Executive Summary

This VBA project is inspired in the current events project posted on the VBA projects blog. The blog can be accessed on http://vbaprojects.blogspot.com/ and the original description for this project, as well as the original solution, are available under the title "Current Events!" in the 2015 subdivision.

We all know that in business it is essential to be aware of the most recent facts and events around the world. These events can change companies' strategies, deals, and impact the decision making in all enterprises. Business leaders have to find a way to learn about those events so that they can use the information in their daily jobs.

At the same time, all the demands of the modern life creates a challenge to stay current with the events around us. There are some many responsibilities in people's lives that even taking a short pause to read the news is something really hard to remember and to do.

The proposed VBA project will help people to get the current news from two of the most important sources in the market (The Wall Street Journal and The Economist) by automatizing the access and sending the headline of the publications to the user via email. The excel spreadsheet will also set up an automatic time to process the solution every day, so that people can get the daily news in their inbox with no additional effort.

Since The Wall Street Journal and The Economist are restricted to subscribers, this project will use BYU library privileges to download the news and the user will have to provide a valid BYU ID and password to complete the access.

By using this VBA solution the user will be able to keep current with the main events happening around the world.

Implementation Documentation

The solution was developed entirely in one module, it also utilizes two Userforms for interaction with the user and one class to implement the Internet Explorer control interface, the object that implements this class was declared as a global variable to be accessible to all procedures in the module. Ribbon customization was applied to make the user experience more intuitive, there's a tab called "News" and inside that the user will find a button "Get news!" that he/she will use to start the procedures. Links were created inside the spreadsheet to allow the use to access and read the news, it required a change in the *Worksheet_FollowHyperlink* native method so that the code makes sure the user is loged in Learning Suite before opening the news, this was one big difference to the original proposal and this change was made because I couldn't guarantee that the user was corrected loged in the BYU channels using a simple link in an email message

There are two main parts happening during the process:

- Accessing the news and sending it to the user
- Enabling the user to read the news

For the first part (Accessing the news and sending it to the user) there's a main sub procedure that orquestrates the steps of the VBA code, this procedure is called getNews, then the four major steps inside that are: getting the news from WSJ, getting the news from The Economist, ending the mail to the user, and scheduling the automatic macro execution.

For the second part (Enabling the user to read the news) there's one main procedure that supports the access and viewing of the news, this procedure is called "openPage". We will go over the details for those parts below.

Accessing the news and sending it to the user

This part starts when the user clicks on the "Get News!" button in the "News" tab. Using a callback method (*MacroNews*) the button activates the user form called "InputForm", the user enters his data for Learning Suite Login, email login and destinatary of email. When the user click "OK" *getNews* sub procedure called. This is the orquestrator that will command the steps to retrieve the news from WSJ and The Economist. If the user clicks "Cancel" the form is unloaded.

The *getNews* sub procedure only calls other sub procedures inside the module and one method from the excel spreadsheet to set the right position on the screen. This is a printscreen from the *getNews* method:

```
Sub getNews()

getWSJ
getEconomist

a.terminateIE

sendEmail

ThisWorkbook.Sheets("Login").Cells(1, 1).Activate
schedule

End Sub
```

Figure 1 - getNews print

As shown above, the first sub procedure called is *getWSJ*, as the name suggests this procedure is responsible to retrieve the data from WSJ. *getWSJ* has 5 variables and 1 constant to store values used in the logic. The sub procedure makes IE visible and then navigates to the portal that make WSJ accessible to BYU's library users (http://dbs.lib.byu.edu/factiva), with the page loaded the code checks if the user is logged into the BYU system, by checking the document location, if IE is in the login page, then the login data is retrieved from the *InputForm* and populated in the page, then the form is submitted.

With the user logged in the browser continues the navigation to the Factiva portal and then to its home page. Once in the home page the solution will retrieve the headlines from WSJ, since there are always five (5) headlines in the home page the code uses a for loop to get the information. The sub procedure ends and control goes back to *getNews.* The picture below shows the Factiva home page.

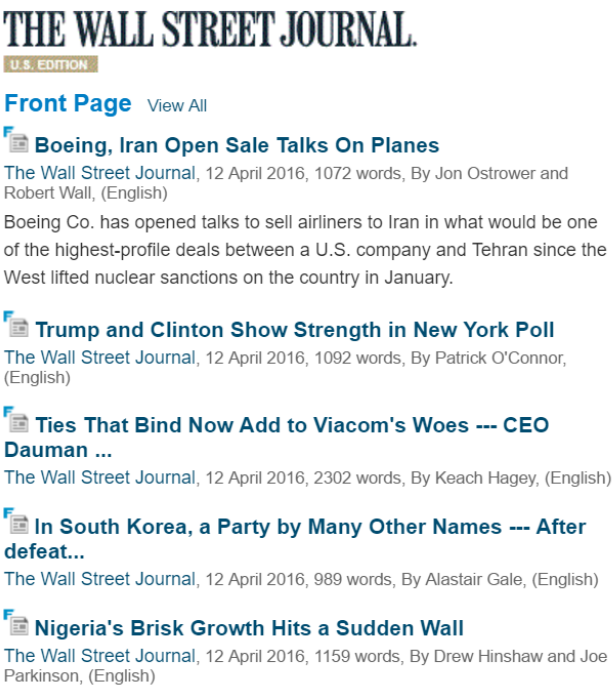

Figure 2- Factiva Home

The next sub procedure called is *getEconomist*, it is responsible to retrieve the news from The Economist. This procedure is very similar to *getWSJ*, it checks for login status, navigates to The Economist portal using a BYU script (https://www.lib.byu.edu/cgi-bin/remoteauth.pl?url=http://www.economist.com/printedition/covers) and then navigates to the last issue of the The Economist, retrieving the review for business and politics. Since we only get these two (2) links there's no loop here. The following picture shows part of The Economist page.

Figure 3 - Economist page

Both sub procedures (*getWSJ and getEconomist*) make calls to the *LinkMaker* sub procedure. This procedure is responsible to create a link making a reference to the cell that stores the address of the news. It allows us to modify the native method *Worksheet_FollowHyperlink* and use our own code to open the news.

Again, the control returns to *getNews*. The IE agent is terminated, since it won't be used anymore in this part of the process, and the *sendEmail* sub procedure is called.

To send the email the sender and destinatary data are retrieved from the *InputForm*, the procedure then saves a copy of the file, clears the news from the spreadsheet and send the email, writing the subject "News Feed", the body of the message, and attaching the file just created to the email. The sub procedure sendGmail is called to effectively send the message, this is the same procedure provided in class by Prof. Allen. Only email addresses can be used to send the message in this project. The message is sent and the file is deleted from the user's machine. The picture below shows the *sendEmail* code.

```
Sub sendEmail()
Dim email As Boolean
Dim sender As String, destinatary As String, password As String
Dim newhour As Integer, newminute As Integer, newsecond As Integer
Dim waittime As Date

sender = InputForm.txtSender
password = InputForm.txtSenderPS
destinatary = InputForm.txtTo

ThisWorkbook.SaveCopyAs ThisWorkbook.path & Application.PathSeparator & "NewsFeed.xlsm"

ThisWorkbook.Sheets("Login").Range("D1").CurrentRegion.Clear

email = sendGMail(destinatary, sender, password, _
        "News Feed", "Here are the updated news from WSJ and Economist.", _
        ThisWorkbook.path & Application.PathSeparator & "NewsFeed.xlsm")

newhour = Hour(Now())
newminute = Minute(Now())
newsecond = Second(Now()) + 3
waittime = TimeSerial(newhour, newminute, newsecond)
Application.Wait waittime

On Error Resume Next
Kill ThisWorkbook.path & Application.PathSeparator & "NewsFeed.xlsm"

End Sub
```

Figure 4 - sendEmail code

Once again control is back to *getNews* and it finally call the *schedule* sub procedure to schedule the automatic execution for this code. The *schedule* sub procedure is shown below:

```
Sub schedule()
    On Error Resume Next
    Application.OnTime TimeValue("17:00:00"), "getNews"
End Sub
```

Figure 5 - Schedule sub procedure

For both of the main sub procedures used to get the news (*getWSJ and getEconomist)* I used the resources available in the agent class to navigate among the different pages I needed *(agent.openPage method)* and locate the links and content that I needed to store and use (agent.moveTo and agent.getText methods). The string functions, like inStr, Mid, Left, and Right were also used throughout the project. The picture below shows an example from the code where the macro retrieves news from WSJ.

```
doc = a.document.Location
doc = Left(doc, InStr(1, doc, "np/") - 1) & "sb/SimpleSearch.aspx?NAPC=p"

a.openPage doc, True

a.position = 1

For x = 1 To 5

    a.moveTo (a.HTMLEncode("class=""headline"">"))
    a.moveTo "href=""../"
    Link = a.HTMLDecode(a.getText("""))

    a.moveTo "</a>"
    a.moveBackTo "> "
    title = a.HTMLDecode(a.getText("<"))

    ActiveSheet.Cells(x, 4).Value = title
    ActiveSheet.Cells(x, 5).Value = factiva & Link
    ActiveSheet.Cells(x, 5).Activate
    LinkMaker
    ActiveSheet.Cells(x, 6).Value = WSJ

Next
```

Figure 6 - Example of methods used in the project

At this point the first part of the process is completed.

Enabling the user to read the news

The user will receive an email with a spreadsheet containing the news and its links. The picture below shows the email:

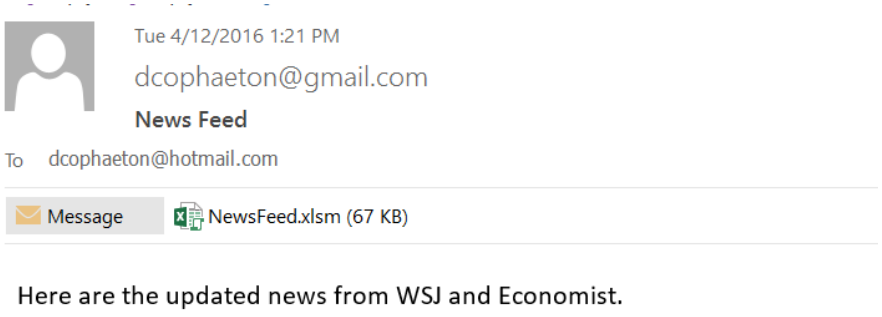

Figure 7 - Email sample

The user will open the excel file and will see the following screen:

Boeing, Iran Open Sale Talks On Planes	https://global-factiva-com.erl.lib.byu.edu/aa/default.aspx?pp=1&napc=p&sa_from=GL&ref=J0000000201604l2ec4c0002v	The Wall Street Journal
Trump and Clinton Show Strength in New York Poll	https://global-factiva-com.erl.lib.byu.edu/aa/default.aspx?pp=1&napc=p&sa_from=GL&ref=J0000000201604l2ec4c0002g	The Wall Street Journal
Ties That Bind Now Add to Viacom's Woes --- CEO Dauman ...	https://global-factiva-com.erl.lib.byu.edu/aa/default.aspx?pp=1&napc=p&sa_from=GL&ref=J0000000201604l2ec4c0002c	The Wall Street Journal
In South Korea, a Party by Many Other Names --- After defeat...	https://global-factiva-com.erl.lib.byu.edu/aa/default.aspx?pp=1&napc=p&sa_from=GL&ref=J0000000201604l2ec4c0002j	The Wall Street Journal
Nigeria's Brisk Growth Hits a Sudden Wall	https://global-factiva-com.erl.lib.byu.edu/aa/default.aspx?pp=1&napc=p&sa_from=GL&ref=J0000000201604l2ec4c0002f	The Wall Street Journal
Politics this week	http://www.economist.com/news/world-week/21696570-politics-week	The Economist
Business this week	http://www.economist.com/news/world-week/21696564-business-week	The Economist

Figure 8- Excel file with news

The user will click in the link he/she wants to see and the modified *Worksheet_FollowHyperlink* will call the *openPage* sub procedure, the *ReadForm* will be activated, the user will input his/her Learning Suite login information there and the sub procedure will open the link using the IE agent and checking for the login status. If it's a WSJ news it will connect to Factiva, if it's a The Economist news it will connect to The Economist site. In the end the form is hide, so that the user information is still available for our macro.

This finishes the whole process.

Discussion of Learning and Difficulties Encountered

Most of the commands and skills needed for this project were learned in class, this is mainly a web scraping and the classes we had cover most of the subjects needed. However there were some specific needs to accomplish the task that required google research:

- Set automatic execution of the macro: I was very concerned with this part of the project, I thought I would need to build a .BAT file and schedule the execution through windows. It was a good surprise to discover that excel has an easy way to set automatic execution, after a simple search on google the very first article about it was very straightforward and the code showed was simple. Figure 5, above, shows the code for automatic execution.
- Send email with an .xlsm file: it was another big concern that I had when I started the project. When I first ran the original project from vbaprojects.blogspot.com I thought it did a very good, but I was a little bit frustrate because I couldn't read the news by clicking in the links (the portals blocked access due to user authorization), that's why I decided to save the news in a excel file and use VBA to access the content. However, I was afraid that the email server would not allow an .xlms attachment in the email. In fact, when we open the attachment all the macros are disabled, but it is very easy to enable them and during the tests I didn't have any problem sending the attachment.
- Server authentication: it was a little tricky to authenticate in the news portals to access the news. The Learning Suite login doesn't give access to them, so I spent some time adjusting the login to go to the right addresses for WSJ and The Economist from the BYU library website so that BYU's user and password could be used to make the authentication.
- Page errors: The code is still fragile to errors happening in the external sites, for example this very morning, when I was testing, The Economist had a script that was not answering in its website and it would stop IE and make the code crash. To work around these problems I introduced several statements *On Error Resume Next* so that the code at least runs till the end. Nevertheless, if I had more time I would create more robust error handling code to try some different ways to complete the access.

- Creating a tab and button in the ribbon: I had to refer to the class video where we changed the ribbon and added a tab and a button for activate the macro. It was not difficult to do that task but since I had not attended that class I had no idea how to do that. It was really valuable to have the video available.
- Delete the created file: I had a few problems to delete the copy of the spreadsheet I created, several time Excel showed a message stating that it couldn't delete the file because the program couldn't access it. After a while I realized it was because of a sync problem between attaching the file in the email and trying to delete it. The problem was solved using the *Application.wait* method.

Overall, the project was a good way to solidify the concepts and subjects learned in class and at the same time expand the knowledge to solve different kinds of problems that we had not faced in the course.

Assistance

I needed assistance to understand what exactly was expected from this project, so I ran the original project available in the blogspot (but didn't look into the code) to see the result and get a feeling of what I was supposed to do.

As disclosed above I used the agent class and the sendGmail function that were provided by Professor Allen in class, with no changes to the respective source codes.

Besides that I solved all my questions and difficulties by searching in Google and did not require any help from the Professor, the TA's, or any other third party.