Development of a Design Master Tool for Upcoming Wärtsilä Engines

Daniel Öster

Bachelor's thesis

Automation Technology

Vaasa 2015

BACHELOR'S THESIS

Author:	Daniel Öster
Degree Programme:	Electrical Engineering, Vaasa
Specialization:	Automation
Supervisors:	Susanne Österholm

Title: Development of a Design Master Tool for Upcoming Wärtsilä Engines

27.4.2015 | 23 pages | 2 appendices

Summary

This Bachelor's thesis was done for Wärtsilä at the department "Ship Power Engines, Research & Development" situated in Vaasa. The goal of this thesis was to create a design tool for simplifying the generation of partitioning lists. Some of the requirements issued were semi-automatic updating of information sources, export functionality for completed lists and also a concept study on how all this could be improved in the future. The end result is a tool based on Microsoft Excel, with functions coded in Visual Basic for Applications.

Language: English Key words: Wärtsilä, Excel, Design tool, VBA, UNIC

OPINNÄYTETYÖ

Tekijä:	Daniel Öster
Koulutusohjelma ja paikkakunta:	Sähkötekniikka, Vaasa
Suuntautumisvaihtoehto:	Automaatiotekniikka
Ohjaaja:	Susanne Österholm

Nimike: Design Master -työkalun kehitys tuleville Wärtsilän moottoreille

27.4.2015 Sivumäärä 23 Liitteet 2

Tiivistelmä

Tämä lopputyö on tilattu Wärtsilän Ship Power Engines, Research & Development -osastolle Vaasassa. Työn päämääränä oli luoda suunnittelutyökalu partitiontilistojen suunnittelutyön helpottamiseksi. Työkalulle asetetut vaatimukset olivat muun muassa tietolähteiden semi-automaattinen päivitys, valmiiden listojen vienti PDF-muotoon ja myös tutkimus siitä, miten näitä prosesseja pystyttäisiin tulevaisuudessa parantamaan. Lopputulos on työkalu, joka perustuu Microsoft Exceliin, ja johon on lisätty VBA:lla ohjelmoituja toimintoja.

Kieli: Englanti Avainsanat: Wärtsilä, Excel, Suunnittelutyökalu, VBA, UNIC

EXAMENSARBETE

Författare:	Daniel Öster
Utbildningsprogram och ort:	Elektroteknik, Vasa
Inriktning:	Automationsteknik
Handledare:	Susanne Österholm

Titel: Utveckling av ett Design Master-verktyg för kommande Wärtsilämotorer

27.4.2015	23 sidor	2 bilagor

Abstrakt

Detta examensarbete gjordes åt Wärtsilä på avdelningen Ship Power Engines, Research & Development i Vasa. Målet med arbetet var att skapa ett designverktyg för att underlätta och förenkla genereringen av partitioneringslistor. Några av kraven på verktyget var semi-automatisk uppdatering av informationskällor, export av färdiga listor till PDF och även en undersökning på hur detta skulle kunna förbättras i framtiden. Slutresultatet blev ett verktyg baserat på Microsoft Excel, med funktioner programmerade i Visual Basic for Applications.

Språk: Engelska Nyckelord: Wärtsilä, Excel, Designverktyg, VBA, UNIC

Table of Contents

Abbreviations

UNIC = UNIfied Control

UNITool = Wärtsilä engine software configurator tool

CR = Common Rail

DF = Dual Fuel

SG = Spark ignited Gas engine

VBA = Visual Basic for Applications

ISO code = Standardized instrument code and description

CSV = Comma Separated Value file

USC = Unic Standard Component list

SID = Standard Instrument Database

RFC = Request for change

PME = Project Management & Engineering

IDM = Internet Document Manager

1 Introduction

Wärtsilä is a global corporation with its headquarter situated in Finland. It consists of 200 locations scattered across 70 countries and it employs more than 17,700 workers as of 2014. The company divisions Ship Power and Power Plants specialize in complete lifecycle solutions for the marine and power generation market. Wärtsilä also has a division called Services. It aims to provide everything from spare parts, field service, commissioning to environmental solutions. Wärtsilä is listed on NASDAQ OMX. [1]

Figure 1. Wärtsilä logo [2]

1.1 Background

I first became a trainee at Wärtsilä in 2012 for a summer job at the department “Ship Power Engines, Research & Development”, situated in Vaasa. My daily tasks there were making configuration packages and maintaining these configurations for laboratory engines. What the configuration does is that it tells the software package used on the engine how many cylinders are present, and how all the instruments are wired. Troubleshooting errors caused by bad configurations were also a common practice here. My contract got renewed in 2013, and I’ve worked part time in between summer jobs. In the middle of the summer 2014, I attended a workshop with the topic “UNIC design tool”. The outcome of the workshop was an opportunity for me to start my Bachelor’s thesis. The work was to develop a tool to simplify the way of working with partition lists. After my current contract ended I was hired to develop this tool.

1.2 Thesis objectives

The timespan for this thesis spans from Q3 2014 to Q2 2015.The end goal is a tool that simplifies making device/partitioning lists for the UNIC system. The tool should also semi-automatically fetch the newest available instrument information from external sources. To make the tool more compatible with other in-house tools, it was decided that it should be based on a Microsoft Excel template, running all specialized features as VBA code. Figure 2 shows how the Design Master’s import/export features should function.

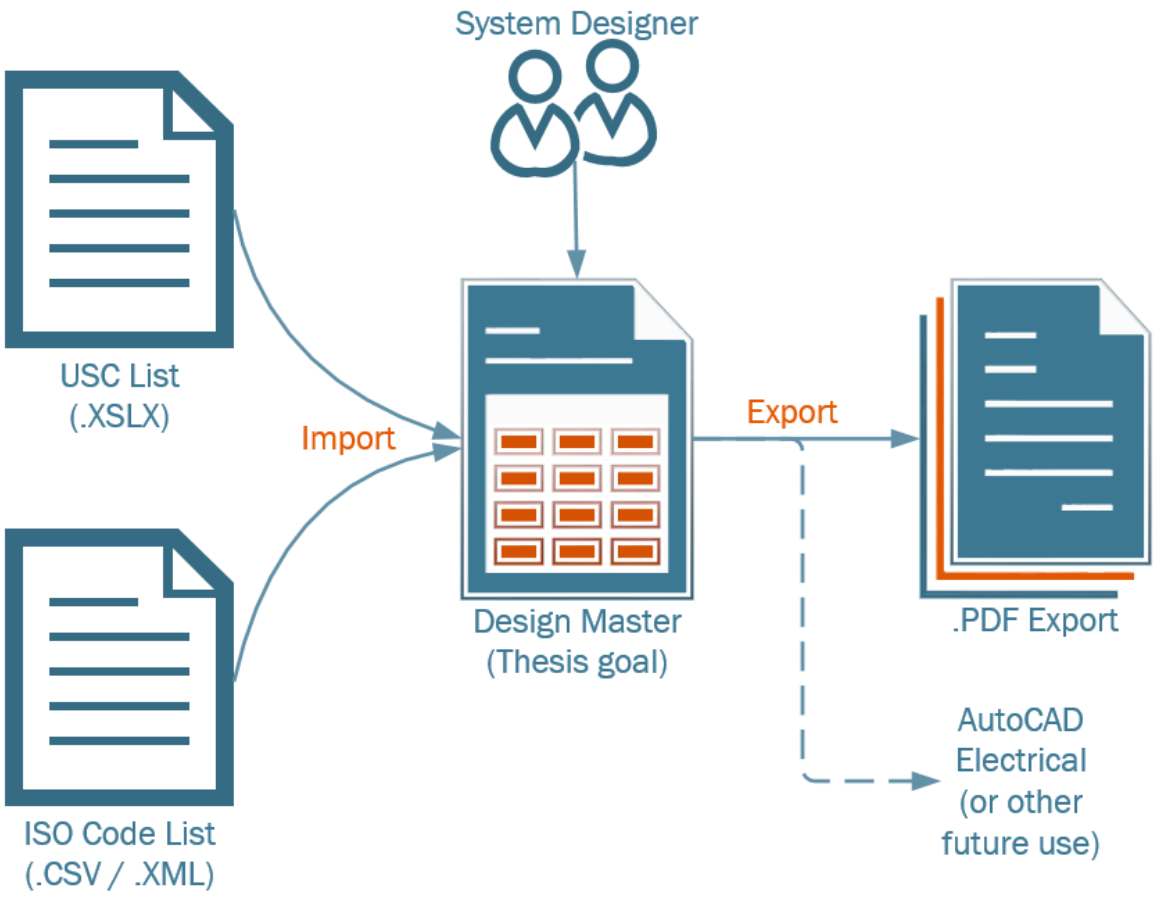

Figure 2. Data flow visualized. Design Master template semi-automatically updates data sources (USC and ISO) with VBA. It can also export the completed design as a .PDF file

Semi-automatic updating implies that the user may initiate an update of the data sources via a macro equipped button, and must manually confirm the revision he wishes to update to. Fully-automated updating would fetch the newest revision directly from the database, but the current SIDEX database doesn't support this type of queries yet. This functionality would be possible to add further down the line when the database matures.

The tool is by its nature a short term solution, which will be used for a few years until a better alternative is available. While AutoCAD electrical matures, the need for a better alternative is growing fast. Part of the thesis was also to plan for a future alternative.

2 Theory

2.1 USC and ISO lists

The Unic Standard Component (USC) list contains instruments used on Wärtsilä engines. This list goes into detail on what make and model the instruments are, and technical drawing links as well. The USC list is an .xslx file maintained on the Wärtsilä intranet.

The ISO code list contains code abbreviations and their descriptions. This list also contains min/max values for communication protocols, hardware limits and much more. This task will only focus on the abbreviations and the code descriptions. An example of an ISO code would be TE101. The TE part defines it as a temperature associated code, and the 101 number defines it as fuel output. There are thousands of ISO codes, the list constantly increasing with time. The ISO codes are maintained in the SID database and can be fetched from this database via a Java applet.

The Design Master needs these two information sources to automate and autofill content when creating a partitioning list. This reduces manual labour. Handling the revision number for these data sources is also crucial. This functionality is described in chapter 4.1.

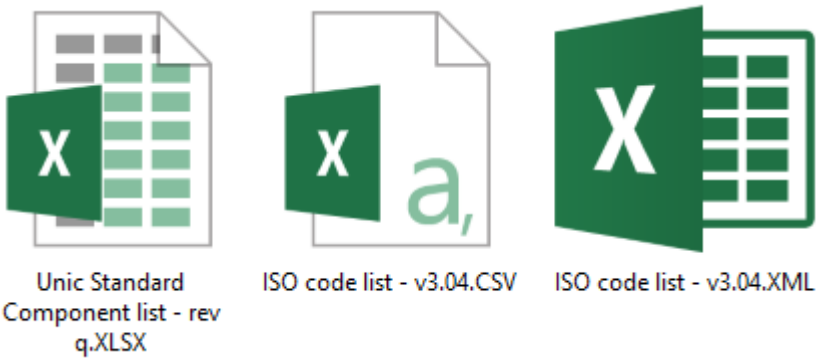

Figure 3. Information sources and their formats. Note that ISO codes can be in .CSV or .XML format

2.2 UNIC

UNIC is Wärtsilä's engine automation system. It stands for **UNI**fied **C**ontrols – "One system for one Wärtsilä". Depending on the complexity of the engine, different levels of automation are required. By having a unified system, Wärtsilä can optimize reliability, flexibility and cost efficiency. The same architecture can be applied to all engines, regardless of fuel setup, amount of cylinders or rated engine output. Especially gas engines have paved the way for more electronics and adaptive software. Diagnostics and fault tracing also become easier with a standardized system.

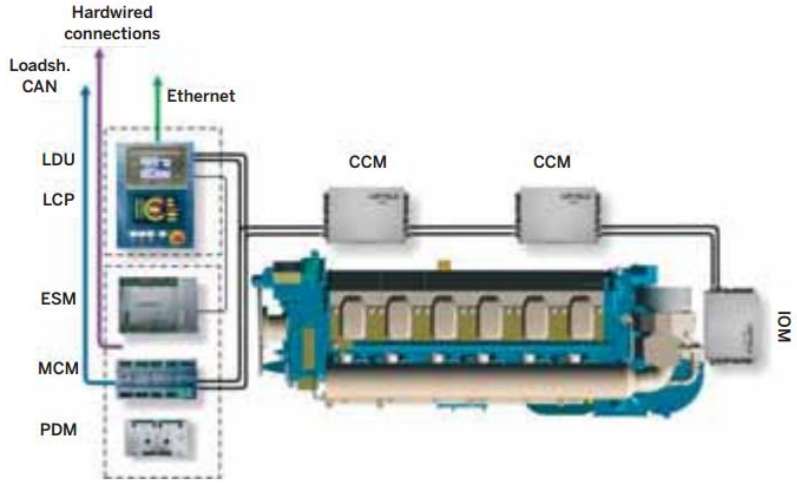

Figure 4. UNIC Bus design and main components. [3]

The engine control system on a Wärtsilä engine consists of different modules connected with a dual CAN bus for redundancy. All these modules have sensors, actuators, injector outputs etc. wired to appropriate hardware channels. These channels are in turn named after their instruments' intended functions, e.g. Analog Input X, Digital Output Y etc. They can be further broken down into pin numbers, for instance X25-1, X25-2 etc. The main concern when creating a partitioning list is connecting the instruments to hardware channels. Factors like but not limited to cable length, bank division, previous installations or specialized instruments, all affect the planning phase of the hardware partitioning. [3]

2.3 Partitioning / Device lists

The device and partitioning lists contain information on what instruments are used on an engine, and where they are connected. These lists can in turn be used to create drawings for electrical connections or used as templates when creating configuration packages for the engine automation system. A partitioning list shows hardware channel, ISO-code, ISO-code description, Material nr, Drawing nr, Article and a few additional descriptions. A device list filters away everything that doesn't have a material number and sorts the system from A – Z. All engine installations have their own lists, but the lists created with this tool will mainly be for upcoming engine types. The current way to handle these lists are in spreadsheet format for speeding up automated processes but can also come as a regular text file to improve readability.

2.4 AutoCAD Electrical

AutoCAD Electrical 2015, a product made by Autodesk, was selected as the preferred tool within Wärtsilä when it comes to creating electrical drawings and schematics. To make AutoCAD more efficient for Wärtsilä's purpose, custom blocks and functions containing the models and naming schemes for the UNIC system have been created. During the test runs with it, software limitations were noted and reported directly to Autodesk, and the application will hopefully improve with future releases. Getting the link between partitioning lists and AutoCAD is not in the scope for this thesis, but it is relevant for future development. Achieving this link would imply quicker creation of content and less re-entering of already existing information.

Figure 5. AutoCAD Electrical Logo [7]

3 Visual Basic for Applications

Visual Basic for Applications, a product made by Microsoft, is an implementation of their programming language Visual Basic. VBA is included in every Microsoft Office application. VBA is an event-driven programming language determined e.g. by mouse clicks, key presses or messages from other programs. The ability to record macros, to turn complex manual labour and formatting into easily understandable code is one of the main selling points for this programming language. VBA can also import data from external sources and files, a functionality which is crucial for the Design Master. VBA has been used in many custom tools inside Wärtsilä, mainly because of its simplicity, but also because it is readily available on every workstation within the organization.

3.1 Microsoft Excel & VBA

Microsoft Excel, current version 2013, is a spreadsheet application. All workstations within Wärtsilä come with the Microsoft Office package preinstalled, with Excel being one of the members in this package. During the planning stage for the thesis, the decision whether to use Access or Excel had to be made. They both contain VBA capability in their standard form. Access can be more useful when it comes to complex queries or bigger projects that won't fit in a single spreadsheet. [8] Since a typical list created with the Design Master will only contain around 350 entries, it was decided that Excel will be more than enough for the task.

3.2 Modules and subs

All code written in the VBA language is put into modules. These can be edited with the Visual Basic Editor. These modules can contain subs and functions. A sub is code that executes a manipulation with objects. A function is like a sub but returns a value when it has finished. Figure 6 contains an example of a sub and a function. [8]

```
Sub Arithmetics()
    Result = 1 + 3 - 2
    MsgBox "The result is " & Result
End Sub

Function Subtract(num1, num2)
    Subtract = num1 - num2
End Function
```

Figure 6. Sub and function example

3.3 Recording macros

An example on how to record a macro can be seen in figure 7. By first clicking the Record Macro button, under the Developer tab, the recording starts. The user can then manually step through the task that needs automating. In this case, the user starts by selecting the A column, right-clicking it and selecting hide. After that the recording is stopped by pressing the Stop Recording button also found under the Developer tab. The generated code can then be viewed and saved.

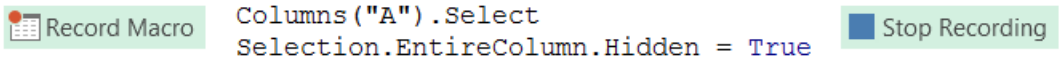

Figure 7. Macro record and resulting code

By generating code this way, no previous coding experience is required. It might not always be the most resource efficient way to complete the task, but it's a good place to start. After generating the code, it can be placed in a module and the functionality can be verified easily by debugging it with the F8 key, one line at a time.

3.4 Declaring data types

A variable declaration system is essential to coding languages, and VBA is very flexible in this aspect. Commonly used data types in many languages are Integer, Double and Long, but VBA has a special

datatype called Variant. This datatype can contain all types of data. While convenient, this can cause increased resource allocation. In a small script this is not a problem, but when applications become more complex, careful thought should be given to defining data types. Clearly defined data types also improve code readability. Figure 8 contains the data type list sorted by size, lowest first.

Type	Size in bytes	Description
Byte	1	Number between 0-255
Boolean	2	True(0) or False (-1)
Integer	2	–32,768 to 32,767
Long	4	–2,147,483,648 to 2,147,486,647
Object	4	Any object (ActiveX, Access or Class)
Single	4	–3402823E38 to –1.401298E–45
Currency	8	Monetary value between ±922,337,203,685,477.5808
Date	8	Date and time, Jan1. 100 to Dec.31 9999
Double	8	28-digit number,±1.79769313486232e308 to ±4.94065645841247e–324
Decimal	14	±79,228,162,514,264,337,593,543,950,335
String	String length	Set of characters from 1-65400
String (variable)	10 + length	Set of characters from 0-20E9
Variant (num.)	16	Any valid numeric data type up to the size of a double
Variant (char.)	22 +length	Stores any valid non-numeric data type or types larger than a double

Figure 8. Some of VBA's datatypes sorted by order of size [8]

An example on how the flexible variable declaration works in VBA.

```
Dim z
```

Dim stands for dimension, z is the variable. The declaration above would assign the variant data type to the z variable, resulting in maximum flexibility for storing data. The z variant would then consume 16 bytes of memory. Suppose we know that the z variable will only contain integers between 0 – 255. It is then smarter to declare z like this:

```
Dim z As Byte
```

This will cut down memory usage from 16 bytes to 1 byte, which is a 94% decrease in resource usage. Another benefit is that it becomes easier to debug.

3.5 Form Controls

One of the advantages of automating with VBA is the easy linking between scripts and form controls. A drag-n-drop system makes planning the worksheet layout a breeze. Figure 9 is an example on how to make the connection between a script and a button.

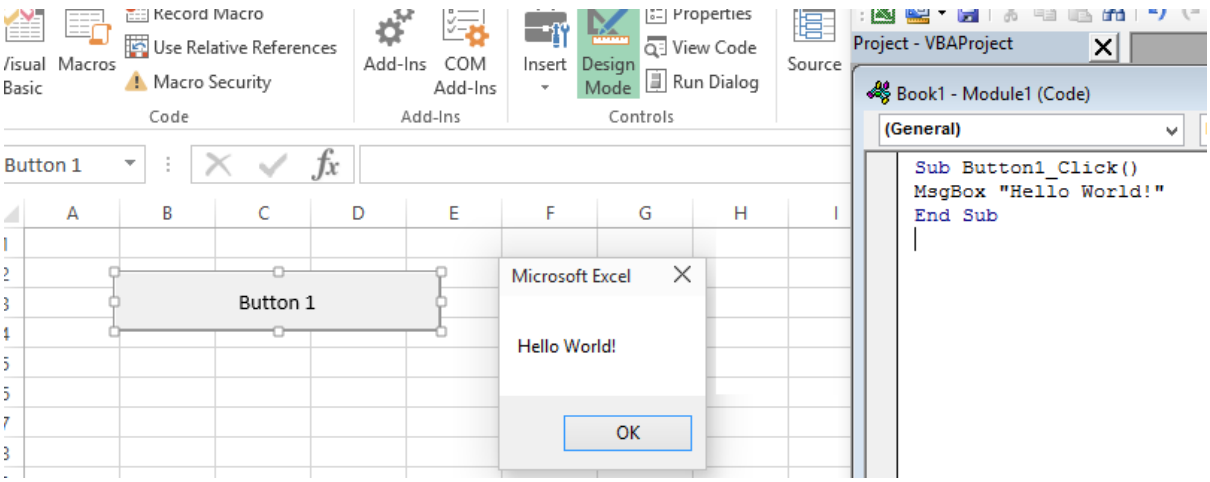

Figure 9. Linking a command button control to VBA script

3.6 ActiveX

ActiveX is a software framework and an alternative to the Java programming language. ActiveX components can be more tailored than standard Excel form controls. For instance their appearance, fonts or behaviour can be customized. ActiveX can only be run on Microsoft Windows operating systems. In Excel, a few ActiveX components are included by default and can be used with or without VBA. [8]

Figure 10. Option button, Form controls compared to heavily modified ActiveX counterpart

The negative side of ActiveX is that you rely on a third party plugin for your functions. During the development of the Design Master, ActiveX components were used due to their more appealing aesthetics. After the infamous December 2014 Windows update launched, KB2553154, ActiveX components stopped working altogether in VBA. This was patched within a month but was enough to switch development over to standard Excel form controls to avoid similar incidents in the future.

3.7 Error handling

Error handling is also an important part when designing an application. Most VBA coding neglects this part and when things go wrong, very cryptic messages appear to the end-user. Sadly, you don't have much to choose from in VBA, it is not as polished as other programming languages when it comes to error handling, "On Error" combined with "GoTo" being the most common remedy. An implementation of this can be seen in figure 11.

```
On Error GoTo errHandler

'Main code here

exitHandler: 'When completed, exit the function
    Exit Function
errHandler: 'On error, jump here
    MsgBox "Script has failed, aborting" 'Tell the user an error occured
    Resume exitHandler
End Function
```

Figure 11. Code snippet for terminating erroneous scripts gracefully in VBA.

4 Development, problems and solutions

To keep the information easily readable and manageable, each entity in the partitioning list is added as a row in the main worksheet. To start things off, a toolbar was placed on top of the worksheet. A few different clickable form controls were placed on this toolbar. The clickable elements are then linked to macros. The toolbar is always on top when scrolling. This is achieved by using the "Freeze panes" function.

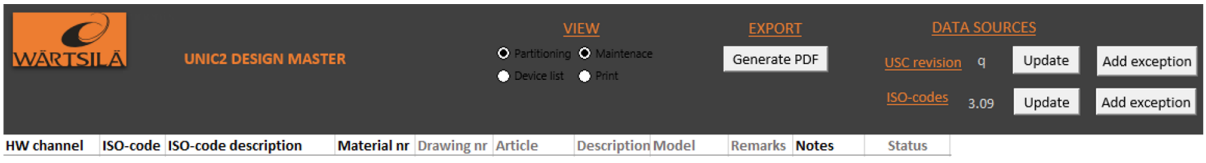

Figure 12. Completed main application toolbar with form controls

4.1 Importing data sources

Semi-automatic update of data sources was one of the first requirements issued at the start of the development. It is also one of the more challenging tasks, mainly because there are three different file types to handle. Macros to import data from .XLSX (USC), .XML/.CSV (ISO) were constructed.

Figure 13. Updateable data sources. Current revision number next to update button.

To kick off this task, four form buttons were placed on the toolbar. The update buttons are linked to individual update macros. The exception buttons are used to add exceptions to the data sources, a function that jumps to the end of the data source so the user can add his nonstandard instrument at the bottom of the list. This is to be used in the event that a nonstandard component is needed, and data sources haven't yet included it.

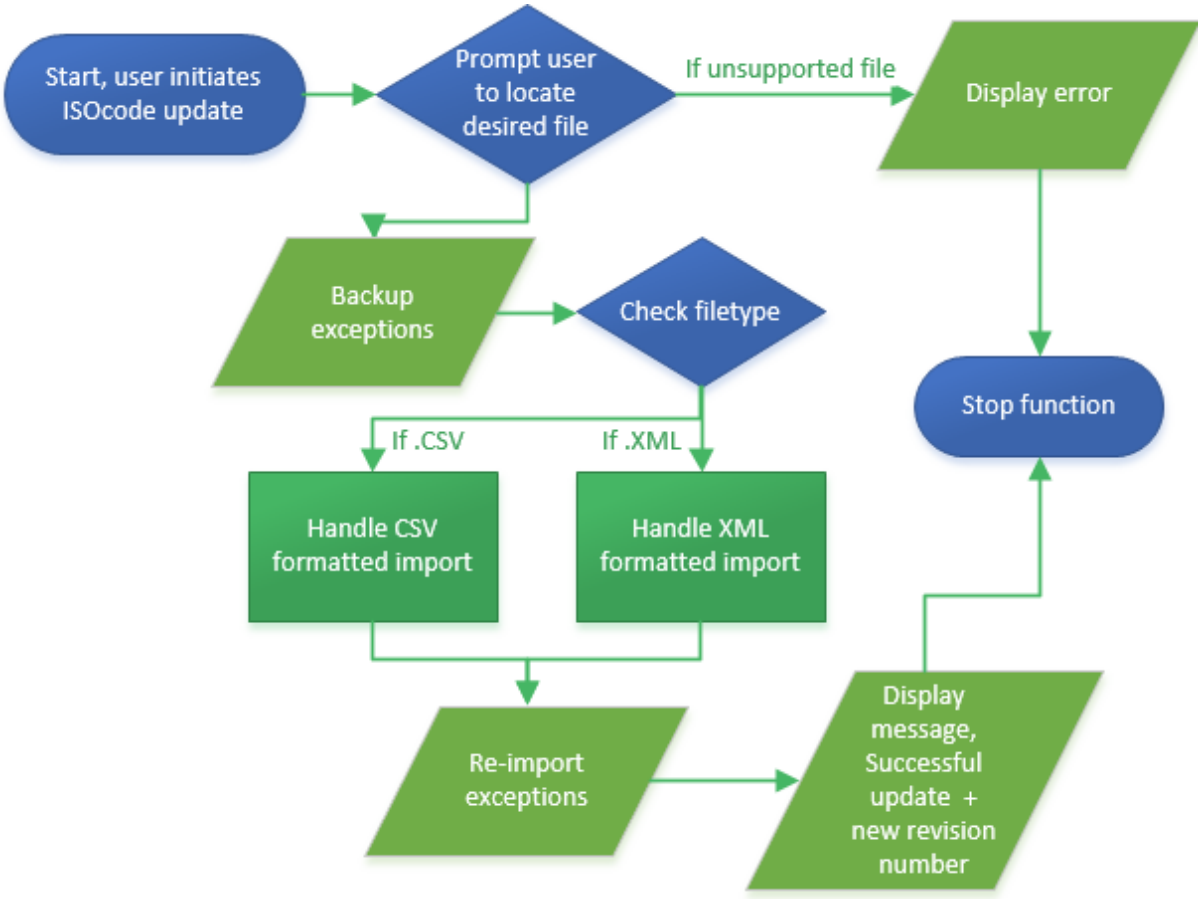

Figure 14. Flowchart for ISO-code import macro

The macro behind figure 14 is constructed like this. First, it asks the user to locate the file he wishes to import. The macro checks if the selected file is in the right format and determines the correct approach. For instance, ISO codes can be updated either as an .xml or as a .csv file. A backup of current exceptions is created, on a temporary sheet. The new data is then copied over to the Design Master document, overwriting the entire page. Finally, the exceptions get reimported at the end of the macro. If the import should fail, the error handler takes over and ends the macro gracefully. The macro behind the very similar USC import can be viewed in appendix #1.

An issue when it comes to importing data is that the screen flickers a lot when switching between sheets, selecting data and jumping between the two open documents. The solution is to disable refreshing the screen while the macro is running. Another benefit here is that it doubles as a speed optimization. This optimization is applied at the beginning and end on all macros that have any tendency to flicker.

```
Application.ScreenUpdating = False
'Main code here
Application.ScreenUpdating = True
```

Figure 15. Code snippet for disabling screen updating

However, caution should be used when using this snippet. If the macro ends prematurely and screen updating isn't reverted to true, excel will appear frozen and won't respond to any input. This can be disastrous for the end user, so the re-enabling of screen updating must be coded into the error handler.

If the planned macro is going to take a long time to execute, disabling screen refreshing will make it look like Excel has stopped responding. Another approach to this inconvenience would be to use the status bar in the lower left corner to inform the user what stage of completion the code is at. The macros for the Design Master are fairly quick, so disabling screen updating is enough for the task.

```
Application.StatusBar = "Running code... Step 1/2"
'Main very long code here
Application.StatusBar = "Running code... Step 2/2"
'Continuation of long code
Application.StatusBar = False
```

Figure 16. Code snippet for displaying progress with StatusBar code

4.2 Lookups from data sources

The tool automatically fetches ISO code descriptions, drawing numbers, article descriptions, part status etc. from data sources. The information is located in two sheets attached to the main Design Master document. More than half of the partition/device list is automatically generated with these “vlookup” functions.

ISO-code	ISO-code description
	Fuel system and position
LS107A	FO leakage, dirty fuel FE,
LS107B	FO leakage, dirty fuel FE,

```
=IFERROR(VLOOKUP(B9,'ISO codes'!A$5:B$3600,2,FALSE), "-")
```

Figure 17. Vlookup for ISO Code description field

If a *vlookup* fails, there is a way to avoid “N/A” showing up instead of the failed fetched value. An *IfError* wrapped around the *vlookup* makes the partitioning list a much more visually appealing list. This can also be seen in figure 17.

4.3 Filter by system type

Filtering the list is probably one of the most important feature in the whole spreadsheet. Lack of a good way to filter columns makes the tool appear unstructured. This is the idea behind the *System* column, as seen in figure 18.

ISO codes in the UNIC system are labelled in hundreds, ranging from 100-900 depending on what type of system they belong to (i.e. Charge Air System = 600, Gas System = 900). To quickly get a better overview of the list, Excel’s built-in filter function needs to format the whole list by looking at the system number. The system number column contains a number ranging from 1-9 and is automatically derived from the adjacent ISO code. This column requires no user input.

ISO-code	System Number
LS107A	=MID(B9;MIN(IF
LS107B	1

```
=MID(B9;MIN(IF(ISNUMBER(FIND({1;2;3;4;5;6;7;8;9;0};B9));FIND({1;2;3;4;5;6;7;8;9;0};B9)));1)
```

Figure 18. Function for extracting first digit. The two columns are ISO-Code and System Number.

Figure 18 looks at what number comes first in the ISO code, in the previous example, LS**1**07A (blue box) is the adjacent ISO code entered, and "1" is extracted to the system number column seen in the green function box above.

4.4 Filtering list with views

Sometimes a view of the entire partitioning list is wanted, but also the ability to switch to device list mode. The functionality to switch between these different modes is set up with option buttons on the toolbar and linking them to macros.

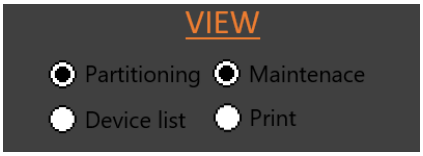

Figure 19. Different views section on toolbar

Another part of the view toolbar is the possibility to change between a print friendly mode, and a maintenance mode for regular usage of the tool. The print mode hides unnecessary tabs like system number, cylinder, PIP etc. This makes the spreadsheet more suitable for landscape printing without clipping the content. The document automatically enters print mode when exporting as pdf.

The macro switches the hidden true/false attribute on the columns depending on which view mode is desired. Screen updating is also disabled when this macro runs, a functionality which can be seen in figure 15.

4.5 DASLO lifecycle model

All instruments currently in use by Wärtsilä have a designated lifecycle status noted in the USC list. DASLO stands for **D**evelopment, **A**ctive, **S**upported, **L**imited and **O**bsolete. The target is to keep the automation systems in an Active status. After the Active lifecycle state follows the Supported state. The policy for this stage is to be able to provide new spare parts. Finally the instrument enters the Limited lifecycle status which implies that Wärtsilä can no longer guarantee availability of new spare parts. The length of these different lifecycles are confidential and cannot be included in this document. [4] In figure 20 the different stages are visualized:

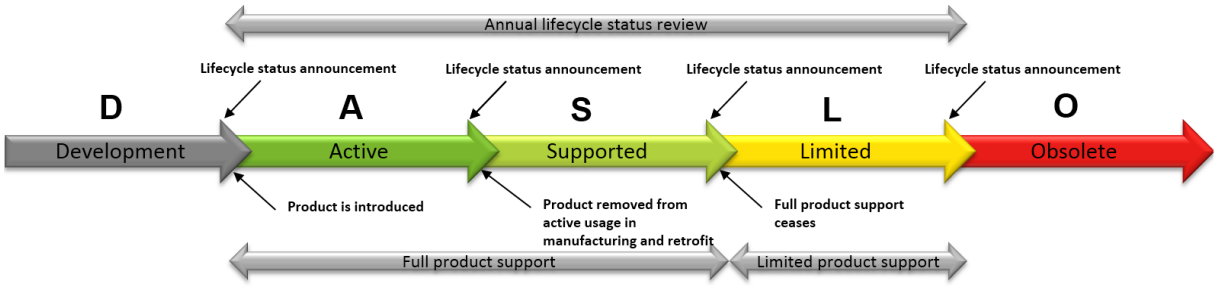

Figure 20. DASLO model from Wärtsilä powerpoint [4]

It is very convenient to be able to spot discrepancies in the planned hardware. The Implementation of this is a "vlookup" function that determines the lifecycle status for the planned hardware. This information is fetched from the USC sheet. A conditional formatting is applied to the whole column, which paints the cell to match the DASLO model colour. Note that "Obsolete" is striped to help distinguish it for colour-blind users.

Figure 21. The five different lifecycle models as they appear in the Design Master

4.6 Exporting as PDF

Hassle free sharing of these lists is achieved by exporting them as PDFs. In this way you only share the actual content and get a smaller file size. The PDF format is perfect for a human readable printout, guaranteed to have the same formatting no matter what device you open it with.

A macro is applied here, to quickly get the formatting right. The macro selects the "Print" view, and formats the whole document as a landscape template. The header sheet is put as the first page, and then the design master sheet follows. The export also suggests a filename according to current date & time. If the export should fail for some reason, the error handler takes over and ends the macro gracefully.

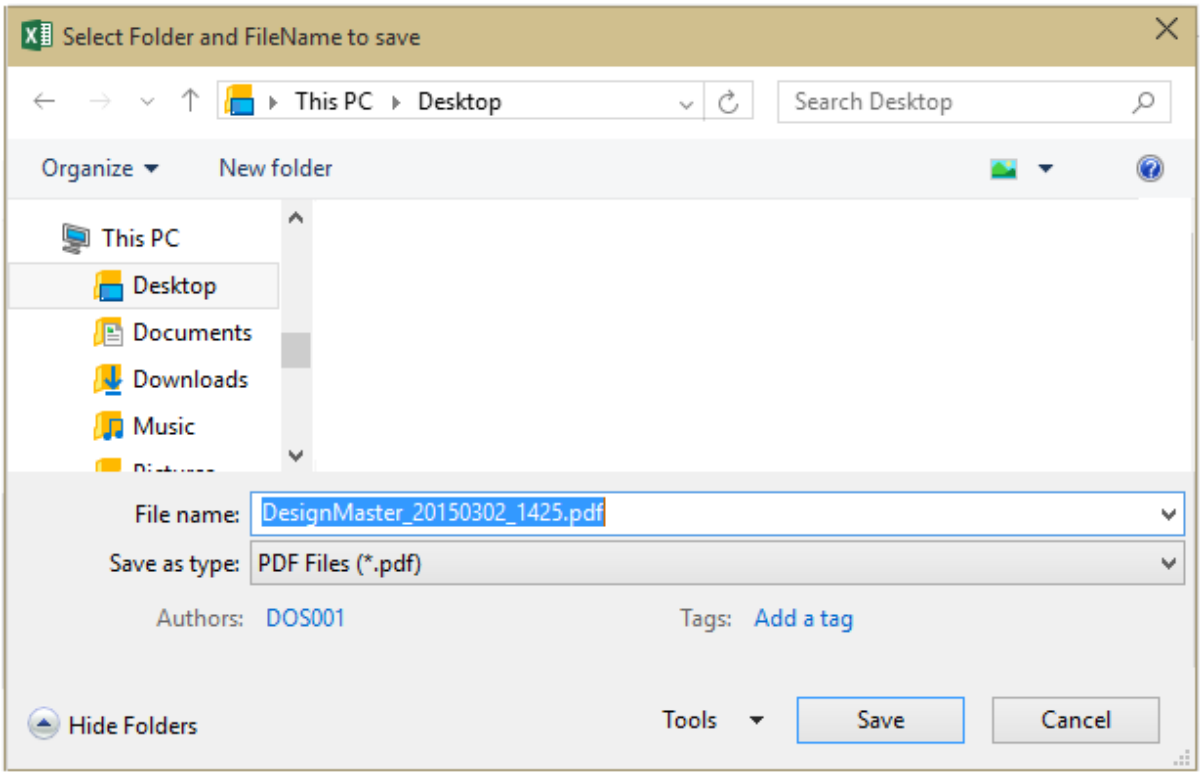

Figure 22. After pressing the export button, the PDF export window appears

5 Finished program overview and results

The application was put in use in Jan.2015. A draft for an upcoming engine type was created with the tool. After reviewing the work process with the end-user, corrective fixes for a few bugs were applied. An example of what the finished tool looks like, with dummy values for secrecy reasons, can be viewed in appendix #2.

5.1 Using the Design Master

The actual usage is straightforward. The columns with black titles require user input. The grey columns are automatically filled with *vlookups*. If a new row is entered, the functions from the adjacent grey cell can be copied to the newly generated row. The grey and black columns can be seen in appendix #2.

5.2 Feedback from users

The feedback sessions gathered good data from the users. These sessions were held monthly during development. A very good way to improve usability is to communicate directly with the end-users. An example of such an improvement would be the “HW Channel” lookup sheet, which from a user perspective speeds up design work significantly.

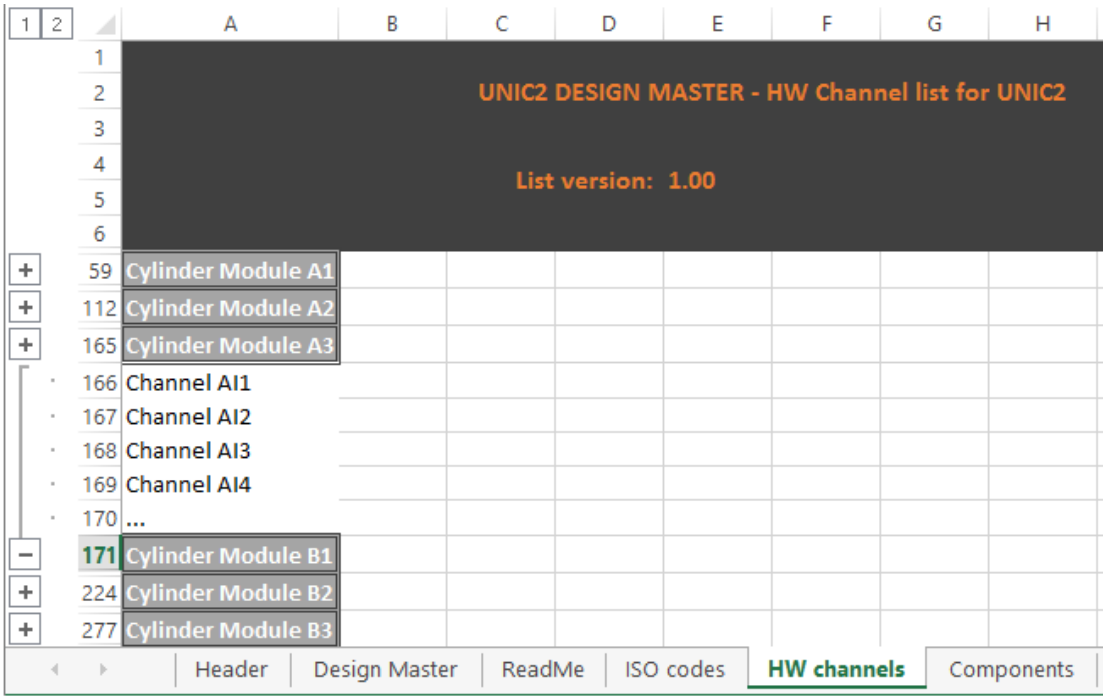

Figure 23. HW channel tab for vlookup, user initiative

The information for this tab was not available in this format, but it was created manually from looking at hardware diagrams. By using the built-in data validation function on the HW channel, duplicate entries are avoided.

6 Discussion and future plans

The practical part of this thesis was very interesting and also sufficiently challenging for a person with no in-depth understanding of Excel's programming language. I would like to thank my supervisor at Wärtsilä for believing in my ability to solve upcoming problems along the way.

A concept study on how this information flow should be handled in the future was also made as a part of this thesis. The Design Master is a short term solution and in the future, these lists are most likely automatically generated and imported/exported effortlessly from Unitool. This speeds up the process significantly and would be a huge benefit for both PME and Services. The thesis also sparked some RFC cases, mainly concerning the content in the USC list.

Continuous development and tweaking of this tool will continue to be part of my responsibility while it continues to develop. Although the code is well documented, it is always faster to have the person who made the code in the first place to troubleshoot it.

After a meeting with a senior design engineer from product support, Technical Services' mission became clear. The current way of working involves a lot of manual work when it comes to handling drawings and wiring diagrams. Electrical drawings made with AutoCAD should have as little user input as possible. Information should be fetched from already existing data. It is unnecessary to manually input hundreds of entries already existing in some other format into AutoCAD. The risk for user error increases when inputting hundreds of similar entries in a list. It became clear to me that the scope of this is not part of the original plan with the Design Master, but it can easily evolve into so much more.

6.1 Risk analysis

Handling the partitioning list in this way still poses a few risks. When relying mostly on automatic generation, combined with data validation on entered fields, you have a very small risk of spelling errors. The colour formatting is one way to highlight discrepancies, thus reducing the error rate. With these ways of working, the error risk is significantly smaller than with a manually generated list.

The tool itself has its toolbar protected with Excel's built-in function "Protect Sheet", which prevents any accidental deletion of crucial formulas or macro buttons.

Loss of data should not be of any concern with this tool. All documents are stored within Wärtsilä's intranet. The partitioning lists are no exception to this. This is a very fool-proof way to handle documents. This in combination with Excel's built-in AutoRecover and AutoSave functionality keeps information from being lost if there's a power outage or if Excel stops responding.

6.2 Interview with Unitool developer

This particular interview proved very fruitful with many good ideas.

Unitool can already generate a partitioning list based on the configuration. This list, however, is in a pretty unfriendly format seen from an automation point of view, currently exported in the OpenDocument format .odt. If this list were to become available as an .xlsx format, it would be easier to automate processes that need to fetch information from a partitioning list.

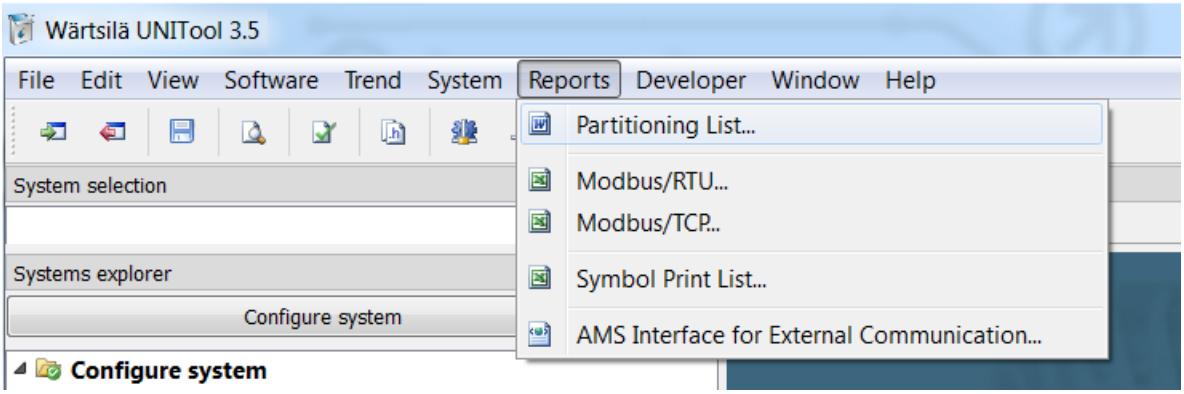

Figure 24. Unitool Partitioning List export function

Another future idea would be that Unitool would also have access to the USC information and include this in the partitioning list export. The USC list could then become a part of the current SID database, instead of a free floating document in IDM. [5]

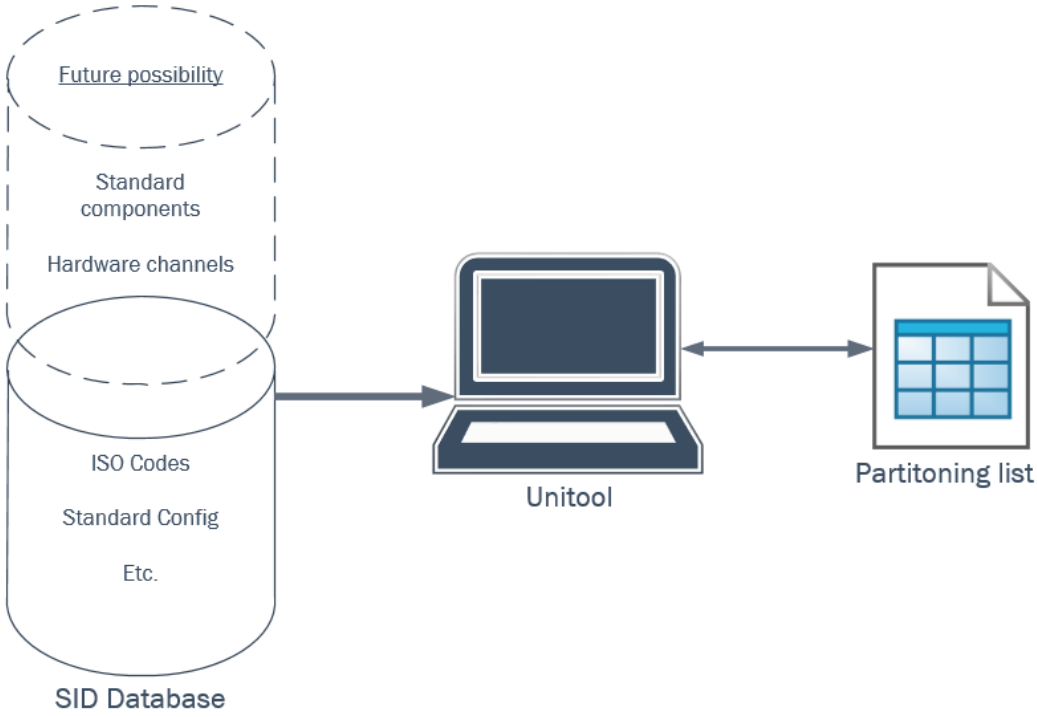

Figure 25. Possible information flow in future

When evaluating the planned scope for this thesis, it is safe to say that there is much more to be done when it comes to automating related tasks within Wärtsilä. The learning experiences at Wärtsilä have been invaluable, and I hope to be part of this development in the future.

7 References

Electronic sources

[1] "Products & Solutions, 2012" Only accessible from Wärtsilä intranet. http://compass.wartsila.com/productsandsolutions/Pages/Default.aspx (Retrieved 03.2015)

[2] "Wärtsilä's Financial Statements Bulletin January-December, 2014" http://www.wartsila.com/en/press-releases/wartsilas-financial-statements-bulletin-january-december-2014 (Retrieved 01.2015)

[3] UNIC Automation ABC, 2009. Wärtsilä internal document. (Retrieved 03.2013)

[4] DASLO Model, 2014. Wärtsilä internal document. (Retrieved 08.2014)

[5] SID Manual for Code Tool, 2010. Wärtsilä internal document. (Retrieved 01.2015)

[6] Standard Instrumentation List – rev o, 2012. Wärtsilä internal document. (Retrieved 10.2014)

[7] AutoCAD Electrical 2015 logo. http://www.autodesk.com/products/autocad-electrical/overview (Retrieved 12.4.2015)

Printed sources

[8] Mueller John Paul. (2007). *VBA for Dummies*. 5th Edition. ISBN: 978-0-470-04650-0

Meetings & Interviews

Meetings held at Wärtsilä related to this thesis:

Meeting	10.11.2014
Meeting	18.12.2014
Meeting	10.1.2015
Meeting	13.3.2015
Interview	27.3.2015
Meeting	8.4.2015

8 Appendices

Appendix 1: Example of VBA code, USC update macro

Appendix 2: Completed application overview

APPENDIX 1.

USC Update Macro

```
Function USCUpdate_Click()
'This sub updates the Unic standard component List'
'It first prompts the user to input a USC list (.xlsx format)
'Then it backups current USC exceptions, and imports the new USC data.
'Finally, the USC revision number is imported.

   Dim FilePath As String        'Filepath, ex. C:\\Filename.txt , defined by user input
   Dim USC_code_version As String  'String used for filename validation.
   Dim choice1 As String         'Error handling string for message box
   Dim DesignMaster As String     'Design master filename used when switching between windows
   On Error GoTo errHandler       'If macro fails, goto errHandler

      DesignMaster = ThisWorkbook.Name          'Establish current workbook name
                                'Ask user to locate the USC list'
      FilePath = Application.GetOpenFilename _
      (Title:="Enter the newest USC List .xlsx", _
      FileFilter:="Excel Files *.xlsx* (*.xlsx*),")

      If strFileToOpen = "False" Then           'Error message
      MsgBox "No file selected.", vbExclamation, "Error!"
      Exit Function
      Else
      End If

      USC_code_version = Right(FilePath, 20)
                                'USC list file validation based on filename

         If USC_code_version = "Components list.xlsx" Then
         MsgBox "Component list successfully located, will now be updated to this sheet"

        Else
                                'Error message if file is not found or broken'
        choice1 = MsgBox("USC List not found or invalid, make sure that 1. File path is correct 2. The
USC file name hasn't changed since last release 3. The export file is in .XLSX format. USC import
canceled.", vbOKOnly + vbCritical, "Invalid USC List file")
        Exit Function

        End If
        Application.ScreenUpdating = False     'Speed optimization enable (disables screen updating)
                                   'Backup custom USC exceptions, these are temporarily moved
          to the readme sheet
        Sheets("Components").Select
        Sheets("Components").Rows("1000:1200").Select
        Selection.Cut
        Sheets("ReadMe").Select
        Sheets("ReadMe").Range("A14").Select
        ActiveSheet.Paste
```

```
        Application.DisplayAlerts = False      'Alerts are disabled, to speed up the macro
                                               '(Save file prompt when closing disabled)
        Sheets("Components").Select            'Delete the old component list sheet'
        ActiveWindow.SelectedSheets.Delete

                                'Import components from USC list'
          Workbooks.Open (FilePath)
          Windows("UNIC Standard Components list.xlsx").Activate

                                'Import revision number to design master sheet
            Sheets("Title sheet").Range("A13").Select
            ActiveCell.Formula = "=KRONO_version"
            Sheets("Title sheet").Range("A13").Select
            Selection.Copy
            Windows(DesignMaster).Activate
            Sheets("Design Master").Select
            Range("O3").Select
            Selection.PasteSpecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks _
            :=False, Transpose:=False
            Windows("UNIC Standard Components list.xlsx").Activate
                                'Resume importing components

          Sheets("Components").Select
          Sheets("Components").Copy after:=Workbooks(DesignMaster). _
          Sheets(3)
          Windows("UNIC Standard Components list.xlsx").Activate
          Application.DisplayAlerts = False   'Alerts are disabled, to speed up the macro
          ActiveWorkbook.Close            'Close the workbook, we are done with it
          Application.DisplayAlerts = True    'Alerts re-enabled

                                'Import custom USC exceptions, back from the readme sheet
          Sheets("ReadMe").Select
          Sheets("ReadMe").Rows("14:214").Select
          Selection.Cut
          Sheets("Components").Select
          Sheets("Components").Rows("1000:1000").Select
          ActiveSheet.Paste
                                'Alert the user that USC List update completed'
          Sheets("Design Master").Select
          Application.ScreenUpdating = True'Speed optimization disable (screen updating re-enabled)
          MsgBox "USC List updated successfully"

exitHandler: 'When completed, exit the function
    Exit Function
errHandler: 'On error, jump here
    MsgBox "Could not update USC. Macro threw exception" 'Tell the user an error occured
    Resume exitHandler 'Jump to function exit

End Function
```

WÄRTSILÄ

UNIC2 DESIGN MASTER

VIEW
Partitioning Maintenace
Device list Print

EXPORT
Generate PDF
Generate XSLX

DATA SOURCES
USC revision q Update Add exception
ISO-codes 3,09 Update Add exception

HW channel	ISO-code		ISO-code description	Material n	Drawing r	Article	Descriptio	Model	Remark	Notes	Status	PIP
			Fuel system and position									
Analog Input 1	LS107A	1	FO leakage, dirty fuel FE, A-b	PAAF000000	DAAF000001	Sensor	1234	AA1		For DF only	Active	57
Analog Input 2	LS107B	1	FO leakage, dirty fuel FE, B-b	PAAF000001	DAAF000002	Sensor	1234	AA2		For DF only	Active	
Analog Input 3	OS163	1	Speed/load increase	PAAF000002	DAAF000003	Sensor	1234	5m flying lead			Active	
Analog Input 4	OS164	1	Speed/load decrease	PAAF000003	DAAF000004	Sensor	1234	5m flying lead			Active	
Analog Input 5	OT190	1	Analogue speed reference	PAAF000004	DAAF000005	Sensor	1234	5m flying lead			Active	
Analog Input 6	ST173	1	Engine speed 1	PAAF000005	DAAF000006	Sensor	1234	5m flying lead			Active	
Analog Input 7	ST174	1	Engine speed 2	PAAF000006	DAAF000007	Sensor	1234	5m flying lead			Active	
Analog Input 8	ST196P	1	Engine speed, primary	PAAF000007	DAAF000008	Sensor	1234	5m flying lead			Active	
Analog Input 9	ST196P-1A	1	Primary Speed signal A1	PAAF000008	DAAF000009	Sensor	1234	5m flying lead			Active	
Analog Input 10	ST196P-1B	1	Primary Speed signal B1	PAAF000009	DAAF000010	Sensor	1234	5m flying lead			Active	
Analog Input 11	ST196P-2A	1	Primary Speed signal A2	PAAF000010	DAAF000011	Sensor	1234	5m flying lead			Active	
Analog Input 12	ST196P-2B	1	Primary Speed signal B2	PAAF000011	DAAF000012	Sensor	1234	5m flying lead			Active	
Analog Input 13	ST196S	1	Engine speed, secondary	PAAF000012	DAAF000013	Sensor	1234	5m flying lead			Active	
Analog Input 14	ST196S-1A	1	Secondary Speed signal A1	PAAF000013	DAAF000014	Sensor	1234	5m flying lead			Active	
Analog Input 15	ST196S-1B	1	Secondary Speed signal B1	PAAF000014	DAAF000015	Sensor	1234	5m flying lead			Active	
Analog Input 16	ST196S-2A	1	Secondary Speed signal A2	PAAF000015	DAAF000016	Sensor	1234	5m flying lead			Active	
Analog Input 17	ST196S-2B	1	Secondary Speed signal B2	PAAF000016	DAAF000017	Sensor	1234	5m flying lead			Active	
Analog Input 18	ST197P	1	Engine phase, primary	PAAF000017	DAAF000018	Sensor	1234	5m flying lead			Active	
Analog Input 19	ST197S	1	Engine phase, secondary	PAAF000018	DAAF000019	Sensor	1234	5m flying lead			Active	