|  |
| --- |
| **Nordic Blockchain Semi-Centralized Distributed Ledger Technology for banks** |

**Luca Francioni (240068)**

**Supervisor: Michael Viuff**

**VIA University College - Horsens**

**[Number of characters]**

**ICT Engineering**

**7th Semester**

**Table of content**

[Abstract iv](#_Toc8645500)

[1 Introduction 1](#_Toc8645501)

[2 Domain Model 2](#_Toc8645502)

[3 Analysis 6](#_Toc8645503)

[3.1 Requirements 6](#_Toc8645504)

[3.2 Functional Requirements 7](#_Toc8645505)

[3.3 Non-Functional Requirements 9](#_Toc8645506)

[4 Design 11](#_Toc8645507)

[4.1 Entity structures 11](#_Toc8645508)

[4.1.1 Transaction 11](#_Toc8645509)

[4.1.2 Block 12](#_Toc8645510)

[4.1.3 Blockchain 13](#_Toc8645511)

[4.1.4 Node 14](#_Toc8645512)

[4.1.5 Encapsulation (Crypto Layered Message) 15](#_Toc8645513)

[4.1.6 Transaction Request Path 17](#_Toc8645514)

[4.1.7 Miner Confirmation Process 18](#_Toc8645515)

[4.1.8 Operator Statistics Request 19](#_Toc8645516)

[4.1.9 Genesis Block 20](#_Toc8645517)

[4.1.10 Serialization 21](#_Toc8645518)

[4.1.11 Overall architecture 22](#_Toc8645519)

[4.2 Choice of technologies 22](#_Toc8645520)

[4.2.1 Asymmetric Cryptography 22](#_Toc8645521)

[5 Implementation 25](#_Toc8645522)

[6 Test 26](#_Toc8645523)

[6.1 Test Specifications 26](#_Toc8645524)

[7 Results and Discussion 27](#_Toc8645525)

[8 Conclusions 28](#_Toc8645526)

[9 Project future 29](#_Toc8645527)

[10 Sources of information 30](#_Toc8645528)

[11 Appendices 1](#_Toc8645529)

# Abstract

Nordic Blockchain is a software service for banks, its purpose is to replace previous mechanisms and systems used in banks for transactions with a modern blockchain based one, reducing costs of operations and improving inter-banks data synchronization and trust.

The following report describes the analysis, design and implementation of such service but properly distinguishing the intended design and the resulting prototype. The project also raises technical and security aspects that can be subject to academic scrutiny and further development.

The appendix section contains the full code of the service; documentation towards the modular frameworks and components being used are described in the document.

The service is structured with the intent of being easily understandable, replicable and extendable.

# Introduction

The banks’ services have been alive and used for millennia, being the core structure of the worldwide economy development and actively participating in assets exchange, monetary value fluctuation, investments and everyday money exchange for services.

The transfer of assets of a bank’s customer to another bank’s customer is a process that involves several systems – either human or technology based - that must be properly configured and synchronized, offering as well adequate security and control for the bank’s administration, transparency and certainty for both the banks involved that the transactions are valid, secure and effectively requested by the originating customer: this may lead to slow downs and the overall system must always be available, scalable and stable.

The current systems involve years of software and hardware development and research, making a user operation fee based his/her desire of guaranteed speed of processing for the operation to be completed and the complexity of the transaction request path (e.g. international transactions), yet faster than physical writing and human elaboration, but still exposing the mechanism to exploits by malicious individuals (e.g. frauds, fake signatures, system penetrations, phishing, etc.).

Nordic Blockchain changes the way the whole underlying structure of a bank transaction management system work, unlocking unprecedented levels of scalability, speed of the operation and its costs, including security and transparency between multiple banks with absolute certainty of validity of the data by replacing the traditional systems with a modern blockchain design. Aiming to be the most independent and collaborative middle-ware system for transactions of assets and multiple possible operations that need to be synchronized between multiple banks in real time with least effort possible and maximum speed of transmission available; putting the storage control system software in each bank internal system and allowing banks to run undefined number of elaboration software components to process the data with fraud and intrusion detection algorithms, including already adopted ones.

The blockchain technology has a strong backward data validity, allowing intrusion or tampering of transactions almost impossible and immediately reported to the administration party, also allowing the banks to apply real-time monitoring for the operations inside the system and extending it with fewer and simpler workload and simplified testing, given the modularity nature of a blockchain; completely removing certain attack vectors, giving the chances only to phishing or physical danger.

The Nordic Blockchain is designed in a way that can be integrated, or be the replacement, to the current software systems without the bank’s customers notice as it mimics the same current functionalities while radically changing the mechanism behind it and the quality of life.

# Domain Model

The current monitoring and traceability system used by bank is SWIFT, it is a global cooperative owned software used for secure financial messaging services.

The software is from 1974 and is subject to millions of transactions requests to elaborate worldwide, considered the current best due to high security standards, better bandwidth usage and highly monitoring capabilities.

Such technology is, however, not using publicly known structure but is instead using RESTful APIs architecture, making the system especially centralized and closed and/or slowing the process of access another bank’s transactions logs.

The usage of a blockchain, a chain of “blocks”, containing transactions and being synchronized in real time by all the Nordic Blockchain’s network nodes (Figure 1), completely removes slowdowns of accessing certain essential information from the monitoring systems and significantly speeds up the process of a transaction request, as well as offering flexibility for temporary offline nodes, that only have to download latest version of the blockchain to get back to operative status.

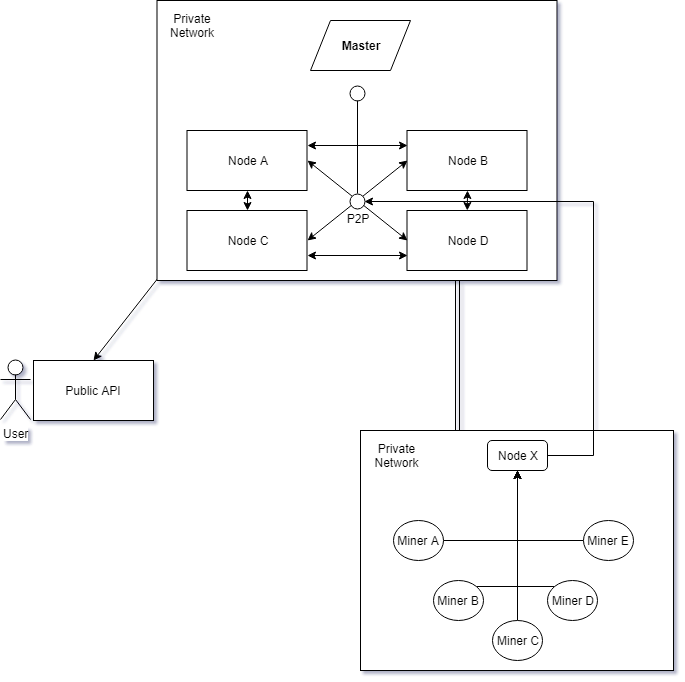


Figure 1 - The overview of a Nordic Blockchain network

In a blockchain, the key role of the bank becomes the node, the node is a copy of the blockchain that continuously synchronizes its content over the time, becomes the single bank’s internal transaction management system to which its customers transactions pass through and queued in the miner’s pending operations.

The node is also the internal monitor system, as it checks the validity of the transaction requests’ origin before being inserted into the queue.

The nodes are not able to communicate to each other directly and transactions aren’t immediately sent to the destination bank specified by the user.

This introduces the known figure of the miner: its original schedule – using Bitcoin as comparison reference – is to discover new block trough specific cryptographic operations, where new transactions are going to be stored between the nodes; and the verification process that the transaction is valid and trustworthy.

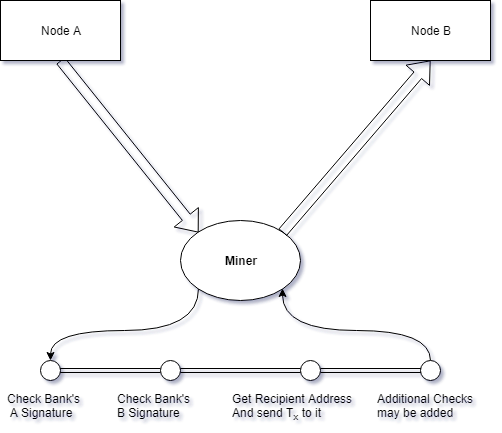


Figure 2 - The miner's process of verification of a transaction

In Nordic Blockchain, the miner’s role remains similar, yet different, as the miner’s objective is to verify the validity of the origin and destination node of the transaction and verify the validity of the transaction content itself without knowing its content (Figure 2), as the miner is being deployed by the bank, and not independent users.

Trough this process, secrecy of the content (such as asset value) is not disclosed but its trustworthy is verified.

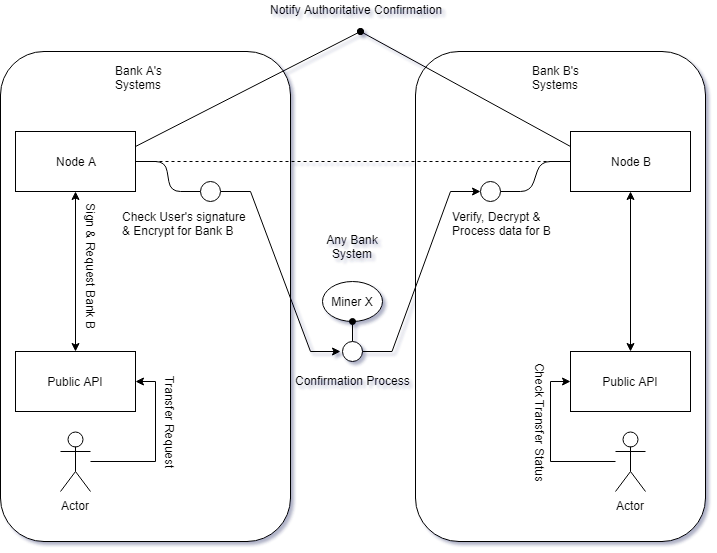


Figure 3 - Miner's macro-role in the Nordic Blockchain network

When a miner ends its verification process, a verdict is sent to the Nordic Blockchain network by increasing the transaction’s synchronized counter by 1.

Once the confirmation counter of a transaction reaches a fair amount, the transaction is to be considered trustworthy, hence, confirmed and concluded (Figure 3).

The process above described may appear long, but given the correct setup of the nodes and resources allocated to the software and enough miner being run by the collaborative network, the transaction time can be drastically cut to minutes (internationally) and the costs of the transaction fees is reduced as well, due to a significant cut of bureaucracy processes that can now be slightly adapted to a natively semi-transparent system; therefore the more miner are deployed in the internal network the faster the transactions are processed, unlocking a significant scalability solution by simply deploying more instances on running systems.

As of the monitoring, the miner’s process can be easily customized by adding custom steps, these can be also the current existing monitoring platforms, allowing perfect modularity and extension of the whole Nordic Blockchain platform with little to no risk of unintentional sabotage/misconfiguration and an easier migration from previous platforms to Nordic Blockchain.

# Analysis

## Requirements

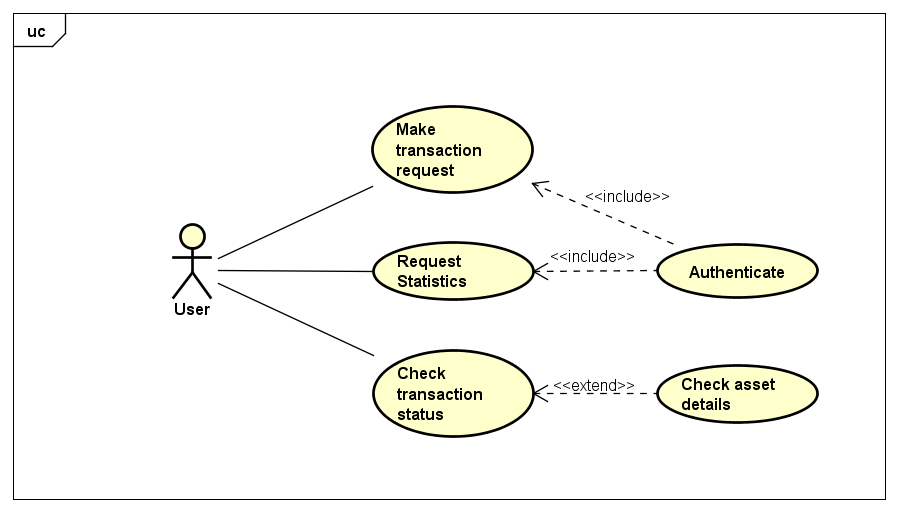
The following are the requirements adopted for this project.

The involved users in the various requirements are mainly of three types:

* **User** – A bank customer, average user that has no knowledge of what is happening behind the scene.
* **Bank Operator** – A bank employee dedicated to the project, depending on its goal.
* **Miner** – An automated system for transactions confirmation, that ensures security and validity.

## Functional Requirements

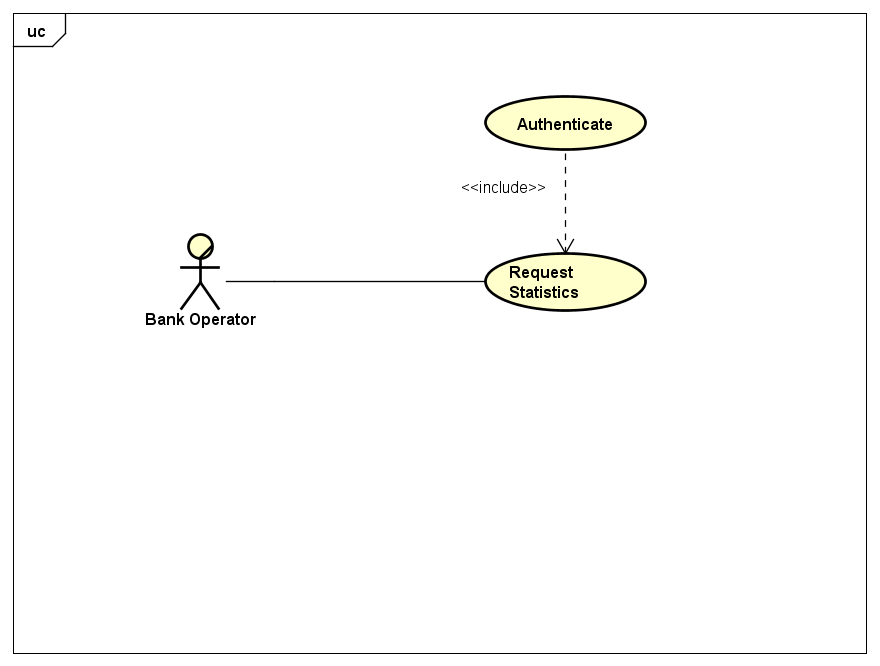
1. User must be able to interact with transactions.



A user must be able to create, show and gather information of his/her transactions or the ones he/she is receiving.

* A transaction can be created by the user:
  + The transaction request transmission should happen within 2 seconds after compilation and confirmation of the operation.
  + If the transaction is not signed by the user’s private key, reject the transaction request.
  + The transaction must be secured before transmission to the other nodes:
    - The transaction must be signed, once transmitted to the user’s bank node, by the bank’s private key; within 1 minute.
    - The transaction hash must be encrypted with the receiver’s bank public key and the sender’s private key.
* The transactions status can be queried any time by the user:
  + The query can be customized by specifying time span, author or receiver identification.
  + The resulting data must be transmitted to the requesting user within 10 seconds from the request.
  + The request must be secure by signing it with the user’s private key.
* The user can check the status of a specific transaction:
  + The request must specify the transaction identifier.
  + If the request is signed by the author’s or receiver’s private key:
    - The response data must contain also asset transfer details (amount).

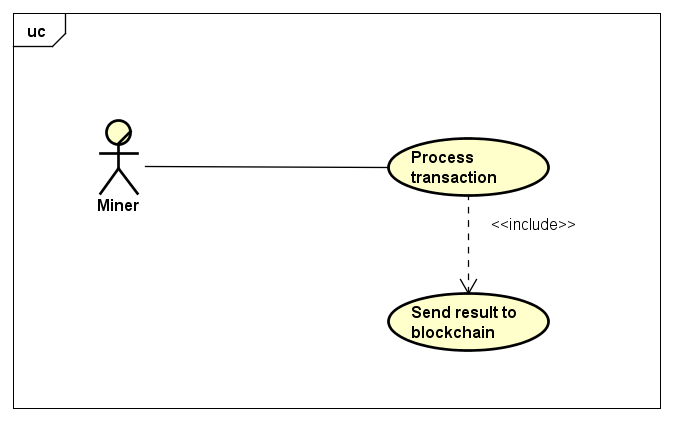
1. Blockchain statistics must be available to operator



A bank operator can query statistics of its bank node:

* The request must be signed by the bank operator’s public key, reject otherwise.
* The response of the query must happen within 10 seconds.
* The response must be encrypted with the operator’s public key.

1. Transaction confirm



The miner must be able to confirm the transaction and increase the confirmation status counter by one:

* The miner must be able to retrieve the latest transactions every 1 minute.
* The transaction request must be elaborated, and a verdict must be chosen within 1 minute a transaction request is picked.
* The verdict must be transmitted to the bank node of the receiving user within 30 seconds the transaction has been elaborated.

## Non-Functional Requirements

1. **Performance**

The Quality of Service offered by the software strictly depends on the performance of the system the bank decides to install.

* 1. Network – It is heavily encouraged to monitor bandwidth usage as a blockchain may have lots of network operations.
  2. Virtual space – The fork of a blockchain may become bigger and bigger by the time, virtual space is the main resource that should be scalable.

1. **Operating constraints**

It is required by the whole bank systems to be connected to a common Virtual Private Network for the broadcasting to work properly, ensuring also additional security and layer of isolation.  
The cryptographic fingerprinting is to be exchanged by human entities.

1. **Accuracy and precision**

The cryptographic fingerprints must be exact to the ones being used, a different key fingerprint will result in the rejection of the node from the network.

1. **Security**

The system must use known and approved cryptography mechanisms to guarantee security of the blockchain, the current asymmetric cryptography algorithm is RSA, can be upgraded to ECDH for better security measure.  
It is also necessary for production usage a layer of security for network communication protocol.

# Design

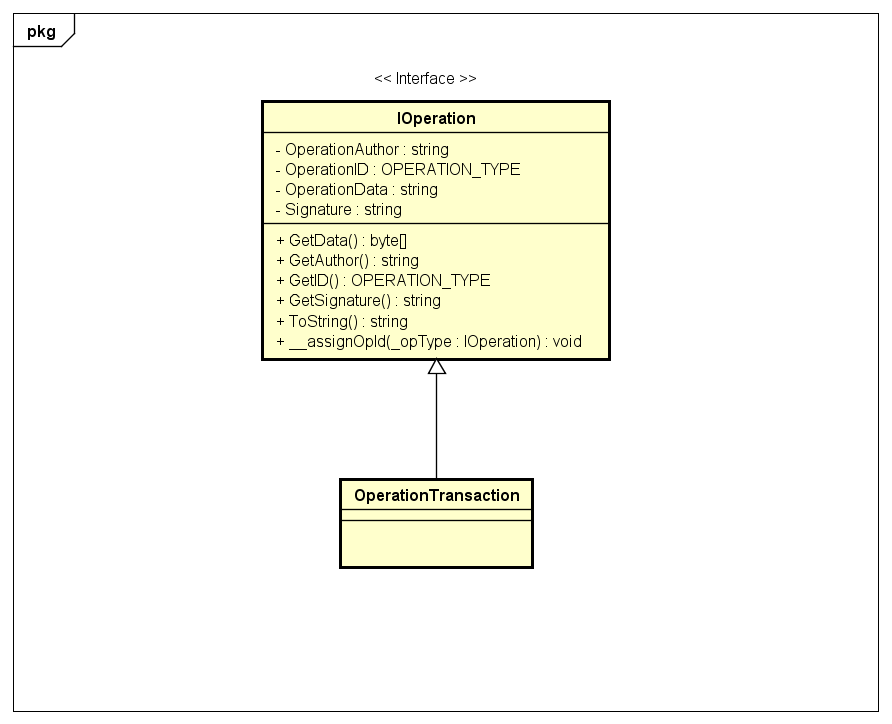
The following section describes the main components of the software, highlighting relevant information for comprehension of the subject and the mechanism of functioning.

## Entity structures

### Transaction

The transaction is the entity containing vital information for the asset transfer from a user to another, containing:

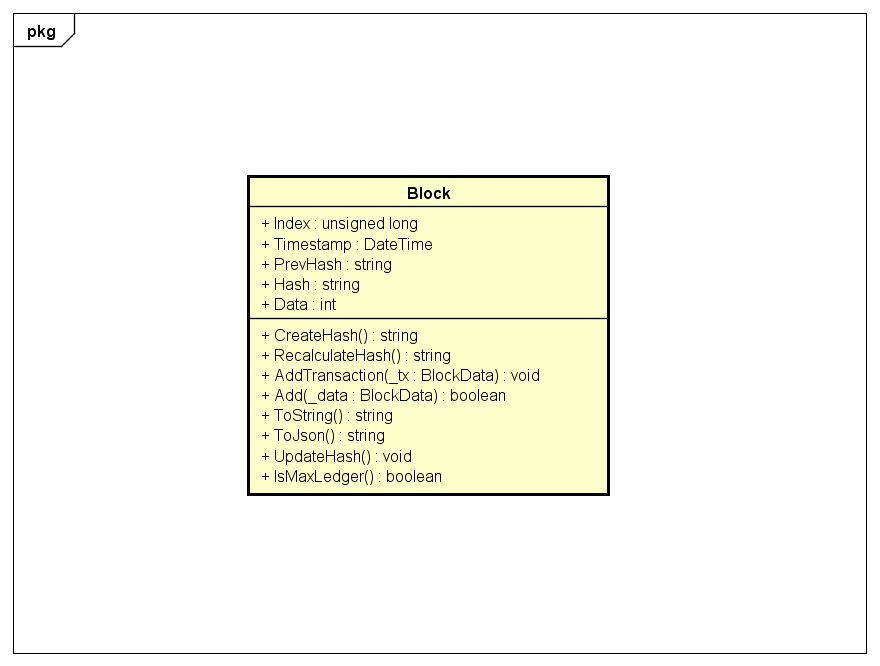
* **Sender Address** – The sender’s address (identifier).
* **Receiver Address** – The receiving user’s address (identifier).
* **Asset Value** – The value of the asset being transferred (e.g. money).



### Block

The block is the container of transactions being synchronized between multiple nodes, the block can be of two types:

* **Pending Block** – The current block being filled with confirmed transactions, still not synchronized between all the nodes and it is still editable.
* **Stored Block** – The block is stored into the blockchain and locked for modifications, it is now ready to be synchronized between multiple nodes and become part of the blockchain.

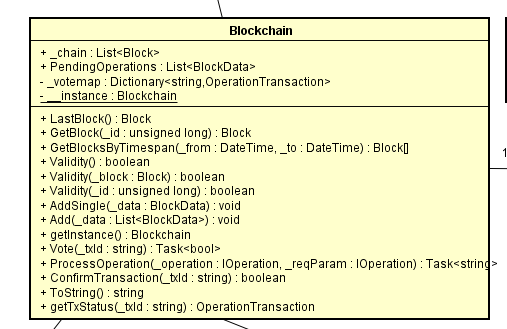


### Blockchain

The blockchain is the combination of stored blocks linked together and it’s the whole tree that composes the history of the confirmed operations inside the network.

A blockchain may be reset (wiped clear) to a completely new one for virtual space usage optimization (and archiving backups), the version of a blockchain is called “Fork”.

The blockchain is structured in the following way:



### Node

The node is the central unit a bank uses for synchronization and storage of pending transactions originated by its own customers (users).

Every bank can have one or more nodes.

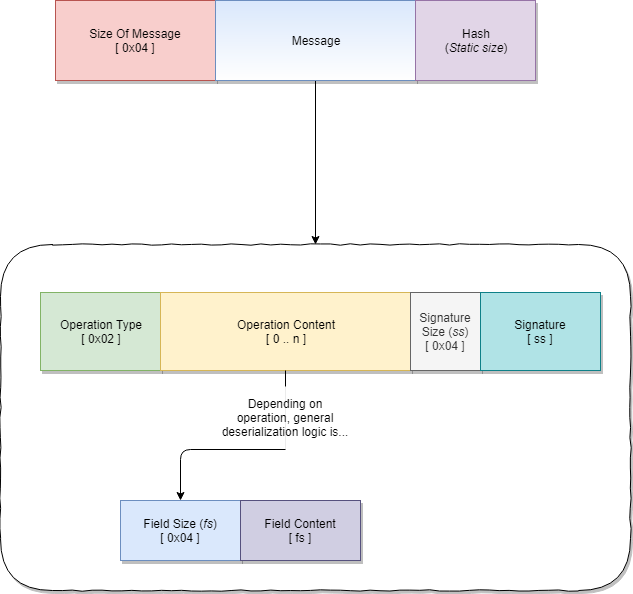
The node is structured by:

* Its own copy of the blockchain, that can be synchronized with other nodes.
* Its own customers pending transactions (awaiting miner’s confirmation).
* Transactions confirmation counters.

### Encapsulation (Crypto Layered Message)

To ensure security and obscurity of vital information, the data being sent would be properly encrypted and signed through asymmetric encryption algorithms.

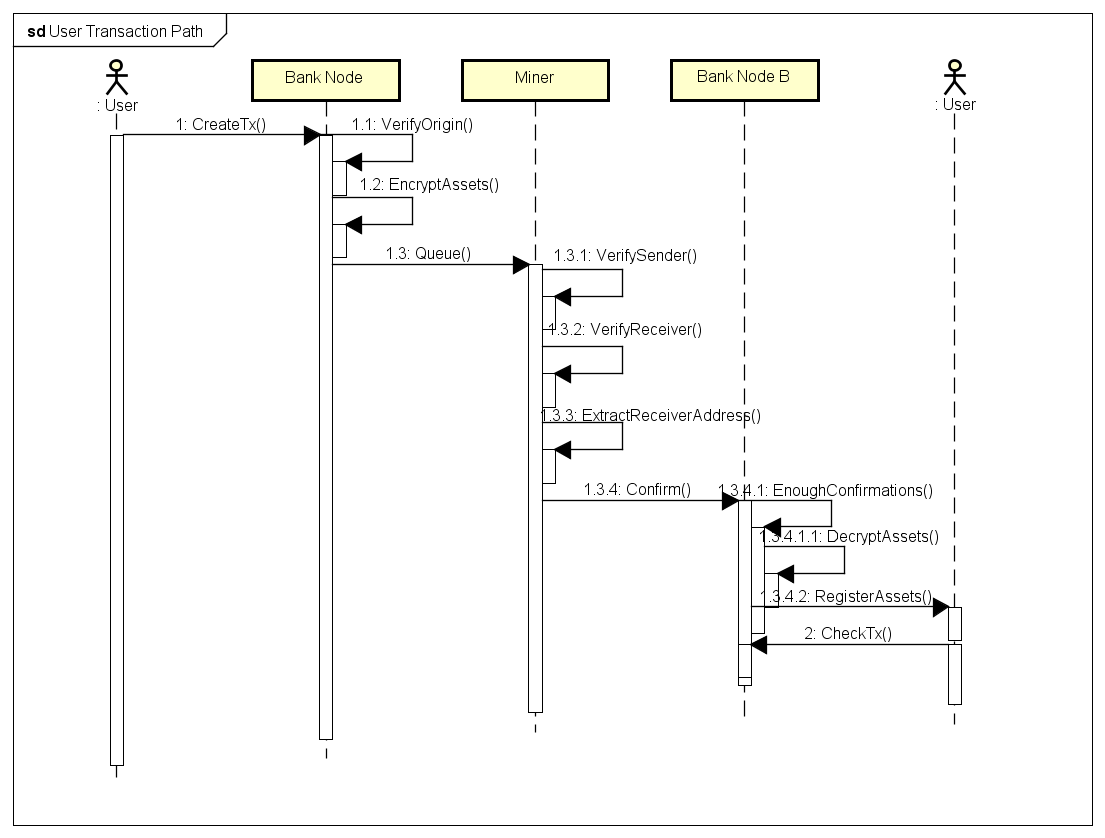
The encapsulation message is the container of the information being sent and the protocol to follow for correct requests and responses; structured in the following way:



The message’s buffer is **not** to be compressed but only encrypted, for security reasons (Refer to “CRIME” or “Compression Oracle”).

Nordic Blockchain already implements AES as block cipher for communication encryption, but the class is not being used for this version of the project.

### Transaction Request Path



A transaction follows a long path of processing before being registered permanently for its scope.

The process can be divided into 3 main phases:

1. **Acceptance Phase**

The transaction request must be created on first place, its origin must be verified to avoid fraud; once authenticated, the request is to be marked valid, encrypted and enqueued into the pending operations, awaiting miner’s notice.

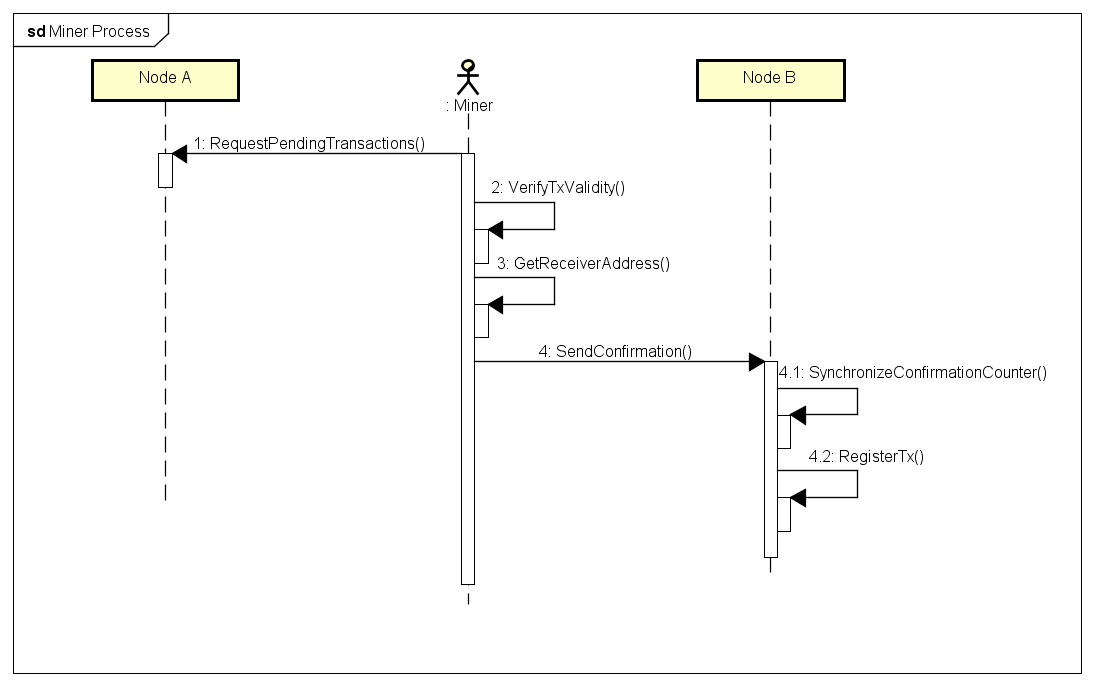
1. **Transmission Phase**

Once the transaction is retrieved from the queue by the miner, the miner has to verify certain aspects of the transaction to be marked definitively acceptable (and therefore “voting” for valid) (Refer to “Section 4.3”) by the miner and finally trace the receiver’s node, to which the notice is transmitted.

1. **Processing Phase**

If a pending operation reached enough confirmations, the node can process its content and apply its actions permanently, registering the transaction into the blockchain’s latest block available.

### Miner Confirmation Process



The miner’s duty is to be considered fundamental for the verification process of a transaction and its actuation.

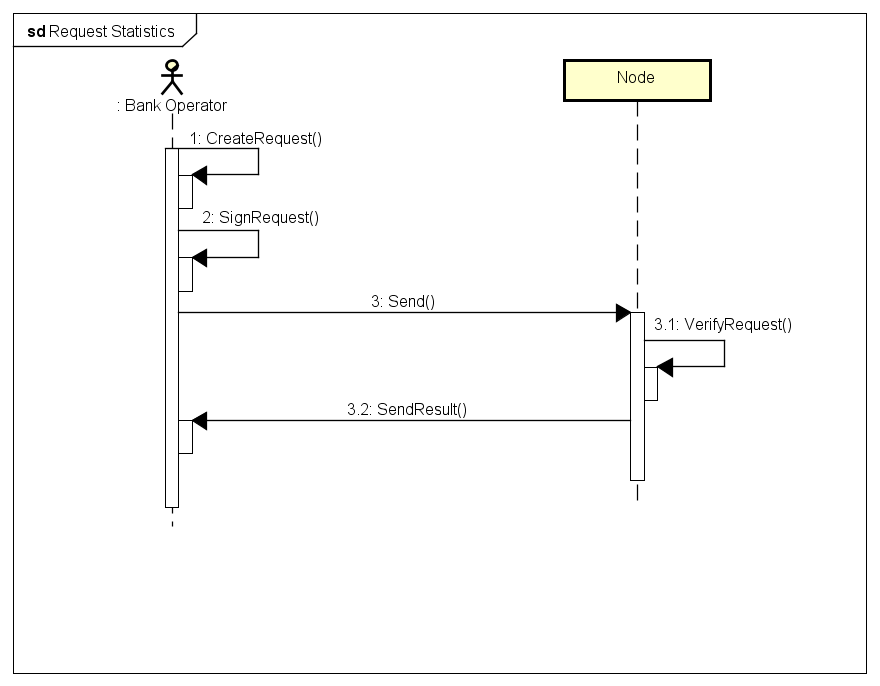
Once a transaction is retrieved by the pending operations, the transaction’s validity comes from the signatures included in the transaction request.

If the transaction signature is by the originating bank (and therefore the bank certified the origin of the transaction is by an authorized user), the miner must recover the receiver’s address and appropriate node(s) to send the vote to (in this project execution, only the one node is online).

The vote is also registered as an operation by implementing “IOperation” interface and will be registered inside the transaction “confirms”, in the block.

Whenever the transaction’s confirms reach the maximum necessary to be considered definitive, the transaction is automatically put into the latest available block and registered permanently.

### Operator Statistics Request



While the previously described operations may appear complicated, the statistics request is a simpler function used as a debug feature for the communication of the node and the content of the blockchain.

If the node recognizes authorization by the requesting’s signature, the following statistics are sent back:

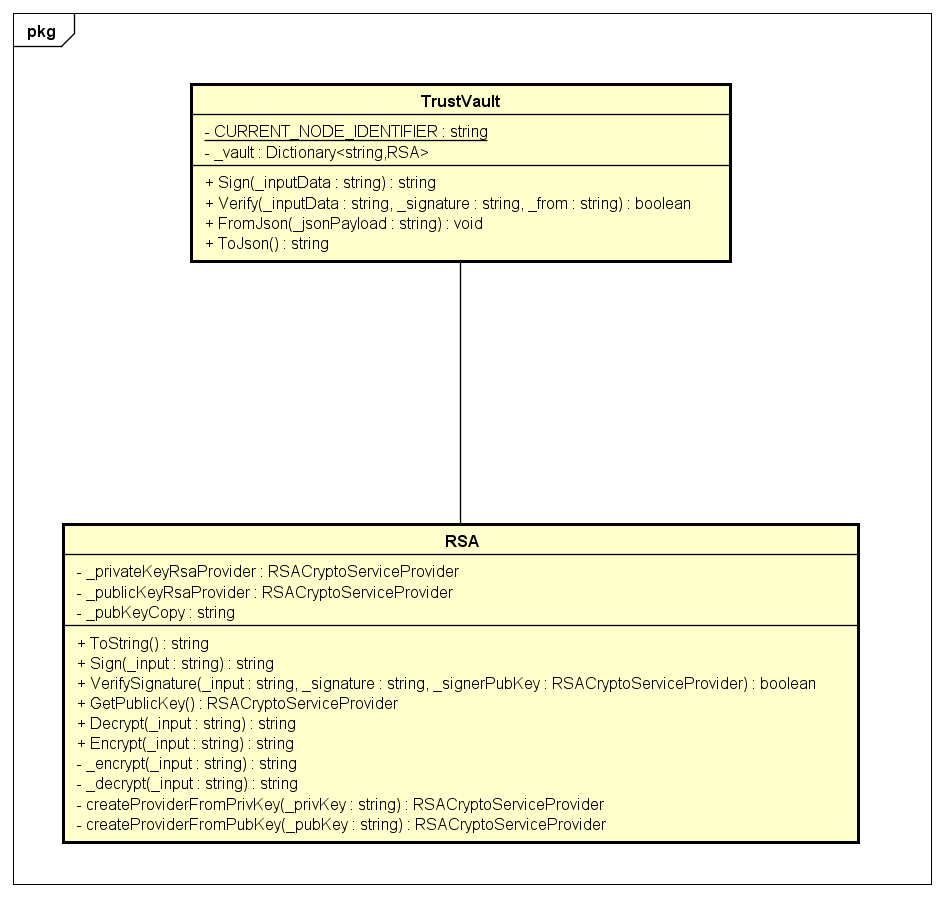
* Pending operations remaining
* Latest block information and content
  + Creation date
  + Containing transactions
  + Containing operations in general

### Genesis Block

The genesis block is the first block created in the blockchain that will start the succession of the other blocks.

Its presence is essential as it ensure the second block (first real-usage block) has the necessary security measures (Hash) coherent from the start.

In this implementation of the Nordic Blockchain, the genesis block is automatically created with non-significant content.

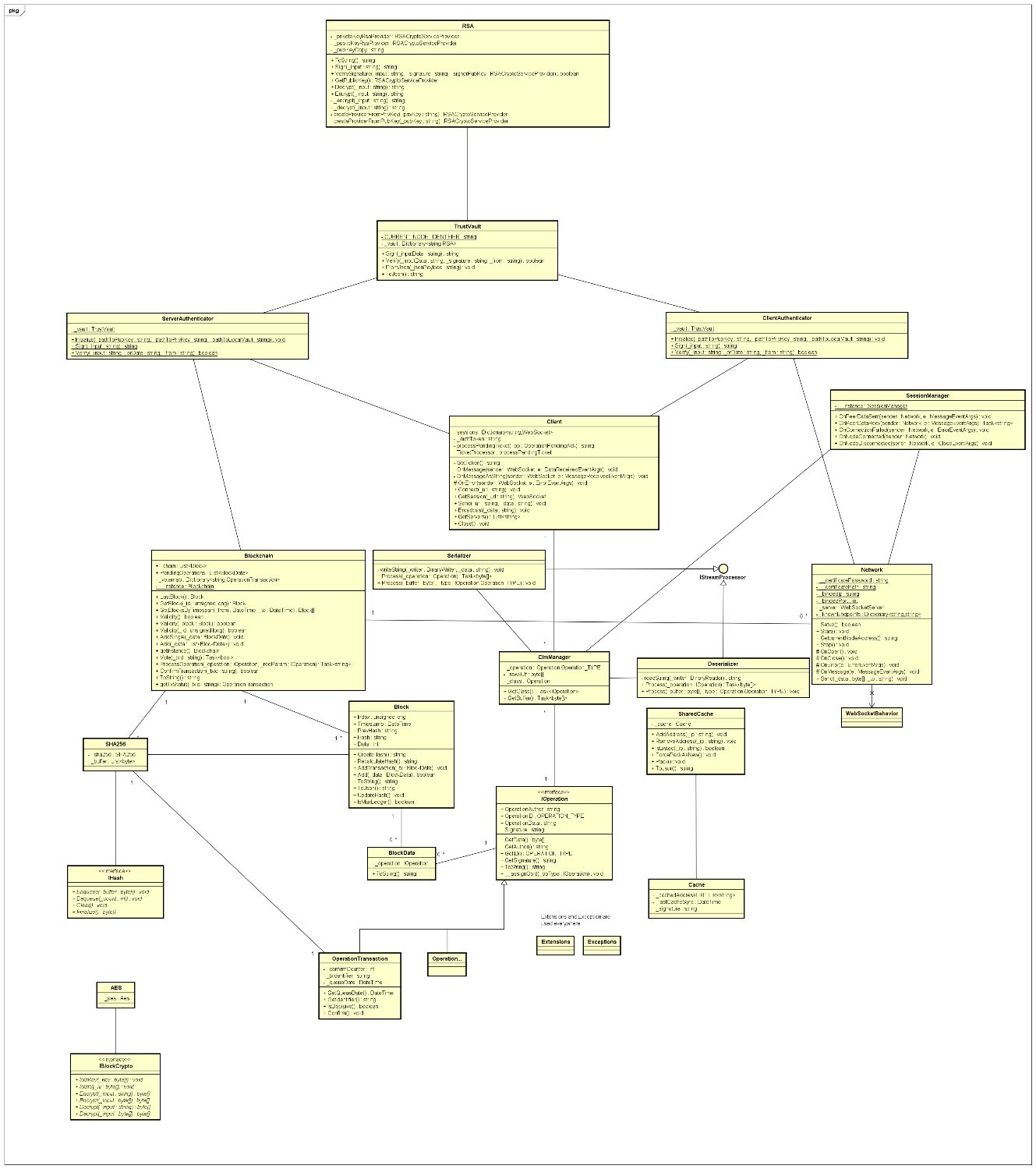
In the project, the genesis block is the deposit of all the public keys of the nodes with relative owner, represented by a plain Block containing a single transaction, of which content is the “Trust Vault” structure.

### Serialization

As the Crypto Layered Message (CLM) is a binary formed information exchange, a proper serialization and deserialization class is required for transmission over the network.

The “ClmManager” class offers such support using templates (Generic) for auto-determination of serialization or deserialization process to employ for the designed class structure.

### Overall architecture



## Choice of technologies

### Asymmetric Cryptography

The project and its security heavily rely on asymmetric cryptography to authenticate and encrypt the data received and sent, as well for authorization levels determining and origin verification.

While an alternative to RSA is ECDSA and is even newer technology, the choice of RSA in this project is purely for simplicity, compatibility coverage and accessibility as well as its maturity.

RSA is a well-known asymmetric cryptography algorithm, created in 1977 compared to ECDSA that has been proposed in 1992; thus, ECDSA is newer and its performance and time-complexity is supposedly better than RSA, despite that, RSA has better performance due to longer time of improvements.

While ECDSA offers better scaling capabilities and bigger key sizes, its vulnerability to Shor’s algorithm is considered weaker than RSA, that offers better resistance to quantum-based attacks, features longer maturity and a long list of adoptions in a huge variety of software, both FOSS and private.

For the above-mentioned reasons, the Nordic Blockchain’s team decided to adopt RSA for simplicity and extended documentation coverage compared to ECSA and a multitude of available APIs covered by different licenses, mainly FOSS-oriented.

### WebSocket

The communication classes (Network, Client, SessionManager) use WebSockets for communication protocol instead of plain sockets why a custom protocol.

The choice of WebSockets is to put a baseline for compatibility with multiple applications, including external ones, that desire to use such service, since the WebSocket’s transportation protocol obeys under the HTTP standards and also supports SSL.

Also, the usage of standardized protocol and technologies is always preferred over custom implementations that might deteriorate over time and decrease maintainability.

The purpose of the design section is to outline HOW the system is structured; i.e. to transform the artefacts of the analysis into a model that can be implemented. The design section is relevant for the programmer, whereas the analysis is relevant for the stakeholder.

Elements that may be relevant in this section:

* Architecture: Find architecture patterns here (Leszek Maciaszek 2004, chap.9).
* Technologies: Describe technologies used, also alternative technologies. Argue for choice of technology according to the project aim.
* Design Patterns: Describe which design patterns (GoF (Gamma et al. 2002) etc.) you are using and why.
* Class Diagrams
* Interaction Diagrams
* UI design choices
* Data models, persistence, etc.

You must explain all diagrams in the report. These diagrams including descriptions are the blueprints for the implementation.

Hint: One way to figure out which objects/classes are needed in the design is to apply the General Responsibility Assignment Software Patterns/principles (GRASP) (Larman 2004, chap.17).

Hint: Consider how to design your system to make it testable.

# Implementation

The purpose of the implementation section is to explain interesting code snippets. An idea is to explain the complete path through your system from UI to database etc.

Remember that your implementation must be consistent with your design (Larman 2004, chap.20).

Which standard libraries are used? How are design patterns implemented, etc.

Hint: Implement your code in a testable manner.

# Test

The purpose of the test section is to document the result of your testing; to verify if the content of the requirements section has been fulfilled. How is the system tested, which strategy has been used; e.g. White Box (Unit Test), Black Box, etc.

## Test Specifications

For functional requirements, test specifications must be listed. These test specifications can be described as soon as the functional requirements have been completed (Use Cases including descriptions).

IEEE can be used as a template for test specification (IEEE Computer Society 2008). VIA Library can give you access to this standard.

# Results and Discussion

The results are satisfying for the functional requirements, as the project is structured as a Proof of Concept.

Certain aspects such as encryption and the extended usage of asymmetric cryptography could not be finished within the time limit due to the complexity towards the functional requirements themselves.

Overall, the code is structured in a maintainable manner and with a central library functioning as base dependency for all the Nordic functionalities, allowing changes towards the base instead of the whole applications.

The cryptography interfaces offer a valid base for future implementations of cryptography instead of intervening directly into the code delegated to communications.

The wallet system (client/customer identification) is a missing feature, but easily addable.

# Conclusions

The purpose of the conclusion section is to compile the results from each section in the report. What is the conclusion? Did the project fulfil the requirements? Etc.

You can only comment on report contents, no new topics or content can be introduced in this section.

# Project future

Reflect on your project from a technical viewpoint and describe what you would change if you could.

Suggest how the project could be improved or made ready for production. Discuss scalability, suggest possible spin offs, what is needed, missing, etc.?

# Sources of information

**Note: Use the standard reference method: Harvard Anglia. A very good reference tool is Mendeley** (Mendeley.com 2016), **ask VIA Library if you need help.**

Banger, D., 2014. A Basic Non-Functional Requirements Checklist « Thoughts from the Systems front line.... Available at: https://dalbanger.wordpress.com/2014/01/08/a-basic-non-functional-requirements-checklist/ [Accessed January 31, 2017].

Business Analyst Learnings, 2013. MoSCoW : Requirements Prioritization Technique — Business Analyst Learnings. , pp.1–5. Available at: https://businessanalystlearnings.com/ba-techniques/2013/3/5/moscow-technique-requirements-prioritization [Accessed January 31, 2017].

Dawson, C.W., 2009. *Projects in Computing and Information Systems*, Available at: http://www.sentimentaltoday.net/National\_Academy\_Press/0321263553.Addison.Wesley.Publishing.Company.Projects.in.Computing.and.Information.Systems.A.Students.Guide.Jun.2005.pdf.

Gamma, E. et al., 2002. *Design Patterns – Elements of Reusable Object-Oriented Software*, Available at: http://books.google.com/books?id=JPOaP7cyk6wC&pg=PA78&dq=intitle:Design+Patterns+Elements+of+Reusable+Object+Oriented+Software&hl=&cd=3&source=gbs\_api%5Cnpapers2://publication/uuid/944613AA-7124-44A4-B86F-C7B2123344F3.

IEEE Computer Society, 2008. *IEEE Std 829-2008, IEEE Standard for Software and System Test Documentation*,

Larman, C., 2004. *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development*,

Mendeley.com, 2016. Homepage | Mendeley. Available at: https://www.mendeley.com/ [Accessed February 2, 2017].

YourCoach, S.M.A.R.T. goal setting | SMART | Coaching tools | YourCoach Gent. Available at: http://www.yourcoach.be/en/coaching-tools/smart-goal-setting.php [Accessed August 19, 2017].

# Appendices

The purpose of your appendices is to provide extra information to the expert reader. List the appendices in order of mention.

Examples of appendices

* Project Description
* User Guide
* Source code – source documentation
* Diagrams
* Data sheets
* Etc.

**Appendix A Project Description**

Insert the original Project Description here