|  |
| --- |
| **MARSE**  **SEP2 S18 – SOFTWARE DEVELOPMENT** |

****

**Mihai Draghiciu, 238971**

**Eric Volmer, 273448**

**Andrei Mungiu, 273473**

**Ronalds Andris Kalnins, 260506**

**Supervisor: Troels Mortensen, Bo Brunsgaard Christensen, Ib Havn**



**[18.684 characters]**

**Software Engineering**

**2st semester, Winter 2018**

**19-Dec-2019**

# Table of content

[1 Table of content ii](#_Toc532900140)

[2 Introduction 1](#_Toc532900141)

[3 Group Description 3](#_Toc532900142)

[4 Project Initiation 4](#_Toc532900143)

[5 Project Description 5](#_Toc532900144)

[6 Project Execution 6](#_Toc532900146)

[7 Personal Reflections 8](#_Toc532900147)

[8 Supervision 9](#_Toc532900148)

[9 Conclusions 9](#_Toc532900149)

# Introduction

It is very important that the reader understands the following. The complexity of running a cryptocurrency mining farm has many hardware implications which inevitably change the way the entire operation runs. However, a solution for more stable activities such as inventory management and financial reports is required to allow focus on the more unstable part of the project, which involves constant market research and fast hardware adjustment according to market prices. It is therefore crucial that inventory and finance management have a very solid base for fast data viewing and editing.

In 2009 Bitcoin became the first established cryptocurrency. Although there have been other attempts at creating other cryptocurrencies, those mostly copied Bitcoin. The decentralization of this technology (which provides data integrity, partial anonymization, and data immutability) relies heavily on difficult mathematical algorithms, randomization, and blockchain technology (a way of storing data in a chain link).

In other words, the cryptocurrency is the result of old technologies combined in a new way.

As a service, the cryptocurrency named Bitcoin is based on the idea of having a very wide network of servers from which one is picked every 10 minutes to execute an encryption task that the network requires. This encryption process combined with a randomly picked server from a big network running the software produces a high level of anonymization. Furthermore, the network layer uses other randomly picked servers to check the reliability for each of the previously executed tasks on the network. These are called “confirmations”, the more confirmations a task (known as a transaction) has, the higher its reliability. Theoretically, you can never be 100% sure, practically it is almost impossible to not be sure after 6 confirmations, which accumulate as time passes.

From a hardware perspective, due to the increased collective work required to support the cryptocurrency network, a mining farm requires expensive hardware to make a profit. The profits being the system's algorithm that rewards hardware connected to the network with Bitcoin to their address. To afford such a server and coordinate the actions of contributors connected the crypto network, the current daily task of a cryptocurrency mine is keeping track of each members contribution. This is done using hardware purchase history, including details about who purchased, at what price, from where, when, its delivery status and its current LIVE activity when the hardware is connected to the mining network. The group eventually uses those details to correctly split among members the costs, profits, and the associated risks such as warranty claims and hardware depreciation.

On the financial side, a cryptocurrency mining operation must also be good at keeping track of its financial performance and monitor its worth in real life currency and cryptocurrency in parallel. Consequently, it is important to record exchange rates at the time of crypto transfers, allocated costs, and profits and keep track of these. They also reflect the hardware depreciation and transferred cryptocurrency to each member as well as their contribution to the group costs. These are all task which must be executed flawlessly.

Fast forward to 2018, our customer a mining farm, formed from 17 people ask us to help them. The project group has the name: JMP – Javelin Mining Project.

While the number of people has been relatively small the above-mentioned requirements where easily met. Now, the number of members is starting to grow, and this makes the management become cumbersome, time-consuming and open to errors. The customer is the one that takes care of recording the information about the mined cryptocurrency value and keeps track of the inventory in terms of GPUs, motherboards, locations, date of purchase, etc. and using excel is starting to become more of a bother than it’s worth.

Furthermore, due to the number of increasing members in the group, and not being in the same country as the hardware or even outside Denmark, the group members would like to have a secure chat system for their monthly meeting while viewing the relevant data at the same time.

Therefore, the customer has decided to invest in a software to reorganize all their information into an online database which can make their daily duties easier to manage.

Upon our client’s wish, the system requires an online chat that can work at the same time as the data review itself so that better communication can be achieved.

# Group Description

At this point we are in our 2nd semester, the group has been together for almost a year. Things are going well, we communicate and help each other when it comes to assignments and tasks that we have to do.

Sadly, this year we lost one of our members Simon Tirsgaard. His leave however affected the project quite hard since when the initial project idea was selected the amount of work load we assumed it will require was a good match for a 5 people group. Especially considering that he was one of the bright Java coders in our semester.

In our team we have:

Andrei Mungiu from Moldova. He has been living and working in countries such as Netherlands, Qatar, UAE, Romania, Moldova and Ukraine. In his free time he studies for Cyber Security certificates while managing and building cryptocurrrency mining rigs for other people.

Ronalds Andris Kalnins from Latvia. He is the gamer of our group. His free time goes into games and naps.

Eric Volmer from Estonia. The fit member of the group, in his free time he is a fitness trainer.

Mihai Draghiciu from Romania. The joker of the group who in his free time bakes cakes and tests them on the group members. So far so good, still 4/5 alive.

Ron – Architecture - Knowledge involving math and phisics. due to the type of education he is focused on bringing solutions together and making them work.

Andrei – Bussiness and Economics - Knowledge about creating and overviewing a bussiness model, monitoring the life cycle of a product and organizational behaviour.

Mihai – Marketing Management – Good knowledge involving client relations and Digital Multi-Media. Focused on comercializing the product.

Eric – Automotive Technology - Knowledge involving math and electronics. Good skills in adjusting machinery and building it from scratch.

# Project Initiation

Group reflections on the project initiation phase.

This semester the project was initiated based on an actual customer request. We had a real-life case scenario with real requirements. We considered this a good opportunity to develop our skills in an environment closer to the reality outside our studies.

We constantly stayed in touch with our customer to make sure that every new suggestion to the functionality or data storage was first approved by them. As well as having our task priorities approved by the customer as well.

The initiation started well due to us having everything organized with the SCRUM methodology, in “Sprints”. This made everything go smoother as every day of the sprint we would have a plan of what to do and how to do it and see if our tasks are pushing the project in the right direction.

Although in the initiation stage we have suddenly lost a group member, the availability and strong communication with our customer ensured that we are all on the same page and adjust our expectations accordingly. This was possible since our customer was not charged for the work we were doing.

# Project Description

The project description phase went well since we have more experience now and we have a broader view of what a project would involve.

The experience taught us the value of documenting the development process as we where building the application. This eventually allowed for a more accurate and less stressful description of the project.

Since we have set aside enough time to take care of the entire project documentation, we had the opportunity to discuss as a group what we should and what we should not add to this part of the project. As a result, we have much better and more relevant content compared to the previous semester.

The project description was a good opportunity for us to make sure we start the right way and monitor our actions as we take notes of what is currently happening. This in combination with the SCRUM methodology have us invaluable insights and many wake-up calls at the right time.

As a group we managed to finish the draft of the project description in time and therefore we could dedicate resources to make sure all information is up to date and relevant according to everything that happened in our team.

# Project Execution

The project execution started with ASTAH diagrams as we spent allot of time discussing on what design patterns we must use and why.

After creating all necessary UML diagrams such as Use Case, Activity Diagrams, Sequence Diagrams and many other we started going back an forth between them during each Sprint and optimized where necessary to reflect the latest findings and suggestions.

Mainly, after all design patterns where sketched out, we started their implementation in code. This organized process has again been a result of the SCRUM methodology which we implemented.

As this process was followed again and again as we kept building more and more logical layers of the application, we were confident that the previously built code (the fundamentals) where exactly what we required and could easily expand upon them without fearing consequences that might arise from the modification of core methods. A large part of this success can be attributed to the dependency inversion principle.

Overall, the project execution went through a controlled and monitored development process that allowed us to predict the likelihood of features being or not being implemented at specific time intervals. As well as what risk management techniques should be implemented to counter issues and weather issues where worth being countered at all.

11/10/2018-01 - 11/2018

Implemented patterns. (MVC, RMI, Observer Design Pattern, Adapter Design Pattern)

Implemented core user commands.

Improved upon UML diagrams.

08/11/2018 - 22/11/2018

Optimized RMI implementation.

Changed functionality of some classes to improve functionality.

Improved the connection between SQL/Java.

29/11/2018 - 07/12/2018

Serialize all the Model Objects.

Adjusted UML diagrams according to the changes.

Started database normalization of SQL tables.

10/12/2018 – 12/12/2018

Database normalization continued.

Altered information in the table body in order to make it more understandable.

Populated data base with information.

Various improvements to the code.

13/12/2018 – 17/12/2018

Finished testing functionality.

Working on the project reports.

18/12/2018 – 19/12/2018

Spell-checking the reports, testing functionality.

Polishing and getting ready for hand-in.

# Personal Reflections

**Mihai Draghiciu**

As the second semester started I looked forward to see what was in store for us and to my expectation most of the work this semester involved it being group work. This was really nice since I could work with my team and learn from them, the semester passed fairly quick and in no time we have reached this point again. Were we have to say our opinions about this semester, group work and how everything went. Having Andrei’s expertise in the code department this made us hit the ground running in terms of development, even though we didn’t finish all the functionality work still went well. Thanks to the fact that our group is made out of people who actually care about doing something at this university, things went well, from assignments to the project itself, I am thankful I got into a group like this and see a bright future. Am looking forward to working with them in future projects.

**Ronalds Kalnins**

The semester started strong, a group of 5 people that are really interested into technology and programming. We chose an amazing project, where one of the group members is our client and requests us to make an online database for him. But few weeks into to the semester we found out that on of the strongest programmers from our group is leaving the school and that affected us as a group a lot, because the planned project was expected to be worked on by 5 people. Later, into the project we started to realize how ambitious we were and how little knowledge we had at the beginning. I enjoyed working on patterns and learning how to make your code simple and easy to read. Our whole group didn’t want to jump into the project without knowledge that is required to build our application, therefore we all focused on assignments in SDJ and worked on them separately to learn more. What I really like about our group is the fact that we know each other very well, and even if we don’t say it, we care about each other. This helps us to work better in group, because we know what our weaknesses and strengths are. Hopefully we will stay strong for the next semesters and build a lot more complicated and ambitious projects.

# Supervision

# Conclusions

Our group was already formed from the 1st semester and had a history, we worked well together and managed to fulfill our tasks.

We had the same rules as before as in always respect one another, help each other, don’t waste time, be active in group meetings. This helped us in making sure we can deliver a fine product.

We abided by the rules we set and are fairly contempt with what we managed to produce.

As a list of recommendations:

1)     Always work together as a group.

2)     Respect the rules you set as group from the beginning.

3)     Listen to everyone, decide on solutions as a vote.

4)     Respect each other and make valid points when arguing about solutions.

5)     Divide the coding part more so we could help out a bit.