The Development of a Software Tool to Reduce Time Spent Identifying and Commenting Writing Errors in Research Papers

Charles L. McDonald, Jr., Ph.D.
Texas A & M University - Texarkana
charles.mcdonald@tamut.edu
Theresa A. McDonald, Ph.D.
Texas A & M University - Texarkana
theresa.mcdonald@tamut.edu

Abstract

Two professors developed a tool that identifies and comments writing errors in research papers. Written entirely in VBA, the tool addresses about 2,000 writing concerns and includes a rules manager for configurations. It operates as a Word Add-in ribbon that presents thirty buttons across three rows. The use of the buttons and the rationale for the code that supports each button's actions are addressed. In addition, modules and code segments that support operations behind the scenes (e.g., statistics and Flesch and Flesch-Kincaid scores) are discussed. The *Rules Manager* ribbon button opens a form to add, edit, or delete rules, and it provides a means to set rules as active or inactive. The code supporting the functions of the rules manager is described. This paper addresses the methodology for development of a software tool using VBA that can assist faculty by identifying and commenting many writing concerns in research papers.

Introduction

In 2006, two professors felt that their productivity was being suppressed by spending too many hours commenting repetitive grammar errors in electronically submitted research papers. These professors developed a software-based editing tool that provided consistency and detail in commenting writing concerns to better serve their students. The tool was named Edit Assist (McDonald C. & McDonald, T.). For several years, the tool was used by ScanMyDocument.com to scan and comment uploaded research papers and dissertations. During this time, the largest uploaded document was an 834-page dissertation from an English as a second language (ESL) student. After 20-minutes of processing, the document was returned with more than 2,800 comments. Presently, Edit Assist identifies and comments:

- About 900 Clichés, colloquial expressions, and conversational writing issues
- Misuse of conjunctive adverbs
- Overuse of common words
- Use of passive voice
- Too many words in a sentence
- Too many sentences in a paragraph
- One sentence paragraphs
- Words that are inappropriate, repeated, or not listed in the dictionary
- Inappropriate word choices for starting sentences
- Incorrect "a" and "an" usage

- Misuse of “not only...but also”, “neither...nor”
- About 540 additional grammar errors

The authors of this paper have continued to develop and enhance Edit Assist in identifying and commenting writing errors in research papers. Much effort has been directed to creating and modifying rules to eliminate erroneous comments that could misdirect students’ efforts in writing papers. These comments provide feedback to the students concerning their papers (Liu, Rios, Heilman, Gerard, & Linn). Comments are attached to sentences or parts of sentences to demonstrate to the students the problem area.

Findings

Edit Assist was written entirely in Microsoft’s Visual Basic for Applications (VBA), which is fully supported in Office 2016 (Anderson). The use of VBA enables users to scan a document without exiting Microsoft Word. The installation of Edit Assist only requires code to be added to the Normal template (Normal.dotm) and to copy the EAdb.mdb file to a folder. Edit Assist has operated for more than ten years on Microsoft’s Word and Windows releases without requiring code changes for compatibility. Concerning performance, a 1,000 word submission can usually be scanned in less than 30 seconds. A utility is available via a ribbon button to customize comments (button text and comment text) and manage the add, edit, delete, and activate/deactivate rule functions without closing the document. Edit Assist addresses about 1,400 writing concerns and includes a rules manager for configurations. It operates as a Microsoft Word Add-Ins that presents 29 buttons across a 3-row ribbon. The use of the buttons and the rationale for the code that supports each button’s action will be addressed. In addition, modules and code segments that support operations behind the scenes (e.g., statistics and Flesch and Flesch-Kincaid scores) will be discussed. The *Rules Manager* ribbon button opens a form to add, edit, or delete rules, and it provides a means to activate or deactivate rules. This paper addresses the methodology for development of a software tool using VBA that can assist faculty by identifying and commenting many writing concerns in research papers.

The ribbon buttons provide a means for the reviewer to highlight and comment text using a user-defined set of comments. In addition, the reviewer may highlight text and click the Blank Comment button to enter an unscripted comment. Below is a chart depicting the caption and comment for each ribbon button. Note that by placing the cursor on a ribbon button the tooltip would display the comment text.

Button Text	**Comment text**
Blank Comment	
Parallel Construction	Parallel construction error - The members of a series must be all nouns, all infinitives, all prepositional phrases, all gerunds, or all clauses.
Choice of Words	Questionable, inappropriate, weak choice, or order of words - needs revision or deletion

Button Text	Comment text
Grammar Errors	Text contains one or more grammar issues that limit the evaluation of content
Avoid Clichés	Avoid clichés, colloquial expressions, and conversational writing
Sentence Structure	Sentence has verb tense or structure problem - needs revision
Nebulous Content	Nebulous, rambling, or confusing content - needs clarification of meaning, focus on topic, or deletion
Personal Scenario	Avoid reflecting scenarios or personal opinions unsupported by references
Delete and Revise	Delete text and revise sentence if necessary
Redundant Content	Redundant content - addressed previously in document
Split Infinitive	Avoid split infinitives - revise as needed
Punctuation Error	This text contains one or more errors in punctuation (e.g., a missing comma)
Incomplete Sentence	This is an incomplete sentence - revise as needed
Acronym not Defined	Acronym not properly defined (e.g., Quality of Service (QoS))
Wordy Writing	Avoid wordiness or disorganized focus on topic
Choppy Sentences	In formal writing, choppy sentences (rapid switching of topics using short sentences) are to be avoided. Consider adding conjunctions, clauses, subjects, and/or verbs to make your writing more intricate.
Avoid Using Slashes	Avoid using slashes to imply word choices
Weak Writing	Weak or nebulous writing makes an evaluation of content difficult
Use Lower Case	Do not place words in proper case without rationale
Content Problem	Inaccurate, incomplete, or misleading content - content quality may be limited by this paper's organization
Review Stopped	The professor's review stopped here. This does not imply that all errors above or below this point were commented. Consider contacting the Success Center for assistance in organizing content and developing sentences.
Reference Format	References contain incorrect formatting - visit the APA Style Form Guide site at www.apastyle.org. (Perhaps links to the required online references are missing)
Delineate this Topic	Additional narrative needed to clarify or expand meaning of content
Citation Error	In-text citations are missing or contain incorrect APA formatting
Unrelated Topic	This content does not support this paper's topic

Microsoft Word's AutoOpen sub procedure runs each time Word is started. The AutoOpen sub calls the Build_Menus sub to build 29 buttons across a 3-row ribbon, which is accessed via the Add-Ins tab. In addition, the Build_Menus sub can be called manually (e.g., after modifying the menu's content). The Build_Menus sub calls the Load_Arrays sub that will create and load seven arrays from seven Access tables contained in EAdb.mdb. The following code segment is used to dimension these arrays.

```
Dim Array_Menu(30, 2) As String ' Array_Menu
Dim Array_Grammar(900, 2) As String ' Array_Grammar
Dim Array_Adverbs(25, 2) As String ' Array_Adverbs
Dim Array_Overused(25, 2) As String ' Array_Overused
Dim Array_Clichés(900, 2) As String ' Array_Clichés
Dim Array_Passive(60) As String ' Array_Passive
Dim Array_Misc(20) As String ' Array_Misc
```

The Build_Menus sub only uses the first array, Array_Menu. It is loaded with the ribbon's menu items and either an associated comment or a function call. The data in this array is used to populate the 3-row ribbon. Twenty-five of these buttons are dedicated to assist the reviewer by placing predefined text in a comment linked to the highlighted text. The code fragment from the Build_Menus sub used to generate the first menu ribbon is listed below. Note that when one of these ribbon buttons is clicked, the Post_Comment sub would be called to post the comment linked to the highlighted text.

```
Sub Build_Menus() ' Use to build or rebuild the menu ribbon
    Dim MyBar As CommandBar, Eye As Integer, MyButton As
    CommandBarControl
    Set MyBar = CommandBars.Item(3)
    ByPassMsgs = False
    LoadArray_Overuseds
    Set MyBar = CommandBars.Add(Name:="Edit Assist")
    MyBar.Visible = True
    MyBar.Position = msoBarFloating
    For Eye = 1 To 10
        Set MyButton = MyBar.Controls.Add(Type:=msoControlButton)
        MyButton.Style = msoButtonCaption
        MyButton.Caption = Array_Menu(Eye, 1)
        MyButton.Tag = Eye
        MyButton.TooltipText = Array_Menu(Eye, 2)
        MyButton.OnAction = "Post_Comment"
    Next Eye
```

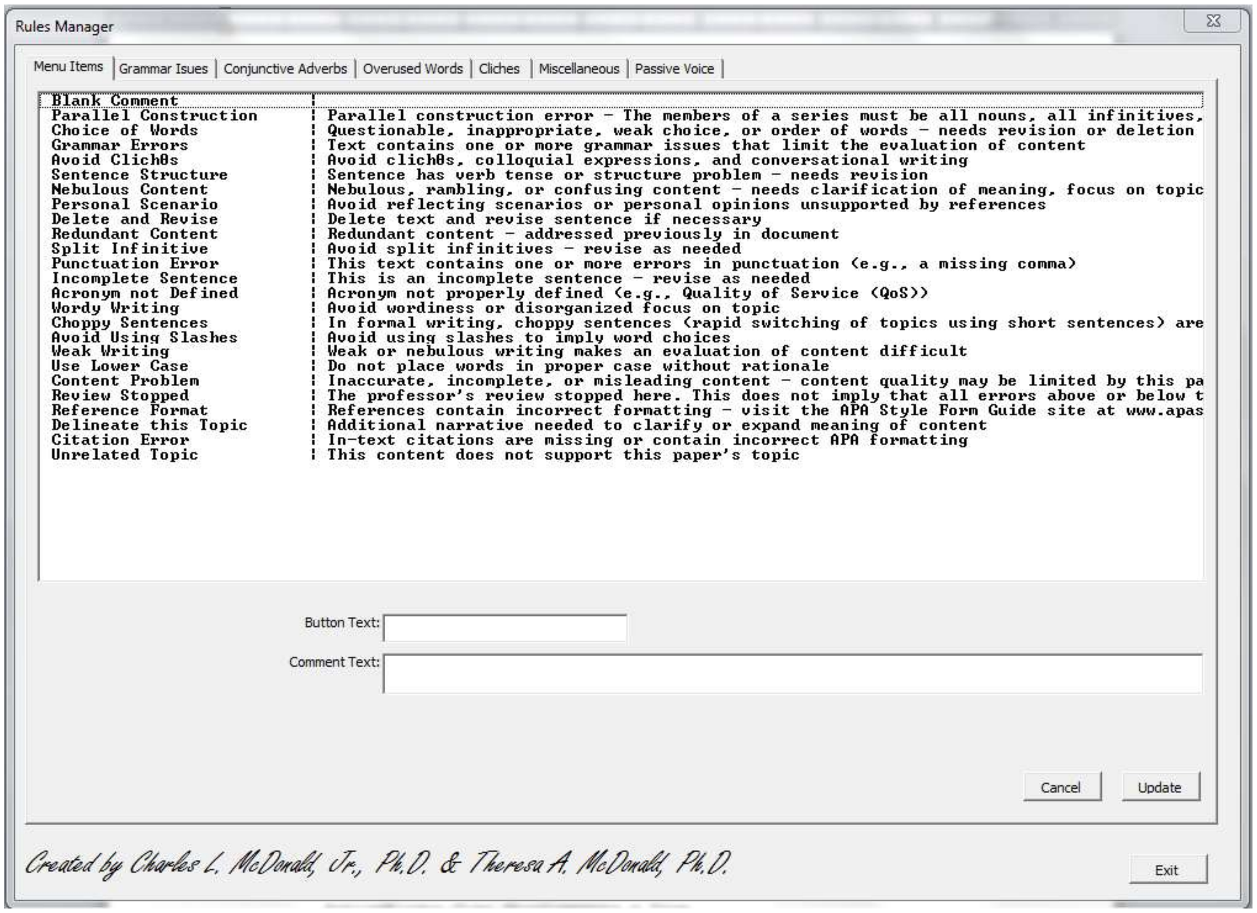
The *Count Comments* button calls the Count_Comments sub to provide a total comment count for the document, which has value as the newer releases of Microsoft Word do not reveal the comment numbers. After a scan completes, the summary report provides the total number of comments, but, additional comments are usually added during the document review and the updated total number of comments would be needed to update the summary report. The code for the Count_Comments sub is listed below.

```
Sub Count_Comments() ' Display total count of comments
    MsgBox ActiveDocument.Comments.Count
End Sub
```

The Delete_Comments sub provides a valuable addition. By highlighting a section of the document or the whole document, the *Delete Comments* button can remove all comments in the selected area with a single click. The code for the Delete_Comments sub is listed below.

```
Sub Delete_Comments() ' Delete selected comment(s)
    Dim Eye As Integer, Kay As Integer
    Dim MyRange As Range
    Set MyRange = ActiveDocument.Range(Start:=Selection.Start,
    End:=Selection.End)
    For Kay = 1 To 5  ' to remove possible duplicates
        For Eye = MyRange.Comments.Count To 1 Step -1
            MyRange.Comments(Eye).Delete
        Next
    Next Kay
End Sub
```

The *Rules Manager* Button is a recent enhancement that opens a form depicting a tool created for ease in adding, editing, or deleting rules. In addition, it provides a means to activate or deactivate rules. In this application, rules may include “search text” and content to be placed in the resulting comment box. Other rules are more subjective as they locate and comment passive voice, sentences with more than a set number of words, or paragraphs with more than a set number of sentences. Specifically, it provides an easy means to manage menu items and the rules that detect grammar issues, conjunctive adverbs, clichés, over-used words, passive voice, and miscellaneous items. A snapshot of the rules manager viewing the Menu Items tab is provided below.



```
Blank Comment               ¦
Parallel Construction       ¦ Parallel construction error - The members of a series must be all nouns, all infinitives,
Choice of Words             ¦ Questionable, inappropriate, weak choice, or order of words - needs revision or deletion
Grammar Errors              ¦ Text contains one or more grammar issues that limit the evaluation of content
Avoid Clichθs               ¦ Avoid clichθs, colloquial expressions, and conversational writing
Sentence Structure          ¦ Sentence has verb tense or structure problem - needs revision
Nebulous Content            ¦ Nebulous, rambling, or confusing content - needs clarification of meaning, focus on topic
Personal Scenario           ¦ Avoid reflecting scenarios or personal opinions unsupported by references
Delete and Revise           ¦ Delete text and revise sentence if necessary
Redundant Content           ¦ Redundant content - addressed previously in document
Split Infinitive            ¦ Avoid split infinitives - revise as needed
Punctuation Error           ¦ This text contains one or more errors in punctuation (e.g., a missing comma)
Incomplete Sentence         ¦ This is an incomplete sentence - revise as needed
Acronym not Defined         ¦ Acronym not properly defined (e.g., Quality of Service (QoS))
Wordy Writing               ¦ Avoid wordiness or disorganized focus on topic
Choppy Sentences            ¦ In formal writing, choppy sentences (rapid switching of topics using short sentences) are
Avoid Using Slashes         ¦ Avoid using slashes to imply word choices
Weak Writing                ¦ Weak or nebulous writing makes an evaluation of content difficult
Use Lower Case              ¦ Do not place words in proper case without rationale
Content Problem             ¦ Inaccurate, incomplete, or misleading content - content quality may be limited by this pa
Review Stopped              ¦ The professor's review stopped here. This does not imply that all errors above or below t
Reference Format            ¦ References contain incorrect formatting - visit the APA Style Form Guide site at www.apas
Delineate this Topic        ¦ Additional narrative needed to clarify or expand meaning of content
Citation Error              ¦ In-text citations are missing or contain incorrect APA formatting
Unrelated Topic             ¦ This content does not support this paper's topic
```

By clicking the *Scan This Document* button, the Start_Scan sub will call 16 additional sub procedures to process the document and comment identified concerns. The code for the Start_Scan sub is listed below. Note that in VBA, the call statement is not required (MSDN Microsoft, 2016).

```
Sub Start_Scan()
   Load_Arrays                     ' Loads the seven arrays
   Application.ScreenUpdating = False
   Convert_Citations               ' Convert citations to text
   Cleanup_Doc                     ' Set Print Layout and clean
                                   up document
   Words_Grammar_Check             ' Make Word perform grammar
                                   check
   OverUsed_Words                  ' Check for overuse of common
                                   words
   Cliches                         ' Scan for clichés - search
                                   is not case-sensitive
   Grammar_Rules                   ' Scan for grammar errors
                                   based on created rules
   Conjunctive_Adverbs             ' Scan for misuse of
                                   conjunctive adverbs
   In_Addition                     ' Scan for misuse of 'In
                                   addition...'
   Usage_of_A                         ' Scan for misuse of 'a'
   Usage_of_AN                        ' Scan for misuse of 'an'
   Check_Paragraphs                   ' Check for one sentence
                                      paragraphs, too many
                                      sentences in paragraph
   Long_Sentences_Plus                ' Check for not only...but
                                      also, neither...nor, long
                                      sentences
   Comment_Words                      ' Comment words that are
                                      inappropriate, repeated, or
                                      not listed in dictionary
   Passive_Voice                      ' Check for passive voice
   Summary_Report                     ' Generate Summary Report
   Selection.WholeStory
   Selection.ParagraphFormat.Space2 ' Double space document after
                                      scanning
   ActiveWindow.View.ShowComments = True
   Selection.GoTo what:=wdGoToLine, which:=wdGoToAbsolute,
   Count:=1
End Sub
```

The Load_Arrays sub is called first to create and load seven arrays from seven Access tables contained in EAdb.mdb. The first array, Array_Menu, is loaded with the ribbon’s menu items and either an associated comment or a function call. The data in this array is used to populate the 3-row ribbon. The second array, Array_Grammar, is used to address grammar issues based on user-created rules. Presently, about 540 user-created grammar rules have been added to the associated data table. Each row in this array is loaded with an active yes/no indicator, a search string, and the associated comment text. The third array, Array_Adverbs, is used to identify and comment the misuse of conjunctive adverbs. It contains a list of conjunctive adverbs, each with an active yes/no indicator. The fourth array, Array_Overused, contains a list of commonly overused words, each with an active yes/no indicator. The fifth array, Array_Clichés, contains a list of clichés that are to be avoided in formal writing, each with an active yes/no indicator. The sixth array, Array_Passive, contains a listing of passive voice verbs and a list of trailing strings that are indicators of passive voice. The list of trailing strings includes active yes/no indicators. The seventh array, Array_Misc, contains a collection of configuration options that include message text, maximum number of words in sentences, maximum number of sentences in a paragraph, and several yes/no settings. After the Load_Arrays sub has completed, the Start_Scan sub would continue by setting the screen update control to false, which allows faster program execution by prohibiting the screen from tracing through the many document scans.

The Convert_Citations sub is called to search for all citation fields and convert them to static text. This is performed to avoid program crashes when scanning for ASCII text across various Unicode control characters contained in citation fields. The code for the Convert_Citations sub is listed below.

```
Sub Convert_Citations() ' Find all citation fields and convert
them to static text.
```

```
    Dim MyField As Field
    For Each CF In ActiveDocument.StoryRanges
        For Each MyField In CF.Fields
            If MyField.Type = wdFieldCitation Then
                MyField.Select
                WordBasic.BibliographyCitationToText
            End If
        Next
    Next
End Sub
```

The Cleanup_Doc sub is called to set the document for Print Layout. In addition, it sets the balloon text to Arial 8-point font, and the comment text to Arial 10-point font. These settings are hard-coded, but the code can be edited to suit the reviewer's preferences. The code for the Cleanup_Doc sub is listed below.

```
Sub Cleanup_Doc() ' Set document for Print Layout and clean up
document
    If ActiveWindow.View.Type <> wdPrintView Then
        PrtLayTxt = "Submit document in Print Layout" & vbCr
        ActiveWindow.View.Type = wdPrintView
    End If
    ActiveWindow.ActivePane.View.Zoom.Percentage = 100
    With ActiveDocument.Styles("Balloon Text").Font
        .Name = "Arial"
        .Size = 8
    End With
    With ActiveDocument.Styles("Comment Text").Font
        .Name = "Arial"
        .Color = wdColorBlack
        .Bold = False
        .Size = 10
    End With
End Sub
```

The Words_Grammar_Check sub calls Microsoft Word to scan the document for grammar errors. Errors identified during this scan are commented when the Comment_Words sub runs. It would seem logical to comment the errors before leaving this sub, but it was discovered in early development that Word's grammar check runs in the background. Initially, when the errors were commented before leaving this procedure, the commenting procedure would complete before Word's grammar check completed identifying concerns; thus, several errors were frequently left uncommented. The code for the Words_Grammar_Check sub is listed below.

```
Sub Words_Grammar_Check() ' Make Word perform grammar check
    Dim MyRange As Range
    Selection.GoTo what:=wdGoToLine, which:=wdGoToAbsolute,
     Count:=1
    ' Send to top of document
    Set MyRange = Selection.Range
    MyRange.WholeStory
```

```
        Application.ResetIgnoreAll
        ActiveDocument.SpellingChecked = False
        ActiveDocument.GrammarChecked = False
        With Options
            .CheckSpellingAsYouType = True
            .CheckGrammarAsYouType = True
            .SuggestSpellingCorrections = True
            .SuggestFromMainDictionaryOnly = False
            .CheckGrammarWithSpelling = True
            .ShowReadabilityStatistics = False
            .IgnoreUppercase = True
            .IgnoreMixedDigits = True
            .IgnoreInternetAndFileAddresses = True
            .AllowCombinedAuxiliaryForms = True
            .EnableMisusedWordsDictionary = True
            .AllowCompoundNounProcessing = True
            .UseGermanSpellingReform = True
        End With
        ActiveDocument.ShowGrammaticalErrors = True
        ActiveWindow.View.ShowComments = True
        Languages(wdEnglishUS).SpellingDictionaryType = wdSpelling
    End Sub
```

The OverUsed_Words sub is called to check the document for an overuse of common words. The document is scanned using the array containing the list of commonly overused words. Each time a listed word is detected, a counter is incremented. At the end of the scan, any word with a count greater than 5 and the count is greater than the total word count divided by 100, would be identified as an overused word. Information about overused words identified during this scan would be included when building the Summary Report. The code for the OverUsed_Words sub is listed below.

```
    Sub OverUsed_Words() ' Check for overused words
        If Array_Misc(16) = "True" Then
            Dim DyLog As Dialog
            Set DyLog = Dialogs(wdDialogToolsWordCount) ' Total words
            in document
            DyLog.Execute
            TotWordCnt = DyLog.Words
            OverUsedText = ""
            Dim OverUsed As Integer
            Dim WordCnt As Single
            Selection.GoTo what:=wdGoToHeading, which:=wdGoToFirst
            ActiveDocument.UndoClear
            For Eye = 1 To 20
                WordCnt = 0
                Do While Selection.Find.Execute(FindText:=
                Array_Overused(Eye, 1), Forward:=True, Format:=False,
                Wrap:=wdFindStop) = True
                    WordCnt = WordCnt + 1
                Loop
                Selection.GoTo what:=wdGoToHeading,
                which:=wdGoToAbsolute, Count:=1
                If WordCnt > 5 And WordCnt > TotWordCnt / 100 Then
                    OverUsedText = OverUsedText & "The word '" &
                    Array_Overused(Eye, 1) & "' was used " & WordCnt &
                    " times." & vbCr
                    OverUsed = 1
                End If
            Next Eye
        End If
    End Sub
```

The Cliches sub scans the document and comments the use of clichés, colloquial expressions (slang), and conversational writing. Formal papers should be devoid of these elements. Presently, this sub searches for about 900 word strings that may include preceding or trailing spaces. The search is not case sensitive. When found, the located search string is highlighted and a comment is generated. The routine cycles searching the document for each search string. The same comment is used for all search strings, but the comment text can be edited via the rules manager tool. The comment presently states, "In formal writing, avoid clichés, colloquial expressions, and conversational writing." The code for the Cliches sub is listed below.

```
    Sub Cliches() ' Scan for clichés - search is not case-sensitive
        Dim Eye As Integer
        Selection.WholeStory
            For Eye = 1 To 1000 ' using Array_Clichés
                ActiveDocument.UndoClear
                Selection.Find.ClearFormatting
                With Selection.Find
                    .ClearFormatting
                    .Text = Array_Clichés(Eye, 1)
                    .Replacement.Text = ""
                    .Forward = True
                    .Wrap = wdFindContinue
                    .Format = False
                    .MatchCase = False ' Ignore case
                    .MatchWholeWord = False
                    .MatchWildcards = False
                    .MatchSoundsLike = False
                    .MatchAllWordForms = False
                End With
                Selection.WholeStory
                Do While Selection.Find.Execute(FindText:=
                Array_Cliches(Eye, 1), Forward:=True,
                Format:=False, Wrap:=wdFindStop) = True
                    Selection.Comments.Add Range:=Selection.Range,
                    Text:=Array_Misc(17)
                Loop
            Next Eye
        Erase Array_Clichés
    End Sub
```

The Grammar_Rules sub scans the document using the grammar rules created by users. Presently it searches for about 540 case sensitive search strings that may include preceding or trailing spaces. As a grammar issue is found, the search strings is highlighted and the associated comment text is placed in the comment. The routine cycles searching the document for each search strings, and as they are located, a comment is generated. The code for the Grammar_Rules sub is listed below.

```
Sub Grammar_Rules() ' Scan for grammar errors based on created
rules
    Dim Eye As Integer
    Selection.WholeStory
    For Eye = 1 To 900 ' using Array_Grammar
        ActiveDocument.UndoClear
        Selection.Find.ClearFormatting
        With Selection.Find
            .ClearFormatting
            .Text = Array_Grammar(Eye, 1)
            .Replacement.Text = ""
            .Forward = True
            .Wrap = wdFindContinue
            .Format = False
            .MatchCase = True  ' Set True...this needs to be case
             sensitive
            .MatchWholeWord = False
            .MatchWildcards = False
            .MatchSoundsLike = False
            .MatchAllWordForms = False
        End With
        Selection.WholeStory
        Do While
Selection.Find.Execute(FindText:=Array_Grammar(Eye, 1),
Forward:=True,
        Format:=False, Wrap:=wdFindStop) = True
            Selection.Comments.Add Range:=Selection.Range,
            Text:=Array_Grammar(Eye, 2)
        Loop
    Next Eye
    Erase Array_Grammar
End Sub
```

The Conjunctive_Adverbs sub searches the document for any misuse of conjunctive adverbs. When a conjunctive adverb is found, the sub looks for a preceding semicolon followed by a space and a comma and a space trailing the conjunctive adverb. If either of these conditions are false or if a conjunctive adverb is used to start a sentence, this comment will be generated; ” If a conjunctive adverb (e.g., however, therefore, hence...) is used to join sentences, it should have a semicolon in front of it and a comma behind it. If it is not being used as a conjunctive adverb, delete it.” The code for the Conjunctive_Adverbs sub is listed below.

```
Sub Conjunctive_Adverbs() ' Scan for misuse of conjunctive
adverbs
    If Array_Misc(14) = "True" Then
        Dim Eye As Integer
        Dim AC As String, BC As String
        Selection.WholeStory
        MyCharCnt = Selection.Characters.Count
        For Eye = 1 To 25
            ActiveDocument.UndoClear
            Selection.Find.ClearFormatting
            With Selection.Find
                .ClearFormatting
                .Text = Array_Adverbs(Eye, 2)
                .Replacement.Text = ""
                .Forward = True
                .Wrap = wdFindContinue
                .Format = False
                .MatchCase = False
                .MatchWholeWord = False
                .MatchWildcards = False
                .MatchSoundsLike = False
                .MatchAllWordForms = False
            End With
            Selection.WholeStory
            Do While Selection.Find.Execute(FindText:=
                Array_Adverbs(Eye, 2), Forward:=True,
                Format:=False,
                Wrap:=wdFindStop) = True
                BC = ActiveDocument.Range(Selection.Start - 2,
                Selection.Start).Text
                AC = ActiveDocument.Range(Selection.End,
                Selection.End
                + 2).Text
                If BC <> "; " Or AC <> ", " Then
                Selection.Comments.Add
                Range:=Selection.Range, Text:=Array_Misc(15)
            Loop
        Next Eye
        Erase Array_Adverbs
    End If
End Sub
```

The In_Addition sub searches the document for the misuse of "In addition". If "In addition" is used to start a sentence this rule will generate a comment if the search text does not have a trailing comma and a space or a space and the word "to" followed by a space. The comment states, "If 'In addition' is used to start a sentence, it must be followed either by a comma or the word 'to'." The code for the In_Addition sub is listed below.

```
Sub In_Addition() ' Scan for misuse of 'In addition...'
    Dim AC As String, BC As String
    Do While Selection.Find.Execute(FindText:="In addition",
    Forward:=True, Format:=False, Wrap:=wdFindStop) = True
```

```
            AC = ActiveDocument.Range(Selection.End, Selection.End +
            2).Text
            BC = ActiveDocument.Range(Selection.End, Selection.End +
            3).Text
            If AC <> ", " And BC <> " to" Then Selection.Comments.Add
            Range:=Selection.Range, Text:="If 'In addition' is used
            to start a sentence, it must be followed either by a
            comma or the word 'to'"
        Loop
        ActiveDocument.UndoClear
    End Sub
```

The Usage_of_A sub and the Usage_of_AN sub search for the misuse use of “a” or the misuse use of “an”. Initially, it was thought these would be easy rules to create as “an” usually precedes a vowel, but the exceptions made the rules complicated. As an example “a house…an hour”, or “an uncle…a unit”. These are the longest rules representing about two pages of code and, occasionally, a condition is discovered that requires an update. The code for the Usage_of_A and Usage_of_AN subs search for “ a ” or “ an “ and captures the trailing 14 characters in each instance. A select case statement with a series of cases are used in each sub to determine if “a” or “an” was properly used. The comment would be either “Replace 'a' with 'an'” or “Replace 'an' with 'a'”.

The Check_Paragraphs sub checks for two conditions. It searches for one-sentence paragraphs or too many sentences in a paragraph. If a one-sentence paragraph is found, the paragraph is highlighted and a comment is generated that states,” This appears to be a one-sentence paragraph.” The maximum number of sentences in a paragraph can be easily set using the rules manager. Presently, if a paragraph contains more than 15 sentences, the first word of the paragraph is highlighted and a comment is generated that states, “Long paragraphs frequently contain wordy writing, choppy sentences, redundant content, or a mix of topics.” The rules for one-sentence paragraphs and a paragraph with too many sentences can each be set as active or inactive. The code for the Check_Paragraphs sub is listed below.

```
    Sub Check_Paragraphs() ' Check for one sentence paragraphs, too
    many sentences in paragraph
        Dim NumSent As Long, MyWord As String
        Dim NewRange As Range, MySlash As String, MiniRange As Range,
        MyPos As Integer, MyChar As String
        Set MyRange = Selection.Range
        MyRange.WholeStory
        ParagCnt = MyRange.Paragraphs.Count
        NumSent = MyRange.Sentences.Count
        Dim aPara As Paragraph, SentCnt As Integer
        Dim aSent As Object, aWord As Object, bWord As String
        Dim WordCnt As Integer
        For Each aPara In MyRange.Paragraphs
        ActiveDocument.UndoClear
        SentCnt = 0
        For Each aSent In aPara.Range.Sentences ' counting sentences
        in each paragraph
```

```
            SentCnt = SentCnt + 1
        Next
        Set NewRange = aPara.Range
            If Array_Misc(7) = "True" Then  ' Check for long
            paragraphs
                If SentCnt > Array_Misc(8) Then ' Check for too many
                sentences in a paragraph
                    NewRange.Comments.Add Range:=NewRange,
                    Text:="This paragraph contains more than " &
                    Array_Misc(8) & " sentences"
                End If
            End If
            If Array_Misc(5) = "True" Then ' Check for one-sentence
            paragraphs
                WordCnt = NewRange.Words.Count ' Not an accurate
                count
                If NewRange > "   " And SentCnt < 2 And WordCnt > 10
                Then
                    If InStr(NewRange, ":/") = 0 And InStr(NewRange,
                    "=")=0 Then ' one sentence paragraphs
                        NewRange.Comments.Add Range:=NewRange,
                        Text:=Array_Misc(6)
                    End If
                End If
            End If
        Next
    End Sub
```

The Long_Sentences_Plus sub performs three checks. It searches for misuse of "not only…but also", misuse of "neither…nor", and sentences that are too long. If "not only" is used in a sentence, it must be followed by "but also" in that sentence or a comment will be added stating, "If 'but also' is used it must be preceded by '...not only…'." Consider deleting 'also' or revising sentence." If "neither" is used in a sentence, it must be followed by "nor" in that sentence or a comment will be added stating, "If 'neither' is used it must be paired with 'nor'". If a sentence contains more than a set number of words, the sentence will be highlighted and a comment generated. Presently the maximum number of words allowed in a sentence is set to 35, but this value is easily changed using the rules manager. For sentences with more than the set number of words, a comment is added stating, "This sentence appears to contain more than ## words. Long sentences with too much content tend to ramble and tire the reader." The code for the Long_Sentences_Plus sub is listed below.

```
    Sub Long_Sentences_Plus() ' Check for not only...but also,
    neither...nor, long sentences
        Dim MyRange As Range, NumSent As Long
        Set MyRange = Selection.Range
        MyRange.WholeStory
        Dim DyLog As Dialog, SentRange As Range
        Set DyLog = Dialogs(wdDialogToolsWordCount)
        Dim SelSentRange As Range
        Selection.WholeStory
```

```
NumSent = MyRange.Sentences.Count
ActiveDocument.UndoClear
For Eye = 1 To NumSent ' number of sentences in document
    ActiveDocument.Sentences(Eye).Select
    If Selection.Find.Execute(FindText:="but also",
    Forward:=True,
    Format:=False, Wrap:=wdFindStop) = True Then
        ActiveDocument.Sentences(Eye).Select
        If Selection.Find.Execute(FindText:="not only",
        Forward:=True, Format:=False, Wrap:=wdFindStop) =
        False Then
            ActiveDocument.Sentences(Eye).Select
            If Selection.Find.Execute(FindText:="but also",
            Forward:=True, Format:=False, Wrap:=wdFindStop) =
            True Then
                Selection.Comments.Add
                Range:=Selection.Range,
                Text:=Array_Misc(18) ' not only...but also
                message
            End If
        End If
    End If
    ActiveDocument.Sentences(Eye).Select
    If Selection.Find.Execute(FindText:="neither",
    Forward:=True,
    Format:=False, Wrap:=wdFindStop) = True Then
        ActiveDocument.Sentences(Eye).Select
        If Selection.Find.Execute(FindText:="nor ",
        Forward:=True,
        Format:=False, Wrap:=wdFindStop) = False Then
            ActiveDocument.Sentences(Eye).Select
            If Selection.Find.Execute(FindText:="neither",
            Forward:=True, Format:=False, Wrap:=wdFindStop) =
            True Then
                Selection.Comments.Add
                Range:=Selection.Range,
                Text:=Array_Misc(19) ' neither...nor message
            End If
        End If
    End If
    ActiveDocument.Sentences(Eye).Select
    DyLog.Execute
    WordCnt = DyLog.Words
    If Array_Misc(3) = "True" Then
        If WordCnt > Val(Array_Misc(4)) And WordCnt < 300
        Then 'check for long sentences
            On Error GoTo Errormanager
            If Len(Selection.Range) > 30 Then
            ActiveDocument.Comments.Add
            Range:=Selection.Range,
            Text:="This sentence appears to contain more
            than " & WordCnt & " words. Long sentences with
```

```
                too much content tend to ramble and tire the
                reader."
            End If
        End If
Errormanager:
    Next
    ActiveDocument.UndoClear
End Sub
```

The Comment_Words sub searches for inappropriate words, repeated words, or words not listed in Microsoft Word's dictionary, which includes concerns previously detected by the Words_Grammar_Check sub. For each discovery, the comment "Word inappropriate, repeated, or not listed in dictionary" is generated. The comment purposely avoids the word "misspelled" as author's names, medical terms, new expressions, or acronyms are frequently not found in Word's dictionary. If 200 of these concerns are commented, the routine would exit and this message is added to the summary report, "The scanner identifies up to 200 comments that identify words as inappropriate, repeated, or not listed in dictionary." The code for the Comment_Words sub is listed below.

```
Sub Comment_Words() 'Comment words that are inappropriate,
repeated, or not listed in dictionary
    If Array_Misc(12) = "True" Then
        BadSpellingWords = ""
        Dim MisWord As Range, WordCnt As Integer
        For Each MisWord In ActiveDocument.SpellingErrors
            ActiveDocument.UndoClear
            WordCnt = WordCnt + 1
            MisWord.Select
            MisWord.Comments.Add Range:=MisWord,
            Text:=Array_Misc(13)
            If WordCnt > 199 Then
                BadSpellingWords = "The scanner identifies up to
                200 comments that identify words as
                inappropriate, repeated, or not listed in
                dictionary."
                Exit For
            End If
        Next
    End If
End Sub
```

The Passive_Voice sub searches for the use of passive voice. When found, the text is highlighted and a simple hard-coded comment, "Use of passive voice", is added. In the Summary Report, a more detailed comment is added, "The use of passive voice was commented. Although the use of passive voice is not a grammar error, it is a stylistic issue that can weaken the clarity of writing. Although most reviewers prefer the active voice, the use of passive voice may be appropriate in some situations. A passive construction occurs when you make the object of an action into the subject of a sentence". Searching for passive voice can be set as active or inactive and the comment can be edited using the rules manager. The code for the Passive_Voice sub is listed below.

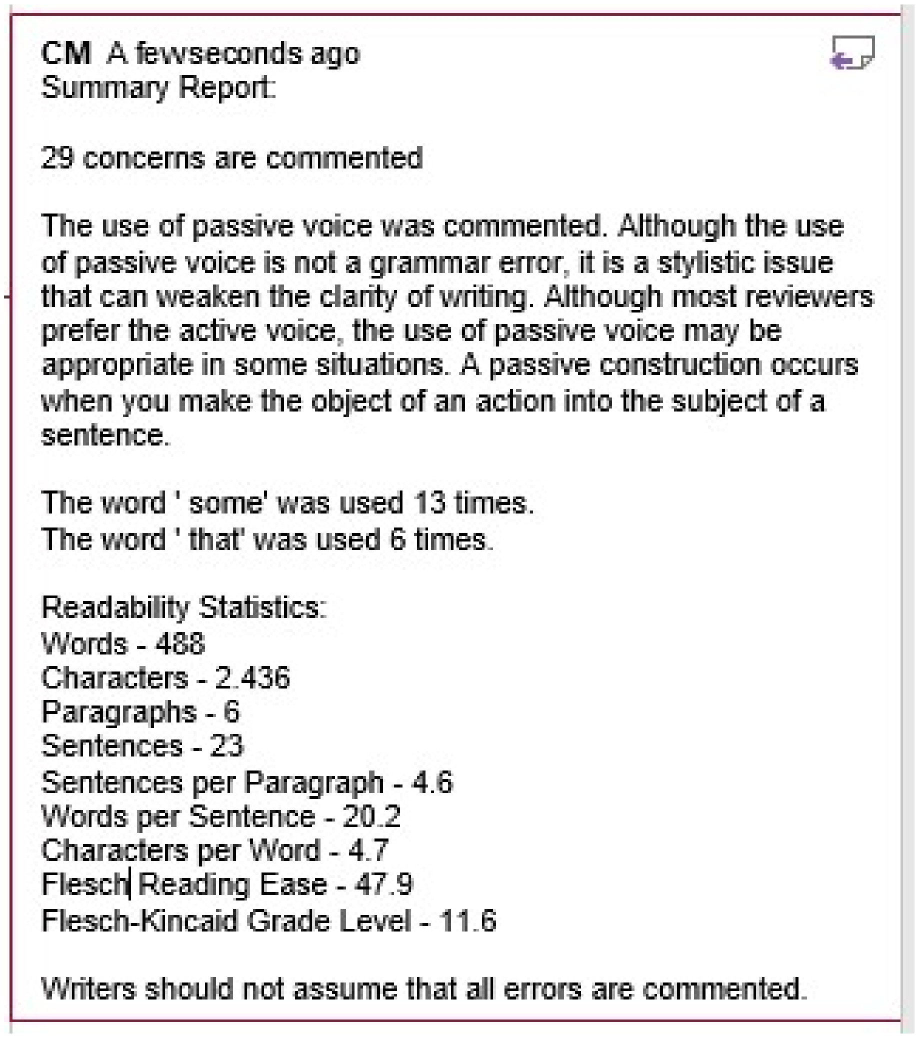
```
Sub Passive_Voice() ' Check for passive voice
    If Array_Misc(10) = "True" Then
        Dim Eye As Integer, Jay As Integer, MyText As String, Pos
        As Long
        Dim NextWord As String, MyPos As Integer, MyRange As
        Range, ComLen As Integer
        Dim Word1 As String, Word2 As String
        Selection.WholeStory
        Pos = Selection.End
        PassiveVoice = False
        PVCnt = 0
        On Error Resume Next
        For Eye = 0 To 22
            ActiveDocument.UndoClear
            Selection.Find.ClearFormatting
            With Selection.Find
                .ClearFormatting
                .Text = Array_Passive(Eye)
                .Replacement.Text = ""
                .Forward = True
                .Wrap = wdFindContinue
                .Format = False
                .MatchCase = False ' when searching - search is
                 not case-sensitive
                .MatchWholeWord = False
                .MatchWildcards = False
                .MatchSoundsLike = False
                .MatchAllWordForms = False
            End With
            Selection.WholeStory
            Do While
            Selection.Find.Execute(FindText:=Array_Passive(Eye),
            Forward:=True, Format:=False, Wrap:=wdFindStop) =
            True
                If Pos < Selection.End + 20 Then Exit Do
                NextText = ActiveDocument.Range(Selection.End,
                Selection.End + 20).Text
                Word1 = ActiveDocument.Range(Selection.End,
                Selection.End + 2).Text
                Word2 = ActiveDocument.Range(Selection.End,
                Selection.End + 4).Text
                If Word1 <> "a " Then
                    If Word2 <> "the " And Word2 <> "one " And
                    Word2 <> "red " Then
                    For Jay = 23 To 60
                        If Array_Passive(Jay) <> "" Then
                            MyPos = InStr(1, NextText,
                            Array_Passive(Jay))
                            If MyPos > 0 Then
```

```
                                        ComLen = MyPos +
                                        Len(Array_Passive(Jay)) - 1
                                        If Jay > 33 Then MyPos =
                                        Len(Array_Passive(Jay))
                                        Set MyRange =
                                        ActiveDocument.Range
                                        (Selection.Start +1,
                                        Selection.End + ComLen)
                                        Selection.Comments.Add MyRange,
                                        Text:="Use of passive voice"  '
                                        Text:=Array_Misc(11)
                                        PVCnt = PVCnt + 1
                                        If PVCnt > 199 Then Exit Sub
                                        PassiveVoice = True
                                    End If
                                End If
                            Next Jay
                            End If
                        End If
                    Loop
Errormanager:
        Next Eye
    End If
End Sub
```

The Summary_Report sub creates a summary report that represents the first comment at the top of the document. The first entry reveals the total number of comments in the document. If the detection of passive voice is active and the use of passive voice was detected in the document, a message addressing the use of passive voice is included. If any overused words were detected, they are included in the report stating how many times they appeared in the document.

Document statistics are provided that include the Flesch and Flesch-Kincaid Readability and Grade Level scores. The Flesch–Kincaid readability tests were developed under contract to the US Navy in 1975, which established it as a standard that has gained wide acceptance (The Flesch). The algorithms utilize total number of words, sentences, and syllables to determine scores. Interestingly, the value used for total syllables is derived from vowel counts and word lengths. If Unicode characters are detected during the readability scan, statistical calculations are abandoned and the text, “Statistics were not included as Unicode characters embedded in this document were interpreted as a non-English language.”

The summary report concludes by stating, “Writers should not assume that all errors are commented.” Below is the summary report from the test document.


CM A fewseconds ago
Summary Report:

29 concerns are commented

The use of passive voice was commented. Although the use of passive voice is not a grammar error, it is a stylistic issue that can weaken the clarity of writing. Although most reviewers prefer the active voice, the use of passive voice may be appropriate in some situations. A passive construction occurs when you make the object of an action into the subject of a sentence.

The word ' some' was used 13 times.
The word ' that' was used 6 times.

Readability Statistics:
Words - 488
Characters - 2.436
Paragraphs - 6
Sentences - 23
Sentences per Paragraph - 4.6
Words per Sentence - 20.2
Characters per Word - 4.7
Flesch Reading Ease - 47.9
Flesch-Kincaid Grade Level - 11.6

Writers should not assume that all errors are commented.

The code for the Summary_Report sub is listed below.

```
Sub Summary_Report() ' Generate Summary Report
    Dim StrTotCnt As String, MyErrCnt As Integer, StatText As
    String, Eye As Integer, OneRpt As Integer
    OneRpt = 0
    Application.ScreenUpdating = False
    MyErrCnt = ActiveDocument.Comments.Count + 1 ' Count comments
    RptText = "Summary Report:" & vbCr ' Start Building RptText
    If MyErrCnt > 0 Then RptText = RptText & vbCr & MyErrCnt & "
    concerns are commented" & vbCr
    StrTotCnt = FormatNumber(TotWordCnt, 0, , , vbUseDefault)
    If BadSpellingWords <> "" Then RptText = RptText & vbCrLf &
    BadSpellingWords & vbCrLf ' Words inappropriate, repeated, or
    not listed in dictionary
    If PVCnt > 0 Then RptText = RptText & vbCrLf & Array_Misc(11)
    & vbCrLf
    If PVCnt > 199 Then
```

```
            RptText = RptText & vbCr & "Identifying Passive Voice
            stopped at 200 comments."
        End If
        PVCnt = 0
        Dim rs As Variant ' Readability Statistics
        On Error GoTo Errormanager
        StatText = " Readability Statistics:" & vbCr
        For Each rs In Documents(1).ReadabilityStatistics
            If rs <> "Passive Sentences" Then
                If rs <> "Words" Then ' rs.value is wrong for number
                of words
                    StatText = StatText & rs.Name & " - " & rs.Value
                    & vbCr
                Else
                    StatText = StatText & rs.Name & " - " & StrTotCnt
                    & vbCr
                End If
            End If
        Next rs
    Errormanager:
        If Err.Number = 4658 Then 'Unicode characters detected in
        document
            StatText = vbCr & "Statistics were not included as
            Unicode characters embedded in document were interpreted
            as a non-English language."
        End If
        If OneRpt = 0 Then ' Work-around the on error goto problem
            If OverUsedText > " " Then OverUsedText = vbCr &
            OverUsedText
            RptText = RptText & OverUsedText & vbCr & StatText & vbCr
            & Array_Misc(1)   ' summary comment text
            OneRpt = 1
        End If
        ' Display summary comment
        Selection.GoTo what:=wdGoToLine, which:=wdGoToAbsolute,
        Count:=1
        If Err.Number = 4605 Then Selection.GoTo what:=wdGoToPage,
        which:=wdGoToRelative, Count:=1 ' page down if graphics are
        at top of first page
        Selection.Comments.Add Range:=Selection.Range, Text:=RptText
        Selection.GoTo what:=wdGoToLine, which:=wdGoToAbsolute,
        Count:=1
        Selection.Text = Array_Misc(0) & vbCr & vbCr
        Selection.Font.Color = wdColorBlack
        Selection.Font.Size = 12
        Selection.Font.Bold = False
        With Selection
            For Eye = 1 To ActiveDocument.Comments.Count
                ActiveDocument.Comments(Eye).Author = Array_Misc(2) '
                Place initials in comments
            Next Eye
        End With
        Options.DefaultHighlightColorIndex = wdYellow
        Selection.Range.HighlightColorIndex = wdYellow
        Selection.GoTo what:=wdGoToLine, which:=wdGoToAbsolute,
        Count:=1
        Application.ScreenUpdating = True
    End Sub
```

There were 29 errors discovered in the test document using the Edit Assist tool. As an experiment, the test document was uploaded to four online grammar check sites. The http://www.grammarcheckforsentence.com/ site identified 15 errors in the document, but only 7 were valid. The http://sentencechecker.com/ site identified 5 concerns, but only 3 were valid. The http://www.polishmywriting.com/ site found 19 concerns, but only 6 were valid. The www.paperrater.com site reported 3 possible misspelled words and 2 suggestions. The first suggestion was to use "phone" for "phones" and the second suggestion was to insert a comma between "phone" and "dialed". Both of these suggestions were incorrect.

Edit Assist continues to evolve as new rules are added. One feature that has been considered would be to evaluate references and in-text citations for APA style. A significant enhancement would be to add an artificial intelligence element that could analyze a research paper's content and insert meaningful comments concerning the paper's sequencing of ideas through well-developed paragraphs, thoughtful supporting detail in well-structured sentences, and smooth transitions that enhance the paper's organization. The development of this element is beyond the scope of the author's effort.

Conclusion

Several faculty have found value in Edit Assist for more than ten years to reduce the drudgery in commenting common (and repetitive) writing issues. When used as a tool to provide feedback concerning writing issues, faculty reported that papers submitted for feedback usually scored about 30 points higher on final submissions, but statistics were not available to validate this claim. As a result of this development and a trial use of the tool by selected faculty, Edit Assist is now available to faculty in the College of Business, Engineering, and Technology at Texas A&M University – Texarkana. In addition, a student version offering ribbon buttons for "Scan Document" and "Delete Comments" has been placed throughout the campus for open use. There is an opportunity for a study to validate the effectiveness of this tool not only concerning improved grades, but improved writing skills in formal business writing.

References

Anderson, Tim. "You Lucky Devs: It's Microsoft Office 2016…and VBA Lives on." *The Register*, 1 Oct. 2015, http://www.theregister.co.uk/2015/10/01/.

Liu, Rios, Heilman, Gerard, & Linn. "Validation of Automated Scoring of Science Assessments." *Journal of Research in Science Teaching*, vol. 53, no. 2, 2016.

McDonald, Charles & McDonald, Theresa. “A Technology-Based Solution to Reduce Time Spent Identifying and Commenting Writing Errors in Research Papers.” *ACET Journal of Computer Education and Research,* vol. 6, no. 1, 2010.

“The Flesch Grade Level Readability Formula.” 31 Oct. 2016. *ReadabilityFormulas.com*. http://www.readabilityformulas.com/flesch-reading-ease-readability-formula.php.