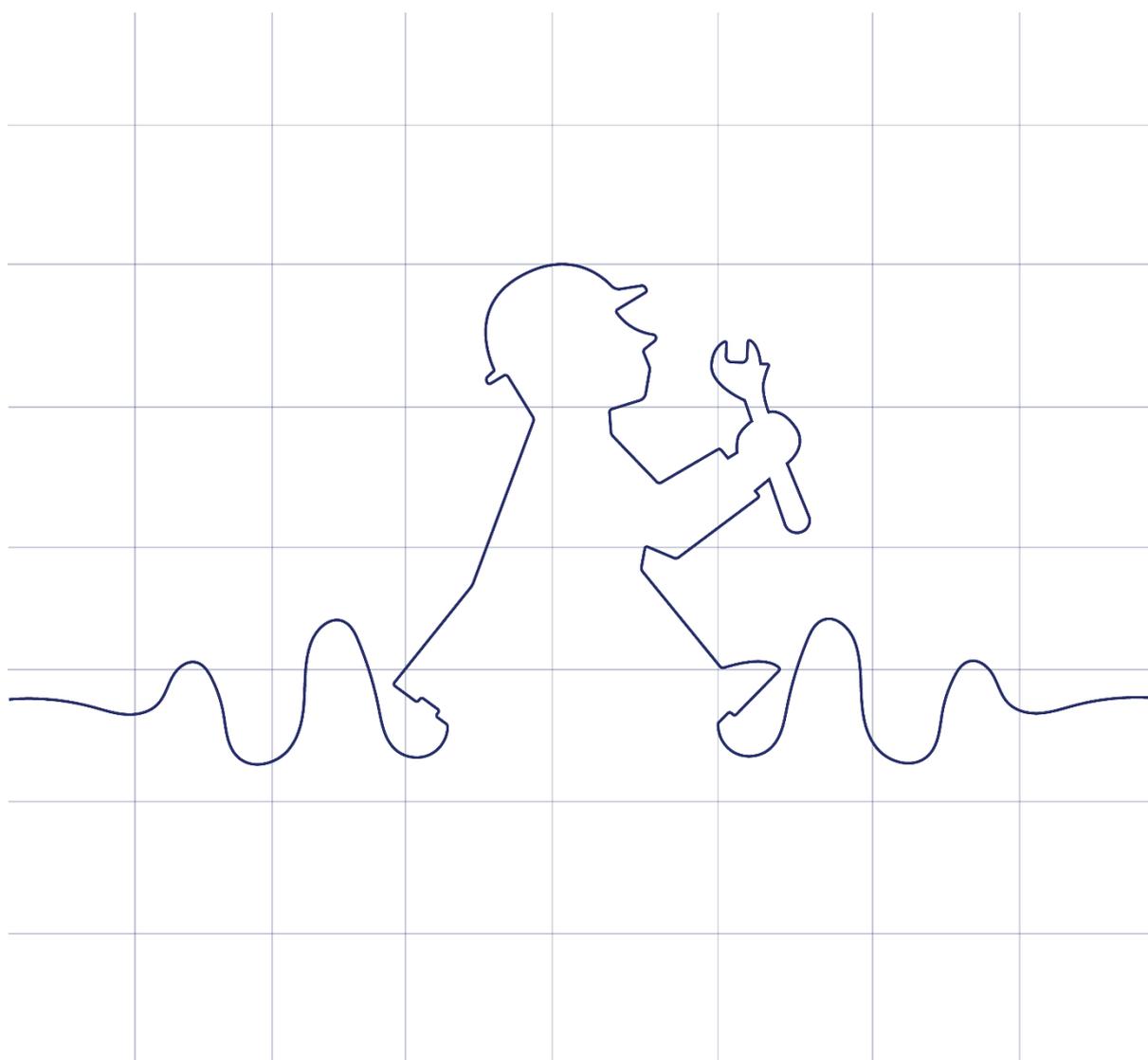

THE AMPELMANN DDT

Development of the AmpeLmann DDT data analyzing tool



JUNE 3, 2016

AMPELMANN

Devin van Tuijl

Acknowledgements

You are about to read my thesis report, which is meant as a reference guide for future Ampelmann employees who want to use the developed tool. It is also for my mentors from The Hague University Of Applied Sciences.

This thesis has been written to gain a bachelor degree in engineering. The thesis period was scheduled from the 8th of February 2016 until the 3rd of June 2016.

During my graduation period, I had the opportunity to witness and operate the Ampelmann System itself. I have developed myself in ways of communicating, programming and planning. I found the graduation period at Ampelmann very interesting and learnful.

First of all I would like to thank Martijn Krutzen and Niels van der Geld, for guiding and helping me whenever it was necessary during these 17 weeks and for sharing their knowledge.

Another thanks goes out to the other Motion Control Engineers and in particular Roman Janssen, Jurjen de Vriend and Alexander Verweij, who had several interesting ideas for my tool and were able to help whenever necessary. I would also like to thank Olaf Orlandini for working out my front page logo idea.

I would like to thank Ampelmann Operations B.V. for the opportunity to grow and learn in their company and for the opportunity to have a really great assignment in which I was able to completely let myself go and do what I found was best.

Finally I would like to thank Suzanne D. de Jong and Theo J. Koreneef for their guidance and their honest opinions about my progress and ideas.

June 2016, Delft

Devin van Tuijll

Summary

In this document, the development of a data analyzation tool is being described. The tool is developed for Ampelmann Operations B.V. to help the Motion Control engineers with the debugging process of their systems. Ampelmann Operations B.V. is a company that focusses on offshore access. Their main product is a motion compensated gangway, which allows people to safely transfer from a boat towards an offshore construction or another boat.

Since about four months, Ampelmann Operations B.V. is logging data with a self-made tool from the Ampelmann Systems' PLC to the Box-PC over an UDP connection. This Box-PC is an industrial computer that can withstand vibrations and shocks on the rough sea environment.

The logs are in a BIN file format and a tool has been created to translate these BIN-files into CSV-files.

But nothing more than that has been done to the logging part of the Ampelmann systems, and thus a demand for an analyzing tool was created.

This is where the assignment focusses on; creating a data analyzation tool that is able to plot graphs and error functions. It should also be able to derive key figures and to export selected data.

A morphological overview combined with the Kesselring method, led to a program written in Visual Basic. This program has a hard-coded structure combined with queries to let the user select relevant data and queries for modularity. The output of the tool shall be in the form of CSV-files, ensuring that they can be loaded in again at any time.

The tool is able to load the CSV-files and able to refer to the conversion tool to open the BIN files. The tool consists out of four different tabs to ensure stability and a clear overview of the functions. The names of the tabs are related to the functionality and are named: Analyze, Query, Save&Send and Key figures. Based on the structure of the CSV-file, the tool makes multiple treeviews of the containing titles within the CSV-file for some of the mentioned tabs. A treeview is a visual way of displaying all containing titles within a CSV-file in a structured way. In the analyze tab, the user is able to plot graphs and error graphs, ensuring that a visual view is gained from the logged data. In the query tab, a basic template has been created to ensure that future programmers/users are able to apply queries on the tool. In the Save&Send tab, the user is able to export data, by either saving the data locally or sending them via a mail server. In the final tab, Key figures, the user is able to derive certain Key figures ensuring that the commonly used functions are calculated automatically.

On base of this report, the conclusion can be made that the tool is a stable way of helping the user with debugging of the Ampelmann System. Based upon the test case, the conclusion can be made that the tool is able to display all of the demands of the customer. The stability can be improved by looking at the control loop for the nodes of the treeview, which lead to unexpected behavior at the moment. Another recommendation is based on workability. The treeview can be improved by adding more hierarchy. In this way the treeview becomes even more structured and this translates into an even better way of debugging.

Inhoud

Summary	1
List of Symbols	4
List of Abbreviations	5
List of Images and Tables	6
1. Introduction	8
1.1 Client and Organization	8
1.2 Process description	8
2. Assignment description.....	9
2.1 Problem definition	9
2.2 Goal:	9
2.3 Thesis question	9
2.4 Sub questions.....	9
2.5 Scope.....	9
3. Approach.....	10
4. Background research	11
4.1 The Ampelmann System	11
4.2 The cylinder.....	13
4.3 End user	15
4.4 Data logging	16
4.5 Other ways of logging	17
4.5.1 Boundaries	17
4.5.2 PLC to PC application	17
5. Conceptual Phase.....	19
5.1 Requirements.....	19
5.2 Morphological analysis	20
5.2.1 Problem definition: Tool in general	20
5.2.2 Problem definition: Data logging	26
5.2.3 Problem definition: Output of data	29
5.3 Kesselring	30
5.3.1 Problem definition: Tool in general	30
5.3.2 Problem definition: Output of data	32
5.3.3 Conclusion.....	32
6. Design.....	33
6.1 Plotting data.....	33



6.1.1 Workability.....	35
6.2 Key figures.....	40
6.2.1 Design.....	43
6.2.2 Workability.....	43
6.3 DDT.....	44
6.4 Save / Send.....	45
6.5 Online / Query.....	48
7. Programming.....	50
8 Proof of Concept	57
9. Conclusions	61
10. Recommendations	62
11. References	63
Appendix I – Plan	65
Appendix II – Accountability research for Kesselring method.....	83
Appendix III – Test case results.....	89
Appendix IV – Manual for Tool	99
Appendix V – Program	115

List of Symbols

$Abs_{cumlength}$	The absolute cumulative length
$AveCon_{err}$	The average control error of two given functions a and b
$Err_{graph}(x)$	The resulting graph function
$Fa_{err}(x)$	One of two functions of which the error graph needs to be calculated for
$Fb_{err}(x)$	One of two functions of which the error graph needs to be calculated for
$Fa_{ace}(i)$	One of two functions to calculate the average control error for
$Fb_{ace}(i)$	One of two functions to calculate the average control error for
l_{stroke}	Necessary stroke lengths for compensation
Max_{func}	Maximum value of a function
$Mean$	The mean of a function
Min_{func}	Maximum value of a function
$n_{criteria}$	Total amount of criteria
n_{csv}	The amount of items in the CSV file row
p_{kes}	Amount of points per conceptual choice on a certain criteria
T_{kes}	Total amount of points per conceptual choice
w_{kes}	Weightfactor per conceptual choice on a certain criteria
x_{csv}	A value from the read in CSV file
$x_{errgraph}$	A value that runs from the first item in the CSV file row until the last

List of Abbreviations

6-DoF	Six Degrees Of Freedom
BIN	Binary file
CSV	Comma Separated Value file
DDT	Devin Data Tool
DoF	Degrees of Freedom
FOG	Fibre Optic Gyrocompass
GUI	Graphical User Interface
HMI	Human Machine Interface
HPU	Hydraulic Power Unit
MRU	Motion Reference Unit
PLC	Programmable Logic Controller
PTA	Piston Type Accumulator
TCP	Transmission Control Protocol
UDP	User Datagram Protocol

List of Images and Tables

Figure 1 - Schematic overview of phases and work packages	10
Figure 2 - Stewart platform in neutral	11
Figure 3 - Schematic workflow Ampelmann Systems	11
Figure 4 - Motion compensated (Engaged) Stewart platform	12
Figure 5 - Hydraulic flow of cylinder	13
Figure 6 - Cylinder Valve	13
Figure 7 - Ampelmann Cylinder Control	14
Figure 8 - Ampelmann data flow diagram	16
Figure 9 - Interchangeable part	17
Figure 10 - PLC Requests and Replies	17
Figure 11- Snap7 wrapper	18
Figure 12 - Plotting CSV Data	33
Figure 13 - Treeview Function	34
Figure 14 - Error graph function	36
Figure 15 - Loading Bar	37
Figure 16 - PLC Log converter or BIN to CSV converter.	38
Figure 17 - Final Result for offline tab.....	39
Figure 18 - Final result for key figures tab	44
Figure 19 - Logo DDT	44
Figure 20 - Save and Send menu options.....	45
Figure 21 - Naming of a folder when export functions are used	46
Figure 22 - Contents of dedicated folder	46
Figure 23 - Cut off line.....	47
Figure 24 - Result of cut off.....	47
Figure 25 - Query View Connected with server	49
Figure 26 - Error displayed and display error button pushed	49
Figure 27- Use case diagram	50
Figure 28 - Block Definition Diagram	51
Figure 29 - State machine for subject: Analyse / Offline	52



Figure 30 - State machine diagram for subject: Key Figures 53

Figure 31 - State machine for Save 54

Figure 32 - State machine for Send..... 55

Figure 33 - Tabcontrol function - connections between different tabs..... 56

Figure 34 - Test environment..... 57

Figure 35 – Treeview graph node error 60

Figure 36 - Test results speed 84

Figure 37- Visual basic vs java..... 84

Figure 38 - easyness comparison 85

Figure 39 - Result of standalone test. 90

Figure 40 - 32 bit hierarchy test result..... 91

Figure 41 - Reading in CSV files test result..... 92

Figure 42 – Save function test result 93

Figure 43 - Derive Key Figure test..... 94

Figure 44 - CSV result of manual calculation 95

Figure 45 - Fool proof test failure 96

Figure 46 - Result of graph plotting 98

Table 1 - Morphological overview of tool in General 20

Table 2 - Morphological choices for Data Logging..... 26

Table 3 - Morphological choices for Output of Data 29

Table 4 - Kesselring method for programming language..... 30

Table 5 - Kesselring method for data selection 31

Table 6 - Kesselring method for modularity 31

Table 7 - Kesselring Method for the output of data 32

Table 8 - Test case table..... 59

1. Introduction

1.1 Client and Organization

Ampelmann Operations B.V. is a company which has its headquarters located in Delft. The company was founded in 2008 and has been growing ever since. They have got seven offices around the world: In Aberdeen, Delft, Brunei, Houston, Qatar, Rio de Janeiro and Singapore.

Before 2008, it was hard to access offshore structures like windmills and oil platforms. They would have to use helicopters or boats and rope swing towards the various systems, which costs a lot of effort and time and was above all a very unsafe action.

The founders of Ampelmann saw their future in finding a solution to this problem. They came up with the Ampelmann system, which is a ship-based, self-stabilizing platform that actively compensates all vessel motions using a Stewart Platform (A platform on six cylinders to control six degrees of freedom, respectively Surge, Sway, Heave, Roll, Pitch and Yaw) to make access to offshore structures safe, easy and fast.

This process is done by continuously measuring the motions of the host vessel. Then, the required lengths of the six cylinders are calculated to keep the transfer deck completely motionless. Finally, each hydraulic actuator is controlled separately.

1.2 Process description

This report describes the development of a data analysing tool for Ampelmann. The tool is designed to help future Motion Control engineers in debugging the Ampelmann systems, which at the moment is hard due to the large amount of logged data and no dedicated tool to read this data with. The results of the research are described in this report and in its appendices.

To design the tool, a morphological overview is created. In combination with the Kesselring method, one concept has been chosen to make a proof of concept for.

In chapter two, the assignment description is explained in detail, where the user can see the problem description and the goals. Chapter three is about the approach of the assignment. To understand where the tool is focusing on, chapter four has been added. Chapter four describes the background information. In chapter five, the user is able to see the conceptual choices that are made during the graduation period. Chapter six is about the design of the concept that is chosen in chapter five. Then the programming of the tool is explained in chapter seven, so that the underlying connections are clear. Chapter eight gives a proof of concept, proving that the concept is working. After chapter eight, conclusions and recommendations follow.

2. Assignment description

2.1 Problem definition

At this moment, troubleshooting an the Ampelmann system is done by reading data and using experience. The problems that occur are solved by Ampelmann motion control engineers. The problem is that it is hard to read logged data, as the data log files grow enormously due to the available amount of data.

2.2 Goal:

Ampelmann's solution to this problem is to implement a data analyzing tool, which can translate logged data into an user friendly environment wherein one is able to see what the reason is some errors occur or to adjust the Ampelmann System its settings to optimize its behavior.

The final user of this tool should be able to see data plots of the logged data and other relevant debugging information such as control errors.

Besides checking logged data, the tool should be able to receive queries and respond with a proper response (containing the information asked in the query).

2.3 Thesis question

"What is the most stable way of creating a data analysing tool and how can this be implemented inside the Ampelmann system?"

2.4 Sub questions

- Who is/are the end user(s)?
- How is data currently logged?
- How can we extract data from logged files in a structured way?
- How can we show other relevant debugging data?
- How is the user able to send queries?

2.5 Scope

To get a good result in a time span of 17 weeks, the tool should at least work for an Ampelmann hexapod's cylinder data.

The reason an Ampelmann cylinder has been chosen, is because the whole Ampelmann system has a lot of error functions. All these error functions are too much to program in a timespan of 17 weeks. The cylinder holds a subset of error functions, large enough to prove the functionality of the tool.

3. Approach

In order to come up with an approach, the project is divided into different phases and work packages. This is done to gain a good structure in the approach of the solution to the thesis question.

This project can be divided into five phases which are described below:

The first one is the start-up phase, wherein the actual start-up will take place. This includes defining a plan on how to successfully make a thesis.

The second phase is the definition of the project. It will consist out of background research and the documentation of its results. The background research consists of reading available documents to understand the hardware of the current Ampelmann systems and understanding the Ampelmann system software, by looking into the current software code. Also hydraulics and electrical courses specifically designed for the Ampelmann systems will be followed.

The third phase will be about the design. Here, the actual research will be translated into a proof of concept. This will include a morphological overview, a Kesselring method to choose the correct solutions based on some given weight factors and working out the concepts.

Then the fourth phase is the testing of the system. The project needs to be realized and tested, to ensure its workability and so that the outcomes of the analyzing are correct. Also a well written report has to be created, so that documentation of the tool is always available.

The final phase is the writing of the Thesis and writing documentation, to ensure the total system is documented.

The schematic overview for the given phases and work-packages are shown in the figure below. The detailed description is left out and elaborated in Appendix I – Plan.

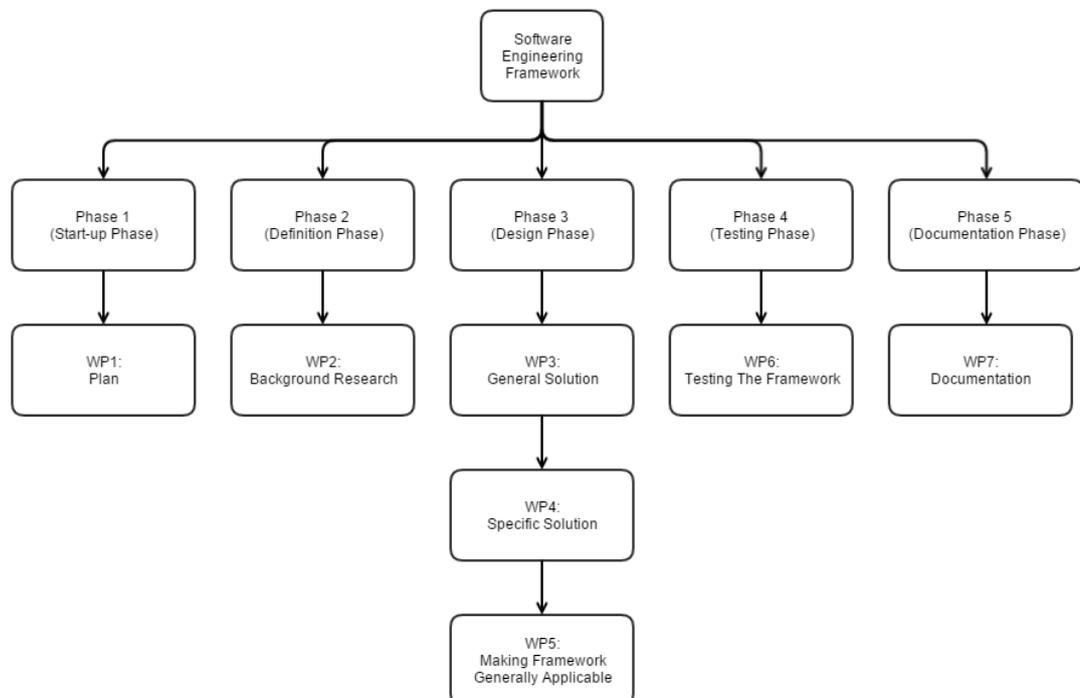


FIGURE 1 - SCHEMATIC OVERVIEW OF PHASES AND WORK PACKAGES

4. Background research

4.1 The Ampelmann System

The Ampelmann system is a ship-based, self-stabilizing platform that actively compensates all vessel motions using a Stewart Platform.

A Stewart platform is a platform that consists of six cylinders to give the user six degrees of freedom (6-DoF) to move the platform upon. Another name for the Stewart platform is “hexapod”.

A schematic view is shown below wherein the Stewart platform is in its neutral position, which is the position wherein the cylinders are send to the middle of the total stroke length. In this way it can go down and up the same amount of meters, ensuring that the system is able to compensate sinusoid like waveforms.

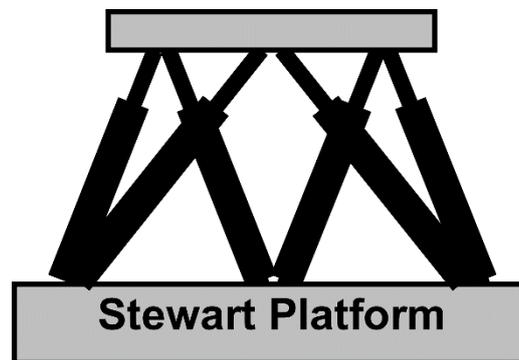


FIGURE 2 - STEWART PLATFORM IN NEUTRAL ¹

The motion control loop constantly checks the current vessel/ship motions and will try to compensate these motions by using the six hydraulic cylinders. These will try to keep the top frame horizontal with respect to the horizon.

To understand how the cylinders are moved, a schematic flow is given to represent what is actually happening:

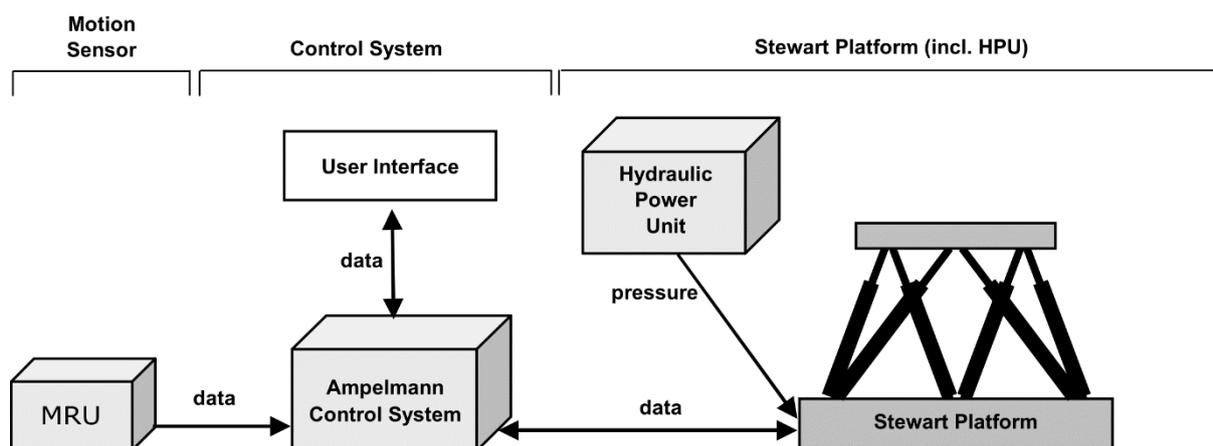


FIGURE 3 - SCHEMATIC WORKFLOW AMPELMANN SYSTEMS ²

¹ Cerda Salzmann, D. J. (2010, 10 7). *We at sea*. [\[1\]](#)

² ""

The Motion Reference Unit or MRU, measures the vessel motions using a FOG technique. FOG stands for Fiber Optic Gyrocompass and uses the Sagnac effect to measure its Roll, Pitch, Yaw which are respectively the rotation amongst x, y and z. The Surge, Sway and Heave, respectively the translation amongst x, y and z are done by an accelerometer.

The Sagnac-effect³ is the phase-shift-effect of two rotating waves that undergo a rotation themselves. If pulses of light are sent through an optic fiber coil in the clockwise and anti-clockwise direction simultaneously and the coil itself moves, the same effect occurs. The light that moves in the same direction as the coil moves, takes a bit longer to arrive than the opposite direction, thus resulting in a phase shift. This phase shift can be calculated and thus the rotation angles of the moved FOG.

The data of the FOG is send to the control unit of the Ampelmann system. The controls of the Ampelmann systems are realized by using a PLC or Programmable Logic Controller. These controllers are quick and reliable. The PLC contains analog inputs, analog outputs, digital inputs and digital outputs. With these in- and outputs, the control loop of the system is able to move the cylinders valves, which let hydraulic flow in or out and thus letting the cylinder extend or retract.

In the next figure, the position of the motion compensated (Engaged) Stewart platform is schematically displayed. The base of the frame is attached to the boat, and it is tilted due to the waves that hit the boat. The motion control loop calculates the necessary stroke lengths l_{stroke} , to overcome the ships current angles in the 6-DOF motion, which are schematically shown in the picture with one angle: θ_{ship} .

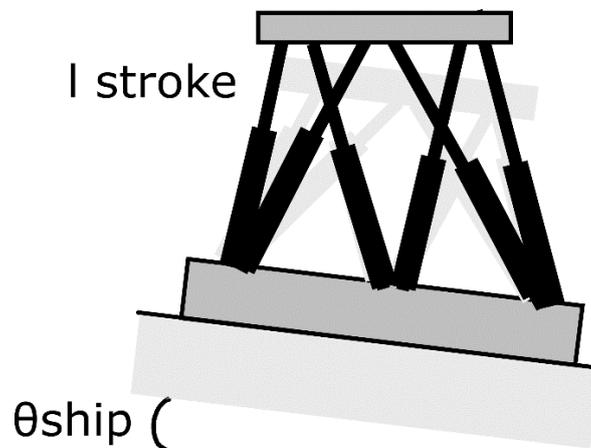


FIGURE 4 - MOTION COMPENSATED (ENGAGED) STEWART PLATFORM

³ Sagnac Effect [5]

4.2 The cylinder

As described in the project scope, the main goal focusses solely on a cylinder due to the large amount of errors the total Ampelmann system has. In order to understand the project completely, background info of the cylinder Ampelmann uses is necessary. Therefore this information will be given in this paragraph.

The cylinders in the Ampelmann system are controlled by valves which control hydraulics, the hydraulic flow is demonstrated below:

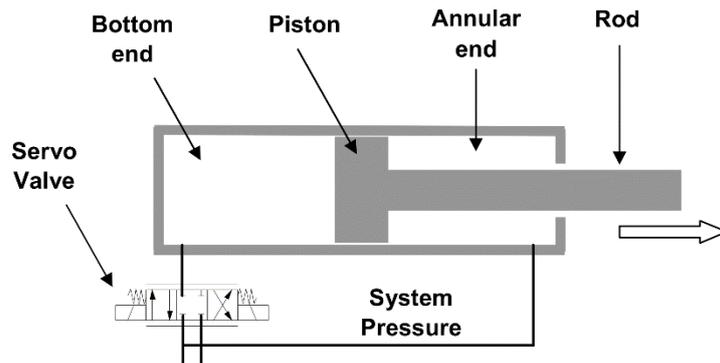


FIGURE 5 - HYDRAULIC FLOW OF CYLINDER ⁴

The hydraulic pressure comes from a diesel operated pump or electric pump and pumps oil throughout the system, while a PTA (Piston Type Accumulator) makes sure that there is pressure even when the hydraulic power supply is cut off. This knowledge is left out of this description, due to its irrelevance to the project.

The valve used in the picture above is shown in detail below:

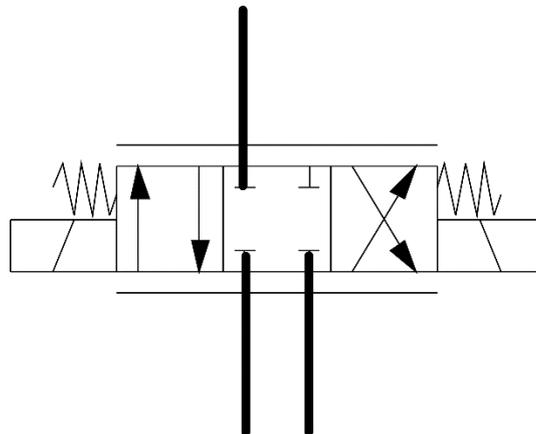


FIGURE 6 - CYLINDER VALVE

The valves that are used are proportional and they are solenoid controlled. Proportional means that any position between fully open and fully closed can be used to send flow through the valves. The valve is controlled by a solenoid which pushes the valve from one condition to the other condition. In its rest condition, the springs will put the valve in the middle condition, so that it is closed. This valve let's a flow come through either inwards or outwards and this controls the positioning of the cylinder.

⁴ Cerda Salzman, D. J. (2010, 10 7). *We at sea*. [\[1\]](#)

To ensure that the cylinder is at its desired position, there is a position transducer installed. This transducer calculates the actual length of the cylinder and its length is compared to what the actual position of the cylinder is. If the cylinder is not at the desired position, the valve control adapts to make sure it will be. This system can be seen as following:

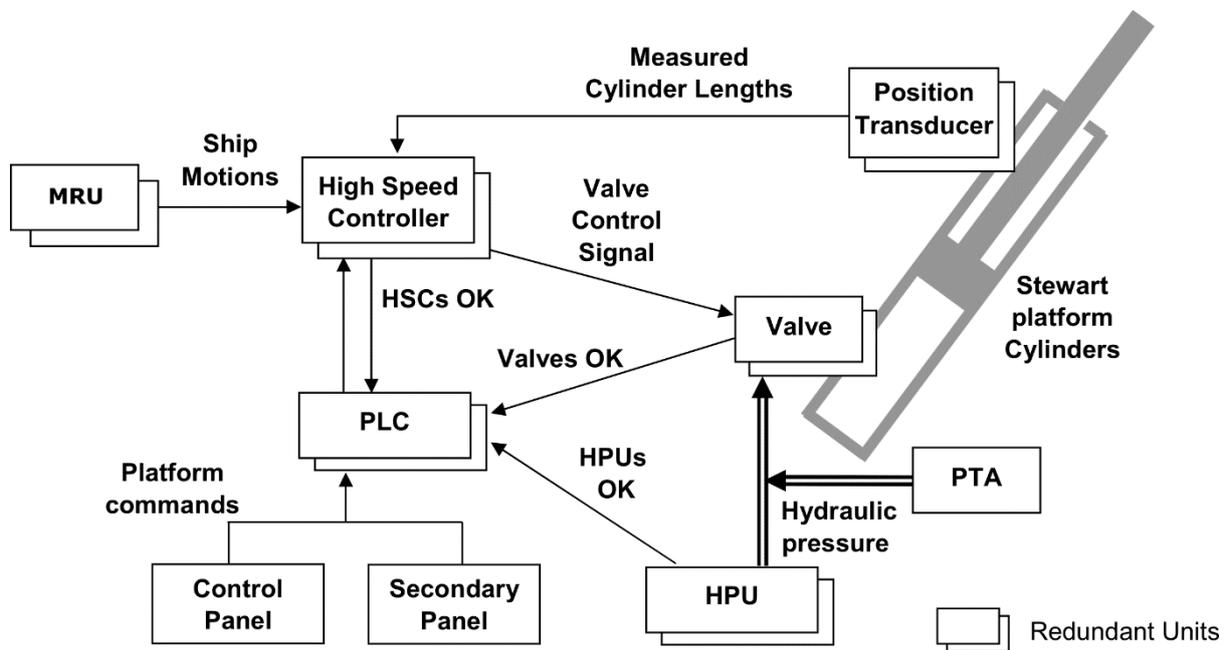


FIGURE 7 - AMPELMANN CYLINDER CONTROL ⁵

In above picture, the mentioned loop between the valve control, valve and position transducer is easily seen. The Hydraulic Power Unit or HPU supplies the system with hydraulic pressure. The system operators are able to operate the system with the control panel and GUI (Graphical User Interface) that is installed on a touchscreen panel. Combined with buttons and other inputs and outputs, they form a Human Machine Interface or HMI. They can put the Stewart platform in different modes with this interface, while the Octans sends data towards the controller so that in a case of motion compensation, the system knows how to respond.

The cylinders have several components as described before, and these are the ones that are meant to be allocated to the tool. The functions on which the tool should focus, are determined by the project owners: The control error on the cylinder length and the control error on the valve position. Besides these error functions, the total absolute cumulative length, the minimum, maximum and the average values of a given input are also meant to be calculated. These functions are applied to the currently loaded in csv files over certain intervals.

⁵ Cerda Salzmann, D. J. (2010, 10 7). *We at sea*. [1]

4.3 End user

The end user is an important aspect in this project. The tool must be designed towards the end user to ensure that the tool is understandable.

The end user is defined by the project owners. It is defined by the following quote:

“The tool has to be designed in such a way, that the motion control engineers of Ampelmann are able to understand the tool almost immediately and if this is not the case, a help function should help them to do so.”

The wishes and demands of the end user will give boundaries for the tool. A summary of what the demands and wishes for the end user are, are shown below:

- There has to be a tool wherein the user is able to read in logged data.
- Inside the tool, graphs of logged data should be plotted.
- The tool should make it easier to select parts of data.
- The tool should be able to derive certain key figures for a selected period of time.
- The final prototype must be a standalone tool.
- A working test environment has to be made.
- The final prototype should be safe (e.g. not vulnerable for hacking, steady connections, etc.)
- The final prototype should be able to select data that is relevant to the user.
- The system should be written in a language that is known to most of the motion control engineers.
- The system should be “fool-proof”.
- The final prototype should have a catchy name, to ensure the tool will be remembered and used.

The education level on which the tool is based, is higher education, meaning that the end user will have an HBO-bachelor degree or higher.

4.4 Data logging

In order to understand how the solution is found for the research question, the principles of how the data logging is currently done should be known.

The data logging is currently done by sending data from the PLC to a Box-PC and possibly to a PC/Laptop. A Box-PC is an industrial computer which is more steady than a personal computer. This will be elaborated in the 5. Conceptual Phase chapter.

A short representation of the current system is shown in the following figure:

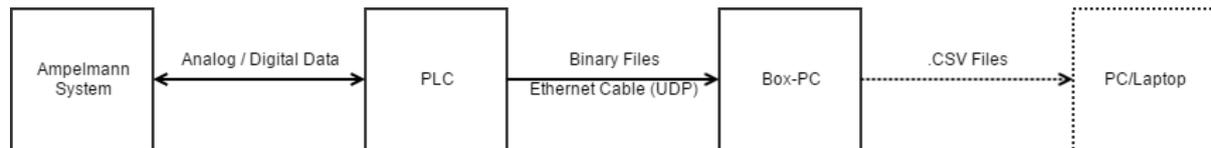


FIGURE 8 - AMPELMANN DATA FLOW DIAGRAM

The Ampelmann System's electronics and the PLC are connected via standard copper wires. Whilst this is happening, the PLC keeps track of all of the inputs and outputs of the system. The PLC stores this data inside a buffer and once the buffer is full, the data will be transmitted towards the Box-PC. The Box-PC is an industrial computer that is able to withstand vibrations and shocks. These transmitted files are sent via an UDP socket-connection and stored in binary format on the Box-PC. This UDP-connection is a one-way connection: The PLC sends out data, without getting an acknowledge of someone receiving it. In this way, the PLC won't notice any remarkable drops in computing time, whilst with a TCP-connection it would. This is because of the two-way communication in TCP, the PLC will actively listen to what is being communicated on the dedicated socket port.

The output files are in a binary format, which can be decoded to CSV files if the structure files are supplied. These structure files are necessary to decode the Bin files.

The available .CSV files that are the output result, are readable data logs that contain info about time and what has happened during that time.

4.5 Other ways of logging

Currently the data logging is done in the way described above. This might not be the best way of doing this, so other methods were researched. First some boundaries are given and then the researched methods are described.

4.5.1 Boundaries

The PLC-program should not be changed, due to the fact that it is not only used for logging, but also for controlling the Ampelmann system.

So all interchangeable logging parts are the parts that come after the Ethernet cable that comes from the PLC, which is displayed with X in the following figure:

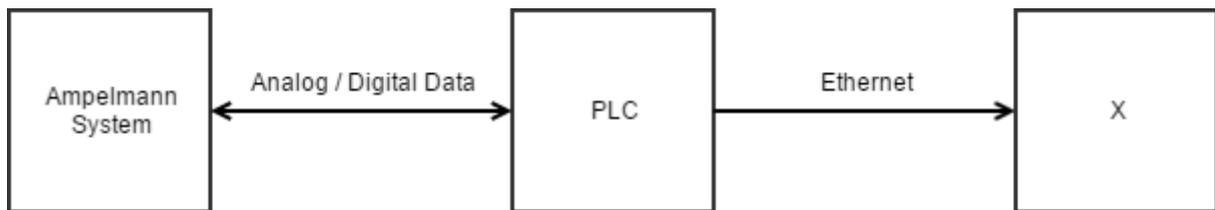


FIGURE 9 - INTERCHANGEABLE PART

4.5.2 PLC to PC application

Given the boundaries above, it is still possible to connect the Ethernet cable towards a PC. However, the data still needs to be extracted by the PC. The PC can do this by connecting to the PLC and read its data when done by a dedicated application, because the PC cannot do this directly. One of these ways is mentioned below.

4.5.2.1 PLC to Python

Some documents^{6 7} mention the connection between a Siemens PLC and Python. Python is a programming language with a lot of libraries.

One of these libraries is the Snap7 library. This is a python wrapper for connecting with Siemens PLCs. To understand how this communication could be established, the basic way of how Siemens PLCs work is described first:

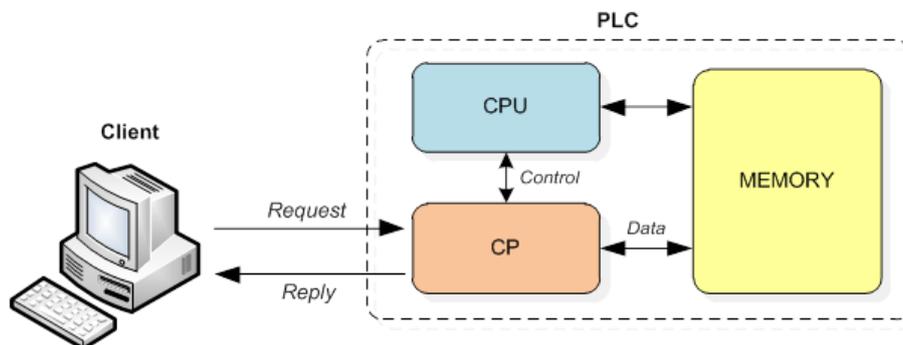


FIGURE 10 - PLC REQUESTS AND REPLIES⁶

⁶ Snap7. (n.d.). *Snap7*. [7]

⁷ Molenaar, G., & Preeker, S. (n.d.). [6]

As seen in Figure 10, the client can request data from the PLC. The CP,CPU and the memory will transmit data so that the CP can send a reply back.

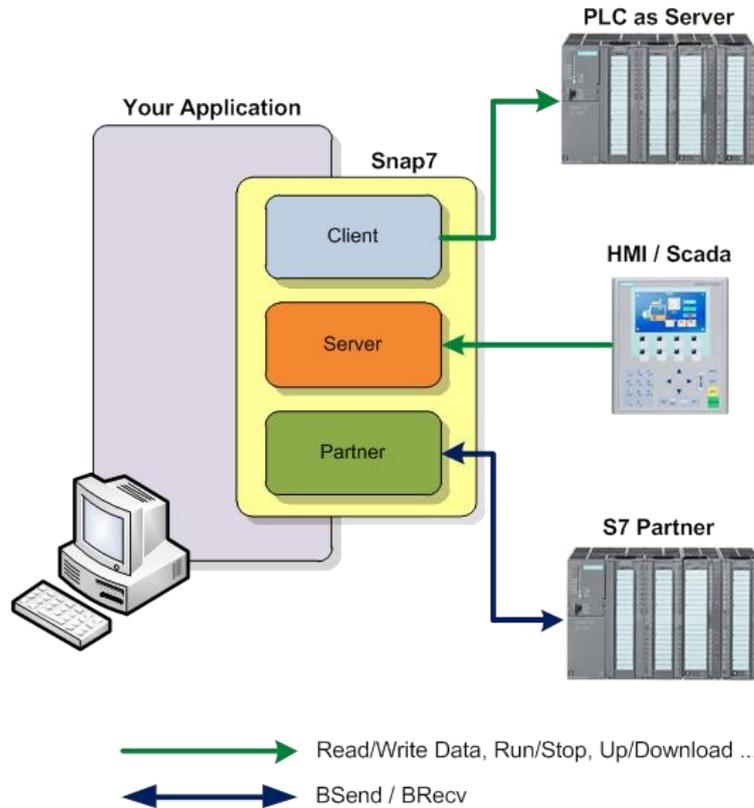


FIGURE 11- SNAP7 WRAPPER ⁸

In Figure 11 is shown how Snap7 works. Snap7 is a communication protocol between several applications and the Siemens PLC. The application can act as a client, server or partner. Because the same structure is used as within the Siemens PLCs, the PLC won't notice that it is communicating with an application instead of a Siemens Client/Server/Partner.

A big downside is that the client sends data towards the PLC as well, and this tends to slow down the computing time of the whole PLC which can have negative consequences for the Ampelmann system.

Another downside is that this will actively open the PLC for incoming hacking threads, which can put the PLC in stop mode during operation.

Due to this, the way of logging remains the same as mentioned before (See 4.4 Data logging): Logging over UDP with binary files.

5. Conceptual Phase

5.1 Requirements

First the project requirements need to be known in order to get a concept that covers the demands of Ampelmann. A method that is useful to determine and write down requirements, is the MoSCoW-method.

The MoSCoW-method is used to describe the main project objectives and requirements and can be used for this project.

MUST

Product requirements:

- There has to be a tool wherein the user is able to read in logged data.
- Inside the tool, graphs of logged data should be plotted.
- The final prototype has to be in a 32-bit hierarchy.
- The final prototype has to be a standalone file.
- The tool should make it easier to select parts of data.
- The tool should be able to derive certain key figures for a selected period of time.
- There has to be a working test environment.

SHOULD

Product requirements:

- The system should be able to select data that is relevant to the user.
- The system should be written in a language that is known to most of the motion control engineers.
- The system should be “fool-proof”.
- The final prototype should have a catchy name, to ensure that it will be remembered and used.
- The system its graphical user interface should be easy to understand.

COULD

Product requirements:

- It could be an advantage if the final prototype will not cost a lot / anything.

WOULD

Product requirements:

- It would be great if the final prototype is a simple executable file.

5.2 Morphological analysis

To come up with the best solution, several options of coming to a solution need to be reviewed. One way of doing this, is using a morphological overview which describes the different solutions for different parts of the requirements. Both pros and cons of all parts are described. At the end of this chapter, a final choice as concept will be made according to those advantages.

At first, the project needs to be split up in multiple sections, so that the morphological analysis can be used upon every single part.

5.2.1 Problem definition: Tool in general

The tool needs to be designed and the way of doing that has to be determined. There are several solutions to this problem and the morphological analysis will make sure the best solution is chosen.

The requirements for this problem from the MoSCoW-method are:

- The system must be built in a modular way.
- System should be able to select data that is relevant to the user.
- The system should be written in a language that is known to most of the motion control engineers.
- The system should be “fool-proof”.
- It would be great if the final prototype is a simple executable file.
- The documentation must be clear. Either on paper, as commenting code.

Function:	Option 1:	Option 2:	Option 3:	Option 4:	Option 5:
Code Language	Visual Basic	python	Java	Combination	
Data Selection	Rule Base	Machine learning	Hard Coding	Query	Combination
Modularity	Query	XML	Combination		

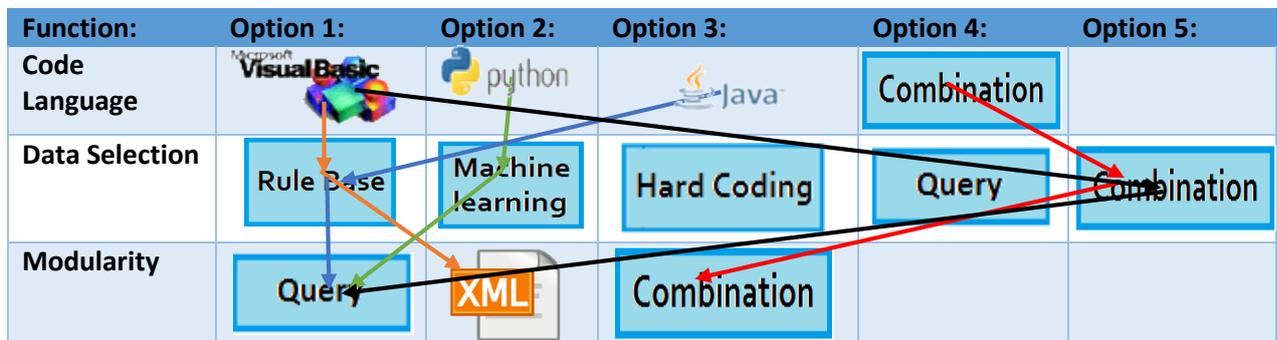


TABLE 1 - MORPHOLOGICAL OVERVIEW OF TOOL IN GENERAL

5.2.1.1 Concept A

Concept A uses Visual Basic as programming language, in combination with a rule base selection method and XML for modularity. Concept A is represented in Table 1 by the orange line.

Visual Basic:

Visual basic is a computer scripting language that most of the engineers at Ampelmann have mastered. It is an easily understandable language and there are a lot of functions from which good working GUI's can be made. A downside is that live graphs are less easy to program, which is a function that could make the tool better.

Pros:

- Easy GUI
- Easily understandable
- Employees already master the language

Cons:

- Live graphs not easily made

Rule Base:

The relevant selection process, as described before, is the process that should make sure that the user of the tool will only get results that are relevant.

“Rule-based systems automate problem-solving know-how, provide a means for capturing and refining human expertise, and are proving to be commercially viable”⁹.

For this project this might come in very handy, because this ensures a certain amount of modularity. If used correctly, it can predict the “likeliest places to look for relevant information” and “probable causes”. A con might be that when the system is expanded in the future and more rules are added, the system’s computation time will increase.

Pros:

- Places of relevant info
- Probable causes of failures

Cons:

- Slow computation time

⁹ (Hayes-Roth, 1985)[3]

XML:

XML is short for Extensible Markup Language, and is very useful when you want certain messages to be in the same structure every time. It is platform independent and it is easily extendible, hence the name. A con of XML is that not many browsers support it and it thus needs an extra application to be implemented.

Pros:

- Easily Extendible
- Platform Independent

Cons:

- Not many supported browsers

5.2.1.2 Concept B

Concept B uses Python as programming language, Machine learning to select relevant data and queries to ensure modularity. Concept B is represented in Table 1 by the **green** line.

Python:

Python is a programming language, with many built-in libraries and many libraries to extend it with. In Python, it is very easy to plot live graphs. Python also uses indentation to make the structure of the program in a smooth, easily readable way. Python is open-source and thus free for use. A con for Python is that at the moment there are two supported versions, the 2.7 and 3.4 version, which may lead to conflicting code.

Pros:

- Many Libraries
- Live graphs easy
- Smooth and readable
- Free

Cons:

- Two supported versions

Machine Learning:

Machine learning is a way of artificial intelligence, which relies on algorithms. It can predict data and can be used for classification problems. For this project it could be useful to predict failures and to select specific data for the user. A con of machine learning is that it is hard to implement (you will need extra datasets for training, testing and validating) and it can be hard to visualize what is going on.

Pros:

- Prediction of failures
- Select specific data

Cons:

- Hard implementation
- Hard visualization

Queries:

Queries are questions that are asked to a system which will interpret them, process them and reply with a certain output. Inside this project, it is enabling the user to get answers on their specific question or query. A pro of queries are that is known what the user actually wants. Another pro is that is able to make the system modular with this. A con is that for modularity, the structure of the queries need to be handled the same way every time, and this may take extra time to implement.

Pros:

- Clear demands (questions) of user
- Modularity is available

Con:

- Modularity implementing may take extra time

5.2.1.3 Concept C

Concept C uses Java as programming language, with a rule base selection method and queries to ensure modularity. Concept C is represented in Table 1 by the purple line.

Java:

Java is a programming language, which aims for the WORA-solution, which means “*Write once, Run Anywhere*”¹⁰. This makes sure that the final program will work anywhere, which is a benefit. Within Java it is also easy to plot live graphs. It’s also an open-source language, which makes it free to use.

The cons of Java are that it can use a lot of memory, and like XML it requires an interpreter.

Pros:

- WORA solution
- Easy Live Plots
- Free

Cons:

- Memory
- Requires Interpreter

Rule Base:

See ‘Rule Base’ on page 21.

Query:

See ‘Queries’ on page 22.

¹⁰ Computer weekly [2]

5.2.1.4 Concept D

Concept D uses a combination of programming languages to write the tool in. It uses a rule base structure in combination with queries and hard coding to select relevant data and it uses a combination of XML and queries to ensure modularity. Concept D is represented by the red line in Table 1.

The combination of programming languages is to get the best out of both. Python is good for plotting live graphs and Visual Basic can make good GUI's and is mastered by the engineers of Ampelmann.

The combination of the Rule Base structure, with queries and hardcoding, is to ensure the modular use of the final prototype. The queries will be used to let the user give in queries, which the tool will process and translate into an output.

Python:

See 'Python' on page 22.

Visual Basic:

See 'Visual Basic' on page 21.

Rule Base:

See 'Rule Base' on page 21.

Hard Coding:

With hard coding, the way of selecting data is done with input of the user with predefined choices. Pros of hard coding are that the user cannot make a mistake if programmed correctly and the user knows which options are available. A con is that the options are limited, due to the given choices.

Pros:

- No mistakes can be made by the user if programmed correctly
- Clear options to choose from

Cons:

- Limited choices

Query:

See 'Queries' on page 22.

XML:

See 'XML' on page 21.

5.2.1.5 Concept E

Concept E uses visual basic as programming language, with a combination of choices for data selection and queries for modularity. Concept E is shown in Table 1 with the **black** line.

Visual Basic:

See 'Visual Basic' on page 21.

Rule Base:

See 'Rule Base' on page 21.

Machine Learning:

See 'Machine Learning' on page 22.

Hard Coding:

See 'Hard Coding' on page 24.

Query:

See 'Queries' on page 22.

5.2.2 Problem definition: Data logging

The data logging currently is done by the method described in the data logging chapter. This is one way of doing it, but there may be better ways of doing this hence the morphological overview below.

Function:	Option 1:	Option 2:	Option 3:
Data Logging			
Data structure			

TABLE 2 - MORPHOLOGICAL CHOICES FOR DATA LOGGING

5.2.2.1 Concept A

Concept A relies on the Box-PC, in combination with .Bin files. Concept A is shown in Table 2 - Morphological choices for Data Logging with the orange line.

Box-PC:

The Box-PC is an industrial computer that is able to withstand vibrations and shocks. This is one of the reasons it is being used at the moment in every Ampelmann system, because they need to withstand the ship's motion vibrations. Another pro is that because it is already implemented in every Ampelmann system, it is not necessary to buy a whole new item for the Ampelmann system, and this will suppress costs of the final prototype.

A con of the Box-PC is that it is a really slow computer. This is fixable, by replacing the internal hard disk drive with a solid state drive. Solid state drives are much quicker, because they do not use the spinning disks, but rather use a set of chips to speed up the process.

Pros:

- Already implemented, low distribution costs.
- Able to withstand shocks.

Cons:

- Slow, but this is solvable with an SSD drive.

Bin Files:

A Bin file is a file type which is primarily associated with 'Binary File'. It consists out of a sequence of bytes.

One of the major pros of binary files, is that they are relatively small in storage space. This is convenient in the future, when the data logs will be transmitted over bad sea-internet, because they will have to transmit small amounts of data. Binary files will most of the time be unreadable, because the original structure has to be known in order to open the file correctly.

Pros:

- Relatively small in storage space.
- Useful with bad internet connections.

Cons:

- Hard to read without knowing its original structure.

5.2.2.2 Concept B

Concept B relies on a Raspberry Pi, in combination with CSV-files as data structure. It is shown in Table 2 - Morphological choices for Data Logging, by the [blue](#) line.

Raspberry Pi:

The Raspberry Pi is a small programmable computer. Its major pros are that it is very small, easily programmable (Open-source software) and is relatively cheap compared to a Box-PC. A con is the slow Ethernet connection it has compared to a normal computer or Box-PC. Another con is that all Box-PC's should be replaced with the raspberry pi, or they should at least be compatible with each other.

Pros:

- Cheap
- Compact and Small
- Easily Programmable

Cons:

- Slow Ethernet
- Replacement or combining (synergy) with existing Box-PC

CSV:

CSV stands for "Comma-Separated Values". This data structure is text-based, which means that it is easily readable for every user, which is a pro in comparison with Binary files. Another pro is that it is easily editable. The cons of CSV is that confusion can be created with the use of punctuation, as the files are separated by commas.

Pros:

- Easily Readable
- Easily Editable

Cons:

- Confusion with the use of punctuation.

5.2.2.3 Concept C

Concept C relies on a laptop in combination with a CSV data structure. Concept C is shown in Table 2 - Morphological choices for Data Logging, by the **green** line.

Laptop:

Another way of data logging is by using a laptop. The laptop has the most user friendly interface in comparison with the Box-PC and the Raspberry Pi. Another pro is that it is a mobile, standalone device. A con is that it is large in comparison to the Box-PC and the Raspberry Pi. Another con is that it is more expensive to replace all the Box-PC's rather than using the existing Box-PC's.

CSV:

See 'CSV' on page 27.

5.2.2.4 Concept D

Concept B relies on the Box-PC in combination with the combination of BIN and CSV. Concept D is shown in Table 2, by the **red** line.

Box-PC:

See 'Box-PC' on page 26.

BIN:

See 'BIN' on page 27.

CSV:

See 'CSV' on page 27.

5.2.2.5 Concept E

Concept E relies on a Box-PC with a CSV data structure. It is shown in Table 2, by the **black** line.

Box-PC:

See 'Box-PC' on page 27.

CSV:

See 'CSV' on page 27.

5.2.3 Problem definition: Output of data

The output of the data should be researched, because the system has to be able to reproduce earlier asked queries. These queries can be stored in different ways.

Function	Option 1	Option 2
Output File		

TABLE 3 - MORPHOLOGICAL CHOICES FOR OUTPUT OF DATA

5.2.3.1 Concept A

Concept A relies on an output file in the form of a .CSV file. It is shown in the table above with the red circle. This way of creating output files ensures that the files can be read in at any time and that they are readable without needing a special program to open them. A con of a CSV file might be that the structure is preset and that there is a possible cause for trouble for when the file needs to be reproduced on the screen, because it is the same structure as the files that are read in as input.

Pros:

- Easily readable

Cons:

- Preset structure that can cause troubles when trying to reproduce

5.2.3.2 Concept B

Concept B relies on a dedicated output file, that contains all information to reproduce an earlier state of the tool. It is shown in Table 3Table 1 with the blue circle. A pro of this manner is that the output is dedicated, which implies that the output is exactly as wanted and the files have an extension that is only able to be opened by the tool itself. A con of this manner is that it will tend to have the property that it is unreadable without a special program to open it. The output file extension will be in the form of .Ampelmann.

Pros:

- Dedicated output, easy for reproducing

Cons:

- Unreadable without a special program

5.3 Kesselring

With the morphological overviews shown in the previous chapter, it is possible to design concepts out of them by using weight factors to determine which pros and cons are more important than the other. A great way of displaying these weight factors is by using the Kesselring Method. This method calculates with the weight factor, which option is the best for the final solution (This method is designed by F. Kesselring of the Technical University of Twente¹¹).

Each weight factor can differ from one to four, wherein one is the least important and four is the most important. The scores of the individual choices are also numbered one to four, where 1 is the lowest and 4 is the highest score.

The total amount of points per choice is calculated by the following formula:

$$T_{choice} = \sum_{i=1}^{n_{criteria}} p_i w_i$$

EQUATION 1 - KESSELRING CALCULATION

Where:

$n_{criteria}$ = *The total amount of criteria*

T_{kes} = *Total amount of points per conceptual choice*

p_{kes} = *Amount of points per conceptual choice on a certain criteria*

w_{kes} = *Weight factor per conceptual choice on a certain criteria*

Each grade in the Kesselring method is accounted for in Appendix II – Accountability research for Kesselring method.

5.3.1 Problem definition: Tool in general

The first thing the Kesselring method will be used upon is the programming language. It is divided into five categories, which compares them at speed, flexibility, easyness, experience and stability. Then they are multiplied with a weight factor. The weight factors are based upon the demands of AmpeLmann. Therefore, speed and stability are necessarily the most highest weight factors of them all with a weight factor of four. The other three sections are also important, but less important than the speed and stability and are thus weighted with a factor of three.

Programming Language	Visual Basic	Python	Java	Combination	Weight Factor
Speed	4	3	3	3	4
Flexibility	3	4	3	4	3
Easyiness	3	3	1	2	3
Experience	3	4	1	2	3
Stability	4	3	3	2	4
Total	59	57	39	44	68
Percentage	86.8%	83.8%	57.3%	64.7%	100%

TABLE 4 - KESSELRING METHOD FOR PROGRAMMING LANGUAGE

¹¹ Kesselring, F. (1954). *Technische Kompositionslehre*. Berlin [4]

In the table on the previous page, the difference between Visual Basic and Python is only 2 points out of 68. This is a difference of 3%.

Java scores bad on overall score, and is thus left out of the conceptual phase.

The combination of languages doesn't score better than the languages on their own, hence this will be left out of the conceptual phase.

Because this difference is so marginal between Python and Visual Basic, it is not pretty clear which system wins from the other. Therefore another factor has to eliminate the one from the other. The practical use is still not incorporated in the comparison. Because Ampelmann employees are more common with Visual Basic than Python, Visual Basic will be used for the conceptual design.

Data Selection	Rule Base	Machine Learning	Hard Coding	Query	Combination	Weight Factor
Flexibility	2	3	1	2	4	3
Easyiness	2	2	4	3	3	3
Experience	1	3	4	2	3	3
Stability	1	2	3	3	4	4
Total	19	32	39	33	43	52
Percentage	36.5%	61.5%	75%	63.5%	82.7%	100%

TABLE 5 - KESSELRING METHOD FOR DATA SELECTION

In the table above, rule base, query and machine learning score the lowest. Hard coding could be a good option, but hard coding makes the prototype less dynamical which is unwanted. A combination of the other options comes out best with 82.7% and is thus used for the conceptual design.

The combination that will be used will become clear in the conceptual design starting on page 33.

Modularity	Query	XML	Combination	Weight Factor
Flexibility	2	2	3	3
Easyiness	2	2	1	3
Experience	3	2	1	3
Stability	3	3	4	4
Total	33	30	31	52
Percentage	63.5%	57.7%	59.6%	100%

TABLE 6 - KESSELRING METHOD FOR MODULARITY

In the table above, using queries comes out as best option. Because the differences are not too big, it is possible to implement XML, but queries will mainly be used for modularity.

So the concept that will be used for the tool in general is concept E. It is shown on page 20 in **black**.

5.3.2 Problem definition: Output of data

The output of the data is also important, as described in the morphological analysis. This will be shown in the table below:

Output of Data	CSV	Ampelmann file	Weight
Flexibility	2	4	3
Easyiness	4	2	3
Experience	4	1	3
Total	30	21	36
Percentage	83.3%	58.3%	100%

TABLE 7 - KESSELRING METHOD FOR THE OUTPUT OF DATA

Table 7 suggests that a CSV file type is the best way of storing the data, due to experience and it's relatively easy understandable structure.

The outcome is thus equal to concept A, which is shown in table 29 with the red circle.

5.3.3 Conclusion

As a conclusion to this chapter, a summary of the chosen set-up is given:

The tool shall be programmed in Visual Basic, with assistance of Python if felt necessary. For data selection a combination of hard coding and queries come out as best, hence the tool will focus on both.

For modularity, queries will mainly be used and xml if found necessary.

As output, the CSV files will be used, so that the output can be used as input in the same tool whenever necessary.

6. Design

To realize the conceptual choices in the previous chapter, the design for each conceptual part will be elaborated in this chapter.

One of the requirements of Ampelmann, was to make an easy understandable graphical user interface or GUI (see '5. Conceptual Phase

5.1 Requirements').

The GUI depends on the language in which it is build. For this case it is Visual Basic.

Within Visual Basic, the programmer is able to create "Windows-style" GUI's.

To make the GUI, the demands are split into parts. In this way, each part gets enough attention to make it run smooth and "fool-proof". The parts are:

- Plotting Logged Data / Offline
- Key Figures
- Saving or Sending

For each part will be elaborated which choices are made for the design.

6.1 Plotting data

For plotting data within Visual Basic, is a build-in function called Chart.

With the Chart-function, the programmer is able to give parameters for the X and Y axis of a chart so that a graph will be made with a predefined shape (e.g. a "line-style" graph or "bar-style" graph).

To implement the chart in this project, the X and Y values should come from the CSV-files.

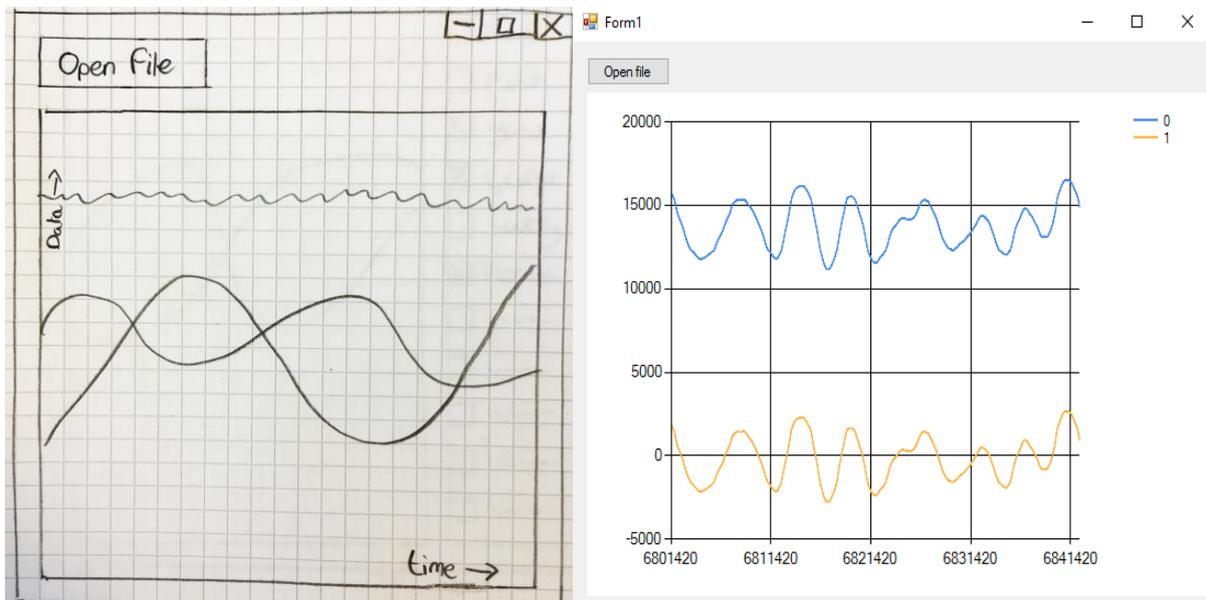


FIGURE 12 - PLOTTING CSV DATA (LEFT - SKETCH, RIGHT - DESIGN)

However, this method relies solely on the CSV-file that has been loaded and not on the user's choice. The user has to be able to select relevant data itself as well, for the system to function properly. Therefore, the Treeview function was added to the system, making it more open for user input. The Treeview function is a function that is build-in in Visual Basic. It is a structured way of displaying data, and in combination with checkboxes, it can make the user select preferred data. It's structure is divided in branches, hence the name.

The second design is shown below:

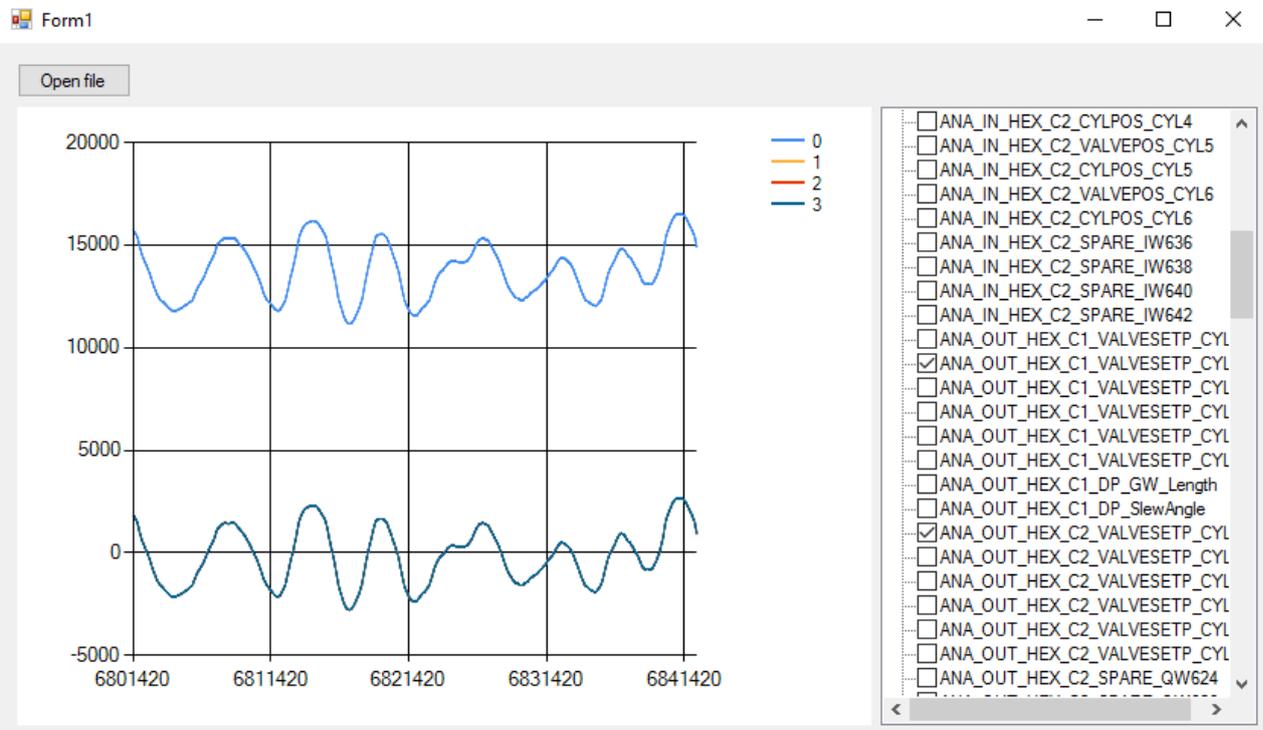
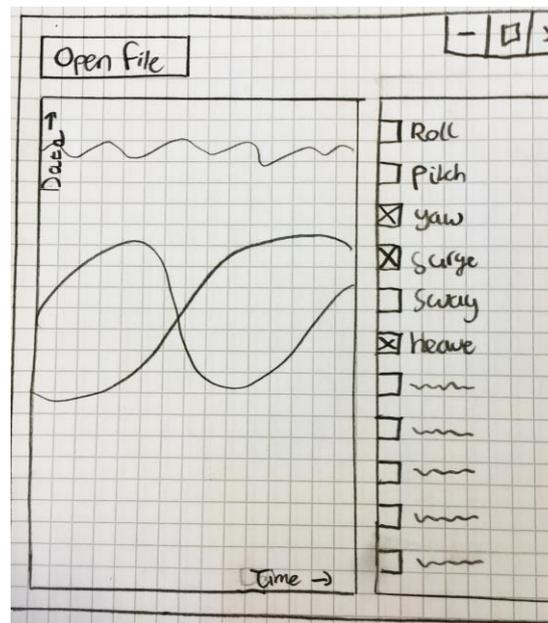


FIGURE 13 - TREEVIEW FUNCTION (UPPER - SKETCH, LOWER – DESIGN)

6.1.1 Workability

To increase workability, several other function have been added to the tool.

X-axis:

The tool is only relying on time as X-axis, and eventually it should be interchangeable, so that any item in the treeview can be used as X-axis. This is possible by pressing the “Change X-Axis” button.

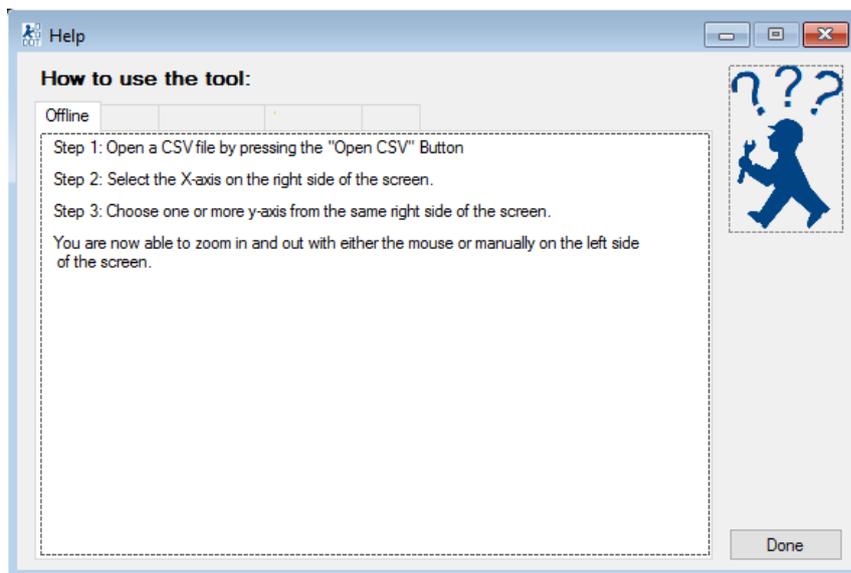
Adjustable Scale/View:

Besides the interchangeable X-axis, the view/scale of the axis is only relying on the automatically assigned scale, which comes forth out of the Auto Scale function in Visual Basic.

To make it possible for the user to adjust the scale, several textboxes are added to read in the user’s preferred scale or view. The user can then apply the settings by pressing the “Apply” button or reset to its previous values by pressing the “Reset” button.

Help Button:

A help button is added, to give the user tips on how to use the tool. The following window will pop-up:



Select / Deselect All:

A select and deselect all button are added to let the user select all items in the treeview or to deselect them all.

Error Graph:

To add more functionality to the actual data, an error graph function is added.

This graph is the difference between one graph and another. For instance the difference between the set point of a valve and its actual position.

The calculation is done by the following formula:

$$Err_{graph}(x_{errgraph}) = Fa_{err}(x_{errgraph}) - Fb_{err}(x_{errgraph})$$

EQUATION 2 – ERROR GRAPH CALCULATION

Where:

$x_{errgraph}$ = A value that runs from the first item in the CSV file row until the last.

$Err_{graph}(x)$ = The resulting graph.

$Fa_{err}(x)$ = One of two functions of which the error graph needs to be calculated for.

$Fb_{err}(x)$ = One of two functions of which the error graph needs to be calculated for.

If the example of the cylinder's set point and its actual position is used, and we take the absolute value of the results per data point we get the following result:

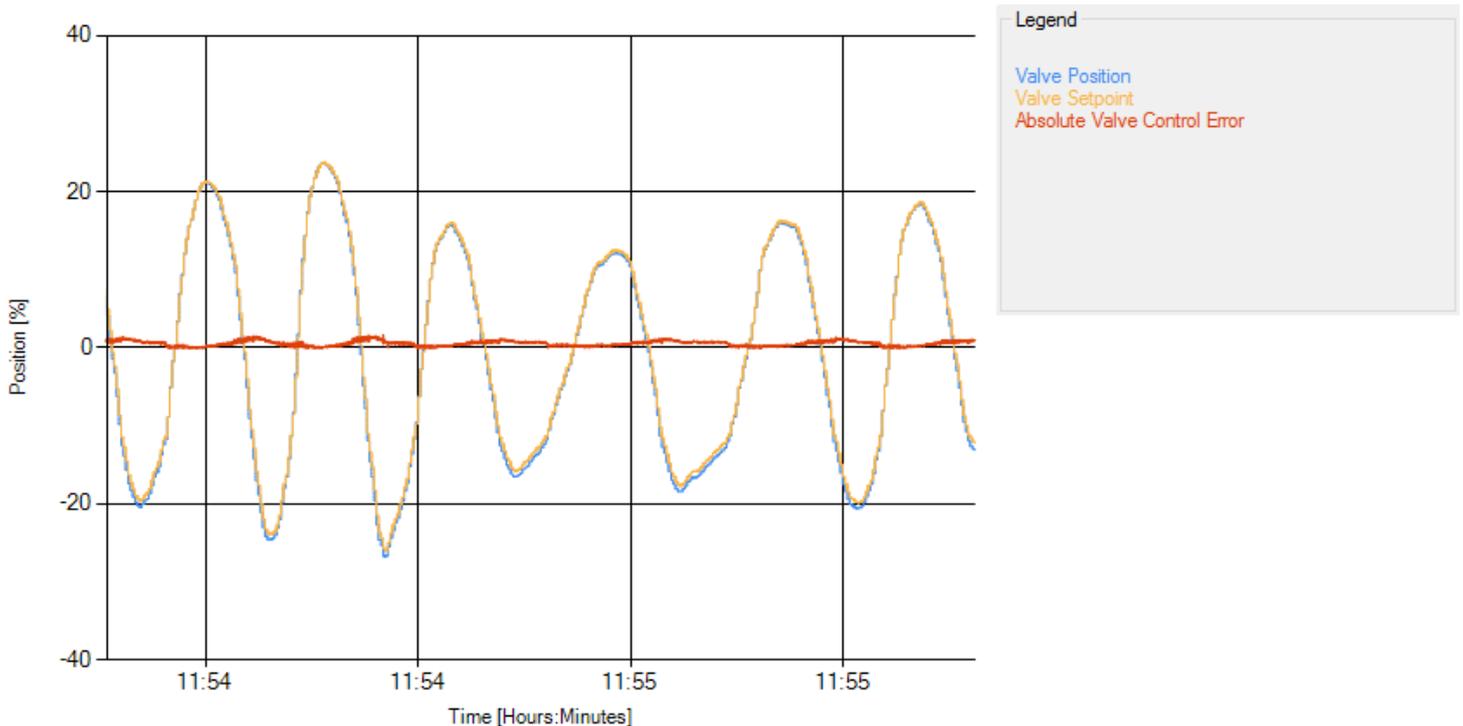


FIGURE 14 - ERROR GRAPH FUNCTION

The red line is the error graph. In the most ideal situation, this line would be completely flat, and thus totally 0, indicating that the cylinder is perfectly following its set point. In reality this will never occur, but it will actually fluctuate around 0.

Absolute zero:

Another function is the absolute zero function, which scales the X-axis values to start from 0. This convenient when debugging the system, to accurately see the amount of passed time. This will help the Motion Control engineers to see when the PLC has started.

Information section:

An information section is added, so that the user can see relevant information when a certain action occurs. For instance the selection of a x-axis or an error that occurs.

Loading bar:

A loading bar or progress bar is added, to let the user know the program is still working when adding a big CSV-file. Big CSV-files will take more time to load and the progress bar will prevent the user from thinking that the tool is frozen or crashed. The bar is displayed in the information section mentioned above.

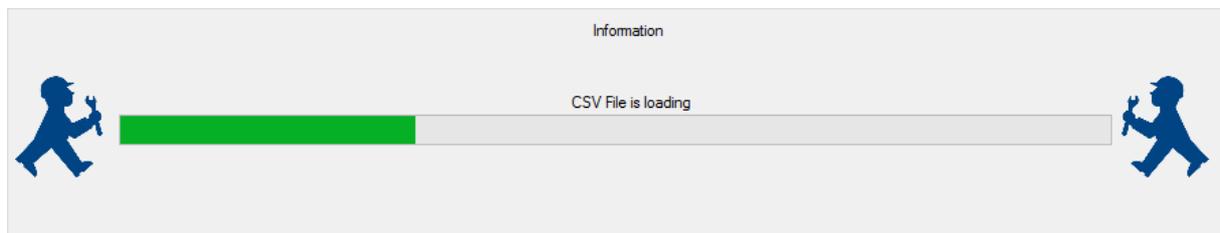


FIGURE 15 - LOADING BAR

Legend text and axis label text:

To make clear which line is which function, legend text has been added. Each line contains the equivalent name as its selected item of the treeview.

Also for the axes, the label texts are shown, to see which function is shown on what axis.

Tab control:

To ensure that the following parts of the tool can fit on the screen, the tab control function has been added, with each tab containing one of the four parts mentioned on page 33.

The tab control is also added to the help menu button, so that the user can see tips per tab.

Open Bin File:

Because the project owners wanted to have the functionality of opening a bin file directly, methods were researched to implement the already existing Bin to CSV converter that has been made by one of the motion control engineers of Ampelmann. There are two ways of doing this. One is building the converter in the tool. The other way is by referencing directly towards the converter and letting its output stream directly towards the tool.

The second way has been chosen, because this ensures that in future release of the converter, the tool doesn't have to be re-written in order to function properly. It will just reference towards the new version of the existing converter.

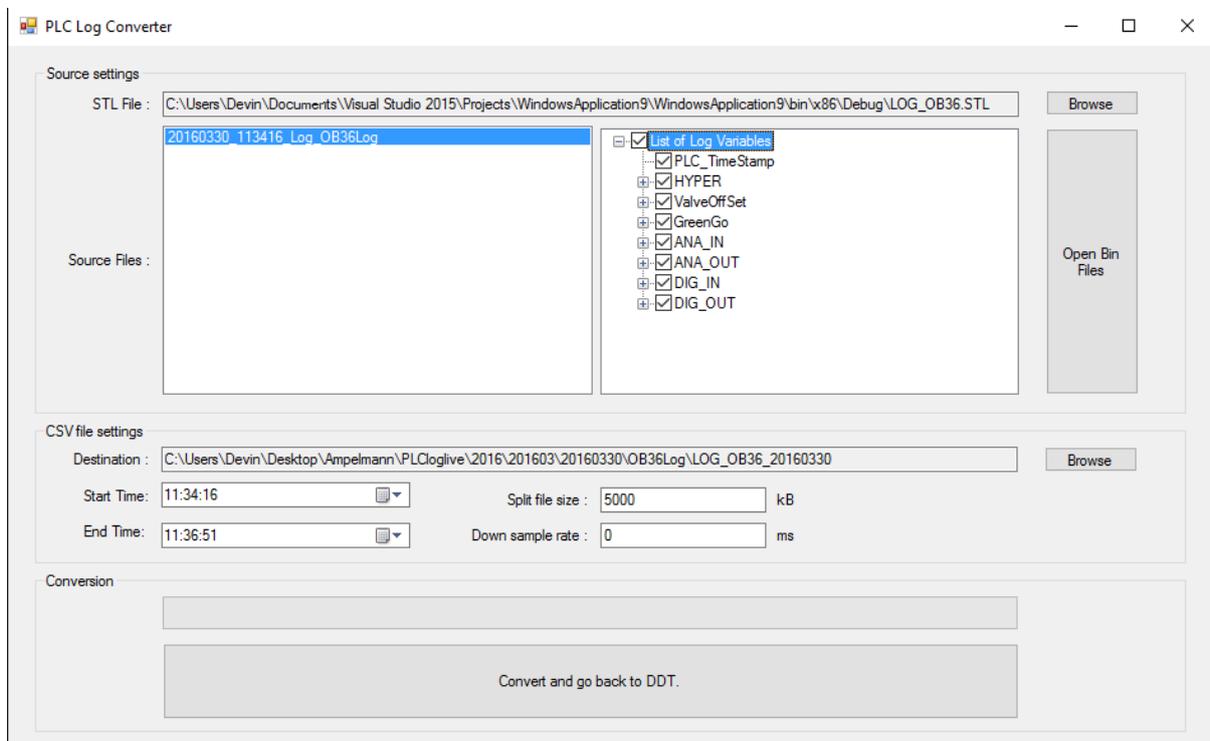


FIGURE 16 - PLC LOG CONVERTER OR BIN TO CSV CONVERTER.

In Figure 16 the converter is shown. The user can load a STL-file, which contains the structure of the binary (.BIN) file. These STL-files were provided by the converter. The user can then load in the binary files and press the “Convert and go back to DDT” button, to convert the binary files to CSV-files and go back to the tool.

Final Result:

The final result for the offline tab is shown below, containing the above mentioned functions.

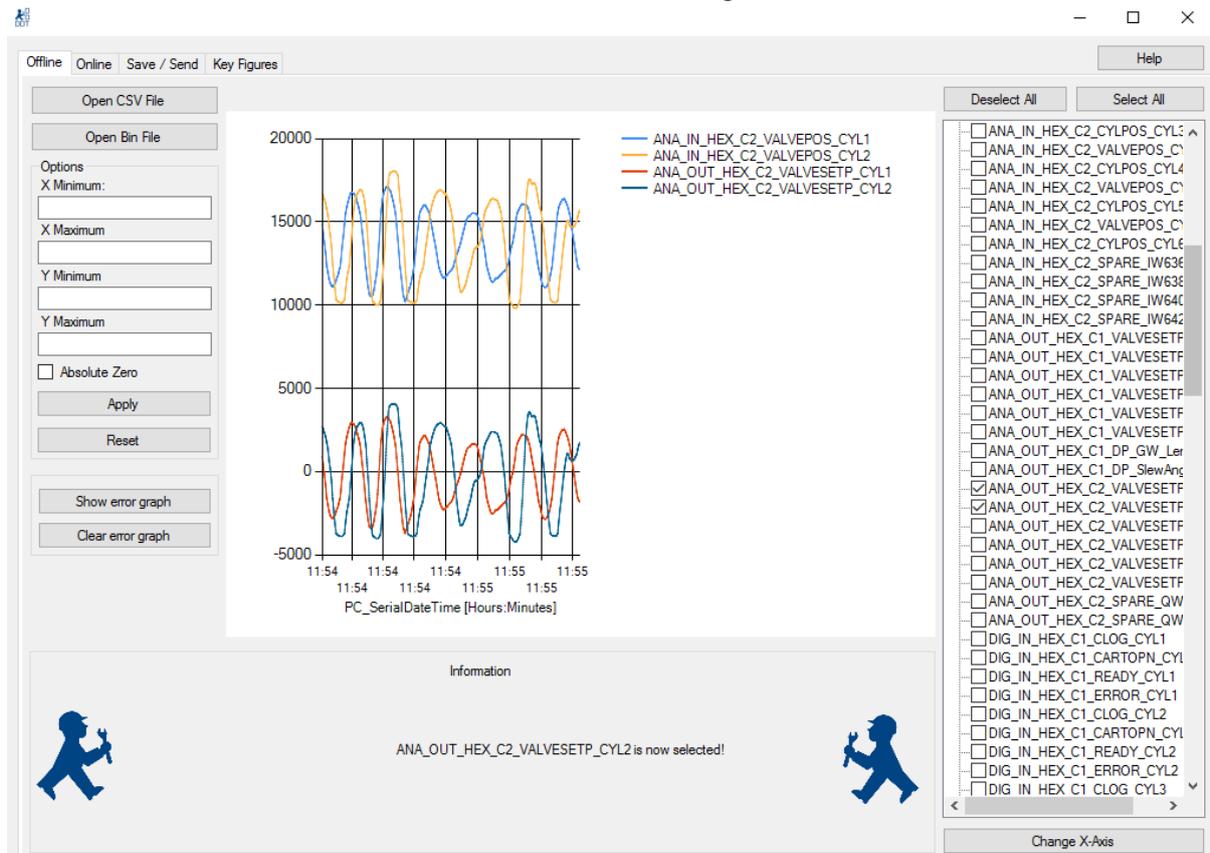


FIGURE 17 - FINAL RESULT FOR OFFLINE TAB

6.2 Key figures

Key figures is another tab in the tab control mentioned earlier on page 39.

The benefit of key figures are that the user of the tool is able to get a quick analysis of most commonly used key figures.

Total absolute cumulative length:

One of the commonly used key figures is checking how far a cylinder has moved since a certain amount of time. So the tool should calculate the total amount of length that a cylinder has travelled.

This is also convenient for future use, for instance for predictive maintenance. If the average travelled amount of distance for cylinders is known at their breaking point, predictions can be made on when a working cylinder may break.

The total absolute cumulative length is calculated by the following formula:

$$Abs_{cumlength} = \sum_{i=1}^{n_{csv}} |x_{csv_i} - x_{csv_{i-1}}|$$

EQUATION 3- TOTAL ABSOLUTE CUMULATIVE LENGTH

Where:

$Abs_{cumlength}$ = The absolute cumulative length.

n_{csv} = The amount of items in the CSV file row.

x_{csv} = A value from the read in CSV file.

Mean:

The mean of a function is used commonly too. To calculate the mean, the following formula has been used:

$$Mean = \sum_{i=1}^{n_{csv}} \frac{x_{csv_i}}{n}$$

EQUATION 4 - MEAN OF A FUNCTION

Where:

$Mean$ = The mean of a function.

n_{csv} = The amount of items in the CSV file row.

x_{csv} = A value from the read in CSV file.

Maximum:

The maximum of a function can be critical information, especially when an error has occurred peaks are relevant. The maximum of a function is also commonly used for debugging and analyzing the Ampelmann systems.

To calculate the maximum, the following formula has been used:

$$Max_{func} = Max\{x_{csv_i} | i = 1 \dots n_{csv}\}$$

EQUATION 5 - MAXIMUM OF FUNCTION

Where:

Max_{func} = Maximum value of a function.

n_{csv} = The amount of items in the CSV file row.

x_{csv} = A value from the read in CSV file.

Minimum:

The minimum of a function can also be critical information.

To calculate the minimum, the following formula has been used:

$$Min_{func} = Min\{x_{csv_i} | i = 1 \dots n_{csv}\}$$

EQUATION 6 - MINIMUM OF FUNCTION

Where:

Min_{func} = Maximum value of a function.

n_{csv} = The amount of items in the CSV file row.

x_{csv} = A value from the read in CSV file.

Average Control Error:

The average control error of two functions is convenient to know, because the error rate can tell whether to adjust the gains of the control loop to get an even better result, which is that the error function becomes as close to zero as possible.

The average control error is calculated by the following formula:

$$AveCon_{err} = \sum_{i=1}^{n_{csv}} \frac{Fa_{ace}(i) - Fb_{ace}(i)}{n_{csv}}$$

EQUATION 7 - AVERAGE CONTROL ERROR

Where:

$AveCon_{err}$ = The average control error of two given functions a and b.

$Fa_{ace}(i)$ = One of two functions to calculate the average control error for.

$Fb_{ace}(i)$ = One of two functions to calculate the average control error for.

n_{csv} = The amount of items in the CSV file row.

Standard Deviation:

The standard deviation is a number that gives an indication of spread. The number indicates how much the given values differ from each other.

$$\sigma = \sqrt{\frac{\sum(x - mean)}{n_{csv}}}$$

EQUATION 8 - STANDARD DEVIATION

Where:

σ = The standard deviation.

x = The current value.

$Mean$ = The mean as calculated in Equation 4.

n_{csv} = The amount of items in the CSV file row.

6.2.1 Design

With above functions in mind, the design of the Key Figures tab has been made.

The formulas used, have to be applied to data that has been load in on the “offline” tab. To make it easy for the user to understand how to apply these functions, the same treeview structure has been used in order to have a clear overview.

The formulas selection has been implemented inside an option window, wherein the user can select which formula he/she wants to apply to the selected data.

To show the key figures on screen, a ‘show’ button has been added.

To ensure stability and speed, the formulas can only be selected one by one. In this way, the tool can calculate each formula once, preventing it from crashing once the tool is on the Box – PC.

The results are displayed in a separate field, with dynamic checkboxes. Each checkbox will be created dynamically, each time the user presses the ‘show’ button. In this way, almost unlimited amount of key figures can be displayed on screen.

6.2.2 Workability

To increase workability, several extra functionalities have been added.

Select / Deselect All:

A select and deselect all button are added to let the user select all items in the treeview or to deselect them all.

Inverse Selection:

An inverse selection option is added to ensure that the user is able to select the other keyfigures easily and unselect the current ones.

Delete:

A delete button is added to ensure that the user is able to delete the key figures that are unwanted / unnecessary.

Clear All:

A clear all button is added to let the user clear the total field of key figures.

Round / Truncate:

Round and truncate buttons are added to gain the mobility to round or truncate the selected key figure. This can be done by selecting the amount of decimals and pressing the corresponding buttons.

Formula(s) used:

To fill up the empty space between the treeview and the options menu, a used formula window has been created. In this way the user can see which formula has been used in order to gain the shown result.

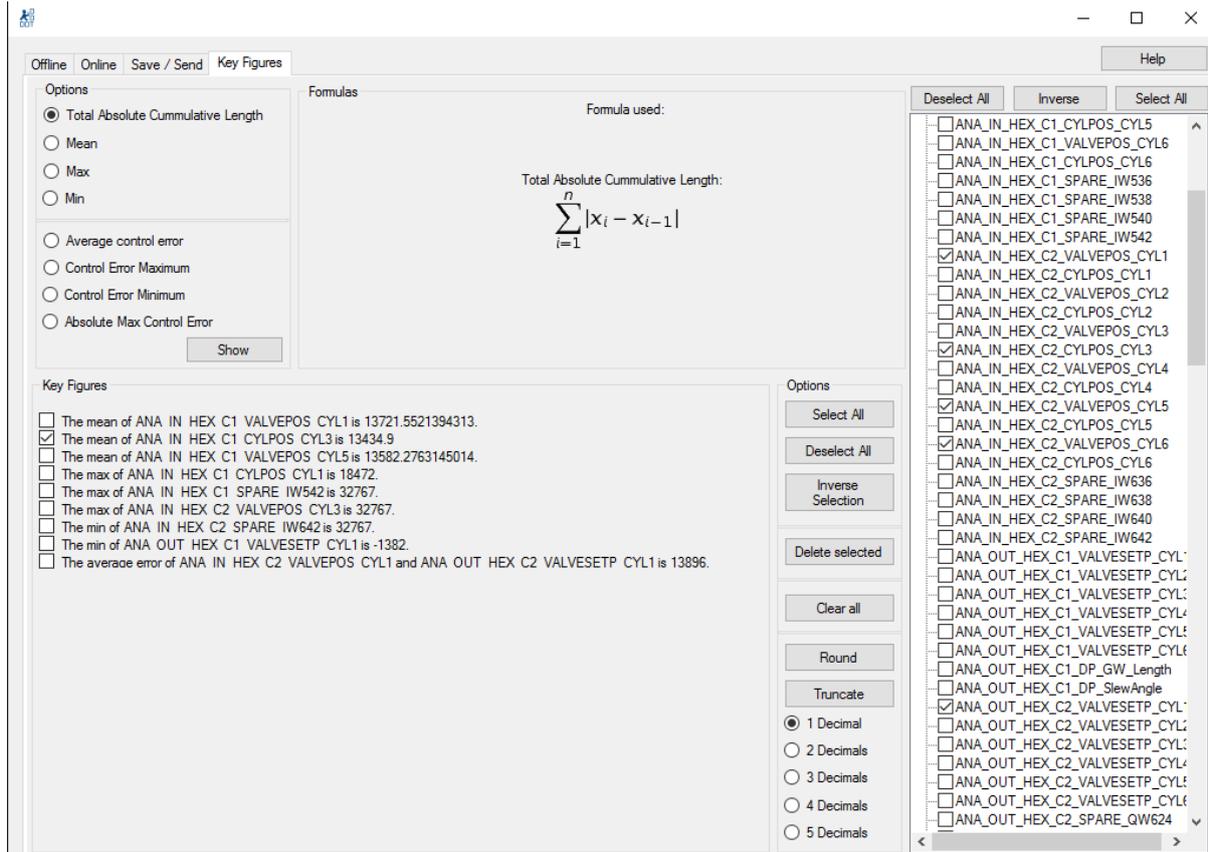


FIGURE 18 - FINAL RESULT FOR KEY FIGURES TAB

In Figure 18, the final result is shown for the key figures tab. In this picture, some key figures are shown. The one that has been checked, has its values rounded to 1 decimal. This shows the round function.

6.3 DDT

One of the requirements of Ampelmann was to implement a name which would not be forgotten by the motion control engineers. After brainstorming with the mentors, the name DDT was brought up. Simple in pronunciation and to remember. It stands for Devin's Data Tool, referring to the name of the maker of the tool. This has been translated into an Ampelmann logo, seen below:



FIGURE 19 - LOGO DDT

6.4 Save / Send

The save and send tab, is a tab wherein the user is able to export data, either via saving to the computer or via mail.

Saving and sending is one of the requirements of the project owners. Both are ways of exporting data and ways of compressing loads of data into smaller useful parts.

Several save and send functions have been defined by Ampelmann:

- Exporting graphs
- Exporting used CSV treeview items
- Exporting used key figures
- Exporting functions with specific cut off values

All these functions are clickable via the corresponding checkboxes, making it able to select one or two, three, etc. of the available options.

The information about a succeeded or failed export is shown within a separate box, containing the names of either save info or send info, which are related to the respective functions.

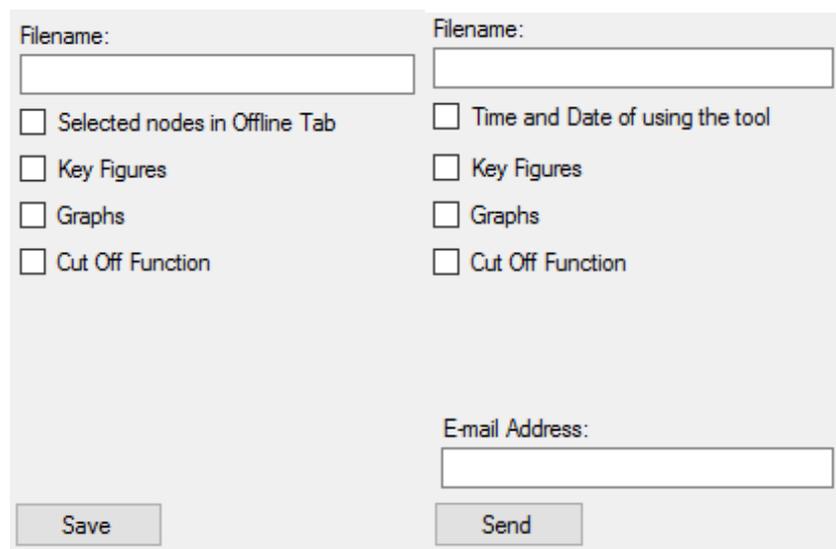


FIGURE 20 - SAVE AND SEND MENU OPTIONS (LEFT: SAVE , RIGHT: SEND)

In figure above, the explained options are shown as they are designed inside the program.

A file name has to be given in order for the functions to work, as they will give the files a dedicated name. For sending, an email address is required as well.

Each saved file will be saved in a folder that contains the name of the date of the day that the functions are used. In this way, retrieving the used functions is convenient, because the user can search on the date as well instead of only on the file name.

Savefile 23 05 2016

FIGURE 21 - NAMING OF A FOLDER WHEN EXPORT FUNCTIONS ARE USED

In the figure above, the earlier mentioned way of saving files is shown. These folders contain the selected items of the options window of the send function.

An example of its contents is shown in the figure below:

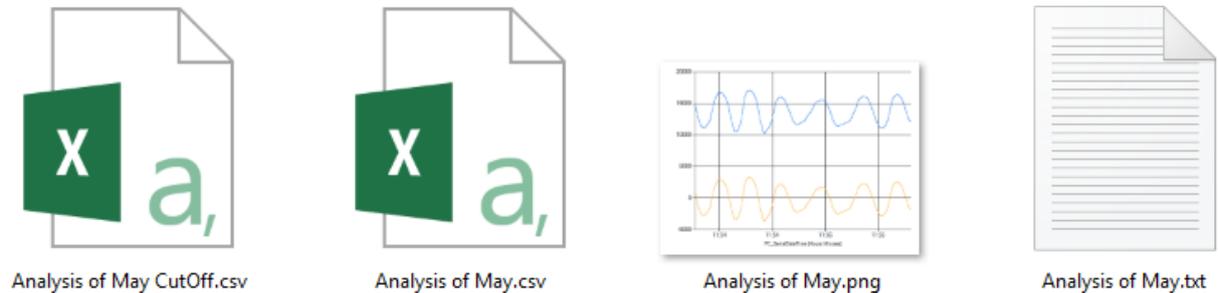


FIGURE 22 - CONTENTS OF DEDICATED FOLDER

These are the results of the save function. The first item in the above figure, contains the result of the “cut-off” function.

This cut-off function is a function, wherein the user can say whether to look for minima or maxima and cut out values above or below these minima or maxima. In this way, the end-user or motion control engineer is able to check when a selected function has passed a certain key value, and this function will then crop the values, so that a reduced file with only the requested data will come out as output. This makes the debugging of the Ampelmann system easier, since it deletes unnecessary data and only keeps the wanted data that is beneath a certain key figure.

Once the cut-off function has been used, the tool automatically adds a line to show the user where the cut-off function has cut off the function. This is shown in the picture on the following page with as example a cut-off value of maximum 15000.

In Figure 24 Figure 24 - Result of cut off, on the next page, the result of the cut-off function is shown. The flat lines indicate that there is no data on those times, and the chartview compensates by filling up the gaps. A recommendation is to crop the flat lines at the top, because they add no significant benefits.

These functions can also be used as a warning type of function, warning the user whenever a certain criteria has been passed, resulting for instance in an error that lets the user know there is something wrong.

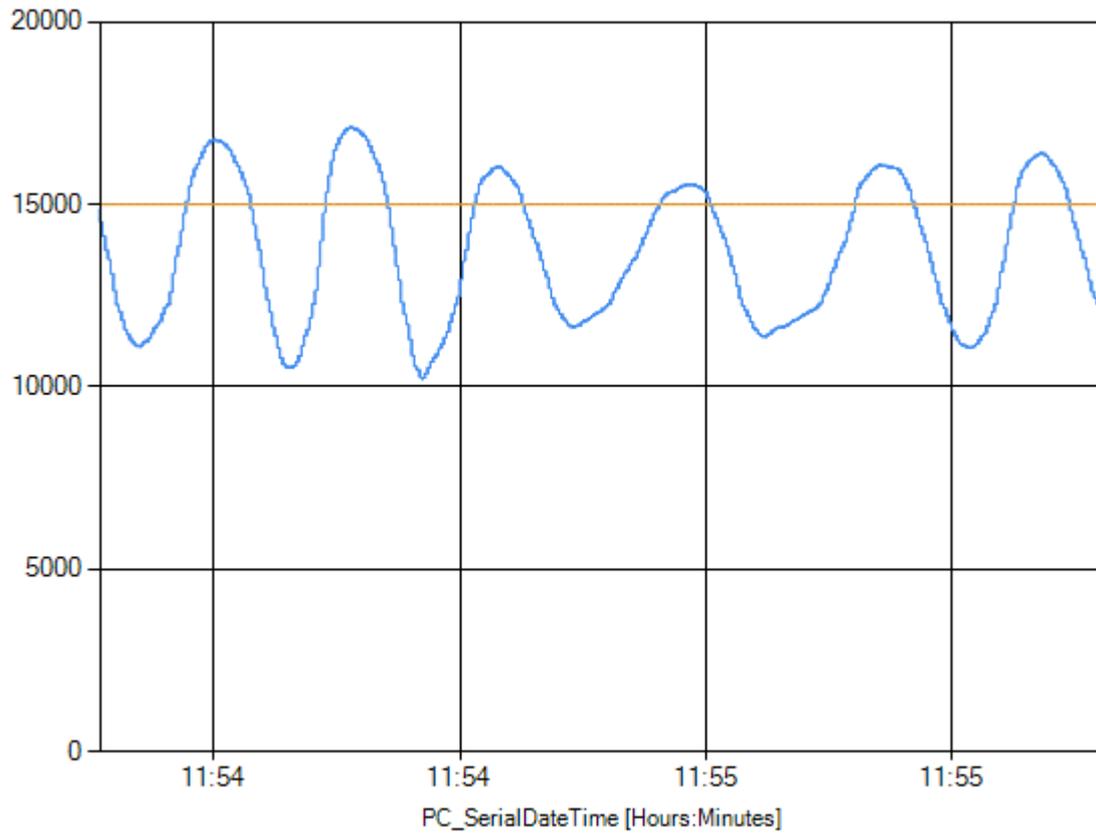


FIGURE 23 - CUT OFF LINE

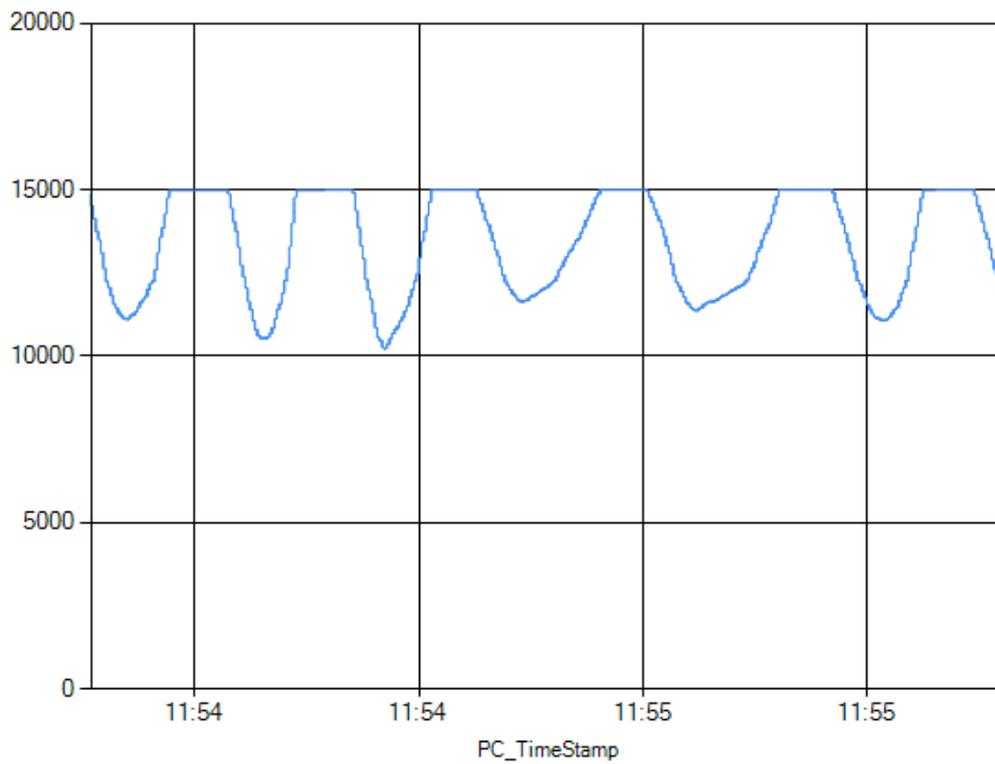


FIGURE 24 - RESULT OF CUT OFF

6.5 Online / Query

To let the end-user request specific data from the Box-PC, there should be a online tab or query tab on the tool. Queries are commands that can be send towards a database to get potential results.

After onset of the research, the project owners decided to focus more on the analysing part and to only set-up a basic template for in the future for the online or query mode. This due to the scope and the availability of going online via a shared desktop application such as TeamViewer.

The template has been created with the idea in mind that the end-user should be able to use the same program to be either the sender as the receiver, respectively the client or the server.

Therefore a design has been made wherein the user is able to let the program know whether he/she is either client or server.

This is done by using radiobuttons, to ensure that not both options can be used at the same time.

6.5.1 Client mode

In client mode, the user can press the connect button, which will execute a command that sends a TCP message throughout a predefined IP-address. Whenever there is a server listening, and it replies with the suspected handshake code, a TCP connection has been set up. From that moment on, the user would be able to send queries towards the server.

If an error occurs, a “Display Error” button will appear. Once pressed the user will get the information needed to understand the occurring error.

6.5.2 Server mode

In server mode, the program sets up a server that is constantly looping and trying to find a client to connect with. Once the server gets a TCP request, it replies with a handshake code to confirm the connection and thus setting up the two-way TCP connection.

6.5.3 Design

In the future, this tab can be extended so that the queries can be send and replied to. For now, the basic template has been set up and on the next page, in Figure 25 the tab has been shown for the queries, with a connection Set up.

6.5.4 Modularity

Because the query tab is in the form of a basic template, the modularity of the tool is hard coded. This is visible in many ways. For instance the exporting of data always occurs on the same way and in the same format, ensuring that future programmers / users are able to proceed with every shape of CSV-files, because the tool can read them. Also the code is commented intensively, ensuring that most engineers will know how to continue in the program for future functions.

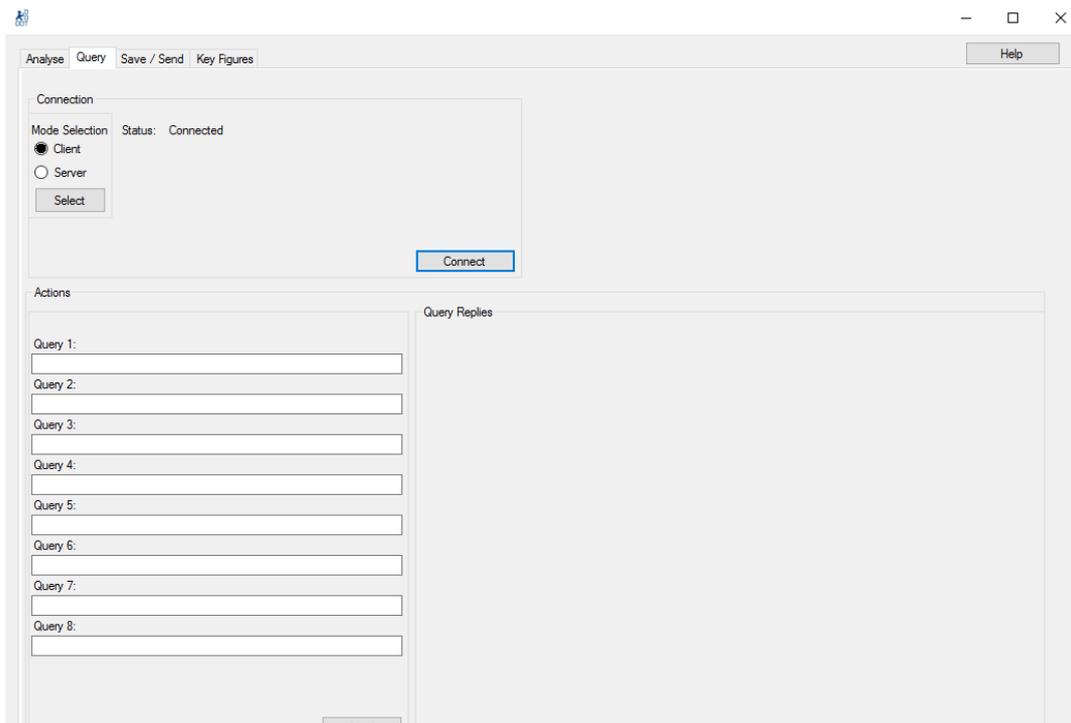


FIGURE 25 - QUERY VIEW CONNECTED WITH SERVER

As mentioned before when a connection fails an error message occurs, resulting in the following image:

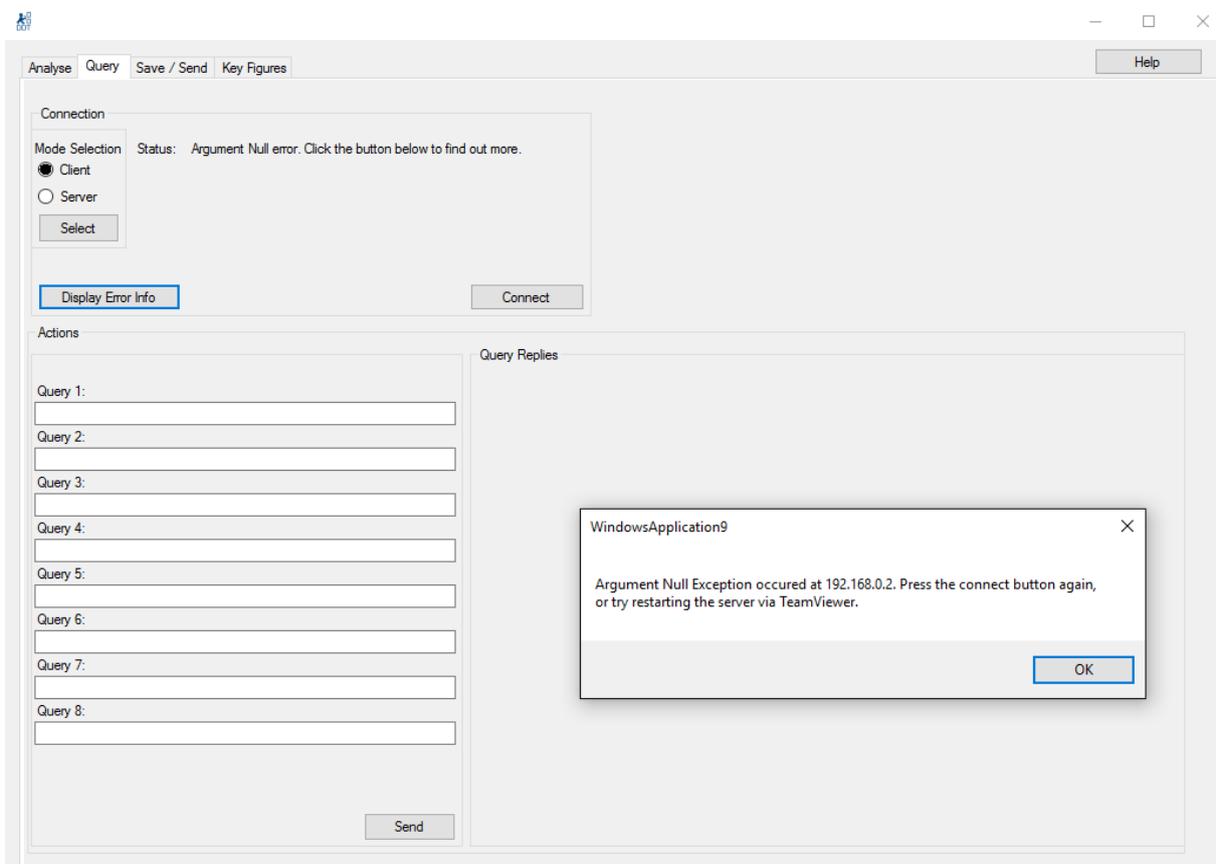


FIGURE 26 - ERROR DISPLAYED AND DISPLAY ERROR BUTTON PUSHED

7. Programming

To program the tool, the main users or actors need to be clear. For this tool the main actor is the Ampelmann Motion Control engineer. The functionality of their input is shown below in a use case diagram. The other actor is the tool itself, which is a system actor.

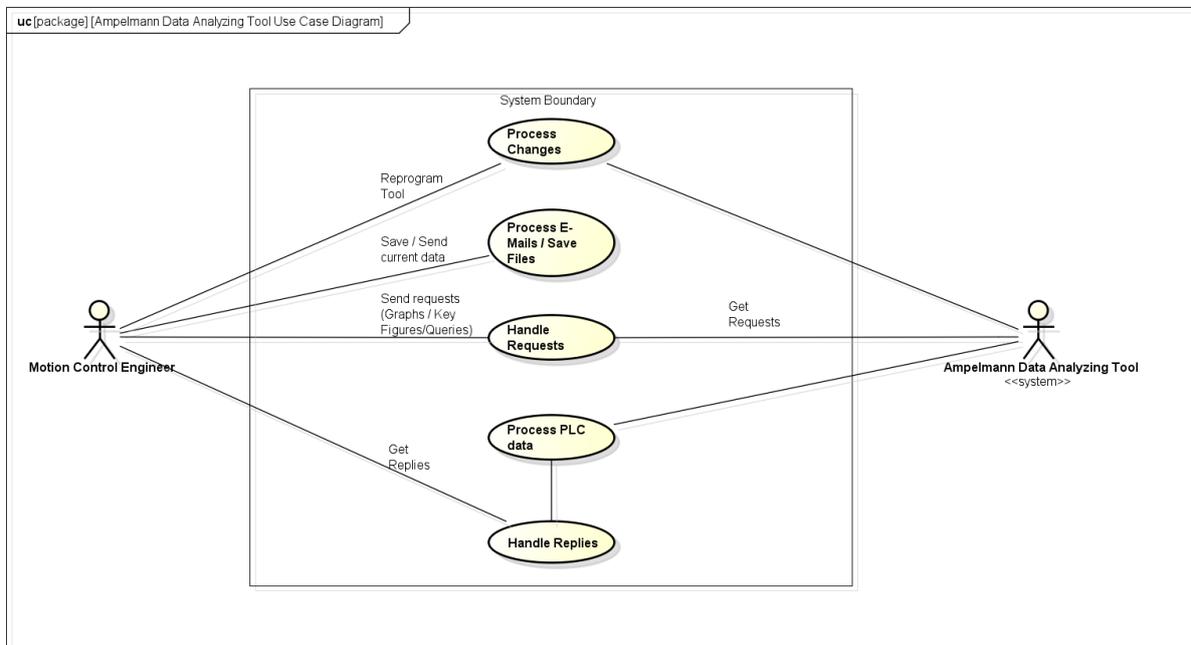


FIGURE 27- USE CASE DIAGRAM

The in- and outputs of the system are shown in the use case-diagram in the figure above. These inputs and outputs need to be programmed in order to get a good proof of concept.

The Motion Control engineer has to be able to send requests to the tool like graphs, key figures and queries. Also saving and sending functions have to be realized in order for the Motion Control engineers to save the data or sending it towards the Ampelmann headquarters.

Each function within the program has to be determined on beforehand to ensure that underlying relations are known. This will result in a more clear view on how to program the different parts in such a way that the other parts will be able to work with it as well.

As said in the Design chapter on page 33, there are three main subjects:

- Plotting Logged Data / Offline
- Key Figures
- Saving or Sending

For each of these subjects, the corresponding functions are shown in a Block Definition Diagram:

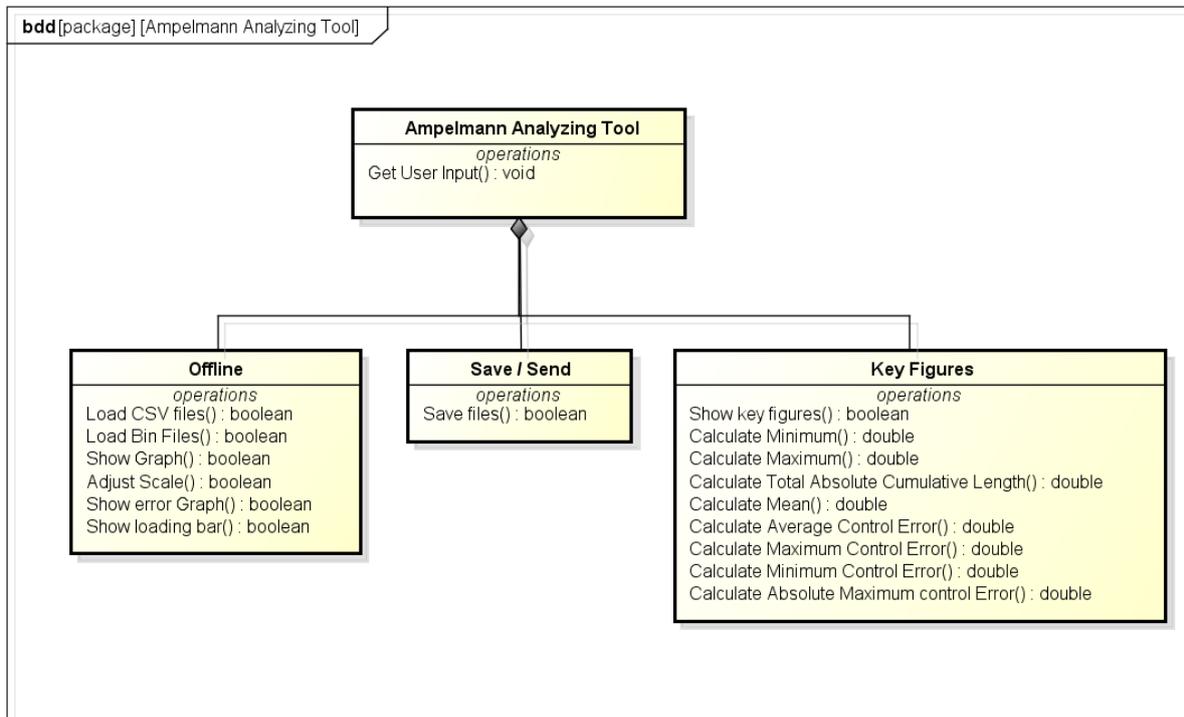


FIGURE 28 - BLOCK DEFINITION DIAGRAM

Also for each of these subjects a state machine diagram is made to give a detailed overview of the functions it contains. At the end a state machine is given for the connection between all of them.

Each of these functions are programmed in such a way that whenever the exit button is pushed, the program quits immediately.

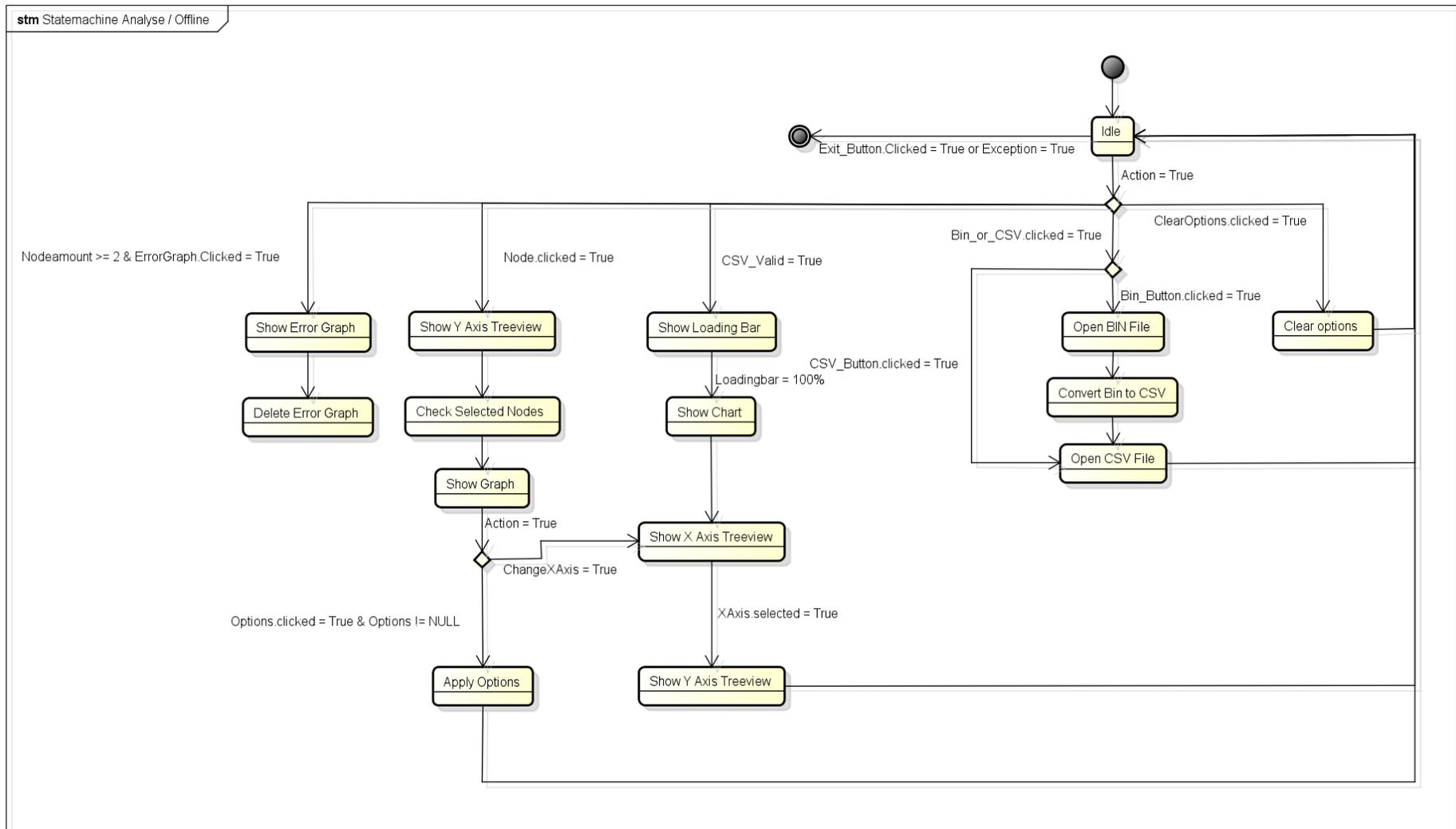


FIGURE 29 - STATE MACHINE FOR SUBJECT: ANALYSE / OFFLINE

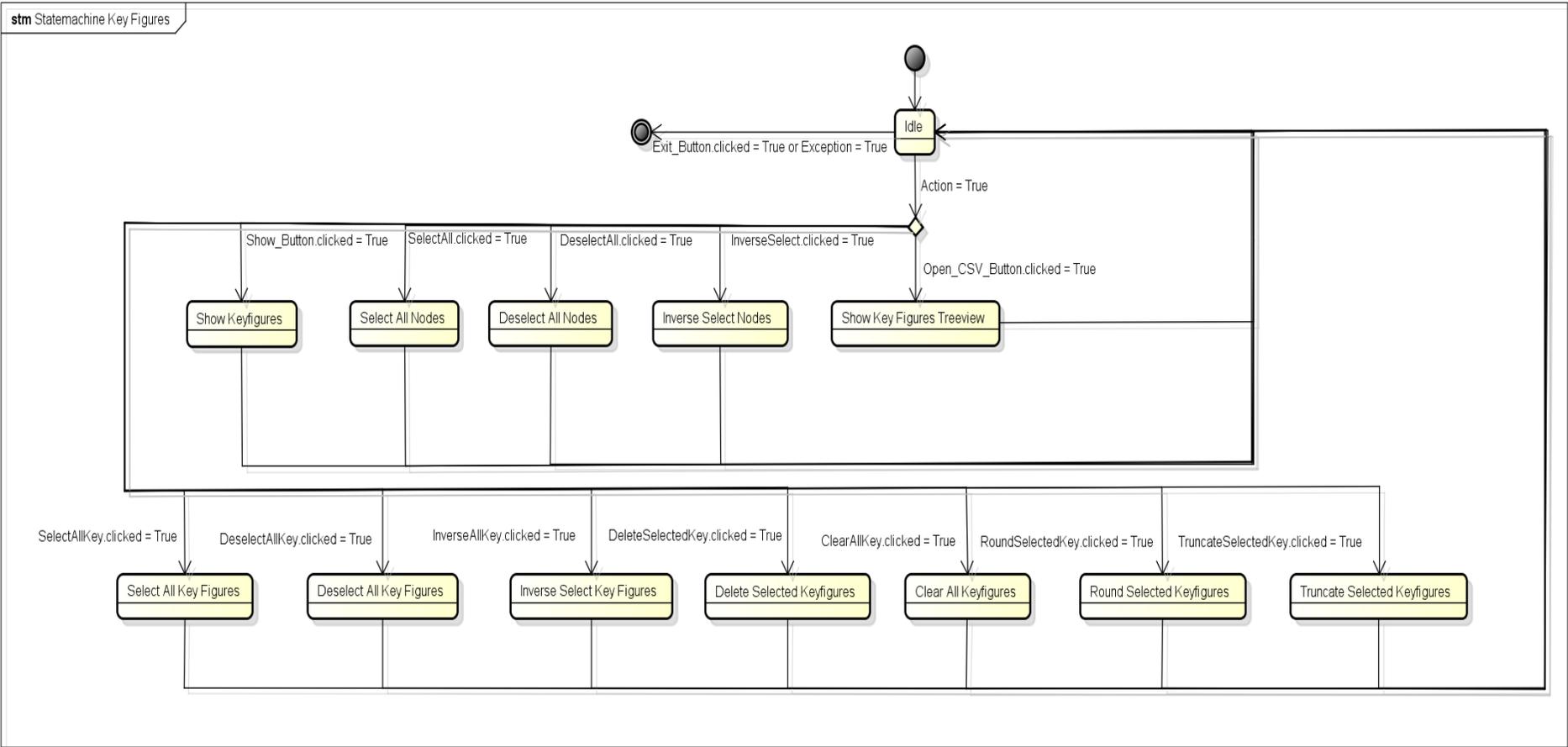


FIGURE 30 - STATE MACHINE DIAGRAM FOR SUBJECT: KEY FIGURES

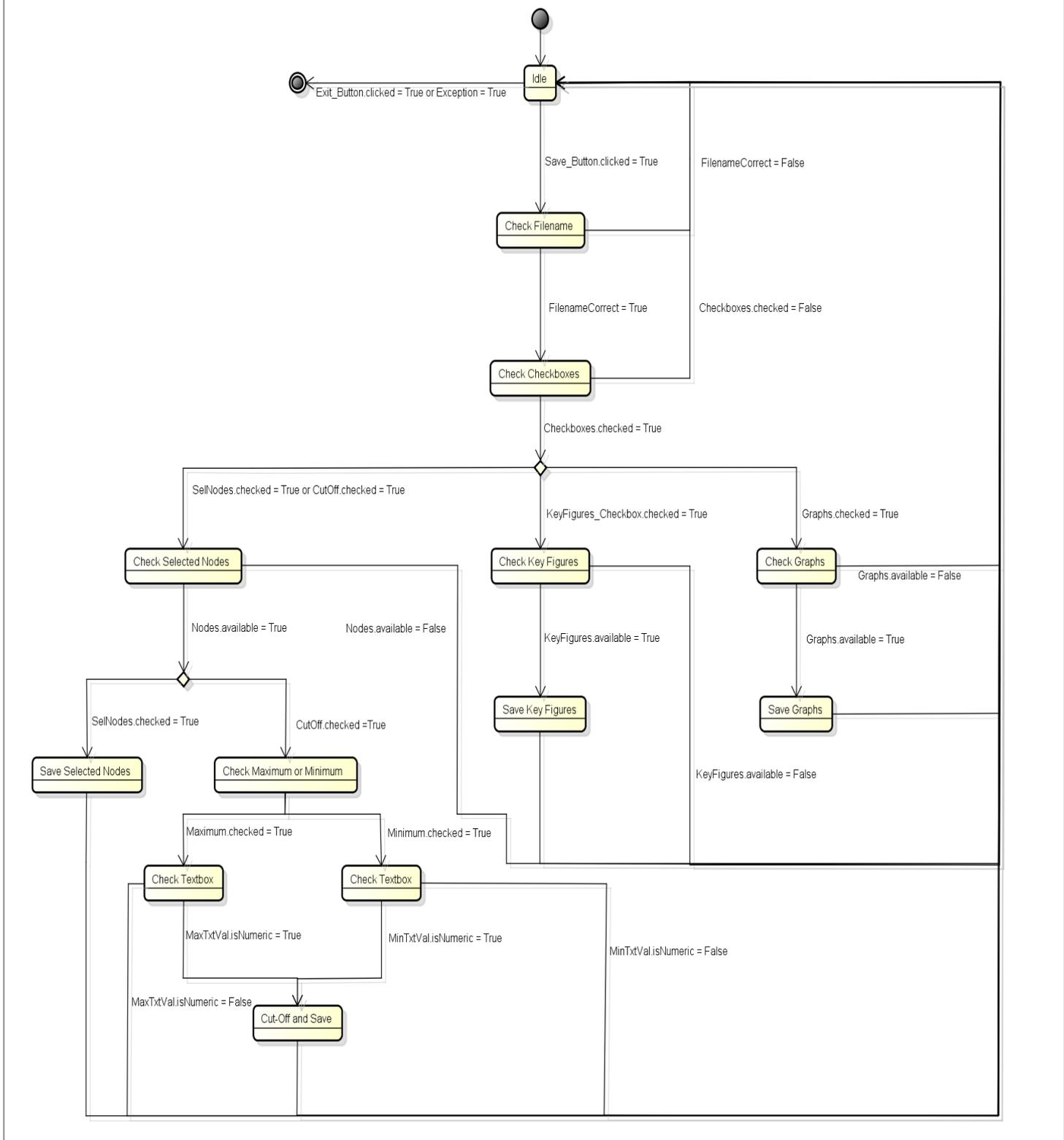


FIGURE 31 - STATE MACHINE FOR SAVE

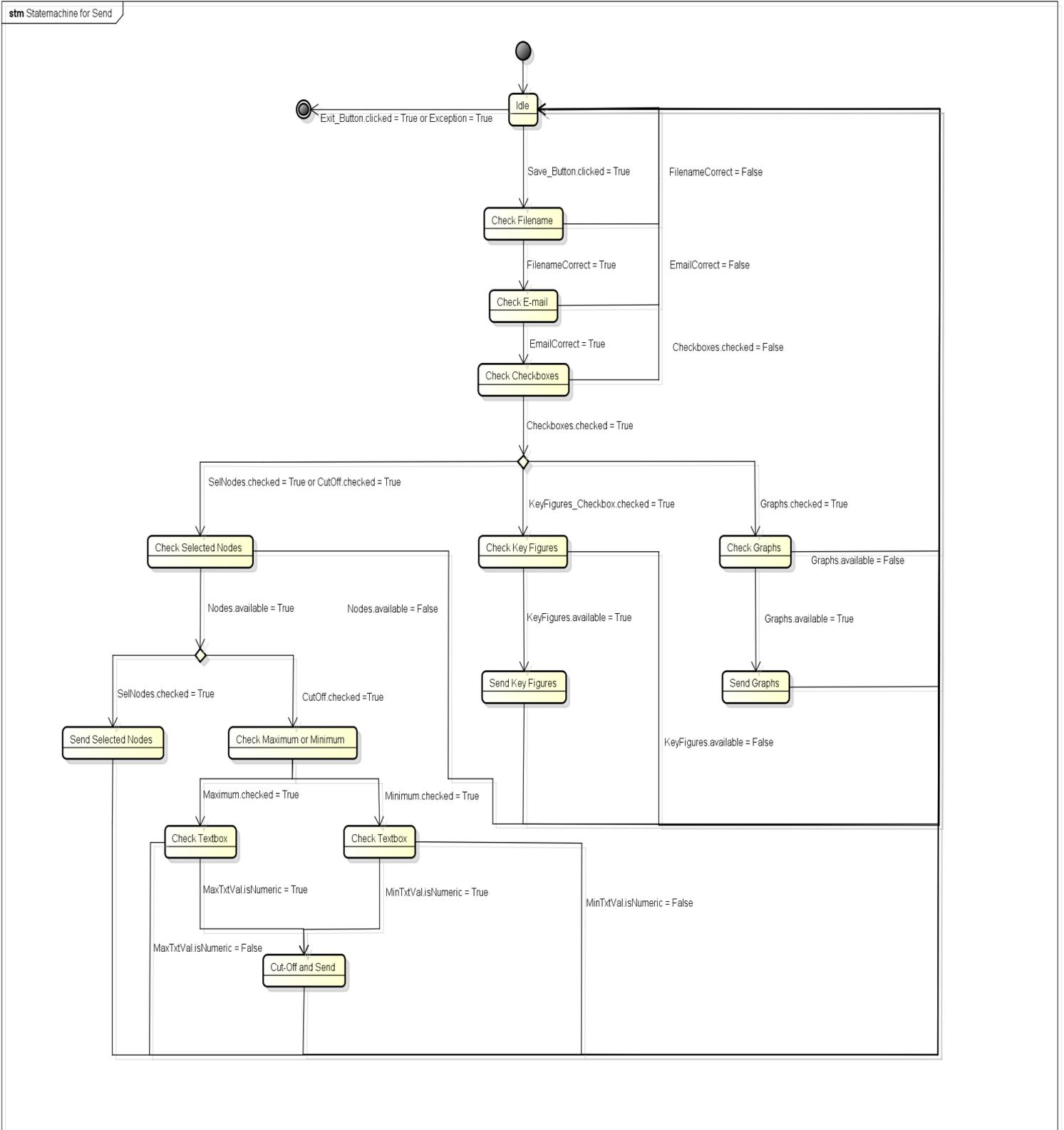


FIGURE 32 - STATE MACHINE FOR SEND

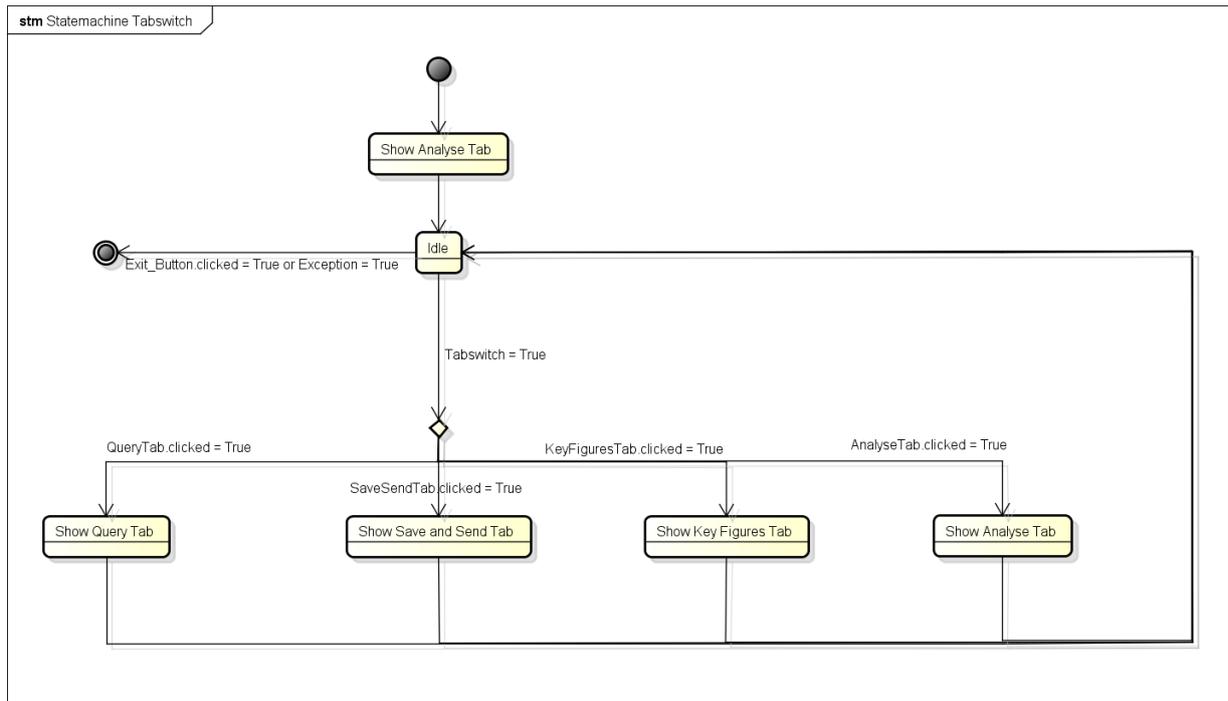


FIGURE 33 - TABCONTROL FUNCTION - CONNECTIONS BETWEEN DIFFERENT TABS

In the figure above, the interconnection between all tabs has been shown. If the user starts the tool, the program starts with the analyse tab. After that, every time a tab has been clicked it switches to that particular tab. In every tab, an Idle state has been implemented. This is the state wherein the tool is when nothing happens. As mentioned before, the user is always able to exit the tool whenever he/she is in the Idle state.

The programming is built up in an event based structure, which lets the program go to idle state until an event occurs. This ensures that no unnecessary background processing time is required.

To ensure stability, the loop in the Query/Online tab for the server mode is written in a multi threaded way, so that one thread is running and looking for clients, while the other thread maintains computational abilities for the rest of the program.

The programming code can be found in Appendix V – Program.

8 Proof of Concept

In order to proof that the concept is working, the designed concept must be tested. This will be done in combination with a test case. The test case describes the different amount of tests and how these tests are done step by step. Also the criteria on which the test is passed or failed is mentioned.

These tests are executed on a test environment, which has to be created as it is one of the requirements.

The test environment should exist of the following parts in order to get as close as possible to a real Ampelmann system:

- Moxa (Box-PC)
- Ethernet and/or Wi-Fi for PLC dependencies
- Power Supply
- PLC that imitates the PLC in the Ampelmann control panel one
- PLC that imitates the PLC in the Ampelmann control panel two
- DDT on monitor that is connected to Moxa

If we put this all together, we get the following result:



FIGURE 34 - TEST ENVIRONMENT

In the previous figure (Figure 34 - Test environment), the mentioned parts are installed, where:

- Yellow = Moxa (Box-PC)
- Green = Ethernet and/or Wi-Fi for PLC dependencies
- Purple = Power Supply
- Red = PLC that imitates the PLC in the Ampelmann control panel one
- Brown = PLC that imitates the PLC in the Ampelmann control panel two
- Blue = DDT on monitor that is connected to Moxa

With the test environment set-up, the test case can be made.

The test case will test the requirements that are testable from the requirements list.

The test case will consist of the following items for testing:

Test 1: Stand-alone test

Test 2: 32-Bit hierarchy test

Test 3: Read in CSV files test

Test 4: Export data test

Test 5: Deriving Key figures test

Test 6: Fool proof test

Test 7: Easy understandable GUI check/test

Test 8: Graph plotting test

The testcase for these tests is displayed on the next page.

Test Case #	Test Title	Test Summary	Goals	Test Steps	Criteria for Pass/Fail	Expected Result	Actual Result	Notes
1	Stand-alone test	Test to check whether the DDT can work stand alone	Validating if the DDT is stand alone.	1. E-mail tool executable 2. Run Executable	Pass: Executable runs without installing Fail: Executable doesn't run without installing	Pass	Pass	--
2	32 - Bit Hierarchy Test	Test to check whether the DDT works with a 32 bit hierarchy	Validating if the DDT is working in a 32-bit hierarchy.	1. Run the program on a 32-bit hierarchy computer. This will be tested on the Moxa (Box-PC). The moxa has got a 32-bit hierarchy.	Pass: Executable runs without errors on hierarchy Fail: Executable doesn't run	Pass	Pass	--
3	Read in CSV files test	Test to check whether the tool can read in CSV files.	Validating if the tool can read in CSV files.	1. Run the DDT 2. Load different sets of CSV files	Pass: Treeview will open up the data from the CSV files Fail: An error will occur or the treeview will not fill with data from the csv files	Pass	Pass	--
4	Export Data test	Test to check whether the tool is able to save and send data	Validating if the tool can send and save data.	1. Run the DDT 2. Load different sets of CSV files 3. plot graphs and request keyfigures 4. Save or Send files.	Pass: Selected items are saved or send on the computer Fail: Selected items are not saved on the computer or send via e-mail	Pass	Pass & Fail	Saving is working, Sending not yet.
5	Deriving Key Figures test	Test to check whether the tool is able to derive certain key figures	Validating if the tool is able to derive key figures from selected data.	1. Run the DDT 2. Load different sets of CSV files 3. Derive key figures	Pass: Able to derive key figures with values that seem logical to the motion control engineers of Ampelmann Fail: Unable to derive key figures	Pass	Pass	--
6	Fool proof test	Test to check whether the tool is fool proof	Validating if the tool is fool proof / stable.	1. Run the DDT 2. Load different sets of CSV files 3. Let different people with no knowledge about the DDT try to use every function	Pass: No crashing or unexpected behaviour occurs Fail: Program crashes or unexpected behaviour occurs	Fail	Fail	Checking treenodes only works from up to down
7	Easy understandable GUI test	Test to check whether the tool is easily understandable	Validating if the GUI is easy to understand.	1. Run the DDT 2. Load different sets of CSV files 3. Let different people with no knowledge about the DDT press every button.	Pass: Other engineers know how to use the DDT, or with help of the Manual or Help button Fail: Other engineers fail to understand the DDT	Pass	Pass	--
8	Graph plotting test	Test to check whether the tool is able to plot graphs	Validating if the tool can plot graphs from read in data.	1. Run the DDT 2. Load different sets of CSV files 3. Plot multiple graphs	Pass: DDT plots graphs Fail: DDT does not plot graphs	Pass	Pass	--

TABLE 8 - TEST CASE TABLE

The test case results are shown in the table on the previous page. These results are elaborated in Appendix III – Test case results.

As seen in the table on the previous page, almost every test succeeded, except the sending and saving and the fool proof test.

For the sending and saving part, the problem can be solved by programming the send part in the following way: Save the files to a local place on the computer, grab the saved files and put them in an attachment in an e-mail. In this way, the user is able to send the data as well.

The problem with the treenodes was that if the user would check a node whilst a node below it in the treeview has been checked, the chart handles the newly checked node as if it is the lowest node in the treeview.

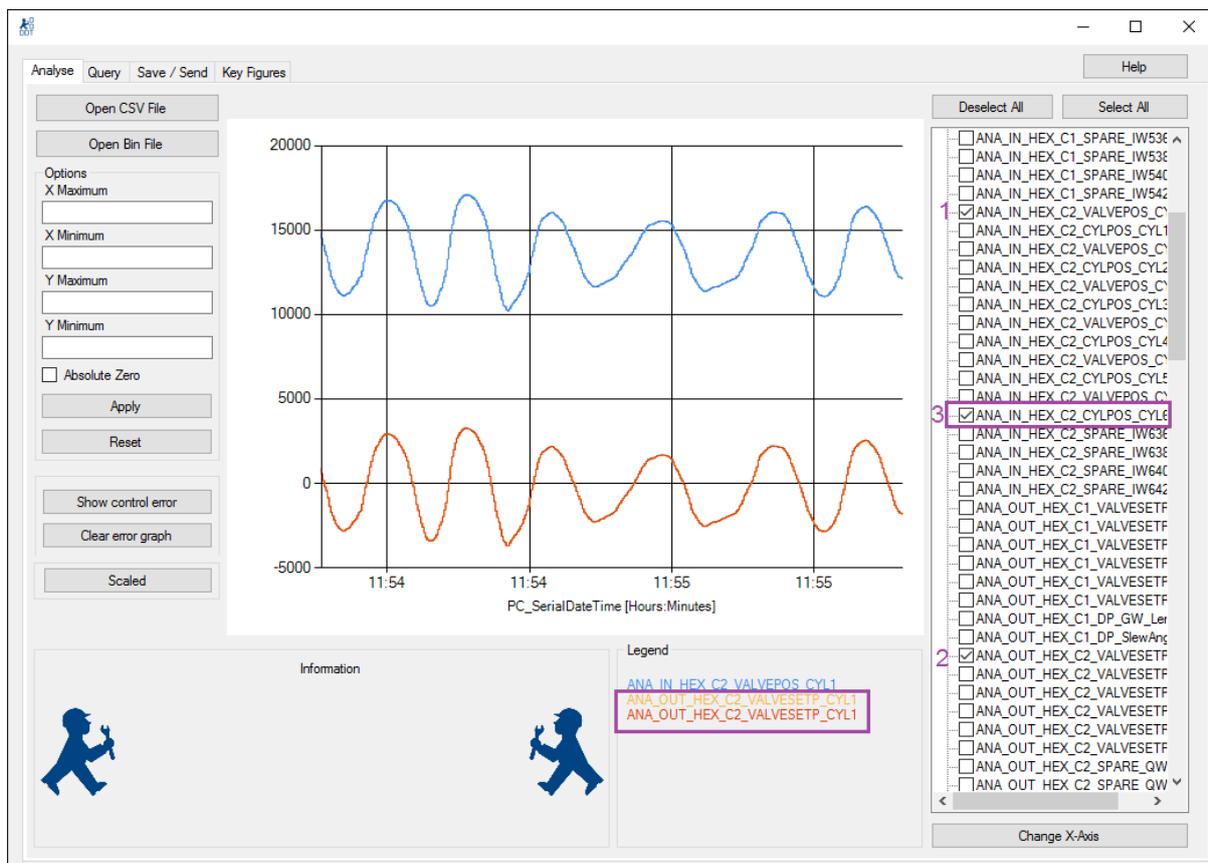


FIGURE 35 – TREEVIEW GRAPH NODE ERROR

In the figure above, the error is more clear to see. The node with a “3” in front of it is checked later than node “2”. But because it is above node 2 in the treeview, the chart thinks it is the last added node and adds an identical graph to the chart and an identical name to the legend. If the two purple boxes in the picture above are compared, this is even more clear to see.

9. Conclusions

After several conceptual ideas at the start, one concept was worked out in detail which resulted in the current design.

The current design meets the requirements of Ampelmann, ensuring that all parties are pleased with the result.

The tool has got a catchy name, Devin's Data Tool or DDT, and has its own logo.

Due to its name, the tool will be remembered and used within the Ampelmann company.

The tool is able to load CSV files and BIN files if the STL files are loaded as well. This ensures that the Ampelmann engineers can use the current data logging tool for the DDT.

Plotting graphs is made easy by the treeview with checkboxes. If the user checks the box, the graph will display and vice versa. Besides the standard requirement of plotting data in a graph, the user also has the ability to apply operations on the chart, allowing them to zoom in or scale the axis to a value selected by themselves.

A basic template has been rolled out for query based information, wherein the user is able to connect with a server, which is the exact same DDT program, only with one marked as server and one marked as client. This ensures stability and consistency in the tool and no other files need to be installed to serve as a server, which makes the tool to still be able to operate as a stand-alone tool.

The exporting of data has been realized by two functions, save and send. Saving items is made easy by giving the user a selection of options, and saving the files locally to a dedicated folder. Sending has not yet been realized, but will be in upcoming weeks.

The Ampelmann engineers are now also able to derive key figures for different kind of purposes. This is made convenient by adding several features that can be applied upon the displayed key figures. These key figures can also be exported via the save and send tab.

10. Recommendations

In order to gain an even better tool, several recommendations to future programmers / users are mentioned below.

The treeview at every tab in the tool can become better, if there are more child nodes used. This ensures that there is more hierarchy, and that there is a better overview of which subject belongs to a certain main chapter. So for instance, every C1 and C2 from the log files, can be a group and inside C1 and C2 there can be another level of hierarchy. These will then be groups of analog inputs, analog outputs, digital inputs and digital outputs. In this way there will be a more simpler overview of the nodes.

Another recommendation is to look at the loop for checking the nodes in the offline / analyse tab, which is at the moment resulting in unexpected behaviour as mentioned within the proof of concept chapter.

A final recommendation is to look at the scaling of the values. At the moment, the values in the CSV files that come from the PLC are raw values, and they have to be calculated to percentages as shown in Figure 14.

For future users, a manual has been added that can help them with learning the tool. It can be found in Appendix IV – Manual for Tool.

11. References

[1] Cerda Salzmänn, D. (2010, 10 7). *We at sea*. Retrieved from <http://www.we-at-sea.org/wp-content/uploads/2013/01/RL5-1-2004-012-Ampelmann-PhD-thesis.pdf>

Computer Weekly. (n.d.). *Computer Weekly*. Retrieved from <http://www.computerweekly.com/feature/Write-once-run-anywhere>

Hayes-Roth, F. (1985, 11 9). *ACM Digital Library*. Retrieved from <http://dl.acm.org/citation.cfm?id=4284.4286>

Kesselring, F. (1954). *Technische Kompositionslehre*. Berlin.

MathPages. (n.d.). *MathPages*. Retrieved from <http://www.mathpages.com/rr/s2-07/2-07.htm>

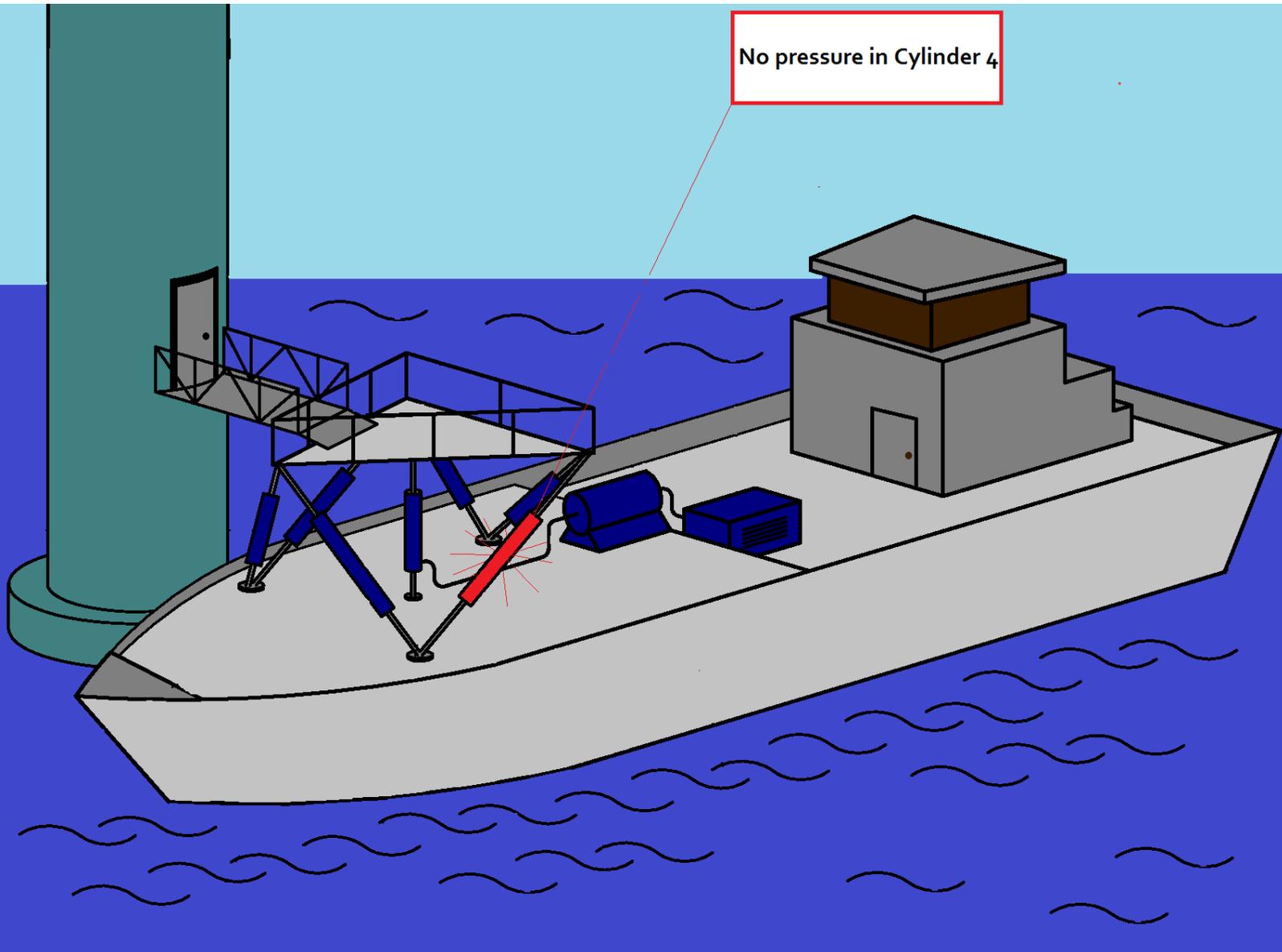
Molenaar, G., & Preeker, S. (n.d.). *Read the Docs*. Retrieved from Read the Docs: <https://media.readthedocs.org/pdf/python-snap7/latest/python-snap7.pdf>

Snap7. (n.d.). *Snap7*. Retrieved from <http://snap7.sourceforge.net/>

Appendix I – Plan

THE AMPELMANN DDT

Development of the AmpeLmann DDT data analyzing tool



AMPELMANN
Kluyverweg 1, 2629 HS, Delft
By: Devin van Tuijll

1. Introduction

Nowadays all the technical tools and work we encounter must be better than its previous version is. In the way of obtaining the best technical solution to a certain problem, one will almost always encounter problems.

To detect and solve these problems, people will use their knowledge to “debug” the used system and find out what the underlying cause was, so that the problem can be tackled and this will be prevented in the future.

In this document, it will become clear which steps and measures are taken to come to a fully working prototype of a data analyzing tool. The research and development of this system is commissioned by Ampelmann Operations B.V. and the planning is displayed for the whole time span of 17 weeks until 03-06-2016.

At Ampelmann Operations B.V., solving problems is nothing different than described above. Their motion control engineers will debug the systems when they encounter problems and will use their knowledge to solve the problems that occur.

At the moment the debugging can become much better and Ampelmann Operations B.V. gave me the opportunity to do this for their company. The detailed description of the assignment can be found in chapter [3.2](#).

The approach of this project can be found in chapter [5](#). It will become clear very soon, that this project is done in phases, to ensure its structure, quality and safety.

2. Background

2.1 Client and Organization

Ampelmann Operations B.V. is a company which has its headquarters located in Delft. The company was founded in 2008 and has been growing/expanding hard ever since. They've got a lot of offices around the world. Ampelmann has offices in Aberdeen, Delft, Brunei, Houston, Qatar, Rio de Janeiro and Singapore.

Before 2008, it was hard to access offshore buildings like windmills and oil platforms. They would have to use helicopters or boats and rope swing towards the various systems, which cost a lot of effort and time.

The founders of Ampelmann saw their future in finding a solution to this problem. They came up with the Ampelmann system, which is a ship-based, self-stabilizing platform that actively compensates all vessel motions using a Stewart Platform (A platform on six cylinders to ensure six degrees of freedom) to make access to offshore structures safe, easy and fast.

This process is done by continuously measuring the motions of the host vessel. Then, the required lengths of the six cylinders are calculated to keep the transfer deck on which people can cross to the windmills, oil platforms, other offshore buildings or ships completely motionless. Finally, each hydraulic actuator is controlled separately.

2.2 Stakeholders

- Ampelmann: Contractor which will use the tool later on for their company
- Ir. N. van der Geld: First company mentor, will learn certain aspects of mentoring a graduate student, which are convenient for future graduate students.
- Ir. M. Krutzen: Second company mentor, will learn certain aspects of mentoring a graduate student, which are convenient for future graduate students.
- Ms. S.D. de Jong: First Graduate student coordinator, will have benefits from this project, because she will learn certain aspects of mentoring a graduate student, which are convenient for future graduate students.
- Mr. T.J. Koreneef: Second Graduate student coordinator, which will have benefits from this project, because he will learn certain aspects of mentoring a graduate student, which are convenient for future graduate students.
- The Hague University of Applied Sciences: they have the potential to further improve the reputation of the school and potentially increase the number or quality of graduate students.
- Ampelmann Motion Control engineers: They are the ones that will use the tool for their work.

3. Assignment Description

3.1 Problem Definition

At this moment, troubleshooting an the Ampelmann system is done by reading data and using experience. The problems that occur are solved by Ampelmann motion control engineers. The problem is that it is hard to read logged data, as the data log files grow enormously due to the available amount of data.

3.2 Goal:

Ampelmann's solution to this problem is to implement a data analyzing tool, which can translate logged data into an user friendly environment wherein one is able to see what the reason is some errors occur or to adjust the Ampelmann System its settings to optimize its behavior.

The final user of this tool should be able to see data plots of the logged data and other relevant debugging information such as control errors.

Besides checking logged data, the tool should be able to receive queries and respond with a proper response (containing the information asked in the query).

3.3 Thesis question

“ What is the ideal solution for a data analyzing tool, and what is the best way of implementing it on an Ampelmann System.

3.4 Sub Questions

- How does the common Ampelmann hexapod and it's construction work?
- How is data currently logged?
- How can we extract data from logged files in a structured way?
- How can we show other relevant debugging data?
- How is the user able to send queries?

3.5 Requirements

The MoSCoW method is used to describe the main project objectives and requirements and can be used for this project.

MUST

Product requirements:

- There has to be a tool wherein the user is able to read in logged data
- Inside the tool, graphs of logged data should be plotted.
- The final prototype has to be in a 32-bit hierarchy.
- The final prototype has to be a standalone file.
- The tool should make it easier to select parts of data.
- The tool should be able to derive certain key figures for a selected period of time.
- There has to be a working test environment.

SHOULD

Product requirements:

- System should be able to select data that is relevant to the user.
- The system should be written in a language that is known to most of the motion control engineers.
- The system should be “fool-proof”.
- The final prototype should have a catchy name, to ensure it will be remembered and used.
- The system it’s graphical user interface should be easy to understand.

COULD

Product requirements:

- It could be an advantage if the final prototype will not cost a lot / anything.

WOULD

Product requirements:

- It would be great if the final prototype is a simple executable file.

3.6 Scope

To get a good result in a time span of 17 weeks, the tool should at least work for an Ampelmann hexapod’s cylinder data.

The reason an Ampelmann cylinder has been chosen, is because the whole Ampelmann system has a lot of error functions, which is not reachable to implement in the tool in the time span of 17 weeks. The cylinder obviously has less error functions than the total system, but enough to prove that the tool is working.

3.7 Prerequisites

The following prerequisites have to be known in order to succeed the project:

- The thesis duration will be 17 weeks in total.
- The graduation period will be from Monday 08-02-2016 until Friday 03-06-2016.
- During the graduation period, there should be assistance in the form of a company mentor from Ampelmann.
- Ampelmann will have a designated working place available for the student.
- Ampelmann will have a working system available for testing purposes.

3.8 Product

The final product of this project is a working prototype of a data analyzing tool.

The system should work and shall be tested on an existing Ampelmann cylinder and at the end attempts will be made to expand it even further than only the cylinder.

3.9 Costs

Ampelmann will probably not have to buy any materials, as the tool will be a software solution. If the logging cannot be done via a standard Ethernet connection and a connection module has to be bought, this will be specified in the final report.

3.10 Quality

To ensure quality, weekly meetings will take place. In this way all parties are up to date and will know what is on the planning and what has to be done.

3.11 Safety

Ampelmann strives towards safety. They are setting the standard in offshore safety and thus the project must undergo some safety measures.

The standard safety induction video will tell new employees what the safety rules are and Ampelmann has two main sets of rules, of which the golden rules are the safety rules:

Golden Rules:
Always intervene if you observe an unsafe act or condition
Use fall protection when working at height
Always use the appropriate personal protective equipment(PPE)
No working or walking under a suspended load
Follow road safety rules and drive responsibly
Do not carry out a task unless trained and competent
Assess risks and obtain authorization before starting work
Never work under the influence of alcohol or drugs

Table 1 – Golden rules

4. Risk Analysis

This chapter is about the Risk Analysis. In order to avoid an unexpected loss of time, the risks of the project should be determined. This is done by looking at the Risk probabilities, the impact and the mitigating measures that have been taken in order to minimize the given risks.

4.1. External Risk Factors

Risk	Absence of company coach or graduate student coach
Probability	Average, both parties have a busy schedule, so this will happen often.
Impact	Small, as questions can be asked to other colleagues.
Measures	Finding out the planning of both parties, to ensure their availability.

Table 2 – External Risk 1

Risk	Reservation Failure
Probability	Average, as not every Ampelmann system can be worked upon.
Impact	Major, as the testing shall be done on an real Ampelmann system.
Measures	Ensure that reservation status of alternate machinery is known, and that reservations are made quickly so it will not cause problems later in the project.

Table 3 – External Risk 2

4.2. Internal Risk Factors

Risk	Miscommunication
Probability	Small, as there are weekly meetings.
Impact	Small-Major, the amount of impact is determined by the subject where miscommunication is upon.
Measures	Maintain good contact within the group and keep weekly meetings.

Table 4 – Internal Risk 1

Risk	Missing deadlines
Probability	Small, as there will be a reasonable planning made.
Impact	Major, as a deadline always remains a deadline and thus is missing one never wanted.
Measures	The project must recover from any possible delays as quickly as possible and resume work on schedule.

Table 5 – Internal Risk 2

Risk	Program problems
Probability	Small-Average, depending on the type of program language and detailed code, the probability can vary from small to average.
Impact	Average-Major, depending on the type of problem that occurs, this can have major impacts, as the major part of the project is programming.
Measures	Keep meeting with company coach to ensure the correct way of handling is executed.

Table 6 – Internal Risk 3



Risk	Thesis writing problems
Probability	Small, as there is enough time calculated in to write the thesis.
Impact	Average-Major, depending on what the problem is. If the document is lost, this will have a major impact. If there is not enough time as well.
Measures	Write the program on a cloud based storage location, this will tackle the lost document problem and stick to the planning to avoid time issues.

Table 7 – Internal Risk 4

Risk	No testing due to too little time
Probability	Average, as this is the first time of writing a bachelor thesis.
Impact	Major, as it will not proof its conceptual idea.
Measures	Keep updating the planning, so that unexpected loss of time is minimized.

Table 8 – Internal Risk 5

Risk	Ordering parts
Probability	Small, as there probably will not be ordered many parts.
Impact	Average-Major, depending on what the actual delivery time is.
Measures	Research the current system, so that is known if the prototype needs extra parts.

Table 9 – Internal Risk 6

5. Approach

The project can be divided into different phases. Roughly speaking, into five phases.

The first one is the start-up phase, wherein the actual start-up will take place. This includes defining a plan on how to successfully make a thesis.

The second phase is the definition of the project. It will consist out of background research and the documentation of its results. The background research consists of reading available documents to understand the hardware of the current Ampelmann systems (For instance by reading the “Development of the Access System for Offshore Wind Turbines” book) and understanding the Ampelmann system software, by looking into the current software code. Also some hydraulics and electrical courses specifically designed for the Ampelmann systems will be followed.

The third phase will be the design phase. In this phase, the actual research will be translated into a proof of concept. This will include a morphological overview, a Kesselring method to choose the correct solutions based on some given weight factors and working out the concepts.

Then the fourth phase is the testing of the system. The project needs to be realized and tested, to ensure its workability and so that the outcomes of the analyzing are correct, and a well written report has to be created, so that documentation of the tool is always available.

The final phase is the writing of the Thesis and writing missing documentation, to ensure the total system is documented.

First phase (Start-up phase):

5.1. Plan

- Decide the scope/deliverables of the project
- Find out the requirements of the project
- Indexing the available / required resources
- Form a Plan
- Documenting the progress and the results
- Presenting the progress and the plan to the client (Ampelmann)

Second phase (Definition phase):

5.2. Background research

- Research the Ampelmann system
- Research the current data logging
- Research the Ampelmann cylinders

Third phase (Design phase):

5.3. General solution finding

- Brainstorm sessions
- Make a morphological overview
- Use the Kesselring method to determine logical concepts
- Work out concepts

5.4. Specific solution finding

- Choose a concept
- Start with determining how to read in the log files
- Start with programming
- Report the results and progress

5.5. Making tool generally applicable

- Find out what is necessary to make a system generally applicable
- Determine demands of other employees
- Generalize the system

Fourth phase (Testing phase) :

5.6. Testing the tool

- Write a test case and report
- Acquire the needed materials to perform a test
- Acquire the needed competences to perform a test
- Ensure a safe test environment
- Test the made tool on the Ampelmann cylinder
- Document the results.

Fifth phase (Documentation phase) :

5.7. Documentation

- Gather all information that is available
- Write missing information/documentation
- Combine all information
- Revise the report/documentation

5.8. Work packages

Work packages are a great way of representing various tasks and who is executing them in what way.

If we take a look at the Prince 2 method of creating work packages, we can say that most work packages contain:

- The name of the person who will do the work package
- Work package name / title
- Description of work package
- Time
- Agreements
- Deliverables

The work packages for this project can be found in "[Appendix I](#)". The structure of the work packages is shown below in figure 1:

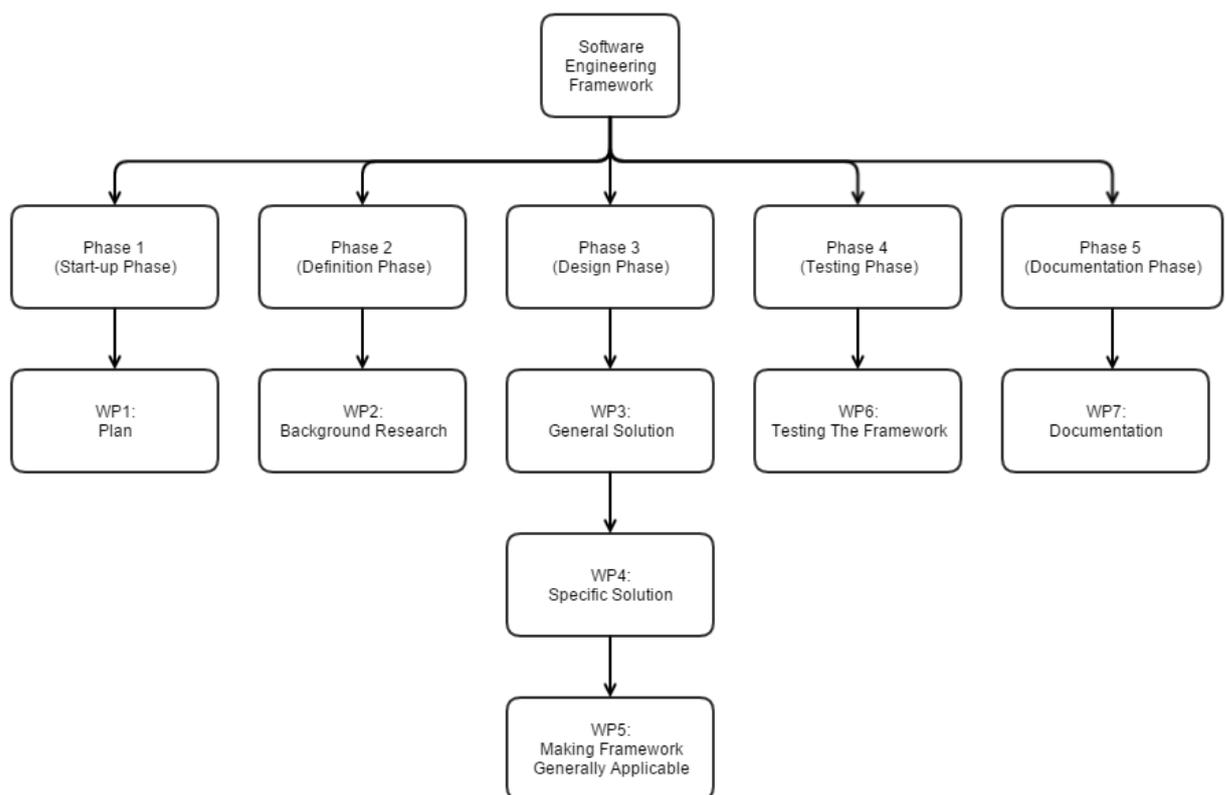


Figure 1 – Structure of work packages

6. Project Organization

In this chapter will be briefly described who the involved parties of the project are.

The current project group consists of a graduate student, two company coaches, two graduate student coordinators and a client.

6.1. Meetings

Because the final result is for a company that relies on the outcome of the research, meetings will take place with the company coaches, to keep both parties updated and sustain a good workflow.

The graduation period lasts 17 weeks. To ensure that everything is going as planned, a meeting will be planned every week. If something goes not as planned, the way of working can be changed easily because of the weekly meetings.

6.2. Contact List

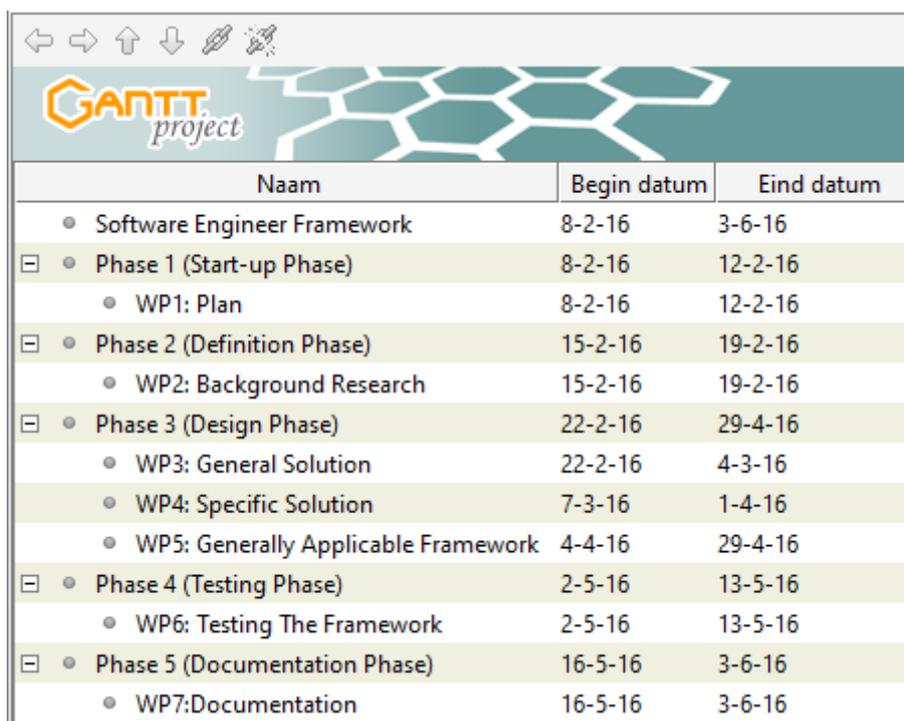
Name:	Role:	E-Mail:
Ampelmann	Client	info@ampelmann.nl
N. van der Geld	Company coach	niels.vandergeld@ampelmann.nl
M. Krutzen	Company coach	martijn.krutzen@ampelmann.nl
S. De Jong	Graduate student coordinator	s.d.dejong@hhs.nl
T.J. Koreneef	Graduate student coordinator	t.j.koreneef@hhs.nl
D. van Tuijl	Graduate student	devinvantuijl@live.nl

Table 10 – Contact list

7. Planning

This chapter contains the planning for the whole time span of 17 weeks, from 08-02-2016 until 03-06-2016. This is to ensure that the project is worked upon in a structured way.

To keep the structure in this project, the phases with its work packages of the “[Work Packages](#)”-chapter are used within the planning. In this way it is clear to see what has been done and what is still left to do.



Naam	Begin datum	Eind datum
• Software Engineer Framework	8-2-16	3-6-16
☐ • Phase 1 (Start-up Phase)	8-2-16	12-2-16
• WP1: Plan	8-2-16	12-2-16
☐ • Phase 2 (Definition Phase)	15-2-16	19-2-16
• WP2: Background Research	15-2-16	19-2-16
☐ • Phase 3 (Design Phase)	22-2-16	29-4-16
• WP3: General Solution	22-2-16	4-3-16
• WP4: Specific Solution	7-3-16	1-4-16
• WP5: Generally Applicable Framework	4-4-16	29-4-16
☐ • Phase 4 (Testing Phase)	2-5-16	13-5-16
• WP6: Testing The Framework	2-5-16	13-5-16
☐ • Phase 5 (Documentation Phase)	16-5-16	3-6-16
• WP7: Documentation	16-5-16	3-6-16

Figure 2 – Planning data

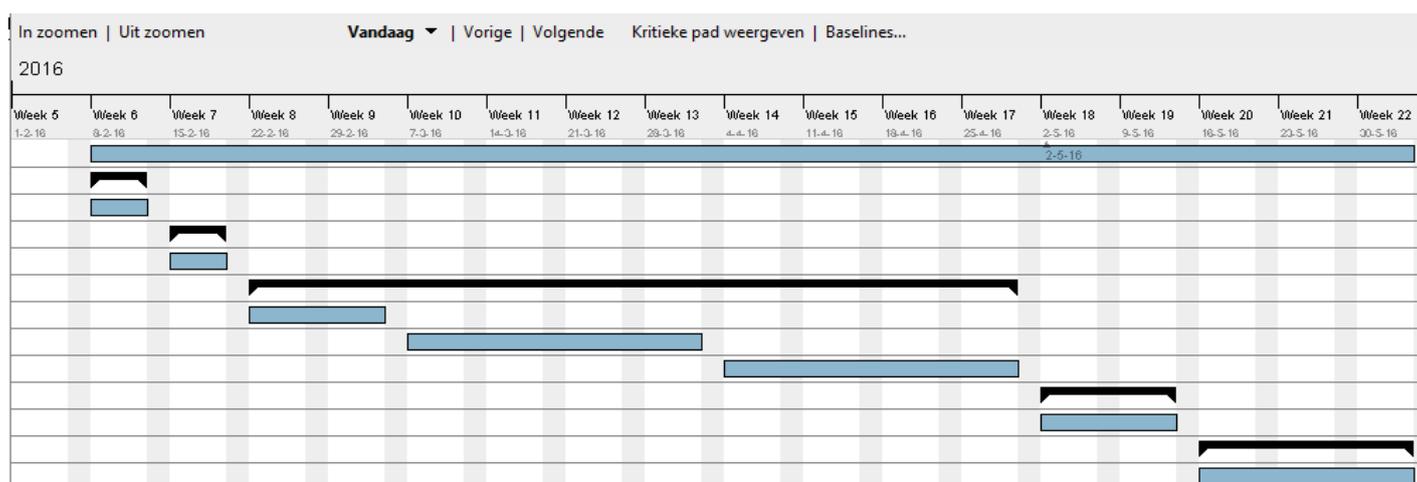


Figure 3 – Planning Overview

Appendix I: Work packages

Phase one – Start-up Phase

Work package 1:	Plan
Executive:	Devin van Tuijll
Description:	The plan will make clear how the approach of the project will be and what the prerequisites, deliverables and requirements are.
Time:	One week
Agreements:	Must contain: <ul style="list-style-type: none"> • Prerequisites • Deliverables • Work Packages • Scope • Goals • Risk analysis • Planning • Requirements
Deliverables:	Plan

Phase two – Definition Phase

Work package 2:	Background Research
Executive:	Devin van Tuijll
Description:	The background research will make sure that there is a proper knowledge achieved on beforehand, so that the total solution can be based upon the whole system eventually.
Time:	One week
Agreements:	<ul style="list-style-type: none"> • Research the Ampelmann system • Research the current data logging • Research the Ampelmann cylinders
Deliverables:	Documentation about research

Phase three – Design Phase

Work package 3:	General Solution
Executive:	Devin van Tuijll
Description:	The general solution is the way to find a general solution to the given problem.
Time:	Two weeks
Agreements:	<ul style="list-style-type: none"> • Do Brainstorm sessions • Make a morphological overview • Use the Kesselring method to determine logical concepts • Work out concepts
Deliverables:	<ul style="list-style-type: none"> • Documentation which concludes the morphological overview and the Kesselring method. • Concepts

Work package 4:	Specific Solution
Executive:	Devin van Tuijll
Description:	Work out a concept, so that there will be a way to start with making the program.
Time:	Four weeks
Agreements:	<ul style="list-style-type: none"> • Choose a concept • Start with determining how to read in the log files • Start with programming • Report the results and progress
Deliverables:	<ul style="list-style-type: none"> • Concept choice • Begin of Tool (Program)

Work package 5:	Making tool generally applicable
Executive:	Devin van Tuijll
Description:	Go further with the tool in such a way that future employees can add the total Ampelmann system to it.
Time:	Four weeks
Agreements:	<ul style="list-style-type: none"> • Find out what is necessary to make a system generally applicable • Determine demands of other employees • Generalize the system
Deliverables:	Generally applicable tool

Phase four – Testing Phase

Work package 6:	Testing the
Executive:	Devin van Tuijll / Testing personnel
Description:	The testing is necessary, so that is known whether the tool is working or not.
Time:	Two weeks
Agreements:	<ul style="list-style-type: none"> • Write a test case and report • Acquire the needed materials to perform a test • Acquire the needed competences to perform a test • Ensure a safe test environment • Test the made tool on the Ampelmann cylinder • Document the results.
Deliverables:	<ul style="list-style-type: none"> • Test Case • Test Results

Phase five – Documentation Phase

Work package 7:	Documentation
Executive:	Devin van Tuijll
Description:	At the end of the graduation period, a complete report has to be handed over to Ampelmann and The Hague University of Applied Sciences. In order to do this, all necessary files to make the tool work need to be documented (if not already documented). This ensures future and current employees to proceed with the tool.
Time:	Three weeks
Agreements:	<ul style="list-style-type: none"> • Gather all information that is available • Write missing information/documentation • Combine all information • Revise the report/documentation
Deliverables:	<ul style="list-style-type: none"> • Complete Report and Documentation • Working Software Engineer Tool

Appendix II – Accountability research for Kesselring method

Accountability research for Kesselring method.

Tool In General

Speed

Java is much faster than Visual Basic and Python. This has been tested by several people on the internet. Some results are shown below:

Test	Program	Seconds	Kilobytes	GigaHertz	CPU	CPU load
regex-dna	Python 3	10.58	266,312	478	23.44	49% 39% 68% 68%
	Java	8.20	749,864	929	24.44	70% 69% 82% 79%
reverse-complement	Python 3	3.11	266,972	800	4.56	20% 99% 2% 29%
	Java	1.17	345,940	1661	2.42	41% 43% 57% 72%
k-nucleotide	Python 3	76.50	162,004	1937	297.12	97% 97% 97% 99%
	Java	6.82	240,412	2568	21.50	77% 71% 77% 92%
binary-trees	Python 3	152.06	804,624	596	516.24	94% 92% 95% 91%
	Java	11.51	622,328	889	40.10	86% 86% 92% 87%

FIGURE 36 - TEST RESULTS SPEED ¹²

In above figure, Python is slower than Java. Visual basic is even faster according to OSnews.com:

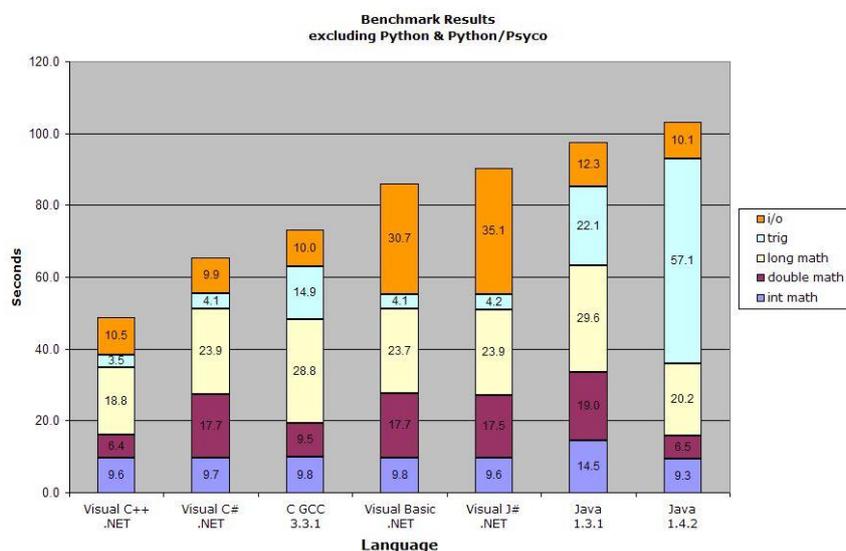


FIGURE 37- VISUAL BASIC VS JAVA ¹³

Hence Visual Basic scores best.

¹² Full test : <https://benchmarksgame.alioth.debian.org/u64q/python.html>

¹³ Full test: http://www.osnews.com/story/5602/Nine_Language_Performance_Round-up_Benchmarking_Math_File_I_O/page3/

Flexibility

Due to the extensibility of python and it's available packages, python scores best. Besides that, Python does not need a lot of set-up in comparison with Java, according to Quora.com¹⁴: "Python requires no "set up." A full python environment is already on every Linux machine, and on Macs. On Linux, the program yum, or the Yellow dog Updater, Modified is written in python, so python is here to stay. Java requires a substantial amount of setup. So if you want to get started with python programming, just type python at the prompt. Now. That's it. To start with Java, call someone who knows it."

Easyness

The easyness of the programming language is based upon the amount of lines it takes to get the same result.

The following picture describes these differences with a small example:

Java	Python	Visual Basic
<pre> if (a > b) { a = b; b = c; } </pre>	<pre> if a > b : a = b b = c </pre>	<pre> if a > b then a = b b = c end if </pre>

FIGURE 38 - EASYNES COMPARISON¹⁵

If the indentation of python is calculated as well to break the if statement, Python and Visual Basic score the same, where Java requires more symbols to get the same result.

Experience

Experience of the languages is based upon the experiences of the writer of this thesis. Before this research, Python has been the mainly used programming language at The Hague University of Applied Sciences. Visual Basic has also been learned, at Ampelmann. With Java, there was no experience at all. The combination of python and visual basic was done earlier before, but on a small scale.

Stability

Due to its structure, visual basic is just a little more stable than python and java. Also, Visual Basic produces an exe file, which can be run whenever wanted, which is a huge advantage compared to Java and Python.

¹⁴ <https://www.quora.com/If-I-had-to-choose-between-learning-Java-and-Python-what-should-I-choose-to-learn-first>

¹⁵ <https://pythonconquerstheuniverse.wordpress.com/2009/10/03/python-java-a-side-by-side-comparison/>

Data Selection

Flexibility

With hard coding what the program must do, the flexibility is not very high, because it can only do that which has been hard coded. With a rule base structure, there is more flexibility, because the program is not completely shut down for other input, but it can change its output depending on the input. Machine learning is more flexible, because it can learn how to behave on a certain way and adapt to certain situations, making it the most flexible. With combining flexible options, the tool tends to get more flexible, hence the four as grade.

Easyness

A rule base structure and a machine learning based structure is more difficult than a hard coded structure and a query structure. Because hard coding is the easiest (No extra code needed to accomplish), it gets the highest grade. A query based structure is somewhat simpler than a machine learning based structure and a rule base structure, hence it gets a three as grade. The combination of more of these options, has been given a three, because hard coding will always be included, compensating for the lower grades of the other options.

Experience

The writer of this thesis has the most experience with hard coding, hence it scores the best. Machine learning has been learned, but the experience is less than hard coding. With queries, only a little experience has been gained in the past. With a Rule Base structure, no experience has been gained. The combination of these options have been applied frequently in the past, giving it a three as grade.

Stability

The stability of a rule based system scores the lowest. According to TeraData¹⁶, the rule based systems tend to have a lower stability in the longer term, in comparison with a machine learning system, due to its complexity how longer the system uses it. To let the rules adapt each time is very time inefficient, and hence this is given a one as grade for stability. Machine learning scores somewhat higher, but lower than a hard coded script, because the adaptation can become unstable due to overfitting etcetera. Queries score the same as hard coding, because they are somehow hard coded as well, and based upon user input. The combination scores the best, because the programmer can use the best of both sides to get the most stable system.

¹⁶ <http://www.forbes.com/sites/teradata/2015/12/15/data-science-machine-learning-vs-rules-based-systems/#21ec7ead5be6>

Modularity

Flexibility

Queries and XML score the same amount on flexibility. This is because they both have pros and cons to support the fact that they have a modular construction. XML is able to give a certain structure to a program, leaving a structure to continue to build on for future programming parts. Queries, if programmed correctly, are able to get the requested data and send it to the user, based upon the input the user gave on beforehand. If it is programmed correctly, the system will always be able to respond in a correct way, no matter what the input is, making it modular. The combination scores higher on flexibility, because this results in a two way modular structure, making it more modular than each option on its own.

Easyness

Queries and XML score the same on easyness. Both take a certain amount of time to implement in the tool, and both are depending on other ways of input, giving it a grade of two. The combination scores even lower, because it takes much more time to implement it.

Experience

The writer of this thesis has had experience with queries for modularity, but less with experience on modularity based on XML, hence XML scores lower. The combination of both has never been experienced, giving it a one as grade.

Stability

Queries tend to be stable, as the programmed structure is able to respond to every request. XML is stable, because it has the same structure over and over. They both score a three on this part. The combination of both can become even more stable, if the structure of XML is used and the reply structure of a query based setup is used, giving the combination grade a four.

Output of Data

Flexibility

A dedicated .Amp file or Ampelmann file, could be in any form, thus it scores a four on flexibility. The CSV file scores a two, because the CSV files are always in the same structure.

Easyness

CSV files are easy to read in, and easy to read with the bare eye when opening with a text editor, resulting in a four for Easyness. The Ampelmann file type is less easy, because the standard format is not yet known.

Experience

The thesis writer has a lot of experience with CSV files, but none with Ampelmann files, leading to a score of four for CSV files and a score of one for Ampelmann files.

Appendix III – Test case results

Test Results

Test 1: Standalone Test

Test:

The standalone test will be done by e-mailing only the executable file and running it. If the executable file runs, the test has passed. If the executable doesn't run because it needs its dependencies, it is not standalone and thus it will fail the test.

The following test has been done on the Moxa (Box-PC), and the result is shown with a photograph. This is to show that it works standalone and on the Moxa.

Test Result:

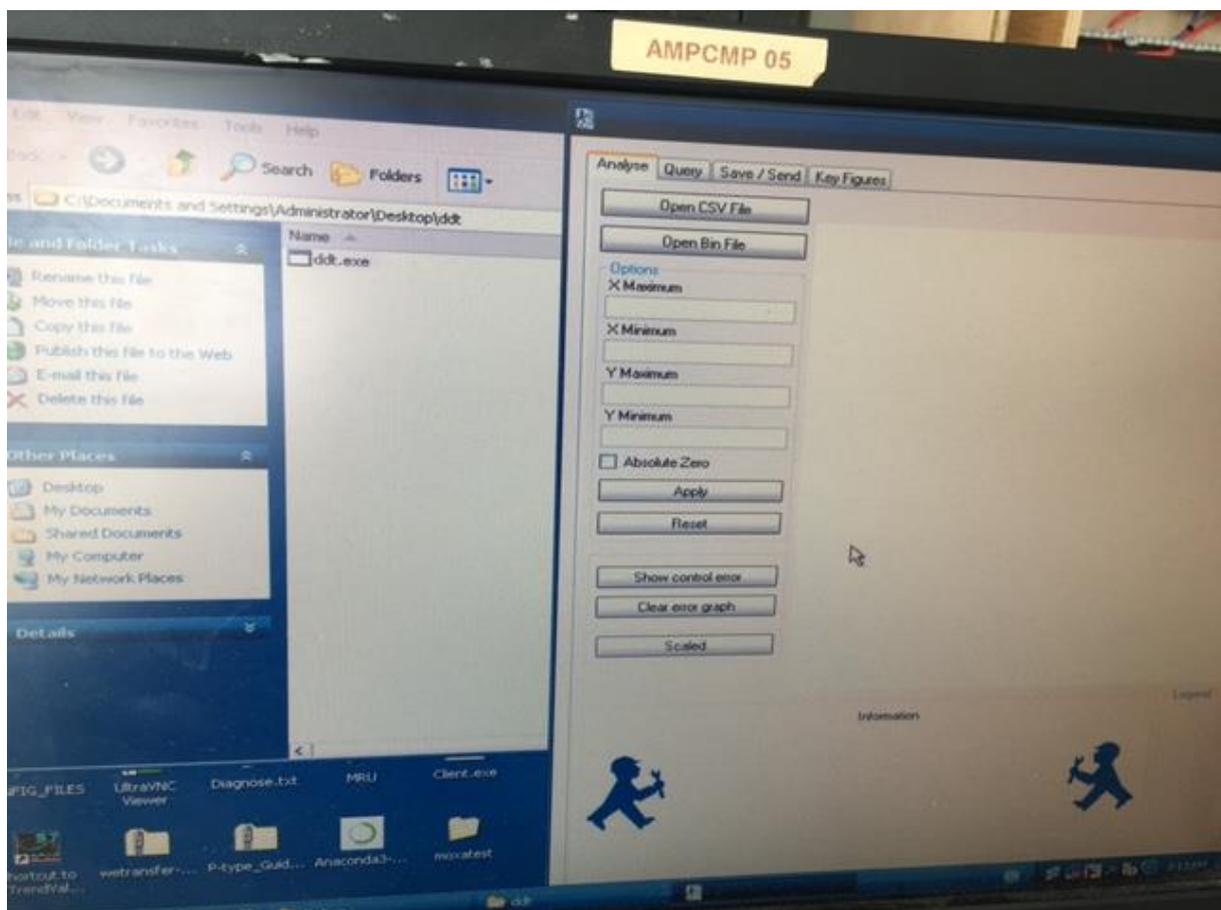


FIGURE 39 - RESULT OF STANDALONE TEST.

In the picture above, it is clear to see that only the ddt.exe file is included and that the tool launches without depending on other files. This means that the test has been passed.

Test 2: 32-bit hierarchy test

Test:

The second test in the series of tests, is the test to validate whether the file works on a 32-bit computer, which was a requirement from the requirements list.

To test if this is valid, the program will be executed on a 32-bit system. Since the Moxa (Box-PC) is a 32-bit system, the test is passed if the program will run. To prove that the system is operating in a 32-bit hierarchy, the system settings will be shown in the result as well. On windows XP, a system is 32 bit when there is no 64-bit information under the header “system”.

Test Result:

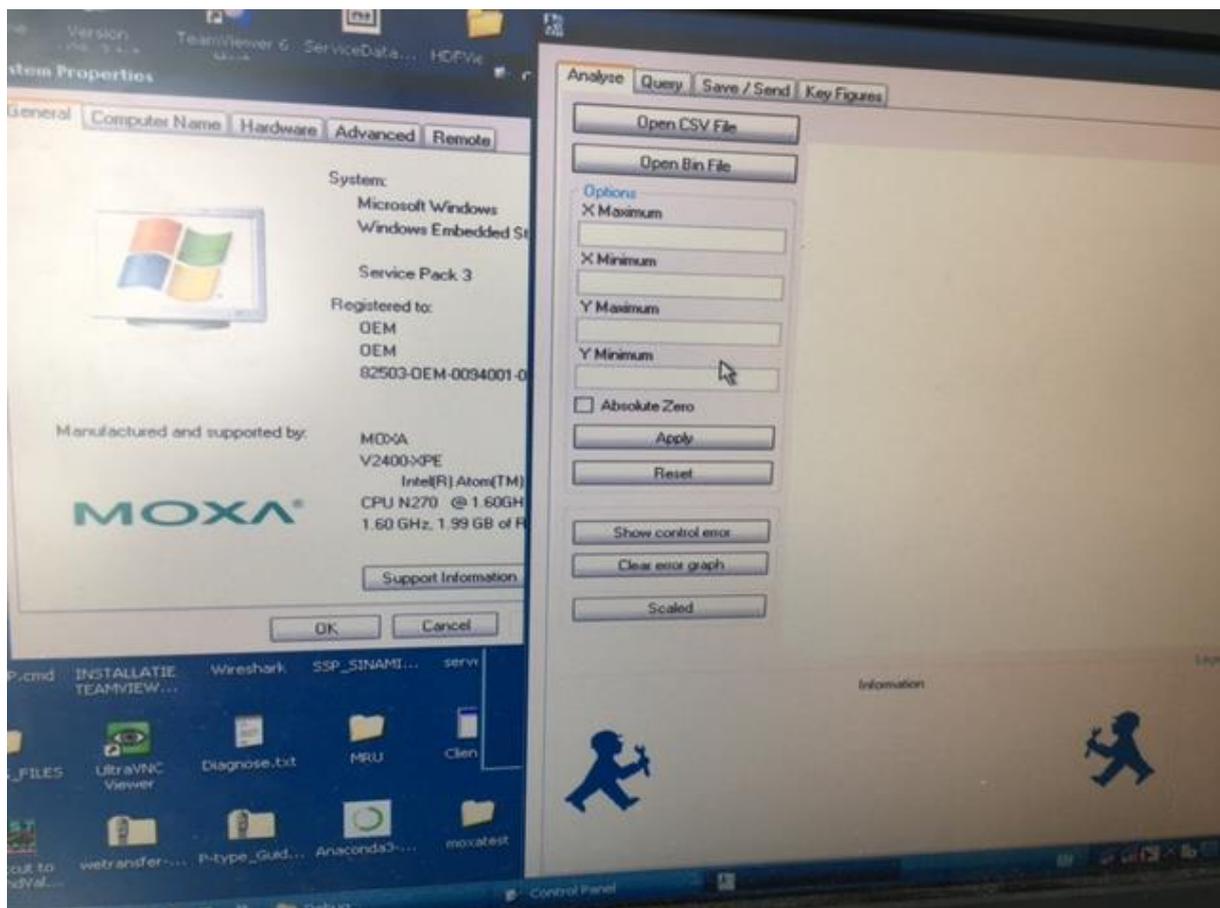


FIGURE 40 - 32 BIT HIERARCHY TEST RESULT

In the picture above it is clear to see that the system settings do not display information about a 64-bit, hence the structure is 32-bit. The program itself runs smoothly without any incompatibility problems occurring. This means that the test has been passed.

Test 3: Read in CSV files test

Test:

In this test, the function of reading in CSV files will be tested. The test will be done by running the tool and loading different CSV files. If the files are loaded correctly, the data will be shown in the treeview on the right, which result in a pass for this test. If not, the test has failed.

Test Result:

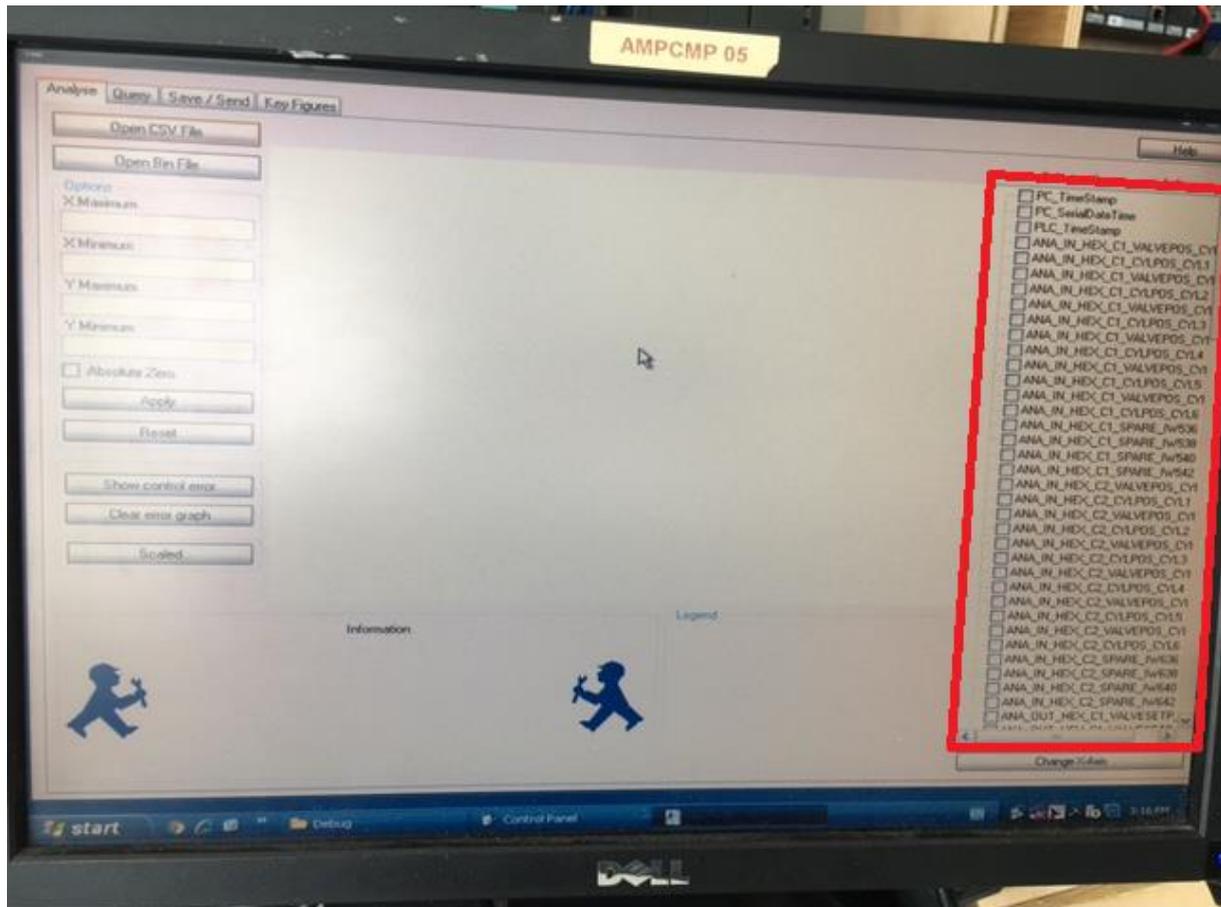


FIGURE 41 - READING IN CSV FILES TEST RESULT

In the picture above, the red box displays the treeview with the loaded data of a CSV file. This means that the CSV files have been loaded correctly and thus the test has been passed.

Test 4: Export data test

Test:

In this test, the functionality of exporting data will be tested. The test will be done by reading in a CSV file and then plotting a graph, adding some key figures and then saving or sending the files.

Test Result:

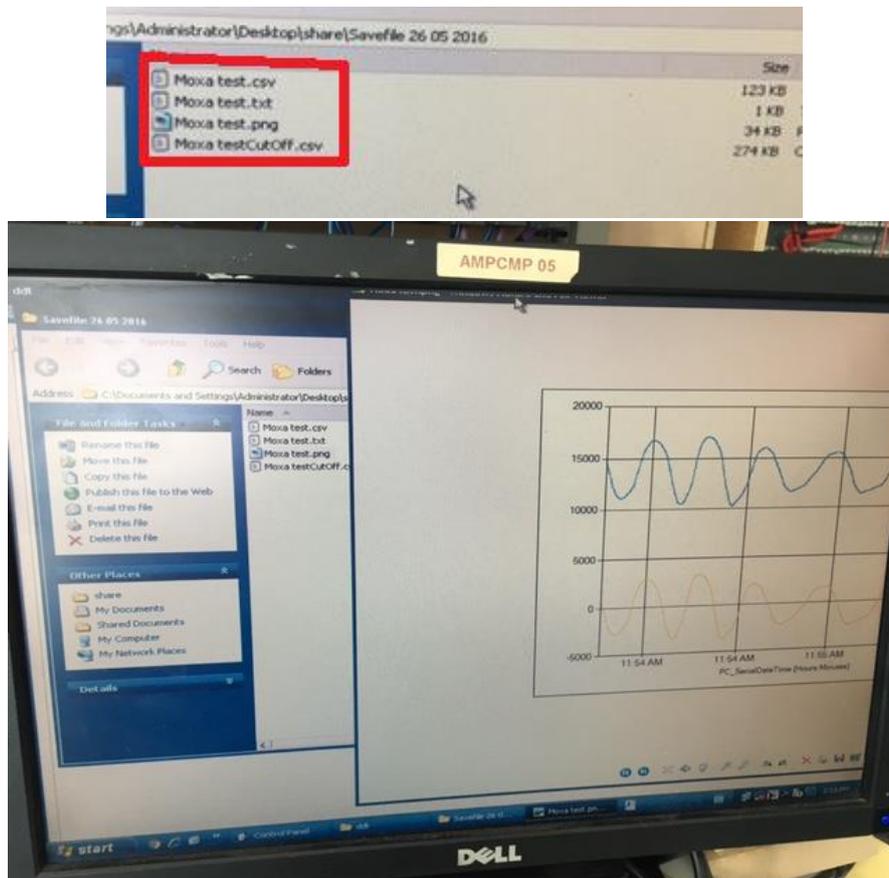


FIGURE 42 – SAVE FUNCTION TEST RESULT (TOP: FOLDER WITH RESULTS. BOTTOM: ONE OF THE RESULTS OPENED.)

In the picture above, the save function is shown and it has passed the test.

The sending of files is not yet programmed and thus this part fails the test, resulting in a pass and a fail as end result.

Test 5: Deriving key figures test

Test:

This test will determine whether the key figures function is functioning properly. The test is done by loading in a CSV file and requesting key figures. If the key figures seem logical, the test is passed. If not the test has failed.

Test Result:

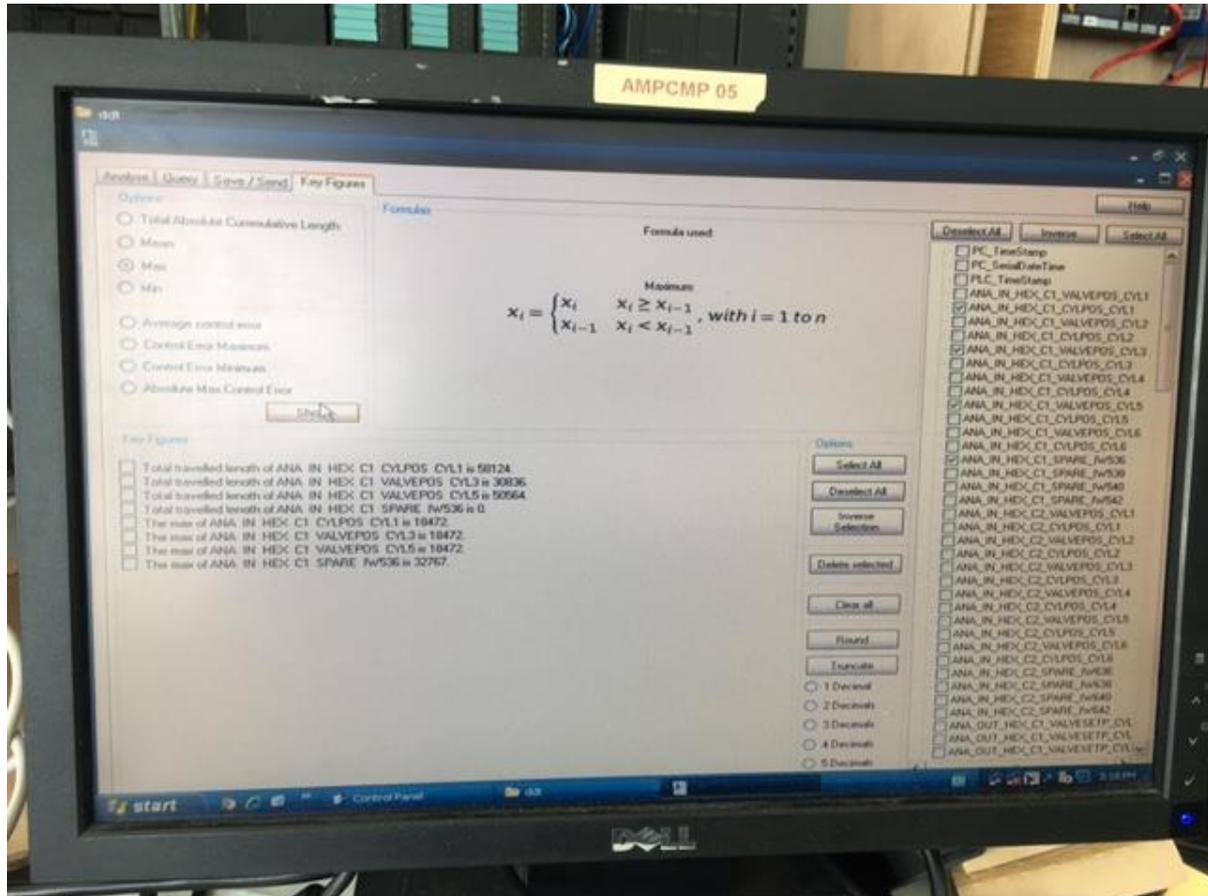


FIGURE 43 - DERIVE KEY FIGURE TEST

The output of the tool are raw values. If we look at these values and compare them to the CSV file below, we can see that the calculation is done correctly. In the CSV file on the next page, the total travelled length for ANA_IN_HEX_C1 is calculated manually. This is done by taking the absolute difference between two data values and then adding the answers to get the total travelled length.

<i>fx</i> =SOM(E10215:E20426)	
D	E
	4
	4
	4
	8
	4
	4
	8
	4
	4
	4
	4
	4
	4
	4
	4
	8
	58124

FIGURE 44 - CSV RESULT OF MANUAL CALCULATION

In the figure above, the calculations are done manually. The manual calculation shows the same result, confirming that the derivation of key figures is working properly.

Test 6: Fool Proof test

Test:

This test will determine whether the DDT is fool proof. The test will be done by letting inexperienced people use the tool. If the tool doesn't crash, it will pass the test. If it crashes or acts with unexpected behavior, the test has failed.

Test Result:

After several people had used the tool, an error was detected. Adding graphs via the treeview in the analyse/offline tab only works when they are checked from up to down. Otherwise it will try to add the same last checked graph again.

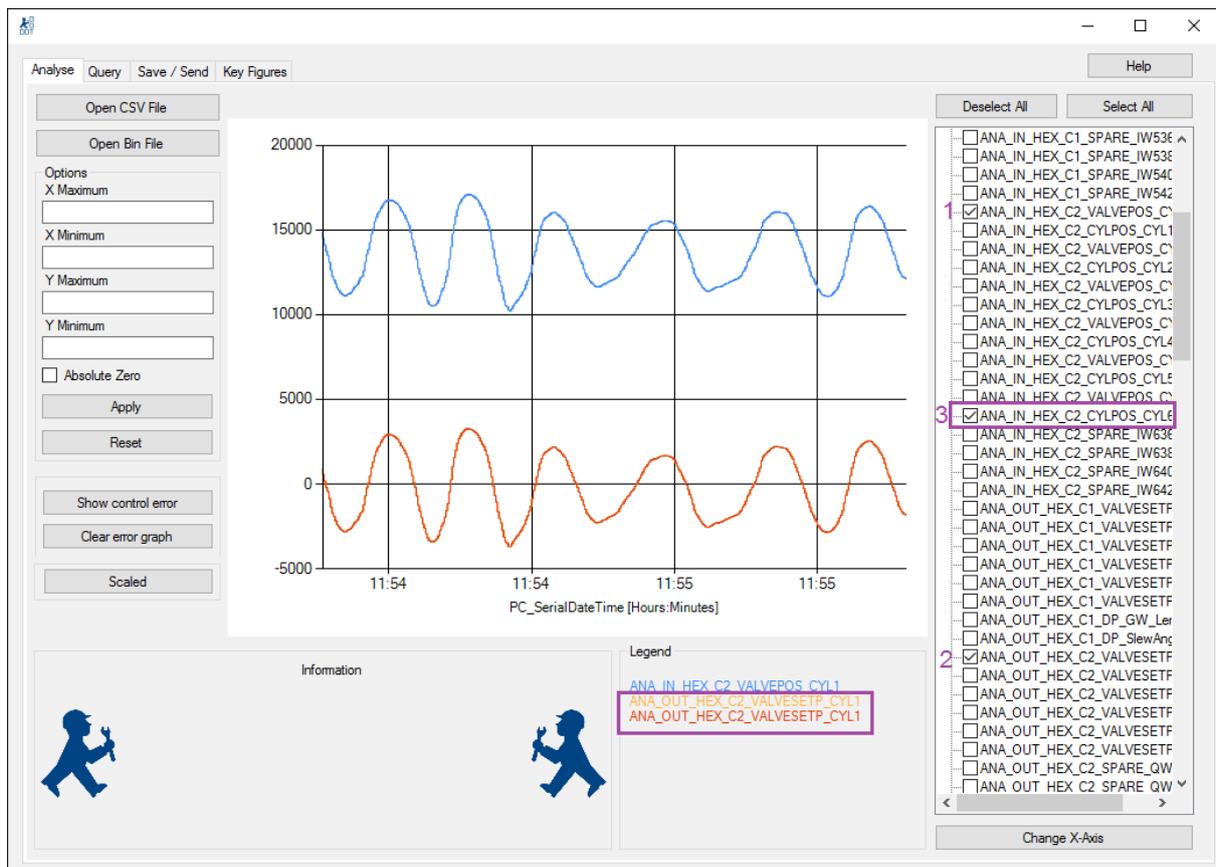


FIGURE 45 - FOOL PROOF TEST FAILURE

The error occurs when a node higher in the treeview is checked later than one beneath it. For instance in the picture above, the node labeled with 3 is checked later than the node labeled with 2, resulting in a graph that is equal to the lowest checked one (See the legend).

This error is probably produced by the loop that checks whether a node has been checked or not. So a recommendation is to check that control loop.

This test has thus failed.

Test 7: Easy Understandable GUI test

Test:

This test will determine whether the DDT GUI is easy to understand or not. The test will be done by letting several people look at the GUI and give their opinion about the GUI. It is done at the presentation and demonstration for almost the whole motion control engineer group.

Test Result:

The motion control engineers found the tool promising. It looked like it worked fine, despite some minor bugs. They had several ideas for functions that are not yet programmed inside the tool. These functions are mentioned in the recommendation area.

The GUI has passed the test.

Test 8: Graph plot test

Test:

This test is meant to determine whether the tool is able to plot graphs. The test will be done by plotting multiple graphs and checking its functionality. The graphs will pass the test once the DDT can plot multiple graphs in different colors, with the corresponding titles in the legend. It will fail if the graphs do not plot correctly.

Test Result:

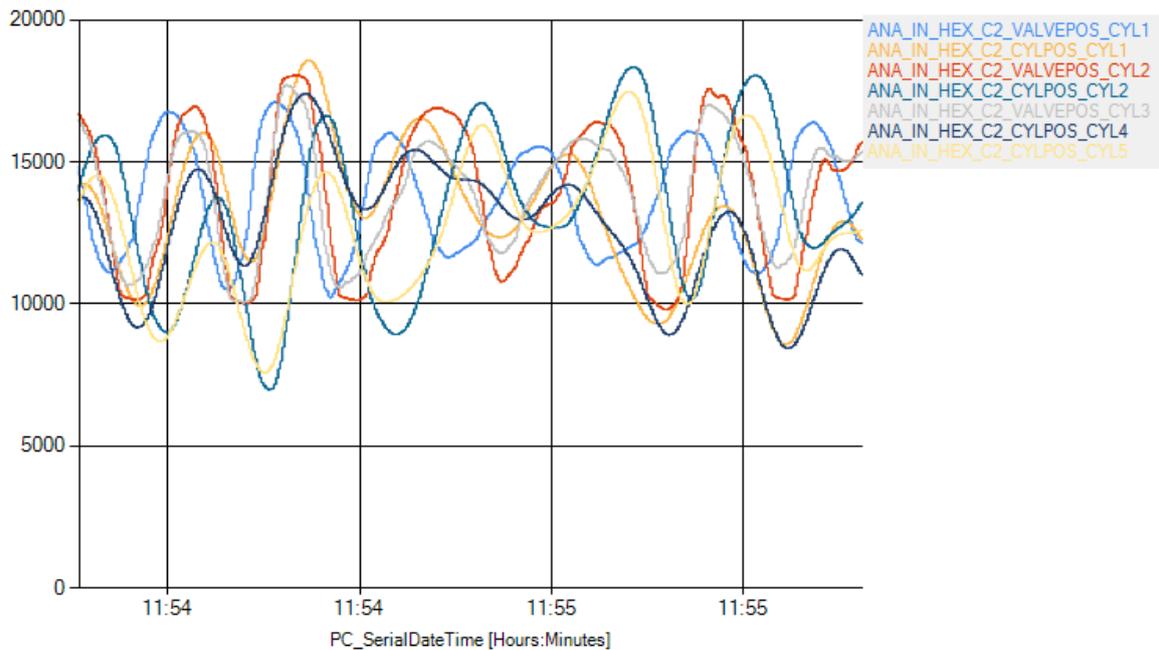


FIGURE 46 - RESULT OF GRAPH PLOTTING

In above picture, it is clear to see that multiple graphs can be plotted without flaws. This means that the graph plotting has passed the test.

Appendix IV – Manual for Tool

The Ampelmann DDT Manual



Manual for The Ampelmann DDT, a data analysing tool.

By Devin van Tuijl
June 3rd, 2016

Content

Introduction	102
Tabs of the Tool	102
Analyse	102
CSV	102
Bin	107
Key Figures	108
Save & Send	111
Save	111
Query	112
Troubleshooting.....	113

Introduction

This manual is meant for Ampelmann employees who want to use the DDT.

The DDT is designed for MCM employees to help them in the debugging process of the Ampelmann systems.

The manual has been written by Devin van Tuijll. In case that there are any questions, please feel free to contact him.

Tabs of the Tool

The main functionality of the tool has been broken down into four main pieces: Analyse, Query, Save/Send and Key Figures. The manual will elaborate per chapter how to use the tool. You can switch between tabs by simply clicking them.



FIGURE 47 - TABS OF DDT

Analyse

The analyse tab is the place that first comes up when the tool is started. Here, the user is able to load in CSV files or BIN files. This can be done by their corresponding buttons, shown in the figure below:

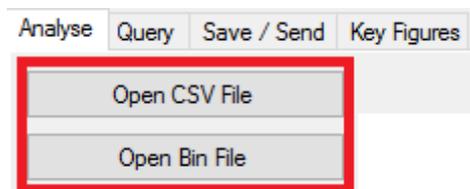


FIGURE 48 - OPEN CSV (TOP BUTTON) AND OPEN BIN (LOWER BUTTON)

CSV

When the “Open CSV File” button is pressed, the program will give a pop-up message, containing a file locator for the desired CSV file, as shown below:

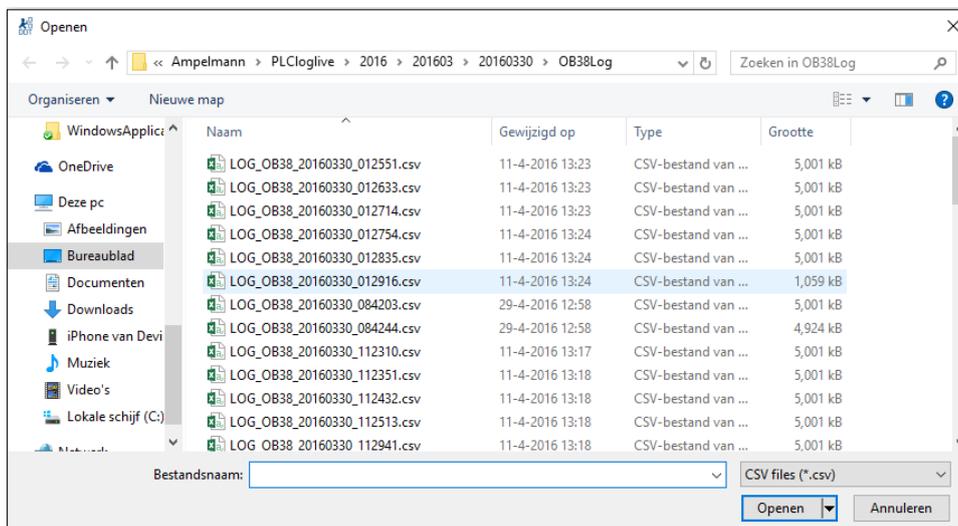


FIGURE 49 - CSV FILE LOCATOR

Once the file locator has been opened, the user can either select a CSV file, or select none and close the CSV file locator. If closed, nothing happens. If a CSV file is opened, the tool tries to allocate the data of the CSV files into arrays. During this process, a loading bar will occur and will tell how far the process is. If this succeeds, the data titles of the CSV file will be shown in the treeview to the right, as shown below:

- PC_SerialDateTime
- PLC_TimeStamp
- ANA_IN_HEX_C1_VALVEPOS_CYL1
- ANA_IN_HEX_C1_CYLPOS_CYL1
- ANA_IN_HEX_C1_VALVEPOS_CYL2
- ANA_IN_HEX_C1_CYLPOS_CYL2
- ANA_IN_HEX_C1_VALVEPOS_CYL3
- ANA_IN_HEX_C1_CYLPOS_CYL3
- ANA_IN_HEX_C1_VALVEPOS_CYL4
- ANA_IN_HEX_C1_CYLPOS_CYL4
- ANA_IN_HEX_C1_VALVEPOS_CYL5
- ANA_IN_HEX_C1_CYLPOS_CYL5
- ANA_IN_HEX_C1_VALVEPOS_CYL6
- ANA_IN_HEX_C1_CYLPOS_CYL6
- ANA_IN_HEX_C1_SPARE_IW536
- ANA_IN_HEX_C1_SPARE_IW538
- ANA_IN_HEX_C1_SPARE_IW540

FIGURE 50 - TREEVIEW AFTER LOADING CSV

This treeview is used for the X axis, which is displayed by the message above the treeview. Here the X axis for graph plots can be selected.

If the X axis is selected, on the same place as the x axis treeview, the Y axis treeview will occur. On top of this treeview, a select all and deselect all button will occur as shown in the following figure:

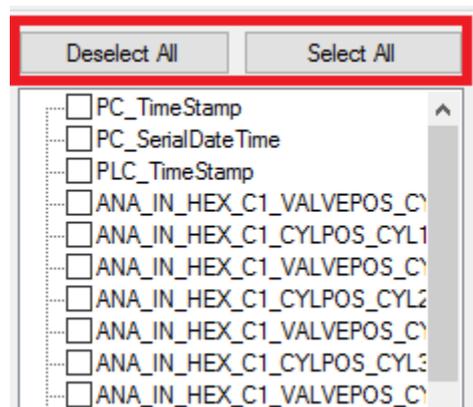


FIGURE 51 - SELECT ALL AND DESELECT ALL

With these buttons, the user is able to select or deselect the items in the treeview.

Once a node is clicked in the treeview, the user will see that the selected item will be plotted against the earlier selected X axis, as seen in the figure on the next page.

The legend of graph is displayed just right below it, marked with the title: "Legend".



FIGURE 52 - PLOTTED GRAPH AND LEGEND

From now on, it is possible to plot any graph against the selected X axis.

To zoom in, the user is able to select either a region with the mouse, or type their range in manually, by using the options menu in the figure below:

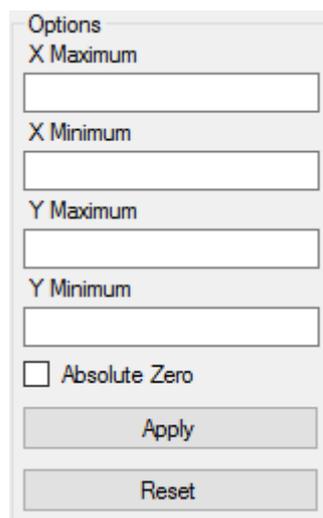


FIGURE 53 - OPTIONS MENU

Within the options menu, the user is able to give values for: Maximum X, Maximum Y, Minimum X and Minimum Y. Once satisfied with the values, the user can press the apply button to apply the settings. If desired, the user can reset the settings to its previous values, by pressing the Reset button. There is also an absolute zero checkbox, which allows the user to resample the time to start

from zero, which can be convenient when using the PLC timestamp since the PLC timestamp is the time since it was turned on.

To display the difference between two graphs, there is an error function implemented, which is accessible by pressing the Show Control Error button. The title aims at the control error, which is for instance the difference between the setpoint position and the actual position of a valve. To delete the error graph again, the user can press the “clear error graph” button. The “Show control error” button only works if there are only two nodes selected in the treeview to the right.

The buttons are shown in the figure below:

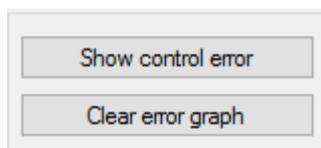


FIGURE 54 - ERROR GRAPH BUTTON

An example of an error function could be:

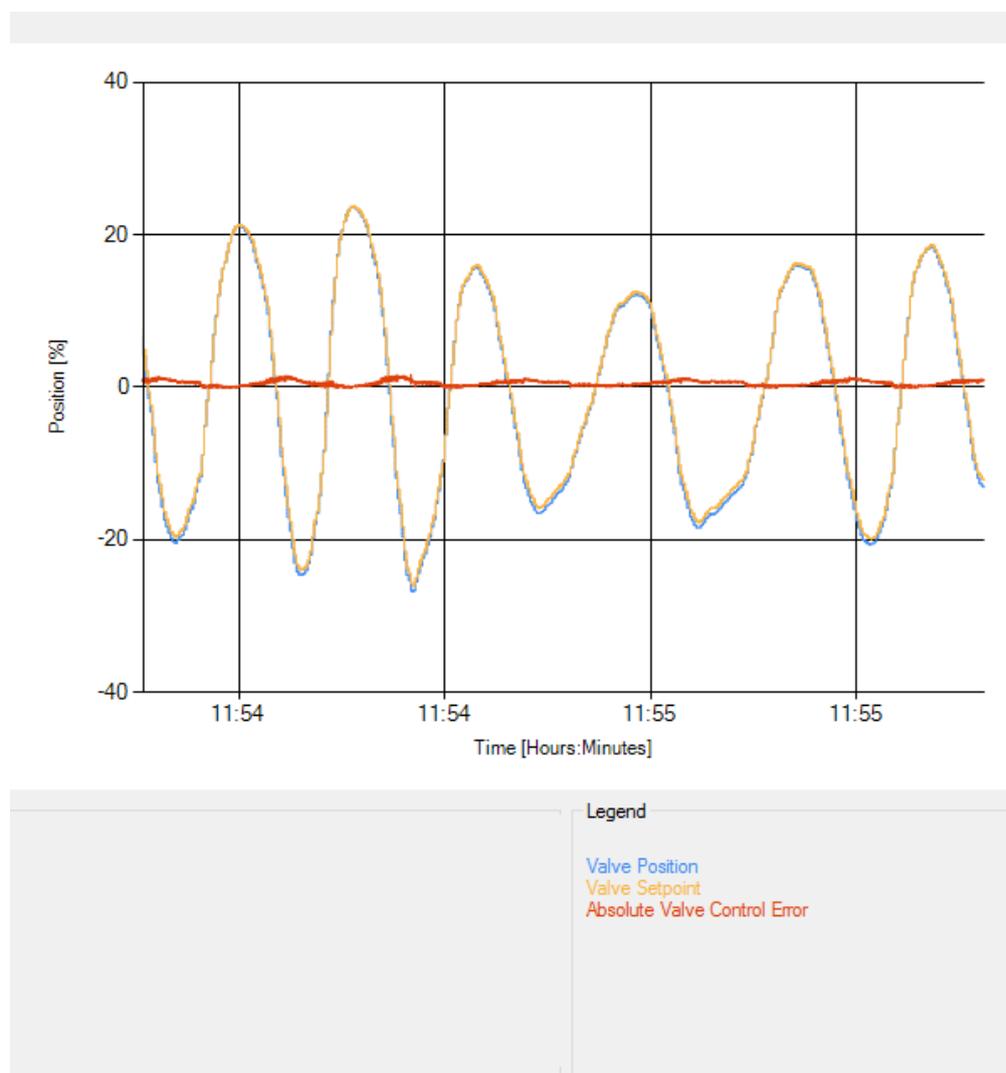


FIGURE 55 - VALVE CONTROL ERROR

In the figure above, the absolute control error for the valves is given. This is an example of what might be possible in the future with the tool. For now, the values are not scaled yet, so with raw values we get the following result:

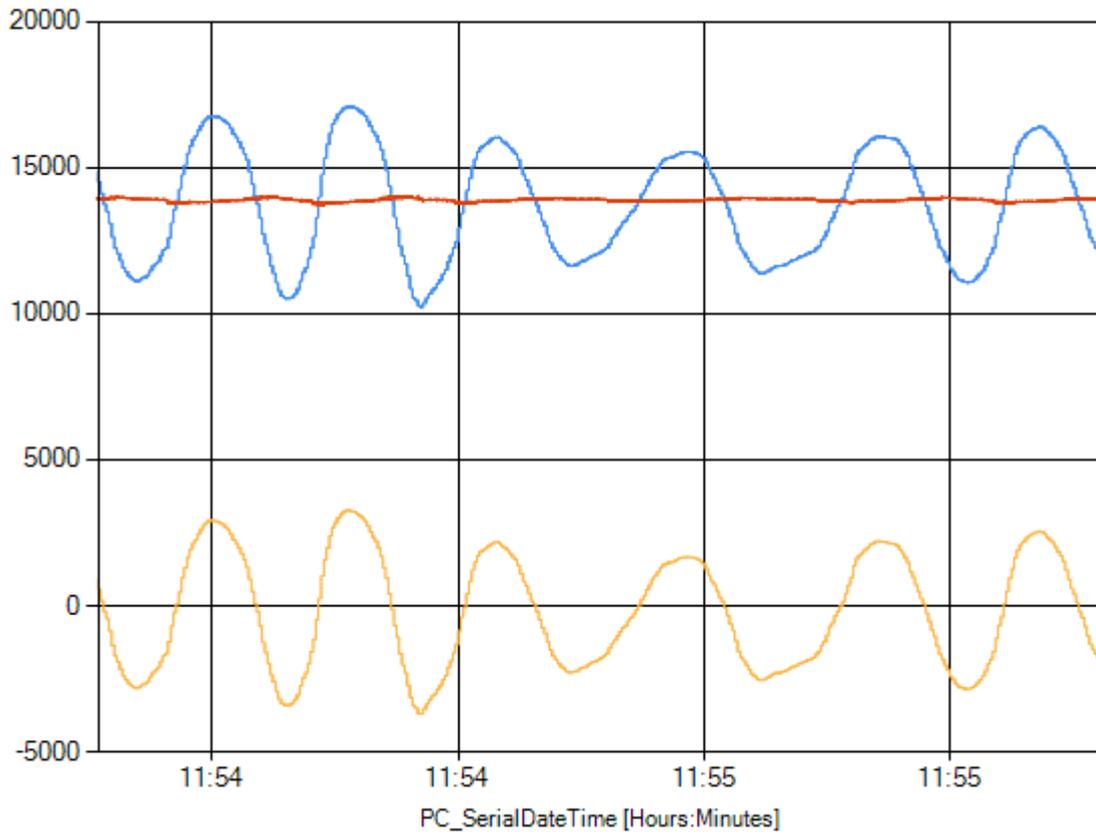


FIGURE 56 - ERRORGRAPH WITHOUT SCALING

Bin

Bin files can be opened as well. This can be done by pressing the “Open Bin File” button, as described in Figure 47.

This will then open up the PLC log converter tool made by J. de Vriend. This will open up the tool, which looks like this:

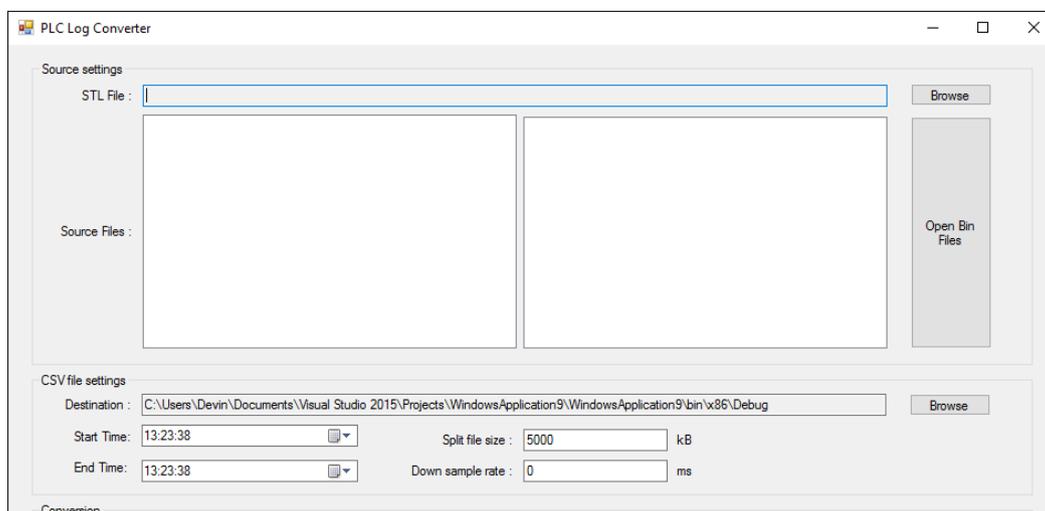


FIGURE 57 - BIN LOG CONVERTER

If you then press the browse button on the upper right side, you can select the corresponding STL file. Then the bin files that correspond to the same type as the STL file can be loaded in. For instance: OB38.stl can be loaded in as an STL file, and every OB38 bin file can then be loaded in as well. This is shown in the figure below:

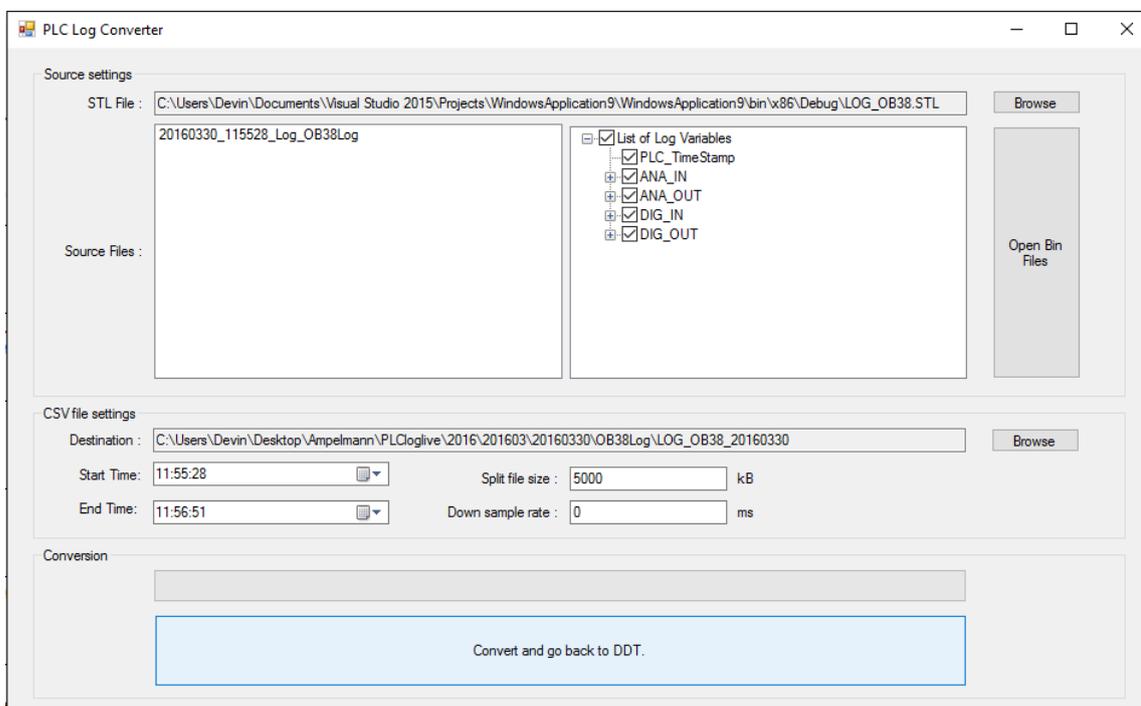


FIGURE 58 - OB38 EXAMPLE

If the “Convert” button is clicked, the tool converts the Bin file to a CSV file, which is automatically opened by the DDT.

Key Figures

In the “Key Figures” tab, the user is able to derive key figures when a CSV file is loaded in the “Analyse” tab.

On the right side, an exact copy of the treeview from the “Analyse” tab will occur. These nodes can be selected to apply key figures upon. On top of the treeview, there are three buttons. A “Deselect All”, an “Inverse” and a “Select All” button.

An example is shown below:

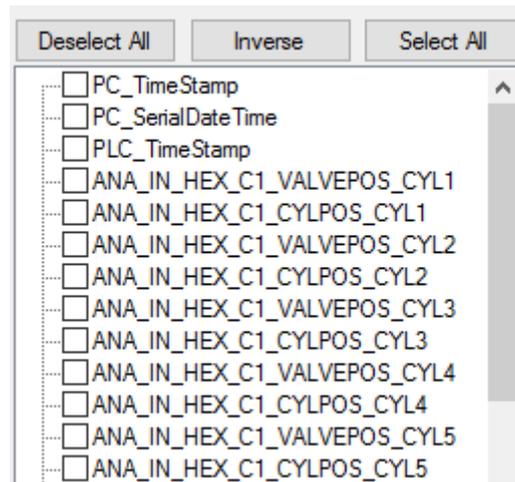


FIGURE 59 - TREEVIEW WITH DESELECT ALL, INVERSE AND SELECT BUTTONS

If one or more nodes are selected, the user can select which formula to apply. This is possible in the “Options” menu, as shown below:

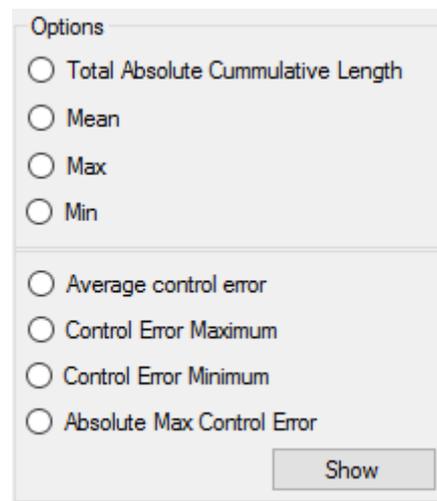
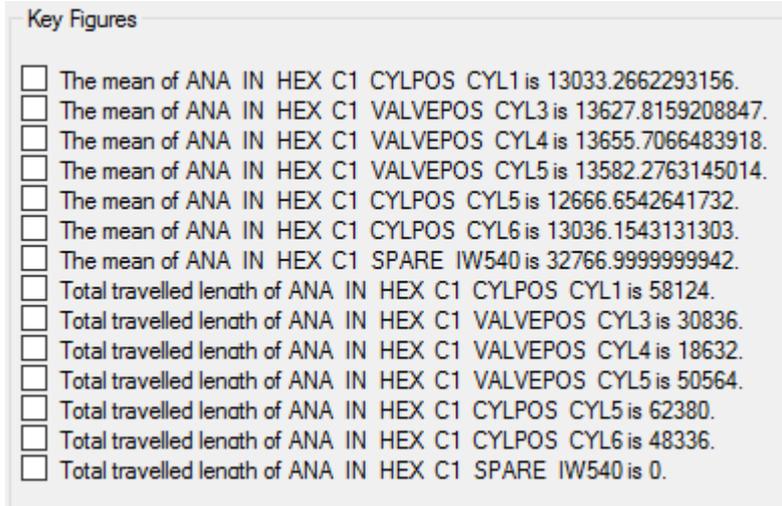


FIGURE 60 - OPTIONS MENU

One of these functions can be selected, which are applied to the selected nodes once the “Show” button is clicked. The formulas in the upper part can be applied to one or more nodes, whilst the lower half can only be applied when two nodes are selected.

The result of the key figure calculations are shown in the Key Figure group box on the lower part of the tab, as seen in the figure below:

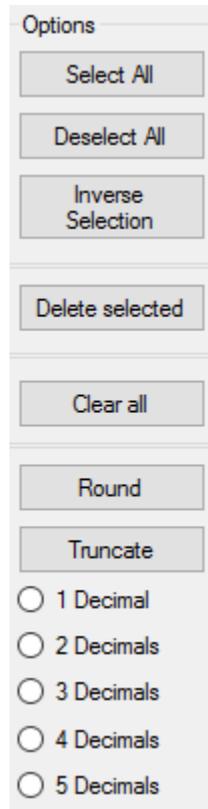


The screenshot shows a window titled "Key Figures" containing a list of 14 key figures, each with an unchecked checkbox to its left. The key figures are:

- The mean of ANA IN HEX C1 CYLPOS CYL1 is 13033.2662293156.
- The mean of ANA IN HEX C1 VALVEPOS CYL3 is 13627.8159208847.
- The mean of ANA IN HEX C1 VALVEPOS CYL4 is 13655.7066483918.
- The mean of ANA IN HEX C1 VALVEPOS CYL5 is 13582.2763145014.
- The mean of ANA IN HEX C1 CYLPOS CYL5 is 12666.6542641732.
- The mean of ANA IN HEX C1 CYLPOS CYL6 is 13036.1543131303.
- The mean of ANA IN HEX C1 SPARE IW540 is 32766.9999999942.
- Total travelled length of ANA IN HEX C1 CYLPOS CYL1 is 58124.
- Total travelled length of ANA IN HEX C1 VALVEPOS CYL3 is 30836.
- Total travelled length of ANA IN HEX C1 VALVEPOS CYL4 is 18632.
- Total travelled length of ANA IN HEX C1 VALVEPOS CYL5 is 50564.
- Total travelled length of ANA IN HEX C1 CYLPOS CYL5 is 62380.
- Total travelled length of ANA IN HEX C1 CYLPOS CYL6 is 48336.
- Total travelled length of ANA IN HEX C1 SPARE IW540 is 0.

FIGURE 61 - KEY FIGURE GROUPBOX

Every created key figure can be selected by their respective checkboxes. Once one or more checkboxes are checked, it is possible to apply certain commands to them. These commands or functions are shown in the other "Options" group box, located on the right side of the key figures. This options menu is shown in the figure below:



The screenshot shows a vertical "Options" menu with the following controls:

- Select All
- Deselect All
- Inverse Selection
- Delete selected
- Clear all
- Round
- Truncate
- 1 Decimal
- 2 Decimals
- 3 Decimals
- 4 Decimals
- 5 Decimals

FIGURE 62 - OPTIONS MENU FOR KEY FIGURES

The first three are “Select all”, “Deselect All” and “Inverse Selection”, which selects, deselects or inverse selects the current nodes that are checked.

Deleting a node can be done by clicking the “Delete Selected” button which is the button below the “Inverse Selection” button. This will then delete the selected node(s).

The “Clear all” button will clear the whole Key Figures group box.

The last two buttons are the “Round” and “Truncate” buttons, which can round or truncate a keyfigure, depending on which button is pushed. The user can determine the amount of decimals to truncate or round off to and on default it is set to two. An example of the truncate function to three decimals is shown below, where the upper is before and the lower after the truncation.

The mean of ANA IN HEX C1 CYLPOS CYL1 is 13033.2662293156.
 The mean of ANA IN HEX C1 CYLPOS CYL1 is 13033.266

FIGURE 63 - TRUNCATION EXAMPLE

And an example for the rounding function has been given, with a rounding to two decimals:

The mean of ANA IN HEX C1 VALVEPOS CYL5 is 13582.2763145014.
 The mean of ANA IN HEX C1 VALVEPOS CYL5 is 13582.28

FIGURE 64 - ROUNDING EXAMPLE

Save & Send

In the “Save and Send” tab, the user is able to export selected nodes, key figures, graphs and cut off function plots.

The export is in either the way of saving to a local folder, or to send them via email.

Save

In the figure below, the “Save” group box is shown. In this box, the user is able to select which features to export and to give a file name.

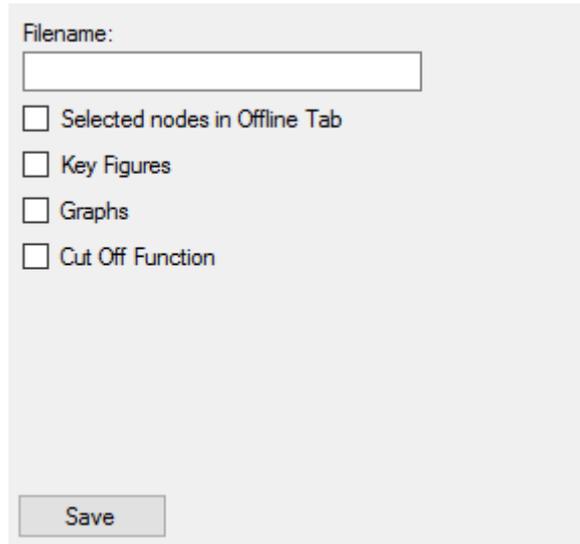


FIGURE 65 - SAVE GROUPBOX

In the file name textbox, the user gives a certain name which will be used for saving the file. The file will be saved locally and the corresponding folder will be opened up when the user presses the save button and the saving has succeeded.

If “Selected nodes in Offline Tab” is selected, the user can save the selected nodes in a new CSV file. The “Offline” tab is the same as the “Analyse” tab.

If “Key Figures” is selected, the user is able to save the key figures that are displayed in the “Key Figures” tab.

If “Graphs” is selected, the tool will save the plotted graphs from the “Analyse” tab.

If “Cut Off Function” is selected, the user is able to select a certain value for which the plots in the “Analyse” Tab should be cut off to. The user can either select a maximum or a minimum.

If the “Save” button is clicked, a dedicated folder will be created, with the current date as name. This has been done to ensure that there is always a good structure in saving and managing files.

If the Cut Off function is used, a CSV file will be created, which can be opened up again in the tool, so that the cut off can be shown visually with graphs. In the figure below, an example of a dedicated folder is shown:

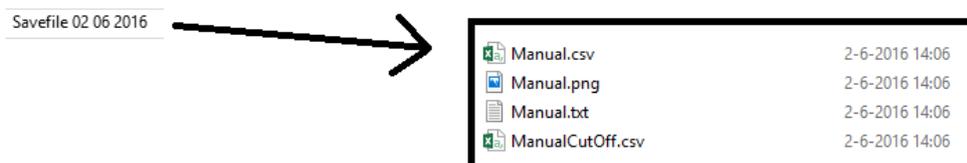


FIGURE 66 - DEDICATED FOLDER EXAMPLE

Query

In the “Query” tab, a template has been made for sending queries and requesting data from the same tool.

In the figure below, the connection group box is shown:

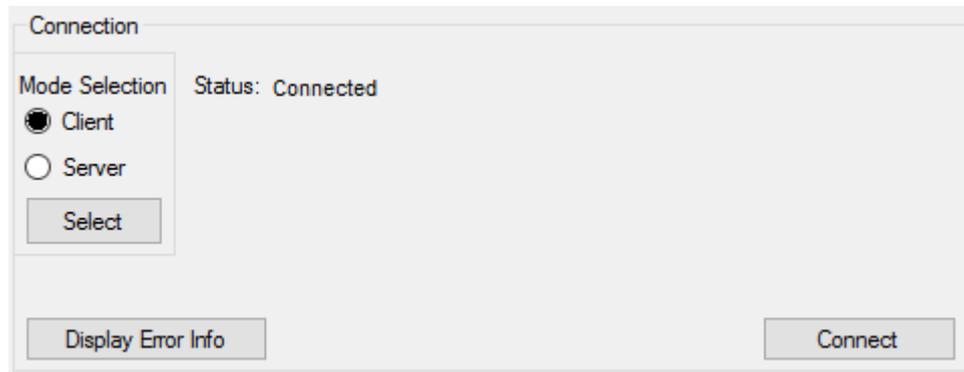


FIGURE 67 - QUERY MODE CONNECTION GROUP BOX

The user is able to select if either the client or the server side is wanted, by selecting their corresponding radio buttons. Once selected, the tool can behave in either a server way or a client way, guaranteeing that only the DDT is necessary to make a connection between both. Because it is a template, the current version only uses the local host. In the future, multiple IP addresses can be given to the program to let it connect to.

The “Display Error Info” button will let the user know what the error is in case that the “Status” label shows that there is an error.

If the connect button is clicked in server mode, the server will start its loop for finding connecting clients. If the connect button is clicked in client mode, the client will try to find a host.

Troubleshooting

In case of trouble, there is a help button located inside of the tool on the right upper side, which is accessible from every tab.

The help button is shown below:

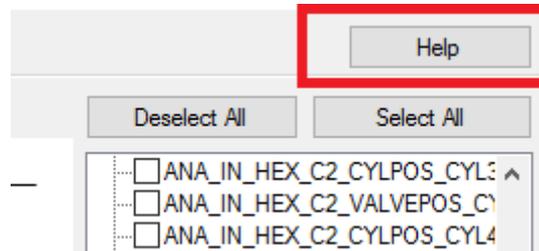


FIGURE 68- HELP BUTTON

Once pressed, the help menu will pop-up, containing likewise info from the manual, displayed per tab. It also contains a F.A.Q. tab which contains the frequently asked questions. This will help finding out errors of the tool.

The help menu is shown in the figure below:

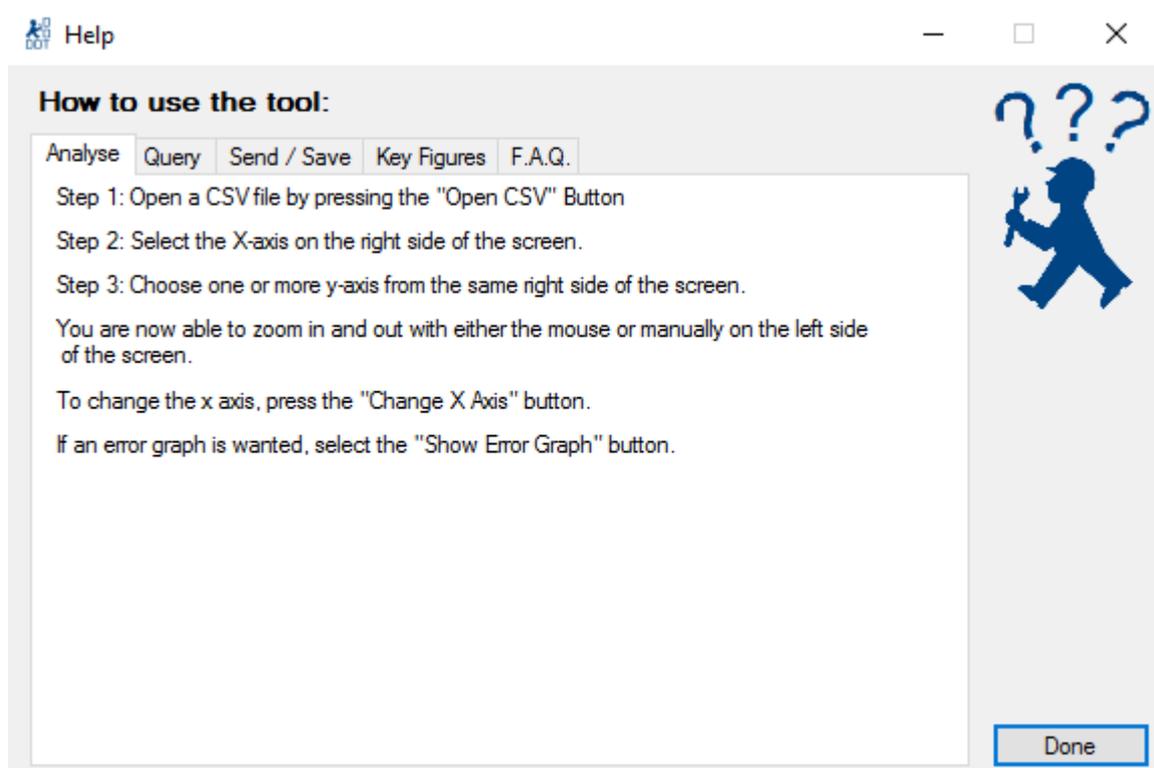


FIGURE 69 - HELP MENU

Appendix V – Program

```

Imports System.Net
Imports System.Net.Sockets
Imports System.Text
Imports System.Threading
Imports System.Math
Imports System
Imports System.IO
Imports System.Drawing.Imaging
Imports System.Drawing
Imports Excel = Microsoft.Office.Interop.Excel
Imports System.Web.UI.DataVisualization.Charting
Imports System.Windows.Forms.DataVisualization.Charting

Public Class test
    'Declaration of variables that are used throughout the whole program.
    Private Shared output As String = ""
    Dim CSVarray As Array
    Dim CSVarray2 As Array
    Dim distance As Integer
    Dim n_true_counter As Integer
    Dim forcounter As Integer
    Dim namearray(500) As String
    Dim prev_xmin As Integer
    Dim prev_xmax As Integer
    Dim prev_ymax As Integer
    Dim prev_ymin As Integer
    Dim Customaxis As Boolean
    Dim Filename As String
    Dim csvclicked As Boolean = False
    Dim proceedclicked As Boolean = False
    Dim xaxis As Integer = 2
    Dim differencenamearray(1) As String
    Dim globaloutput As String
    Dim currentserieamount As Integer = 0
    Dim globaldifferencearray As Array
    Dim errorgraphpushed As Boolean = False
    Dim globalnarray As Array
    Dim checkedornotarray As Array
    Dim globallabelcounter As Integer = 0
    Dim globalk As Integer = 0
    Dim changexaxis As Integer = 0
    Dim changexaxisclicked As Boolean = False
    Dim binfilebool As Boolean = False
    Dim globalfilename As String
    Dim scaledboolean As Boolean = False
    Dim globalscaleerrarray As Array
    Dim globallabeldelete As Boolean = False
    Dim emptylabelspace(500)
    Private Sub Open_CSV_Button_Click() Handles Open_CSV_Button.Click 'Sub that
handles the click event on the "offline" tab of the tool.
        Dim Filename As String
        Static clickamount As Integer = 0
        Dim Rowlength As Integer
        Dim Columnlength As Integer
        Dim FieldCounter As Integer = 0
        Dim Counter2 As Integer = 0
        Dim Counter3 As Integer = 0
        Dim Counter4 As Integer = 0
        Dim Counter5 As Integer = 0
        clickamount = clickamount + 1 'Variable to count the amount of clicks on the
"Open CSV File" Button.
        CSVloadinglabel.Visible = True 'Loading info in the infoscreen is shown
        ProgressBar1.Visible = True 'Loading info in the infoscreen is shown
        Chart2.Visible = True 'Show the graphchart

```

```

If binfilebool = False Then
    OpenFileDialog1.ShowDialog() 'Show the dialog
    Filename = OpenFileDialog1.FileName 'Attach Filename to the selected item
ElseIf binfilebool = True Then
    Filename = Me.globalfilename
End If

ProgressBar1.Value = 10 'Set the progress to a certain value
Try
    Using MyReader As New Microsoft.VisualBasic.
        FileIO.TextFieldParser(Filename)
        MyReader.TextFieldType = FileIO.FieldType.Delimited
        MyReader.SetDelimiters(",") 'Read through data with , as delimiter.

        Dim currentRow As String()
        While Not MyReader.EndOfData
            Try
                currentRow = MyReader.ReadFields() 'Read all fields on the
current line, return them as an array of strings and proceed to the next line.
                Dim currentField As String
                For Each currentField In currentRow
                    FieldCounter = FieldCounter + 1 'Count all fields.
                    Rowlength = currentRow.GetLength(0) 'Get the length of the
rows.
                Next
            Catch ex As Microsoft.VisualBasic. ' Catch exception
                FileIO.MalformedLineException
                MsgBox("Line " & ex.Message &
                    "is not valid and will be skipped.", 0, "DDT")
            End Try
        End While
    End Using
Catch f As Exception
    GoTo line2
End Try
ProgressBar1.Value = 30
Columnlength = FieldCounter / Rowlength
Dim CSVarray((Columnlength - 2) * clickamount, (Rowlength - 1) * clickamount)
As Double
Try
    Using MyReader As New Microsoft.VisualBasic.
        FileIO.TextFieldParser(Filename)
        MyReader.TextFieldType = FileIO.FieldType.Delimited
        MyReader.SetDelimiters(",") 'Read through data with , as delimiter.

        Dim currentRow As String()
        While Not MyReader.EndOfData
            Try
                currentRow = MyReader.ReadFields()
                Dim currentField As String
                For Each currentField In currentRow
                    If Counter4 > Rowlength - 1 Then 'Skip the first line
which contains a string
                        If Counter2 > Rowlength - 1 Then 'Proceed when the end
of the column has reached with the next column
                            Counter3 = Counter3 + 1
                            Counter2 = 0
                        End If
                        CSVarray(Counter3, Counter2) = currentField 'Address
the values to the array
                            Counter2 = Counter2 + 1
                        End If
                    Counter4 = Counter4 + 1
                Next
            End Try
        End While
    End Using
Catch f As Exception
    GoTo line2
End Try

```

```

        Next
    Catch ex As Microsoft.VisualBasic.
        FileIO.MalformedLineException
        MsgBox("Line " & ex.Message &
            "is not valid and will be skipped.", 0, "DDT")
    End Try
End While
End Using
Catch g As Exception
    GoTo line2
End Try
ProgressBar1.Value = 50
Me.CSVarray = CSVarray
Dim Namearray(Rowlength - 1) As String
Try
    Using MyReader As New Microsoft.VisualBasic.
        FileIO.TextFieldParser(Filename)
        MyReader.TextFieldType = FileIO.FieldType.Delimited
        MyReader.SetDelimiters(",") 'Read through data with , as delimiter.

        Dim currentRow As String()
        While Not MyReader.EndOfData
            Try
                currentRow = MyReader.ReadFields()
                Dim currentField As String
                For Each currentField In currentRow
                    If Counter5 <= Rowlength - 1 Then
                        Namearray(Counter5) = currentField
                        Counter5 = Counter5 + 1
                    End If
                Next
            Catch ex As Microsoft.VisualBasic.
                FileIO.MalformedLineException
                MsgBox("Line " & ex.Message &
                    "is not valid and will be skipped.", 0, "DDT")
            End Try
        End While
    End Using
Catch h As Exception
    GoTo line2
End Try
ProgressBar1.Value = 70 'Set the progressbar value to 70%
For i = 0 To Counter5 - 1
    CSVOfflineTreeview.Nodes.Add(Namearray(i)) ' Add the data to the treeviews
    TreeViewKeyFigures.Nodes.Add(Namearray(i)) ' '
    PreviousTreeview.Nodes.Add(Namearray(i)) ' '
    TreeViewXAxis.Nodes.Add(Namearray(i)) ' '
    Me.namearray(i) = Namearray(i) ' Give the global name array the same
values as the local one.
Next
ProgressBar1.Value = 80 'Set the progressbar value to 80%
Me.Filename = Filename
'Show and hide buttons and labels.
Me.csvclicked = True
Label8.Visible = True
CSVOfflineTreeview.Visible = True
TreeViewXAxis.Visible = True
OpencsvofflineLabel.Visible = False
ProgressBar1.Value = 100
Select_All_KeyTreeview_Button.Visible = True
Deselect_All_KeyTreeview_Button.Visible = True
InverseKey_Button.Visible = True
line2:
    ProgressBar1.Visible = False

```

```

    CSVloadinglabel.Visible = False
    xaxisselectionlabel.Visible = True
  End Sub
  Private Sub SelectAll_Click(sender As Object, e As EventArgs) Handles
SelectAll.Click ' Handles the select all button on the offline tab
    Dim parent As TreeNode
    For Each parent In CSVOfflineTreeview.Nodes 'Loop through the CSV Offline
Treeview nodes
        parent.Checked = True ' Check all checkboxes
    Next

  End Sub

  Private Sub DeselectAll_Click(sender As Object, e As EventArgs) Handles
DeselectAll.Click ' Handles the deselect button on the offline tab
    Dim parent As TreeNode

    For Each parent In CSVOfflineTreeview.Nodes 'Loop through the CSV Offline
Treeview nodes
        parent.Checked = False 'Uncheck all checkboxes
    Next

  End Sub
  Private Sub Aftercheck(sender As Object, e As EventArgs) Handles
CSVOfflineTreeview.AfterCheck 'Handles what has to happen after a checkbox in the
treeview on the offline tab has been checked or unchecked
    Dim n_counter As Integer = 0
    Dim n_array(500)
    Dim m_array(500)
    Dim Amount As Integer
    Dim seriercounter As Integer = 0
    Dim n As TreeNode
    Dim deletecounter As Integer = 0
    Dim m As TreeNode
    Dim xaxis As Integer
    Dim emptyarrayindex As Integer = 0
    Static previouscheckedarray(500)
    Dim deletenumber As Integer = 0
    Dim previoustreeviewcheckedcounter As Integer = 0
    Dim delete As Boolean = False
    Dim lbl(500) As Label
    Dim deleteamount As Integer = 0
    Static lblcount As Integer = 0
    For q = 0 To 500
        lbl(q) = New Label
        lbl(q).Text = ""
    Next

    xaxis = Me.xaxis
    Chart2.ChartAreas(0).CursorX.IsUserSelectionEnabled = True
'Let the user select a region to zoom into with the mouse on the X-axis
    Chart2.ChartAreas(0).CursorY.IsUserSelectionEnabled = True
'Let the user select a region to zoom into with the mouse on the Y-axis
    Amount = CSVarray.GetLength(0)
'Get the lenght of the column of the csv file

    For Each n In CSVOfflineTreeview.Nodes
        If n.Checked = True Then
            n_array(n_counter) = n.Index
'Fill an array with the necessary index values. This is to overcome a problem that
occurs when checking a box and another one has already been checked.
            n_counter = n_counter + 1
            infolabel.Text = namearray(n.Index) & " is now selected!"
'Display the info on the infolabel

```

```

        n.Tag = "checked"
        ElseIf n.Checked = False And PreviousTreeview.Nodes(n.Index).Checked =
True And Me.changexaxis = 0 Then ' Check if the current value was checked before to
know which graph to delete
            'n.index = the number which now is deselected.
            n.Tag = "remove"
            delete = True

        End If
    Next

    If delete = True Then
        For Each n In CSVOfflineTreeview.Nodes
            If n.Tag = "checked" Then 'If the current node has
a "checked" tag
                deletecounter = deletecounter + 1 'Count 1 up at the
deletecounter for each node that doesn't need to be deleted
                ElseIf n.Tag = "remove" And Me.changexaxis = False Then 'If the
node is tagged for removal and the user hasn't changed the x axis
                    Try
                        Chart2.Series.RemoveAt(deletecounter) 'Try to
remove the current one
                        LegendGroup.Controls.RemoveAt(deletecounter) 'Remove
the legendtext
                        'Get the created empty space
                        globallabeldelete = True ' Set the
global variable for a deleted label to true
                        emptylabelspace(deleteamount) = deletecounter * 13
                    ,
                        deleteamount = deleteamount + 1
                    Catch
                        GoTo line2 'On
failure, exit the for loop
                    End Try
                    Me.currentserieamount = Me.currentserieamount - 1 'Remove
one from the seriecounter, to make the global seriecounter even again.
                    n.Tag = ""
                    GoTo line2
                End If
            Next
            delete = False 'Set the delete boolean back to false to reset the
delete procedure.
        End If

line2:
    For j = 0 To 500
        If n_array(j) Is Nothing Then
            emptyarrayindex = j
            GoTo Line1
        End If
    Next

Line1:
    For i = Me.currentserieamount To emptyarrayindex - 1 'Start with the last
added serie
        Chart2.ChartAreas(0).CursorX.IsUserSelectionEnabled = True
        'Let the user select a region to zoom into with the mouse on the X-axis
        Try
            Chart2.Series.Add(i) 'Add a serie
            If Me.xaxis = 1 Or Me.xaxis = 0 Then
                Chart2.Series(i).XValueType = ChartValueType.Time
            End If
        End Try
    Next

```

```

        Chart2.Series(i).ChartType =
DataVisualization.Charting.SeriesChartType.Line           'Make a line-style
graph instead of a bar-style
        End If
        Chart2.Series(i).ChartType =
DataVisualization.Charting.SeriesChartType.Line           'Make a line-style
graph instead of a bar-style
        Catch
        Chart2.Series.Add(i + 1)   'Add a serie +1 if the serie already
exists.
        If Me.xaxis = 1 Or Me.xaxis = 0 Then
        Chart2.Series(i + 1).XValueType = ChartValueType.Time
        Chart2.Series(i + 1).ChartType =
DataVisualization.Charting.SeriesChartType.Line           'Make a line-style
graph instead of a bar-style
        End If
        Chart2.Series(i).ChartType =
DataVisualization.Charting.SeriesChartType.Line           'Make a line-style
graph instead of a bar-style
        End Try

        'Chart2.Series(i).LegendText = namearray(n_array(i))
'Set the legend text to the correct relevant name
        LegendGroup.Controls.Add(lbl(i))
        lbl(i).Text = namearray(n_array(i))
        lbl(i).Size = New Size(400, 13)
        If globallabeldelete = False And deleteamount = 0 Then
        lbl(i).Location = New Point(6, 30 + lblcount) 'Give the correct
location of the new label
        lblcount = lblcount + 13
        End If

        If globallabeldelete = True And Not deleteamount = 0 Then
        lbl(i).Location = New Point(6, 30 + emptylabelspace(deleteamount))
'Give the correct location of the new label
        deleteamount = deleteamount - 1
        globallabeldelete = False ' Reset that a label has been deleted
        End If

        Chart2.ApplyPaletteColors() 'Let the program be able to access the colors
of the series
        lbl(i).ForeColor = Chart2.Series(i).Color

        For k = 0 To Amount - 1
        If Not xaxis = -1 And Not Me.xaxis = 1 Then
        Chart2.Series(i).Points.AddXY(CSVarray(k, xaxis), CSVarray(k,
n_array(i))) 'Add Data points to make the graph.
        ElseIf Me.xaxis = 1 Then
        Chart2.Series(i).Points.AddXY(CSVarray(k, xaxis), CSVarray(k,
n_array(i))) 'Add Data points to make the graph.
        Else
        Chart2.Series(i).Points.AddXY(CSVarray(k, 2), CSVarray(k,
n_array(i))) 'Add Data points to make the graph.
        End If
        Next
        Me.currentserieamount = Me.currentserieamount + 1
    Next

    globalnarray = n_array

    previouscheckedarray = n_array
    For Each n In CSVOfflineTreeview.Nodes

```

```

    If n.Checked = True Then
        PreviousTreeView.Nodes(n.Index).Checked = True 'copy the checkboxes in
another treeview for deleting purposes
    ElseIf n.Checked = False Then
        PreviousTreeView.Nodes(n.Index).Checked = False 'copy the checkboxes
in another treeview for deleting purposes
    End If
Next
End Sub

```

```

Private Sub ApplyButton_Click(sender As Object, e As EventArgs) Handles
ApplyButton.Click ' Handles what happens when the apply button is pushed on the
offline tab.
    Dim seriecouter As Integer = 0

    seriecouter = Chart2.Series.Count

    If seriecouter > 0 Then 'Prevent applying settings before a serie has been
added
        Dim newxarray(CSVarray.GetLength(0) - 1) As String
        Me.prev_xmin = Chart2.ChartAreas(0).AxisX.Minimum 'Store the current
values of the scale for later use in the reset button.
        Me.prev_xmax = Chart2.ChartAreas(0).AxisX.Maximum
        Me.prev_ymin = Chart2.ChartAreas(0).AxisY.Minimum
        Me.prev_ymax = Chart2.ChartAreas(0).AxisY.Maximum
        If Not X_Min_TextBox.Text = vbNullString Then 'check if the boxes are
not empty.
            Chart2.ChartAreas(0).AxisX.Minimum = X_Min_TextBox.Text
            infolabel.Text = "Custom axis values applied!"
            Customaxis = True 'Set customaxis to true for other parts in
the tool
        End If
        If Not X_Max_TextBox.Text = vbNullString Then
            Chart2.ChartAreas(0).AxisX.Maximum = X_Max_TextBox.Text
            infolabel.Text = "Custom axis values applied!"
            Customaxis = True
        End If
        If Not Y_Min_TextBox.Text = vbNullString Then
            Chart2.ChartAreas(0).AxisY.Minimum = Y_Min_TextBox.Text
            infolabel.Text = "Custom axis values applied!"
            Customaxis = True
        End If
        If Not Y_Max_TextBox.Text = vbNullString Then
            Chart2.ChartAreas(0).AxisY.Maximum = Y_Max_TextBox.Text
            infolabel.Text = "Custom axis values applied!"
            Customaxis = True
        End If
        If Abs_Zero.Checked = True Then 'Make the timestamp start from 0
            For i = 0 To CSVarray.GetLength(0) - 1
                newxarray(i) = CSVarray(i, xaxis) - CSVarray(0, xaxis)
                Chart2.Series(0).Points(i).AxisLabel = newxarray(i).ToString
                Chart2.ChartAreas(0).AxisX.Interval = 10000
            Next
        End If
    Else
        MsgBox("No graph to apply settings to.", 0, "DDT")
    End If
End Sub

```

```

Private Sub Reset_Button_Click(sender As Object, e As EventArgs) Handles
Reset_Button.Click 'Handles the reset button on the offline tab.

```

```

Chart2.ChartAreas(0).AxisX.Minimum = Me.prev_xmin
Chart2.ChartAreas(0).AxisX.Maximum = Me.prev_xmax
Chart2.ChartAreas(0).AxisY.Minimum = Me.prev_ymin
Chart2.ChartAreas(0).AxisY.Maximum = Me.prev_ymax
X_Min_TextBox.Text = "" 'Delete the text in the boxes.
X_Max_TextBox.Text = ""
Y_Min_TextBox.Text = ""
Y_Max_TextBox.Text = ""

infolabel.Text = "Axis settings set back to default!"
Me.Customaxis = False
Else
  MsgBox("Nothing to reset.", 0, "DDT")
End If

End Sub

Private Sub Aftercheck2(sender As Object, e As TreeViewEventArgs) Handles
TreeViewXAxis.AfterCheck ' Handles the treeview for the x-axis.
  Dim n As TreeNode
  Dim counter As Integer = 0
  For Each n In TreeViewXAxis.Nodes 'Loop through the x-axis treeview. This
will only happen once, because only 1 can be clicked per time.
    If n.Checked = True Then
      Me.xaxis = n.Index 'Select the x-axis
      MsgBox(namearray(n.Index) & " is selected as x-axis.", 0, "DDT")
      infolabel.Text = namearray(n.Index) & " is selected as x-axis."
      counter = counter + 1
    End If
  Next
  If counter > 0 Then 'If an x axis has been chosen, hide the x axis
treeview
    TreeViewXAxis.Visible = False
    xaxisselectionlabel.Visible = False
    SelectAll.Visible = True
    DeselectAll.Visible = True
    Change_X_Xaxis.Visible = True
  End If
  If Not Me.xaxis = 1 Then
    Chart2.ChartAreas(0).AxisX.Title = namearray(xaxis)
  ElseIf Me.xaxis = 1 Then
    Chart2.ChartAreas(0).AxisX.Title = namearray(xaxis) & " [Hours:Minutes]"
  End If

  Me.changexaxis = 0
End Sub

Private Sub Change_X_Xaxis_Click(sender As Object, e As EventArgs) Handles
Change_X_Xaxis.Click 'Handles the change x-axis button.
  Dim amount As Integer = 0
  Dim newxaxis As Integer = 2
  Me.changexaxis = 1
  System.Threading.Thread.Sleep(1000)
  amount = Me.currentserieamount
  xaxisselectionlabel.Visible = True
  SelectAll.Visible = False
  DeselectAll.Visible = False
  MsgBox("Please select a new x-axis in the treeview to the right.", 0, "DDT")

  TreeViewXAxis.Nodes(Me.xaxis).Checked = False
  TreeViewXAxis.Visible = True
  Me.changexaxis = 1
  For i = amount - 1 To 0 Step -1
    Chart2.Series.RemoveAt(i)

```

```

    CSVOfflineTreeview.Nodes(globalnarray(i)).Checked = False 'Uncheck the
previous checked values for the y axes, to overcome that double x-axis are used, which
will give an error.

```

```

    Next
    Me.changexaxis = 1

```

```

    Me.currentserieamount = 0
End Sub

```

```

Private Sub Help_Button_Click(sender As Object, e As EventArgs) Handles
Help_Button.Click ' Handles the global help button.
    My.Forms.Form2.Show() ' Show the second form.
End Sub

```

```

Private Sub SaveButton_Click(sender As Object, e As EventArgs) Handles
SaveButton.Click ' Handles the event that happens when the save button is clicked.
    Dim never As Boolean = False
    Dim array(500)

```

```

    If FilenameTextbox.Text = vbNullString Or FilenameTextbox.Text.Contains("/")
Or FilenameTextbox.Text.Contains("\") Or FilenameTextbox.Text.Contains(":") Or
FilenameTextbox.Text.Contains("*") Or FilenameTextbox.Text.Contains("?") Or
FilenameTextbox.Text.Contains("<") Or FilenameTextbox.Text.Contains(">") Or
FilenameTextbox.Text.Contains("|") Or FilenameTextbox.Text.Contains(Chr(34)) Then
        MsgBox("Please type in a valid filename.", 0, "DDT") 'Display a message
when the filename contains characters that are not allowed.
    Else

```

```

        Dim regDate As Date = Date.Now() 'Get the current time.
        Dim strDate As String = regDate.ToString("dd MM yyyy") 'Translate the
current time into a specific format
        If (Not System.IO.Directory.Exists("Savefile " & strDate)) Then 'Make a
folder with the time string if it doesn't exist already.
            System.IO.Directory.CreateDirectory("Savefile " & strDate)
        End If
        If Not SaveCheck1.Checked = True And Not SaveCheck2.Checked = True And Not
SaveCheck3.Checked = True And Not SaveCheck4.Checked = True Then ' check whether the
boxes are checked

```

```

            MsgBox("Please check at least one of the check boxes.", 0, "DDT")
        Else

```

```

            If SaveCheck1.Checked = True Then
                Dim objWriter As New System.IO.StreamWriter("Savefile " & strDate
& "/" & FilenameTextbox.Text & ".csv")
                Dim n As TreeNode
                Dim counter As Integer = 0

```

```

                For Each n In CSVOfflineTreeview.Nodes
                    If n.Checked = True Then
                        array(counter) = n.Index ' fill an array to be able to get
the checked values later on
                        counter = counter + 1
                    End If
                Next

```

```

                Dim savearray(CSVarray.GetLength(0), counter) 'make an array for
the selected nodes

```

```

                For k = 0 To counter - 1
                    For m = 0 To CSVarray.GetLength(0) - 1
                        savearray(m, k) = CSVarray(m, array(k)) 'Store the values
in the array

```

```

                    Next
                Next

```

```

    For i = 0 To counter - 1
      objWriter.Write(namearray(array(i))) 'write the names
      If i < counter - 1 Then 'only write a comma when the line is
not at it's end
        objWriter.Write(",")
      End If
    Next

    objWriter.Write(objWriter.NewLine) 'Write a new line, to separate
the head titles from the numbers.

    For j = 0 To CSVarray.GetLength(0) - 1
      For k = 0 To counter - 1
        objWriter.Write(savearray(j, k)) 'Write the values in line
shape (from left to right)
        If k < counter - 1 Then 'only write a comma when the line
is not at it's end
          objWriter.Write(",")
        End If
      Next
      If j < CSVarray.GetLength(0) - 1 Then
        objWriter.Write(objWriter.NewLine) 'After every line,
write an empty line for the next piece of code, if the code is not yet at the end
      End If
    Next

    objWriter.Close() 'Close the stream
    Process.Start("Savefile " & strDate)
  End If

  If SaveCheck2.Checked = True Then ' If box two is checked
    Dim m As CheckBox
    Dim objWriter As New System.IO.StreamWriter("Savefile " & strDate
& "/" & FilenameTextbox.Text & ".txt") 'Open up a textfile
    objWriter.WriteLine(strDate) ' Write the current date to the file
    For i = 0 To globallabelcounter - 1 ' For each keyfigure
      m = GroupBox6.Controls(i)

      objWriter.WriteLine(m.Text) 'write the keyfigure result in the
text file

      If i = globallabelcounter - 1 Then
        objWriter.Close() 'Close the file when all keyfigures are
written to the file.

        Keyfiguressaveinfo.Text = "The Keyfigures are saved in:
" & "Savefile" & strDate & "/" & FilenameTextbox.Text & ".txt"
        Process.Start("Savefile " & strDate)
      End If
    Next
  End If

  If SaveCheck3.Checked = True Then ' If box three is checked.
    Chart2.SaveImage("Savefile " & strDate & "/" &
FilenameTextbox.Text & ".png", System.Drawing.Imaging.ImageFormat.Png) ' Save the
chart as a png file.

    Graphsaveinfo.Text = "The Graphs have been saved!"
    Process.Start("Savefile " & strDate)
  End If

  If SaveCheck4.Checked = True Then
    If MaxCutOff_Rad.Checked = True And MaxCutOff_Text.Text <>
vbNullString And IsNumeric(MaxCutOff_Text.Text) Then
      Dim arraylength As Integer
      Dim n As TreeNode
      Dim counter As Integer = 0
      Dim counter1 As Integer = 0
      Dim counter2 As Integer = 1

```

```

Chart2.Series.Add(Me.currentserieamount)
For r = 0 To CSVarray.GetLength(0) - 1

Chart2.Series(Me.currentserieamount).Points.AddXY(CSVarray(r, Me.xaxis),
MaxCutOff_Text.Text)
Next
Chart2.Series(Me.currentserieamount).ChartType =
DataVisualization.Charting.SeriesChartType.Line

For Each n In CSVOfflineTreeview.Nodes
If n.Checked = True Then
array(counter) = n.Index ' fill an array to be able to
get the checked values later on
counter = counter + 1
End If
Next
If counter = 0 Then
GoTo line1
End If

Dim savearray(CSVarray.GetLength(0), counter) 'make an array
for the selected nodes
For k = 0 To counter - 1
For m = 0 To CSVarray.GetLength(0) - 1
savearray(m, k) = CSVarray(m, array(k)) 'Store the
values in the array
Next
Next

Dim cutoffarray(CSVarray.GetLength(0), counter)
For a = 0 To counter - 1
For g = 0 To CSVarray.GetLength(0) - 1
If MaxSmaEqu_Rad.Checked = True Then

If savearray(g, a) <= MaxCutOff_Text.Text Then
cutoffarray(counter1, 0) = CSVarray(g, 1)

'Store timestamp data
a)
cutoffarray(counter1, counter2) = savearray(g,
counter1 = counter1 + 1 'Count one up

End If
ElseIf MaxSma_Rad.Checked = True Then
If savearray(g, a) < MaxCutOff_Text.Text Then
cutoffarray(counter1, 0) = CSVarray(g, 1)

'Store timestamp data
a)
cutoffarray(counter1, counter2) = savearray(g,
counter1 = counter1 + 1 'Count one up

End If
Else
GoTo line1
End If
Next
counter2 = counter2 + 1
counter1 = 0

Next

Dim objWriter As New System.IO.StreamWriter("Savefile " &
strDate & "/" & FilenameTextbox.Text & "CutOff" & ".csv")
objWriter.Write(namearray(0) & ",") 'Write the timestamp

```

```

'write the names
For i = 0 To counter - 1
  objWriter.Write(namearray(array(i))) 'write the names
  If i < counter - 1 Then 'only write a comma when the line
    objWriter.Write(",")
  End If
Next

For o = 0 To cutoffarray.GetLength(0)
  If IsNothing(cutoffarray(o, 0)) Then 'If there is nothing
    arraylength = o 'Get the length of the total filled
  End If
Next

line2:
  objWriter.Write(objWriter.NewLine) 'Write a new line, to
  seperate the head titles from the numbers.

  For j = 0 To arraylength - 1
    For l = 0 To counter
      objWriter.Write(cutoffarray(j, l)) 'Write the values
      If l < counter Then 'only write a comma when the line
        objWriter.Write(",")
      End If
    Next
    If j < arraylength - 1 Then
      objWriter.Write(objWriter.NewLine) 'After every line,
      write an empty line for the next piece of code, if the code is not yet at the end
    End If
  Next
  objWriter.Close()
  Process.Start("Savefile " & strDate)
  ElseIf MaxCutOff_Rad.Checked = True And (MaxCutOff_Text.Text =
vbNullString Or Not IsNumeric(MaxCutOff_Text.Text)) Then 'If the radiobutton is
checked, but it is empty or contains something other than numbers display a warning
  MsgBox("Enter a valid number.", 0, "ddt")
End If
  If MinCutOff_Rad.Checked = True And MinCutOff_Text.Text <>
vbNullString And IsNumeric(MinCutOff_Text.Text) Then
    Dim arraylength As Integer
    Dim n As TreeNode
    Dim counter As Integer = 0
    Dim counter1 As Integer = 0
    Dim counter2 As Integer = 1
    arraylength = CSVarray.GetLength(0)

    For Each n In CSVOfflineTreeview.Nodes
      If n.Checked = True Then
        array(counter) = n.Index ' fill an array to be able to
        counter = counter + 1
      End If
    Next
    If counter = 0 Then
      GoTo line1
    End If
    Dim savearray(CSVarray.GetLength(0), counter) 'make an array
    For k = 0 To counter - 1

```

```

    For m = 0 To CSVarray.GetLength(0) - 1
        savearray(m, k) = CSVarray(m, array(k)) 'Store the
values in the array
    Next
Next

Dim cutoffarray(CSVarray.GetLength(0), counter)
For a = 0 To counter - 1
    For g = 0 To CSVarray.GetLength(0) - 1

        If MinGreaEqu_Rad.Checked = True Then

            If savearray(g, a) >= MinCutoff_Text.Text Then
                cutoffarray(counter1, 0) = CSVarray(g, 1)

                cutoffarray(counter1, counter2) = savearray(g,
a)

                counter1 = counter1 + 1
            End If

            ElseIf MinGrea_Rad.Checked = True Then
                If savearray(g, a) > MinCutoff_Text.Text Then
                    cutoffarray(counter1, 0) = CSVarray(g, 1)

                    cutoffarray(counter1, counter2) = savearray(g,
a)

                    counter1 = counter1 + 1

                End If

            Else
                GoTo line1
            End If
        Next
        counter2 = counter2 + 1
        counter1 = 0
    Next

    Dim objWriter As New System.IO.StreamWriter("Savefile " &
strDate & "/" & FilenameTextbox.Text & "CutOff" & ".csv")
    objWriter.Write(namearray(0) & ",") 'Write the timestamp

    For i = 0 To counter - 1
        objWriter.Write(namearray(array(i))) 'write the names
        If i < counter - 1 Then 'only write a comma when the line
is not at it's end
            objWriter.Write(",")
        End If
    Next

    For o = 0 To cutoffarray.GetLength(0)
        If IsNothing(cutoffarray(o, 0)) Then 'If there is nothing
left in the array, get the length of the total filled part
        arraylength = o 'Get the length of the total filled
array

        GoTo line3 'exit the for loop
    End If
Next

line3:
    objWriter.Write(objWriter.NewLine) 'Write a new line, to
seperate the head titles from the numbers.

    For j = 0 To arraylength - 1
        For l = 0 To counter

```

```

        objWriter.Write(cutoffarray(j, 1)) 'Write the values
in line shape (from left to right)
        If l < counter Then 'only write a comma when the line
is not at it's end
            objWriter.Write(",")
        End If
    Next
    If j < arraylength - 1 Then
        objWriter.Write(objWriter.NewLine) 'After every line,
write an empty line for the next piece of code, if the code is not yet at the end
    End If
    Next
    objWriter.Close()
    Process.Start("Savefile " & strDate)
    ElseIf MinCutoff_Rad.Checked = True And (MinCutoff_Text.Text =
vbNullString Or Not IsNumeric(MinCutoff_Text.Text)) Then 'If the radiobutton is
checked, but it is empty or contains something other than numbers display a warning
        MsgBox("Enter a valid number.")
    End If
End If

    End If
End If
If never = True Then
line1:
    MsgBox("No nodes were checked", 0, "DDT")
End If

End Sub

Private Sub SendButton_Click(sender As Object, e As EventArgs) Handles
SendButton.Click
    If EmailTextbox.Text = vbNullString Then
        MsgBox("Please type in a valid e-mail address.", 0, "DDT")
    End If
    If Not SendCheck1.Checked = True And Not SendCheck2.Checked = True And Not
SendCheck3.Checked = True Then
        MsgBox("Please check at least one of the check boxes.", 0, "DDT")
    End If
End Sub

Public Sub Connect(ByVal serverIP As String, ByVal message As String)
    Dim output As String = ""
    While (True)
        System.Threading.Thread.Sleep(10)
        Try
            ' Create a TcpClient.
            ' The client requires a TcpServer that is connected
            ' to the same address specified by the server and port
            ' combination.
            Dim port As Int32 = 13
            Dim client As New TcpClient(serverIP, port)

            ' Translate the passed message into ASCII and store it as a byte
array.
            Dim data(255) As [Byte]
            data = System.Text.Encoding.ASCII.GetBytes(message)

            ' Get a client stream for reading and writing.
            ' Stream stream = client.GetStream();
            Dim stream As NetworkStream = client.GetStream()

            ' Send the message to the connected TcpServer.
            stream.Write(data, 0, data.Length)

```

```

output = "Sent: " + message
MsgBox(output)

' Buffer to store the response bytes.
data = New [Byte](255) {}

' String to store the response ASCII representation.
Dim responseData As String = String.Empty

' Read the first batch of the TcpServer response bytes.

If stream.DataAvailable Then
    Dim bytes As Int32 = stream.Read(data, 0, data.Length)
    responseData = System.Text.Encoding.ASCII.GetString(data, 0,
bytes)
    output = responseData
End If

If output = "Goodbye" Then
    Me.globaloutput = output
    Status_Label.Text = "Connected"
    Error_Info_Button.Visible = False
    GoTo line1
Else
    Status_Label.Text = "No response from server."
End If
' Close everything.
stream.Close()
client.Close()
Catch e As ArgumentNullException
    output = "ArgumentNullException: " + e.ToString()
    Me.globaloutput = output
    MsgBox(output)
    Status_Label.Text = "Argument Null error. Click the button below to
find out more."
    Error_Info_Button.Visible = True

Catch e As SocketException
    output = "SocketException: " + e.ToString()

    Me.globaloutput = output
    Status_Label.Text = "No response from server. Click the button below
to find out more."
    Error_Info_Button.Visible = True

End Try
End While
line1:
End Sub

Private Sub Connect_Button_Click(sender As Object, e As EventArgs) Handles
Connect_Button.Click
    'In this code example, use a hard-coded
    'IP address And message.
    Dim serverIP As String = "localhost"
    Dim message As String = "Hello"
    Connect(serverIP, message)

End Sub

Private Sub Error_Info_Button_Click(sender As Object, e As EventArgs) Handles
Error_Info_Button.Click

```

```

  MsgBox("No connection with Server. Press the connect button again, or try
  restarting the server via TeamViewer.")
  End Sub

```

```

  Private Sub Show_Button_Click(sender As Object, e As EventArgs) Handles
  Show_Button.Click ' Handles the show button click event on the keyfigures tab
    Dim n As TreeNode
    Dim indexarray(500)
    Dim indexcounter As Integer
    indexcounter = 0
    Dim emptyarrayindex As Integer
    Dim countarray As Array
    countarray = Me.CSVarray
    Dim k As Integer
    Dim arraylength As Integer
    Dim biggerprevious As Integer = 0
    Dim smallerprevious As Integer = 999999999
    Dim summation As Double = 0
    Dim summation2 As Double = 0
    Dim currentvalue As Double = 0
    Dim mean As Double = 0
    Dim lbl(500) As CheckBox
    Dim treenodecheckcounter As Integer = 0
    'Dim w As TreeNode
    Dim summation3 As Integer = 0
    Dim arraylength As Integer = 0
    Dim average As Integer = 0
    Dim executed As Boolean = False
    Dim namedifferencearray(1)
    Dim biggererrorprevious As Integer = 0

    If Me.csvclicked = True Then          'If a CSV file has been loaded in the
  offline tab, proceed
      arraylength = CSVarray.GetLength(0)
      arraylength = countarray.GetLength(0)
      Dim differencearray(arraylength)
      Opencsvofflinelabel.Visible = False
      For q = 0 To 500
          lbl(q) = New CheckBox
          lbl(q).Text = ""
      Next
      k = Me.globalk          'Get the location value
      If Not Mean_Rad.Checked = True And Not Tot_Len_Rad.Checked = True And Not
  Max_Rad.Checked = True And Not Min_Rad.Checked = True And Not AvgConErr_Rad.Checked =
  True And Not ConErrMax_Rad.Checked = True And Not ConErrMin_Rad.Checked Then
          MsgBox("Please check an option first.", 0, "DDT") ' If no option has
  been checked, show a warning message
      Else
          For Each n In TreeViewKeyFigures.Nodes 'Loop through all nodes in the
  keyfigures tab treeview
              If n.Checked = True Then
                  indexarray(indexcounter) = n.Index
                  indexcounter = indexcounter + 1
              End If
          Next
          If Not indexcounter = 0 Then
              For j = 0 To 500
                  If indexarray(j) Is Nothing Then
                      emptyarrayindex = j
                      GoTo Line1
                  End If
              Next
          End If
      End If
  
```

Line1:

```

For i = 0 To emptyarrayindex - 1
  If Tot_Len_Rad.Checked Then
    For j = 1 To arraylength - 1
      currentvalue = Abs(countarray(j, indexarray(i)) -
countarray(j - 1, indexarray(i))) 'Calculate cummulative absolute distance
      summation = summation + currentvalue 'Sum all
values
    Next
    GroupBox6.Controls.Add(lbl(i)) 'Add a label
    lbl(i).Text = "Total travelled length of " &
namearray(indexarray(i)) & " is " & summation & "." 'Add text to the label
    lbl(i).Size = New Size(400, 15)
    lbl(i).Location = New Point(6, 30 + k) 'Give the
correct location of the new label
    lbl(i).Tag = "totlen"
    k = k + 15 'Change the location +15
    summation = 0 'Reset the summation
  End If

  If Mean_Rad.Checked = True Then
    For l = 0 To arraylength - 1
      mean = countarray(l, indexarray(i)) / arraylength
      summation2 = summation2 + mean
    Next
    GroupBox6.Controls.Add(lbl(i))
    lbl(i).Text = "The mean of " & namearray(indexarray(i)) &
" is " & summation2 & "."
    lbl(i).Size = New Size(400, 15)
    lbl(i).Location = New Point(6, 30 + k) 'Give the correct
location of the new label
    lbl(i).Tag = "mean"
    k = k + 15 'Change the location +15
    summation2 = 0 'Reset the summation
  End If

  If Max_Rad.Checked = True Then
    For m = 0 To arraylength - 1
      If biggerprevious > countarray(m, indexarray(i)) Then
'If the previous value was larger, than the larger value is the most large and the
current value is smaller
        biggerprevious = biggerprevious
      Else
        biggerprevious = countarray(m, indexarray(i))
'If the previous value was smaller, the larger one is the current one
      End If
    Next
    GroupBox6.Controls.Add(lbl(i))
    lbl(i).Text = "The max of " & namearray(indexarray(i)) & "
is " & biggerprevious & "."
    lbl(i).Size = New Size(400, 15)
    lbl(i).Location = New Point(6, 30 + k) 'Give the correct
location of the new label
    lbl(i).Tag = "max"
    k = k + 15 'Change the location +15
  End If

  If Min_Rad.Checked = True Then
    For o = 0 To arraylength - 1
      If smallerprevious < countarray(o, indexarray(i)) Then
' check whether the previous value if smaller, if so than the current one is larger
and hence the previous one remains the smallest
        smallerprevious = smallerprevious

```

```

        Else
            smallerprevious = countarray(o, indexarray(i))
'The other way around as the one above.
        End If
    Next
    GroupBox6.Controls.Add(lbl(i))
    lbl(i).Text = "The min of " & namearray(indexarray(i)) & "
is " & smallerprevious & "."
    lbl(i).Size = New Size(400, 15)
    lbl(i).Location = New Point(6, 30 + k) 'Give the correct
location of the new label
    lbl(i).Tag = "min"
    k = k + 15 'Change the location +15
End If

If AvgConErr_Rad.Checked = True And executed = False Then
    If Not indexcounter = 2 Then
        MsgBox("Please select two checkboxes from the
treeview.")
    Else
        For y = 0 To 1
            namedifferencearray(y) = namearray(indexarray(y))
'Store the name in the difference array for comparing two functions
        Next

        For t = 0 To arraylength - 1
            differencearray(t) = CSVarray(t, indexarray(0)) -
CSVarray(t, indexarray(1)) 'Subtract both functions to get the difference.
        Next

        For p = 0 To arraylength - 1
            summation3 = differencearray(p) + summation3
        Next

        average = summation3 / (arraylength - 1)
        GroupBox6.Controls.Add(lbl(i))
        lbl(i).Text = "The average error of " &
namedifferencearray(0) & " and " & namedifferencearray(1) & " is " & average & "."
        lbl(i).Size = New Size(10000, 15)
        lbl(i).Location = New Point(6, 30 + k)
        lbl(i).Tag = "avgerr"
        k = k + 15
        executed = True
    End If
End If

If ConErrMax_Rad.Checked = True Then
    If Not indexcounter = 2 Then
        MsgBox("Please select two checkboxes from the
treeview.")
    Else
        For y = 0 To 1
            namedifferencearray(y) = namearray(indexarray(y))
        Next

        For t = 0 To arraylength - 1
            differencearray(t) = CSVarray(t, indexarray(0)) -
CSVarray(t, indexarray(1)) ' get the difference
        Next

        For p = 0 To arraylength - 1
            If biggererrorprevious >= differencearray(p) Then
' If the previous biggest error is bigger than the current one

```

```

        biggererrorprevious = biggererrorprevious
'The previous error is the biggest
    ElseIf biggererrorprevious < differencearray(p)
Then 'If the previous biggest error is smaller than the current one
        biggererrorprevious = differencearray(p)
'The biggest error is the current one
    Else
        MsgBox("An error occured.")
    End If
Next
GroupBox6.Controls.Add(lbl(i))
lbl(i).Text = "The maximum control error of " &
namedifferencearray(0) & " and " & namedifferencearray(1) & " is " &
biggererrorprevious & "."
        lbl(i).Size = New Size(10000, 15)
        lbl(i).Location = New Point(6, 30 + k)
        lbl(i).Tag = "ConerrMax"
        k = k + 15
        executed = True
    End If
End If
Me.globalk = k
Me.globallabelcounter = Me.globalk / 15 'Count the amount of
labels and put the result in the global label counter
Next
Else
    MsgBox("Select an item in the treeview to the right.", 0, "DDT")
End If
End If

Else
    MsgBox("Select a CSV file in the 'Offline' Tab!", 0, "DDT")

End If

End Sub

```

```

Private Sub Mean_Rad_CheckedChanged(sender As Object, e As EventArgs) Handles
Mean_Rad.CheckedChanged 'Handles what to do when the mean radiobutton has been checked
or unchecked

```

```

    Static checkcounter As Integer = 0
    checkcounter = checkcounter + 1
    If checkcounter Mod 2 > 0 Then ' If the button is checked
        Formula.Visible = True ' Show the formula
        MeanFor.Visible = True
        MeanLabel.Visible = True
    Else 'else
        Formula.Visible = False 'hide the formula
        MeanFor.Visible = False
        MeanLabel.Visible = False
    End If
End Sub

```

```

Private Sub Tot_Len_Rad_CheckedChanged(sender As Object, e As EventArgs) Handles
Tot_Len_Rad.CheckedChanged 'Handles what to do when the total length radiobutton has
been checked or unchecked

```

```

    Static checkcounter As Integer = 0
    checkcounter = checkcounter + 1
    If checkcounter Mod 2 > 0 Then ' If the button is checked
        Formula.Visible = True ' Show the formula
        TotLenFor.Visible = True

```

```

        TotAbsLenLabel.Visible = True
    Else
        Formula.Visible = False
        TotLenFor.Visible = False
        TotAbsLenLabel.Visible = False
    End If
End Sub

Private Sub Max_Rad_CheckedChanged(sender As Object, e As EventArgs) Handles
Max_Rad.CheckedChanged 'Handles what to do when the maximum radiobutton has been
checked or unchecked
    Static checkcounter As Integer = 0
    checkcounter = checkcounter + 1
    If checkcounter Mod 2 > 0 Then ' If the button is checked
        Formula.Visible = True ' Show the formula
        Maxform.Visible = True
        MaxLabel.Visible = True
    Else
        Formula.Visible = False
        Maxform.Visible = False
        MaxLabel.Visible = False
    End If
End Sub

Private Sub Min_Rad_CheckedChanged(sender As Object, e As EventArgs) Handles
Min_Rad.CheckedChanged 'Handles what to do when the minimum radiobutton has been
checked or unchecked
    Static checkcounter As Integer = 0
    checkcounter = checkcounter + 1
    If checkcounter Mod 2 > 0 Then ' If the button is checked
        Formula.Visible = True ' Show the formula
        MinForm.Visible = True
        MinLabel.Visible = True
    Else
        Formula.Visible = False
        MinForm.Visible = False
        MinLabel.Visible = False
    End If
End Sub

Private Sub Showerrorgraphbutton_Click(sender As Object, e As EventArgs) Handles
Showerrorgraphbutton.Click 'Show the error graph, which is a graph that subtracts
two graphs from eachother.
    Dim errorarray As Array
    Dim n As TreeNode
    Dim m As TreeNode
    Dim mindexarray(1)
    Dim indexcounter As Integer = 0
    errorarray = Me.CSVarray
    Dim ncounter As Integer = 0
    Dim differencenamearray As Array
    differencenamearray = Me.differencenamearray

    If csvclicked = True And Me.errorgraphpushed = False Then 'If a csvfile has
been opened and the error graph has not yet been shown on screen
        Dim arraylength As Integer = errorarray.GetLength(0)
        Dim differencearray(arraylength)
        For Each n In CSVOfflineTreeview.Nodes 'loop through all nodes
            If n.Checked = True Then
                ncounter = ncounter + 1 'Count all checked nodes
            End If
        Next
        Chart2.Series(0).XValueType = ChartValueType.Double
        Chart2.Series(1).XValueType = ChartValueType.Double

```

```

    If ncounter > 2 Then 'if more than two nodes are checked, give a
warning.
        MsgBox("Please select only two nodes in the treeview.", 0, "DDT")
    ElseIf ncounter = 2 Then

        Chart2.Series.Add(Me.currentserieamount) ' Add a new serie, which will
contain the error graph

        For Each m In CSVOfflineTreeview.Nodes
            If m.Checked = True Then 'loop thourgh all nodes
                mindexarray(indexcounter) = m.Index 'Fill an array with the
values on which places the treeview are checked
                indexcounter = indexcounter + 1
            End If
        Next

        For z = 0 To 1
            differencenamearray(z) = namearray(mindexarray(z))
        Next

        For i = 0 To arraylength - 1
            differencearray(i) = errorarray(i, mindexarray(0)) - errorarray(i,
mindexarray(1)) ' The error is the difference between the actual value and the wanted
value. Therefore a subtraction is used.

        Next

        For j = 0 To arraylength - 1
            Chart2.Series(Me.currentserieamount).Points.AddXY(errorarray(j,
axis), differencearray(j)) ' Add the relevant datapoints to plot the
error graph

            Chart2.Series(Me.currentserieamount).ChartType =
DataVisualization.Charting.SeriesChartType.Line 'Make a line-style
graph instead of a bar-style
            Chart2.Series(Me.currentserieamount).LegendText = "Error Graph"
        Next
        Me.globaldifferencearray = differencearray
        Me.errorgraphpushed = True
        Chart2.Series(0).XValueType = ChartValueType.Time
        Chart2.Series(1).XValueType = ChartValueType.Time
        Chart2.Series(2).XValueType = ChartValueType.Time

        Me.currentserieamount = Me.currentserieamount + 1
    Else
        MsgBox("Select two nodes from the treeview.", 0, "DDT")
    End If

ElseIf csvclicked = True And Me.errorgraphpushed = True Then
    MsgBox("The error graph is already shown.")
Else
    MsgBox("Please select a CSV file first.", 0, "DDT")
End If

End Sub

Private Sub AvgConErr_Rad_CheckedChanged(sender As Object, e As EventArgs) Handles
AvgConErr_Rad.CheckedChanged
    Static checkcounter As Integer = 0
    checkcounter = checkcounter + 1
    If checkcounter Mod 2 > 0 Then
        Formula.Visible = True
        Averagelabel.Visible = True
        AveConErr_Pic.Visible = True
    Else
        Formula.Visible = False

```

```

    Averagelabel.Visible = False
    AveConErr_Pic.Visible = False
  End If
End Sub

Private Sub ClearErrorGraph_Button_Click(sender As Object, e As EventArgs) Handles
ClearErrorGraph_Button.Click
  If errorgraphpushed = True Then
    Chart2.Series.RemoveAt(Me.currentserieamount - 1)
    Me.currentserieamount = Me.currentserieamount - 1
    Me.errorgraphpushed = False
  Else
    MsgBox("There is no plotted error graph.", 0, "DDT")
  End If

End Sub

Private Sub DeleteGraph_Button_Click(sender As Object, e As EventArgs)

  If Not Me.currentserieamount = 0 And Not errorgraphpushed = True Then
    Chart2.Series.RemoveAt(Me.currentserieamount - 1)
    Me.currentserieamount = Me.currentserieamount - 1
    Try
      CSVOfflineTreeview.Nodes(globalnarray(Me.currentserieamount)).Checked
= False
    Catch
      GoTo Line3 'A catch has been used to overcome a crash. Instead it now
produces a messagebox.
    Finally
      End Try
Line3:
  ElseIf errorgraphpushed = True Then
    MsgBox("First delete the error graph by pressing the 'Clear Error Graph'
button.", 0, "DDT")
  Else
    MsgBox("There is no graph to delete.", 0, "DDT")
  End If

End Sub

Private Sub Button1_Click(sender As Object, e As EventArgs) Handles
Clear_All_Key_Button.Click
  If globallabelcounter <> 0 Then
    For i = globallabelcounter - 1 To 0 Step -1
      Me.globalk = 0
      GroupBox6.Controls(0).Dispose()
    Next
    Me.globallabelcounter = 0
  Else
    MsgBox("No Keyfigures Present.", 0, "DDT")
  End If

End Sub

Private Sub Deselect_All_Key_Button_Click(sender As Object, e As EventArgs)
Handles Deselect_All_KeyTreeView_Button.Click
  Dim parent As TreeNode
  For Each parent In TreeViewKeyFigures.Nodes
    parent.Checked = False
  Next
End Sub

Private Sub Select_All_Key_Button_Click(sender As Object, e As EventArgs) Handles
Select_All_KeyTreeView_Button.Click

```

```

Dim parent As TreeNode
For Each parent In TreeViewKeyFigures.Nodes
    parent.Checked = True
Next
End Sub

Private Sub Round_Key_Button_Click(sender As Object, e As EventArgs) Handles
Round_Key_Button.Click
    Dim m As CheckBox
    Dim changestring As String
    Dim islocation As Integer
    Dim stringlength As Integer
    Dim dotlocation As Integer
    Dim substrlength As Integer
    Dim finalstring As String
    Dim roundinteger As Integer
    Dim rounding As Boolean
    Dim truncateinteger As Integer
    If globallabelcounter > 0 Then

        For i = 0 To globallabelcounter - 1
            m = GroupBox6.Controls(i)
            If m.Checked = True Then
                changestring = m.Text
                If changestring.Contains("is") Then 'Detect the word is, to
know where the numerical value will start.
                    islocation = changestring.IndexOf("is") + 2
                Else
                    MsgBox("Error.", 0, "DDT")
                End If
                stringlength = changestring.Length - 1 ' Get the length of the
string until the dot.
                Dim tempstring As String = changestring.Substring(islocation,
(stringlength - islocation)) 'Get string which contains the numerical(single/double)
value.
                Dim finalbeginstring As String = changestring.Substring(0,
islocation) ' Get the first part of the string to use at the end.
                If tempstring.Contains(".") Then
                    dotlocation = tempstring.IndexOf(".") + 1
                    'Search the location of the dot for truncation, + 1 for the number to start with
                    substrlength = tempstring.Length
                    Dim subsubstring As String = tempstring.Substring(dotlocation,
(substringlength - dotlocation))
                    Dim beginpartstring As String = tempstring.Substring(0,
dotlocation)
                    If OneDec_Rad.Checked = True And subsubstring.Length >= 1 Then
                        ' Rounding of numbers
                        Dim roundstring As String = subsubstring.Substring(1, 1)
                        roundinteger = roundstring
                        Dim truncatestring As String = subsubstring.Substring(0,
1)
                        truncateinteger = truncatestring
                        If roundinteger >= 5 And truncateinteger < 9 Then
                            finalstring = finalbeginstring & beginpartstring &
truncatestring + 1
                        Else
                            m.Text = finalstring
                        ElseIf roundinteger < 5 And truncateinteger <= 9 Then
                            finalstring = finalbeginstring & beginpartstring &
truncatestring
                        Else
                            m.Text = finalstring
                        ElseIf roundinteger >= 5 And truncateinteger = 9 Then
                            finalstring = finalbeginstring & " " & beginpartstring
                            + 1 & "." & 0
                        Else
                            m.Text = finalstring
                    End If
                End If
            End If
        Next
    End If
End Sub

```

```

    End If
ElseIf subsubstring.Length < 1 And OneDec_Rad.Checked = True
Then
    MsgBox("Truncation is not possible.", 0, "DDT")
End If
If TwoDec_Rad.Checked = True And subsubstring.Length >= 2 Then
    Dim truncatestring As String = subsubstring.Substring(0,
2)
    Dim firstnumber As String = subsubstring.Substring(1, 1)
    Dim secondnumber As String = subsubstring.Substring(2, 1)

    If firstnumber < 9 And secondnumber >= 5 Then
        finalstring = finalbeginstring & beginpartstring &
truncatestring + 1
        m.Text = finalstring
    ElseIf firstnumber <= 9 And secondnumber < 5 Then
        finalstring = finalbeginstring & beginpartstring &
truncatestring
        m.Text = finalstring
    ElseIf firstnumber = 9 And secondnumber >= 5 Then
        finalstring = finalbeginstring & " " & beginpartstring
+ 1 & "." & "00"
        m.Text = finalstring
    End If

ElseIf subsubstring.Length < 2 And TwoDec_Rad.Checked = True
Then
    MsgBox("Truncation is not possible.", 0, "DDT")
End If
If ThreeDec_Rad.Checked = True And subsubstring.Length >= 3
Then
    Dim truncatestring As String = subsubstring.Substring(0,
3)
    Dim firstnumber As String = subsubstring.Substring(2, 1)
    Dim secondnumber As String = subsubstring.Substring(3, 1)

    If firstnumber < 9 And secondnumber >= 5 Then
        finalstring = finalbeginstring & beginpartstring &
truncatestring + 1
        m.Text = finalstring
    ElseIf firstnumber <= 9 And secondnumber < 5 Then
        finalstring = finalbeginstring & beginpartstring &
truncatestring
        m.Text = finalstring
    ElseIf firstnumber = 9 And secondnumber >= 5 Then
        finalstring = finalbeginstring & " " & beginpartstring
+ 1 & "." & "000"
        m.Text = finalstring
    End If

ElseIf subsubstring.Length < 3 And ThreeDec_Rad.Checked = True
Then
    MsgBox("Truncation is not possible.", 0, "DDT")
End If
If FourDec_Rad.Checked = True And subsubstring.Length >= 4
Then
    Dim truncatestring As String = subsubstring.Substring(0,
4)
    Dim firstnumber As String = subsubstring.Substring(3, 1)
    Dim secondnumber As String = subsubstring.Substring(4, 1)

    If firstnumber < 9 And secondnumber >= 5 Then
        finalstring = finalbeginstring & beginpartstring &
truncatestring + 1
        m.Text = finalstring
    End If

```

```

    ElseIf firstnumber <= 9 And secondnumber < 5 Then
      finalstring = finalbeginstring & beginpartstring &
truncatesting
      m.Text = finalstring
    ElseIf firstnumber = 9 And secondnumber >= 5 Then
      finalstring = finalbeginstring & " " & beginpartstring
+ 1 & "." & "0000"
      m.Text = finalstring
    End If
  ElseIf subsubstring.Length < 4 And FourDec_Rad.Checked = True
Then
    MsgBox("Truncation is not possible.", 0, "DDT")
  End If
  If FiveDec_Rad.Checked = True And subsubstring.Length >= 5
Then
    Dim truncatesting As String = subsubstring.Substring(0,
5)
    Dim firstnumber As String = subsubstring.Substring(4, 1)
    Dim secondnumber As String = subsubstring.Substring(5, 1)

    If firstnumber < 9 And secondnumber >= 5 Then
      finalstring = finalbeginstring & beginpartstring &
truncatesting + 1
      m.Text = finalstring
    ElseIf firstnumber <= 9 And secondnumber < 5 Then
      finalstring = finalbeginstring & beginpartstring &
truncatesting
      m.Text = finalstring
    ElseIf firstnumber = 9 And secondnumber >= 5 Then
      finalstring = finalbeginstring & " " & beginpartstring
+ 1 & "." & "00000"
      m.Text = finalstring
    End If
  ElseIf subsubstring.Length < 5 And FiveDec_Rad.Checked = True
Then
    MsgBox("Truncation is not possible.", 0, "DDT")
  End If
  If OneDec_Rad.Checked = False And TwoDec_Rad.Checked = False
And ThreeDec_Rad.Checked = False And FourDec_Rad.Checked = False And
FiveDec_Rad.Checked = False Then
    Dim truncatesting As String = subsubstring.Substring(0,
2)
    Dim firstnumber As String = subsubstring.Substring(1, 1)
    Dim secondnumber As String = subsubstring.Substring(2, 1)

    If firstnumber < 9 And secondnumber >= 5 Then
      finalstring = finalbeginstring & beginpartstring &
truncatesting + 1
      m.Text = finalstring
    ElseIf firstnumber <= 9 And secondnumber < 5 Then
      finalstring = finalbeginstring & beginpartstring &
truncatesting
      m.Text = finalstring
    ElseIf firstnumber = 9 And secondnumber >= 5 Then
      finalstring = finalbeginstring & " " & beginpartstring
+ 1 & "." & "00"
      m.Text = finalstring
    End If
  End If
Else
  MsgBox("Nothing to truncate.", 0, "DDT")
End If
Else

```

```

        End If
        m.Checked = False
    Next
Else
    MsgBox("No Keyfigures Present.", 0, "DDT")
End If
End Sub

Private Sub Select_All_KeyCheck_button_Click(sender As Object, e As EventArgs)
Handles Select_All_KeyCheck_button.Click
    Dim m As CheckBox
    For i = 0 To globallabelcounter - 1
        m = GroupBox6.Controls(i)
        m.Checked = True
    Next
End Sub

Private Sub Deselect_All_KeyCheck_Button_Click(sender As Object, e As EventArgs)
Handles Deselect_All_KeyCheck_Button.Click
    Dim m As CheckBox
    For i = 0 To globallabelcounter - 1
        m = GroupBox6.Controls(i)
        m.Checked = False
    Next
End Sub

Private Sub DeleteSelected_Key_Button_Click(sender As Object, e As EventArgs)
Handles DeleteSelected_Key_Button.Click
    Dim m As CheckBox
    Dim indexarray(500)
    Dim truecounter As Integer = 0

    For i = 0 To Me.globallabelcounter - 1
        m = GroupBox6.Controls(i)
        If m.Checked = True Then 'First check which checkboxes are selected
            indexarray(truecounter) = i
            truecounter = truecounter + 1
        End If
    Next
    For j = 0 To truecounter - 1
        GroupBox6.Controls(indexarray(j)).Dispose()
        Me.globallabelcounter = Me.globallabelcounter - 1
        Me.globalk = Me.globalk - 15
        truecounter = truecounter - 1
    Next

End Sub

Private Sub Server_Rad_CheckedChanged(sender As Object, e As EventArgs)
    Static checkcounter As Integer = 0
    checkcounter = checkcounter + 1
    If checkcounter Mod 2 > 0 Then
        'StartServer_Button.Visible = True
    Else
        ' StartServer_Button.Visible = False
    End If
End Sub

Private Sub StartServer_Button_Click(sender As Object, e As EventArgs)
    Dim TestThread As New System.Threading.Thread(AddressOf Serverthread)
    TestThread.Start()
End Sub

Private Sub Serverthread()

```

```

End Sub

Private Sub InverseSelectionButton(sender As Object, e As EventArgs) Handles
InverseSelection_Button.Click
    Dim m As CheckBox
    For i = 0 To Me.globallabelcounter - 1
        m = GroupBox6.Controls(i)
        If m.Checked = True Then
            m.Checked = False
        ElseIf m.Checked = False Then
            m.Checked = True
        End If
    Next
End Sub

Public Structure BinFileInfo
    Public FileName As String
    Public FileSize As Int64
    Public StartTime As Date
    Public EndTime As Date
    Public StartLogBytePos As Int64
    Public StopLogBytePos As Int64
End Structure

Private Sub OpenBinClick(sender As Object, e As EventArgs) Handles
Open_Bin_Button.Click
    Dim m As Message
    Process.Start("PLC_LOG_CONVERTER.exe")

    Static pcount As Integer = 0
    While (True)
        System.Threading.Thread.Sleep(100)
        Dim p() As Process
        p = Process.GetProcessesByName("PLC_LOG_CONVERTER")
        If p.Count > 0 Then
            pcount = pcount + 1
            MsgBox("Close this whenever ready.")
        Else
            ' Process is not running
            If pcount > 0 Then
                Exit While
                GoTo line1
            Else
                MsgBox("Fileconverter has not been opened.")
            End If
        End If
    End While
line1:
    Me.binfilebool = True
    'MsgBox("Now select the CSV File that has been made. If it's not made, press
the open bin file button again.")
    'OpenFileDialog1.ShowDialog()
    'Me.globalfilename = OpenFileDialog1.FileName

    ' Open the file using a stream reader.
    Using sr As New StreamReader("Filelocation.txt")
        Dim line As String
        ' Read the stream to a string and write the string to the console.
        line = sr.ReadToEnd()
        Me.globalfilename = line
    End Using
    Open_CSV_Button_Click()
End Sub
Private Sub fileopenthread()

```

```

End Sub

Private Sub Inverse_Click(sender As Object, e As EventArgs) Handles
InverseKey_Button.Click
    Dim n As TreeNode
    For Each n In TreeViewKeyFigures.Nodes
        If n.Checked = True Then
            n.Checked = False
        ElseIf n.Checked = False Then
            n.Checked = True
        End If
    Next
End Sub

Private Sub Button1_Click_1(sender As Object, e As EventArgs) Handles
Button1.Click
    Dim m As CheckBox
    Dim changestring As String
    Dim islocation As Integer
    Dim stringlength As Integer
    Dim dotlocation As Integer
    Dim substringlength As Integer
    Dim finalstring As String
    Dim roundinteger As Integer
    Dim rounding As Boolean
    Dim truncateinteger As Integer
    If globallabelcounter > 0 Then

        For i = 0 To globallabelcounter - 1
            m = GroupBox6.Controls(i)
            If m.Checked = True Then
                changestring = m.Text
                If changestring.Contains("is") Then      'Detect the word is, to
know where the numerical value will start.
                    islocation = changestring.IndexOf("is") + 2
                Else
                    MsgBox("Error.", 0, "DDT")
                End If
                stringlength = changestring.Length - 1 ' Get the length of the
string until the dot.
                Dim tempstring As String = changestring.Substring(islocation,
(stringlength - islocation)) 'Get string which contains the numerical(single/double)
value.
                Dim finalbeginstring As String = changestring.Substring(0,
islocation) ' Get the first part of the string to use at the end.
                If tempstring.Contains(".") Then
                    dotlocation = tempstring.IndexOf(".") + 1
                    'Search the location of the dot for truncation, + 1 for the number to start with
                    substringlength = tempstring.Length
                    Dim subsubstring As String = tempstring.Substring(dotlocation,
(substringlength - dotlocation))
                    Dim beginpartstring As String = tempstring.Substring(0,
dotlocation)
                    If OneDec_Rad.Checked = True And subsubstring.Length >= 1 Then
                        ' Rounding of numbers
                        Dim truncatestring As String = subsubstring.Substring(0,
1) ' Handles what to do when 1 decimal is selected
                        m.Text = finalbeginstring & beginpartstring &
truncatestring
                    End If
                    If TwoDec_Rad.Checked = True And subsubstring.Length >= 2 Then
                        Dim truncatestring As String = subsubstring.Substring(0,
2) ' Handles what to do when 2 decimal is selected
                    End If
                End If
            End If
        Next
    End If
End Sub

```

```

        m.Text = finalbeginstring & beginpartstring &
truncatestring
        End If
    If ThreeDec_Rad.Checked = True And subsubstring.Length >= 3
Then
        Dim truncatestring As String = subsubstring.Substring(0,
3) ' Handles what to do when 3 decimal is selected
        m.Text = finalbeginstring & beginpartstring &
truncatestring
        End If
    If FourDec_Rad.Checked = True And subsubstring.Length >= 4
Then
        Dim truncatestring As String = subsubstring.Substring(0,
4) ' Handles what to do when 4 decimal is selected
        m.Text = finalbeginstring & beginpartstring &
truncatestring
        End If
    If FiveDec_Rad.Checked = True And subsubstring.Length >= 5
Then
        Dim truncatestring As String = subsubstring.Substring(0,
5) ' Handles what to do when 5 decimal is selected
        m.Text = finalbeginstring & beginpartstring &
truncatestring
        End If
    If OneDec_Rad.Checked = False And TwoDec_Rad.Checked = False
And ThreeDec_Rad.Checked = False And FourDec_Rad.Checked = False And
FiveDec_Rad.Checked = False Then
        Dim truncatestring As String = subsubstring.Substring(0,
2)
        m.Text = finalbeginstring & beginpartstring &
truncatestring ' Default function
        End If
    Else
        MsgBox("Nothing to truncate.", 0, "DDT")
    End If
Else
    End If
m.Checked = False
Next
Else
    MsgBox("No Keyfigures Present.", 0, "DDT")
End If
End Sub

Private Sub SaveCheck4_CheckedChanged(sender As Object, e As EventArgs) Handles
SaveCheck4.CheckedChanged ' Handles the cutoff function checkbox
    Static checkcounter As Integer = 0
    checkcounter = checkcounter + 1
    If checkcounter Mod 2 > 0 Then 'If it is checked, show the following items:
        MaxCutOff_Rad.Visible = True
        MinCutOff_Rad.Visible = True
    Else 'Else reset/hide the following items
        MaxCutOff_Rad.Visible = False
        MinCutOff_Rad.Visible = False
        MaxCutOff_Rad.Checked = False
        MinCutOff_Rad.Checked = False
    End If

End Sub

Private Sub MinCutOff_Rad_CheckedChanged(sender As Object, e As EventArgs) Handles
MinCutOff_Rad.CheckedChanged
    Static checkcounter As Integer = 0

```

```

checkcounter = checkcounter + 1
If checkcounter Mod 2 > 0 Then 'If it is checked, show the following items:
    'MinEqu_Rad.Visible = True
    MinGreaEqu_Rad.Visible = True
    MinGrea_Rad.Visible = True
    MinCutOff_Text.Visible = True
Else 'Else reset/hide the following items
    'MinEqu_Rad.Visible = False
    MinGreaEqu_Rad.Visible = False
    MinGrea_Rad.Visible = False
    'MinEqu_Rad.Checked = False
    MinGreaEqu_Rad.Checked = False
    MinGrea_Rad.Checked = False
    MinCutOff_Text.Visible = False
End If
End Sub

Private Sub MaxCutOff_Rad_CheckedChanged(sender As Object, e As EventArgs) Handles
MaxCutOff_Rad.CheckedChanged
    Static checkcounter As Integer = 0
    checkcounter = checkcounter + 1
    If checkcounter Mod 2 > 0 Then 'If it is checked, show the following items:
        'MaxEqu_Rad.Visible = True
        MaxSmaEqu_Rad.Visible = True
        MaxSma_Rad.Visible = True
        MaxCutOff_Text.Visible = True
    Else 'Else reset/hide the following items
        'MaxEqu_Rad.Visible = False
        MaxSmaEqu_Rad.Visible = False
        MaxSma_Rad.Visible = False
        'MaxEqu_Rad.Checked = False
        MaxSmaEqu_Rad.Checked = False
        MaxSma_Rad.Checked = False
        MaxCutOff_Text.Visible = False
    End If
End Sub

Private Sub ConErrMax_Rad_CheckedChanged(sender As Object, e As EventArgs) Handles
ConErrMax_Rad.CheckedChanged
    Static checkcounter As Integer = 0
    checkcounter = checkcounter + 1
    If checkcounter Mod 2 > 0 Then ' If the button is checked
        Formula.Visible = True ' Show the formula
        MinForm.Visible = True
        MinLabel.Visible = True
    Else
        Formula.Visible = False
        MinForm.Visible = False
        MinLabel.Visible = False
    End If
End Sub

Private Sub Showabsconerr_Click(sender As Object, e As EventArgs)
    Dim errorarray As Array
    Dim n As TreeNode
    Dim m As TreeNode
    Dim mindexarray(1)
    Dim indexcounter As Integer = 0
    errorarray = Me.CSVarray
    Dim ncounter As Integer = 0
    Dim differencenamearray As Array
    differencenamearray = Me.differencenamearray

```

```

    If csvclicked = True And Me.errorgraphpushed = False Then 'If a csvfile has
    been opened and the error graph has not yet been shown on screen
    Dim arraylength As Integer = errorarray.GetLength(0)
    Dim differencearray(arraylength)
    For Each n In CSVOfflineTreeView.Nodes 'loop through all nodes
    If n.Checked = True Then
    ncounter = ncounter + 1 'Count all checked nodes
    End If
    Next

    If ncounter > 2 Then 'if more than two nodes are checked, give a
warning.
    MsgBox("Please select only two nodes in the treeview.", 0, "DDT")
    ElseIf ncounter = 2 Then

    Chart2.Series.Add(Me.currentserieamount) ' Add a new serie, which will
contain the error graph

    For Each m In CSVOfflineTreeView.Nodes
    If m.Checked = True Then 'loop through all nodes
    mindexarray(indexcounter) = m.Index 'Fill an array with the
values on which places the treeview are checked
    indexcounter = indexcounter + 1
    End If
    Next

    For z = 0 To 1
    differencenamearray(z) = namearray(mindexarray(z))
    Next
    For i = 0 To arraylength - 1
    differencearray(i) = Abs(errorarray(i, mindexarray(0)) -
errorarray(i, mindexarray(1))) ' The error is the difference between the actual value
and the wanted value. Therefore a subtraction is used.

    Next
    For j = 0 To arraylength - 1

    Chart2.Series(Me.currentserieamount).Points.AddXY(Abs(errorarray(j, xaxis)),
Abs(differencearray(j))) ' Add the relevant datapoints to plot the error
graph

    Chart2.Series(Me.currentserieamount).ChartType =
DataVisualization.Charting.SeriesChartType.Line 'Make a line-style
graph instead of a bar-style
    Chart2.Series(Me.currentserieamount).LegendText = "Error Graph"
    Next
    Me.globaldifferencearray = differencearray
    Me.errorgraphpushed = True
    Me.currentserieamount = Me.currentserieamount + 1
    Else
    MsgBox("Select two nodes from the treeview.", 0, "DDT")
    End If

    ElseIf csvclicked = True And Me.errorgraphpushed = True Then
    MsgBox("The error graph is already shown.")
    Else
    MsgBox("Please select a CSV file first.", 0, "DDT")
    End If

End Sub

Private Sub Button2_Click(sender As Object, e As EventArgs) Handles Button2.Click
    Dim percentarrayone(CSVarray.GetLength(0))
    Dim percentarraytwo(CSVarray.GetLength(0))
    Dim errarray(CSVarray.GetLength(0))

```

```

Dim lbl(3) As Label

lbl(1) = New Label
lbl(2) = New Label
lbl(3) = New Label

LegendGroup.Controls.Add(lbl(1))
LegendGroup.Controls.Add(lbl(2))
LegendGroup.Controls.Add(lbl(3))

Chart2.Series.Add(0)
Chart2.Series.Add(1)
Chart2.Series(0).ChartType = DataVisualization.Charting.SeriesChartType.Line
Chart2.Series(1).ChartType = DataVisualization.Charting.SeriesChartType.Line
Chart2.ChartAreas(0).CursorX.IsUserSelectionEnabled = True
'Let the user select a region to zoom into with the mouse on the X-axis
Chart2.ChartAreas(0).CursorY.IsUserSelectionEnabled = True
'Let the user select a region to zoom into with the mouse on the Y-axis
'Chart2.Series(0).LegendText = "Valve Position"
'Set the legend text to the correct relevant name
'Chart2.Series(1).LegendText = "Valve Setpoint"
For t = 0 To CSVarray.GetLength(0) - 1
    percentarrayone(t) = (CSVarray(t, 43) * 100) / 13824
    percentarraytwo(t) = (((CSVarray(t, 19) * 2000) / 27648) - 1000) / 10
Next

For k = 0 To CSVarray.GetLength(0) - 1
    Chart2.Series(0).Points.AddXY(CSVarray(k, 1), percentarrayone(k))
    Chart2.Series(1).Points.AddXY(CSVarray(k, 1), percentarraytwo(k))
Next

For j = 0 To CSVarray.GetLength(0) - 1
    errarray(j) = Abs(percentarrayone(j) - percentarraytwo(j))
Next

Chart2.Series.Add(2)
Chart2.Series(2).ChartType = DataVisualization.Charting.SeriesChartType.Line
' Chart2.Series(2).LegendText = "Absolute Valve Control Error"
For i = 0 To CSVarray.GetLength(0) - 1
    Chart2.Series(2).Points.AddXY(CSVarray(i, 1), errarray(i))
Next
Chart2.Series(0).XValueType = ChartValueType.Time
Chart2.Series(1).XValueType = ChartValueType.Time
Chart2.Series(2).XValueType = ChartValueType.Time
Chart2.ChartAreas(0).AxisX.Title = "Time [Hours:Minutes]"
Chart2.ChartAreas(0).AxisY.Title = "Position [%]"
Chart2.ApplyPaletteColors()
lbl(1).Text = "Valve Position"
lbl(1).ForeColor = Chart2.Series(0).Color
lbl(2).Text = "Valve Setpoint"
lbl(2).ForeColor = Chart2.Series(1).Color
lbl(3).Text = "Absolute Valve Control Error"
lbl(3).ForeColor = Chart2.Series(2).Color
lbl(1).Size = New Size(400, 13)
lbl(2).Size = New Size(400, 13)
lbl(3).Size = New Size(400, 13)
lbl(1).Location = New Point(6, 33) 'Give the correct location of the new label
lbl(2).Location = New Point(6, 46) 'Give the correct location of the new label
lbl(3).Location = New Point(6, 59) 'Give the correct location of the new label
End Sub

```