency due to the number of signals necessary between the Controller and the drivers. These signals will then no longer happen within the application but will happen between a server and the application.

The terminal architecture can have a significant impact on the number of signals that need to be send and their distance. Therefore, the architecture decisions will impact the transaction speed or communication latency.

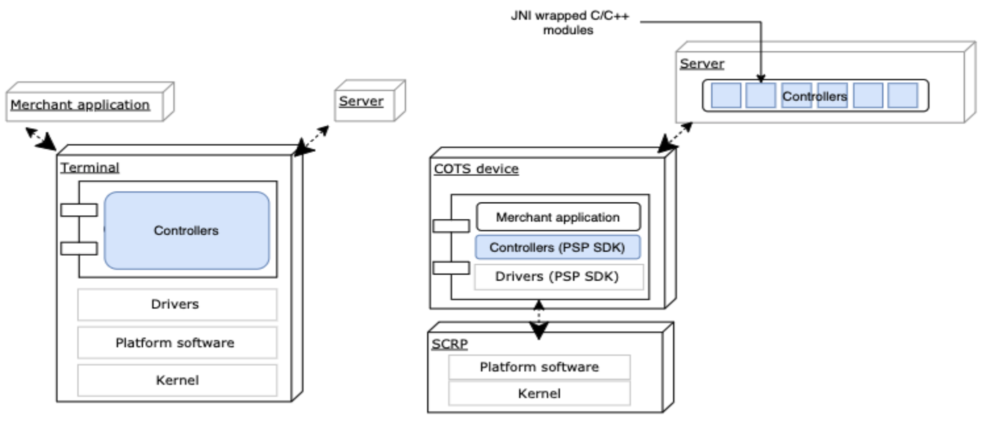


Figure 5-5 Modularized Controller design choice

5.7.2 Option 1: Controller in the application

For both the CPOC and the SPOC solution having the Controller in the application would mean that little has changed in comparison with the classical terminal. The Controller will live in the PSP SDK/Application and applies business logic to the input gotten from the drivers via the COTS device hardware. The difference between CPOC and SPOC in this scenario is that the driver in SPOC will communicate with the SCRP as is shown in Figure 5-5. The driver for the CPOC solution will communicate with the NFC chip reader. The Controller will remain the same.

Having the Controller living in the application will result in a similar transaction flow as the classical terminal. Therefore, no additional latency will occur. An over simplified example of such a transaction flow is visualized in the transaction flow diagram shown in Figure 5-6. The simplistic authorization flow will consist of the following steps:

1. The transaction will be started by the Controller
2. The card data will be inserted
3. The Controller applies the business logic. Dependent on this logic more communication with the card or the shopper is necessary to gather data on the applications, currencies, PIN etc.
4. Data and instructions go back and forth between the driver and Controller until all relevant business logic is applied
5. A HTTP connection is opened and the transaction is processed over the server

As indicated in Figure 5-6, the majority of the communication is between the driver and the Controller. The Controller start the transaction and depended on the business logic built into the Controller a number of signals will be send back and forward between the driver and the Controller. At the end of the transaction an authorized request will be send to the server via a HTTP connection. This request will receive a response indicating whether the transaction was successful or not. The single moment of communication between the Controller and the server allows for a high transaction speed.

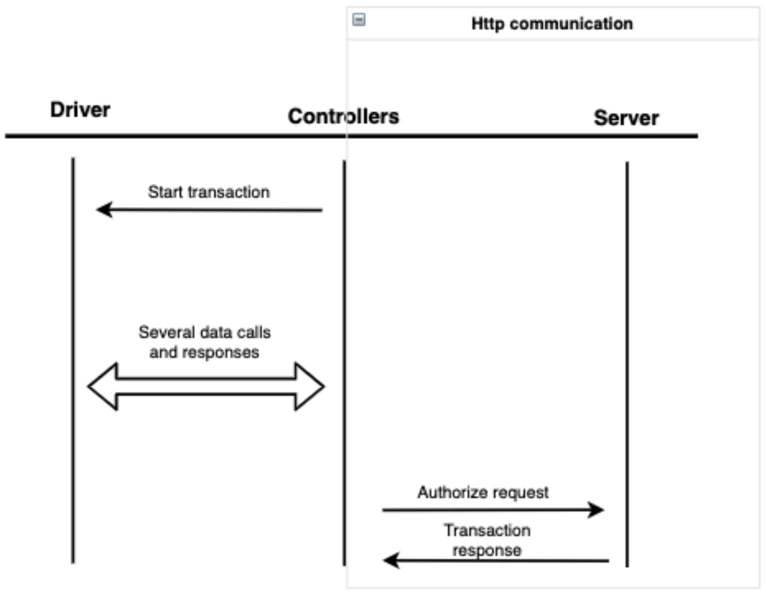


Figure 5-6 Controller – driver transaction flow classical terminal

5.7.3 Option 2: Moving Controller to the server

Moving the Controller to the server will result in a significant increase of communication between the server and the application or SDK. Moving the Controller to the server was therefore not considered feasible. The expert interviews and informal conversations with payment service vendors have identified some architectural innovations that when combined could compensate for the latency created by the increase of communication between Controller and servers.

Firstly, the *Websocket protocol,* as explained in the background paragraph 3.5, the Websocket protocol will increase communication speed in comparison with the widely used HTTP communication.

Secondly, *Modularity,* by making the Controller modular it would be possible to move parts of the business logic to the server dependent on the availability of certain transaction speed variables such as internet speed and server location. This allows architects to build a hybrid Controller in which business logic can be moved to the application for faster transaction speeds.

Thirdly, *the rule-based engine*
Expert brainstorming has resulted in an optional architecture that could allow for moving Controller to the servers by using a rule-based engine. As explained in the background paragraph of this chapter, using a rule-based engine will significantly decrease the number of signals that need to be send between the driver and the Controller.

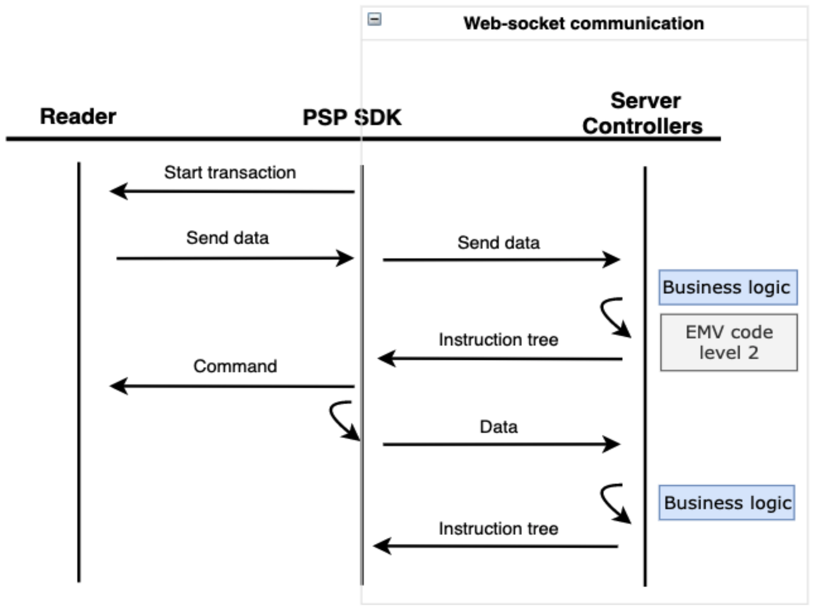


Figure 5-7 Controller – driver transaction flow option 2

Figure 5-7, gives a simplistic overview of the transaction flow diagram resulting from option 2, moving Controller to the server. This architecture allows for the following generic flow:

1. The transaction is started
2. Card data is inserted
3. The Websocket connection is already opened and allows for direct communication of the data
4. The instruction tree consisting out of all rule scenarios is send to the applications rule engine
5. The rule engine applies the decision tree (business logic) on the data.
6. Dependent on the business logic the application will request additional PIN entry or manual application selection
7. The entered date is sent through the open WebSocket connection
8. If necessary a second instruction tree is sent.
9. The payment is authorized or declined

As can be seen in Figure 5-7 the number of signals still increases in comparison with the authorization flow shown in Image 16 This will be compensated by the use of websocket communication but it will still result in latency. Furthermore, the amount of data send over the signals will also increase and therefore have an impact on its latency.

5.7.4 Tradeoff

The Controller in the application – The default option of keeping the Controller in the application will result in a faster transaction speed due to small distance to the devices native components. Furthermore, the connection between the Controller and the driver has no requirement for an internet connection and therefore does not require the device to be online. Offline functionality is therefore still a possibility.

The Controller in the server – Moving all business logic from the Controller in the application to the server will reduce the likelihood that a third party will be able to access the business logic code. This increases the confidentiality of the solution. Furthermore, the portability of the solution will increase significantly on the sub aspects adaptability, install ability and replaceability. The application can then be adapted for different hardware and software environment since it is independent of its nature and can connect to abstractions and API's. The install ability of the application will increase because the Controller no longer needs to be installed. Instead it has a websocket connection to the application. The replaceability will increase as well since the modularized nature of the new Controller allows for easy replacements of components as well as the entire Controller. Components can be replaced and reconnected to the application by an API.

Moreover, the maintainability will be improved due to the modularity and the location of the business logic. The server is in full control of the PSP and therefore the components can be easily replaced and modified without the need for an update. Whereas components living in the application have a dependency on updates initiated by its user.

As mentioned before, moving components to the server will have effect on the transaction speed. However, the experts interviewed for this research agree that the increase in latency will not be significant and therefore the transaction speed will most likely be within the required limitations set by EMV protocols for internet speeds in Western Europe. Furthermore, the Websocket connection is reliant on a fully connected terminal. The offline functionality will therefore no longer be possible by using this architecture option.

5.7.5 Summary of the Controller design options and their effect on quality criteria

According to the expert interviews conducted for this chapter three design options will fit within the technical boundaries of this research. A component-based Controller enables the architect to move the Controller partly to the server. Which sub-components are then used in the server and which are used in the application can then be made dependent on the latency that occurs. Therefore, the option of a hybrid Controller is added. Table 5-5 clearly indicates the effects that will take place as a consequence of these design decisions. The quantity of these effects is dependent on the environment in which the architecture operates. Therefore, the hybrid Controller is deemed a flexible solution that can be adjusted to a variety of situations.

Table 5-5 Application design options

Architecture	Application based Controller	Server based Controller	Hybrid Controller
Security		++ **confidentiality** The business logic will not be accessible to third parties	+ **confidentiality** The business logic will not be accessible to third parties
Performance efficiency	-- **Resource utilization** More code and data will live in the application	-- **Time behavior** The transaction speed will decrease	- **Time behavior** The transaction speed will decrease slightly
Portability		++ **Adaptability, Install ability, replaceability** The Controller will be fully portable and can be used independent on the hardware or software of the device.	++ **Adaptability, Install ability, replaceability** The Controller will be fully portable and can be used independent on the hardware or software of the device.
Maintainability		++ **Modularity** Components can be easily replaced ++ **Modifiability** The app can be easily modified in a scalable manner ++ **Analyzability** All business logic data will live in the server and can directly be analyzed	+ **Modularity** Components can be easily replaced + **Modifiability** The app can be easily modified in a scalable manner + **Analyzability** All business logic data will live in the server and can directly be analyzed
Functional suitability		-- **Functional completeness** The solution will no longer provide offline payments functionality	-- **Functional completeness** The solution will no longer provide offline payments functionality

5.8 Kernel

This paragraph will elaborate on the design decisions that have become available due to the move from hardware-based terminals to software-based terminals with regards to its Kernel component. Additional background information on the Kernel can be found in Chapter 3.5. The experts that have provided input in the form of design options and validations are shown in table 5-6. These experts have a specific expertise on Kernels. The experts mentioned in Chapter 4 have given more general input on Kernels from the perspective on their expertise, i.e. a security perspective.

Table 5-6 Kernel experts

EXPERTISE	EXPERT	ROLE	EXPERIENCE	METHOD
C DEVELOPMENT / KERNELS	A	Senior C developer	10+ years of work on C development and payment architectures	Semi-structured interview, expert panels and multiple informal interviews
JAVA DEVELOPMENT	B	Senior Java developer/ Tech lead	8+ years of work on Java development and responsible for SCRP architectures	Semi-structured interview and expert panel

There are three different Kernel levels. Kernel level 1 indicates the native software directly connected to the hardware (the bits and bytes). Level 3 considers the payment application and level 2 contains the logic to retrieve card data. In this and in most other literature, when talked about the Kernel this means level 2 Kernels. In Figure 5-8, the application component is similar to Level 3 and the reader consists of the earlier mentioned driver and the Kernel level 2.

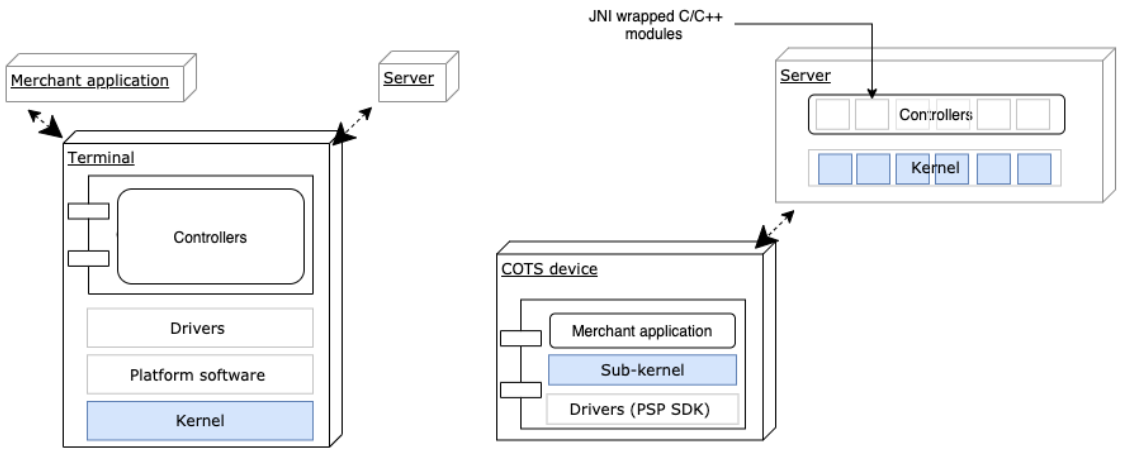


Figure 5-8 Kernel design options

The currently used EMV level 2 Kernel applications for both contact and contactless payments live inside the POS hardware. Moving the Kernel outside of the hardware would result in benefits as well as some challenges. In the older terminals no heavy business logic was being stored causing the POS based terminals to work well. However, with the increase of payment methods the number of L2 Kernels is increasing. Furthermore, new requirements such as the U.S. Common debit AID requirement are being implemented making it more and more complex to manage the various processes on the device. Terminal vendors have been reluctant to moving the Kernel to the cloud until recent times due to three factors: network latency, security, and the uncertainty of the level 2 Kernel certification process. However, recently the concept of a cloud Kernel is being revisited due to new technological innovations and network improvements. Various pilot projects are currently running to demonstrate a proof of concept on the cloud Kernel (Yazmaci, 2021).

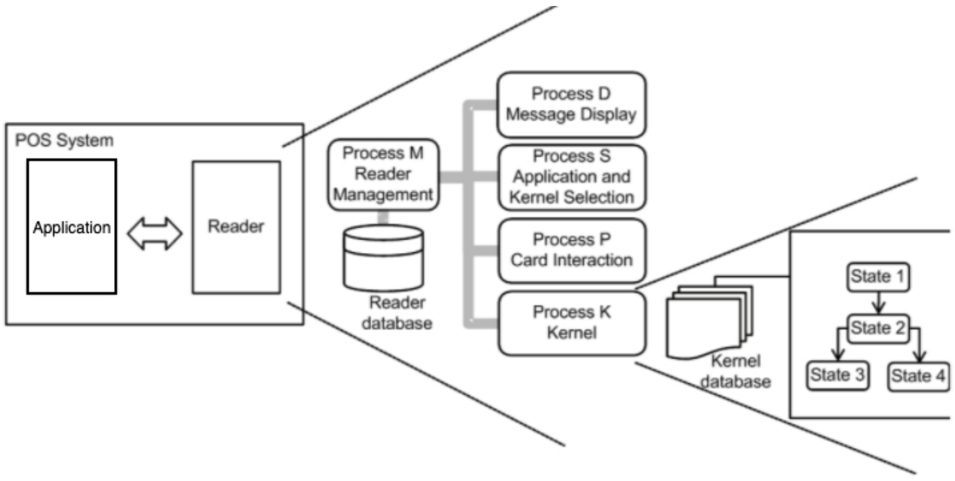

POS System
Application
Reader
Process M Reader Management
Reader database
Process D Message Display
Process S Application and Kernel Selection
Process P Card Interaction
Process K Kernel
Kernel database
State 1
State 2
State 3
State 4

Figure 5-9 Kernels detailed architecture

The Kernel process shown in Figure 5-9 is explained in Chapter 3.5. For explanatory purposes Figure 5-9 is also shown here. According to the EMV Kernel 2 specifications (EMVCO Book C-2, 2021), there are roughly three types of POS system architectures:

- *A fully integrated terminal:* All functionality in one single device
- *An intelligent Reader:* The Reader handles the transaction processing and passes the result on to the Terminal Application for completion.
- *A hybrid variant:* This combines the terminal Application and a transparent reader. The Reader provides card communication and the other processes including the Kernel take place in the Terminal.

According to the Kernel experts mentioned in Table 5-6, these options can be extended. Adding one additional option:

- *The cloud Kernel:* This combines the terminal Application, a transparent reader and the cloud. The reader only provides raw card communications and the other processes including the Kernel take place in the cloud. The reader will then only provide a gateway for communication.

5.8.1 CPOC versus SPOC

As the classical terminal uses the fully integrated terminal in which all functionality lives in one single device the future solutions use different architectures. The SPOC solution uses the SCRP device to retrieve card data. The majority of the transaction processing is handled in the SCRP itself. Therefore, the SCRP has its own software including a Kernel to process the card data. In the EMV Kernel protocol this would be called the intelligent reader.

The CPOC enabled device does not have a separate reader. The CPOC solution combines the payment application with the NFC reader of the COTS device. Therefore, this would be called the hybrid variant in the EMV Kernel protocol. In this variant the NFC reader provides card communication and all other processing including the Kernel is being handled by the terminal application.

The three type of POS systems architectures do not allow for all mPOS innovations. The hybrid variant is only suitable for CPOC, the intelligent reader for SPOC and the fully integrated terminal for a classic terminal.

5.8.2 The cloud Kernels

In Figure 5-9 the different processes that live in the level 2 Kernel are shown. Due to the hardware dependency of the Kernel it is not possible to move the entire Kernel to the server. Therefore, it is important to divide the Kernel is processes so it can be identified which processes can be separated and moved to the server. For this reason, additional background information and a simplistic view of the Kernel processes has been explained in paragraph 3.5.5.

As indicated in the short explanation on the Kernel processes the process K Kernel processing is responsible for the communication to and from process D, S and P. Due to the high number of signals being send between these processes their location is very important. Moving these processes could cause additional latency.

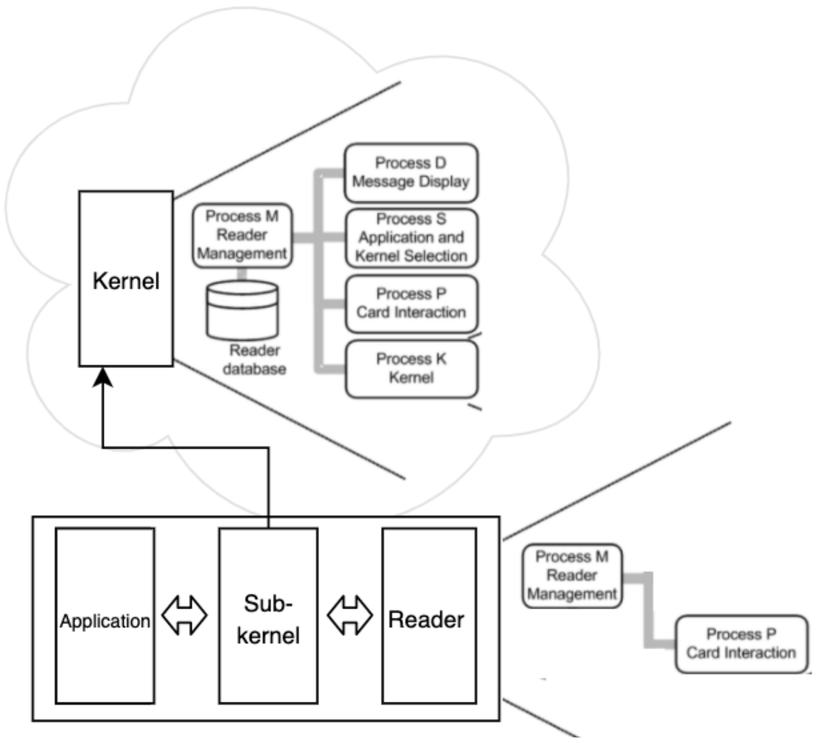

Kernel
Process M Reader Management
Reader database
Process D Message Display
Process S Application and Kernel Selection
Process P Card Interaction
Process K Kernel
Application
Sub-kernel
Reader
Process M Reader Management
Process P Card Interaction

Figure 5-10 Cloud Kernel

Figure 5-14 visualizes the potential architecture of a cloud Kernel. Since it is not possible to remove the entire Kernel a sub Kernel should be used to facilitate the communication between the cloud Kernel, the terminal application and the reader. The sub-Kernel performs EMV entry point processing and it coordinates the flow of the transaction between the terminal application, card reader and the L2 cloud Kernel. It exposes APIs and callback functions to the terminal application to drive EMV payment transactions. It gets transaction amounts and other transaction-related data from the POS application, it communicates with the card reader to get the EMV card data and sends all this information to the cloud to execute EMV transaction steps. At the end of the transaction, it gives the authorization data to the terminal application to go online for authorization and receipt printing.

Process M initiates the Kernel and also provides information on the availability of the terminal antenna. Furthermore, Process P needs to directly communicate with the hardware components of the terminal reader and therefore these components partly remain in the physical terminal.

5.8.3 The effects of moving the Kernels to the server

To visualize the effect of these architecture changes a simplistic transaction flow is shown in Figure 5-15. The flow cab be explained as follows:

1. The terminal application gets the transaction amount, type and initiates the transaction
2. The application sends the acquirer name, EMV configuration ID and transaction parameters. The sub-Kernel sends these to the cloud.
3. The Kernel then sends the AID's from the reader database and sends this to the Sub-Kernel
4. Application selection is then handled according to configuration
5. Send get processing options response
6. The application protocol data unit (APDU) is send. This is the communication unit between a smart card reader and a smart card
7. Cardholder verification method (CVM) (i.e. PIN) parameters are send and responded.
8. Complete transaction

The communication shown between the Sub-Kernel and the Kernel is handled between a server and the application in Figure 5-15. In the classical terminal variants as mentioned in paragraph 5.8 this communication would remain within the application.

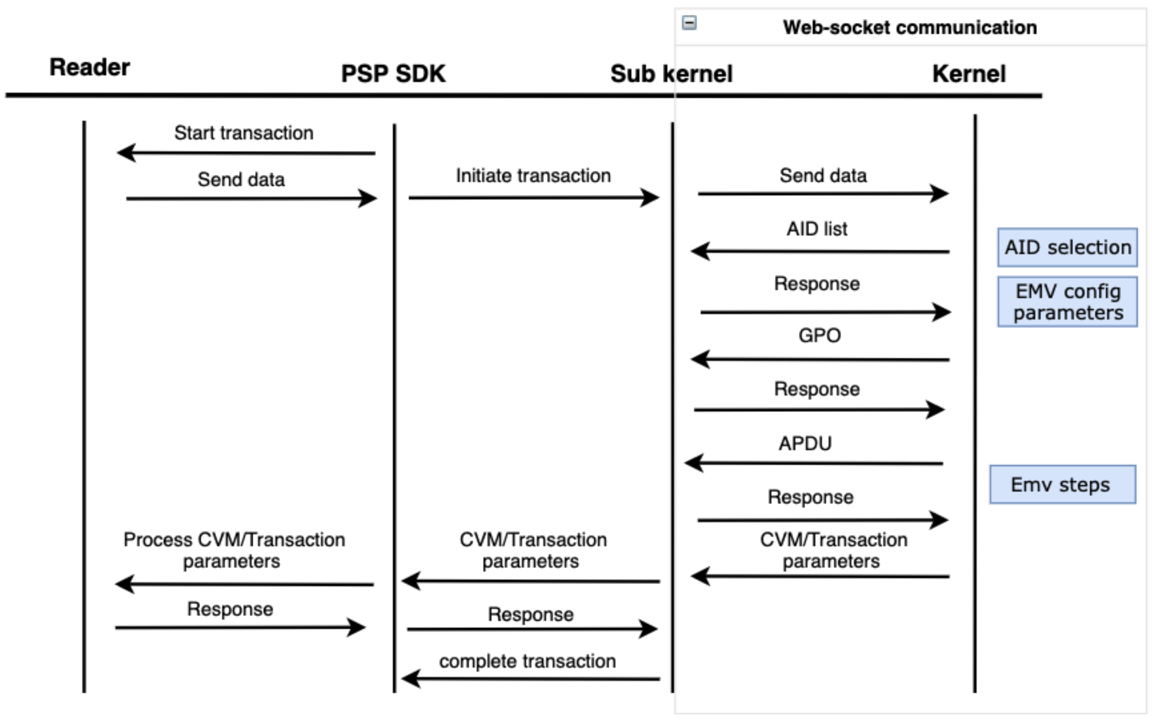


Figure 5-11 Cloud Kernel transaction flow

The communication between the sub-Kernel and Kernel can increase for more complex payment scenarios. The main responsibility of the Kernel is to send and receive data. Therefore, moving the Kernel partly while implementing a sub-Kernel will have a significant effect on the number of signals send and received by the server. Although the use of a Web-socket communication will compensate the increase in latency partly, the expert interviews as well as technological literature indicate that increasing the server communication by moving the Kernel to the cloud would likely still increase latency (Oliveira *et al*., 2018)

5.8.4 Tradeoff

The tradeoff will be simplified by grouping the classical Kernel variants as one option. For the scope of this thesis these can be grouped because as explained earlier in this chapter, all three classical Kernel variants are necessary to provide for all considered mPOS innovations.

The fully integrated terminal, intelligent reader and the hybrid variant – All three original Kernel structures are necessary to provide an architecture for the current as well as the future (m)POS innovations. Therefore, these will all have to be supported by the various (m)POS solution. This means that these solutions have a significantly lower portability in comparison with the second hybrid variant.

Furthermore, the connection between the Controller and the driver has no requirement for an internet connection and therefore does not require the device to be online. Offline functionality is therefore still a possibility.

The cloud Kernel – The server-based Kernel will not allow for offline transactions. This has a negative effect on the functional completeness and also makes the system less reliable. Once the network is interrupted it will be more difficult for the server system to recover the transaction.

As mentioned earlier in this chapter, the cloud Kernel will likely cause more latency due to the increase of server communication. However, some (technological) innovations have the opportunity to compensate this latency:

Firstly, the *Websocket protocol,* as explained in Chapter ,3 the Websocket protocol will increase communication speed in comparison with the widely used HTTP communication.

Secondly, the increase of contactless payments. According to research by De Nederlansche bank, Dutch consumers used contactless payments for 67% of their transactions in 2020. This number is still increasing in a fast pace partly due to Covid-19 (Jonker, 2021). The number of signals that are send during a contactless payment is smaller in comparison with other entry methods because a contactless payment is only allowed to take 500 milliseconds (EMVCO Book C-2, 2021). Therefore, the use of contactless payments will decrease the number of signals that have to be send ant with that the systems latency.

Thirdly, the NFC reader used for a CPOC solution already decreases the transaction speed significantly. A waver has been given by the EMVCO allowing CPOC transactions to have a longer duration than is normally accepted by EMVO protocol. The decrease in transaction speed as a consequence of the COTS hardware increases the amount of time that can be used for Kernel communication.

Moving the Kernel to the cloud means that the solution certification will no longer be necessary for each device. By moving the Kernel and the transaction processing to the server the majority of the solution certification will only have to be done once. Improves the testability of the system and saves time and costs because it will no longer be necessary to certify each separate device. Furthermore, the level 2 Kernel will no longer be tied to the hardware or operating system platform and will therefore be able to support different devices with a limited effort.

Moreover, configuration files no longer need to be pushed to the terminals. Changing parameters in the server will directly cause an effect for all selected terminals. Also, the cloud Kernel can provide detailed transaction data and detailed error logs for failing transactions. This data can be stored in the database for monitoring and reporting. The cloud Kernel will also have an effect on security, because the Kernel lives in the cloud, no secure processor is needed limiting the amount of memory space on the COTS device. Lastly, according to the interviews with solution vendors the large payment schemes are now providing well defined certification processes allowing cloud Kernels to be used for the same payment method schemes as regular L2 Kernels.

5.8.1 Summary of the Kernels design options and their effect on quality criteria

There are four design options possible for the Kernel architecture. However, the three classical design options all need to be combined to provide for the mPOS solutions considered in this research. The newly proposed cloud Kernel can provide for all solutions and therefore the three classical options are combinedly compared with the cloud-based Kernel. The main difference is the amount of communication necessary for a cloud-based Kernel between the server and the sub-Kernel living in the payment application. The effects of the Kernel design choices on quality criteria derived from the expert interviews are visualized in table 5-8.

Table 5-7 Kernel Design Choices

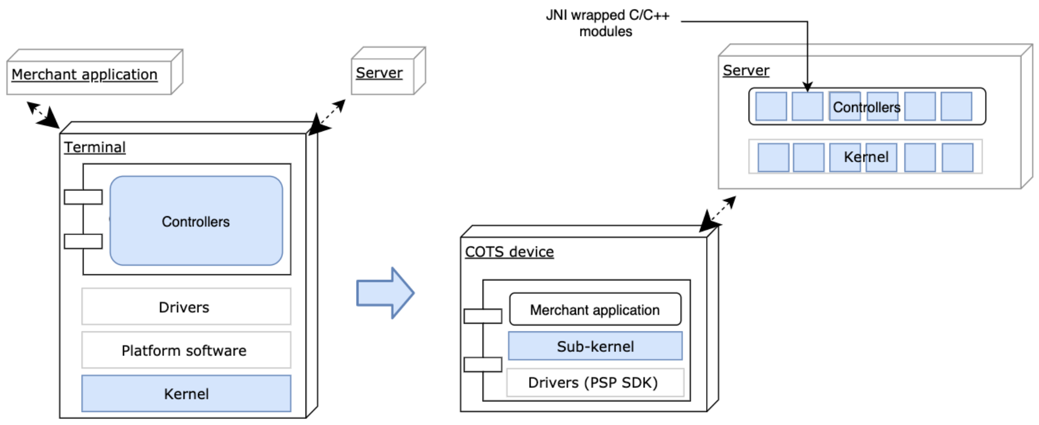
Architecture	Terminal Based terminal	Server based terminal
Security		++ **Integrity** The users of the application will not be able to access the Kernel code.
Performance efficiency		-- **Time behavior** Transaction time will increase
Portability		++ **Adaptability, Install ability, replaceability** The Controller will be fully portable and can be used independent on the hardware or software of the device.
Maintainability	-- **Modifiability** Updates need to be pushed to all COTS devices to modify the system	++ **modifiability** The system can be fully modified by changing parameters in the server ++ **Testability** The solution only lives in one location and therefore the software can be easily tested and certified.
Functional suitability	++ **Completeness** Allows for offline transactions	-- **Completeness** Does not allow for smaller schemes yet
Reliability		-- **Recoverability** In the event of an interruption it will be difficult to re-establish the transaction.

5.9 Combining the Kernel and Controller architecture

Although the research is visualized for simplicity as if all components work independently it should be considered that moving the components can have an effect on other components. This effect has been considered and is elaborated on in this paragraph.

As can be seen in the previous paragraphs, the different components that are being used by the Controller are intertwined with the Kernel components. The Kernel can be considered as a part of the driver which communicates with the hardware and applies the business logic from the Controller on the retrieved data.

Figure 5-12 Controller and Kernel synergy



As mentioned in the Controller paragraph, the Controller needs to communicate with the driver and therefore moving the Controller will increase the number of signals that need to be send to and from the server. The part of the driver that consists of the Kernel undergoes the same consequence. By moving it to the server as shown in figure 5-16, additional communication is necessary towards the terminal application and its Controller. Therefore, moving both components to the server will create a form of synergy. The communication that would occur between the Controller and the Kernel fully occurred within the COTS device for the classical Kernel and Controller' solution. The proposed solutions in paragraph 5.7 and 5.8 show an increase in communication between the device and the server. Moving both components to the server will result in moving these communication signals within the server. In conclusion, moving both components will increase the processing within the server and therefore decrease the number of signals that need to be send between the server and the application. This is illustrated in Figure 5-17.

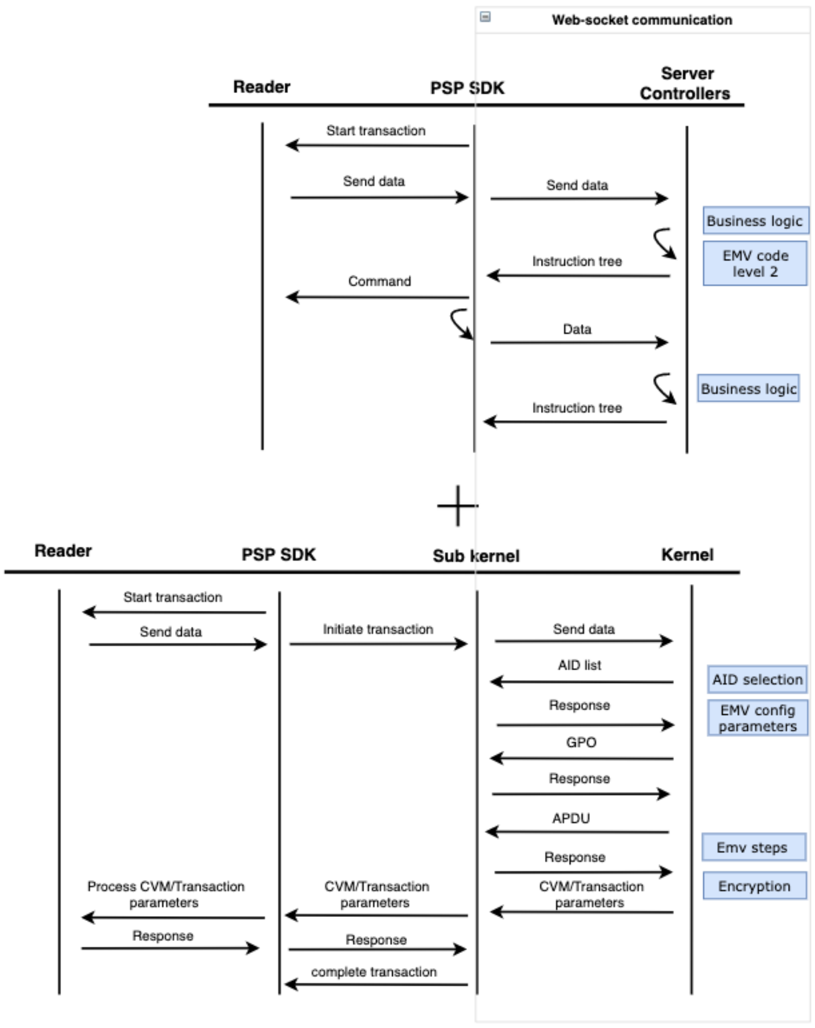


Figure 5-13 Controller and Kernel synergy transaction flow

In an addition to this, the Sub-Kernel that lives in the PSP SDK/Application which creates another opportunity to combine the signals that are send. By moving both the Kernel and the Controller to the server the different signals can be combined in one signal resulting in no increase in the number of signals at all.

This means that after moving the Kernel to the cloud latency will increase due to the increased amount of communication between the application and the server. However, consecutively moving the Controller to the cloud would not result into an increase of latency but will even decrease latency due to the communication between the Kernel and the Controller that will fully be moved to the server.

Combining the Controller and Kernel signals would however increase the amount of data send in the signals and therefore decrease the communication speed. According to the interviewed Java development and C++ development experts this would however cause an insignificant change in latency.

5.10 Combining tradeoffs into an architecture decision flow

This paragraph will explain how the architecture decision flow can be used to provide an overview of the to be made design decisions and the effect that these decisions will have. Firstly, the decision sequence will be explained. Secondly, the final artifact, the architecture decision flow is illustrated in paragraph 5.10.2. Thirdly, the changes necessary for a new mPOS innovation are summarized in paragraph 5.10.3, 5.10.4 and 5.10.5. Finally, a demonstration of an architecture design, made following the architecture decision flow, that is portable enough to fit all mPOS innovations is illustrated in the demonstration phase in chapter 6.

New entrants in the payment industry that build a new architecture will likely decide upon an architecture that is portable enough to fit all illustrated mPOS solutions without the need for expensive iterations to the software architecture. For PSP's that already use a fully hardware-based solution or a hybrid payment solution it is likely that a more gradual approach will be followed. In which case iterations to the payment architecture will be made dependent on the type of innovation that the PSP wants to adopt. The new cloud-based design options earlier explained in this chapter allow for the decision flow to provide for both scenarios.

The sequence of decisions consists of optional movements, obligatory movements and trade-offs. An obligatory movement is a movement initiated by the deprecation of a hardware component. This research only considers the software architecture and therefore these movements are outside of the scope of this thesis. However, for completeness they will be visualized in the decision flow. An optional movement is a component that can be moved for various reasons but doesn't affect any criteria such as the cash register application or Value-Added Services (VAS) applications. It is also possible to keep this component in the same location while switching to a new mPOS innovation. The trade-offs are the design decisions within the software architecture scope that have been explained in the former paragraphs.

5.10.1 Architecture decision sequence

The output of this thesis is an architecture design decision flow instead of one architecture in order to provide flexibility to its user. The design decisions can be made in any sequence. However, due to the various specifications of the different mPOS solutions there is a most logical flow. This flow moves from the hardware-based solution to the fully software-based solution in a gradual manner. The sequence of mPOS innovations in table 5-9 showcases the expected flow in which innovations will be implemented as well as the specification that have an impact on this sequence. The architecture decision flow diagram does not necessarily need a user to follow this chronology. The decisions are an indication and can be made in any order.

Table 5-8 Sequence of mPOS innovations and requirements

	OPTIONS			
	SCR No Pin	SCRP PIN (SPOC)	CPOC No Pin	CPOC PIN
PCI Certification	SCRP	SCRP + SPOC	CPOC	No certification yet
Works on Apple iOS	TRUE	TRUE	FALSE	FALSE
Works on Android	TRUE	TRUE	TRUE	TRUE
Library/Application Needed	TRUE	TRUE	TRUE	TRUE
Kernels on COTS	FALSE	FALSE	TRUE	TRUE
Monitoring & Attestation Requirement	TRUE	TRUE	TRUE	TRUE
Allows EMV	TRUE	TRUE	FALSE	FALSE
Allows MSR	TRUE	FALSE	FALSE	FALSE
Allows EMV Ctls	TRUE	TRUE	TRUE	TRUE
Allows for PIN Entry	FALSE	TRUE	FALSE	TRUE
SCRP Certified Device	TRUE	TRUE	FALSE	FALSE
Use of Hardware protection (TEE)	TRUE	TRUE	TRUE	TRUE

It is important to take into consideration that due to the synergy effect of combining a server-based Kernel with server-based Controller the order in which these will be implemented will have an effect on the quality criteria performance efficiency. If the Controller are moved to the server in an architecture that already has the Kernel living in the server no additional latency will occur. This is due to the structure explained in Figure 5-17. Due to the high number of signals send and received by the cloud Kernel the Controller will not create the need for additional communication signals. Furthermore, the processing between the Controller and the Kernel will happen within the server. This will decrease latency. If this process would be the other way around it would have an effect. However, moving the Kernel to the server in an architecture that already supports the Controller in the server will increase latency due to the high increase in the need for signals.

5.10.2 Design decision flow diagram

The decisions in this diagram are shown from left to right and are indicated in colors. Similar colors indicate that the colored component can live in either of those locations. The description of this flow is summarized in the following paragraphs.

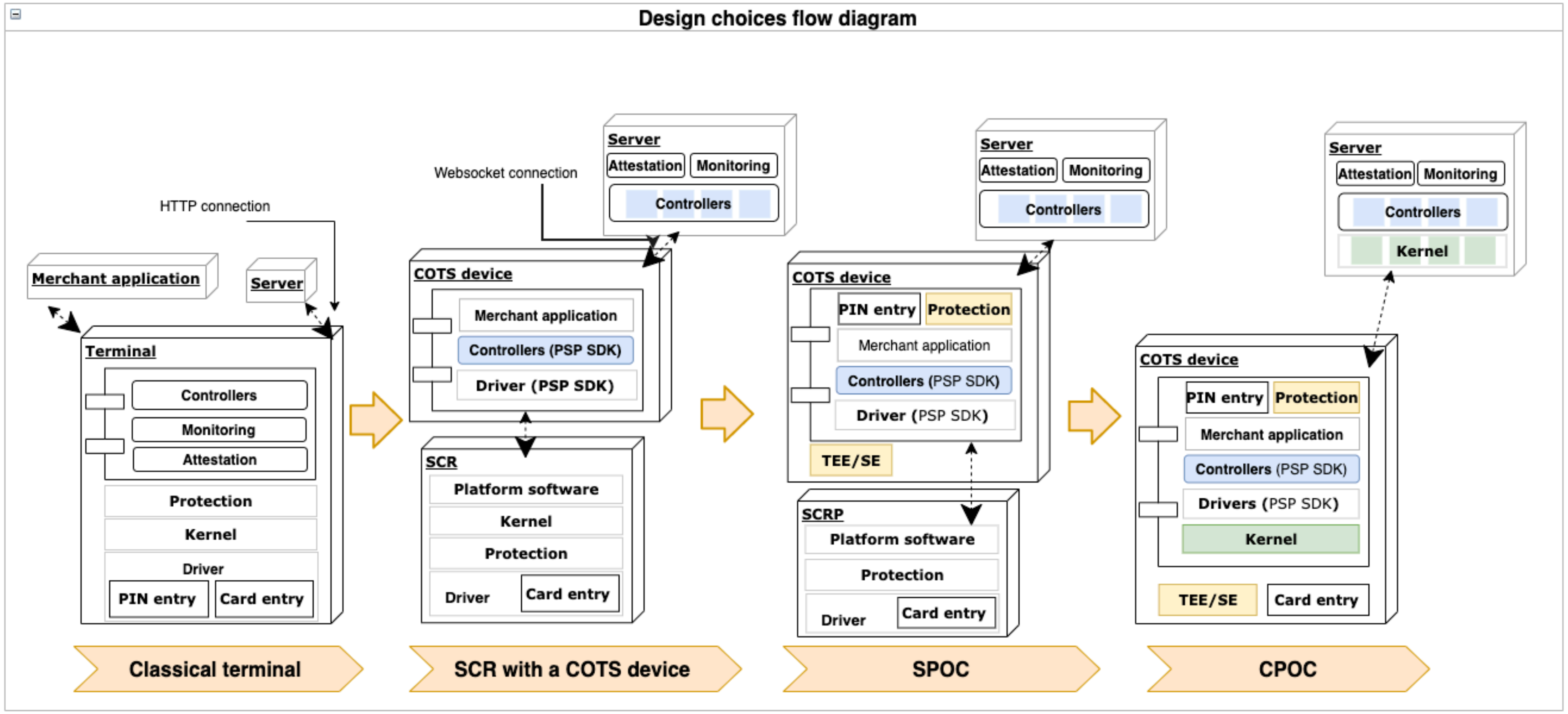


Figure 5-14 Design Choices Flow diagram

5.10.3 Classical terminal → SCR with a COTS device

Starting from a classical terminal the most logical mPOS solution to implement to existing architecture is the SCR with a COTS device. As shown in table 5-9 this solution requires no PIN on the COTS device and therefore less security is needed on the COTS device. Furthermore, the Kernel is embedded in the SCR hardware and therefore out of scope of the software architecture. More extensive information on this decision can be found in paragraph 5.7.

Action	Action item
Obligatory movement	Application • Driver to application SCR • Kernel to SCR • Protection to SCR • Driver to SCR Server • Attestation to Server • Monitoring to server
Tradeoff	1. Controller to application 2. Modularized Controller to server 3. Modularized Controller partly to server

Options	Option 1	Option 2	Option 3
Security		++ confidentiality	+ confidentiality
Performance efficiency	-- Resource utilization	-- Time behavior	- Time behavior
Portability		++ Adaptability ++ Install ability ++ replaceability	+ Adaptability + Install ability + replaceability
Maintainability		++ Modularity ++ Modifiability ++ Analyzability	++ Modularity + Modifiability + Analyzability
Functional suitability		-- Functional completeness	-- Functional completeness

Table 5-9 Effects of moving from the classical terminal to the SCR

Table 5-10 shows that in this first phase the Controller can be relocated. As mentioned earlier in this chapter the Controller can either stay in the application or be moved to the server. The consequences of this relocation is summarized in the bottom part of table 5-10.

The obligatory movements indicated in table 5-10 consists of hardware components and monitoring and attestation. As mentioned in Chapter 3, monitoring and attestation should live in the application because of PCI requirements.

5.10.4 SCR with a COTS device → SPOC

As can be seen in table 5-9, the SPOC solution allows for PIN on the COTS device. Therefore, additional protection is needed on the COTS device. Assuming that this new innovation is implemented after the implementation described in paragraph 5.10.3. the Controller decision should already have been made.

Table 5-11 summarized the necessary architectural movements and the security tradeoff that can be made. Although the design decision flow illustrated in figure 5-18 indicates that the merchant application lives in the COTS application, this is not an obligatory movement. The merchant application could still reside in different hardware. More extensive information on this decision can be found in paragraph 5.6.

Action	Action item
Obligatory movement	Application • Pin entry to application
Tradeoff	1. Hardware based security 2. Software and OS based security 3. Hardware and software-based security

Options	*Option 1*	*Option 2*	*Option 3*
Security	+ confidentiality		++ confidentiality
Performance efficiency	+ Resource utilization	-- Resource utilization	- Resource utilization
Portability	-- Adaptability	+ Adaptability	-- Adaptability
Maintainability	-- Modifiability -- Testability	+ Modifiability + Testability	-- Modifiability -- Testability

Table 5-10 Effects of moving from SCR to SPOC

5.10.5 SPOC → CPOC

The move from a SPOC solution to a CPOC solution is the final stage of moving from hardware to software. This allows the payment solution to deprecate the hardware-based card reader and rely fully on the software architecture. Therefore, the Kernel can now be relocated. The driver as well as the Card entry processing move to the COTS device. The COTS device will use its NFC reader to process the Card entry. Further processing will be done by the Kernel. The two tradeoffs mentioned in paragraph 5.8 is summarized in table 5-12.

Table 5-11 Quality criteria effects Kernel

Action	Action item
Obligatory movement	COTS device • Driver and Card entry to COTS device
Tradeoff	1. Application based Kernel 2. Server based Kernel

Options	Option 1	Option 2
Security		++Integrity
Performance efficiency		-- Time behavior
Portability		++Adaptability ++Install ability ++Replaceability
Maintainability	-- Modifiability	++Modifiability ++Testability
Functional suitability	+ Completeness	- Completeness
Reliability		-- Recoverability

5.11 Summary of Chapter 5

In the design and development phase, sub question 2 "*What are the stakeholder quality attributes for the most significant current and future mobile POS solutions?*" and sub question 3 *"What payment architecture fits the requirements derived from sub-question one and two?"* are answered.

an extensive research was conducted involving experts from all three stakeholder groups to answer sub question two. The requirements derived from the obtained data were combined with existing architecture validation literature to form a quality attributes utility tree. This utility tree visualizes the quality attributes that are important for the security, merchant and the payment industry stakeholder groups. This utility tree can be utilized for the analysis of payment architectures and architectural decisions and is shown in Figure 5-2.

For sub question 3, a number of multidisciplinary expert interviews revealed the possible design decisions that can be made within the boundaries of the requirements derived from sub question one and two. In this chapter it has been proven to be possible to design an architecture that allows for the current as well as the future mPOS solutions. To determine which payment architecture will fit the variety of requirements by the payment industry an architecture decision flow was developed that would enable a variety of users to build an architecture capable of adopting all earlier mentioned mPOS solutions.

The design decisions that followed from the design flow have been analyzed by determining their effect on the quality criteria. These quality criteria were obtained from the answers on sub question 2. The obtained artifact is visualized in Figure 05-18.
The architecture decision flow illustrates a sequence of decisions that are necessary to adopt the various mPOS solutions. These decisions can be made gradually by iterating an existing architecture and adopting mPOS innovations successively. The decisions can also be made at once, building one architecture that is portable enough to fit all current and future mPOS innovations without the need for expensive architecture iterations.

Designing a portable architecture has been made feasible by the identification of three architecture tradeoffs:

1. Application based Controller versus a modular server-based Controller

By rebuilding the Controller in a modularized manner and using rule-based coding, it becomes possible to efficiently move parts of the Controller to the server. To decrease latency a continues Websocket connection should provide the communication between the server and the application. This results in improved portability, maintainability and security of the mPOS application but dependent on network variables it could increase communication latency.

2. Using hardware-based security versus using software-based and operating system security

By using software-based security the application no longer is reliable on hardware provided by the smart device. Combining the software-based security with security provided by the operating system will result in an almost similar level of security. This will increase the maintainability and portability of the application.

3. Application based Kernel versus a server-based Kernel

Due to the use of Websocket communication, 5G internet and the decreased need for card communication due to contactless payments it becomes feasible to move the transaction Kernel to the server. This will improve the portability, security and maintainability of the application but will likely also increase latency.

Furthermore, moving the various components to the server simultaneously has proven to compensate for the increase of communication. A significant part of the communication between components can now happen directly in the server and communication between the device and the server can be combined. This reduces the added latency and makes the solution feasible in more scenarios.

The demonstration in Chapter 6 will show how the decision tool can be used to create an architecture that does not need iterating to fit to all innovations.

6 Demonstration and Evaluation

Qualitative research is vulnerable to subjectivity and bias because the researchers conducting it often have to use their perceptions and interpretations while drawing conclusions from the gathered or retrieved data. In the paper by Jan Pries-heje (2008), called 'Strategies for Design Science Research Evaluation' multiple DSR evaluation methodologies and strategies are evaluated and combined as a strategic DSR evaluation framework. The framework encompasses ex ante and ex post orientations as well as naturalistic and artificial settings for evaluating research. In this chapter, the process of demonstration and evaluation will be elaborated on. The demonstration and evaluation phase are part of phase 4 and 5 of the DSRM process. In paragraph 6.1 a demonstration will be shown that indicates how the decision flow can lead to a desired architecture. Secondly, in paragraph 6.2 the evaluation process is described which has been used to evaluate the research and its output. Lastly, the remainder of the chapter elaborates on the outcome of an expert panel evaluation that indicated the relevance of this research.

6.1 Demonstration of a fully portable architecture

To demonstrate the use and possibilities of the architecture decision flow developed in Chapter 5 and following the DSRM design phase, a simplified demonstration has been exemplified. Due to resource limitations it was not possible to do a full demonstration and implementation of the architecture. Therefore, a simplistic fictional demonstration has been drafted to demonstrate the use of the architecture decision flow to an expert panel for evaluative purposes.

In this demonstration first, a fictional scenario will be given to an architect that describes the preferred qualities and requirements of the PSP. Secondly, the design decisions will be made using the background information obtained in Chapter 5 and the action summaries shown in paragraph 5.10. The decisions will be made dependent on the scenario given and the mPOS innovations required. Thirdly, by using the quality attribute analysis, it can be seen if the desired effect was generated using these design decisions.

6.1.1 Demonstration scenario

The payment service provider is in the following scenario:

- The PSP operates solely in Western-Europe and only has 5G connected terminals. Therefore, Time behavior of the payment solution is often not an issue for this PSP.
- The PSP's main business was adding value added services to its terminals. Therefore, the hardware and the payment architecture were outsourced to other companies. The PSP would like to make a switch and build its own new architecture.
- The PSP wants to keep supporting its currently used classical terminals but also wants to extend its offer with a cheaper and more mobile variant. This variant wel be used as an extension of the classical terminals and therefore can have less

functionality. Because of the overflow of new mPOS innovations they have not yet decided which mPOS solution they want.

- In a later stage the PSP would like to be able to accept payments via an application on a smartphone independent of the smartphone type or brand. However, this is not yet possible because it does not yet accept PIN.
- The PSP has high security standards but prefers to have the option to directly update their terminal fleet in the case of a security issue above the use of superior security hardware.

6.1.2 Demonstrated architecture

The simplistic scenario in paragraph 6.2.1 exemplifies a situation that can often be seen in the payment industry. There is uncertainty on which mPOS solution should be adopted. However, they operate in Western-Europe which has a consistent internet connection. Furthermore, they require a solution that has high portability to fit every smartphone type and high maintainability to be able to update directly in the case of a security issue.

Because the PSP does not know yet which mPOS solution it will use in the future it would be beneficial to have an architecture that does not need additional iterations to fit on new solutions. Furthermore, it should be easily adaptable to allow them to directly enroll their clients to a CPOC solution in the future. These mentioned PSP requirements are indicated in table 6-1 for comparison.

By comparing the required criteria with the design decisions and their effect on quality criteria in paragraph 5.10 the following decisions were made in the tradeoffs:

1. Software-based security, this will allow the PSP to be independent of smartphone hardware for future references
2. Server-based Kernel, this allows the PSP to keep control and directly maintain its solution from their server. Furthermore, this will increase the portability and maintainability of their solution.
3. Server-based Controller, Since the Kernel is already in the server this will not have any negative effect by increasing latency. Furthermore, by adding the Controller to the server there is now no longer a dependency on hardware. This allows the PSP to use the exact same architecture for all its solutions. Therefore, the PSP is no longer under pressure to make a final decision on which mPOS solution it wants to adopt and it is ready for the future. This architecture is visualized in Figure 6-1.

The effect of these decisions that can be found in Chapter 5 have been combined in table 6-1. The table indicates that the quality criteria fit well with the requirements from the PSP.

Table 6-1 Quality criteria effects Demonstration

Criteria	Flexible "One-size-fits-all" Architecture	Required by PSP
Security	-- Integrity	+- Security
Performance efficiency	-- Time behavior	+- Time behavior
Portability	++Adaptability ++Install ability ++Replaceability	++ Portability
Maintainability	++Modifiability ++Testability	++ Maintainability
Functional suitability	- Completeness	- Completeness
Reliability	-- Recoverability	N/O

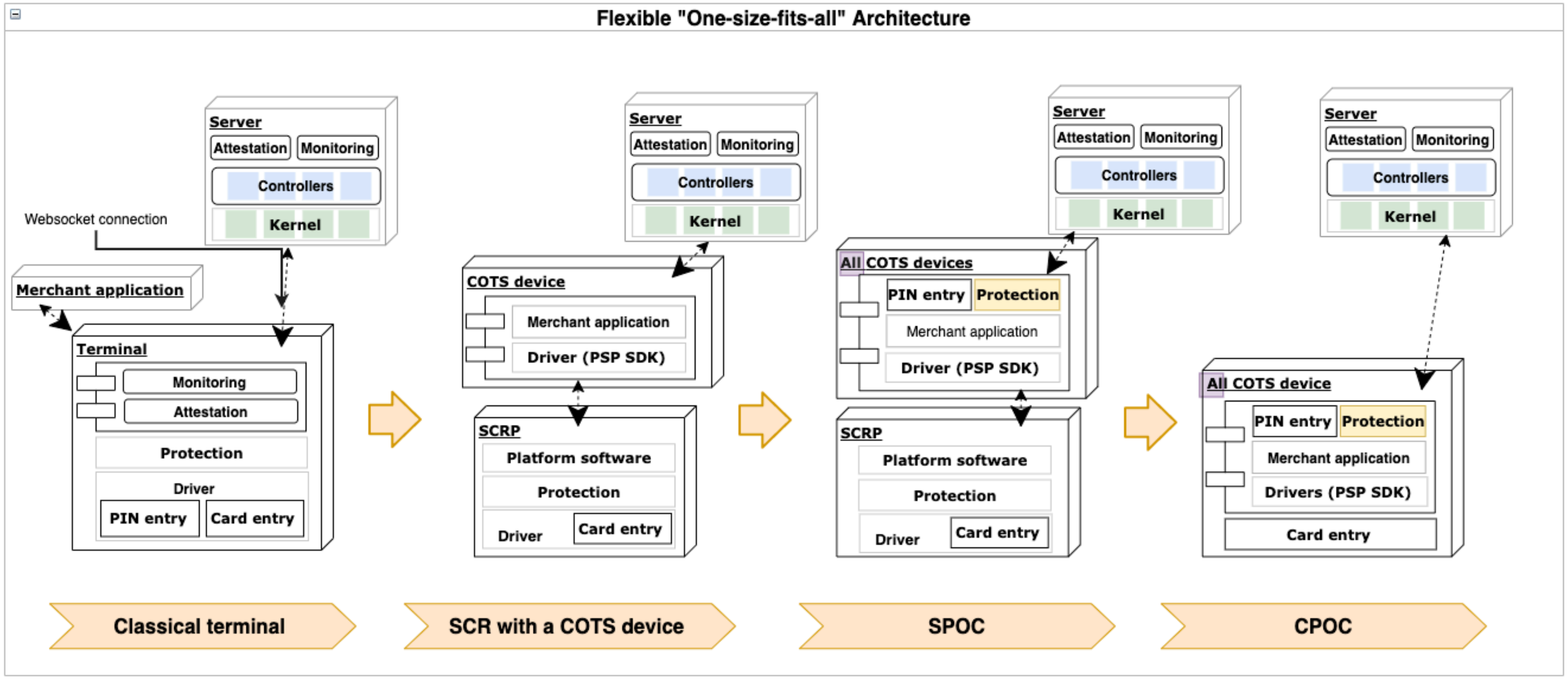


Figure 6-1 Flexible Architecture

6.2 Evaluation process

The research by Jan Pries-heje (2008), states that before conducting the evaluation it is important to establish *What* is actually evaluated, *how* it will be evaluated and *When* it should be evaluated.

The *What* is important to establish whether the design artifact is a product or a process. A product can be evaluated using a quality model, whereas a process should be evaluated on its process-based quality. Whether this research is focused on a product or a process is rather difficult to establish. The architecture decision tree in itself is a developed process but the decisions that are proposed can be considered products in the form of an architecture component. These components can be evaluated with the use of a quality model such as the in this research used ISO 25010 derived quality model that is shown in Chapter 4 of this research. Due to the continues architecture approach the proposed architecture components have continuously been validated on technological feasibility and compliance. Therefore, the evaluation will focus on validating the process outcome.

The *How* determines whether the process will be evaluated naturalistically or artificially. Due to time constraints it is not possible to allow researchers and payment service providers to use this research to help them determine their preferred architecture. Therefore, an artificially and qualitative approach will be used in which a simple case study will be conducted and evaluated by experts. The evaluation is therefore being done before any artifact was developed and can be considered ex ante.

Commonly used techniques for validation are:
The checklist-driven validation – In which a set of questions is prepared and proposed to an expert panel by the evaluation facilitator.
Scenario-based validation - In which an example scenario describes interactions between the user of the application system and the system itself to provide an example of the process.
Decision-centric validation – This technique allows the review panel to review analyze and record the rationale behind the design and design decisions. These decisions are evaluated against quality attributes (Erder & Pureur, 2016).

To establish that all aspects are evaluated, the most wholesome approach is to use all three techniques. The decision-centric validation has been conducted by following the continues architecture approach by gathering continues validation of expert via informal conversations. The scenario-based validation and the checklist driven validation will be combined in the expert panel by presenting a demonstration scenario while validating the design objectives with the use of a question-based checklist.

6.3 Expert panel

As mentioned in the evaluation process an expert panel will be conducted to evaluate this research. According to scientific literature on the validity of qualitative research, the validity of research corresponds to the degree to which it is accepted as sound, legitimate and authoritative by people with an interest in research findings (Yardley, Clarke, Braun, & Hayfield, 2015). This expert panel will be used to convey expert knowledge on the validity of this research and to determine whether the research objectives have been reached successfully. Furthermore, the goal of this expert panel is to identify limitations and provide recommendations for this research design.

The experts have been chosen using similar methodology as explained in Chapter 4.2 on choosing participants. The experts used for the evaluation should comply with the following criteria:

- The panel participants should have different background to guarantee a broad perspective of evaluation.
- The panel participants should be in a position that would require them to be part of payment architecture decision making.
- The panel participant should have knowledge of and preferably experience with qualitative research

In total four experts have been chosen and were willing to conduct the expert panel evaluation. Due to time constraints two of the experts were unable to join the collective meeting and were therefore interviewed separately. To reach a consensus the participants have acknowledged and agreed with the output of the separated meetings.

Table 6-2 Evaluation expert panel

Expertise	Expert	Role	Experience
C DEVELOPMENT/ Architecture design	A	Senior C developer/ Tech lead	10+ years
C DEVELOPMENT / KERNELS	B	Senior C developer	8+ years
JAVA DEVELOPMENT/ Architecture design	C	Senior Java developer/ Tech lead	5+ years
POS PAYMENTS	D	Product manager / Team lead	10+ years

The experts have received the demonstration documentation 2 days up front together with the questions shown in paragraph 3.2.1. During the panel these questions have been discussed together with the demonstration documentation.

6.4 Fulfillment of objectives

The objective of this research is to provide structure and oversight in the architectural changes that need to be made to provide a basis for current as well as future mPOS innovations.

Developing an architecture that is flexible enough for the current and the future mPOS innovations is not sufficient enough because it is not possible to fulfill all requirements of different payment service providers. Furthermore, not all payment service providers will make use of the entire architecture and competition within different aspects will likely occur. Therefore, a decision flow is designed that provides structure and oversight in the architectural changes that need to be made to provide an architecture for both the current as well as the future mPOS innovations. This decision flow should:

- Enable architects to identify a one-size-fits all architecture that is flexible enough to provide for the current as well as the future (m)POS solutions?
- Comply with security and PCI standards

- Be technically feasibly
- Support researchers and companies in making architectural decisions.
- Help identify the effect that changing the architecture will have on the stakeholder quality criteria.
- Identify what scenarios need additional research to monetize the effect of a flexible architecture.

6.4.1 Evaluation checklist

Derived from the design objectives the following checklist will be presented to the evaluation panel:

- Does the architecture decision flow support companies, researchers and architects in their architectural decisions?
- Is the method understandable?
- Can the components form an architecture flexible enough for the coming mPOS and POS solutions?
- Is this solution generalizable to competition in the payment industry?
- What are the strengths and weaknesses of this architecture decision flow?
- Would the architecture comply with security and PCI standards?
- Are the options technically feasible?
- Was the DSRM process in your opinion properly conducted to reach the design objectives?
- Does this expert panel properly validate this research?

These questions were chosen to validate both the architecture decision flow, its components, the demonstration and the process of designing it.

6.5 Findings and Recommendations

This paragraph will elaborate on the outcome of the expert panel evaluation. A common ground could be found for each question that was being discussed in the expert panel meeting. The participants found the design choice flow diagram easily understandable and considered themselves effectively able to go through the process of determining the right architecture for their demonstration criteria.

6.5.1 Determining fulfillment of objectives

The participants agreed upon the usefulness of the decision tool and validated that it would be of assistance for architects. In the participants opinion it allows its users to make a well-educated decision that allows an architect to shape its product accordingly. Furthermore, the combination of requirements analysis and ISO 25010 was considered an excellent approach.

The participants agreed that the proposals would be technically feasible although this is subject to the scenarios and variables that the solution is exposed to. It has to be considered that the proposed solutions will always need to be connected to a good working internet connection. The solution is considered almost fully PCI compliant with exclusion of PIN entry on COTS device. This component is however compliant with Scheme security standards. Furthermore, the connection between quality criteria and design decisions is considered to be clearly visualized.

6.5.2 Strengths

The overall tool was considered very useful and relevant. The various aspects are considered and it gives a clear overview of the effects of certain decisions.

- Clearly structured
- Good use of the IOS 25010 standards
- Extensive information
- Clearly indicates the challenges that need to be further investigated
- The advantages of such a solution are significant
- Objectives are fully approved
- The methodology was properly conducted for an architecture design methodology. Moving from requirements, to options, to design and evaluation was considered a very logical approach.

The strengths indicate the usefulness and feasibility of the tool. Apart from the positive feedback some iterations will be necessary. These are indicated in paragraph 3.4.3.

6.5.3 Recommendations and limitations

Application VS SDK – The flow diagram assumes that an SDK is being used. The application could have other forms or consists of multiple applications. This decision could be considered in the tool.

Feasibility of a one-size-fits-all solution – The tool does allow for designing one single architecture that will be flexible enough to be used for current as well as future solutions however there could be some complications:

- There will always remain a dependency on the driver. Additional software or components could be necessary to communicate between the full server solution and the driver of an unknown device.
- The current mPOS solution get a waiver from EMVCO to allow them to have a slower card communication. This results in the compliance of 500 milliseconds instead of 100 milliseconds for a classical terminal. To use the full server on a classical terminal it should be considered that this speed should be reached or an additional waiver should be provided for the classical terminal solution.

Quantifying latency - For further research it is recommended to try to monetize the amount of latency created by decisions. The expert panel agreed with the assumption that measuring latency is too dependent on various variables such as distance, connection and application structure and that it is therefore not feasible to consider this as part of the scope of this research.

Dependency on network – The solution is very reliant on internet. The consistency on what is available on the hardware devices should be guaranteed before implementing such a solution.

JNI wrapped C/C++ modules – involving the procedure in which Controller' components should be converted from C code to java should not be part of the architectural decisions. The expert panel participants were correct in this assessment and therefore this part has been removed from the design.

Detailed approach – For future research some of the decisions should be further investigated and tested in a real live environment to exclude or include certain consequences. This would provide a more wholesome evaluation.

6.5.4 Evaluation method

Both the research methodology and the evaluation method were considered sufficient for this research. Showcasing a demonstration and following an evaluation checklist gives the right substantiation for the research. However, the participants would have liked more time to read, discuss and test the research more extensively. This was not possible due to time constraints.

6.6 Summary of the demonstration and evaluation

To evaluate this research a demonstration of the architecture decision flow has been conducted and an expert panel was asked to evaluate the research output and come to a consensus on the questions mentioned in paragraph 6.4.1. The expert panel has come to a consensus and evaluated the research. All design objectives are considered obtained. The methodology in an artificial setting allows for theoretical evaluation. A naturalistic implementation of the research outcome would allow for more extensive validation and could provide solutions to the limitations mentioned in paragraph 6.5.3. The expert panel concluded that the outcome of this research is considered to be contributive for academic as well as for social purposes.

7 Discussion

This research provided an architecture decision flow to support organizations, developers and architectural designers in the adoption of software based mPOS solutions. The output of this research consists of an overview of the high-level architecture decisions possible in the upcoming transition from hardware to software-based POS terminals. Furthermore, it provides a quality attributes utility tree derived from the stakeholder attributes. This utility tree is used to indicate the effects of the design choices. This will allow the users of the tool to determine the effects of design choices on future innovations in an early stage and therefore reduce the need for costly architecture iterations. The overview can be used to support building a new payment architecture, an architectural component or to iterate upon an existing architecture. Therefore, the overview is portable and can be utilized by various PSP's.

7.1 Generalization

This research mainly relies on the empirical findings from interviews with industry experts. It is therefore qualitative in its nature and has a limited sample size. This can reduce the validity of the generalization. However, experts with a variety of expertise were interviewed from different companies to provide a holistic approach. The expert panel included different experts than the experts interviewed for data gathering to maximize the extent of the generalization of the outcomes. The expert panel indicates that the output of this research is generalizable to all organizations that are involved in the adoption of mPOS innovations in Western-Europe.

Due to the broad scope of the architecture decision flow both full solution providers as well as companies that only provide components of the POS solution can benefit from this research. This research has opened up the way to an increase in POS competition by distinguishing the different POS components. As these components can operate independently it is likely that competition will occur between PSP component providers. Furthermore, PSP's that already have an architecture in place will benefit from the output of this research because it will provide an overview of the iterations that will be necessary to adopt new mPOS solutions.

The research cannot directly be generalized to scenarios outside of Western-Europe. The scope of this research has only taken stakeholders and variables into account in the Western-European market limiting the scope of the solution. Using this research for other areas will result in some limitations. The network consistency of other areas is likely not comparable with the assumed network consistency for this thesis. Therefore, areas outside of Western-Europe might have a different output. Furthermore, POS legislation and cultural payment behavior can be different in other parts of the world. For example, the lower amount of use of contactless payments and PIN in the U.S. might result in different architectural decision flows in comparison with Western-Europe.

7.2 Connection to the existing literature

Little literature is available on the subject of the new mPOS innovations or its architecture as is shown in the literature review in Chapter 3. However, this research created some awareness on relevant factors and technological innovations contributing to the adoption of mPOS technologies and to the displacement of architectural components. In Chapter 5 some surprising existing architecture design and technology is noted that in combination results in the possibility to move terminal components to the server. These findings were derived from expert interviews. This paragraph will identify how these findings connect to the academic context.

The general effects of applying cloud computing to a normally hardware-based system has often been applied in other industries. The television streaming industry is an example of this and due to the simpler nature of this industry it could already move to the cloud in an early stage. Therefore, many connections can be made with regards to the effects of this movement. According to early findings in literature a server-based solution will be more cost efficient, provide unlimited storage, allows for quick deployment and has a bigger scale of services. All these advantages illustrated in research done by Apostu et al. (2013), are still relevant for this research. The same accounts for the negative effects. Earlier research has already stated that cloud computing can result in possible downtime and an increase complexity of security (Alzahrani et al., 2014). This literature verifies that the expected effects of cloud computing on a payment architecture are similar to the measured effects noted in the literature.

Furthermore, some technologies are proposed to compensate for the increased latency by moving software components. The most important architectural propositions are mentioned in Chapter 5 and consists of the rule-based engine, Websockets, and Edge computing. These existing technologies should decrease latency and therefore compensate for the increased communication as a consequence of server-based payments. Although no academic literature could be found on the combination of these technologies with software payments, research has been conducted on these features for different applications (Schwabe et al., 2019; Silva et al., 2018).

Firstly, research was done on the implementation of a rule-based management system similar to the suggested rule-based engine in this research (Schwabe et al., 2019). The rule-based management system or rule-based engine developed for the research of Schwabe et al. (2019), was applied on a construction application. Although the industry it was applied to is entirely different the added value of using this structure-based coding is similar. It allowed separation of logic from the data and it simplified the process significantly. This research therefore underlines the statements that were made by the experts in this research.

Secondly, the Websocket protocol and Edge computing are both used to increase the speed of communication between the server and a hardware device (Silva, et al., 2018). Premsankar et al. (2018), did research on using edge computing for the IoT (internet of things). Its usefulness for IoT appeared significant due to the increase in communication efficiency The application of a technology to IoT and Server based payments is highly connected as a using a connected device as a payment terminal is similar to the IoT concept. Therefore, many of the innovations mentioned in IoT research are similarly applicable to this research.

Furthermore, other existing literature indicates that researchers have tried to grasp the potential of the upcoming payment innovations by including it in the IoT, Industry 4.0 or smart city applications. Some overlap can be found with this literature. In the paper “Challenges of IoT payments in smart services” by Stankovski et al. (2019), it is stated, similar to this research that the increased connectivity and use of smart systems could provide for the implementation of IoT payments. However, the research by Stankovski et al. (2019), concludes that payment services in IoT is already wide spread because of the use of for example vendor-less kiosks.

This indicates that even though this paper is published in 2019 it is already strongly outdated. This research has therefore contradicted the research by Stankovski et al. (2019), by stating that their proposed IoT solution does not enable full connectivity with IoT whereas the mPOS solutions in combination with the architecture designed in this paper do enable full connectivity and portability.

Finally, although there is no academic literature to be found on cloud migration of a full POS solution, there is an extensive amount of literature that proves the possibilities of cloud migration in general or for different solutions. Various cloud migration strategies have been analyzed in research done by Kehrer and Blochinger. (2019). They state that the cloud evolved into an attractive execution environment for parallel applications. These applications require architectural refactoring to benefit from the cloud-specific properties (Kehrer, & Blochinger, 2019). Furthermore, according to their analysis on several papers architectural refactoring comes with many challenges and cannot be applied to all applications due to performance issues. Their research validates the possibility of cloud migration of complex systems such as payment systems but also indicates that architectural changes are necessary and are challenged by performance issues. This creates a link with the outcome of this research because it actually identifies the architectural refactoring decisions and the effects on performance or quality criteria.

The academic research that could be found indicates the significance and the validity of the proposed solutions in this research by proving their use in other research domains. Furthermore, the literature indicates that there is no comparable output available and that there is need for more research on the subject of cloud migration architectures within the payment industry. Consequently, the amount of original content provided by this research contributes significantly to the overall academic context and adds relevance to existing literature.

7.3 Limitations

As mentioned in paragraph 7.1, due to resource constrictions the scope of this research is limited to Western-Europe. It is therefore not necessarily applicable to other areas. Using this research for other areas than Western-Europe can still provide support but is limited to the constraints mentioned in paragraph 7.1. Furthermore, by request of the market experts only the two largest operating systems, IOS and Android have been considered. This limits the possibility of the research results and therefore more research should be done on the possibility to implement similar structures on other operating systems.

Furthermore, the actual implementation of a proposed architecture is very time consuming. This restricts the demonstration and testing phase to theoretical tests. The validation is therefore highly reliant on expert opinions. The contradictive opinions shown in Chapter 5 are therefore illustrated in the decision flow. Security experts stated that a fully software-based solution can be similarly secure as a hardware-based solution. The security authority expert contradicts this in his statements. Although this can be regarded as a limitation it does not highly affect the research as the outcome of both statements still remains in the scope of the PCI security requirements. In consultation with PSP Adyen a partial implementation will be made feasible and should solve for the reliance on qualitative data in further research.

The objectivity of this research is limited because of the fact that it was conducted by only one researcher under the supervision of subject experts. Therefore, the analyzed data was subject to the perceptions and interpretation of a single researcher. This could be avoided by combing agreements of multiple researchers. The subjectivity of this research was however mitigated by strictly following methodology and the use of continues validation by experts and an expert panel. The continues architecture approach enforced continues validation by experts, decreasing the vulnerability to subjectivity of this research. The positive validation of the expert panel indicates that the interpretations of the researcher were aligned with the data provided by the experts involved in this research.

Due to the time restrictions for this research the data on the market requirements is gathered by interviewing experts on the relevant markets. The number of experts representing the payment industry is limited. More extensive market research can therefore provide more precise data on the markets preferred quality attributes. This research is conducted in collaboration with PSP Adyen who has provided the needed resources. To expand the resources also outside of one single PSP more vendors and market representatives have been interviewed, and an additional collaboration has been formed with Riscure, which is a payment acceptance security laboratory. A significant number of experts will eventually give a broader view on the solution and likely a more viable solution. However, due to the use of a variety of experts that represent a significant part of the market this limitation was perceived acceptable by the expert panel.

Building an architecture in general requires a holistic high-level overview. This resulted in an outcome that considers quality attributes and therefore does not consider detailed requirements and scenarios. The actual implementation of architectural decisions that are made with support of this research should therefore first be tested on the practical implementation. Variables such as latency and application efficiency are highly dependent on implementation details such as the used hardware, the server location and the internet speed. To address this limitation the architecture decision flow consists of a component-based architecture. This allows architecture designers to adjust components dependent their specific scenario. For example, building the Controller in a component-based structure allows for the designer to relocate certain components to increase certain quality attributes such as latency.

7.4 Added value

As mentioned at the start of this chapter the output of this research consists of an architecture decision flow that supports the adoption of new mPOS solutions by providing an overview of the to be made architecture decisions. This was developed by gathering requirements from the relevant markets, the security authorities and from payment experts. These requirements were then used to form quality attributes and guide the interviews with multidisciplinary experts. The experts provided a combination of architectural options that allow for a portability of terminal software that could not be found in the current literature.

The obtained quality attributes combined with the ISO 25010 standard provides a basis for future designers to determine their quality attribute scenarios. The ISO 25010 standard is not specifically designed for a payment architecture and therefore the iterations made to it by this research provide a better suited alternative according to industry experts. This will be beneficial for new payment architecture research and for the next steps of designing a more specific mPOS solution architecture.

The high-level overview provided by this research allows both providers of an end-to-end payment solution as well as providers of parts of a solution to benefit from the outcome. This gives an overview of the opportunities that become possible for companies that want to specialize in a specific architecture component. Therefore, the outcome of this research increases the simplicity of adopting architecture components and provides a foundation for competition between payment architecture component providers.

The architecture decision flow enables architects to identify a portable architecture that is flexible enough to provide for the current as well as the future mPOS solutions. This creates awareness and manages expectations for organizations that want to keep innovating their POS solution. The demonstrated combination of design decisions resulted in an architecture that can be used for Classical terminals, mobile terminals, SPOC devices and CPOC devices. This allows PSP's to be flexible in their decisions and therefore simplifies the adoption of new mPOS innovations.

Furthermore, the architecture design flow provides an oversight of how the mPOS architecture can be divided in components. This structures decisions on which parts should be used in an internal server and which parts can live elsewhere. The payment industry can use this information to decide upon only investing in a part of the solution on which their confident about building. Therefore, it provides support for making complex buy or build decision. Furthermore, this will enable a more flexible form of competition within the POS market. It is no longer necessary to compete between full POS solutions. Instead, it is now possible to specialize in a single component and compete between POS components such as UI, Kernel and Controller. User can therefore switch components at their convenience.

Lastly, the combination of technological innovations mentioned in Chapter 5 that allow for the relocation of the various components result in several advantages. Moving components away from the user and into internal servers gives direct control to the solution builder. This will significantly increase the efficiency in testing, updating, certification, configuration and data monitoring of the mPOS solution.

7.5 Open research problems

This paragraph focusses on open research problems that are exposed by this research. The further research that will be conducted by the same researcher as a follow up of this research can be found in Chapter 8.

Future research should be focused on two aspects. Firstly, the focus should lie on putting the architecture decision flow into practice and developing the architecture. Secondly, further research can be derived from the knowledge gaps determined in this research. The first part will allow the researcher to fully validate and quantify the effects on the in Chapter 5 mentioned quality criteria. By developing a component and server-based architecture as shown in the demonstration future research will be possible on quantifying important variables such as latency and portability in different scenarios. This would provide additional data that can better determine and extend the boundaries of this research. Developing the proposed architecture would enable researchers to verify the different variables that have an effect on the determined quality criteria. Quantifying these effects will help in determining the feasibility of the proposed solution in various locations and scenarios. Dependent on the available resources it would take multiple years to develop such an architecture and therefore this could not be part of this research.

Furthermore, some aspects have not been considered for this research due to the boundaries of this research or the limitation in resources. The use of web-assembly technology could potentially play a role in moving more aspects of the application to the server. This would result in a fully cloud rendering solution. Limited information and expertise on this technology is currently available due to its novelty. Further research on the potentials of this solution in a later stage would therefore be interesting. Additionally, from the data gathered for this research information could be derived on the various structures the eventual application could potentially have. This is not within the scope of this thesis because it does not relocate any component. However, this data indicated that the decision on the application structure as is shown in Appendix B has an effect on the quality attributes. Therefore, additional research on the effects of using an SDK or an application or hybrid variants as mentioned in Appendix would be beneficial for the payment industry.

Moreover, this research only focused on the West-European market. Other markets might have different requirements and characteristics that potentially make this research less or more relevant to such markets. Areas with a bad network infrastructure will likely benefit less from this research and areas that have less strict security regulations such as the U.S. will likely be able to adopt the mPOS solutions more easily. Further research is therefore necessary to determine the differences with other markets and identify the changes that are necessary to this research outcome to be able to utilize this research for a larger scope.

Finally, due to the novelty of most innovations mentioned in this research little literature is available on the market potential of such solutions. This research focusses on the adoption of mPOS innovations by PSP's and also considers the requirements from the mPOS users. However, to give a more holistic view of the potential of these innovations more extensive research should be done on where these innovations can be adopted and which benefits they will bring to those areas. Combining this research on accepting payments on a smartphone or tablet with the currently widely used technology of paying with a smartphone has significant potential for eliminating the need for cash payments. The rise of global network connectivity combined with easily implementable mPOS solutions could open up the way for areas that do not have a payment infrastructure to directly be connected to cashless payments. This has significant potential for upcoming societies and should be further investigated.

8 Conclusion

The aim of this research is to decrease the complexity of the adoption of new mPOS innovations. Designing a payment architecture that allows for the adoption of all current and future mPOS innovations is not sufficient because a single architecture cannot fulfill the variety of requirements that the different users in the payment industry might have. Furthermore, a single payment architecture cannot always be built by iterating upon existing architecture. Therefore, a decision flow that provides an overview of the architectural changes possible for both the current as well as the future mPOS solutions is designed. This decision flow enables architects to identify a portable architecture that is flexible enough to provide for the current as well as the future (m)POS solutions. Furthermore, it complies with security standards and is technically feasible according to an expert validation panel. This architecture decision flow supports researchers and companies in making architectural decisions and provides insight the effects these decisions will have for the adoption of future mPOS innovations. Therefore, the objective of this research, to provide structure and oversight in the architectural changes that need to be made and thus form a basis for current as well as future mPOS innovations is succeeded. This chapter summarizes the sub-research questions and concludes on the main research questions. Figure 8-1 below presents the sub-research questions and in which chapters these questions are addressed.

The main motivation for this research comes from the knowledge gap in the academic literature and the informal request for such a structured design by the payment industry. The research is conducted in a phase wise approach guided by the design science research methodology combined with the continues architecture principles. The architecture support tool for mPOS adoption is based on literature research, multidisciplinary expert interviews, expert panels and informal conversations with market representatives. Compact overviews and illustrations are developed to provide a simple insight in the analyses made on the gathered data.

Furthermore, the output of this research is evaluated by an expert panel with extensive industry knowledge. The experts found common ground on the societal and scientific contribution of this research. They agreed that the output of this research is generalizable to the full payment industry in Western-Europe and acknowledged its significance outside of the Western-European scope.

8.1 A summary of the answers to the sub-research questions

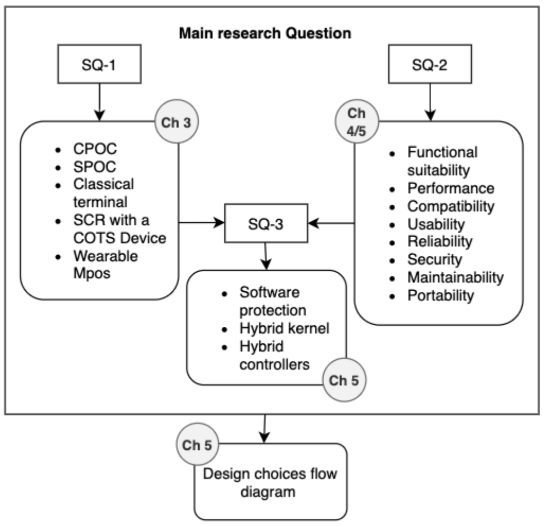


Figure 8-1 Answering sub-questions

This research provides an answer to the main research question:

How can one single payment architecture provide a secure payment solution that can be used for the current as well as the future (mobile) Point of Sale (POS) solution?

The answers to this main research question are derived from designing a decision flow that allows for the design of a single payment architecture that can be used for the current as well as the future (m)POS solutions. This design is made following the answers of three sub questions.

SQ1 - What are the most significant current and future mobile POS solutions?

The significant POS solutions consists of the solutions that can be certified by the PCI security standards council or that will likely be certifiable in the near future. Therefore, the following POS systems are considered: The classical terminal, the wearable mPOS terminal, the SCR with a COTS device, The SPOC solution and the CPOC solution. These solutions are visualized in Figure 8-2.

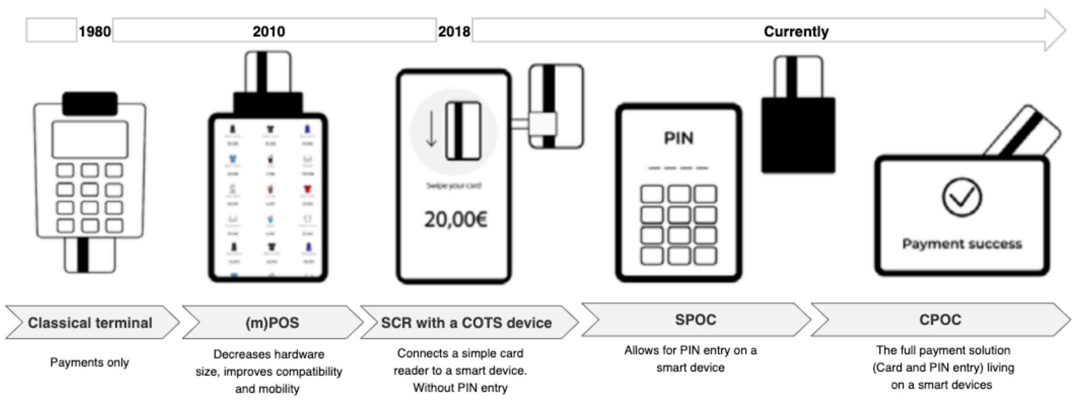

1980
2010
2018
Currently
Swipe your card
20,00€
PIN
Payment success
Classical terminal
Payments only
(m)POS
Decreases hardware size, improves compatibility and mobility
SCR with a COTS device
Connects a simple card reader to a smart device. Without PIN entry
SPOC
Allows for PIN entry on a smart device
CPOC
The full payment solution (Card and PIN entry) living on a smart devices

Figure 8-2 Significant mPOS innovations

The chronological list of expected mPOS solutions and their specifications and requirements are shown in table 2. The research to answer sub-question 1 is done by conducting literature research and informal interviews. These POS solutions are further elaborated on in Chapter 3 and visualized in table 2.

SQ2 - What are the stakeholder's quality attributes for the most significant current and future mobile POS solutions?

First the stakeholders are identified and elaborated on in Chapter 1. The stakeholders are grouped in merchants, PSP's and security authorities. Secondly, an extensive research is conducted using various methodologies such as literature research, unstructured interviews, informal interviews, expert panels and surveys to answer sub question two. The requirements derived from the obtained data are combined with existing architecture validation literature to form a quality attributes utility tree. The quality attributes are visualized below in Figure 8-3 The utility tree illustrates the quality attributes as well as its quality refinements. These are further elaborated on in Chapter 4.

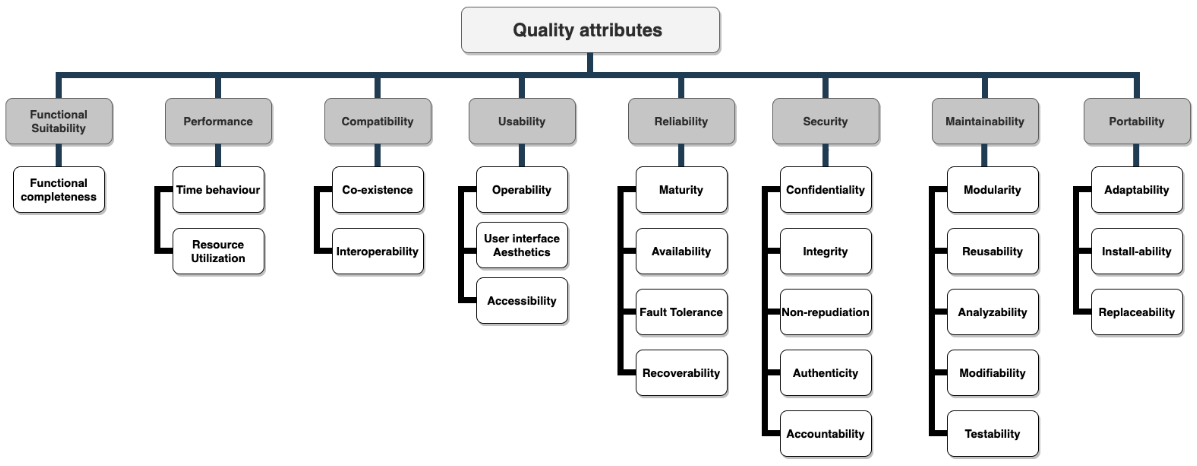


Figure 8-3 Quality attributes utility tree

In conclusion, the utility tree forms the basis for research question three and for the design decisions shown in the output of this research. Furthermore, the quality attributes gathered can be used for further research to identify and quantify the effects of architecture design decisions.

SQ3 - What payment architecture fits the requirements derived from sub question one and two?

A number of multidisciplinary expert interviews reveal the possible design decisions that can be made within the boundaries of the requirements derived from sub question one and two. Following the process of this research it is determined that one single architecture cannot fulfill the variety of requirements within the payment industry. However, this research proved it to be possible to design an architecture that allows for the current as well as the future mPOS solutions. To determine which payment architecture will fit the variety of requirements by the payment industry an architecture decision flow was developed that would enable a variety of users to build an architecture capable of adopting all earlier mentioned mPOS solutions. Furthermore, the design decisions following from the design flow are analyzed by determining their effect on the quality criteria. These quality criteria are obtained from the answers on sub question two.

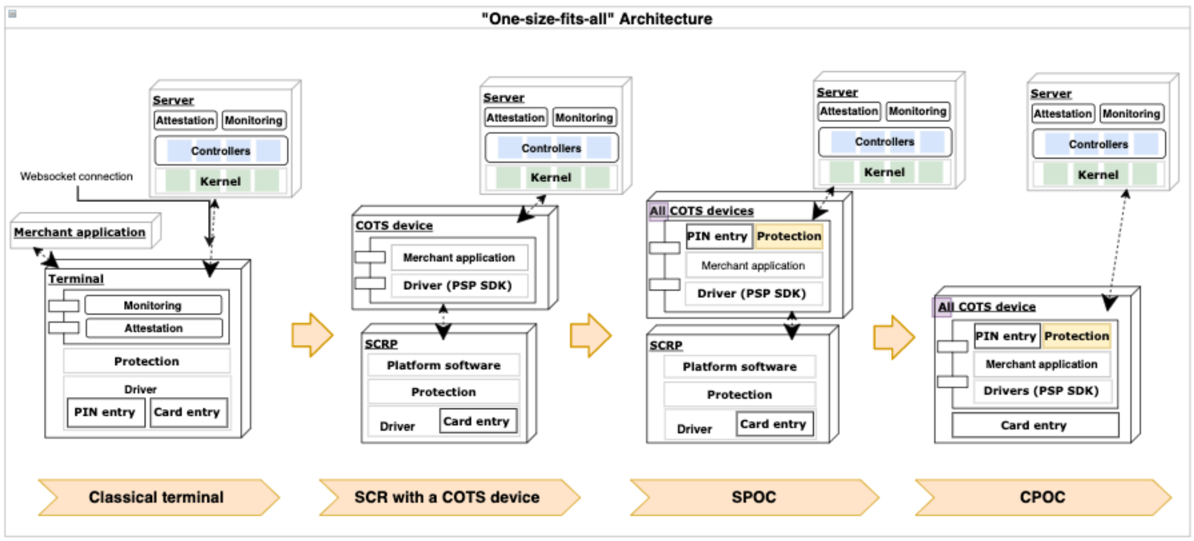


Figure 8-4 Portable architecture

Figure 8-4 illustrates the design of a payment architecture that fits all solutions mentioned earlier in this chapter. Cloud migration of the Controller and the Kernel combined with the movement from hardware based to software-based security enable the architecture to fit on COTS devices as well as classical terminals and other payment hardware. Furthermore, it fulfills all security and technical requirements.

RQ - "How can one single payment architecture provide a secure payment solution that can be used for the current as well as the future (mobile) Point of Sale (POS) solution?"

The answers found for sub question one, two and three lead to the identification of design decisions that allow an architect to design an architecture that fits the most significant mPOS solutions. These decisions can be quantified by measuring their effect on the quality criteria defined by the answers of sub question two. The design decisions are illustrated below in Figure 8-5 and are further elaborated on in Chapter 5. By migrating the colored components indicated in Figure 8-5 to the server a single architecture can provide a solution for the current as well as the future mPOS solutions. Due to the broad variety of unknown variables for different PSP's a single payment architecture cannot be built for all PSP's. Therefore, the decision flow that is designed enables architects to design a single payment architecture that can provide a secure payment solution for the current as well as the future mPOS solutions for their specific requirements.

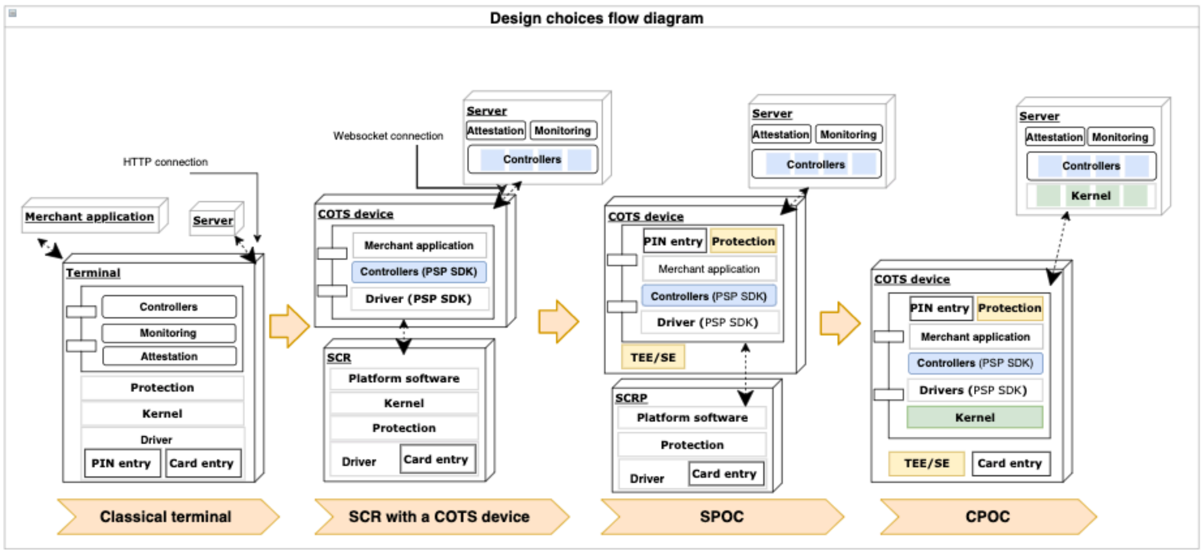


Figure 8-5 Architecture decision flow

In conclusion, following the DSRM methodology an architecture decision flow is designed that provides an overview of the architectural decisions possible to build an architecture for both the current as well as the future mPOS solutions. This enables payment service providers to build an architecture fitting their quality scenarios while maintaining visibility on the changes necessary for future mPOS solutions. It also allows new entrants to design an architecture that allows for all mPOS innovations without iterating upon the architecture. This decreases the complexity that was created by the move from hardware to software payment devices and therefore simplifies its adoption.

8.2 Scientific contribution

Although a significant amount of academic literature is available on cloud-based solutions in various industries, no literature can be found on similar applications for the payment industry. This research improves current literature on evaluating software architectures. Adjusting the general quality attributes often used in literature to evaluate software architecture results in the design of a quality attribute utility tree that can be used specifically for payment industry architecture. Future research on payment architecture can use this method of evaluation to conduct a more detailed evaluation of their proposed architectural changes. Moreover, Chapter 7 indicates the extensive amount of further research possibilities that have been exposed as a consequence of this research.

Furthermore, the mPOS innovations have just recently been proposed and therefore no academic research is available on the involved mPOS solutions. By illustrating the necessary steps for building a flexible architecture ready for future purposes the current literature on payment solutions has been made more recent. This has resulted in providing an academic base for the implementation of practical tests. This research will create awareness of the potentials of the upcoming mPOS innovations and provides a direction of new academic research. Having accurate and recent knowledge is important for the fast changing and innovative nature of the academic software engineering discipline. To allow for efficient use of this research a 4-page abstract has been added involving easily understandable illustrations. This will support academic researchers in conducting research on further payment system improvements.

8.3 Societal contribution

This research is beneficial to society from multiple perspectives.

- This research simplifies the adoption of mPOS technology
- This research decreases the need for architecture iterations and therefore development costs
- This research improves the check-out experience of shoppers
- This research increases the number of devices that can accept payments and therefore decreases the need for cash as well as the need for additional hardware.

According to the expert validation panel this research simplifies the adoption of mPOS technology. This is done by providing an overview of the architectural decisions necessary to adopt the upcoming technologies. The possibility of building an architecture that is portable enough to fit the current as well as the future mPOS solutions decreases the need for expensive architecture iterations or new architecture developments. Therefore, adopting new mPOS technologies becomes easier and less expensive.

As a result of an increase in adoption of new mPOS innovations the society as a whole will benefit by experiencing a better check-out experience. A flexible payment architecture in which most of the software lives in a server will allow for a variety of devices to become payment acceptance devices. This creates new use-cases for shoppers such as paying for clothing by tapping your card on a mirror. The increase of the number of devices that can become payment acceptance devices will prove to result in multiple contributions to society. More people will have cheaper access to a payment infrastructure which will likely increase market growth and decrease the need for cash. Furthermore, no additional hardware is required and therefore the payment solution becomes more sustainable.

An interesting link with the societal contribution can be made with a recent press release by retail chain Walmart. Walmart has recently bought 740.000 smartphones for their employees (“Walmart Unveils Associate App ", 2021). Walmart bought the Samsung Xcover Pro model which is one of the few smartphones that is already EMV level 1 certified. This means that their hardware is already secure enough to transform the smartphone into a payment acceptance device with the click on a button as soon as this technology is ready to launch. This gives an indication of the potential relevancy of this research for society. It is furthermore interesting that that Walmart chose for a hardware security model. This indicates the importance of not building one architecture but instead building a decision flow to consider the variety of requirements that different payment service providers might have.

Moreover, the Complex Systems Engineering program of the TU Delft has provided the capabilities necessary to conduct this research. The issue addressed in this research required a multidisciplinary approach. Requirements gathering from stakeholders, development of utility trees, decision analysis, and the consideration of complex legislation are all aspects of this research that requires system engineering. This results in the simplification of a complex system and thus showcases an example of how organizations can benefit from the CoSEM approach.

8.4 Reflection and future research

With the increase of internet coverage, cloud computing is becoming an increasingly significant technology implementable in a variety of use-cases. This results in cloud computing playing a significant role in movie streaming, games, healthcare and other partly computer driven areas. The shift in the POS industry from hardware to software-based POS systems creates new opportunities for cloud computing and POS payments. The new mPOS innovations allowing for software-based payments clear the road for more research on the possibility of a fully cloud based payment solution. This allows the payment industry to finally start benefitting from the advantages and practicalities that cloud computing brings to society.

Cloud computing is a technology that has been widely adopted in other markets already. Various new innovations make it possible to finally implement similar cloud architectures in the payment industry. The increase of global network coverage and the growth of contactless payments in combination with the adoption of NFC chips in smartphones are mainly responsible for this development. To allow for full adoption of these new mPOS innovations further growth in these areas is required. Furthermore, the implementation of new developments such as edge computing, 5G and peer2peer services such as Apple and Google pay will fasten and improve the adoption further. The increased adoption of these innovations will make this research even more relevant in more areas. This will require significant investments in infrastructure and will therefore not likely occur in the near future. For this solution to succeed globally the first and crucial step is to fully switch to contactless payments and provide the global population with an NFC readable payment card as has been done in most of Western-Europe.

This new revolution in payments will have a significant effect on micro-merchants who can now have access to less expensive payment acceptance devices. Due to the incapability of the proposed solution to accept offline payments the larger retailers will require classical terminals next to the mPOS solutions until network connectivity is deemed sufficiently consistent. At that point CPOC will be the dominant solution in the entire industry and will provide a more sustainable a smooth check-out for shoppers. The real challenge lies in providing a global connection consistent enough to allow for not only the industry to use these mPOS solutions but also the consumers. Payment architecture excludes parts of society that does not have access to sufficient payment infrastructure and cash payments are considered increasingly inefficient. By enabling consumers to both accept and conduct payments using a COTS device there will be no need for cash anymore. This will result in the inclusion of the global society in the payment infrastructure and excluding the need for cash payments by simply downloading an application. Although this seems like an ambitious goal, this research has proven the technical feasibility of this goal. The only constraints keeping us from reaching global payment infrastructure access is a global internet connection and policy.

Dependent on the available resources it would take multiple years to develop, implement and test the proposed architecture and therefore this could not be part of this research. Therefore, this research will be proceeded at payment service provided Adyen. This will allow the researcher to fully validate and quantify the effects on the in Chapter 5 mentioned quality criteria. By developing a component and server-based architecture as shown in the demonstration, future research will be possible on quantifying important variables such as latency and portability in different scenarios. This would provide additional data that can better determine and extend the boundaries of this research. Developing the proposed architecture will enable researchers to verify the different variables that have an effect on the determined quality criteria. Quantifying these effects will help in determining the feasibility of the proposed solution in various locations and scenarios. This will also help identify the changes that need to be made in for example network infrastructure for areas that cannot support server-based payments.

References

Alzahrani, A., Alalwan, N., & Sarrab, M. (2014, April). Mobile cloud computing: advantage, disadvantage and open challenge. In *Proceedings of the 7th Euro American Conference on Telematics and Information Systems* (pp. 1-4).

Android vs iOS market share 2023 | Statista. (2021). Retrieved 20 June 2021, from https://www.statista.com/statistics/272307/market-share-forecast-for-smartphone-operating-systems/

Apostu, A., Puican, F., Ularu, G., Suciu, G., & Todoran, G. (2013). Study on advantages and disadvantages of Cloud Computing–the advantages of Telemetry Applications in the Cloud. *Recent advances in applied computer science and digital services*, *2103*.

Assadi, H. (2020). Driving wider adoption of digital payments at the point-of-sale. Retrieved from https://dejamobile.com/blog_dejamobile/the-rise-of-digital-payments-at-the-point-of-sale/

Bojjagani, S., & Sastry, V. N. (2019). A secure end-to-end proximity NFC-based mobile payment protocol. Computer Standards and Interfaces, 66(1), 103348. https://doi.org/10.1016/j.csi.2019.04.007

Bulman, J., & Garraghan, P. (2020). A cloud gaming framework for dynamic graphical rendering towards achieving distributed game engines. In 12th {USENIX} Workshop on Hot Topics in Cloud Computing (HotCloud 20).

Bruno, P., Townsend, Z., & Zell, J. (2019). The emerging era of PaaS: "payments as a service." Global Payments Report 2019: Amid Sustained Growth, Accelerating Challenges Demand Bold Actions, September, 33. https://www.mckinsey.com/~/media/McKinsey/Industries/Financial Services/Our Insights/Tracking the sources of robust payments growth McKinsey Global Payments Map/McK-2019-Global-Payments-Report.ashx

Britten, N. (2006). Qualitative interviews. Qualitative research in health care, 3, 12-20.

Bittner K. Software requirement practices are ripe for disruption. Forrester Research, April 2014.

Curran, R. W. F. A., Lochrie, S., & O'Gorman, K. (2014). Gathering qualitative data. Research Methods for Business & Management, 152-172.

ChenL,BabarMA,NuseibehB.Characterizingarchitecturallysignificantrequirements. IEEE Software 2013;30(2):38–45.

Chung L. Non-functional requirements in software engineering. Norwell, MA: Kluwer Academic Publishers; 2000.

Cai, Y., Xiao, L., Kazman, R., Mo, R., & Feng, Q. (2018). Design rule spaces: A new model for representing and analyzing software architecture. IEEE Transactions on Software Engineering, 45(7), 657-682.

De Reuver, M., & Ondrus, J. (2017). When technological superiority is not enough: The struggle to impose the SIM card as the NFC Secure Element for mobile payment platforms. Telecommunications Policy, 41(4), 253–262. https://doi.org/10.1016/j.telpol.2017.01.004

DiCicco-Bloom, B., & Crabtree, B. F. (2006). The qualitative research interview. Medical Education, 40(4), 314–321. https://doi.org/https://doi.org/10.1111/j.1365-2929.2006.02418.x

Ghosh, S., Majumder, A., Goswami, J., Kumar, A., Mohanty, S. P., & Bhattacharyya, B. K. (2017). Swing-Pay: One Card Meets All User Payment and Identity Needs: A Digital Card Module using NFC and Biometric Authentication for Peer-To-Peer Payment. IEEE Consumer Electronics Magazine, 6(1), 82–93. https://doi.org/10.1109/MCE.2016.2614522

EMVCO. (2021). EMV Contactless Book C-2 Kernel 2 Spec v2.10.

Erder, M., & Pureur, P. (2016). Continuous architecture: sustainable architecture in an agile and cloud-centric world. Amsterdam [u.a]: Morgan Kaufmann/Elsevier.

Glinz M. A risk-based, value-oriented approach to quality requirements. IEEE Softw 2008;25(2):34–41.

Guille, G. (2018). Cloud-Based transaction methods and systems (Vol. 2). https://patentimages.storage.googleapis.com/52/6a/3f/9eb54a594f5a97/US9959779.pdf

Hamed Taherdoost. Sampling Methods in Research Methodology; How to Choose a Sampling Technique for Research. International Journal of Academic Research in Management (IJARM), 2016, 5. ffhal-02546796

Hartog, T., & Moreno, A. (2018). Analyzing the security of Riscure. (2019). Tap on Phone transactions on Moreno, A. (2018). Analyzing the security of

Hartog, T., & Moreno, A. (2019). Tap on Phone transactions on smartphones. Riscure.

Haoues, M., Sellami, A., Ben-Abdallah, H., & Cheikhi, L. (2017). A guideline for software architecture selection based on ISO 25010 quality related characteristics. International Journal of System Assurance Engineering and Management, 8(2), 886-909.

Jonker, N. (2021). Point of sale payments during the COVID-19 pandemic. Retrieved 5 June 2021, from https://www.dnb.nl/en/actueel/dnb/dnbulletin-2020/contactless-payments-on-the-rise-during-the-covid-19-pandemic/

Kazman, R., Klein, M., Barbacci, M., Longstaff, T., Lipson, H., & Carriere, J. (1998, August). The architecture tradeoff analysis method. In Proceedings. Fourth IEEE International Conference on Engineering of Complex Computer Systems (Cat. No. 98EX193) (pp. 68-78). IEEE.

Kehrer, S., & Blochinger, W. (2019). A survey on cloud migration strategies for high performance computing.

Kitchenham, B. A. (2004). Performing Systematic Reviews. Joint Technical Report Software Engineering Group. Department of Computer Science Keele.

Levialdi Ghiron, S., Sposato, S., & Maria Medaglia, C. (2009). NFC ticketing: A prototype and usability test of an NFC-Based virtual ticketin application. IEEE.

Liao, Y., He, Y., Li, F., & Zhou, S. (2018). Analysis of a mobile payment protocol with outsourced verification in cloud server and the improvement. Computer Standards and Interfaces, 56(September 2017), 101–106. https://doi.org/10.1016/j.csi.2017.09.008

Lucero, B., Linsey, J., & Turner, C. J. (2016). Frameworks for organising design performance metrics. Journal of Engineering Design, 27(4–6), 175–204. https://doi.org/10.1080/09544828.2015.1135235

Leff A and J. T. Rayfield, "Web-application development using the Model/View/Controller design pattern," Proceedings Fifth IEEE International Enterprise Distributed Object Computing Conference, 2001, pp. 118-127, doi: 10.1109/EDOC.2001.950428.

Marvasti, A. (2018). Research methods. The Cambridge Handbook of Social Problems, 1(3), 23–37. https://doi.org/10.1017/9781108656184.00
PCI. (2019). Payment Card Industry (PCI) Contactless Payments on COTS (CPoC ™) Security and Test Requirements. December.

McCormick, M. (2012). Waterfall vs. Agile methodology. MPCS, N/A.

Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A design science research methodology for information systems research. Journal of Management Information Systems, 24(3), 45–77. https://doi.org/10.2753/MIS0742-1222240302

PCI Security standard council. (2021). Contactless Payments on COTS (CPoC™) , Security and test requirements.

Pries-Heje, J., Baskerville, R., & Venable, J. R. (2008). Strategies for design science research evaluation.

Pourghomi, P., Qasim, M., & Ghinea, G. (2013). A Proposed NFC Payment Application. International Journal of Advanced Computer Science and Applications, 4(8), 173–181. https://doi.org/10.14569/ijacsa.2013.040824

Oliveira G. M. B. et al., "Comparison Between MQTT and WebSocket Protocols for IoT Applications Using ESP8266," 2018 Workshop on Metrology for Industry 4.0 and IoT, 2018, pp. 236-241, doi: 10.1109/METROI4.2018.8428348.

Ritchie J., Lewis J., Nicolls C. M. & Ormston, R. (2014). Qualitative research practice: A guide for social science students and researchers (Second ed.). London: SAGE Publications.

Ritchie, J. & Spencer, L.. (1994). ‘Qualitative data analysis for applied social research’, in Bryman, A., & Burgess, R. G. (eds). Analyzing qualitative data. London, England;New York, New York;: Routledge. doi:10.4324/978020341308

Raqib S., Rizwan M. (2020) NFC Payment Security with Cloud Based Authentication System. In: Bajwa I., Sibalija T., Jawawi D. (eds) Intelligent Technologies and Applications. INTAP 2019. Communications in Computer and Information Science, vol 1198. Springer, Singapore. https://doi.org/10.1007/978-981-15-5232-8_63

Runeson, P., & Höst, M. (2008). Guidelines for conducting and reporting case study research in software engineering. Empirical Software Engineering, 14(2), 131. https://doi.org/10.1007/s10664-008-9102-8

Schwabe, K., Teizer, J., & König, M. (2019). Applying rule-based model-checking to construction site layout planning tasks. *Automation in construction*, *97*, 205-219.

Shakeel Ahmad, S., Al-shourbaji, I., & Al-janabi, S. (2016). A secure NFC mobile payment protocal based on biometrics with formal verification. IJITST, 6(2).

Sushil, M., Senant, Y., Drummond, A., & Ampenberger, M. (2019). Global payments; Tapping into pockets of growth. https://image-src.bcg.com/Images/BCG-Global-Payments-2019-Tapping-into-Pockets-of-Growth-September-2019-rev_tcm9-231986.pdf

Stankovski, S., Ostojic, G., Tarjan, L., Stanojevic, M., & Babic, M. (2019). CHALLENGES OF IoT PAYMENTS IN SMART SERVICES. Annals of DAAAM & Proceedings, 4-10.

Stenberg D. 2014. HTTP2 explained. SIGCOMM Comput. Commun. Rev. 44, 3 (July 2014), 120–128. DOI:https://doi.org/10.1145/2656877.2656896

Silva, D. R. C.   G. M. B. Oliveira, I. Silva, P. Ferrari and E. Sisinni, "Latency evaluation for MQTT and WebSocket Protocols: an Industry 4.0 perspective," 2018 IEEE Symposium on Computers and Communications (ISCC), 2018, pp. 01233-01238, doi: 10.1109/ISCC.2018.8538692.

Shafique, K., Khawaja, B. A., Sabir, F., Qazi, S., & Mustaqim, M. (2020). Internet of things (IoT) for next-generation smart systems: A review of current challenges, future trends and prospects for emerging 5G-IoT scenarios. Ieee Access, 8, 23022-23040.

Turk, I., & Cosar, A. (2016). An open, NFC enabler independent Mobile payment and identification method: NFC feature box. IEEE 17th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM), 1(3).

Visa Tap-on-phone specifications. (2021). Retrieved 7 June 2021, from https://partner.visa.com/site/programs/visa-ready/tap-to-phone.html

Walmart Unveils Associate App, Giving 740,000 Associates Smartphones. (2021). Retrieved 17 June 2021, from https://risnews.com/walmart-unveils-associate-app-giving-740000-associates-smartphones

Walmart Unveils Associate App, Giving 740,000 Associates Smartphones. (2021). Retrieved 17 June 2021, from https://risnews.com/walmart-unveils-associate-app-giving-740000-associates-smartphones

Yazmaci, E. (2021). How Cloud Kernel Addresses Issues Payment Terminals Have Today. Retrieved 5 June 2021, from https://info.poynt.com/blog/how-cloud-kernel-addresses-issues-payment-terminals-have-today

Appendices

8.5 Appendix A: Functional requirements from PSP experts.

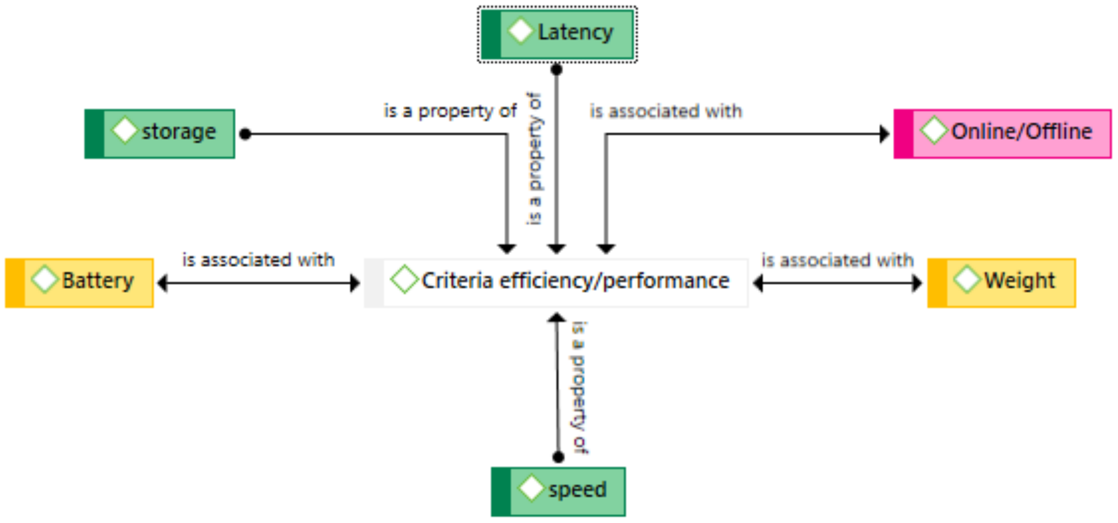
Product requirements		Must have	Should have	Could have	Will not have
Request Types	Pre-Authorization	✅			
	Top-up Authorization	✅			
	Authorization Release	✅			
	Sales Completion / Capture	✅			
	Sale/Purchase (PayOnly)	✅			
	Referenced Refund (Reversal)		✅		
	Unreferenced Refund	✅			
	Cancel Unreferenced Refund	✅			
	Cancel In-Progress Payment (Abort)	✅			
	Transaction Status			🔮	
	Log In and Log Out			🔮	
	Retrieve Totals			🔮	
	Reconciliation			🔮	
	Card Acquisition		✅		

	Collect Input				✅
	Display Data				✅
Payment Features	Giving				✅
	Apple VAS				✅
	Application Selection	✅			
	Cashback				✅
	Chip cards	✅			
	CTLS	✅			
	CVM Settings	✅			
	DCC				✅
	Giftcards			✅	
	Installments				✅
	MKE				✅
	MOTO				✅
	MSR	✅			
	Offline EMV			🔮	
	Pay@table				✅
	Pay by Link in-store			✅	
	Payments with Tokens (ContAuth)	✅			

	PIN CVM		✓		
	QR-code Payments			✓	
	SAF			✓	
	Shopper Recognition	✓			
	Tax Free Shopping				✓
	Tipping	✓			
User experience	Device discovery on the local network (bonjour protocol)			✓	
	Fully customizable UI	✓			
	Long battery life (tbd what long means)	✓			

8.6 Appendix B ATLAS TI merchant findings

Efficiency/performance

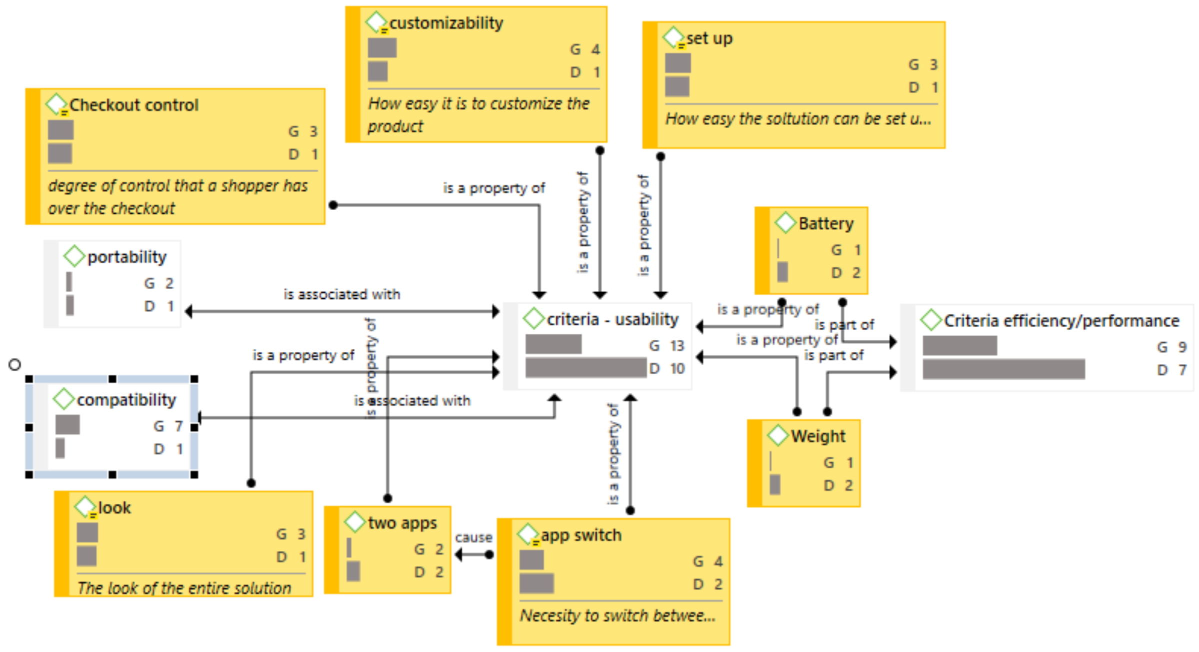


The performance of the eventual solution is of importance to the merchants and represents the performance relative to the amount of resources used under stated conditions. The merchant is not technically skilled enough to estimate the effect of certain conditions and therefore only listed the important requirements regarding performance. The green codes indicate direct criteria indicating the solutions performance. All associate codes are criteria that fall under a different code group but that do have a correlation with performance.

Latency and speed - the solution should not decrease the speed or increase latency compared to the current solutions.
Storage - Storage indicates the amount of data that can be saved. This is independent of whether this will happen on the device or somewhere else. The solution should provide sufficient storage for data analysis. These criteria are associated with the performance of the device.

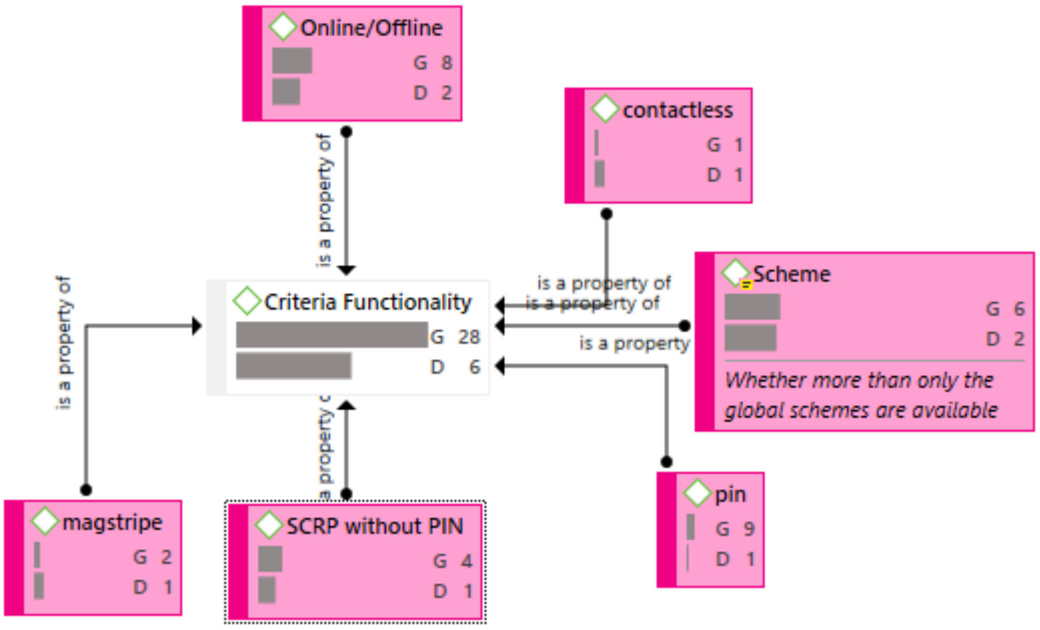
Battery and Weight - The battery life and the weight of the solution determine how mobile the terminal will be.
Online/offline - This indicates the capability of the mPOS solution to process an offline payment.

Usability
The usability of a solution refers to the degree to which the solution can be used by specified users to achieve the determined goals with a measure of efficiency, effectiveness and satisfaction. The following criteria help establish how merchants measure the usability of the proposed system.



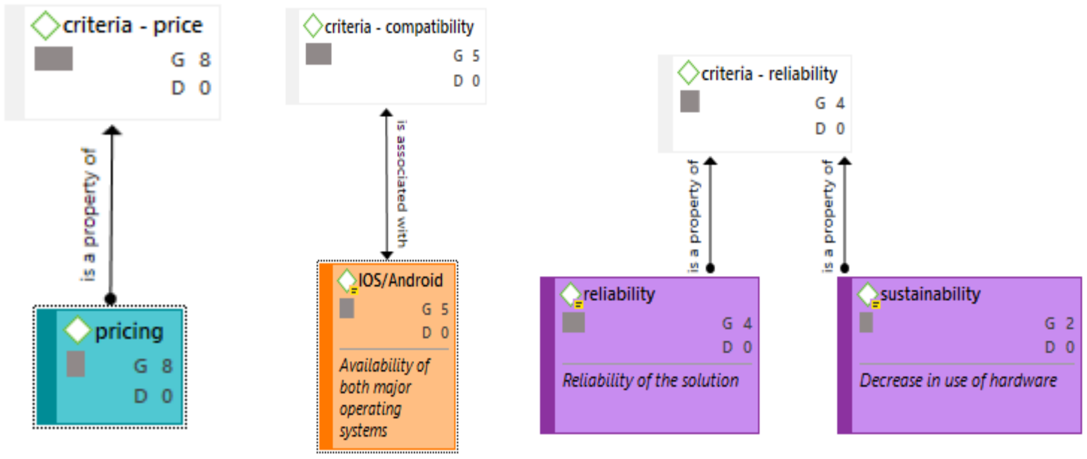
Checkout control - Merchants require their shoppers to have more control over a checkout. No longer relying on a predetermined checkout location because of a more mobile solution that allows communication to the shoppers will have an effect on the amount of control a shopper has during the checkout.
Customizability - Allowing the merchant to customize the solution will improve the user experience of their users.
Set-up: The amount of effort and time that goes into setting up the solution from arrival or download to being able to use the solution is an important criteria for merchants.
Battery - The battery life has an effect on the durability and the mobility of the solution.
Weight - Similar to the battery life the weight will have an effect on the mobility of the solution. A lower weight would increase the mobility and therefore also the look and the checkout control.
Two apps/App switch - The usability of the solution is highly dependent on the architecture of the applications. The use of a merchant application next to a PIN entry application will create the necessity for an application switch.
Look - The move from a hardware towards a software based solution will provide the opportunity to change the look and feel of a payment acceptance device.

Functionality
The degree to which the designed architecture provides the mentioned requirements.



Online/offline - This indicates the capability of the mPOS solution to process an offline payment.
Magstripe, Contactless and chip - This indicates the entry modes that should be available on the solution.
Pin and SCRP without PIN - This indicates the possible use case for PIN entry or for a solution that does not support pin entry.
Scheme - The number of schemes that are available.

Pricing
The costs of the solution are an important criterion. The reduction of hardware makes it easier to alter current pricing.



Compatibility

The compatibility indicates the degree to which an architectural component can exchange information with other systems or products. More simply put, how compatible is the build software with other systems?
Ios/Android - This criterion indicates the compatibility of the mPOS solutions with both major operating systems.

Requirements
The experts mentioned in Chapter 4 have indicated their requirements during an unstructured interview as well as a survey. Their requirements regarding the various quality criteria are shown in Table 8-1to give insight in the changes they expect from new solutions the first column indicates the functionality that their current device allows. The second column indicates how the required functionality should behave.

Table 0-1 Merchant requirements

Requirement	Platform D		Platform C		Company A		Platform A		Platform E		platform C	
Pin	++	++	++	+-	++	++	++	++	++	+-	++	++
Magstripe			+	+-	++	++	++	++				
Chip			+	+-			++	++				
CTLS			+	+			++	++				
Offline							++	++	++	+-		
Look & Feel	--	++	--	+-	--	++	--	++			--	++
Checkout Control	+-	++	-+	++	+-	++	--	++			--	++
Sustainability					+-	++						
Scheme availability			+-	+-	++	+-	++	++				
Speed					++	++						
Reliability					++	++						
Price	--	++	--	++	--	+-	--	+-	--	++	--	++
Battery/Weight			-+	++	--	++						
IOS	--	++	--	+			--	++	--	++	--	++
Set-up	--	+-	--	++			--	+-			--	++

8.7 Appendix D Application versus SDK

The application
SDK or application

Merchant application + PSP application
- Full control over app and updates
- App switching (By the merchant)
- Merchant needs to control app switching

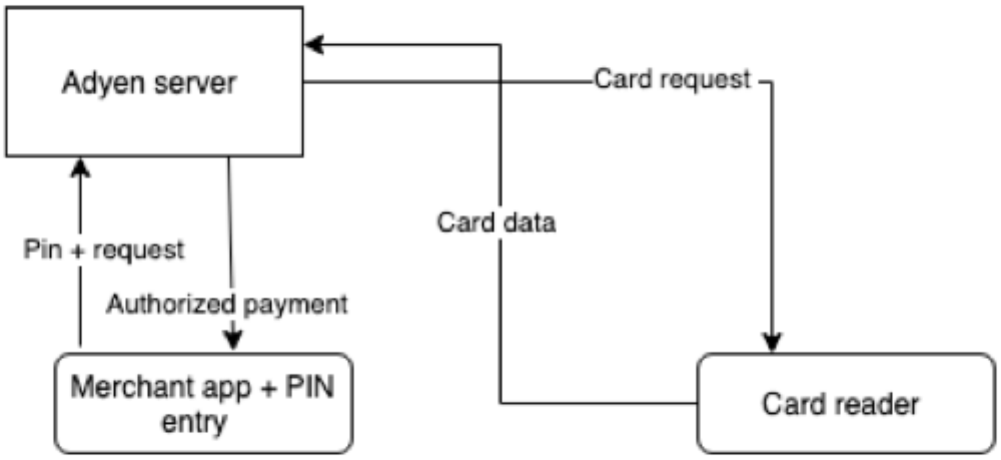
Merchant application + SDK with user interface + application
- No visible app switching
- Some control because the PSP can push updates to the app
- Continues communication between the apps is necessary
- Two apps need to be downloaded

Merchant application + SDK
- No control over updates
- Single app needed
- No app switching

Merchant application + SDK + Application with user interface

- Mostly control with the app
- Visible app switching but done by the PSP
- Two apps needed

.



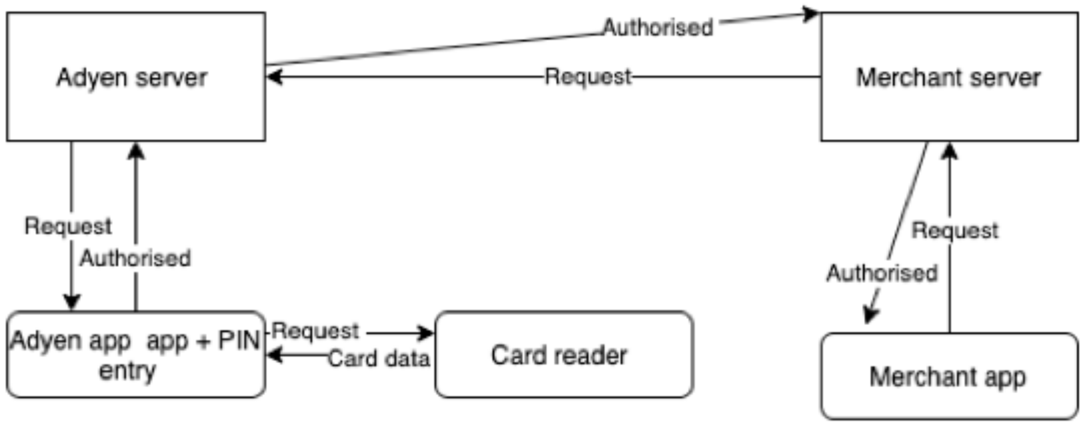
This scenario would have the merchant app receiving the PIN and sending it via a websocket connection. The Card reader would encrypt and send the raw Card data. The processing of the actual PIN and card data will then happen in the server.

This would still cause the PIN entry to be in the merchant app and therefore its security will be compromised.

It would be possible to have the app living fully in backend servers. A small app on the device will then only be a communication bridge between the data entry and the backend system.

This could be beneficial because the builder of the app is then less dependent on updates.

Parts in the app:
- Connection (Websocket) with the backend system
- Connection with the card reader
- General interface to provide the option of adding functionalities from the backend



An architecture in which our app would communicate with a backend system instead of with the merchant can increase the flexibility of the PSP. The flexibility is increased because this structure allows the majority of decisions to be made by the backend which in turn will be in control of the PSP. This part can also be updated without having to ask a user to update the actual application.

Changes in the merchant app will also have a limited effect because the app will first communicate with its own server which in turn will communicate with the PSP's server.

The difficult part of this structure is that the merchant app has to open the Adyen app for IOS because it cannot stay active in the background once a card reader is no longer in place. This means that the server cannot connect to the app until it is opened. This could cause some reliability issues. The server should keep asking response calls until the application is active. **Whether this can be solved is dependent on whether IOS changes their operating system such that applications can run and communicate in the background.**

This issue can already be solved by not using a second application and combining the PSP and the merchant application with the use of a SDK. As mentioned before, the downside of this solution is that the PSP is no longer able to control this which decreases the security aspect. By decreasing the functions that the SDK has and moving those functions to the PSP's servers this decrease in control will be limited.

Fully websocket

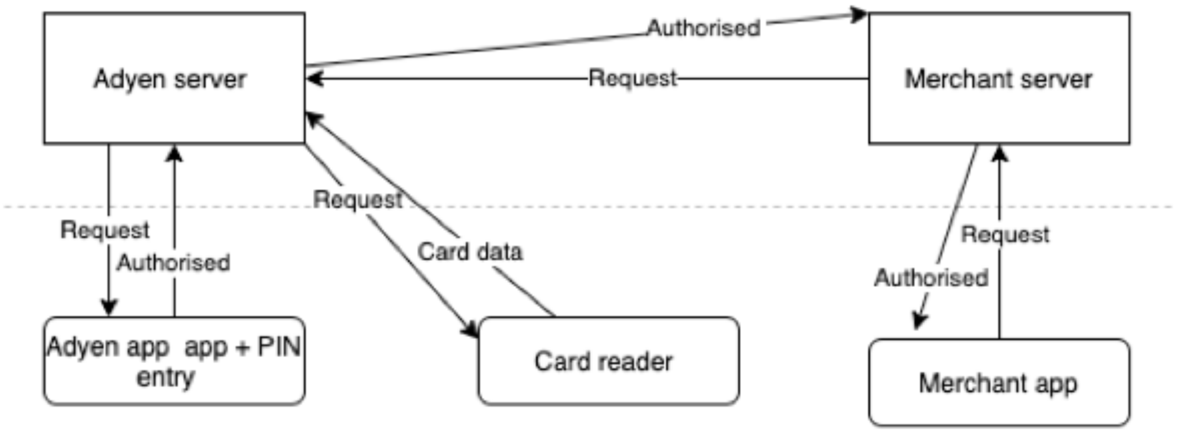
CON

- You would still have to communicate with the hardware, the card reader or the NFC reader. This can be difficult.

PRO

- No external updates necessary
- Merchants can make their own UI
- The system could run on various platforms without any significant changes

The difficulty of communicating with the hardware via Websockets could partly be solved by allowing the card reader to directly communicate via Websockets as well.



Although this could be a solution in which the user interface can be in control of the merchant while the backend system is fully in control of the PSP some other issues will arise. You wouldn't be able to use NFC and latency would increase significantly.

A solution could be that an application would be formed as a shill around the websocket/browser connection. In this scenario the application could take care of the native functions.

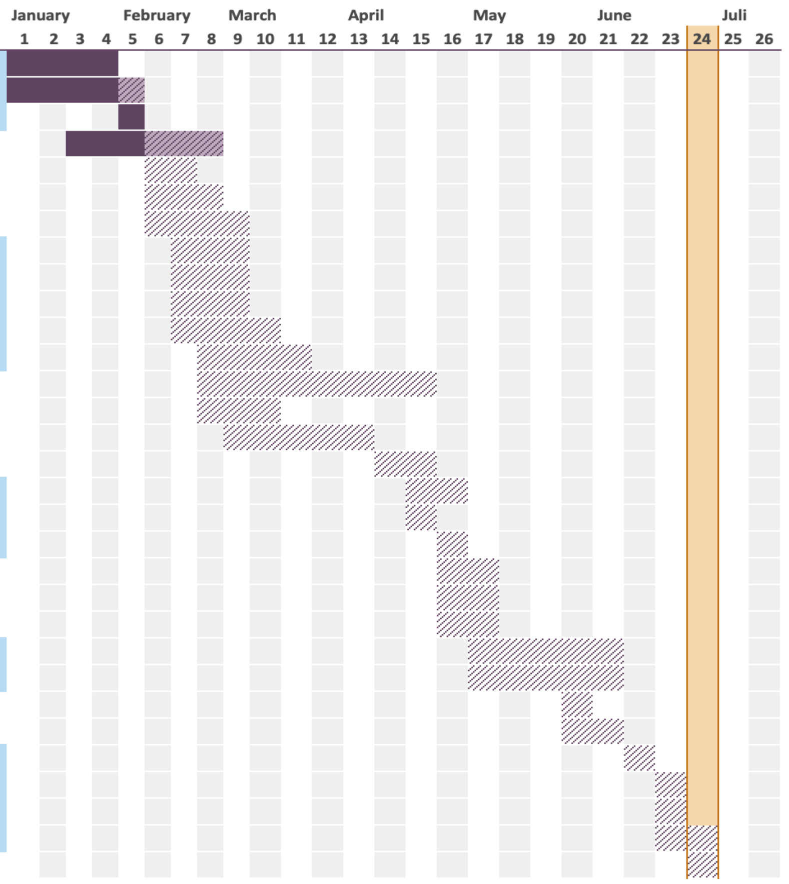
8.8 Appendix E Project Timeline

Project Planner

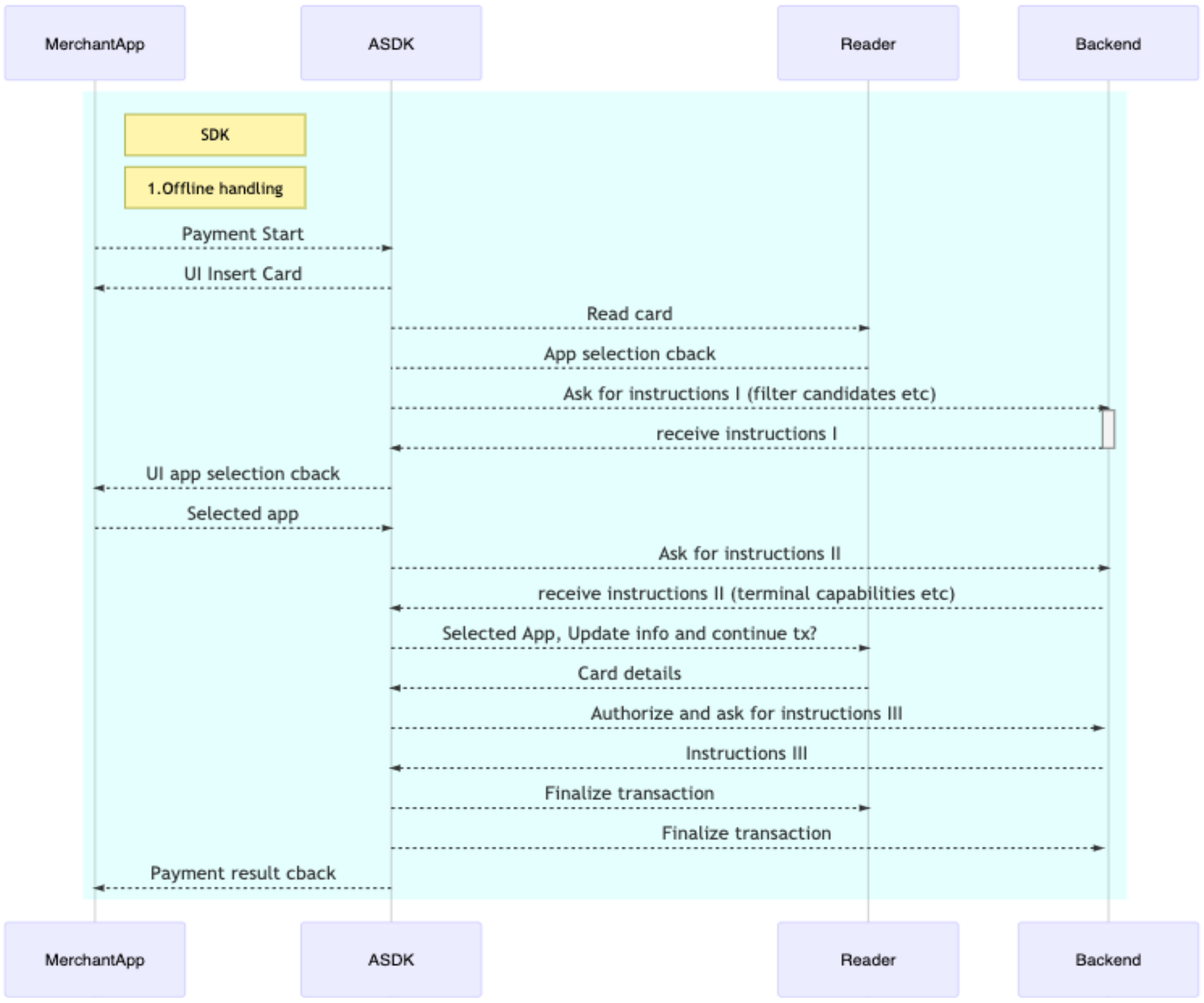
Vince Vissers Thesis

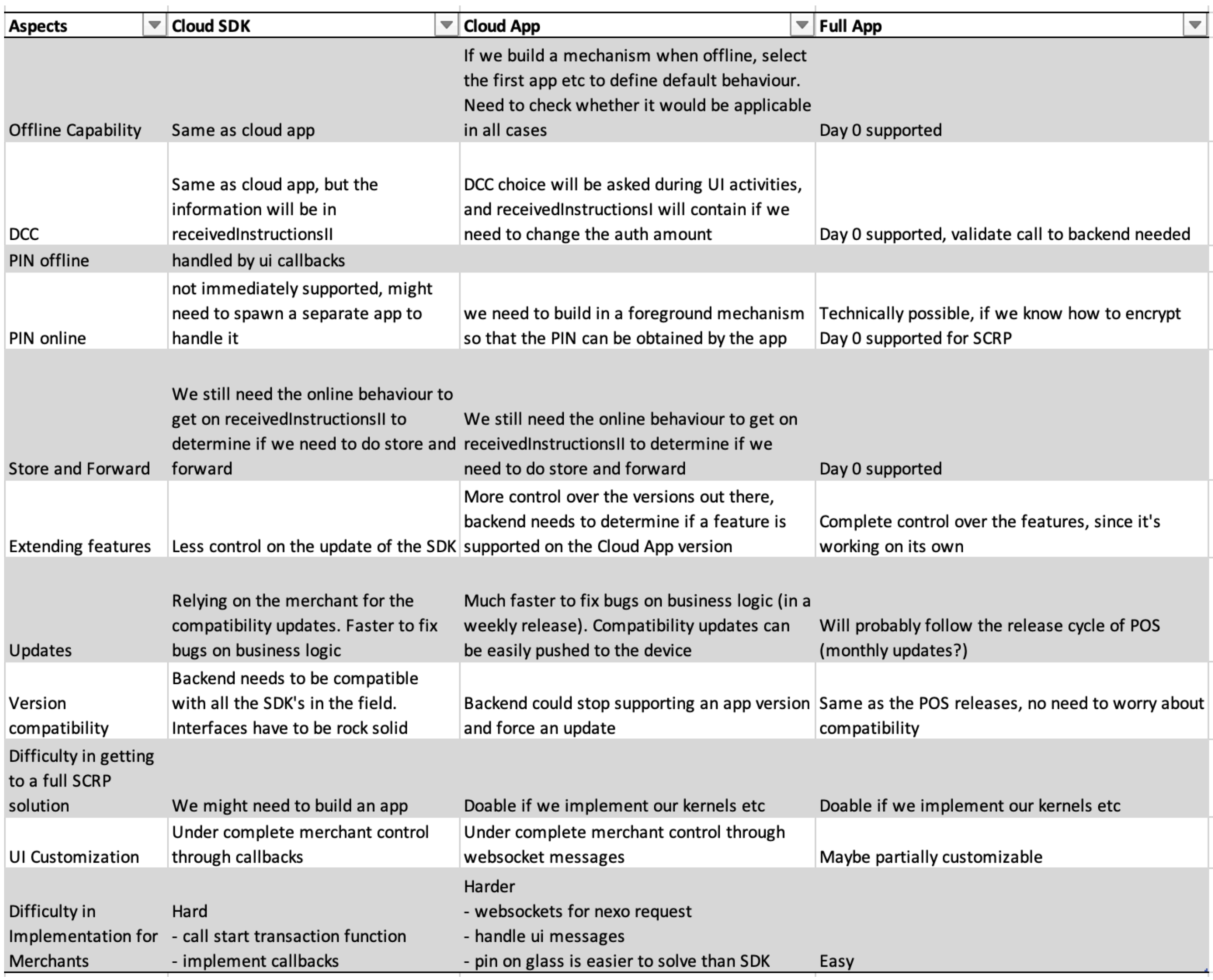
Period Highlight: 24 ▨ Plan Duration ▨ Actual Start ■ % Complete ▨ Actual (beyond plan)

ACTIVITY	PLAN START	PLAN DURATION	ACTUAL START	ACTUAL DURATION	PERCENT COMPLETE
Proposal writing	1	4	1	4	100%
Preparing the Kick off	1	5	1	5	80%
Kick-off meeting	5	1	5	1	35%
Identify Problem	3	6	3	6	50%
Revise understanding of the problem domain	6	2	0	0	0%
Revise the research approach	6	3	0	0	0%
Insert kick off meeting feedback	6	4	0	0	0%
Define the objectives of the Solution	7	3	0	0	0%
Elaborate on literature research	7	3	0	0	0%
Scope literature research	7	3	0	0	0%
Revise and rewrite	7	4	0	0	0%
Conclude research gap	8	4	0	0	0%
Design and development	8	8	0	0	0%
Plan expert interviews	8	3	0	0	0%
Conduct interviews	9	5	0	0	0%
Analyse interviews	14	2	0	0	0%
Demonstration	15	2	0	0	0%
Receive feedback pannel 1	15	1	0	0	0%
Receive feedback pannel 2	16	1	0	0	0%
Evaluation	16	2	0	0	0%
Include feedback	16	2	0	0	0%
Test design on PCI	16	2	0	0	0%
Communication	17	5	0	0	0%
Document design and conclusion	17	5	0	0	0%
End meeting	20	1	0	0	0%
preparing green light meeting	20	2	0	0	0%
Green light meeting	22	1	0	0	0%
Process feedback on GL meeting	23	1	0	0	0%
Thesis refinements	23	1	0	0	0%
Prepare defence	23	2	0	0	0%
Final presentation	24	1	0	0	0%

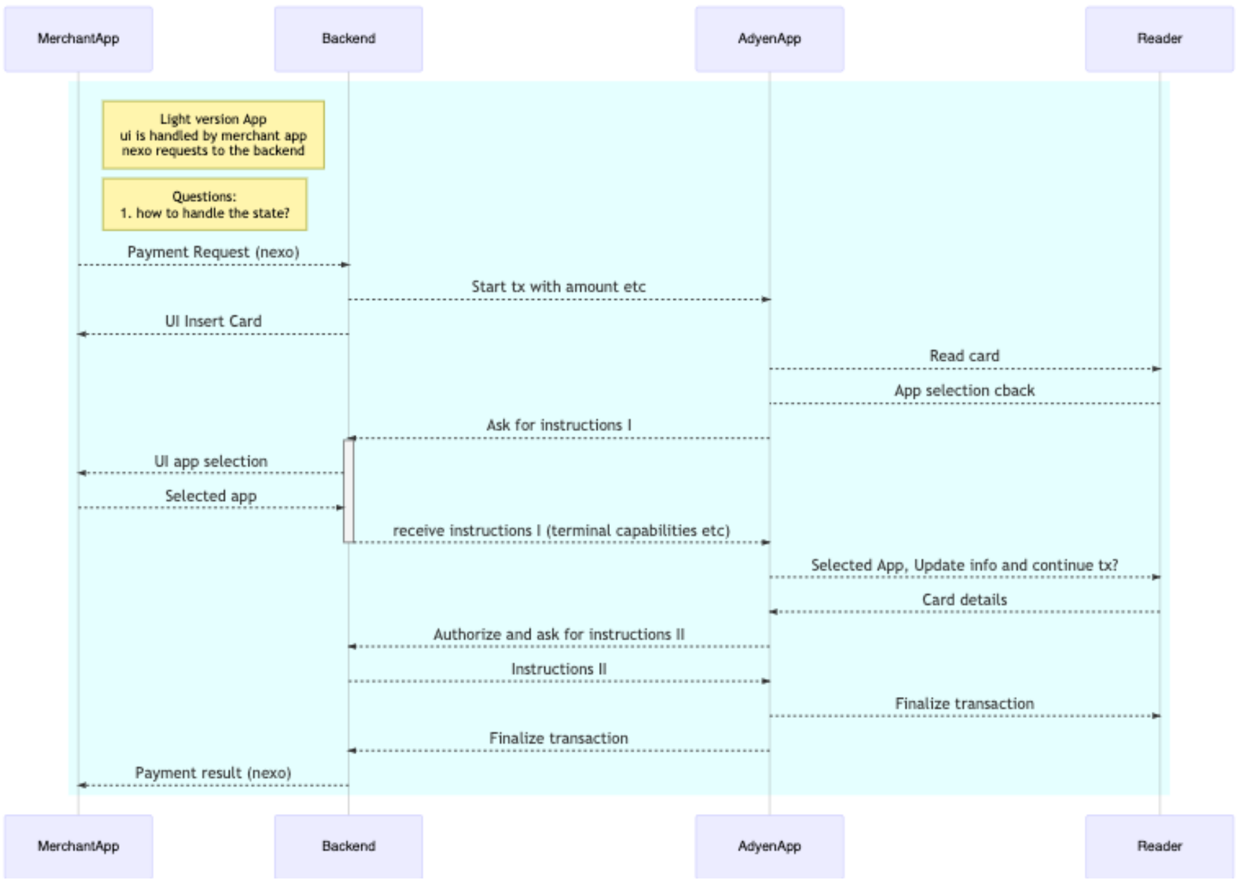


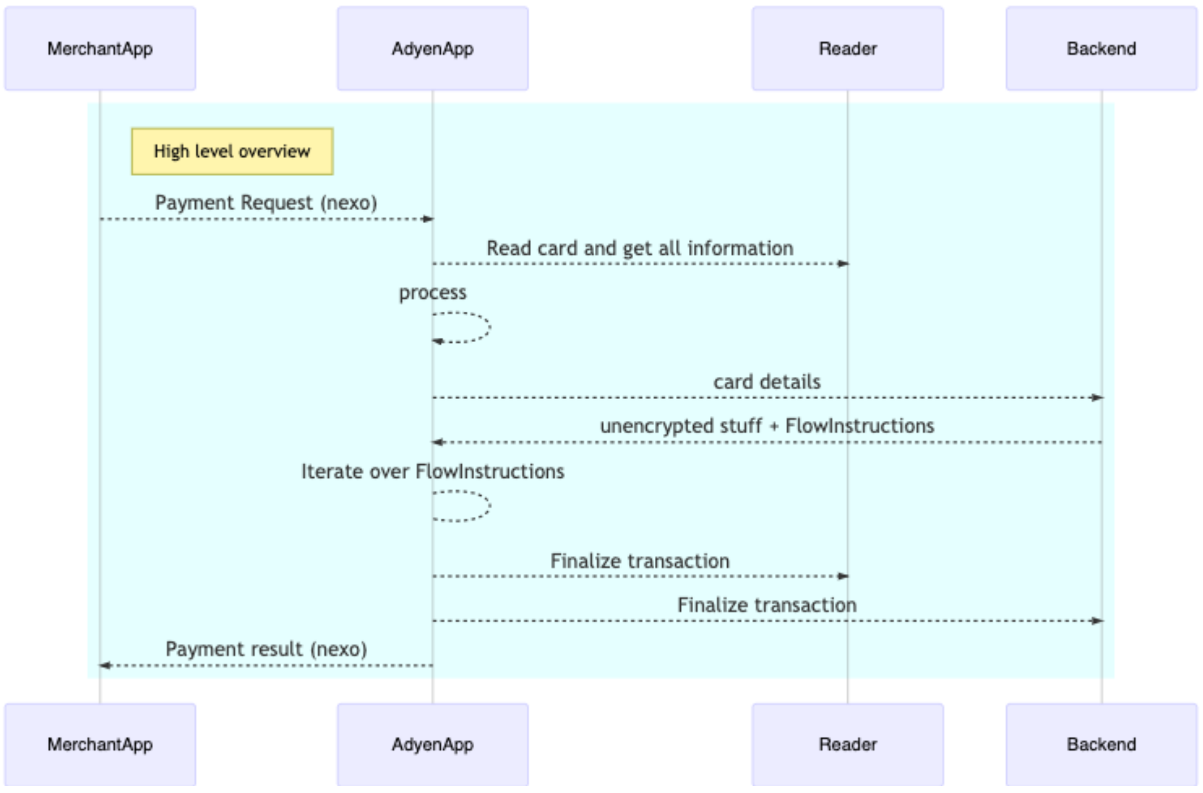
8.9 Appendix F Transaction flow differences between SDK and lightweight Application

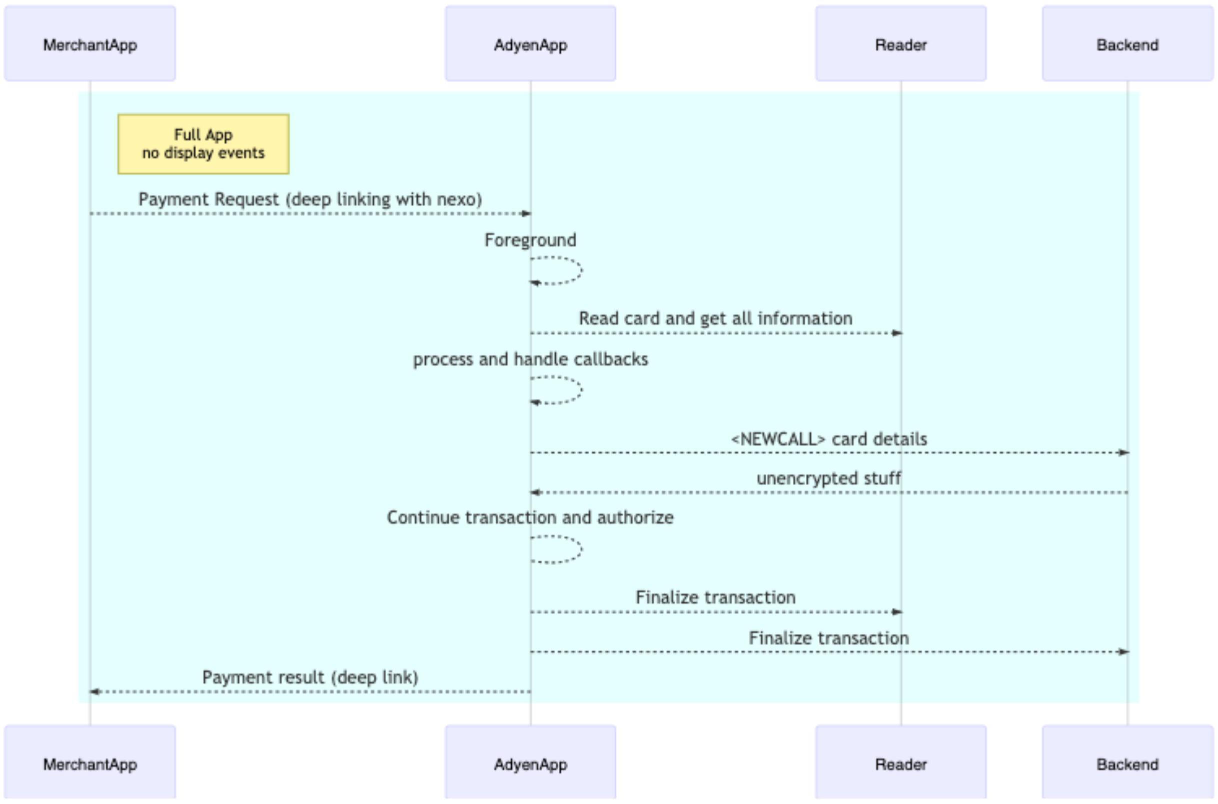




Aspects	Cloud SDK	Cloud App	Full App
Offline Capability	Same as cloud app	If we build a mechanism when offline, select the first app etc to define default behaviour. Need to check whether it would be applicable in all cases	Day 0 supported
DCC	Same as cloud app, but the information will be in receivedInstructionsII	DCC choice will be asked during UI activities, and receivedInstructionsI will contain if we need to change the auth amount	Day 0 supported, validate call to backend needed
PIN offline	handled by ui callbacks		
PIN online	not immediately supported, might need to spawn a separate app to handle it	we need to build in a foreground mechanism so that the PIN can be obtained by the app	Technically possible, if we know how to encrypt Day 0 supported for SCRP
Store and Forward	We still need the online behaviour to get on receivedInstructionsII to determine if we need to do store and forward	We still need the online behaviour to get on receivedInstructionsII to determine if we need to do store and forward	Day 0 supported
Extending features	Less control on the update of the SDK	More control over the versions out there, backend needs to determine if a feature is supported on the Cloud App version	Complete control over the features, since it's working on its own
Updates	Relying on the merchant for the compatibility updates. Faster to fix bugs on business logic	Much faster to fix bugs on business logic (in a weekly release). Compatibility updates can be easily pushed to the device	Will probably follow the release cycle of POS (monthly updates?)
Version compatibility	Backend needs to be compatible with all the SDK's in the field. Interfaces have to be rock solid	Backend could stop supporting an app version and force an update	Same as the POS releases, no need to worry about compatibility
Difficulty in getting to a full SCRP solution	We might need to build an app	Doable if we implement our kernels etc	Doable if we implement our kernels etc
UI Customization	Under complete merchant control through callbacks	Under complete merchant control through websocket messages	Maybe partially customizable
Difficulty in Implementation for Merchants	Hard - call start transaction function - implement callbacks	Harder - websockets for nexo request - handle ui messages - pin on glass is easier to solve than SDK	Easy


MerchantApp
Backend
AdyenApp
Reader
Light version App
ui is handled by merchant app
nexo requests to the backend
Questions:
1. how to handle the state?
Payment Request (nexo)
Start tx with amount etc
UI Insert Card
Read card
App selection cback
Ask for instructions I
UI app selection
Selected app
receive instructions I (terminal capabilities etc)
Selected App, Update info and continue tx?
Card details
Authorize and ask for instructions II
Instructions II
Finalize transaction
Finalize transaction
Payment result (nexo)
MerchantApp
Backend
AdyenApp
Reader


MerchantApp
AdyenApp
Reader
Backend
High level overview
Payment Request (nexo)
Read card and get all information
process
card details
unencrypted stuff + FlowInstructions
Iterate over FlowInstructions
Finalize transaction
Finalize transaction
Payment result (nexo)
MerchantApp
AdyenApp
Reader
Backend


MerchantApp
AdyenApp
Reader
Backend
Full App
no display events
Payment Request (deep linking with nexo)
Foreground
Read card and get all information
process and handle callbacks
<NEWCALL> card details
unencrypted stuff
Continue transaction and authorize
Finalize transaction
Finalize transaction
Payment result (deep link)
MerchantApp
AdyenApp
Reader
Backend