

Assignment 2 - Full Report

by Declan Davies

FILE	11279_DECLAN_DAVIES_ASSIGNMENT_2_-_FULL_REPORT_126969_1472169734.DOCX		
TIME SUBMITTED	01-MAY-2018 05:33PM (UTC+0100)	WORD COUNT	8768
SUBMISSION ID	88044648	CHARACTER COUNT	57353



CAE Systems > ADO Checking Tools > Honda Tools > Honda Wire Comparison VS HBS (SA)

Home

Customer Data Tools

HBS Common Tools

QC-2018

HONDA WIRE LIST COMPARISON

UNDER TESTING

***WIRE LIST COMPARISON CHECKS WIRE ENDS ONLY**

***SPICE ENDS NEED TO BE CHECKED SEPERATELY**

INSTRUCTIONS TO COMPLETE HONDA WIRE LIST COMPARISON

- Step 1: Upload All Relevant HBS Text Files
 - Select HBS Text Files
- Step 2: Upload Only JH1 Sub Assembly Files (.JH1)
 - Select JH1 Files (.JH1)
- Step 3: Upload Honda Connector List File (.XLS)
 - Select Honda File (.xls)
- Step 4: Select Button To Complete Comparison
 - Select For Comparison

HBS Comparison Tools

PSDE 5007 – INDIVIDUAL REPORT

Declan Davies | HNC Electrical Engineering | 18/04/2018

Contents

Table Of Figures.....	3
1. Introduction and Background Research	6
1.1 Introduction.....	6
1.2 Background Research	6
2. Project Proposal	8
2.1 Summary Of Project	8
2.2 Company Background	8
2.3 Problem Project Facing.....	8
2.4 Objectives	8
2.5 technical approach.....	9
2.6 tasks to be carried out	9
2.7 deliverables	10
2.8 Budget.....	10
3. Specification	11
3.1 Technical Specification	11
3.2 Key Dates	15
3.3 Budget	16
4. Planning.....	17
5. Possible Project Solutions	20
5.1 Solution one	20
5.2 Second Solution.....	21
5.3 Third Solution	22
5.4 Conclusion	22
6. Final Solution And Implementation.....	23
6.1 Building User Interface (To Company standard).....	25
6.2 Designing Customer Wire Comparison Tool	30
7. Testing and Results.....	44
7.1 Agreed upon testing table.....	44
7.2 Testing Results	44

8.	Discussion and Conclusions.....	48
9.	Reference List and Bibliography	49
10.	Appendices	51
10.1	CAE Interface Standard	51

Table Of Figures

Figure 1 - Customer Data.....	7
Figure 2 - Text Files	7
Figure 3 - JH1 Files	7
Figure 4 - Customer Wire Information.....	11
Figure 5 - Wiring Harness	12
Figure 6 - Wiring Harness Drawing	12
Figure 7 - HBS (Internal CAD System) Output	13
Figure 8 - HBS Text File Output.....	13
Figure 9 - HBS JH1 File Output.....	14
Figure 10 - Sub Assembly Text File.....	14
Figure 11 - HBS Example Once Processed(Before JH1 Files)	14
Figure 12 - HBS Example With Both JH1 And Text Files Processed	15
Figure 13 - Aim On Completion	15
Figure 14 - Project Plan.....	17
Figure 15 - First Part Of Project Plan	18
Figure 16 - Second Part Of Project Plan	18
Figure 17 - Third Part Of Project Plan	18
Figure 18 - Fourth Part Of Project Plan.....	19
Figure 19 - Fifth Part Of Project Plan	19
Figure 20 - CAE Interface	20
Figure 21 - Flow Diagram.....	20
Figure 22 - Navigation bar.....	21
Figure 23 - Flow Diagram	21
Figure 24 - Creating New Project	23
Figure 25 - Selecting Authentication Type	24
Figure 26 - Access Point on Company Intranet System.....	24
Figure 27 - Default interface Upon Completion.....	25
Figure 28 - Site Master Page Interface Code Required	25
Figure 29 - CSS Classes.....	26

Figure 30 - CSS Class	27
Figure 31 -User Interface (Top With Breadcrumbs)	27
Figure 32 – Breadcrumbs Site Master Code	27
Figure 33 - Breadcrumbs Site Map.....	27
Figure 34 - Breadcrumbs Example.....	28
Figure 35 - Navigation Bar.....	28
Figure 36 – Navigation Bar ASPX Code.....	29
Figure 37 - Navigation Bar Sub Menu's	29
Figure 38 - Adding Required Control Fields.....	30
Figure 39 - CSS Class For Lists.....	30
Figure 40 - ASPX Page With Relevant Control Fields.....	30
Figure 41 – Folder Path Code For Uploading Files	31
Figure 42 - Code To Retrieve Username	31
Figure 43 - Declaring Variables.....	31
Figure 44 - Saving Engineering Files	32
Figure 45 - Function To Move Customer Data Into A Data Table.....	33
Figure 46 - Catch Part Of Try Catch.....	33
Figure 47 - Delete Folder Function	33
Figure 48 - Sort Through HBS Text Files (Part 1)	34
Figure 49 - HBS Add Column Function	34
Figure 50 - Sorting HBS Files	35
Figure 51 - Removes Duplicate Wire Ends	36
Figure 52 - Sort HBS Function	36
Figure 53 - HBS Data Looks At This Point	37
Figure 54 - Master Wire No	37
Figure 55 - Making Use of Sub Assembly Files	37
Figure 56 - 2nd Part Of Taking Care Sub Assembly's.....	38
Figure 57 - 3rd Part Of Taking Care Sub Assembly's.....	38
Figure 58 - Final Part Of Sub Assembly's.....	39
Figure 59 – Comparison	39

Figure 60 - Comparison Part 2.....	40
Figure 61 - Display the comp data table on screen.....	40
Figure 62 - Highlight Discrepancies.....	40
Figure 63 - Example Of Completed Comparison	41
Figure 64 - Wire With Mismatch	41
Figure 65 - The discrepancy	42
Figure 66 - Publishing Project	42
Figure 67 - Publishing Project 2.....	43
Figure 68 - Project Published.....	43
Figure 69 - Deleted Wire From Text File.....	44
Figure 70 - HBS Derivative Missing.....	45
Figure 71 - Change In Customer Data	45
Figure 72 - Honda Derivative Missing.....	46
Figure 73 - Additional Wire	46
Figure 74 - Empty Temp Folder.....	46
Figure 75 - Show All Wires.....	47
Figure 76 Export Excel.....	47

1. Introduction and Background Research

1.1 INTRODUCTION

The aims and objectives for this project will be to build a set of tools available to the engineers on the company's intranet. The aims and objectives for this tool will be to reduce the amount of time it currently takes engineers to carry out an individual engineering task, which in this case is to reduce the amount of manual checks that need to be carried.

The aim of reducing this is to make these tasks as automated as possible for the engineers. The reason this is vital as it will reduce the workload for the engineers and keep the department from running past deadlines. As a result of running past deadlines due to the amount of checks that need to be manually checked by the engineers this results in the department having to do a lot of overtime as each check can take up to a couple of days.

The aim in making these checks automated is that it will speed up the time taken to complete the comparisons and free up engineers to carry out other tasks while the checks are being run. The aim for the automated checks will be no check should take longer than 5 minutes within an automated system.

The check that will be automated is a wire list comparison between our internal data and that of the customer data. SEWS-E data should always match up with the customer as the customer data is what we need to work off. The system should highlight any error in red while highlighting anything that is correct in green to make it clear and easy for the engineers to see if there are any issues.

Some additional features that would be useful for this tool is to allow the engineers to view either all the wire ends or just mismatches that have been found between our data and the customer data.

1.2 BACKGROUND RESEARCH

Good introduction

When carrying out some research into the problem the figure below should display the data that we receive from the customer that we need to confirm our own data against to confirm we are meeting the customer requests.

Circuit No.	Terminal Label	Circuit Name	Terminal Feature		32107-TEK-G100	32107-TEY-H700	32107-TEZY-P000	32107-TEJ-B300	32107-TFKY-G000
A32	A	EPB_L_ACT+	EPB & VSA		-	0	0	0	0
A33	A	EPB_L_ACT-	EPB & VSA		0	0	0	0	0
A24	B	RR_L_ABS_SNSR+	VSA		0	0	0	0	0

Figure 1 - Customer Data

After looking at the customer data the columns that need to be used for comparison against our internal data have been highlighted. These column need to be compared against our data which is an output from our internal CAD system called HBS. The output from this does not match the file format that we receive from the customer so the system will need to take care of that as the output from our system is in text files and JH1 files. That can be seen in figure 2 and 3.

2300_32107-TEC-B011-F__140002_009E	28/08/2017 15:28	Text Document	212 KB
2300_32107-TEC-B111-F__140002_009E	28/08/2017 15:28	Text Document	230 KB
2300_32107-TEC-B211-F__140002_009E	28/08/2017 15:28	Text Document	207 KB
2300_32107-TEJ-B111-F__140002_009E	28/08/2017 15:28	Text Document	245 KB
2300_32107-TEJ-B211-F__140002_009E	28/08/2017 15:28	Text Document	240 KB

Figure 2 - Text Files

2300_77961-TEA-7A0101__140002_009E_A	28/08/2017 15:02	JH1 File	8 KB
2300_77961-TEA-7B0101__140002_009E_A	28/08/2017 15:02	JH1 File	11 KB
2300_77961-TEA-7C0101__140002_009E_A	28/08/2017 15:02	JH1 File	6 KB
2300_77961-TEA-7D0101__140002_009E_A	28/08/2017 15:02	JH1 File	5 KB

Figure 3 - JH1 Files

After looking at both the JH1 and text files it became clear that they both use the same denominator to split each line into columns, which in this case is “;”. This can be used within the system to format the data correctly.

2. Project Proposal

2.1 SUMMARY OF PROJECT

The project that I am going to be looking to propose is to build an intranet system that will reduce the time taken to carry out an individual engineering task significantly. The task that I am looking to significantly improve is the Customer vs HBS Wire List comparison due to the fact that this comparison can take a couple of days per wiring harness to perform this comparison. The system will compare the customer data against our internal data and check for any discrepancies and highlight them for the engineer.

2.2 COMPANY BACKGROUND

Sumitomo Electric Wiring Systems is a global company but this project will be based in the Ystradgynlais site but will be used on multiple sites worldwide. As a company we are responsible for supplying automotive wiring harnesses to multiple customers for their cars some of the examples of this can be seen below:-

- PSA
- Renault
- Nissan
- Toyota
- Suzuki
- Honda

The company has multiple manufacturing sites worldwide which includes Morocco, Romania and Egypt. A wiring harness in the automotive industry means it is the wiring system that is in a car that allows the systems within that car to work as expected. There are many different types of wiring harnesses that we need to design and manufacture that all vary in size and the components that are used on each of them.

2.3 PROBLEM PROJECT FACING

The problem that this project is looking to solve is the amount of time it takes engineers to carry out the wire comparison. The reason this is so important is due to the amount of time this task currently takes to perform and also the risk of human error when carrying out the comparison due to the amount of wire no's that need to be checked along with the fact that there can be hundreds of derivatives with each wire number that need to be checked as well. An error can result in the wiring harness being delivered to the customer with an error and for example the car not starting because of a discrepancy in the data.

2.4 OBJECTIVES

The main objective for this project is going to design a system that will have the ability to identify any discrepancies between the customer data and our internal data. The discrepancies could include a missing wire along with a missing derivative associated with

any of the specific wires. Along with giving the user the option to only view the discrepancies in the data, while also giving them the option to view all the wire ends.

2.5 TECHNICAL APPROACH

This system is going to be built from scratch using Microsoft Visual Studio and an ASP.NET project. The system interface will need to be designed to meet CAE user interface standards to be in line with other applications on the CAE part of the company's intranet. When the system has been built the user should be able to select all the relevant documents needed in order to carry the comparison out. Then they should just be able to press a button and the comparison should be complete within and 5 minutes to make it a significant improvement on the manual process and then they should be able to view only the discrepancies by using a button.

2.6 TASKS TO BE CARRIED OUT

The expected completion date for this project is going to be the 04/03/2018 and the tasks that need to be carried out before then can be seen below:-

ACTIVITY	START	END	NOTES
Project Start	10/09/2017		
Confirm Project Idea	10/09/2017	20/09/2017	
Research Into Three Solutions	20/09/2017	21/10/2017	
Confirm Final Solution	21/10/2017	28/10/2017	
Design Specification	31/10/2017	14/11/2017	
Design Test Plan	15/11/2017	29/11/2017	
Retrieve Relevant Documents	29/11/2017	03/12/2017	
Create Visual Studio Project	03/12/2017	10/12/2017	
Start Building User Interface	10/12/2017	09/01/2018	
Start Code For Saving Relevant Documents	09/01/2018	16/01/2018	
Start Sorting Through HBS Text Files	16/01/2018	26/01/2018	
Code To Take Care Of JH1 Files	26/01/2018	30/01/2018	
Put Honda Text Files Into Data Table	30/01/2018	02/02/2018	
Start Code For Comparison Of Documents	02/02/2018	04/03/2018	
Project End	04/03/2018		

2.7 DELIVARABLES

On completion of this project the ADO department should have an application that is able to do the following:-

- An Intranet System able to handle text files that have been outputted from HBS our internal CAD system.
- That will be able to take care of Sub-Assemblies through the JH1 files.
- Compare Customer Data Vs our Internal Data from HBS.
- Highlight Discrepancies in Red and matches in green
- Engineers be able to view only discrepancies

2.8 BUDGET

For this project no budget was needed as it was felt all the necessary software and equipment was already available on site.

There is always a cost incurred. In this case the cost of the time on the software and third party assistance is a cost. Compare this to the cost of outsourcing this project.

3. Specification

For the project to be considered a success on completion the specification to follow will need to be met.

3.1 TECHNICAL SPECIFICATION

The objectives for this project will be to have the ability to identify any discrepancies between our data and the data that is received from the customer. This will require the system on completion to be somewhere where all engineers have easy access to perform the checks.

The aims for these tools that will be developed for the engineers are to make engineering tasks automated and quicker and more efficient for the engineers. The reason this is vital to the project is due to the fact that some of the tasks that will be performed by an automated system to be designed can take days being performed by a single engineer that can result in errors and can be a constraint on resources in a department.

The problem with engineers spending days carrying out these tasks is that they are currently required to compare two excel documents against each other manually and these documents can contain 100's of columns and thousands of rows, which to be honest means the likely hood of a mistake being made is greatly increased.

Wire No		Derivatives (Options On Car)	
Circuit No	Terminal Label	Circuit Name	Terminal Feature
A32	A	EPB_L_ACT+	EPB & VSA
A33	A	EPB_L_ACT-	EPB & VSA
A24	B	RR_L_ABS_SNSR+	VSA
A25	B	RR_L_ABS_SNSR-	VSA
A30	A	EPB_R_ACT+	EPB & VSA

32107-TK-G100	32107-TEY-H700	32107-TEZY-P000	32107-TEJ-8300	32107-TFY-G000	32107-TEZY-P100	32107-TEY-H500	32107-TEZY-P200	32107-TES-H010	32107-TES-H110
1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Figure 4 - Customer Wire Information

As can be seen in figure 1 is a small example of what the engineers could expect unless it's for a wiring harness from the door of the car being built as that will have only a few connections, whereas on an instrument harness it will contain thousands of wires. As a result the engineer needs to check each wire and the derivatives associated to make sure they match the information in our internal system. The problem if they don't could result in a problem with the harness like them being shipped to the customer installed in the car on a manufacturing site and possible the car not starting due to a derivative or wire missing.

[Good explanation here](#)

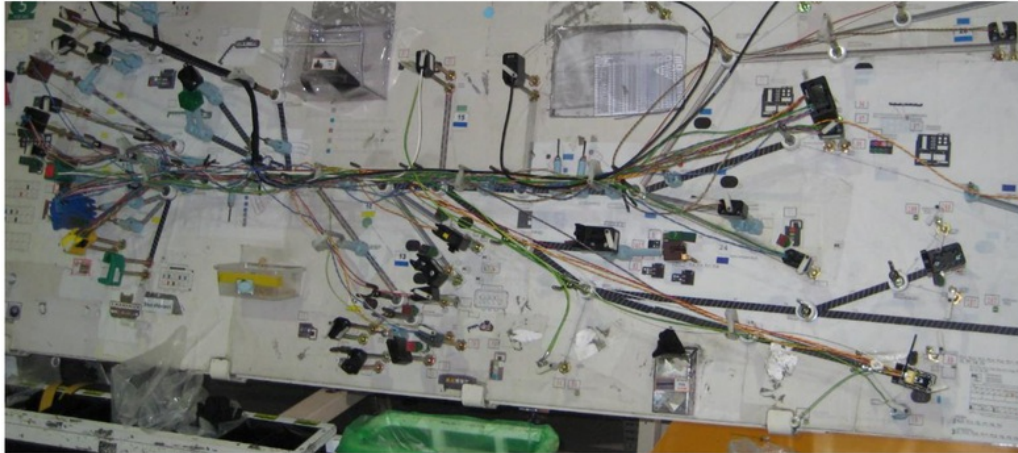


Figure 5 - Wiring Harness

In figure 5 it shows the picture of a wiring harness being manufactured on a layup board. The layup board was designed using the company's internal CAD system that uses the information that is being manually compared by the engineers currently. In figure 1 behind each wire there are pictures and line showing which where needs to go where, which means any wire missing during the design process due to human error will not be displayed on the lay-up board meaning there will be an error.

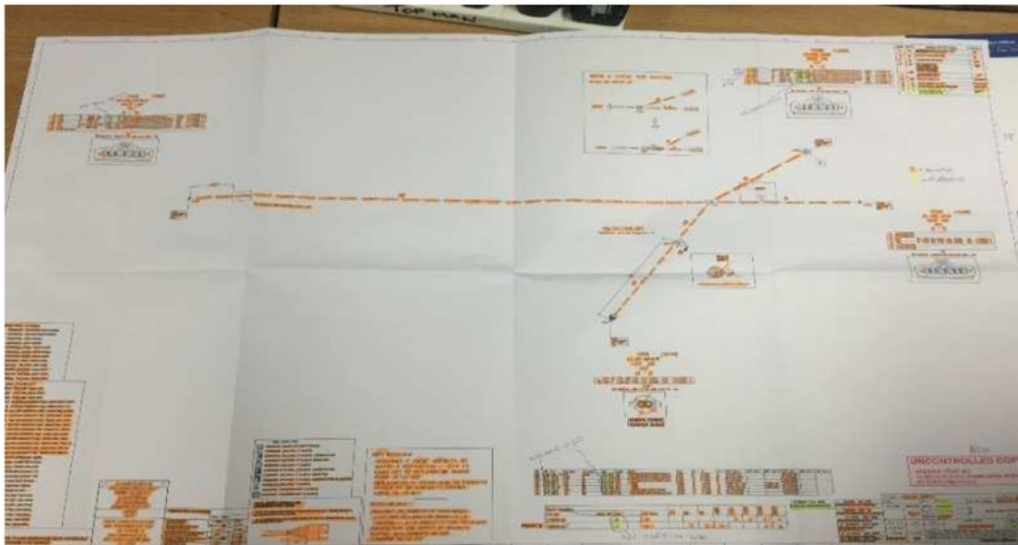


Figure 6 - Wiring Harness Drawing

In figure 6 it displays a picture of a wiring harness drawn on the company's internal CAD system. The data that will be compared in the tools that are going to be built is taken from

the output from the CAD system for these drawings and they are outputted as Text Files and JH1 files.

	2300_77961-TEA-7A0101__140002_009E_A	28/08/2017 15:02	JH1 File
	2300_77961-TEA-7B0101__140002_009E	28/08/2017 15:31	Text Document
	2300_77961-TEA-7B0101__140002_009E_A	28/08/2017 15:02	JH1 File
	2300_77961-TEA-7C0101__140002_009E	28/08/2017 15:31	Text Document
	2300_77961-TEA-7C0101__140002_009E_A	28/08/2017 15:02	JH1 File
	2300_77961-TEA-7D0101__140002_009E	28/08/2017 15:31	Text Document
	2300_77961-TEA-7D0101__140002_009E_A	28/08/2017 15:02	JH1 File
	2300_77961-TEA-7E0101__140002_009E	28/08/2017 15:31	Text Document
	2300_77961-TEA-7E0101__140002_009E_A	28/08/2017 15:02	JH1 File
	2300_77961-TEA-7F0101__140002_009E	28/08/2017 15:31	Text Document
	2300_77961-TEA-7F0101__140002_009E_A	28/08/2017 15:02	JH1 File
	2300_77961-TEA-7G0101__140002_009E	28/08/2017 15:31	Text Document
	2300_77961-TEA-7G0101__140002_009E_A	28/08/2017 15:02	JH1 File
	2300_77961-TEA-7H0101__140002_009E	28/08/2017 15:31	Text Document
	2300_77961-TEA-7H0101__140002_009E_A	28/08/2017 15:02	JH1 File
	2300_77961-TEA-7I0101__140002_009E	28/08/2017 15:31	Text Document
	2300_77961-TEA-7I0101__140002_009E_A	28/08/2017 15:02	JH1 File
	2300_77961-TEA-7J0101__140002_009E	28/08/2017 15:31	Text Document
	2300_77961-TEA-7J0101__140002_009E_A	28/08/2017 15:02	JH1 File

Figure 7 - HBS (Internal CAD System) Output

In figure 7 it shows the output files from HBS but in order for these files to be compared with the customer data they need to be compiled into one table. This requires knowing the relevant information along with how to breakdown the documents.

	Section Number	Derivative Number	Wire No's
0;32107-TEA-ZZ14MP	14	0002;32107-TEC-B011-F	14 0002;0081;0170;0174;0699;0009;0000;0000;0000;0000;0000;0000;0;
7;32107-TEA-ZZ14MP	14	0002;009E;9999;01;2300;156;32107-TEC-B011-F	14 0002;1;20170825;Y;2VR1;3;
2;32107-TEA-ZZ14MP	14	0002;009E;2300;32107-TEC-B011-F	14 0002;*17;*18;;1;156;500;;;ATW;N-7902;;;30;1.25;3672.0;30
2;32107-TEA-ZZ14MP	14	0002;009E;2300;32107-TEC-B011-F	14 0002;804;805B;;2;156;500;;;ATW; 050;;Y;10;0.50;3703.0
2;32107-TEA-ZZ14MP	14	0002;009E;2300;32107-TEC-B011-F	14 0002;805A;805B;;3;156;500;;;ATW; 050;;BR;10;0.50;3703.0
2;32107-TEA-ZZ14MP	14	0002;009E;2300;32107-TEC-B011-F	14 0002;*19;*20;;4;156;500;;;ATW;N-7364;;;30;1.25;3669.0;30
2;32107-TEA-ZZ14MP	14	0002;009E;2300;32107-TEC-B011-F	14 0002;806B;806C;;5;156;500;;;ATW; 050;;L;10;0.50;3700.0

Figure 8 - HBS Text File Output

In figure 8 it shows the contents of a text file that has been output from the company's internal CAD system. As can be seen in figure 1 each item is split un using a denominator of ";" to separate each column. The first column is identified as Section Number and each section contains different information relevant to the task of comparing wire information. For the task of comparing the two wire lists for our customer data against our own the sections most relevant to me are sections 0 and sections 2 as they contain the derivative number and the wire no's respectively.

This is quite a task to get a

Sub Assembly	Derivative No	Section Number	Normal Derivative No
2300;77961-TEA-7A0101;14	0002;009E;2016;A;02;0092;17;A;37.4;		
2300;77961-TEA-7A0101;14	0002;009E;2016;A;02;0093;17;M;34.0;		
2300;77961-TEA-7A0101;14	0002;009E;2016;A;02;0503;01;A;67.5;		
2300;77961-TEA-7A0101;14	0002;009E;2016;A;20;2300;32107-TES-H011;14		0002;009E;040;
2300;77961-TEA-7A0101;14	0002;009E;2016;A;20;2300;32107-TEC-B211;14		0002;009E;040;
2300;77961-TEA-7A0101;14	0002;009E;2016;A;20;2300;32107-TEJ-B301;14		0002;009E;040;
2300;77961-TEA-7A0101;14	0002;009E;2016;A;04;*47;*48;ATW;N-7473; ;2485.0;;;0015;12.5;;;		
2300;77961-TEA-7A0101;14	0002;009E;2016;A;04;*59;*60;ATW;N-7473; ;2259.0;;;0015;12.4;;;		
2300;77961-TEA-7A0101;14	0002;009E;2016;A;03;145027;0031;29.0;# ;		

Figure 9 - HBS JH1 File Output

In figure 9 it displays the JH1 Output from the company internal CAD system in HBS and this file is used to determine what derivatives each sub assembly is used in. An example of how this process works is each sub assembly has a JH1 output and a text file output. The text file contains the wires that will be used within that sub assembly whereas the JH1 file will tell you what derivatives that Jh1 file is used in.

Sub Assembly	Derivative No	Section Number	Wire No's
0;32107-TEA-ZZ14MP	14	0002;77961-TEA-7A0101	14 0002;0017;0034;0022;0049;
7;32107-TEA-ZZ14MP	14	0002;009E;9999;01;2300;001;77961-TEA-7A0101	14 000
2;32107-TEA-ZZ14MP	14	0002;009E;2300;77961-TEA-7A0101	14 0002;*47;*48;;1
2;32107-TEA-ZZ14MP	14	0002;009E;2300;77961-TEA-7A0101	14 0002;S55C;S55A;
2;32107-TEA-ZZ14MP	14	0002;009E;2300;77961-TEA-7A0101	14 0002;S56C;S56A;
2;32107-TEA-ZZ14MP	14	0002;009E;2300;77961-TEA-7A0101	14 0002;*59;*60;;4
2;32107-TEA-ZZ14MP	14	0002;009E;2300;77961-TEA-7A0101	14 0002;S57C;S57A;

Figure 10 - Sub Assembly Text File

In figure 10 it shows the sub assembly text file that relates to the JH1 file in figure 9 that shows that wire no's in figure 10 will be used in the normal derivative no's that can be seen within figure 9.

Wire No	32107-TES-H011	32107-TEC-B211	32107-TEJ-B301	77961-TEA-7A01
S55C				O
S55A				O
S56C				O
S56A				O
S57C				O
S57A				O

Figure 11 - HBS Example Once Processed(Before JH1 Files)

In figure 11 it shows an example of what can be expected once the Text Files have been processed before the JH1 files.

Wire No	32107-TES-H011	32107-TEC-B211	32107-TEJ-B301
S55C	0	0	0
S55A	0	0	0
S56C	0	0	0
S56A	0	0	0
S57C	0	0	0
S57A	0	0	0

Figure 12 - HBS Example With Both JH1 And Text Files Processed

In figure 12 it shows an example of how the HBS table should look once both JH1 files and Text Files have been processed it shows that like can be seen in figure 7 and 8 the wire no's in figure 7 are in the derivatives that can be seen in figure 7. Following getting both the customer data and our internal data into the correct format the comparison should look like figure 14 on completion.

Wire No	32107-TES-H011	32107-TEC-B211	32107-TEJ-B301
S55C	0	1	1
S55C	0	0	1
S55A	1	0	1
S55A	1	0	1

Derivative No's

SEWS DATA

Customer Data

Figure 13 - Aim On Completion

As can be seen in figure 13 for each wire number in our data and the customer data should be next to each other showing the differences between each. The SEWS data lines should be shown in blue and the customer data should be in white with the differences highlighting the number red and matches highlighting the number green.

3.2 KEY DATES

The aim for the project will be to complete it by the 16th of April 2018, which means all parts of the project will need to be completed by this date. In order to meet this it will require the following key aspects being completed on time:-

- Confirm Project Idea 20/9/2017
- Decide On Final Solution 21/10/2017
- Complete Interim Report 10/01/2018
- Complete User Interface By 09/01/2018
- Write Code Behind User Interface By 13/03/2018
- Write Testing Plan By 15/11/2017
- Write Final Report By 01/05/2018

3.3 BUDGET

For this project no budget was needed as it was felt all the necessary software and equipment was already available on site.

The design and implementation on cost of a system like this could be quite substantial if outsourced. In this case the budget is the time of all persons connected and contributed to the completion of the project. This will come out as a figure.

4. Planning

The project was planned using Microsoft Excel and the complete timing plan can be seen in figure 14.

ACTIVITY	START	END	NOTES
Project Start	10/09/2017		
Confirm Project Idea	10/09/2017	20/09/2017	Went According To Plan And Finished On Time
Research Into Three Solutions	20/09/2017	21/10/2017	Went According To Plan And Finished On Time
Confirm Final Solution	21/10/2017	28/10/2017	Went According To Plan And Finished On Time
Design Specification	31/10/2017	14/11/2017	Took A Couple Days Longer than expected Due To Boss Being Of Work
Design Test Plan	15/11/2017	29/11/2017	Still Completed On Time Even Though Started At a Later Date
Retrieve Relevant Documents	29/11/2017	03/12/2017	Went According To Plan
Create Visual Studio Project	03/12/2017	10/12/2017	Went According To Plan
Start Building User Interface	10/12/2017	09/01/2018	Finished Early Due To Having To Design User Interface To CAE Standard
Start Code For Saving Relevant Documents	09/01/2018	16/01/2018	Went According To Plan
Start Sorting Through HBS Text Files	16/01/2018	26/01/2018	Went According To Plan
Code To Take Care Of JH1 Files	26/01/2018	30/01/2018	Went According To Plan
Put Honda Text Files Into Data Table	30/01/2018	02/02/2018	Went According To Plan
Start Code For Comparison Of Documents	02/02/2018	04/03/2018	Went According To Plan
Project End	04/03/2018		

Figure 14 - Project Plan

In figure 14 the project overall went according to plan for the most part besides the few tasks that run past the expected end times due to the ADO supervisor being away on business during that period. Also

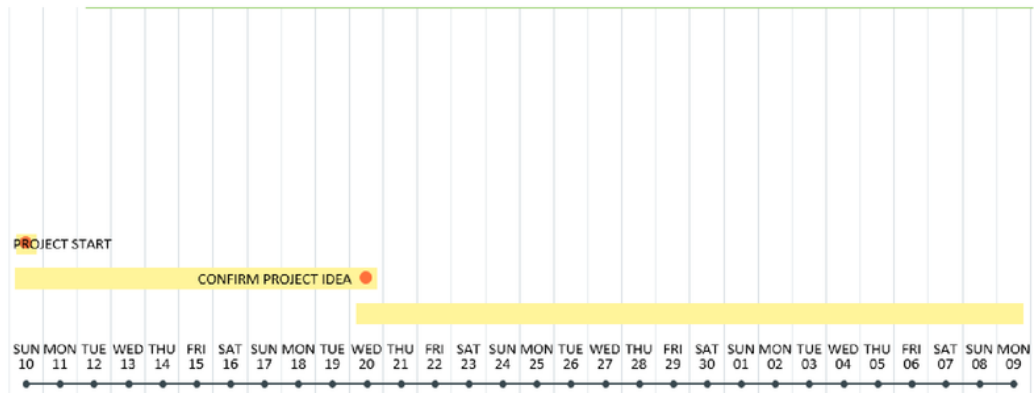


Figure 15 - First Part Of Project Plan

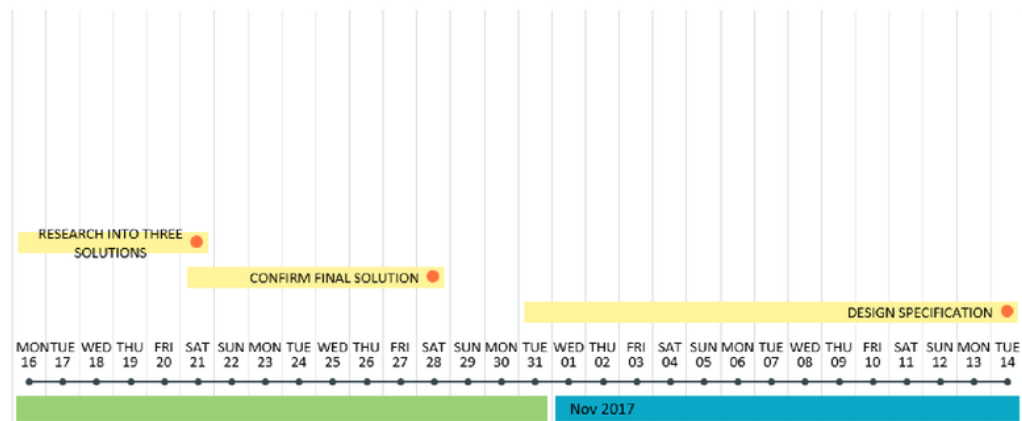


Figure 16 - Second Part Of Project Plan

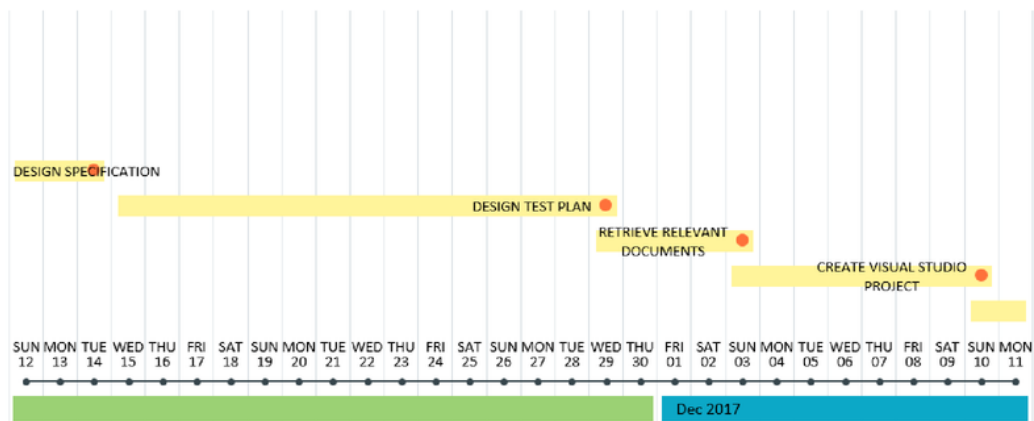


Figure 17 - Third Part Of Project Plan

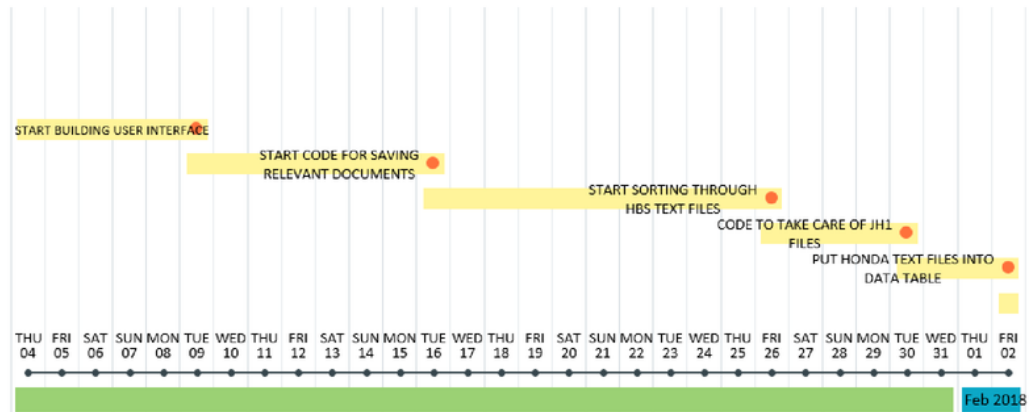


Figure 18 - Fourth Part Of Project Plan

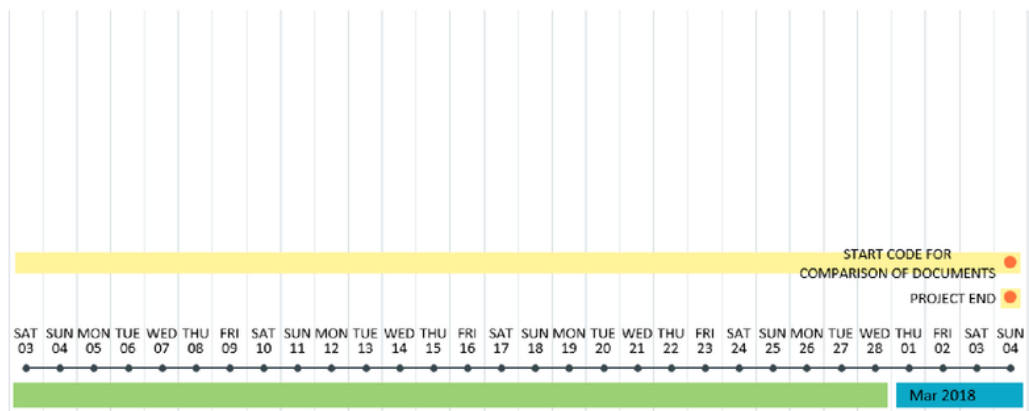


Figure 19 - Fifth Part Of Project Plan

Good
plan

5. Possible Project Solutions

As part of this project there has been a solution chosen that will solve the issue of human error when checking electrical information. Each of the following solutions that will be laid out will meet the specifications requirements for the project.

5.1 SOLUTION ONE

The first solution for the project will be to create an ASPX project that will hold all of the electrical comparison tools that need to be created as part of the project. The following tools will then be created using the programming language C#, while creating the user interface based off of the CAE standard for the intranet. The CAE standard for an interface can be seen in figure 20.



Figure 20 - CAE Interface

As can be seen in figure 20 the CAE interface has a blue and white color scheme to match the company colors. The ADO Comparison tools will be split up into the following sections as can be seen in figure 21.

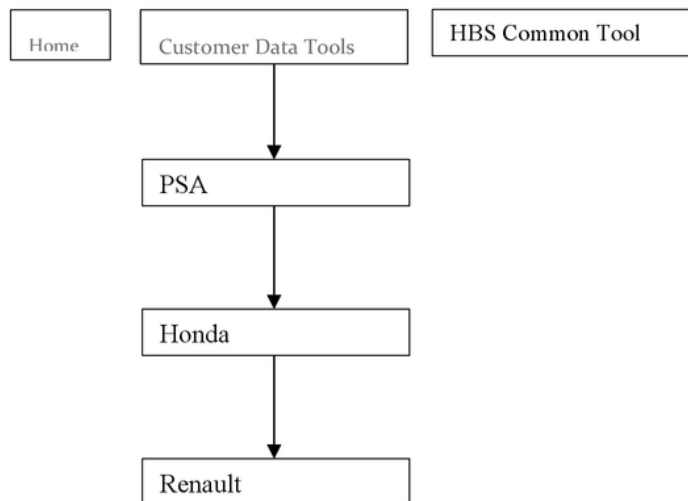


Figure 21 - Flow Diagram

In figure 21 it shows the navigation bar for this solution and how it would be set up and an example of this being made in a visual studio also can be seen in figure 22.



Figure 22 - Navigation bar

For this solution after creating the navigation bar the programming constructs for this solution that will be required in order to create the comparison tools are creating a function to upload excel worksheets into a data table and along with having the ability to loop through text files and dsi files in order to store the HBS information into a data table. Once the HBS and Customer information is stored into data tables so that they can be compared through making use of linq queries. On Completion of the comparison the electrical data will be placed into grid view and will be highlighted in red for any differences and green for any match ups.

5.2 SECOND SOLUTION

The second possible solution for this project will be to use excel in order to create a macro that will carry out the comparison tools. This will require copying both the customer information and HBS information into two separate sheets in the same excel document. Once both have been added into the excel document the macro will be programmed using visual basic programming. In figure 10 it shows a block diagram of what would be required from the user.

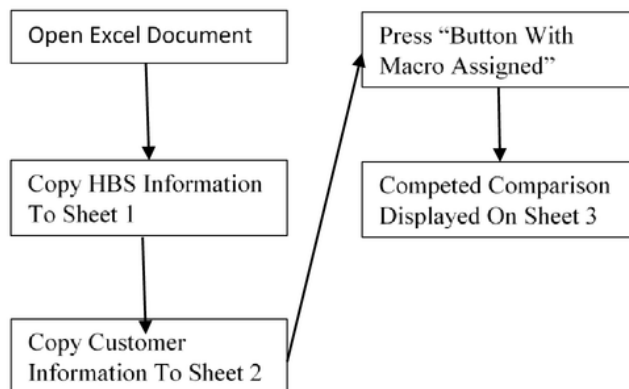


Figure 23 - Flow Diagram

In figure 23 it shows the steps that would need to be taken by the user if this solution was to be implemented and in order to program this using visual studio would require the use of a for loop to loop through each sheet and an IF statement would need to be nested

Good arguments. This could have included an example of an existing macro or other application as a case study. There are plenty of case studies available.

inside the loop to make sure that any comparison being carried out between wire ends are matching so that no wires are being incorrectly compared to each other. The negatives to this kind of solution is it would limit the amount of version control that would be out there as this will be used over multiple manufacturing sites within the company.

5.3 THIRD SOLUTION

The third solution for this project will be to create an executable file that will be on each engineers computer and will be programmed using C# programming. The advantage to creating an executable file over using an excel spreadsheet with a macro is it is harder for the user to make any changes to the program without the original visual studio file. Another issue with this type of solution is again how to keep tabs on the version control and to make sure everyone has access to the file that needs it. The advantage to using an executable file over a aspx document that will be posted on the company's intranet is that its run time is not impacted by the internet's signal strength as it will be run locally.

5.4 CONCLUSION

In conclusion for the final report a solution in solution 1 has been chosen for the final solution to be implemented. The reason this decision came about is due to the fact that this allows for the tools to be posted on the company's intranet system because this is something that engineers will be used to using on a regular basis. Another reason was down to the fact that it is the easiest solution to keep version control as it can be updated without having to worry about sending about a new file to every user. As can be seen in the project plan once a final solution has been selected it is time to start writing the code for the tools.

6. Final Solution And Implementation

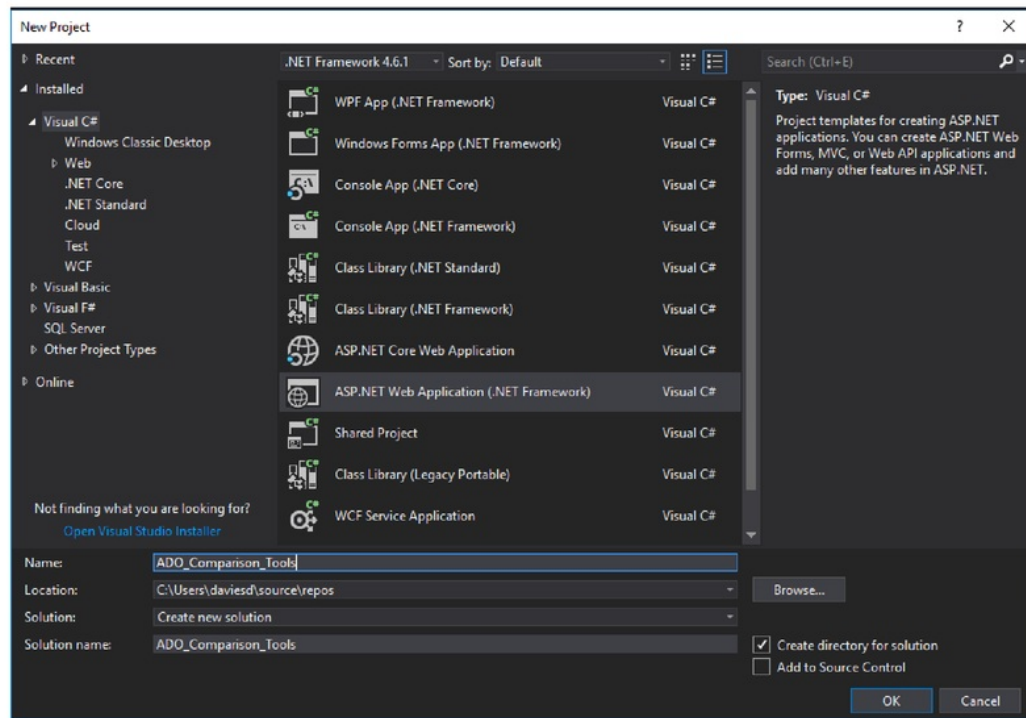


Figure 24 - Creating New Project

In figure 10 it shows the first step in the process of creating a new project using Microsoft visual studio to create the application for the engineering tools that need to be designed.

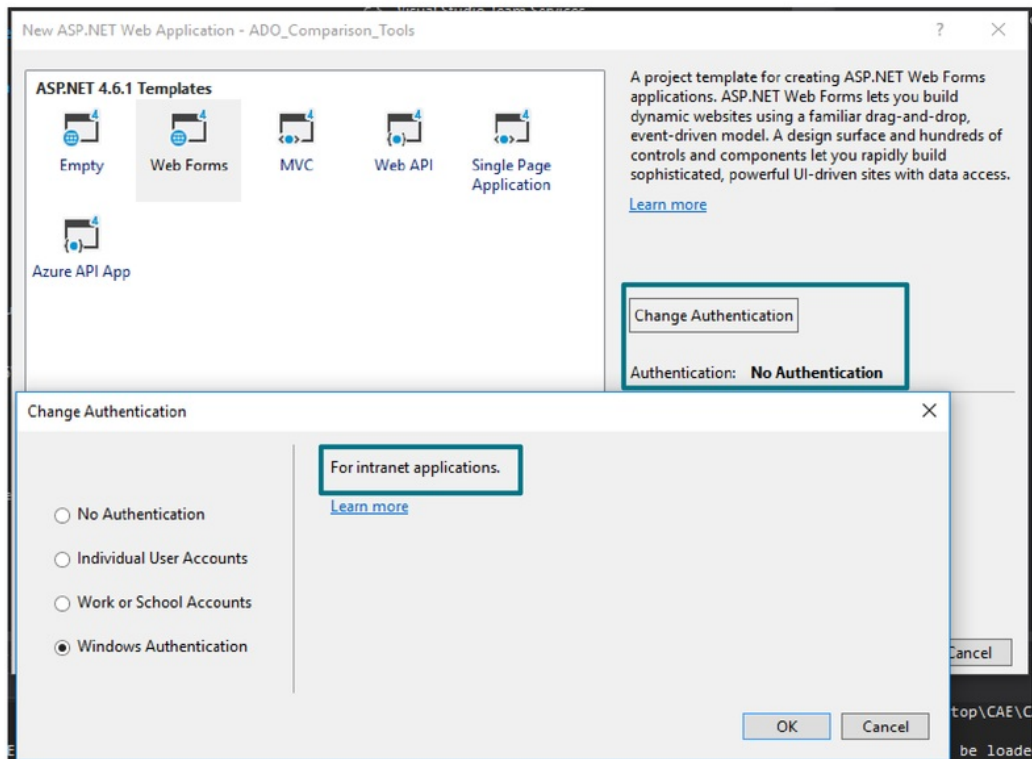


Figure 25 - Selecting Authentication Type

In figure 11 it shows the authentication type that has been selected which in this case is Windows authentication since this will be an Intranet application that will be accessed through the company's current intranet system that can be seen in figure 12.



Figure 26 - Access Point on Company Intranet System

In figure 12 it display the location where the engineers will be able to access the tool on completion to carry out the comparisons required.

6.1 BUILDING USER INTERFACE (TO COMPANY STANDARD)

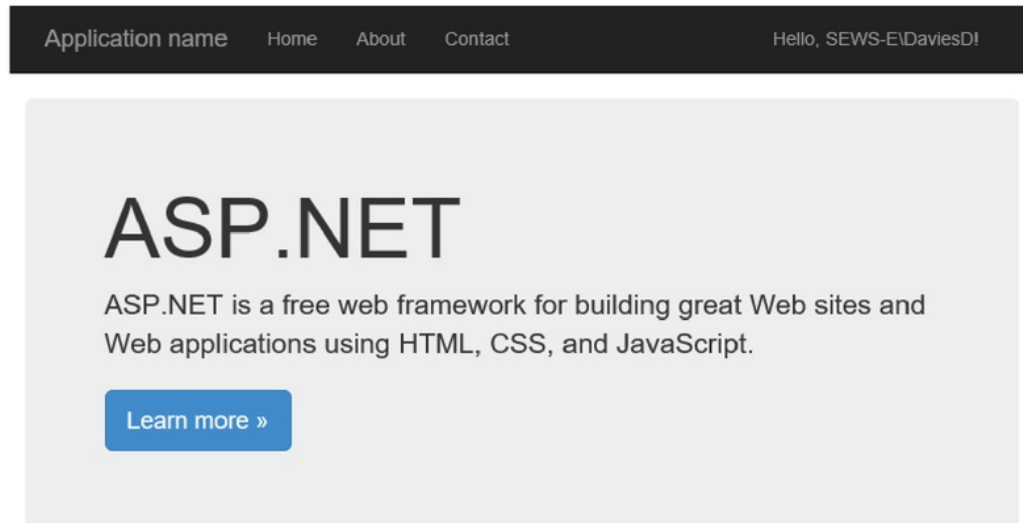


Figure 27 - Default interface Upon Completion

In figure 13 it displays the default interface that is generated for the designer by Microsoft initially but this will need to be altered to meet the company standard.

```
<div id="HeaderLogo">
  
</div>
<div id="Header">
  <div id="HeaderImage">
    <a href="//ukhqengs001/CAE/TheETC.aspx"
      onmouseover="status='CAE Systems : About the European Technical
      Centre at Ystradgynlais'; return true;"
      onmouseout="status=''; return true;"></a>
  </div>
  <div id="HeaderLink" align="right">
    <!--#INCLUDE FILE="caeLinkList.html"-->
  </div>
  <div id="TagLine">
    <b>FOR</b> Engineers <b>BY</b> Engineers
  </div>
</div>
```

Diagram annotations for Figure 28:

- Arrows point from the text "CSS Class" to the `id="HeaderLogo"` and `id="HeaderImage"` attributes.
- An arrow points from the text "Image" to the `src="images/TagImage.gif"` attribute.
- An arrow points from the text "Link To CAE Page (Figure 12)" to the `href="//ukhqengs001/CAE/TheETC.aspx"` attribute.
- An arrow points from the text "Image" to the `src="images/ETC.jpg"` attribute.
- An arrow points from the text "CSS Class" to the `id="HeaderLink"` attribute.
- An arrow points from the text "CSS Class" to the `id="TagLine"` attribute.

Figure 28 - Site Master Page Interface Code Required

In figure 14 it shows the code for the CAE standard for SEWS-E user interfaces as can be seen above it requires two photos with the use of css code to format each of them correctly.

```
/* --- HEADER --- */
#Header
{
    float: right;
    background-color: #FFFFFF;
    color: #000080;
    font-size: 0.8em;
    margin-right: 10px;
}
#HeaderImage
{
    padding: 0px 0px 5px 0px;
    float: left;
    vertical-align: bottom;
}
#TagLine
{
    margin: 0px 0px 5px 0px;
    float: right;
    padding: 5px 15px 5px 15px;
    text-align: right;
    background-color: #000080;
    color: #FFFFFF;
    vertical-align: middle;
    font-size: 1.0em;
}
#TagLine b
{
    color: Orange;
}
.TagLineFOR
{
    margin-right: 30px;
}
```

Figure 29 - CSS Classes

In figure 15 it shows some of the css code that is used to create the user interface that the user will see. The css class called “#Header” has a float right property that will move it to the right of the page along with setting a background colour to “#FFFFFF”, which is the colour code for white. The CSS class called “#HeaderImage” will float property’s with this to the left of the page and then it will be aligned at the bottom. The next CSS class called “#TagLine” to float the content to the right and then it has a background colour of dark blue with a fore colour of white. Along with the class “TagLine b” having the fore colour set to orange and the CSS class “TagLineFOR” has a margin of 30px set to the right which is the amount of space between the object and the right side of the page.

```
.TagLineBY
{
    margin-left: 30px;
}
```

Figure 30 - CSS Class

In figure 16 it shows the CSS class called "TagLineBY" which sets a margin of 30px to the left which is the distance between the left side of the page and the contents with this class assigned.



Figure 31 -User Interface (Top With Breadcrumbs)

In figure 17 it shows the result of the code seen in figures 14 and 15, the only thing that has yet to be explained is the breadcrumbs that has been added in figure 17. This is important as it shows the user where exactly they are on the page at any time.

```
<div class="clear" id="Breadcrumbs">
    <asp:SiteMapDataSource ID="SiteMapDataSource1" runat="server" ShowStartingNode="true" />
    <asp:SiteMapPath ID="SiteMapPath1" runat="server" PathSeparator=" " RenderCurrentNodeAsLink="false">
    </asp:SiteMapPath>
    <hr />
</div>
```

Figure 32 - Breadcrumbs Site Master Code

In figure 18 it displays the aspx code on the site master page for the breadcrumbs on the web page as can be seen in figure 17. In figure 18 it shows that in order to create breadcrumbs it requires creating a site map for the web page to act as the data source.

```
<siteMap xmlns="http://schemas.microsoft.com/AspNet/SiteMap-File-1.0" >
  <siteMapNode url="http://ukhgengs001/CAE/Systems" title="CAE Systems" description="a" />
  <siteMapNode url="~/HomePage/Default" title="ADO Checking Tools" description="ADO Comparison Tools" />
  <siteMapNode title="Tools" description="Comparison Tools" />
  <siteMapNode url="~/Tools/Wire" title="Wire Comparison" description="Wire Comparison" />
  <siteMapNode url="~/Tools/ADSIComparison" title="DSI Comparison" description="DSI Comparison VS HBS" />
  <siteMapNode url="~/Tools/AComponent" title="Component Comparison" description="Component Comparison" />
  <siteMapNode url="~/Tools/AMatrixComparison" title="Matrix Comparison" description="Matrix Comparison" />
  <siteMapNode url="~/Tools/ADCSCheck" title="ADCS Check" description="ADCS Check" />
  <siteMapNode url="~/Tools/ASplices" title="Splice Report 1" description="Splice Report 1" />
  <siteMapNode url="~/Tools/ASplices2" title="Splice Report 2" description="Splice Report 2" />
</siteMapNode>
```

Figure 33 - Breadcrumbs Site Map

In figure 19 it shows the site map for the intranet site being created in this case the customers' names have been shaded out for confidentiality. Figure 19 shows that in breadcrumbs the first link will be CAE Systems as whenever the ADO Comparison tools are being used breadcrumbs will always show CAE Systems as ADO Comparison tools are inside CAE systems. Following showing CAE Systems it will show ADO Checking Tools as that's what the user is currently using and then it will depend on what customer the user selects and what tool they are using. An Example of this will be the user is in ADO checking tools and is using a customer wire check.



Figure 34 - Breadcrumbs Example

Once the breadcrumbs have been created the next step is to build a navigation bar for the website that in this case will be a horizontal navigation bar.

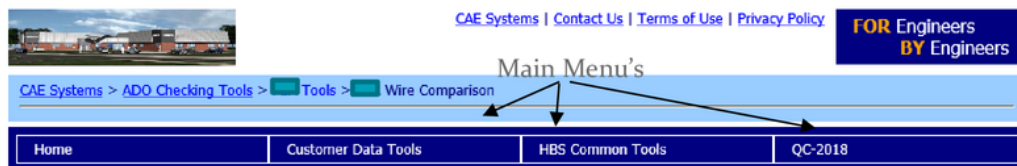


Figure 35 - Navigation Bar

In figure 21 it shows how the navigation bar looked upon completion it requires using the aspx code that can be seen in figure 22.

```

<!-- Menu -->
<div class="clear Menu">
    <asp:Menu ID="menuSite" runat="server" EnableViewState="false" IncludeStyleBlock="false" Orientation="Horizontal"
        StaticDisplayLevels="1" MaximumDynamicDisplayLevels="3">
        <LevelMenuItemStyles>
            <asp:MenuItemStyle CssClass="level1"/>
            <asp:MenuItemStyle CssClass="level2"/>
        </LevelMenuItemStyles>
        <Items>
            <asp:MenuItem NavigateUrl="~/HomePage/Default.aspx" Text="Home"/>
            <asp:MenuItem Text="Customer Data Tools">
                <asp:MenuItem Text="Wire Comparison vs HBS">
                    <asp:MenuItem NavigateUrl="~/Tools/Wire.aspx" Text="Wire Comparison vs HBS"/>
                </asp:MenuItem>
                <asp:MenuItem Text="DCS vs HBS Comparison">
                    <asp:MenuItem NavigateUrl="~/Tools/DCSComparison.aspx" Text="DCS vs HBS Comparison"/>
                </asp:MenuItem>
                <asp:MenuItem Text="Parts Comparison vs HBS">
                    <asp:MenuItem NavigateUrl="~/Tools/Component.aspx" Text="Parts Comparison vs HBS"/>
                </asp:MenuItem>
                <asp:MenuItem Text="Matrix Comparison vs HBS">
                    <asp:MenuItem NavigateUrl="~/Tools/MatrixComparison.aspx" Text="Matrix Comparison vs HBS"/>
                </asp:MenuItem>
                <asp:MenuItem Text="ADCS Crimp Concerns">
                    <asp:MenuItem NavigateUrl="~/Tools/ADCSCheck.aspx" Text="ADCS Crimp Concerns"/>
                </asp:MenuItem>
                <asp:MenuItem Text="Splice Matrix Report">
                    <asp:MenuItem NavigateUrl="~/Tools/Splices.aspx" Text="Splice Matrix Report"/>
                </asp:MenuItem>
                <asp:MenuItem Text="Splice Matrix Report DCS">
                    <asp:MenuItem NavigateUrl="~/Tools/Splice2.aspx" Text="Splice Matrix Report DCS"/>
                </asp:MenuItem>
            </asp:MenuItem>
            <asp:MenuItem Text="Wire Comparison vs HBS (SA)">
                <asp:MenuItem NavigateUrl="~/Tools/Wire.aspx" Text="Wire Comparison vs HBS (SA)" Tooltip="">
                </asp:MenuItem>
            </asp:MenuItem>
            <asp:MenuItem Text="Wire Comparison vs HBS (No SA)">
                <asp:MenuItem NavigateUrl="~/Tools/WireNoSub.aspx" Text="Wire Comparison vs HBS (No SA)" Tooltip="">
                </asp:MenuItem>
            </asp:MenuItem>
            <asp:MenuItem Text="Matrix Comparison vs HBS">
                <asp:MenuItem NavigateUrl="~/Tools/MatrixComparison.aspx" Text="Matrix Comparison vs HBS"/>
            </asp:MenuItem>
            <asp:MenuItem Text="Wire End Report For Input">
                <asp:MenuItem NavigateUrl="~/Tools/SpliceSort.aspx" Text="Wire End Report For Input"/>
            </asp:MenuItem>
            <asp:MenuItem Text="Wire Comparison Customer vs Customer (T1)">
                <asp:MenuItem NavigateUrl="~/Tools/Wire.aspx" Text="Wire Comparison Customer vs Customer (T1)">
                </asp:MenuItem>
            </asp:MenuItem>
            <asp:MenuItem Text="DCS vs HBS Comparison">
                <asp:MenuItem NavigateUrl="~/Tools/DCS.aspx" Text="DCS"/>
            </asp:MenuItem>
            <asp:MenuItem Text="HBS Common Tools">
                <asp:MenuItem NavigateUrl="~/Common_Tools/HBSADCSCheck.aspx" Text="ADCS Crimp Concerns"/>
            </asp:MenuItem>
            <asp:MenuItem Text="QC-2618">
                <asp:MenuItem NavigateUrl="~/Common_Tools/QC_Tool.aspx" Text="QC Tool"/>
            </asp:MenuItem>
        </Items>
    </asp:Menu>

```

CSS Class
 Main Menu CSS Class
 Sub Menu CSS Class
 Link Main Menu
 Tells Code This Menu's Orientation Will Be Horizontal
 Sub Menu's

Figure 36 – Navigation Bar ASPX Code

In figure 22 it shows the ASPX code for the navigation bar and as can be seen above this shows that there are four main menu's and three of which have sub menu's in them and in the case of the customer data tools they have sub menu's inside the initial sub menu.

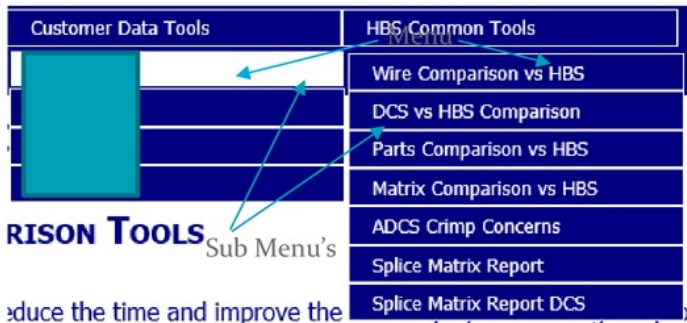


Figure 37 - Navigation Bar Sub Menu's

In figure 23 it shows that the first set of sub menu's detail each of the customer's name and inside those sub menu's it displays each of the tools that have been created for them. Once the user interface was completed and meets the company standard it's time to look to create the wire comparison tool.

6.2 DESIGNING CUSTOMER WIRE COMPARISON TOOL

The first step in implementing the wire comparison tool is to start with the ASPX page, which at first requires deciding what data fields are required in this case it requires three file upload controls with a button that will be used to execute the code.

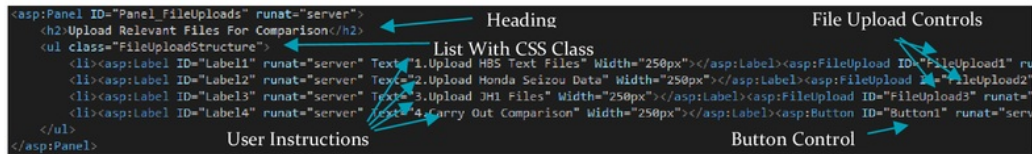


Figure 38 - Adding Required Control Fields

In figure 24 it shows the code added for the required control fields, which are all put into a list with labels assigned to then explaining to the user what needs to be entered into each of the required field.

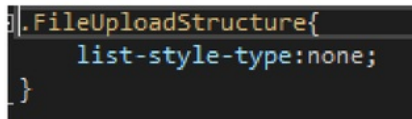


Figure 39 - CSS Class For Lists

In figure 25 it removes the bullet point from the list that has been coded into the ASPX page for the customer wire list comparison.

UPLOAD RELEVANT FILES FOR COMPARISON



Figure 40 - ASPX Page With Relevant Control Fields

In figure 26 it shows the display created through the ASPX code and the CSS class assigned to it. Now that the relevant controls have been added the c# code needs to be written to carry out each task.

Good use of the software tools for visual application.

```
//For File Uploading and Creating Directories
string name = userName.ToString();
string FolderPath = Server.MapPath("~/Temp/");
FolderPath = FolderPath + "/" + name;
System.IO.Directory.CreateDirectory(FolderPath);
FolderPath = FolderPath + "/";
```

This can be quite tricky because the paths can be of different lengths depending how far the source is from main directory.

Figure 41 – Folder Path Code For Uploading Files

In figure 27 it shows the first part of the code that needs to be written which is to take care of the documentation and where on the server it is saved. The reason the files need to be saved on the folder is due to windows will not allow files to be altered through browser, which means they need to be saved onto server where the program can use these files. In order to do this it requires creating a “Temp Folder” within the project. Every time a tool is used by the engineer it will create a folder directory in the “Temp Folder” with the name of their username.

```
//Retrieves Users Username
public string userName = HttpContext.Current.Request.ServerVariables["LOGON_USER"].Replace("SEWS-E\\", "");
```

Figure 42 - Code To Retrieve Username

In figure 28 it shows the code that is being used to get the engineers username when they are using the wire comparison tool. The reason individual folders are created for each user is to make sure there are no duplicate files saved and that no files are deleted before they should be.

```
//Additional Folder Paths For File Uploads
string FolderText = FolderPath;
string Folder[redacted] = FolderPath;
string FolderJH1 = FolderPath;
Customer DataTable
// [redacted] DataTable
DataTable [redacted] = new DataTable();
```

Figure 43 - Declaring Variables

In figure 29 it shows the variables being declared for each folder paths as new folder directories will be created later inside the folder of the engineers username and a datatable is created to hold the customer information. The next step in the process is to save all the text files into their correct directory's.

```

try
{
    //Step 1: Need to Deal with Text Files
    //Gets Text Files Selected By User
    string TextFiles = FileUpload1.PostedFile.FileName;
    FolderText = FolderText + "Text_Files";
    Directory.CreateDirectory(FolderText);

    //Loops Through The Posted Files Selected In FileUpload 1 To Save In Temp Folder On Server
    foreach (HttpPostedFile postedFile in FileUpload1.PostedFiles)
    {
        //Saves Files To Temporary Folders
        string fileName = Path.GetFileName(postedFile.FileName);
        postedFile.SaveAs(FolderText + "/" + fileName);
    }

    //Step 2: Saving Seizou Data To Directory Created Called Honda
    string HondaFile = FileUpload2.PostedFile.FileName;
    FolderHonda = FolderPath + "Honda";
    Directory.CreateDirectory(FolderHonda);
    FileUpload2.SaveAs(FolderHonda + "/" + FileUpload2.FileName);

    Honda = Honda_Functions.Honda(FolderHonda + "/" + FileUpload2.FileName);

    //Step 3: Saving JH1 Files To Directory Created Called JH1
    string JH1File = FileUpload3.PostedFile.FileName;
    FolderJH1 = FolderJH1 + "JH1";
    Directory.CreateDirectory(FolderJH1);

    //Loops Through The Posted Files Selected In FileUpload 3 To Save In Temp Folder On Server
    foreach (HttpPostedFile postedFile in FileUpload3.PostedFiles)
    {
        //Saves Files To Temporary Folders
        string fileName = Path.GetFileName(postedFile.FileName);
        postedFile.SaveAs(FolderJH1 + "/" + fileName);
    }
}

```

Try Catch Method To Pick Up Any Errors That Occur At This Point In The Code

Adding the name "Text Files" to the end of the current Folder Path. As that will be the name of the new folder directory created.

Section 1
Saving Text Files To Text File Directory

Section 2
Saving Honda Files to Customer Directory Created

Section 3
Customer File Content Added To Datatable

Section 4
Creating Jh1 Files folder directory

Section 5
Saving Jh1 Files into JH1 File Folder Directory

Figure 44 - Saving Engineering Files

In figure 30 in section 1 the way that the text files are saved to the text file directory is through using a for each loop that will go through each of the text files selected in the first file upload control and save them to the text files directory. In section 2 it saves the customer file to the customer directory and then in section 3 the file contents of the customer data is added into a datatable. In section 4 it creates the file directory for the JH1 files and then in section 5 it uses a for each loop to save each of the files selected in the third file upload controls.

There is evidence of substantial development work here.

```

public static DataTable Honda(string filePath)
{
    OleDbConnection cnn = new OleDbConnection(@"Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" + (filePath)
        + ";Extended Properties=\\"Excel 12.0 xml; HDR=No; IMEX=1;\\");
    OleDbCommand oconn = new OleDbCommand("select * from [Summary View$]", cnn);
    cnn.Open();
    OleDbDataAdapter adp = new OleDbDataAdapter(oconn);
    System.Data.DataTable dtIHS = new System.Data.DataTable();
    adp.Fill(dtIHS);
    cnn.Close();
    return dtIHS;
}

```

Inputs

Outputs

Excel Worksheet Name

Returns Data Table

Figure 45 - Function To Move Customer Data Into A Data Table

In figure 31 it shows the function that is used to assign the contents of the customer data into a datatable so that it can be compared later on using a linq query. The way this function works is a filepath needs to be inputted and then using that filepath it will look in the customer data excel file for the worksheet with the name "Summary View" and it will then output it as a datatable.

```

catch (Exception)
{
    //
    Honda_Functions.DeleteFolder(FolderPath);
}

```

Figure 46 - Catch Part Of Try Catch

In figure 32 it shows what will happen in the case that an error occurs when any of the files selected in the file upload controls and in this section it will delete the folders in the users directory. On completion I would look to add possible solutions to fix any of the errors they may have occurred in this process.

```

public static void DeleteFolder(string folderPath)
{
    /*
     * Method to delete the specified folder and all associated files
     */

    // Delete folder including all files and sub-folders
    if (Directory.Exists(folderPath) == true)
    {
        Directory.Delete(folderPath, true);
    }
}

```

Inputs

Void Means No Outputs

Figure 47 - Delete Folder Function

In figure 33 it shows the function for deleting any of the folders the only thing it requires is one input which is the folder path.

```

//Step 4: Sort Through HBS Document
//Create Datatable For HBS Document
DataTable HBS = new DataTable();
try
{
    //4.1 Will Add Column With Name Wire End In Order To Use For Comparison Later
    Honda_Functions.AddColumn(HBS, "Wire End", 0);
}
}

```

Creates Data Table

Try Catch Function

Variables To Be Used In Part 2

//4.2 Variables to count row and Column withing loop

int ColumnCount = 1;

int row = 0;

Data Table Column Name Column Location

Figure 48 - Sort Through HBS Text Files (Part 1)

In figure 34 it shows the first part of the code that will be used to sort through all the text files saved to the directory called "Text Files". As can be seen above it starts by creating a new data table to hold all the HBS information and following this a new try catch function needs to be used in case an error occurs at this point in the code. Within the try catch a new column needs to be added to the table using a function that has been created in C sharp class.

```

public static void AddColumn(DataTable table, string ColumnName, int ColumnLocation)
{
    table.Columns.Add();
    table.Columns[ColumnLocation].ColumnName = ColumnName;
}

```

Three Inputs No Outputs

Adds Column To Data Table

Changes Column Name Of Recently Added Column

Figure 49 - HBS Add Column Function

```

//4.5 Loops Through Each Line Within The Text Files Selected ← Section 1
//By The User
foreach (string line in System.IO.File.ReadLines(fileName))
{
    //4.6 Used To Split Up line Using Denominator ;
    lines = line.Split(';');
    //4.7 Switch Statement For Line Value To Determine Action To Be Taken ← Section 2
    switch (lines[0].ToString())
    {
        //4.8 In Case Value is Equal To 0 Will Then Add Column And Column
        //Name Of The Derivative In The Text File
        case "0":
            //4.9 Adds Columns And Assigns Derivative Name For Column Name ← Section 3
            Honda_Functions.AddColumn(HBS, lines[2], ColumnCount);
            //4.10 Adds Onto The Current Column
            ColumnCount = ColumnCount + 1;
            break;

            //4.11 In Case Value Is Equal To 2 then Will start To Add Wire No's
            //Into The Datatable
            case "2":
                //4.12 Makes Sure No Wire No's Are Added When They Start With ← Section 3
                //As They Are Not Needed
                if (!lines[5].ToString().StartsWith(""))
                {
                    //4.13 Adds Two Rows Into HBS DataTable
                    Honda_Functions.AddRow(HBS, 2);
                    //4.14 Adds 0 To Indicate Two Wires Are In This Derivative
                    HBS.Rows[row][ColumnCount - 1] = "0";
                    HBS.Rows[row + 1][ColumnCount - 1] = "0";

                    //4.15 Adds Two Wire Ends Into DataTable
                    HBS.Rows[row][0] = lines[5];
                    HBS.Rows[row + 1][0] = lines[6];

                    //4.16 Adds 2 To Row Variable As Two More Rows Have Been Added
                    row = row + 2;
                }
                break;
    }
}

```

Figure 50 - Sorting HBS Files

In figure 36 section 1 that has been labelled shows the start of a for each loop that is used to loop through a text file to retrieve the wire information associated. Each line within the text files can be split up using the denominator “;” for each line. Along with section 2 that displays the switch statement used for the first value in the first column of the line in the case that its value is equal to “o” two columns will be added to the HBS data table and the

column names are come from the second column within that line. In the case that the value in column 1 is equal to 2 then aslong as the wire no's do not begin with "*" then two rows are added to the table with the values "O" assigned to the recently added columns to show they are used in that specific derivative. Following this the wire no's are added into the data table.

```
//4.17 Removes All DUPLICATE Wire Ends And Creates One Master Spec
//For Each Wire Found
Honda_Functions.SortHBS(HBS);
```

Figure 51 - Removes Duplicate Wire Ends

In figure 37 a function created to remove all duplicate wire ends to create one master specification for each Is used and the function for this can be seen below in figure 38.

```
public static void SortHBS(DataTable table)
{
    for (int i = 0; i < table.Rows.Count; i++)
    {
        string WireEnd1 = table.Rows[i][0].ToString();
        for (int i2 = i + 1; i2 < table.Rows.Count; i2++)
        {
            string WireEnd2 = table.Rows[i2][0].ToString();

            if (WireEnd1.Equals(WireEnd2))
            {
                for (int i3 = 1; i3 < table.Columns.Count; i3++)
                {
                    string Column1 = table.Rows[i][i3].ToString();
                    string Column2 = table.Rows[i2][i3].ToString();

                    if (Column2.Equals("0"))
                    {
                        table.Rows[i][i3] = "0";
                    }
                }

                table.Rows.RemoveAt(i2);
                i2 = i2 - 1;
            }
        }
    }
}
```

Inputs
No Outputs
Section 1
Section 2
Section 3
Section 4
Section 5

Figure 52 - Sort HBS Function

In figure 52 It shows the function for removing all the duplicate wires and creating one master spec for each the reason this is important is at this point the HBS data has multiple duplicate wire ends with different derivatives associated as can be seen in figure 53. In section 1 it shows that a for loop that has been created to look through the data table that

has been input into the function for each row in the table. In section 2 the first thing it does is find the wire no and then begin another look through the rows in the data table to try to find any duplicates in the data table. In section 3 it checks if a match has been found and if that is the case it will look through the columns in the data tables and then if the variable column2 equals "o" it means that it has a specification in that column and it will be added to the first wire no's row. Before in section 5 it removes the second row from the table and changes the count variable - 1 since 1 row has been deleted.

Wire No	32107-TES-H011	32107-TEC-B211	32107-TEJ-B301
SS5C	O		
SS5C		O	
SS5C			O

Figure 53 - HBS Data Looks At This Point

Wire No	32107-TES-H011	32107-TEC-B211	32107-TEJ-B301
SS5C	O	O	O

Figure 54 - Master Wire No

In figure 53 it shows how the HBS data looks before the Sort HBS function is used as can be seen SS5C comes up three times and each have a different associated derivative and they need to be put together into 1 master wire no as can be seen in figure 54.

```
//Step 5: Make Use Of JH1 Files TO Deal With Sub Assemblys Within HBS
//Output
try ← Try Catch Function
{
    //5.1 Loops Through All Jh1 Files Saved Into Directory Created Previously
    //FolderJh1
    foreach (string fileName in Directory.GetFiles(FolderJH1))
    {
        //5.2 Variables to count row and Column withing loop ← Section 1
        int check = 0;
        string[] lines = new string[check];

        //5.3 Loops Through Each Line Within The JH1 Files Selected
        //By The User
        foreach (string line in System.IO.File.ReadLines(fileName)) ← Section 2
        {
            //5.4 Used To Split Up line Using Denominator ;
            lines = line.Split(';');
        }
    }
}
```

Figure 55 - Making Use of Sub Assembly Files

In figure 55 it shows the beginning of the code in order to deal with all of the sub assembly's associated with the text files previously sorted. In section 1 of figure 55 it first includes a for each loop to go through each JH1 file that has been saved to the file directory previously created for the JH1 files. Then an integer is created called check that will be used later in the process along with a string array to hold each line in an individual JH1 file. In section 2 it shows the foreach look that goes through each line within the Jh1 file and then the line is split up into columns using the split function with the ";" that splits the line.

```
if (lines[6].Equals("20"))
{
    //5.6 Loops Through Every Row Within HBS DataTable
    for (int i = 0; i < HBS.Rows.Count; i++)
    {
        //5.7 Determines Whether JH1 File Is Used For Molde Wires Before
        //Determining The Column Name To Be Used In Comparison Function
        string column = (lines[1].Substring(0, 4).Equals("MOLD")) ? column = lines[1] + lines[2] : lines[1] + " " + lines[2];
        var Type = Honda_Functions.Comparison(HBS, HBS.Rows[i][0].ToString(), "Wire End", "0", column);
    }
}
```

Section 1 points to the for loop starting at line 5.6.

Section 2 points to the inner logic starting at line 5.7.

Figure 56 - 2nd Part Of Taking Care Sub Assembly's

In figure 56 it shows the second part of the code to sort the sub assembly's it begins with checking the line that has been split into individual columns in figure 55 that column 6 in the line is equal to 20 before proceeding with the loop for the HBS data Table. The for each loop will go through all of throws in the HBS data table created from the text files. In Section 2 the variable called column will be dependent on whether the sub assembly is a molde wire or not. Following this a linq query is used to compare the columns.

```
if (Type == true)
{
    //5.9 Loops Through All Columns Within The HBS DataTable
    for (int i2 = 1; i2 < HBS.Columns.Count; i2++)
    {
        //5.10 Loops Through Each String In Column Name From The Fifth
        //String In The Column Name
        for (int i4 = 5; i4 < HBS.Columns[i2].ColumnName.Length; i4++)
        {
            //5.11 Checks That String Is An Integer
            if (int.TryParse(HBS.Columns[i2].ColumnName.Substring(i4, 1), out check))
            {
                //5.12 Checks That Column Name Substring Is Equal To The Value Within
                //JH1 File At Column 8 Within The Jh1 File
                if (HBS.Columns[i2].ColumnName.Substring(0, i4 + 3).Equals(lines[8]))
                {
                    HBS.Rows[i][i2] = "0";
                }
            }
        }
        i4 = 35;
    }
}
```

Section 1 points to the for loop starting at line 5.9.

Section 2 points to the inner logic starting at line 5.10.

Figure 57 - 3rd Part Of Taking Care Sub Assembly's

In figure 57 it takes checks if the result of the linq query returns true and if that the case it starts to loop through each column in the HBS data table starting at column 1. In section 2 it then looks through each character in the string starting at 5 until it reaches a character

that is numeric. The reason for this is due to the difference between the format of the derivatives between our data and the customer data. Following this it will then add the spec in for that wire no as if it gets to this point it means this derivative needs to be in for this wire no.

```
//5.13 Loops Through All Of The Columns Within The HBS DataTable
for (int i = 0; i < HBS.Columns.Count; i++)
{
    //5.14 Checks That Column Name is A Sub Assembly Or A
    if ((HBS.Columns[i].ColumnName.Substring(0, 2).Equals("77")) || (HBS.Columns[i].ColumnName.Substring(0, 4).Equals("MOLD")))
    {
        Honda_Functions.DeleteColumns(HBS, 2, i);
        i = i - 1;
    }
    else
    {
        for (int i2 = 0; i2 < HBS.Rows.Count; i2++)
        {
            if (HBS.Rows[i2][i].ToString().Equals(""))
            {
                HBS.Rows[i2][i] = "--";
            }
        }
    }
}
```

Figure 58 - Final Part Of Sub Assembly's

In figure 58 it removes all columns that are Molde or sub assembly derivatives.

```
//7.0 Carry Out Comparison
try
{
    Honda_Functions.AddComparisonColumns(HBS, HBS.Columns.Count, 1);
    Comp = HBS.Clone();

    int CompRows = 0;
    int AddRows = 0;
    for (int row = 0; row < HBS.Rows.Count; row++)
    {
        var WireCheck = Honda_Functions.ComparisonColumn1(Honda, HBS.Rows[row][0].ToString(), "Wire");
        HBS.Rows[row][1] = WireCheck;
        if (WireCheck == true)
        {

```

Figure 59 – Comparison

In figure 59 it shows the beginning of the code for the comparison that starts with creating a new data table called comp and the reason for this is so that it can show the wire no and its associated derivatives next to eachother as can be seen in 13. Following this it runs a for loop that will loop through each row in HBS data table and check that the first wire no from HBS table is in the customer data. If that is the case it will then move onto the next step that can be seen in figure 60.

```

for (int column = 2; column < HBS.Columns.Count; column = column + 2)
{
    var result = Honda_Functions.Comparison(Honda, HBS.Rows[row][0].ToString(), "Wire", (HBS.Rows[row][column].Equals("0")) ? "o" : "-",
    HBS.Columns[column].ColumnName);
    HBS.Rows[row][column + 1] = result;
}

Comp.Rows.Add(HBS.Rows[row].ItemArray);
Comp.Rows.Add(HBS.Rows[row].ItemArray);

Honda_Functions.ChangeFalseValues(Comp, CompRows + 1);

CompRows = CompRows + 2;

```

Figure 60 - Comparison Part 2

In figure 60 it shows what happens if the wire no in HBS is found within the customer data. The first thing it does is start a for loop to go through each of the HBS columns and to check that for that wire no the derivatives present are the same and then adds a true or false to that. This will then be added to the comp data table twice to represent one from HBS and one from the customer data. Then the change false value function I created will go through and each row and column looking for a value returned as false and change the derivative value next to it accordingly to represent the difference between customer data and the HBS.

```

GridView1.DataSource = Comp;
GridView1.DataBind();

```

Figure 61 - Display the comp data table on screen

In figure 61 it shows the comparison data table on screen for the user the only task left is to highlight the differences accordingly as has been done in figure 62.

```

protected void OnRowDataBound(object sender, GridViewRowEventArgs e)
{
    try
    {
        for (int i = 0; i < TotalColumnCount; i = i + 2)
        {
            if (e.Row.Cells[i + 1].Text == "True")
            {
                e.Row.Cells[i].ForeColor = Color.DarkGreen;
            }
            else if (e.Row.Cells[i + 1].Text == "False")
            {
                e.Row.Cells[i].ForeColor = Color.Red;
            }
            else if (e.Row.Cells[i].Text == "N/A")
            {
                e.Row.Cells[i].ForeColor = Color.Orange;
            }
            e.Row.Cells[i + 1].Visible = false;
        }
    }
    catch (Exception)
    {
    }
}

```

Figure 62 - Highlight Discrepancies

*Your breakdown
of explanation is
excellent. This is a*

Now that the discrepancies have been highlighted the complete example can be seen in figure 63.

COMPARISON HBS AND HONDA MATCHING WIRE ENDS

KEYS

Red = Difference / No Match
Green = Match
HBS = White
Honda = Blue

VIEW ALL SPECIFICATIONS OR JUST MISMATCHES

Mismatches Only

Show All Wire Ends

Comparison Report of: HBS:32107-TEA-ZZ14MP 14 0002 & Cust:TEAY_4D_DAN_APD_HARNESS_UH-32107-TEA -ZZ13-20170619_160404-connector_list.xlsx

COMPLETE COMPARISON

Wire	32107-TEC-B011-F 14 0002	32107-TEC-B111-F 14 0002	32107-TEC-B211-F 14 0002	32107-TEJ-B111-F 14 0002	32107-TEJ-B211-F 14 0002	32107-TEJ-B301-F 14 0002	32107-TEK-M011-F 14 0002	32107-TEK-P011-F 14 0002	32107-TEL-W211-F 14 0002	32107-TEL-W511-F 14 0002	32107-TEF-M011-F 14 0002	32107-TEF-M101-F 14 0002	32107-TEF-H011-F 14 0002	32107-TEF-H111-F 14 0002	32107-TEF-H011-F 14 0002	32107-TEF-H211-F 14 0002
804A	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
804A	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
804B	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
804B	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 63 - Example Of Completed Comparison

In figure 63 it shows a completed comparison and the user can select to only view the mismatches as can be seen in figure 64 and 65.

COMPARISON HBS AND HONDA MATCHING WIRE ENDS

KEYS

Red = Difference / No Match
Green = Match
HBS = White
Honda = Blue

VIEW ALL SPECIFICATIONS OR JUST MISMATCHES

Mismatches Only

Show All Wire Ends

Comparison Report of: HBS:32107-TEA-ZZ14MP 14 0002 & Cust:TEAY_4D_DAN_APD_HARNESS_UH-32107-TEA -ZZ13-20170619_160404-connector_list.xlsx

COMPLETE COMPARISON

Wire	32107-TEC-B011-F 14 0002	32107-TEC-B111-F 14 0002	32107-TEC-B211-F 14 0002	32107-TEJ-B111-F 14 0002	32107-TEJ-B211-F 14 0002	32107-TEJ-B301-F 14 0002	32107-TEK-M011-F 14 0002	32107-TEK-P011-F 14 0002	32107-TEL-W211-F 14 0002	32107-TEL-W511-F 14 0002	32107-TEF-M011-F 14 0002	32107-TEF-M101-F 14 0002	32107-TEF-H011-F 14 0002	32107-TEF-H111-F 14 0002	32107-TEF-H011-F 14 0002	32107-TEF-H211-F 14 0002
A32A	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A32A	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 64 - Wire With Mismatch

This test is a little confusing. What is the mismatch in figure 64

<u>32107- TFJY- G301-F 14 0002</u>	<u>32107- TFJY- G701-F 14 0002</u>	<u>32107- TFK- G010-F 14 0002</u>	<u>32107- TFK- G110-F 14 0002</u>	<u>32107- TFK- G210-F 14 0002</u>
0	0	0	0	0
0	0	0	-	0

Figure 65 - The discrepancy

In figure 64 it shows the wire no that the discrepancy in data is at the reason they are not in the same screenshot is due to the amount of derivatives making it impossible. In figure 65 it shows that we have a derivative in where the customer doesn't not which means the engineer needs to go investigate this problem. Now the project can be implemented on the intranet.

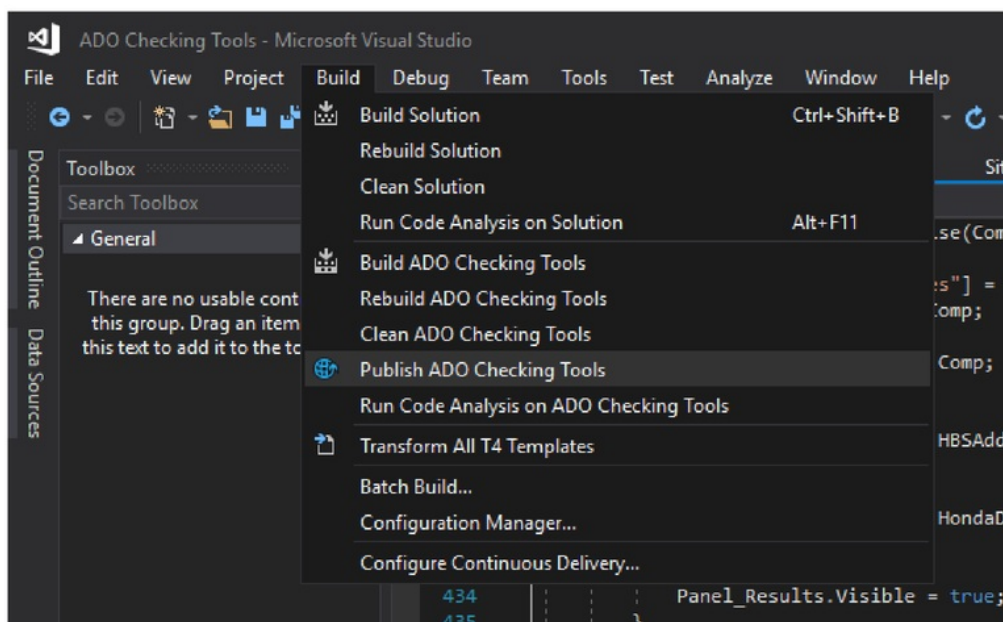


Figure 66 - Publishing Project

In figure 66 the first step in publishing the project is to go to the build tab then to select publish.

Publish

Publish your app to Azure or another host. [Learn more](#)

ADO Checking Tools ▼ Publish

[Create new profile](#)

Summary

Target Location	[Redacted]	Settings...
Delete Existing Files	False	Rename profile...
Configuration	Release	Delete profile

Continuous Delivery

Automatically publish your application to Azure with continuous delivery

Start

Figure 67 - Publishing Project 2

In figure 67 it shows the next step which is to enter the target location for the project which in this case we use a folder structure on our server.

```
2>Publishing folder Scripts/WebForms/MSAjax...
2>Web App was published successfully [Redacted]
2>
===== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped =====
===== Publish: 1 succeeded, 0 failed, 0 skipped =====
```

Figure 68 - Project Published

7. Testing and Results

In this section it will look at the agreed upon testing of the comparison tool and the results that we gained from the table to see whether it meets the specification set out initially.

7.1 AGREED UPON TESTING TABLE

Test No	Test Description	Expected Results
1	Simulate a Discrepancy In HBS Text File	Discrepancy Should be present
2	Simulate a Discrepancy In Honda Text File	Discrepancy Should be present
3	A Missing Wire In Honda	Wire Should Be Displayed As Missing
4	All File uploads to temp folder must be deleted.	File Uploads Deleted
5	Show All Wires	Shows All Wires
6	Show Discrepancies Only	Shows Only Discrepancies
7	Export As An Excel Document	Exports As An Excel Document

7.2 TESTING RESULTS

Test No	Test Description	Expected Results	Actual Results	Figure No
1	Simulate a Discrepancy In HBS Text File	Discrepancy Should be present	Discrepancy were present	69, 70
2	Simulate a Discrepancy In Honda File	Discrepancy Should be present	Discrepancy were present	71, 72
3	A Missing Wire In Honda	Wire Should Be Displayed As Missing	Wire were Displayed As Missing	73
4	All File uploads to temp folder must be deleted.	File Uploads Deleted	File Uploads were Deleted	74
5	Show All Wires	Shows All Wires	Showed All Wires	75
6	Show Discrepancies Only	Shows Only Discrepancies	Showed Only Discrepancies	72
7	Export As An Excel Document	Exports As An Excel Document	Exported As An Excel Document	76

Test No 1



```
2;32107-TEA-ZZ14MP 14 0002;009E;2300;32107-TEC-0011-F 14 0002;804A;804B;;2;156;500;;;ATW; 050;;Y;10;0.50;3703.2;32107-TEA-ZZ14MP 14 0002;009E;2300;32107-TEC-0011-F 14 0002;805A;805B;;3;156;500;;;ATW; 050;;BR;10;0.50;3703
```

Figure 69 - Deleted Wire From Text File

In figure 69 it shows the wire no's that are going to be deleted from the derivative no "32107-TEA-ZZ14.

Wire	<u>32107- IEC- B011-F</u> <u>14</u> <u>0002</u>	<u>32107- IEC- B111-F</u> <u>14</u> <u>0002</u>
A32A	0	0
A32A	0	0
804A	-	0
804A	0	0
804B	-	0
804B	0	0

This is easier to understand and much clearer as to what you are testing for.

Figure 70 - HBS Derivative Missing

In figure 70 it shows the tool has found that the wire No's 804A and 804B are both missing from our internal data which is what was expected since they were deleted from the text file associated with that derivative. The reason A32A is included in screenshot 70 is due to that having a discrepancy elsewhere.

Test 2

[illegible]

Figure 71 - Change In Customer Data

In figure 71 it shows for wire end A32A I have changed the derivative for 32107-TEC-B01 from present to its not.

Wire	32107- TEC- B011-F 14 0002	32107- TEC- B111-F 14 0002	32107- TEC- B211-F 14 0002	32107- TEJ- B111-F 14 0002	32107- TEJ- B211-F 14 0002	32107- TEJ- B301-F 14 0002
A32A	0	0	0	0	0	0
A32A	-	0	0	0	0	0

Figure 72 - Honda Derivative Missing

In figure 72 shows that the wire is present in the derivative for our internal data but not for the customer data which is what was expected since it was removed from the customer data.

Test 3

ADDITIONAL WIRES - HBS VS HONDA	
Wire	
JJJ	
646D	

Figure 73 - Additional Wire

In figure 73 it shows that there are two wires in HBS that are not in customer which is what was expected since I removed them from the customer data.

Test 4

This PC > daviesdS > Projects > ADO Checking Tools > ADO Checking Tools > Temp				
Name	^	Date modified	Type	Size
This folder is empty.				

Figure 74 - Empty Temp Folder

In figure 74 it shows that the temp folder is empty after running the comparison tool which is what was expected.

VIEW ALL SPECIFICATIONS OR JUST MISMATCHES

Comparison Report of: HBS:32107-TEA-ZZ14MP 14 0002 & Cust:TEAY_4D_DAN_APD_HARNESS_LH-32107-TEA-ZZ13-20170619_160404-connector_list.xlsx

COMPLETE COMPARISON

Wire	32107- HIC- B011-F 14 0002	32107- HIC- B111-F 14 0002	32107- HIC- B211-F 14 0002	32107- HEL- B111-F 14 0002	32107- HEL- B211-F 14 0002	32107- HEL- B301-F 14 0002	32107- HIC- M011-F 14 0002	32107- HIC- P011-F 14 0002	32107- HEL- W211-F 14 0002	32107- HEL- W511-F 14 0002	32107- HEM- M011-F 14 0002	32107- HEM- M101-F 14 0002	32107- HIS- H011-F 14 0002	32107- HIS- H111-F 14 0002	32107- HEL- H211-F 14 0002	32107- HEL- H311-F 14 0002	32107- HEL- H411-F 14 0002	32107- HEL- H102-F 14 0002
804A	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
804A	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
804B	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
804B	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
805A	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
805A	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
805B	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
805B	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
806B	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
806B	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 75 - Show All Wires

ADDITIONAL WIRES - HBS VS HONDA

Wire
JJJ
646D

ADDITIONAL WIRES - HONDA VS HBS

Additional Wires Customer Data : None

OUTPUT RESULT TO EXCEL FILE

Export To Excel

Do you want to open or save Output.xls (1.24 MB) from ukhgeng001?

Figure 76 Export Excel

8. Discussion and Conclusions

The aim of this project was to build a tool for the engineers on the company's intranet system which has been achieved as the system is now active and being used on the company's intranet. This tool has significantly reduced the amount of time it takes to carry out a comparison freeing engineers up to be able to manage their time more efficiently.

This has also significantly reduced the risk of human error when checking for any discrepancies and has been a big help in picking up issues so the problem can be solved before reaching the customer.

The tool only take a couple of minutes at most to compare the customer data to our own with the biggest files which is a significant improvement on the manual check as that could take up to a couple of days.

The comparison tool also gives the engineers the ability to view every wire end or only the ones with discrepancies while also highlighting the discrepancies in red and highlighting the ones as no issue in green.

This could be more comparative with regards to user friendliness i.e. how does it stand up in the real world application.

9. Reference List and Bibliography

HP Development Company, 2016. *h20564.www2.hp.com*. [Online]
Available at: <http://h20564.www2.hp.com/hpsc/swd/public/readIndex?sp4ts.oid=5232700>
[Accessed 07 April 2016].

AM Labels, 2014. *www.amlabels.co.uk*. [Online]
Available at: <https://www.amlabels.co.uk/505/citizen-clp-2001-barcode-label-printer>
[Accessed 11 April 2016].

Applegate Marketplace Ltd, 1996-2016. *www.applegate.co.uk*. [Online]
Available at: <https://www.applegate.co.uk/b2b-products-services/automation24-gmbh/simatic-basic-panel-siemens/3130046%20366-3130046.html>
[Accessed 11 April 2016].

BARCODE.CO.UK, 2016. *www.barcode.co.uk*. [Online]
Available at: <http://www.barcode.co.uk/products/citizen-clp2001-thermal-label-printer.htm>
[Accessed 11 April 2016].

Chong, A., n.d. *www.infineon.com*. [Online]
Available at: http://www.infineon.com/export/sites/default/cn/drivingasia_chap1.pdf
[Accessed 29 March 2016].

computeruniverse GmbH Germany, 1999-2016. *www.computeruniverse.net*. [Online]
Available at: <https://www.computeruniverse.net/en/products/90499309/hp-compaq-pro-6300.asp>
[Accessed 07 April 2016].

divmac.pt, n.d. http://divmac.pt/wp-content/uploads/2015/08/Apresentacao-Divmac-EN_web.pdf. [Online]
Available at: http://divmac.pt/wp-content/uploads/2015/08/Apresentacao-Divmac-EN_web.pdf
[Accessed 06 April 2016].

Emdep, 2007. <http://www.emdep.com/en/empresa.asp>. [Online]
Available at: <http://www.emdep.com/en/empresa.asp>
[Accessed 6 April 2016].

EMDEP, 2007.
http://www.emdep.com/en/productos_produccion.asp?metacatid=3&catid=8&prodid=40.
[Online]
Available at:
http://www.emdep.com/en/productos_produccion.asp?metacatid=3&catid=8&prodid=40
[Accessed 06 April 2016].

EMDEP, 2007.

http://www.emdep.com/en/productos_produccion.asp?metacatid=3&catid=8&prodid=40.

[Online]

Available at: <http://www.emdep.com/img/productos/3ESWT.jpg>

[Accessed 06 April 2016].

Generation Robots, n.d. www.generationrobots.com. [Online]

Available at: <http://www.generationrobots.com/media/manuel-plc-siemens-s7-en.pdf>

[Accessed 11 April 2016].

HP Development Company, L.P, 2016. store.hp.com. [Online]

Available at:

<http://store.hp.com/UKStore/Merch/Product.aspx?id=C9E49AA&opt=ABU&sel=DEF>

[Accessed 11 April 2016].

Isuzu Motor America LLC, 2016. www.isuzuengines.com. [Online]

Available at: <http://www.isuzuengines.com/Product.aspx?series=UM>

[Accessed 29 March 2016].

National Instruments Corporation, 2014. www.ni.com. [Online]

Available at: <http://www.ni.com/datasheet/pdf/em/ds-335>

[Accessed 13 April 2016].

National Instruments, 2016. sine.ni.com. [Online]

Available at: <http://sine.ni.com/cs/app/doc/p/id/cs-549#>

[Accessed 13 April 2016].

National Instruments, 2016. www.ni.com. [Online]

Available at: <http://www.ni.com/labview/buy>

[Accessed 13 April 2016].

Sumitomo Wiring Systems LTD, 2008. www.sws.co.jp. [Online]

Available at: <https://www.sws.co.jp/en/product/wireharness/>

[Accessed 29 March 2016].

www.conrad.com, n.d. www.conrad.com. [Online]

Available at: <http://www.conrad.com/ce/en/product/197494/Siemens-6ES7214-1HG31-oXBo-SIMATIC-S7-1200-CPU-1214C-Compact-CPU-DCDCRelay-14-DI-10-DO-2-AI-50Kb>

[Accessed 11 April 2016].

Very good reference list

10.Appendices

10.1 CAE INTERFACE STANDARD



Standard

CAE Development (User Interface)

CAE-ST-DEV-001

Version 1.00

04 Apr 2011

**© Sumitomo Electric Wiring Systems (Europe) Ltd
2011**

Version Control

Version	Primary Author(s)	Description of Version	Date Completed
Version 1.00	David Pope	Version 1.00 of CAE Development (User Interface) Standard CAE-ST-DEV-001.	04 Apr 2011

Amendment Details

With the exception of Version Control and Table of Contents, differences between the current version and the previous version of this document are identified by a vertical line to the right of the amendment (as shown here).

i

Table of Contents

Version Control.....i

Amendment Details.....i Table of Contents..... ii

Web Applications.....1

Layout.....1 1.1.1 Header..... 1

CAE Image.....1

Standard Links.....2 CAE Strap Line 2 Tag Image.....2

Breadcrumbs	
Trail.....	2
Footer	
.....	2
Standard	
Links.....	2
Copyright.....	
...3	
Menu	
.....	3
Application	
Name.....	3
Section	
Name.....	3
Link / Sub-Link	
.....	3
Content.....	
3	
Title	
.....	4
Content	
.....	4
Format.....	
.....4	
1.2.1	
Background.....	
4 1.2.2 Text	
.....	4
1.2.3	
Hyperlinks.....	
4 1.2.4	
Exceptions.....	
4	

Appendix A
.....
6

Web Applications Code
Sample.....6

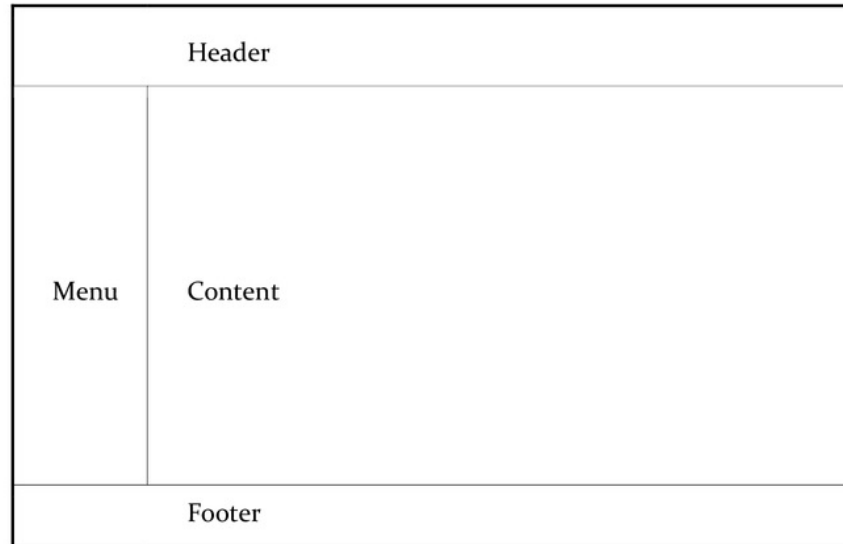
A.1 Header
.....6

1. Web Applications

The CAE User Interface Standard is intended to ensure a unified appearance for all CAE applications.

1.1 Layout

All CAE web applications should conform to the same general layout:



1.1.1 Header

The header section should be common to all CAE web applications:



Breadcrumbs Trail

CAE Image

The CAE image shows the European Technical Centre and provides a link to an associated web page:



To maintain continuity within the user interface for a web application (especially the web application menu), this standard file can be included within the web application rather than linking to the general file.

Standard Links

Standard links include:

CAE Systems

Contact Us

Terms of Use

Privacy Policy

To maintain continuity within the user interface for a web application (especially the web application menu), these standard files can be included within the web application rather than linking to the general files.

CAE Strap Line

A strap line is a branding slogan that illustrates the idea or concept behind a product. The CAE strap line is a general statement of the ethos of the CAE group:

FOR Engineers BY Engineers

Tag Image

The tag image shows the SEWS-E strap line “Connect with the Best”:



Breadcrumbs Trail

The breadcrumbs trail is a navigation aid that provides the user with information about the general structure of the web application with links to web pages higher in the structure.

- CAE Systems ➤ Web Site ➤ Web Page

Irrespective of the architecture of a web application, from a user interface perspective all CAE web applications are extensions of the general CAE Systems web site.

The web site name should be the same as the web application name.

The web page name should be the same as the title of the web page as shown in the content section.

1.1.2 Footer

The layout of the footer section should be common to all CAE web applications:

Standard links
Copyright

Standard Links

Standard links include:

CAE Systems

Contact Us

Terms of Use

Privacy Policy

Copyright

The copyright notice indicates the date or date range by year during which the web application has been under active development:

©2011 SEWS (Europe) Ltd

1.1.3 Menu

The general layout of the menu section should be common to all CAE web applications:

APPLICATION NAME
Link
Sub-Link
SECTION NAME
Link
Sub-Link

The menu comprises one or more headers each of which has one or more links to associated web pages. The main web page links should be grouped under a header named the same as the web application. Other headers may vary between web

applications but typically might include names such as Administration, Standards, and Documents.

Application Name

A header with the name of the web application.

Section Name

One or more headers with the name of any logical sections.

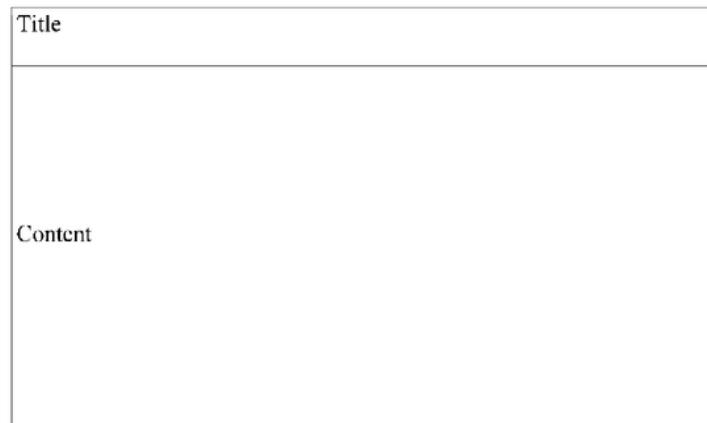
Link / Sub-Link

A link to a web page within the web application. The only difference between a link and a sub-link is a logical distinction to assist the user in understanding the structure of the web application.

A link element may include more than one hyperlink provided the logic can be understood by the user.

1.1.4 Content

The general layout of the content section should be common to all CAE web applications:



Title

The title of the web application followed by the title of the webpage.

Content

The content of the webpage.

1.2 Format

All CAE web applications should conform to a standard general format.

1.2.1 Background

The background of all web pages should be plain white.

It is permissible to include a background image covering part of the content section.

1.2.2 Text

All text should be coloured dark blue, typically Navy.

All text should use a sans-serif font, typically Tahoma, Verdana or Arial.

Text must not be underlined.

1.2.3 Hyperlinks

All hyperlinks should be coloured blue, except where a section background is coloured dark blue in which case hyperlinks should be white.

All hyperlinks must be underlined.

1.2.4 Exceptions

To provide contrast some elements of the user interface reverse the normal colour scheme for background and text. These elements include:

Header CAE Strap Line

Footer Standard Links

.....Copyright

Menu Application Name

.....Section Name

To highlight their functionality some elements of the user interface have a light blue background, typically Light Steel Blue. These elements include:

Header Breadcrumbs Trail

Appendix A

Web Applications Code Sample

A.1 Header

```
<div id="HeaderLogo">



</div>

<div id="Header">

  <div id="HeaderImage">

    <a href="//ukhqengs001/CAE/TheETC.aspx" onmouseover="status='CAE
    Systems : About the European Technical Centre at Ystradgynlais';
    return true;"

    onmouseout="status=''; return true;"></a>

  </div>

  <div id="HeaderLink" align="right">

<!--#INCLUDE FILE="caeLinkList.html"-->

  </div>

  <div id="TagLine">

<b>FOR</b> Engineers <b>BY</b> Engineers

  </div>

</div>
```

Assignment 2 - Full Report

GRADEMARK REPORT

FINAL GRADE

GENERAL COMMENTS

Instructor

56/100

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

Text Comment. Good introduction

PAGE 8

PAGE 9

PAGE 10

PAGE 11

Text Comment. There is always a cost incurred. In this case the cost of the time on the software and third party assistance is a cost. Compare this to the cost of outsourcing this project.

PAGE 12

Text Comment. Good explanation here

PAGE 13

Text Comment. This is quite a task to get a common format for comparison.

Text Comment. The design and implementation cost of a system like this could be quite substantial if outsourced. In this case the budget is the time of all persons connected and contributed to the completion of the project. This will come out as a figure.

Text Comment. Good plan

Text Comment. Good arguments. This could have included an example of an existing macro or other application as a case study. There are plenty of case studies available.

Text Comment. Good use of the software tools for visual application.

Text Comment. This can be quite tricky because the paths can be of different lengths depending how far the source is from main directory.

Text Comment. There is evidence of substantial development work here.

Text Comment. Your breakdown of explanation is excellent. This is a complex project to undertake and you have explained it well.

Text Comment. This test is a little confusing. What is the mismatch in fig 64

Text Comment. This is easier to understand and much clearer as to what you are testing for.

Text Comment. This could be more comparative with regards to user friendliness i.e. how does it stand up in the real world application.

Text Comment. Very good reference list

GRADING FORM: CSG-RUBIC

DECLAN DAVIES

0

RESEARCH-INFORMED LITERATURE



It is obvious that substantial research has been carried out to complete his project due to its complexity. However there are no in text references to indicate where these sources apply. This lets the research down a little. Introduction and background research - 6/10 References - 4/10 due to no in text references

10/20

KNOWLEDGE AND UNDERSTANDING



The specification is good and for a complex project such as this there would probably be many versions of the spec. Specification - 7/10 Solutions considered should have some reference to relevant sources for examples. However there are 3 feasible alternatives discussed. - 6/10 There is substantial detail and an argument for the final solution.

13/20

ANALYSIS



Substantial testing has taken place as there are so many stages to the project. It is also noted that each previous stage has to function correctly before the subsequent stage can be implemented. There is evidence of substantial testing and results analysis. Testing and results - 7.5/10 Conclusion - could have included an evaluation of user friendliness in the real world. - 6.5/10

14/20

PRACTICAL APPLICATION AND DEP.

Solutions considered were a little brief. A case study



for this could easily be included as a source.
Solutions considered - 2.5/5 Final solution has
substantial development that works and has been
implemented by the company- 7.5/10 Quality of
work is good - 3/5

13/20

SKILLS FOR PROFESSIONAL PRAC.



The report was quite involved due to the complexity
of the project, but you have explained the majority
of the development very well.- 6/10 Presentation -?

6/20