# 1 Introduction

After a lot of testing it was decided to work again with “bubble-comments” but to use it with support of a few macros and specific view-settings in Word.

The macros can be found at the end of this document. They are fairly simple so that you will most probably understand quickly what they do before copying them e.g. into your Normal.dotm fiele. Of course, you use them at your own risk.

# 2 Step-By-Step

## 2.1 Getting the macros to work

We will not go into the details of how to use macros in Word. But in short, the steps are as follows:

1. Go to the end of this document and copy the macro text
2. In Word, press Alt-F11 to open the Visual Basic editor
3. Locate the “Normal project”
4. If not already present, create a “module” by right-clicking on “Normal”
5. Choose that (or any existing) module on the left side and then paste the copied macro text into the window on the right side

It may then look as follows:



## 2.3 The correct view

When editing or viewing (large) comments in Print View or when opening a comment in draft view, Word opens by default the “Review Pane”. This view shows not only the review comments but also all tracked changes. The review pane is slow and not really usable for the ASN.1 review. However, when staying in draft view and when the “comment pane” is open, the review pane will usually remain closed. It is therefore suggested to ...

1. Review in Normal View (also known as Draft View)
2. View and edit comments in the “Comment Pane”
3. Use the macro the macro “switchToDraftViewWithCommentPane()” to get the right view.

## 2.4 Adding new RIL items

It causes sometimes problems if several comments are associated with the same text in the document. To avoid this, a comment should be added to one or a few words – NOT to entire sections or paragraphs

1. Comments should be added to one or a few words – NOT to entire sections or paragrpahs.

As mentioned above, review comments are captured in “bubble comments”. But to improve readability and searchability, they should follow a common pattern. To achieve this, we made a little macro called “RILAddComment”. Before calling it, make sure to set the author name in word correctly.

1. Set the Author name in word to the name of your company and (optionally) your first name in parenthesis. E.g.: *Ericsson (Henning)*.

The macro creating the actual RIL comment will then set a preliminary RIL tag and the “delegate” field accordingly.

1. Use the macro “RILAddComment” to add a new review comment

When creating the RIL comment, you find yourself in the “Comment Pane” inside the newly created comment. Move the cursor a few lines up to fill the missing fields of this RIL entry. Detailed instructions on this point will be provided by the specification rappoerteur. But two important points should be given also here:

1. Ensure that the RIL number begins with the letter identifying your company name (e.g. “E” for “Ericsson”) followed by a three digit number that is unique within your company.
2. The red fields shall only be edited by the specification or work item rapporteur that moderates the review.

A short description of the identified issue and a proposed change should preferably be put directly into the comment. Since “Tracked Changes” are not supported inside comments, you may use colored/strike-through text or highlighting to show what you changed.

1. Preferably add the proposed change into the comment

## 2.5 If needed: Additional sections or separate files

If the proposed change requires a more space than what you can fit into the RIL comment itself, you may add a section specific to your RIL-comment at the end of the document. The macro “RILAddDedicatedCommentSection” supports you in doing that: With your cursor placed into a RIL comment, execute the macro. It will create a new section at the end of the document and with the name of your RIL comment as heading. It will also add a hyperlink to your RIL comment by which you and others can quickly navigate from the comment to the new section. Note that all meta information shall still be in the comment itself.

1. If the RIL comment does not have sufficient space, add a dedicated RIL-item section by the macro “RILAddDedicatedCommentSection”.

If even that separate section for not offer sufficient room for your change or if the problem analysis requires a lot of space, you may create a dedicated document. If so, indicate that in your RIL comment. Preferably add a (draft) document directly to FTP in a subfolder containing your RIL-item’s number.

1. If you need even more space, create a separate document. Preferably store a draft in the sub-folder (in FTP) starting with the RIL-ID (e.g. E123) and add a relative hyperlink from the review comment to that file/folder.

To search for RIL comments or to find e.g. all RIL comments marked as “ToDisc” it is recommended to open the Navigation Pane, to use the search function therein and to click the “Result” button in the pane to get a clickable list of all matching items.

1. Use the search function in the Navigation Pane, to navigate across and search for RIL items.

The resulting view including “Comment Pane” and “Navigation Pane” in “Result-view” is shown in the following figure:



## 2.2 Adding Short-Cuts

You will probably use the macro for adding new RIL comments quite frequently. You may hence want to add a keyboard shortcut for this and other macros:

1. In Word, right-click an empty area in any Ribbon and choose “Customize the Ribbon”. At the bottom of that window click “Keyboard Shortcuts: Choose”. On the upper left side choose “Macros”. Then choose a macro on the upper right side and assign a keyboard shortcut on the lower half. Don’t forget to “Assign” it before closing.

Recommended shortcuts could be:

Ctrl-Shift-Alt-A RILAddComment Add a new RIL comment at the current cursor position in the document

Ctrl-Shift-Alt-D RILAddDedicatedCommentSection Add a subsection at the end of the document (call while inside a RIL comment)

Ctrl-Shift-Alt-C switchToDraftViewWithCommentPane Setup the correct view to review and edit comments

Ctrl-Shift-Alt-E RILCommentsExtract Extract all RIL comments from the current file to a new Excel Worksheet

# Annex - Macro souce code

Option Explicit

Sub RILAddComment()

' Creates a new bubble-comment at the current selection and pre-fills it with the required fields.

' Consider assigning this to Ctrl-Alt-Shift-A

 correctSettings

 RemoveBookmarksFromComments ' Comments in bookmarks are useless but can cause Word to delete all subsequent comments.

 If (Selection.Information(wdInCommentPane)) Then

 MsgBox "Please select a part of the main document first. RIL documents are not meant to be inside other (RIL-)comments", vbOKOnly, "Error"

 Exit Sub

 End If

 If (Selection.StoryType <> wdMainTextStory) Then

 MsgBox "Please select a part of the main document first. ", vbOKOnly, "Error"

 Exit Sub

 End If

 ' Disable screen updates

 Application.ScreenUpdating = False

 switchToDraftViewWithCommentPane

 ' Add the comment:

 Selection.Comments.Add Range:=Selection.Range

 ' and populate it with the tags:

 Selection.Font.Bold = True

 Selection.TypeText text:="[RIL]"

 Selection.Font.Bold = False

 Selection.TypeText text:=": " & UCase(Application.UserInitials) & "xyz " ' & vbNewLine

 Selection.Font.Bold = True

 Selection.TypeText text:="[Delegate]"

 Selection.Font.Bold = False

 Selection.TypeText text:=": " & Application.UserName & " " ' & vbNewLine

 Selection.Font.Bold = True

 Selection.TypeText text:="[WI]"

 Selection.Font.Bold = False

 Selection.TypeText text:=": " ' & vbNewLine

 Selection.Font.Bold = True

 Selection.TypeText text:="[Class]"

 Selection.Font.Bold = False

 Selection.TypeText text:=": " ' & vbNewLine

 Selection.Font.Bold = True

 Selection.Font.ColorIndex = wdRed

 Selection.TypeText text:="[Status]"

 Selection.Font.Bold = False

 Selection.TypeText text:=": "

 Selection.TypeText text:="ToDo "

 Selection.Font.ColorIndex = wdAuto

 ' Selection.TypeText text:=vbNewLine

 Selection.Font.Bold = True

 Selection.TypeText text:="[TDoc]"

 Selection.Font.Bold = False

 Selection.TypeText text:=": None " ' & vbNewLine

 Selection.Font.Bold = True

 Selection.Font.ColorIndex = wdRed

 Selection.TypeText text:="[Proposed Conclusion]"

 Selection.Font.Bold = False

 Selection.TypeText text:=": "

 Selection.Font.ColorIndex = wdAuto

 Selection.TypeText text:=vbNewLine

 Selection.Font.Bold = True

 Selection.TypeText text:="[Description]"

 Selection.Font.Bold = False

 Selection.TypeText text:=": " & vbNewLine

 Selection.Font.Bold = True

 Selection.TypeText text:="[Proposed Change]"

 Selection.Font.Bold = False

 Selection.TypeText text:=": " & vbNewLine

 Selection.Font.Bold = True

 Selection.TypeText text:="[Comments]"

 Selection.Font.Bold = False

 Selection.TypeText text:=": " & vbNewLine

 ' Enable screen updates again

 Application.ScreenUpdating = True

End Sub

Sub toggleCommentPane()

 ' Enables or disables the (old fashioned) comment pane.

 ' Consider assigning this to Ctrl-Alt-Shift-C

 WordBasic.ViewAnnotations

End Sub

Sub goToNextComment()

 ' This macro jumps to the next comment in the document (if any). Unlike the in-built button, it does not open the review pane

 ' Consider assigning this to Ctrl-Alt-Shift-N

 Application.ScreenUpdating = False

 Selection.GoTo What:=wdGoToComment, Which:=wdGoToNext, Count:=1

 switchToDraftViewWithCommentPane

 ' The following will never happen since the previous command enforces draft view

 If ((ActiveWindow.View.Type = wdPrintView) And (WordBasic.ViewAnnotations = -1)) Then

 ' In Print View the comment pane does not update automatically.

 ' Hence, if the comment pane is visible, disable and enable it to jump to the right change.

 WordBasic.ViewAnnotations

 WordBasic.ViewAnnotations

 End If

 Application.ScreenUpdating = True

End Sub

Sub goToPreviousComment()

 ' This macro jumps to the previous comment in the document (if any). Unlike the in-built button, it does not open the review pane

 ' Consider assigning this to Ctrl-Alt-Shift-P

 Application.ScreenUpdating = False

 Selection.GoTo What:=wdGoToComment, Which:=wdGoToPrevious, Count:=1

 switchToDraftViewWithCommentPane

 ' The following will never happen since the previous command enforces draft view

 If ((ActiveWindow.View.Type = wdPrintView) And (WordBasic.ViewAnnotations = -1)) Then

 ' In Print View the comment pane does not update automatically.

 ' Hence, if the comment pane is visible, disable and enable it to jump to the right change.

 WordBasic.ViewAnnotations

 WordBasic.ViewAnnotations

 End If

 Application.ScreenUpdating = True

End Sub

Sub editComment()

 ' Opens the current/next comment at/after the current selection

 Dim currentSelection As Range

 Dim commentRange As Range

 ' Don't bother updating the screen:

 Application.ScreenUpdating = False

 Set currentSelection = Selection.Range

 Set commentRange = Selection.Range

 switchToDraftViewWithCommentPane

 ' Locate the next comment.

 commentRange.Collapse direction:=wdCollapseStart

 commentRange.MoveEnd Unit:=wdStory

 If (commentRange.Comments.Count > 0) Then

 commentRange.Comments(1).Edit

 End If

 ' and then start displaying something again:

 Application.ScreenUpdating = True

End Sub

Sub RILCommentsExtract()

 Dim oneComment As Comment

 Dim cText As String

 Dim importHeading As VbMsgBoxResult

 Dim nrofImportedComments As Integer

 nrofImportedComments = 0

 Dim skippedComments As String

 skippedComments = ""

 Dim ObjExcel As Object, ObjWorkBook As Object, ObjWorksheet As Object

 Dim excelRow As Integer

 excelRow = 1

 importHeading = MsgBox("Would you like to see the headings of the sections in which each RIL comment was? If so, it may take quite a bit longer to run the macro.", vbYesNo, "Fast or detailed?")

 Set ObjExcel = CreateObject("EXCEL.APPLICATION")

 ObjExcel.Visible = True

 Set ObjWorkBook = ObjExcel.Workbooks.Add

 Set ObjWorksheet = ObjWorkBook.Worksheets(1)

 With ObjWorksheet

 .Activate

 .Cells(excelRow, 1).Value = "ID"

 .Cells(excelRow, 2).Value = "Delegate"

 .Cells(excelRow, 3).Value = "Work Item"

 .Cells(excelRow, 4).Value = "Class"

 .Cells(excelRow, 5).Value = "TDoc"

 .Cells(excelRow, 6).Value = "Status"

 .Cells(excelRow, 7).Value = "Processed"

 .Cells(excelRow, 8).Value = "Proposed Conclusion"

 .Cells(excelRow, 9).Value = "Description"

 .Cells(excelRow, 10).Value = "Proposed Change"

 .Cells(excelRow, 11).Value = "Comments"

 .Cells(excelRow, 12).Value = "Section"

 End With

 For Each oneComment In ActiveDocument.Comments

 cText = oneComment.Range.text

 cText = Replace(cText, Chr(5), "")

 If (InStr(1, cText, "[RIL]", vbBinaryCompare) = 1) Then

 ' this comment is a RIL comment

 With ObjWorksheet

 excelRow = excelRow + 1

 .Activate

 .Cells(excelRow, 1).Value = extractRowsByTag(cText, "[RIL]:", "[")

 .Cells(excelRow, 2).Value = extractRowsByTag(cText, "[Delegate]:", "[")

 .Cells(excelRow, 3).Value = extractRowsByTag(cText, "[WI]:", "[")

 .Cells(excelRow, 4).Value = extractRowsByTag(cText, "[Class]:", "[")

 .Cells(excelRow, 5).Value = extractRowsByTag(cText, "[TDoc]:", "[")

 .Cells(excelRow, 6).Value = extractRowsByTag(cText, "[Status]:", "[")

 .Cells(excelRow, 7).Value = getColorCode(oneComment, 8, 10)

 .Cells(excelRow, 8).Value = extractRowsByTag(cText, "[Proposed Conclusion]:", vbCr & "[")

 .Cells(excelRow, 9).Value = extractRowsByTag(cText, "[Description]:", vbCr & "[")

 .Cells(excelRow, 10).Value = extractRowsByTag(cText, "[Proposed Change]:", vbCr & "[")

 .Cells(excelRow, 11).Value = extractRowsByTag(cText, "[Comments]:", vbCr & "[")

 If (importHeading = vbYes) Then

 .Cells(excelRow, 12).Value = Replace(getCurrentHeading(oneComment.Scope), vbTab, " ")

 End If

 End With

 nrofImportedComments = nrofImportedComments + 1

 Else

 skippedComments = skippedComments & vbNewLine & "'" & Mid(cText, 1, min(30, Len(cText))) & "'"

 End If

 Next

 beautifyExcelWorkbook ObjWorksheet

 InputBox Prompt:="Skipped comments", Title:="Imported " & nrofImportedComments & " of " & ActiveDocument.Comments.Count & " comments.", Default:=skippedComments

End Sub

Function min(aFirstValue As Integer, aSecondValue As Integer) As Integer

 If (aFirstValue < aSecondValue) Then

 min = aFirstValue

 Else

 min = aSecondValue

 End If

End Function

Function extractRowsByTag(aSourceText As String, aStartTag As String, anEndTag As String) As String

 ' Extracts from aSourceText the text in between aStartTag and anEndTag, i.e., without the two tags.

 Dim startPos As Integer

 Dim endPos As Integer

 Dim length As Integer

 extractRowsByTag = ""

 startPos = InStr(1, aSourceText, aStartTag, vbBinaryCompare)

 If (startPos > 0) Then

 startPos = startPos + Len(aStartTag)

 endPos = InStr(startPos, aSourceText, anEndTag, vbTextCompare)

 If (endPos > 0) Then

 endPos = endPos

 Else

 endPos = Len(aSourceText)

 End If

 length = endPos - startPos

 If (length < 0) Then length = 0

 extractRowsByTag = Trim(Mid(aSourceText, startPos, length))

 End If

End Function

Function getColorCode(ByRef aComment As Comment, aStart As Integer, anEnd As Integer) As String

 Dim aWorkRange As Range

 Set aWorkRange = aComment.Range.Duplicate

 aWorkRange.Collapse wdCollapseStart

 aWorkRange.Move wdCharacter, aStart

 aWorkRange.MoveEnd wdCharacter, anEnd - aStart

 If (aWorkRange.HighlightColorIndex = wdNoHighlight) Then

 getColorCode = ""

 'aComment.Done = False

 ElseIf (aWorkRange.HighlightColorIndex = wdBrightGreen) Then

 getColorCode = "Done"

 'aComment.Done = True

 ElseIf (aWorkRange.HighlightColorIndex = wdGray25) Then

 getColorCode = "NoChange"

 'aComment.Done = True

 ElseIf (aWorkRange.HighlightColorIndex = wdRed) Then

 getColorCode = "ToDisc"

 'aComment.Done = False

 ElseIf (aWorkRange.HighlightColorIndex = wdYellow) Then

 getColorCode = "ToDo"

 'aComment.Done = False

 Else

 getColorCode = aWorkRange.HighlightColorIndex

 End If

End Function

Sub switchToDraftViewWithCommentPane()

 If (Not ActiveWindow.View.Type = wdNormalView) Then

 ' Close any review- or comment-pane if open:

 ActiveWindow.View.SplitSpecial = wdPaneNone

 ' Then switch to normal view (draft) since that is the only safe way to

 ' prevent the ReviewPane from opening later

 ActiveWindow.View.Type = wdNormalView

 End If

 ' If the comment pane is currently not visible...

 If (WordBasic.ViewAnnotations = 0) Then

 ' ... enable it.

 WordBasic.ViewAnnotations 1

 Else

 WordBasic.ViewAnnotations 0

 End If

End Sub

Sub showCurrentHeadingName()

 MsgBox "Heading: " & getCurrentHeading(Selection.Range), vbOKOnly

End Sub

Function getCurrentHeading(aRange As Range) As String

 getCurrentHeading = ""

 Dim currentParagraph As Range

 Set currentParagraph = aRange.Duplicate

 currentParagraph.Expand Unit:=wdParagraph

 Do While (Not isHeading(currentParagraph))

 If (currentParagraph.Move(wdParagraph, -2) = 0) Then

 ' the range did not move. stop here to avoid deadlock

 Exit Function

 End If

 currentParagraph.Expand Unit:=wdParagraph

 Loop

 If (isHeading(currentParagraph)) Then

 getCurrentHeading = currentParagraph.text

 End If

End Function

Function isHeading(aRange As Range) As Boolean

 isHeading = False

 If (Not aRange.Style Is Nothing) Then

 If (InStr(1, aRange.Style.NameLocal, "Heading") = 1) Then

 isHeading = True

 End If

 End If

End Function

Sub RILAddDedicatedCommentSection()

'

' To be executed while in a RIL comment. It adds a new heading (H2) to the end of this document

' with the RIL number as text. It adds also a hyperlink to that heading into the

' currently chosen comment.

'

 Dim currentComment As Comment

 Dim newHeadingRange As Range

 Dim hyperlinkRange As Range

 Dim rilTag As String

 If (Not Selection.Information(wdInCommentPane)) Then

 MsgBox "You must select a comment in the comment pane to add an in-line review section", vbOKOnly, "Error"

 Exit Sub

 End If

 Set currentComment = Selection.Comments(1)

 ' Get and check RIL tag

 rilTag = CleanText(extractRowsByTag(currentComment.Range.text, "[RIL]: ", "["))

 If (rilTag = "") Then

 MsgBox "This is apparently not a RIL comment, i.e., it does not contain a valid RIL ID."

 Exit Sub

 End If

 If (InStr(1, rilTag, "xyz") > 0) Then

 MsgBox "Assign a valid RIL number first. Otherwise, the hyperlinking will fail later"

 Exit Sub

 End If

 ' Go from comment to actual document and jump to the end.

 ActiveDocument.Select

 Selection.Collapse wdCollapseEnd

 ActiveDocument.TrackRevisions = False

 ' Insert a new section of style "H2", fill the heading row and create a bookmark

 Selection.TypeParagraph

 Selection.Style = ActiveDocument.Styles("Heading 2")

 Selection.TypeText text:=rilTag

 Set hyperlinkRange = Selection.Range

 hyperlinkRange.Expand Unit:=wdParagraph

 Selection.TypeParagraph

 ' Add a bookmark to this heading

 With ActiveDocument.Bookmarks

 .Add Range:=hyperlinkRange, Name:="\_" & rilTag

 .DefaultSorting = wdSortByName

 End With

 ' Go back to the comment

 currentComment.Range.Select

 Selection.Collapse wdCollapseEnd

 Selection.TypeParagraph

 Selection.TypeText text:="["

 ActiveDocument.Hyperlinks.Add \_

 Anchor:=Selection.Range, \_

 Address:="", \_

 SubAddress:="\_" & rilTag, \_

 ScreenTip:="", \_

 TextToDisplay:="DedicatedSection"

 Selection.TypeText text:="]"

End Sub

Function CleanText(strInput As String) As String

 Dim i As Long, strTemp As String

 strInput = Trim(strInput)

 Do Until (Asc(Left(strInput, 1)) > 65 And Asc(Left(strInput, 1)) < 90) Or \_

 (Asc(Left(strInput, 1)) > 97 And Asc(Left(strInput, 1)) < 122) Or Len(strInput) = 0

 Select Case Asc(Left(strInput, 1))

 Case 65 To 90, 97 To 122

 Case Else: strInput = Mid(strInput, 2)

 End Select

 Loop

 strTemp = Left(strInput, 1)

 For i = 2 To Len(strInput)

 Select Case Asc(Mid(strInput, i, 1))

 Case 65 To 90, 97 To 122, 95, 48 To 57

 strTemp = strTemp & Mid(strInput, i, 1)

 End Select

 Next

ExitF:

 CleanText = strTemp

End Function

Function beautifyExcelWorkbook(aWorksheet As Object)

 With aWorksheet

 ' column width

 .Columns("A:G").EntireColumn.AutoFit

 .Columns("H:H").ColumnWidth = 35

 .Columns("I:I").ColumnWidth = 35

 .Columns("J:J").ColumnWidth = 25

 .Columns("K:K").EntireColumn.AutoFit

 ' text wrapping

 .Columns("H:J").WrapText = True

 .Columns("A:K").VerticalAlignment = xlTop

 .Columns("C:F").HorizontalAlignment = xlCenter

 .Rows("1:1").Font.Bold = True

 .Rows("1:1").AutoFilter

 End With

End Function

Function correctSettings()

 ' Disable automatic detection of language (reduces the risk of doing random

 ' language changes to a document)

 Application.CheckLanguage = False

 ' Disable tracked changes

 ActiveDocument.TrackRevisions = False

 ' Make sure that Word uses the user information as entered

 ' explicitly in word. Otherwise, all user names in tracked changes

 ' may screw up!

 Application.Options.UseLocalUserInfo = True

End Function

Sub RemoveBookmarksFromComments()

' It happens that Bookmarks end up in such comments. Word does not complain about this while working with

' and saving the document. But when opening it later, the document may be corrupted.

' This macro removes all such bookmarks in Comments.

 Dim oneComment As Comment

 Dim oneBookmark As Bookmark

 Dim nrofBookmarksInComments As Integer

 nrofBookmarksInComments = 0

 Dim nrofBookmarksInScope As Integer

 nrofBookmarksInScope = 0

 Dim commentsWithBoomarks As String

 commentsWithBoomarks = ""

 Dim commentsWithBookmarkInScope As String

 commentsWithBookmarkInScope = ""

 With ActiveDocument.Bookmarks

 .DefaultSorting = wdSortByName

 .ShowHidden = True

 End With

 For Each oneComment In ActiveDocument.Comments

 If (oneComment.Range.Bookmarks.Count > 0) Then

 For Each oneBookmark In oneComment.Range.Bookmarks

 oneBookmark.Delete

 nrofBookmarksInComments = nrofBookmarksInComments + 1

 Next

 commentsWithBoomarks = commentsWithBoomarks & "'" & Mid(oneComment.Range.text, 1, min(40, Len(oneComment.Range.text))) & "'" & vbNewLine

 End If

 ' Search for bookmarks in the scope of the comment (where the comment is linked into the document)

' If (oneComment.Scope.Bookmarks.Count > 0) Then

' For Each oneBookmark In oneComment.Scope.Bookmarks

' oneBookmark.Delete

' nrofBookmarksInScope = nrofBookmarksInScope + 1

' Next

' commentsWithBookmarkInScope = commentsWithBookmarkInScope & "'" & Mid(oneComment.Range.text, 1, min(40, Len(oneComment.Range.text))) & "'" & vbNewLine

' End If

 Next

If (nrofBookmarksInComments > 0) then

 InputBox Prompt:="Comments with bookmarks may leed to data loss and were hence removed:", Title:="Deleted " & nrofBookmarksInComments & " bookmarks from comments.", Default:=commentsWithBoomarks

End if

' If (nrofBookmarksInScope > 0) then

' InputBox Prompt:="Comments in the scope of bookmarks may lead to data loss and were hence removed:", Title:="Deleted " & nrofBookmarksInScope & " bookmarks from comments' scope.", Default:=commentsWithBookmarkInScope

' End if

End Sub