Tyler Whitezell
MBA 614 Spreadsheet Automation
Final Project Write-Up

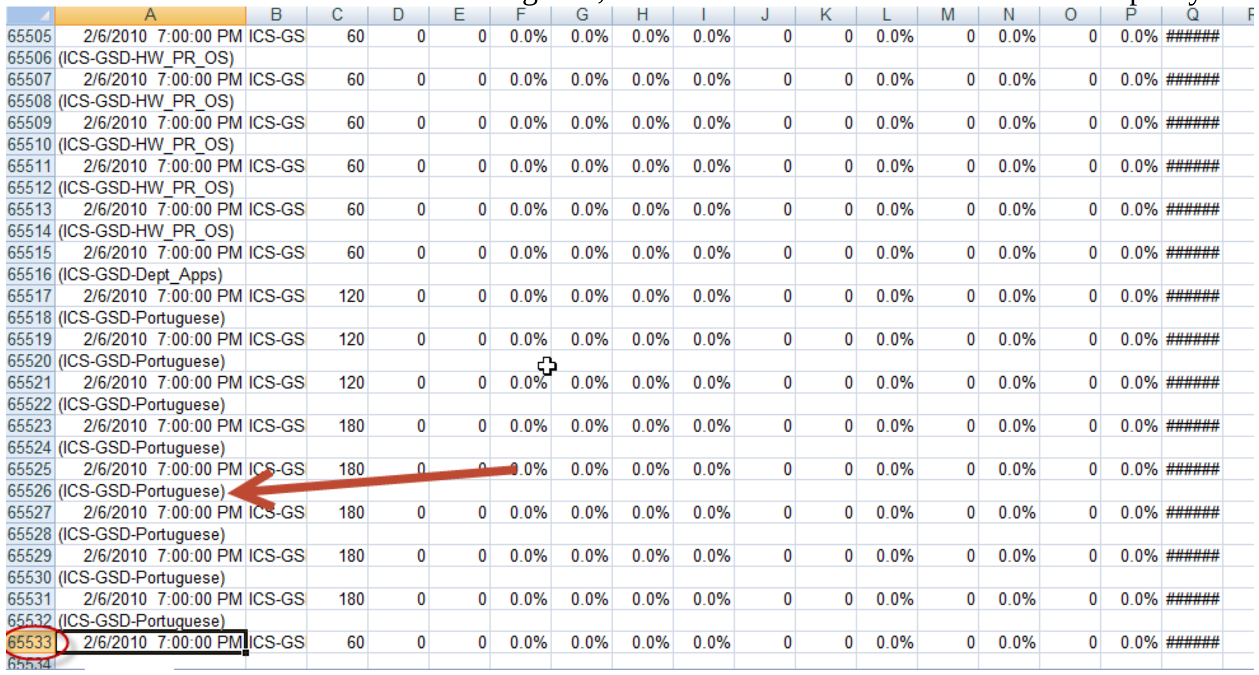
Executive Summary

For a few months I did computer support for the Global Service Desk (GSD). This was basically a call center for the Church of Jesus Christ of Latter Day Saints. The call center took a number of first and second level support calls from church employees, and others associated with the Church. While employed, the structure of the call center was examined, and changed to better meet demand. The call center went from having 3 main groups, general first level support, general second level support, and account management specific second level support, to having multiple focus groups, to better specialize employees and expedite call turnovers. It turned into a great idea. Second level support became a function of first level support, as these tasks became more integrated, and a simple phone tree allowed users calling in to be directed to the team that would best be able to resolve their issue. While this did improve efficiency, the company lost its initial ability to track phone calls. The company purchased software that would provide a report with details of the number of calls that were presented, the number of calls that were handled, the number abandoned, the number dequeued, as well as other data relating to these statistics. The problem with the report is that, in its raw form, it was unusable, which introduces the need for my project. Basically, my program takes the raw data from the report, and breaks it down into a format that allows you to get value out of the report, and in turn, is a very valuable tool for the GSD.

The report that is run is basically a data dump. It returns a number of sheets using the maximum amount of rows allowable. The problem is that every other row only contains 1 item, but that 1 item is crucial to identifying the row above, so it can’t simply be deleted. The first step of this program runs through all of the sheets on the page, inserts a column, and brings that one cell of data up to the row above, where it can be used as an identifier. It then deletes the row that this identifier came from, as it now has no more information. This data is further broken down, and redundant and excess data is removed. The program then sorts each of these rows of data based on the team that the call was directed toward; creating a new tab for each team that has data. The next phase of this program takes the sorted data, and inserts a pivot table into each of the worksheets, and from that pivot table, it creates charts, to better see the data. These charts help management to see which teams need to be more fully staffed, or teams that might be staffed too fully at different points in the day. The users have the option of entering different parameters that they would like to be graphed. This basically just changes what the pivot table charts, but allows for more flexibility for the parameters that wants to look at.

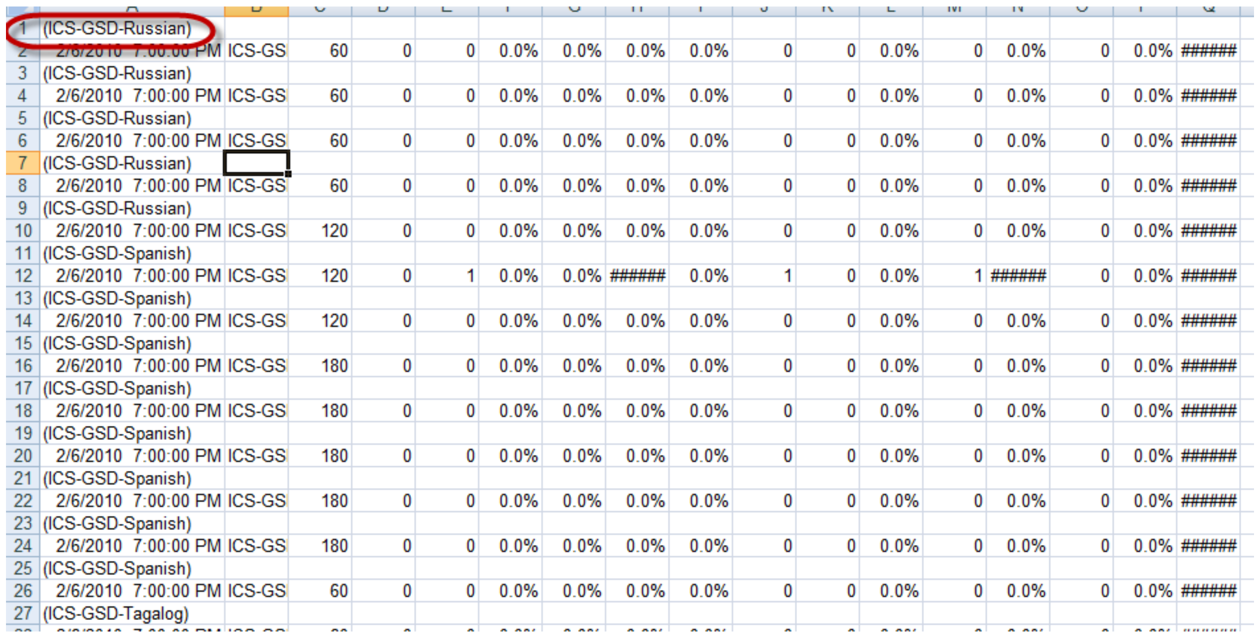
Implementation

Ideally, I would have liked to implement extraction of the data, but I wasn't given access, and had to work with old data. That being said, the data that I had to work with was not pretty.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
65505	2/6/2010 7:00:00 PM	ICS-GS	60	0	0	0.0%	0.0%	0.0%	0.0%	0	0	0.0%	0	0.0%	0	0.0%	######
65506	(ICS-GSD-HW_PR_OS)																
65507	2/6/2010 7:00:00 PM	ICS-GS	60	0	0	0.0%	0.0%	0.0%	0.0%	0	0	0.0%	0	0.0%	0	0.0%	######
65508	(ICS-GSD-HW_PR_OS)																
65509	2/6/2010 7:00:00 PM	ICS-GS	60	0	0	0.0%	0.0%	0.0%	0.0%	0	0	0.0%	0	0.0%	0	0.0%	######
65510	(ICS-GSD-HW_PR_OS)																
65511	2/6/2010 7:00:00 PM	ICS-GS	60	0	0	0.0%	0.0%	0.0%	0.0%	0	0	0.0%	0	0.0%	0	0.0%	######
65512	(ICS-GSD-HW_PR_OS)																
65513	2/6/2010 7:00:00 PM	ICS-GS	60	0	0	0.0%	0.0%	0.0%	0.0%	0	0	0.0%	0	0.0%	0	0.0%	######
65514	(ICS-GSD-HW_PR_OS)																
65515	2/6/2010 7:00:00 PM	ICS-GS	60	0	0	0.0%	0.0%	0.0%	0.0%	0	0	0.0%	0	0.0%	0	0.0%	######
65516	(ICS-GSD-Dept_Apps)																
65517	2/6/2010 7:00:00 PM	ICS-GS	120	0	0	0.0%	0.0%	0.0%	0.0%	0	0	0.0%	0	0.0%	0	0.0%	######
65518	(ICS-GSD-Portuguese)																
65519	2/6/2010 7:00:00 PM	ICS-GS	120	0	0	0.0%	0.0%	0.0%	0.0%	0	0	0.0%	0	0.0%	0	0.0%	######
65520	(ICS-GSD-Portuguese)																
65521	2/6/2010 7:00:00 PM	ICS-GS	120	0	0	0.0%	0.0%	0.0%	0.0%	0	0	0.0%	0	0.0%	0	0.0%	######
65522	(ICS-GSD-Portuguese)																
65523	2/6/2010 7:00:00 PM	ICS-GS	180	0	0	0.0%	0.0%	0.0%	0.0%	0	0	0.0%	0	0.0%	0	0.0%	######
65524	(ICS-GSD-Portuguese)																
65525	2/6/2010 7:00:00 PM	ICS-GS	180	0	0	0.0%	0.0%	0.0%	0.0%	0	0	0.0%	0	0.0%	0	0.0%	######
65526	(ICS-GSD-Portuguese)																
65527	2/6/2010 7:00:00 PM	ICS-GS	180	0	0	0.0%	0.0%	0.0%	0.0%	0	0	0.0%	0	0.0%	0	0.0%	######
65528	(ICS-GSD-Portuguese)																
65529	2/6/2010 7:00:00 PM	ICS-GS	180	0	0	0.0%	0.0%	0.0%	0.0%	0	0	0.0%	0	0.0%	0	0.0%	######
65530	(ICS-GSD-Portuguese)																
65531	2/6/2010 7:00:00 PM	ICS-GS	180	0	0	0.0%	0.0%	0.0%	0.0%	0	0	0.0%	0	0.0%	0	0.0%	######
65532	(ICS-GSD-Portuguese)																
65533	2/6/2010 7:00:00 PM	ICS-GS	60	0	0	0.0%	0.0%	0.0%	0.0%	0	0	0.0%	0	0.0%	0	0.0%	######

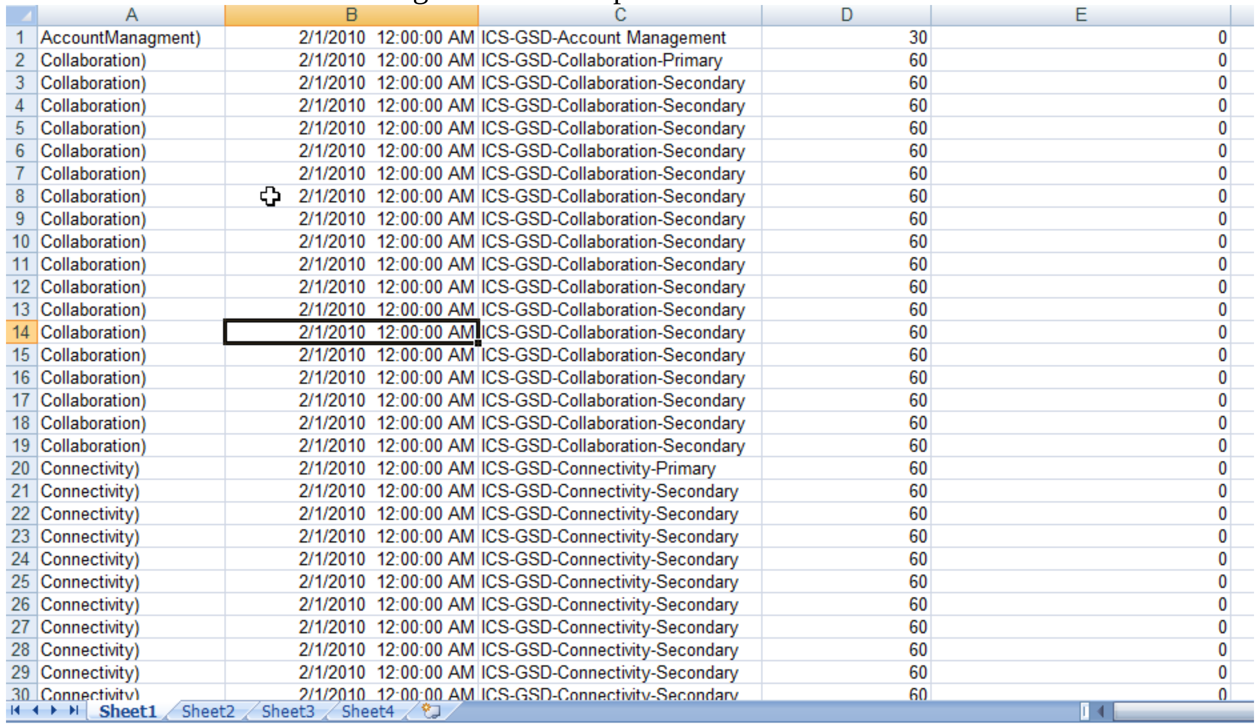
Here's an example of part of the raw data. As is highlighted, this sheet goes to row 65,533, but it is also shown that every other row only has the team name. On the main row of data, there are team names, but it is at too aggregate of a level. The identification on the row with the data is more specific than just team, and for my purposes, it was useless. In order to sort the data by team, these rows needed to be combined. One area that proved to be a problem here is shown in the screen above, as well as below.



1	(ICS-GSD-Russian)																
2	2/6/2010 7:00:00 PM	ICS-GS	60	0	0	0.0%	0.0%	0.0%	0.0%	0	0	0.0%	0	0.0%	0	0.0%	######
3	(ICS-GSD-Russian)																
4	2/6/2010 7:00:00 PM	ICS-GS	60	0	0	0.0%	0.0%	0.0%	0.0%	0	0	0.0%	0	0.0%	0	0.0%	######
5	(ICS-GSD-Russian)																
6	2/6/2010 7:00:00 PM	ICS-GS	60	0	0	0.0%	0.0%	0.0%	0.0%	0	0	0.0%	0	0.0%	0	0.0%	######
7	(ICS-GSD-Russian)																
8	2/6/2010 7:00:00 PM	ICS-GS	60	0	0	0.0%	0.0%	0.0%	0.0%	0	0	0.0%	0	0.0%	0	0.0%	######
9	(ICS-GSD-Russian)																
10	2/6/2010 7:00:00 PM	ICS-GS	120	0	0	0.0%	0.0%	0.0%	0.0%	0	0	0.0%	0	0.0%	0	0.0%	######
11	(ICS-GSD-Spanish)																
12	2/6/2010 7:00:00 PM	ICS-GS	120	0	1	0.0%	0.0%	######	0.0%	1	0	0.0%	1	######	0	0.0%	######
13	(ICS-GSD-Spanish)																
14	2/6/2010 7:00:00 PM	ICS-GS	120	0	0	0.0%	0.0%	0.0%	0.0%	0	0	0.0%	0	0.0%	0	0.0%	######
15	(ICS-GSD-Spanish)																
16	2/6/2010 7:00:00 PM	ICS-GS	180	0	0	0.0%	0.0%	0.0%	0.0%	0	0	0.0%	0	0.0%	0	0.0%	######
17	(ICS-GSD-Spanish)																
18	2/6/2010 7:00:00 PM	ICS-GS	180	0	0	0.0%	0.0%	0.0%	0.0%	0	0	0.0%	0	0.0%	0	0.0%	######
19	(ICS-GSD-Spanish)																
20	2/6/2010 7:00:00 PM	ICS-GS	180	0	0	0.0%	0.0%	0.0%	0.0%	0	0	0.0%	0	0.0%	0	0.0%	######
21	(ICS-GSD-Spanish)																
22	2/6/2010 7:00:00 PM	ICS-GS	180	0	0	0.0%	0.0%	0.0%	0.0%	0	0	0.0%	0	0.0%	0	0.0%	######
23	(ICS-GSD-Spanish)																
24	2/6/2010 7:00:00 PM	ICS-GS	180	0	0	0.0%	0.0%	0.0%	0.0%	0	0	0.0%	0	0.0%	0	0.0%	######
25	(ICS-GSD-Spanish)																
26	2/6/2010 7:00:00 PM	ICS-GS	60	0	0	0.0%	0.0%	0.0%	0.0%	0	0	0.0%	0	0.0%	0	0.0%	######
27	(ICS-GSD-Tagalog)																

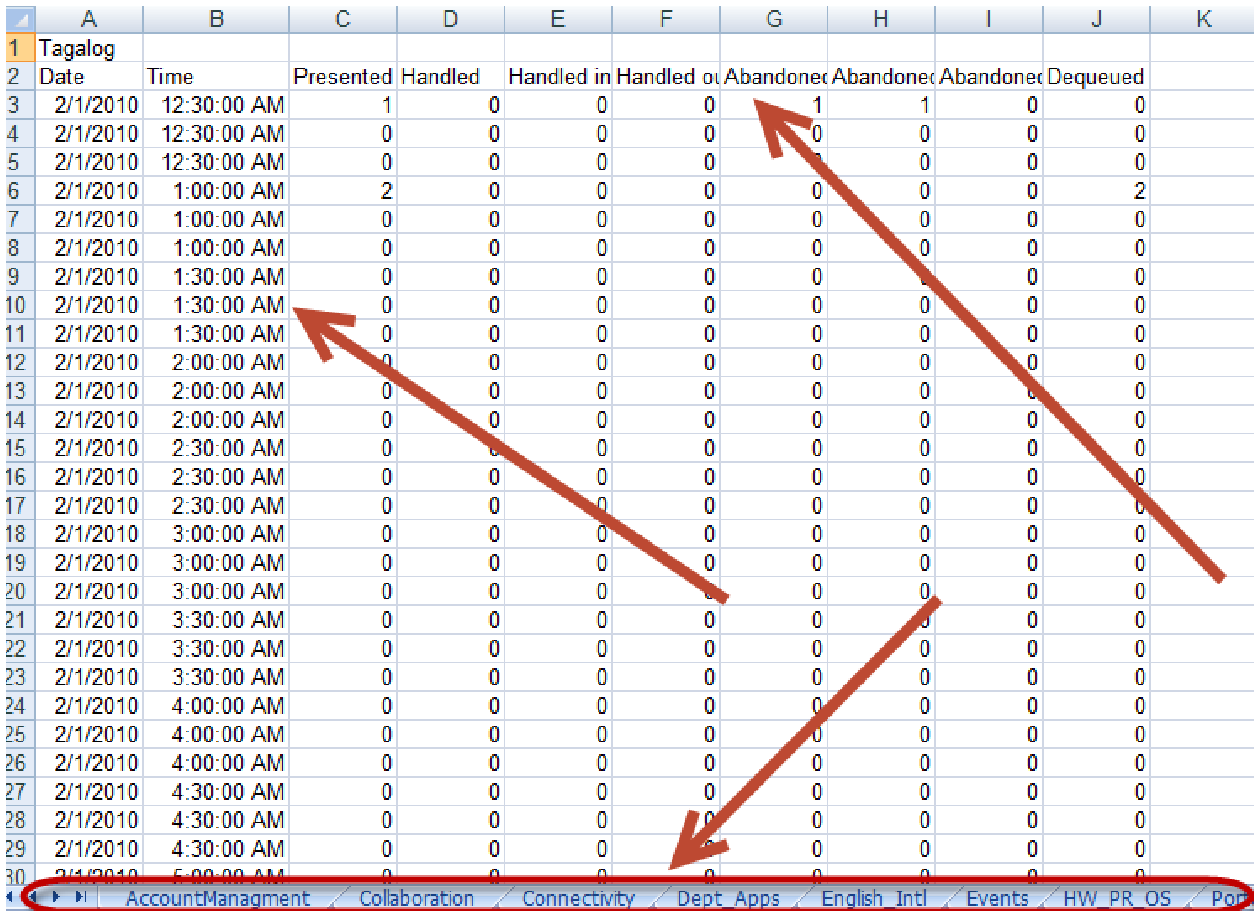
On some of the sheets, depending on the starting point, the team identifier may fall on the next sheet. The program searches for this, and if the last row of data on a given sheet contains the useful information, the team name is extracted from the following sheet.

Once the team is moved up from the row below, that entire row is deleted, because that row no longer contains any useful information. This makes the data more manageable for the remaining steps. Also, throughout the raw data are summary rows, which make a summary of information across all teams across a certain time period, but doesn't provide any useful information as to the adequateness of staffing, which is what my program intends to examine. Because of the large number of rows that need to be examined and data that needs to be moved, this initial portion of the program can take quite a while to run. When I had initially made the program, I left a loophole, which allowed the project to eat a lot of the work that I had done if I were to press "Process Data" after the data had already been processed. The fix was pretty simple, and can still be exploited, but the user would need to add an extra sheet and name it "Sheet1". Without "Sheet1" the user will receive an error message, and the procedure will exit, preventing, on a basic level, the data from being erased. As you can see below, the resulting information is much more manageable for manipulation.



	A	B	C	D	E
1	AccountManagment)	2/1/2010 12:00:00 AM	ICS-GSD-Account Management	30	0
2	Collaboration)	2/1/2010 12:00:00 AM	ICS-GSD-Collaboration-Primary	60	0
3	Collaboration)	2/1/2010 12:00:00 AM	ICS-GSD-Collaboration-Secondary	60	0
4	Collaboration)	2/1/2010 12:00:00 AM	ICS-GSD-Collaboration-Secondary	60	0
5	Collaboration)	2/1/2010 12:00:00 AM	ICS-GSD-Collaboration-Secondary	60	0
6	Collaboration)	2/1/2010 12:00:00 AM	ICS-GSD-Collaboration-Secondary	60	0
7	Collaboration)	2/1/2010 12:00:00 AM	ICS-GSD-Collaboration-Secondary	60	0
8	Collaboration)	2/1/2010 12:00:00 AM	ICS-GSD-Collaboration-Secondary	60	0
9	Collaboration)	2/1/2010 12:00:00 AM	ICS-GSD-Collaboration-Secondary	60	0
10	Collaboration)	2/1/2010 12:00:00 AM	ICS-GSD-Collaboration-Secondary	60	0
11	Collaboration)	2/1/2010 12:00:00 AM	ICS-GSD-Collaboration-Secondary	60	0
12	Collaboration)	2/1/2010 12:00:00 AM	ICS-GSD-Collaboration-Secondary	60	0
13	Collaboration)	2/1/2010 12:00:00 AM	ICS-GSD-Collaboration-Secondary	60	0
14	Collaboration)	2/1/2010 12:00:00 AM	ICS-GSD-Collaboration-Secondary	60	0
15	Collaboration)	2/1/2010 12:00:00 AM	ICS-GSD-Collaboration-Secondary	60	0
16	Collaboration)	2/1/2010 12:00:00 AM	ICS-GSD-Collaboration-Secondary	60	0
17	Collaboration)	2/1/2010 12:00:00 AM	ICS-GSD-Collaboration-Secondary	60	0
18	Collaboration)	2/1/2010 12:00:00 AM	ICS-GSD-Collaboration-Secondary	60	0
19	Collaboration)	2/1/2010 12:00:00 AM	ICS-GSD-Collaboration-Secondary	60	0
20	Connectivity)	2/1/2010 12:00:00 AM	ICS-GSD-Connectivity-Primary	60	0
21	Connectivity)	2/1/2010 12:00:00 AM	ICS-GSD-Connectivity-Secondary	60	0
22	Connectivity)	2/1/2010 12:00:00 AM	ICS-GSD-Connectivity-Secondary	60	0
23	Connectivity)	2/1/2010 12:00:00 AM	ICS-GSD-Connectivity-Secondary	60	0
24	Connectivity)	2/1/2010 12:00:00 AM	ICS-GSD-Connectivity-Secondary	60	0
25	Connectivity)	2/1/2010 12:00:00 AM	ICS-GSD-Connectivity-Secondary	60	0
26	Connectivity)	2/1/2010 12:00:00 AM	ICS-GSD-Connectivity-Secondary	60	0
27	Connectivity)	2/1/2010 12:00:00 AM	ICS-GSD-Connectivity-Secondary	60	0
28	Connectivity)	2/1/2010 12:00:00 AM	ICS-GSD-Connectivity-Secondary	60	0
29	Connectivity)	2/1/2010 12:00:00 AM	ICS-GSD-Connectivity-Secondary	60	0
30	Connectivity)	2/1/2010 12:00:00 AM	ICS-GSD-Connectivity-Secondary	60	0

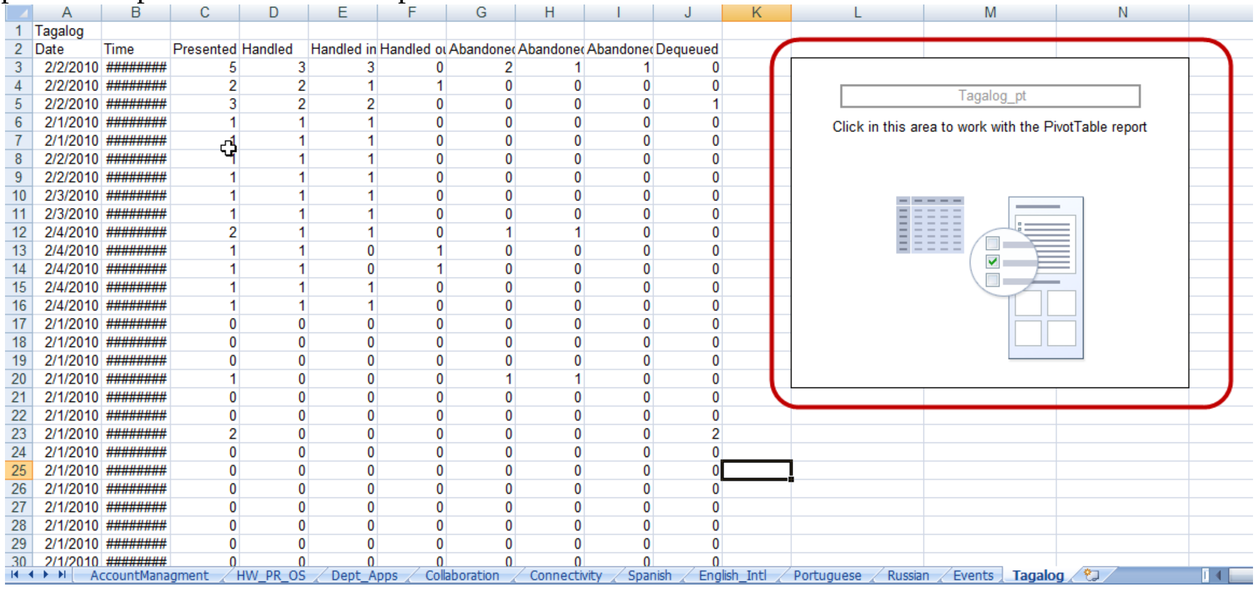
To remove some of the redundant data from the report, a number of columns are deleted, and a new procedure is called. The sortData procedure also takes a considerable amount of time to run. This procedure does just what it says it does; it sorts all of the data from the report, and sorts it into worksheets. This procedure goes through each of the existing worksheets and reads through the data. It looks to column a, the team name, and then compares this data to each of the existing worksheets. If there is an existing worksheet with the given team name, the data is simply added to the next empty row of data on that worksheet. There is nothing special about moving this data, just a loop that takes the values from certain columns, and replaces those in the new worksheet. If the team name is not found as the name of a worksheet, a new worksheet is added with that team name. New worksheets are given proper headings and titles, to allow for a uniform appearance across the entire workbook. Each team will only have one worksheet, which allows us to get the most benefit out of this data. The sortData procedure uses a number of variables, and passes those variables to different procedures to allow for uniformity. After the final row of data is processed, the sheet is deleted, and the procedure moves on to the next worksheet until all of the data is exhausted. The sortData procedure also splits some data from the time and date cell. In the raw data, the time and date are combined in one cell, but the needs of this report require the time and the date to be separated. The most important information to finding when there are shortages of employees is the time of day that the calls come in, so I wanted to be able to breakdown the data based on the time, and not just the date.



	A	B	C	D	E	F	G	H	I	J	K
1	Tagalog										
2	Date	Time	Presented	Handled	Handled in	Handled ou	Abandoned	Abandoned	Abandoned	Dequeued	
3	2/1/2010	12:30:00 AM	1	0	0	0	1	1	0	0	
4	2/1/2010	12:30:00 AM	0	0	0	0	0	0	0	0	
5	2/1/2010	12:30:00 AM	0	0	0	0	0	0	0	0	
6	2/1/2010	1:00:00 AM	2	0	0	0	0	0	0	2	
7	2/1/2010	1:00:00 AM	0	0	0	0	0	0	0	0	
8	2/1/2010	1:00:00 AM	0	0	0	0	0	0	0	0	
9	2/1/2010	1:30:00 AM	0	0	0	0	0	0	0	0	
10	2/1/2010	1:30:00 AM	0	0	0	0	0	0	0	0	
11	2/1/2010	1:30:00 AM	0	0	0	0	0	0	0	0	
12	2/1/2010	2:00:00 AM	0	0	0	0	0	0	0	0	
13	2/1/2010	2:00:00 AM	0	0	0	0	0	0	0	0	
14	2/1/2010	2:00:00 AM	0	0	0	0	0	0	0	0	
15	2/1/2010	2:30:00 AM	0	0	0	0	0	0	0	0	
16	2/1/2010	2:30:00 AM	0	0	0	0	0	0	0	0	
17	2/1/2010	2:30:00 AM	0	0	0	0	0	0	0	0	
18	2/1/2010	3:00:00 AM	0	0	0	0	0	0	0	0	
19	2/1/2010	3:00:00 AM	0	0	0	0	0	0	0	0	
20	2/1/2010	3:00:00 AM	0	0	0	0	0	0	0	0	
21	2/1/2010	3:30:00 AM	0	0	0	0	0	0	0	0	
22	2/1/2010	3:30:00 AM	0	0	0	0	0	0	0	0	
23	2/1/2010	3:30:00 AM	0	0	0	0	0	0	0	0	
24	2/1/2010	4:00:00 AM	0	0	0	0	0	0	0	0	
25	2/1/2010	4:00:00 AM	0	0	0	0	0	0	0	0	
26	2/1/2010	4:00:00 AM	0	0	0	0	0	0	0	0	
27	2/1/2010	4:30:00 AM	0	0	0	0	0	0	0	0	
28	2/1/2010	4:30:00 AM	0	0	0	0	0	0	0	0	
29	2/1/2010	4:30:00 AM	0	0	0	0	0	0	0	0	
30	2/1/2010	5:00:00 AM	0	0	0	0	0	0	0	0	

The screen shot above shows the results of the sortData procedure. The data isn't in the best formatting possible, but that's because the end product isn't this data, but the corresponding pivot table and chart are the end product. That's why you can see that the columns aren't autofitted, and especially why the data doesn't appear to be usable. You can also see that the time and date fields are now split, which will make further use of the data more feasible. Cell "A1" shows the team name, which corresponds to the tab name. Also visible in the screenshot are the tabs that were created for each of the teams. sortData also takes a considerable amount of time to run, because it has to go through each of the rows and then identify where to place the data that it retrieves from that row.

The final procedure of the project is the createPivTable procedure, which creates a pivot table and chart from the corresponding table. This is something that I didn't have a lot of experience with, but after a little recording of macros, I learned that it can be a very powerful procedure. This procedure runs mainly off of a for each loop, running through each of the sheets in the workbook. It starts in the reorganized data, and finds the last row and column of the data to use for the range that the pivot table will refer to. I also code in a string variable on each sheet to name the pivot table, simply the sheet name & "_pt". A uniform naming convention will make it easy for further manipulation of the table. This procedure is called from the organizeData procedure, but I had to make it more robust, because it can also be called when the change parameters button is clicked. Because of this, I had to put an On Error statement into the code. This is placed before the pivot table is formed, and on error it directs the procedure to an exception statement at the end of the code. The error statement has one main task, to identify when a pivot table already exists in the worksheet. Because this procedure can be called a number of times if the user wants to change the parameters, it's important that the code doesn't break if a pivot table already exists. The exception statement deletes the rows that the pivot table is placed in the worksheet. Assuming that there is no pivot table in the current worksheet, the procedure proceeds to create a pivot table.

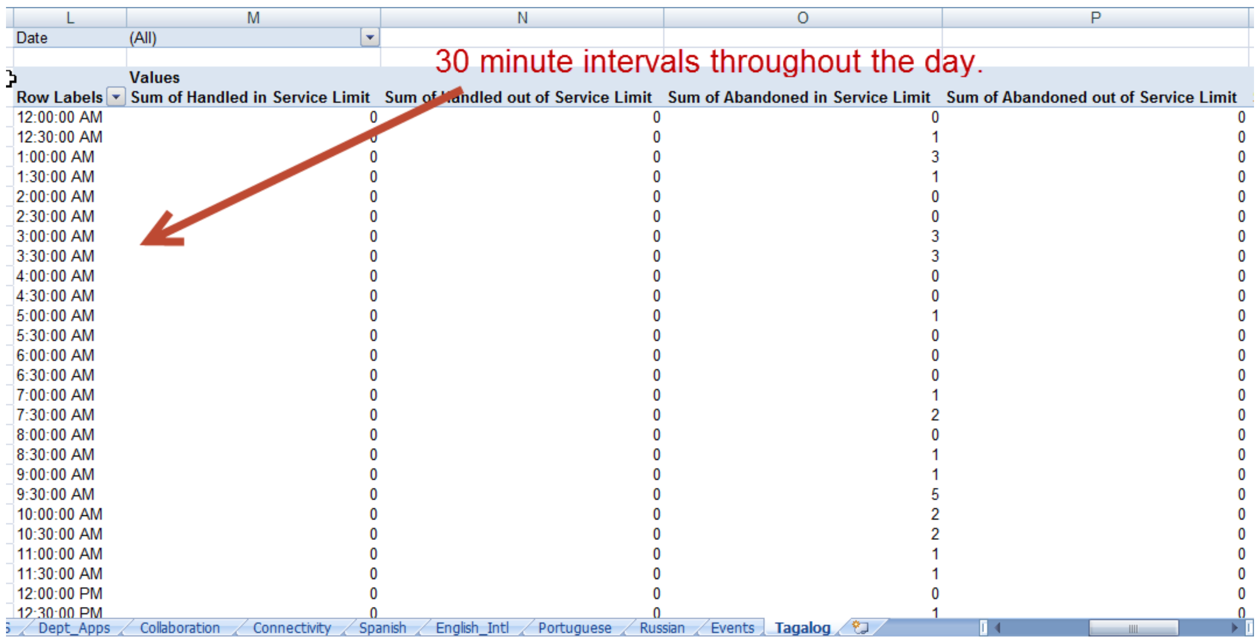


	A	B	C	D	E	F	G	H	I	J
1	Tagalog									
2	Date	Time	Presented	Handled	Handled in	Handled ou	Abandonec	Abandonec	Abandonec	Dequeued
3	2/2/2010	########	5	3	3	0	2	1	1	0
4	2/2/2010	########	2	2	1	1	0	0	0	0
5	2/2/2010	########	3	2	2	0	0	0	0	1
6	2/1/2010	########	1	1	1	0	0	0	0	0
7	2/1/2010	########	1	1	1	0	0	0	0	0
8	2/2/2010	########	1	1	1	0	0	0	0	0
9	2/2/2010	########	1	1	1	0	0	0	0	0
10	2/3/2010	########	1	1	1	0	0	0	0	0
11	2/3/2010	########	1	1	1	0	0	0	0	0
12	2/4/2010	########	2	1	1	0	1	1	0	0
13	2/4/2010	########	1	1	0	1	0	0	0	0
14	2/4/2010	########	1	1	0	1	0	0	0	0
15	2/4/2010	########	1	1	1	0	0	0	0	0
16	2/4/2010	########	1	1	1	0	0	0	0	0
17	2/1/2010	########	0	0	0	0	0	0	0	0
18	2/1/2010	########	0	0	0	0	0	0	0	0
19	2/1/2010	########	0	0	0	0	0	0	0	0
20	2/1/2010	########	1	0	0	0	1	1	0	0
21	2/1/2010	########	0	0	0	0	0	0	0	0
22	2/1/2010	########	0	0	0	0	0	0	0	0
23	2/1/2010	########	2	0	0	0	0	0	0	2
24	2/1/2010	########	0	0	0	0	0	0	0	0
25	2/1/2010	########	0	0	0	0	0	0	0	0
26	2/1/2010	########	0	0	0	0	0	0	0	0
27	2/1/2010	########	0	0	0	0	0	0	0	0
28	2/1/2010	########	0	0	0	0	0	0	0	0
29	2/1/2010	########	0	0	0	0	0	0	0	0
30	2/1/2010	########	0	0	0	0	0	0	0	0

Tagalog_pt

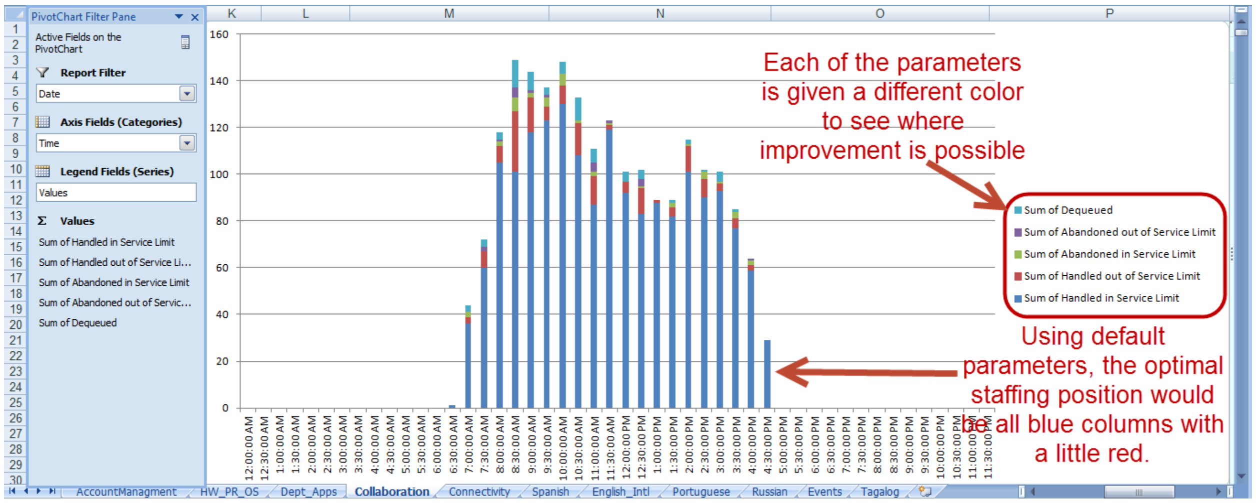
Click in this area to work with the PivotTable report

Creating the pivot table uses the last row and column data to set the range of data, sets the name equal to the variable that was declared, and places the pivot table inside of the current worksheet. I was contemplating creating the pivot table in a new page, however, after taking a look at the number of tabs that already existed, I felt that it would be much more user-friendly to place the table inside of the existing worksheet. This lowers the overall redundancy by having multiple tabs with similar names. Because time is the most important parameter for determining the need for more or less employees, this parameter is selected for all of the pivot tables, regardless of the other parameters that are chosen as a row field. Also, the date parameter is always added as a page field, if the user wishes to examine the data by certain days. The remaining fields are determined by the selections that the user makes when they run the procedures. I'll discuss the fields later when I talk briefly about the user form that allows for selection of the criteria. Using if statements, the procedure checks which parameters were selected and puts them into the pivot table.



Date	(All)			
	Values			
Row Labels	Sum of Handled in Service Limit	Sum of Handled out of Service Limit	Sum of Abandoned in Service Limit	Sum of Abandoned out of Service Limit
12:00:00 AM	0	0	0	0
12:30:00 AM	0	0	1	0
1:00:00 AM	0	0	3	0
1:30:00 AM	0	0	1	0
2:00:00 AM	0	0	0	0
2:30:00 AM	0	0	0	0
3:00:00 AM	0	0	3	0
3:30:00 AM	0	0	3	0
4:00:00 AM	0	0	0	0
4:30:00 AM	0	0	0	0
5:00:00 AM	0	0	1	0
5:30:00 AM	0	0	0	0
6:00:00 AM	0	0	0	0
6:30:00 AM	0	0	0	0
7:00:00 AM	0	0	1	0
7:30:00 AM	0	0	2	0
8:00:00 AM	0	0	0	0
8:30:00 AM	0	0	1	0
9:00:00 AM	0	0	1	0
9:30:00 AM	0	0	5	0
10:00:00 AM	0	0	2	0
10:30:00 AM	0	0	2	0
11:00:00 AM	0	0	1	0
11:30:00 AM	0	0	1	0
12:00:00 PM	0	0	0	0
12:30:00 PM	0	0	1	0

The data run from the report has data set in 30 minute intervals, which you can see in the above finished pivot table. This makes the analysis of the data very easy, because the 30 minute intervals gives the best breakdown of calls received at given times during the day, which will lead to better information of when to staff employees. Because there are 48 30-minute intervals in a day, the pivot table still contains a lot of rows, so by itself is not very useful. For that reason, after the pivot table is created, this procedure then creates a chart based on the information in the pivot table. This part of the procedure also has an on error statement, but simply resumes next if there is an error. The reason that it has this is because the first line in this portion deletes the chart objects on the worksheet. If no chart objects exist, it produces an error, but this should be expected when the data is first being processed, so instead of having an error if a chart does not exist, the code resumes and adds the chart. I've set variables to find the last row and column of the pivot table, which are then used to make a range for the source data for the chart. The column stacked chart seemed best able to present the data for this project, so that is chosen as the default chart. For the default parameters, the chart has been set up to cover the entire screen (at least my laptop's screen), and the window should scroll to fixate on the chart, with the sheet selected so that the field options are visible. This runs on a roe each loop to go through all of the sheets and add the chart for the pivot table in each worksheet.


PivotChart Filter Pane

Active Fields on the PivotChart

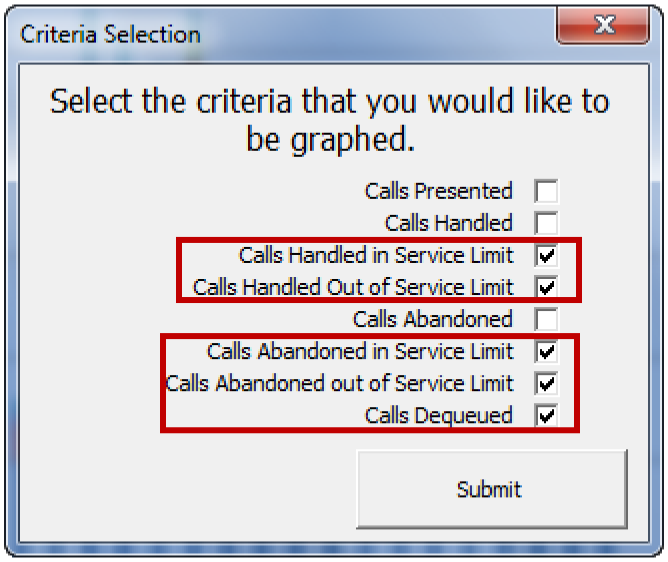
Report Filter: Date

Axis Fields (Categories): Time

Legend Fields (Series): Values

Σ Values
- Sum of Handled in Service Limit
- Sum of Handled out of Service Li...
- Sum of Abandoned in Service Limit
- Sum of Abandoned out of Servic...
- Sum of Dequeued

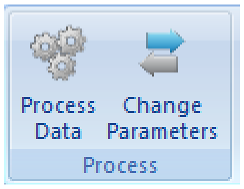
The chart above shows what a typical graph should look like. Some of the lines are closed at certain times of the day, that's why they don't have any calls that come through, but that is to be expected. The chart above uses the default parameters, which can be seen in the screen shot of the user form below.


Criteria Selection

Select the criteria that you would like to be graphed.

- Calls Presented ☐
- Calls Handled ☐
- Calls Handled in Service Limit ☑
- Calls Handled Out of Service Limit ☑
- Calls Abandoned ☐
- Calls Abandoned in Service Limit ☑
- Calls Abandoned out of Service Limit ☑
- Calls Dequeued ☑

These parameters do allow for some redundancy, but it is assumed that the user will know what level of aggregation that they want to look at. Calls presented will show all of the calls received by each team for the given times. This is an aggregate of calls handled, abandoned and dequeued. Calls handled are the calls that are taken by agents in the call center. The report that is run has a set service limit, and indicates which calls are put on hold in excess of the service limit and under the service limit. This is information that managers might find useful, because even though the calls are being handled, if they aren't meeting their goals for responding to calls, it could mean that they are too short staffed. Calls abandoned are calls that were on hold, but the people calling hung up before they got a chance to talk to an agent at the service desk. This also has a breakdown of in the service limit or out of the service limit. This is helpful, because looking simply at abandoned calls doesn't tell the whole story. Someone might call, be placed on hold, and then quickly resolve their own issue. This will count as an abandoned call, but, abandoned calls under the service limit are less severe than calls abandoned over the service limit. Any abandoned calls indicate that there might be a need for further staffing, but when there is a large number of calls abandoned out of the service limit, this should be a red flag if it is continually occurring at the same time periods. The final parameter is calls dequeued. These are calls made to the service center during very busy times. The phone lines can only handle a certain number of calls on hold at a certain time, so any calls that exceed this amount are dequeued. This is something that should be examined in detail. If a large number of calls are being dequeued at the same time each day, this is a definite red flag that more employees need to be staffed at this time. However, large anomalies might also indicate that there was some sort of rush of calls in a short amount of time. This can be the result of a power outage, or a server going down. Regardless, it's important to examine dequeued calls, because an error in this interpretation can result in overstaffing.



Finally, in order to make it more aesthetically pleasing and user friendly, I altered the user interface to have a new tab and two new buttons to run the procedures.

Discussion of learning and Conceptual Difficulties encountered.

There were a number of things that were difficult in creating this project. Probably the biggest difficulty that I encountered was the amount of time that it took to run the procedures. I would work on the code for a few hours, and then begin running the code by debugging and stepping through initial lines to make sure that I thought it was doing what I wanted it to do. However, with as much data as I had, sometime while running, it could run into an issue, which wouldn't necessarily cause an error, it just wouldn't do what I wanted it to. When doing my sortData procedure, I ran it 5-6 times, and each time after I ran it, I found something else that I had made a small mistake on. It didn't affect all of the data, and still ran to completion, but it might have taken a couple wrong cell values due to some irregularities in the data.

Another thing that caused me a lot of problems, and took a lot of time searching on Google to figure it out was using vba split. I spoke briefly about separating the time and date in the raw data in my product description, but it caused me a lot of issues. I was originally trying to split based on the space in between date and time, which I thought worked well. In my last difficulty, I spoke of the time that it took to run being hard, and this mistake was one of the reasons. Because 12:00 AM is considered 0 in Excel, when I ran the procedure with vba split, it only returned the date, because it considered the cell to only contain the date, even though the cell said the date and the time like every other cell. When VBA split populated my array, it put the date into all three slots. I didn't notice this until I had gotten further in the project, but every time 12:00 AM was in a cell in the data, my sorted data had the date instead of the time. This proved to be a problem with the pivot table, and also just a problem that I wanted to fix. I tried a number of things, including the left and right functions, but my problem was that the data was not what it appeared to be. Even though it appeared in the cells, in VBA, the 12:00 AM wasn't a string of 12:00 AM. I'm sure there are other ways around it, but I took a pretty simple way after all of my trials. After the array was dimmed, I hard coded in array(0), array(1), and array(2) in later statements where I wanted it print out. I would get an error when this ran, before changing a few things, so I set up an on error statement that takes the procedure to an exception statement. Here, it redims the array for (0to2) and hard codes a string for 12:00 AM and returns to where the error occurs. This probably isn't the best fix, and if I examined more fully the value and the formatting of the time and date, I might have found a more efficient and correct way to fix it, however, for my purposes, this worked.

Another area that I would have liked to implement, but was restricted was integrating the extraction of the data with the workbook. I would have liked to make a userform that would have accepted all of the inputs needed to enter the database and pull the report. It's not that I didn't have the ability to do this, just when I asked for permission, it wasn't granted because they wanted to limit access to the database.

The final area that I had a lot of problems with was setting up the ribbon with the changes in the user interface. I know we went over this in class, but we went over it rather briefly, so there were a lot of things that I was trying to figure out on my own. I had a number of difficulties. The first was with setting up the basic code in the UI editor. I kept messing up the MSO statements, which was really frustrating, but after carefully reading through the website that we used in class, I was able to resolve my syntax errors. My next big issue with this is actually kind of dumb. I got the buttons to show up in the ribbon where I wanted them to, but when I ran the code, it would freeze like it was entering an infinite loop. After messing around with it for a while, I came to the realization that I had forgotten to change the passed variable in the procedures that I was calling to contain the control as IRibbonControl. I know that's not that huge of an issue, but it caused me a lot of problems.

One other big problem that I had was trying to find a real problem to solve. I'm not a very creative person by any means. The job I work at now has very limited use of excel, other than using preformatted files for journal uploads. Because of this, I spent a lot of time trying to come up with something that would have a big enough scope to qualify for this project. I tried a lot of things, posting on KSL and Craig's list, asking old employers, asking friends and a few professors, looked through most of the projects that students had submitted in the past, but didn't really come across anything that seemed like it would fit. This project kind of came to me as a fluke, but I was glad that it did.

I really learned a lot from this project, specifically, that I can actually do a lot with VBA now. There were a lot of things that as I was starting this project I wasn't really sure how I was going to do it, but when I just sat down and started typing, I was able to figure out how to get things done. I did an internship this last winter, and I was able to do a little bit of VBA for some of the engagements that I was on, and people thought it was amazing. Using what I know now, I'm sure this is going to have a huge impact on the impression that I make on people. I learned that even though I don't really know how to do everything, I can look up things online and tailor those things to meet my own needs. Just with this project, I learned about using VBA split, which can be very valuable when used correctly, I learned how to manipulate the window position, and change the chart size and position. Prior to the project, I didn't really have any experience with the On Error statement, but now I feel pretty confident with using it, and being pretty effective with it.

Assistance

I didn’t receive a lot of help with this project. I did google a lot of things, mainly to learn how to change the window position, and how to use VBA split. I also received assistance from Austin Mcneil when I was trying to get my ribbon buttons to work. He had successfully integrated the buttons into his project, and helped me get the UI editor code correct. That’s the only help that I received on this project.