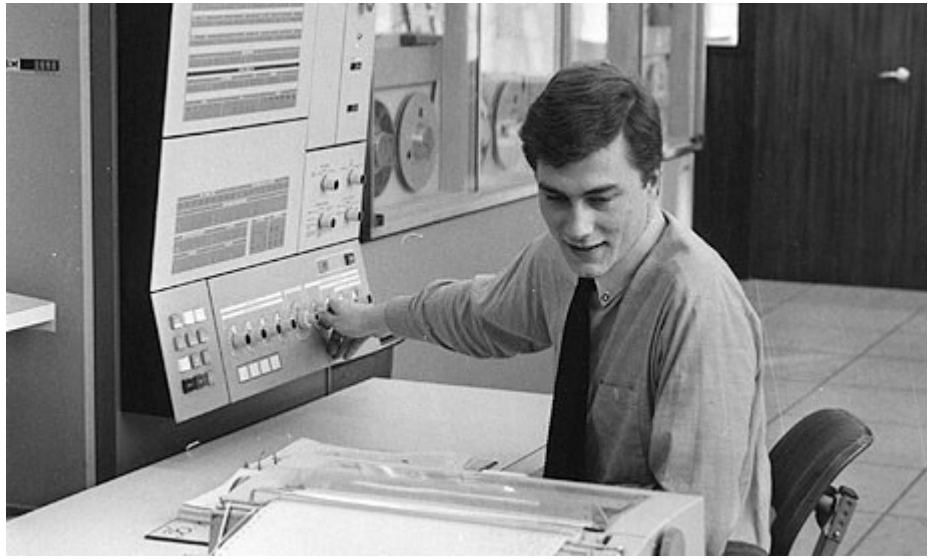
Use cases and examination of XML technologies to process MS Word documents in a corporate environment

Colin Mackenzie
colin@mackenziesolutions.co.uk
XML Consultant

Why develop this solution?



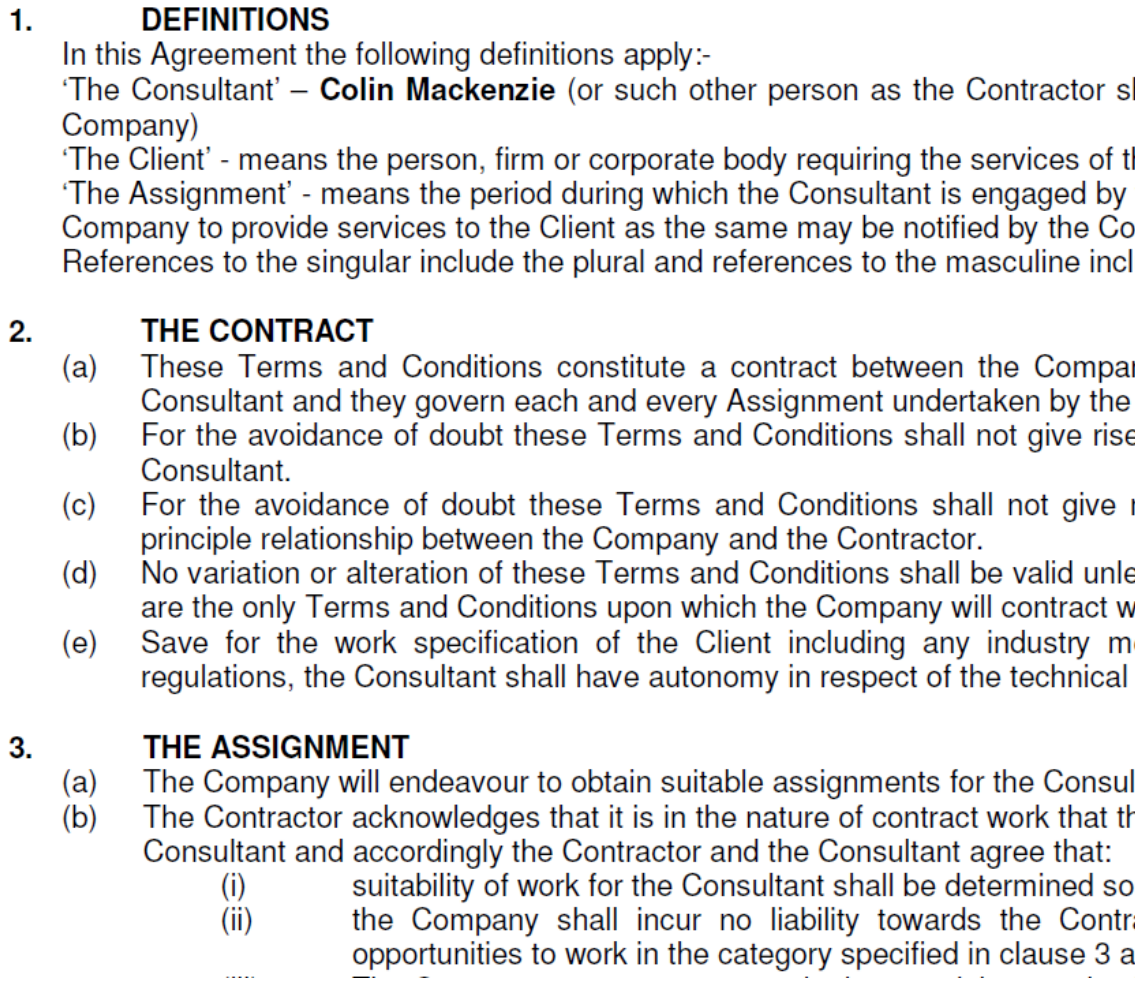
- Learn in a hands-on way
- Many XML developers not using XSLT3 (some not properly utilising XSLT2)
- Used Xproc 1.0 +CX but limited subset of features and non-complex requirements
- Needed a project to increase my skills that I can then transfer to customers

Why choose Word documents?

- XML's popularity now focussed on documents
- MS Word used by most major corporate users of documents
- MS Word uses OOXML
- Processing Word often requires complex development (unpacking, no nested structure etc.)

Use case – quality and consistency of styles

- Content
  - Are all required sections present and correctly named?
- Styling
  - Latest branding
  - Professional looking result
  - Consistent through-out document
  - Numbering and referencing affects legal meaning



1. **DEFINITIONS**

In this Agreement the following definitions apply:-
'The Consultant' – **Colin Mackenzie** (or such other person as the Contractor sl
Company)
'The Client' - means the person, firm or corporate body requiring the services of tl
'The Assignment' - means the period during which the Consultant is engaged by
Company to provide services to the Client as the same may be notified by the Co
References to the singular include the plural and references to the masculine incl

2. **THE CONTRACT**

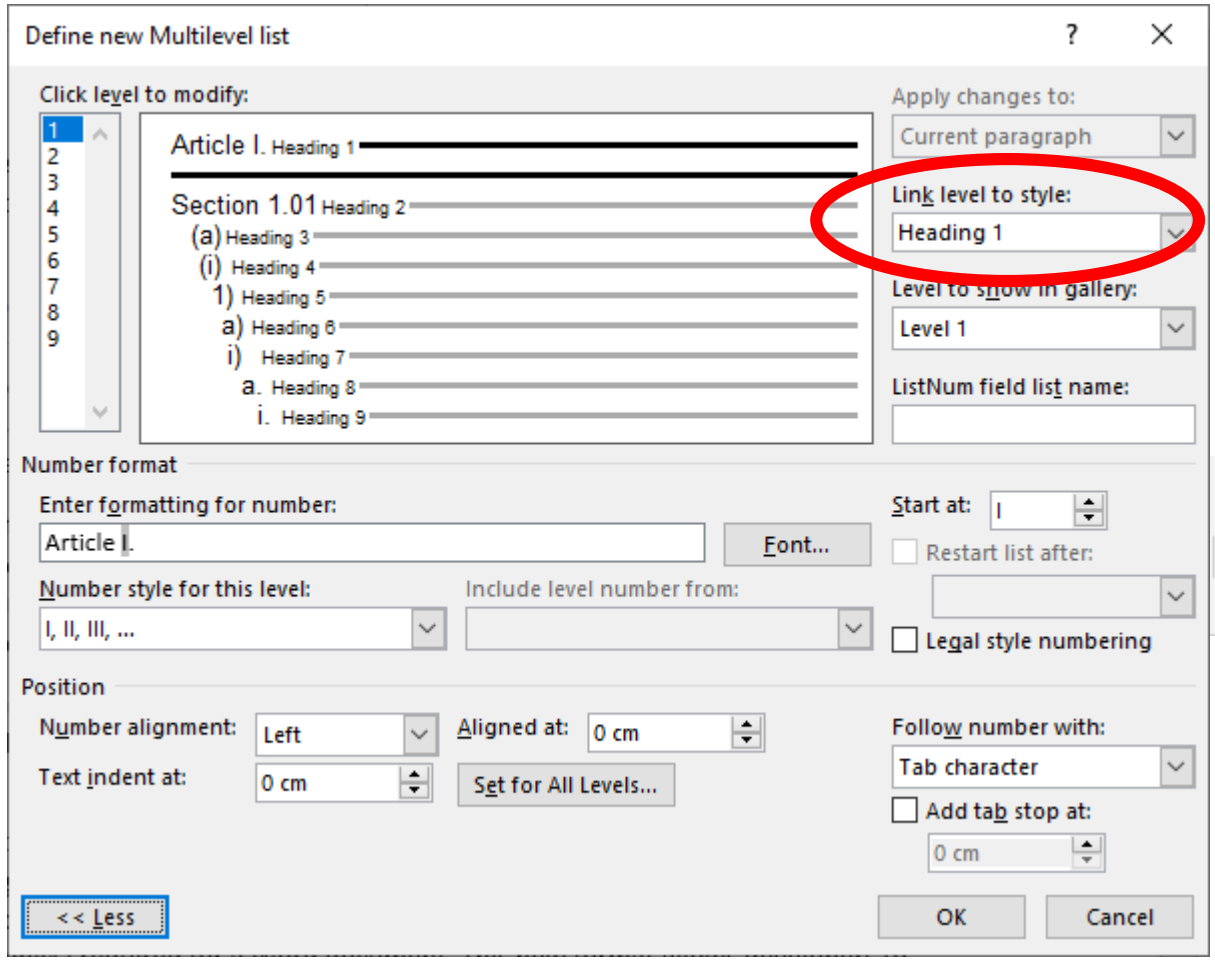
(a) These Terms and Conditions constitute a contract between the Compar
Consultant and they govern each and every Assignment undertaken by the
(b) For the avoidance of doubt these Terms and Conditions shall not give rise
Consultant.
(c) For the avoidance of doubt these Terms and Conditions shall not give r
principle relationship between the Company and the Contractor.
(d) No variation or alteration of these Terms and Conditions shall be valid unle
are the only Terms and Conditions upon which the Company will contract w
(e) Save for the work specification of the Client including any industry m
regulations, the Consultant shall have autonomy in respect of the technical

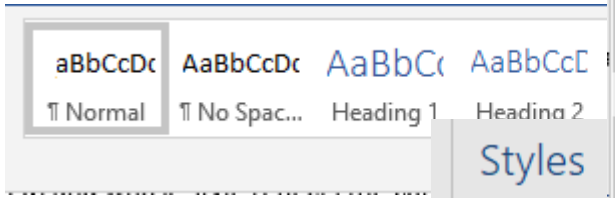
3. **THE ASSIGNMENT**

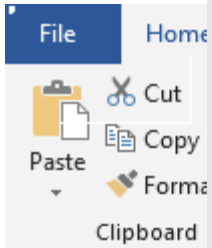
(a) The Company will endeavour to obtain suitable assignments for the Consul
(b) The Contractor acknowledges that it is in the nature of contract work that th
Consultant and accordingly the Contractor and the Consultant agree that:
(i) suitability of work for the Consultant shall be determined so
(ii) the Company shall incur no liability towards the Contr
opportunities to work in the category specified in clause 3 a

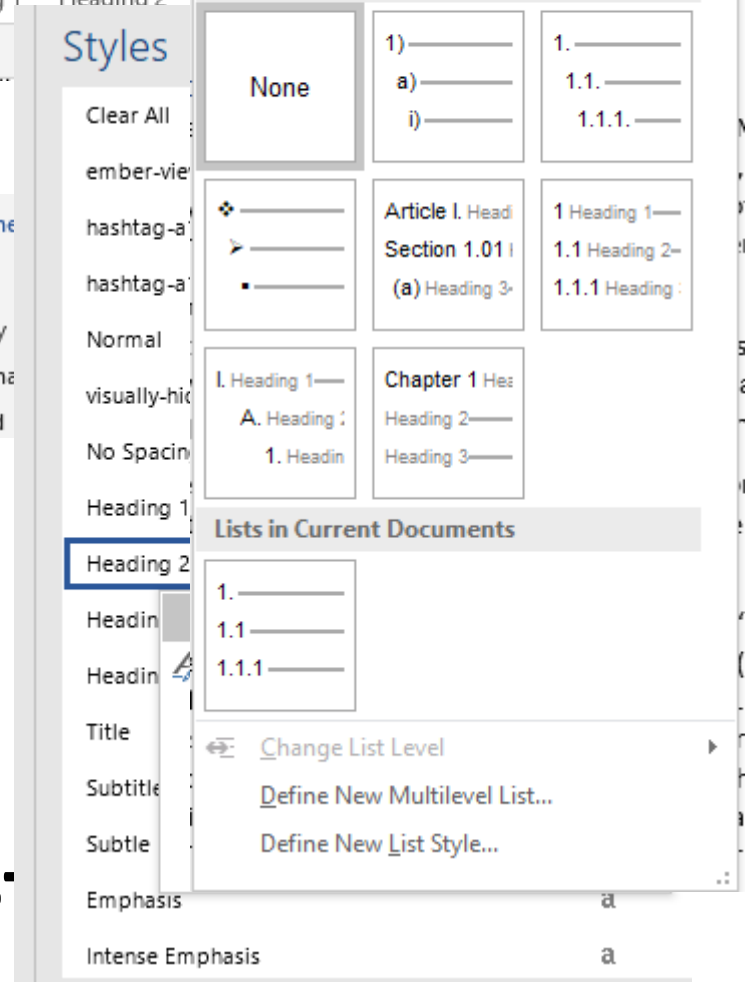
Why do things go wrong with Styles?

- Lack of training



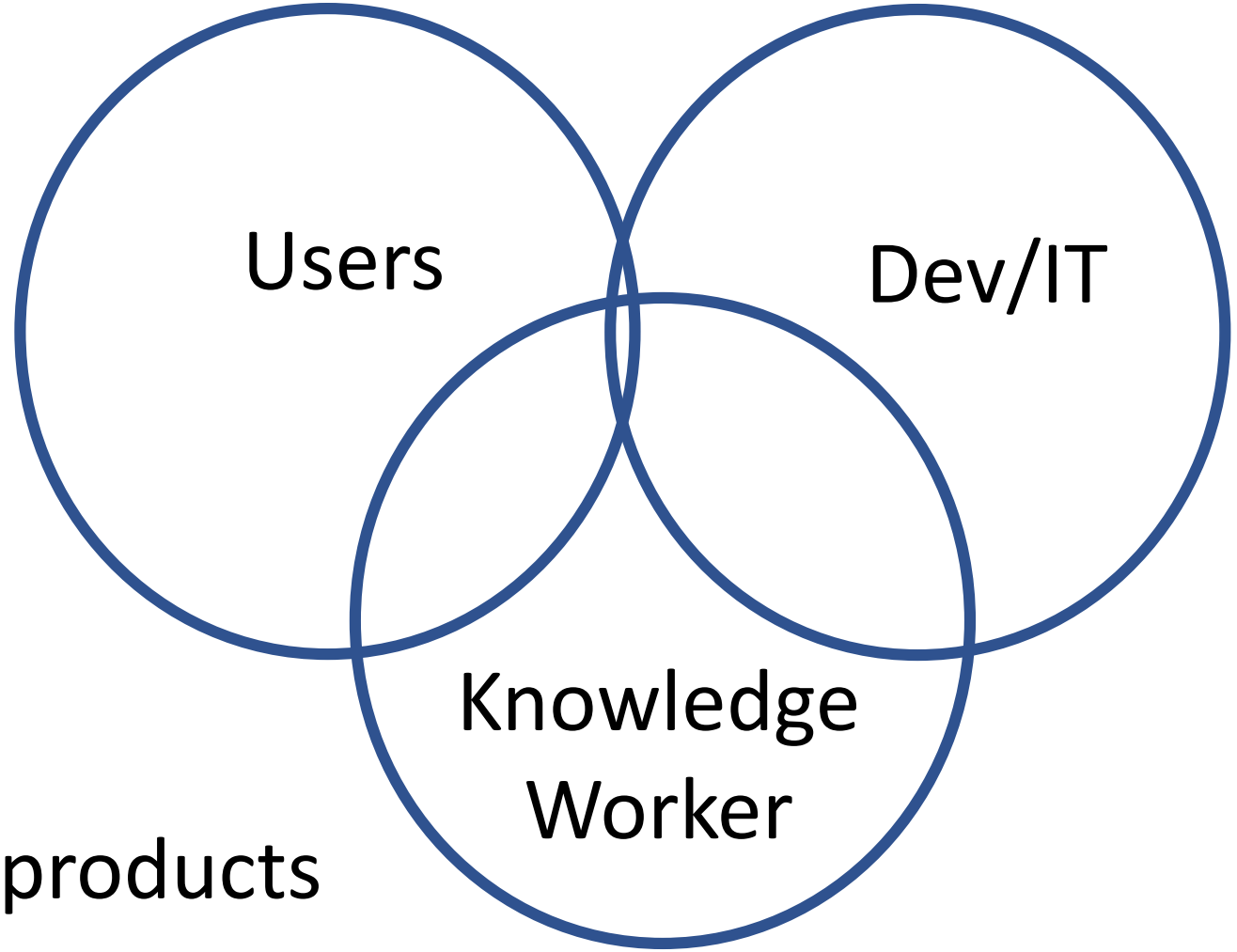






CTRL-

Typical solutions



- Custom templates
- Macros, VB, ribbons
- Training
- Commercial add-ins and products

- Tend to fail over time

So what about a standards-based solution?

- Allow knowledge workers to manage styles in the template
- Leave Word UI as out of the box
- Provide suggestions and feedback to users in a language they can understand
- Define the rules for style and content clearly

Word XML workflow

Word creation,
Review and correction.
Content + styles

Publish

Bad styling leads to invalid or semantically incorrect XML
Lack of validation leads to missing content

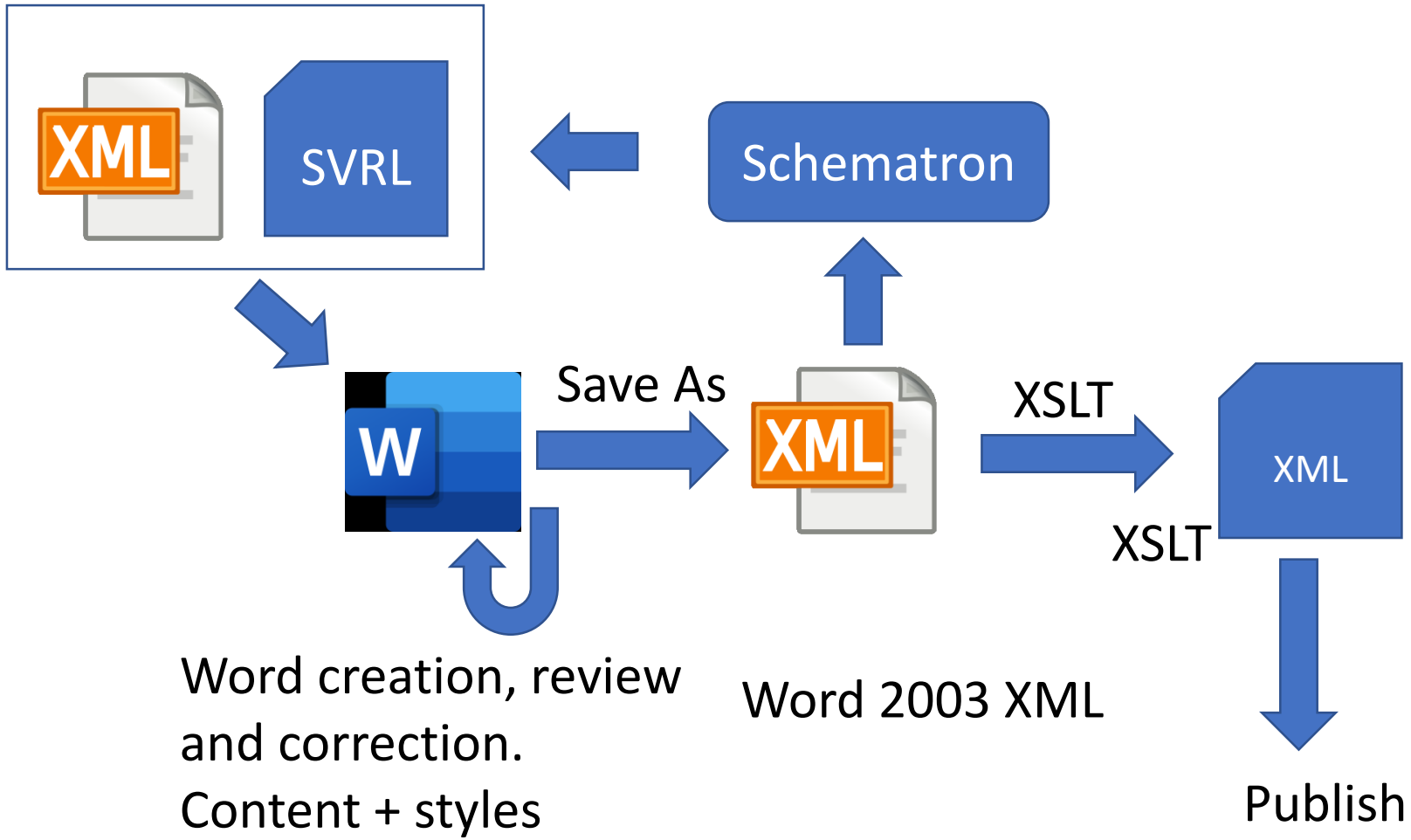
Some XML content

- Word Tables -> HTML/CALS tables
- Footnotes, links,
  graphic references etc. -> XML mark-up
- Flat Headings (style) -> nested XML structure
- Flat Lists (style) -> nested lists
- Other paras -> Semantic XML elements

Semantic conversion

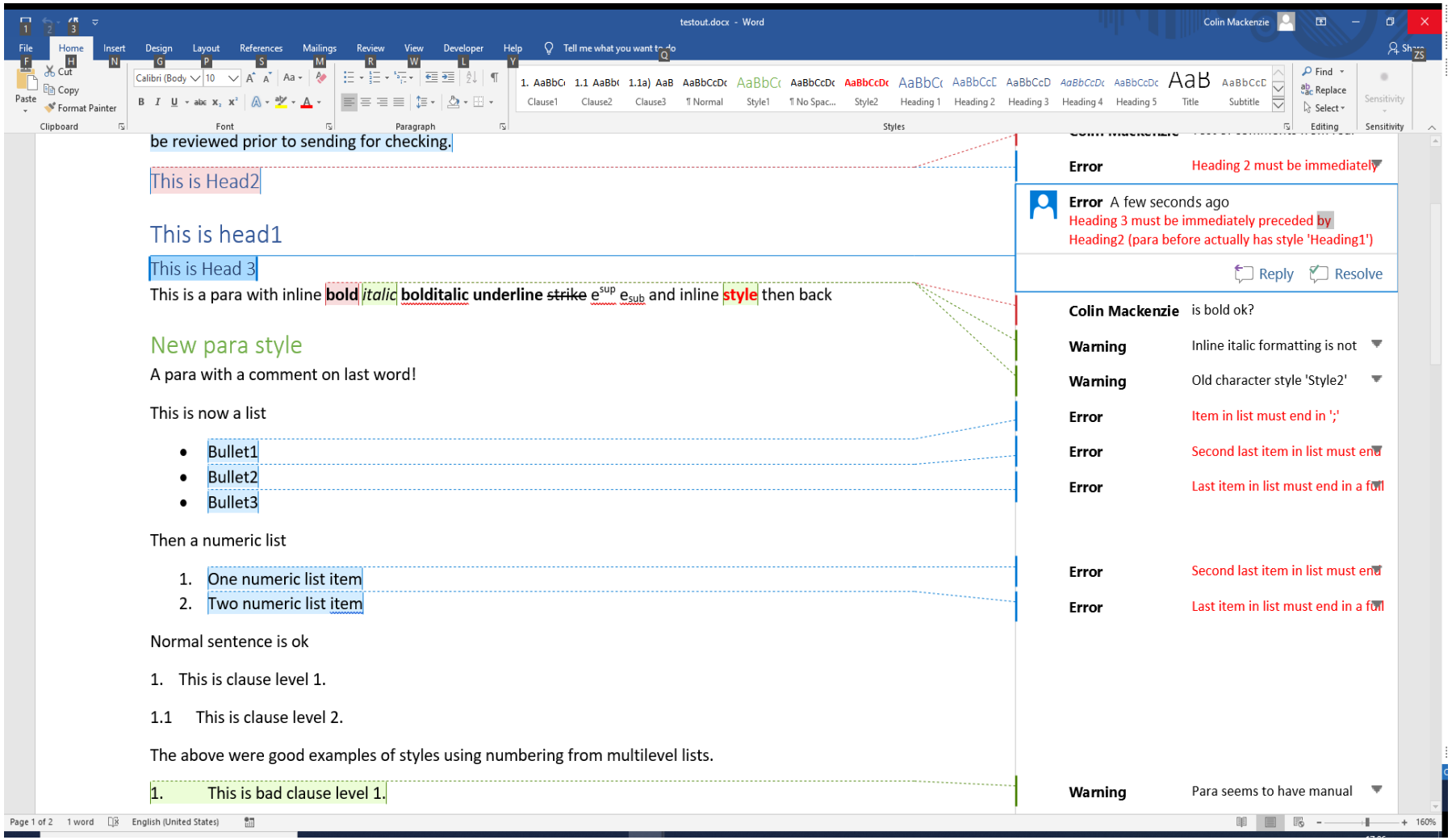
- Para in this style -> that semantic element
  - Group content between this para and another para in some other style
- Para in this style -> that metadata element
- Para with this auto numbering scheme -> that semantic nested element
- Drop fixed text
- Use text as clue to conversion etc.

Word XML + Schematron workflow



See 2017 “The application of Schematron schemas to word-processing documents” by Andrew Sales (Andrew Sales Digital Publishing Limited)

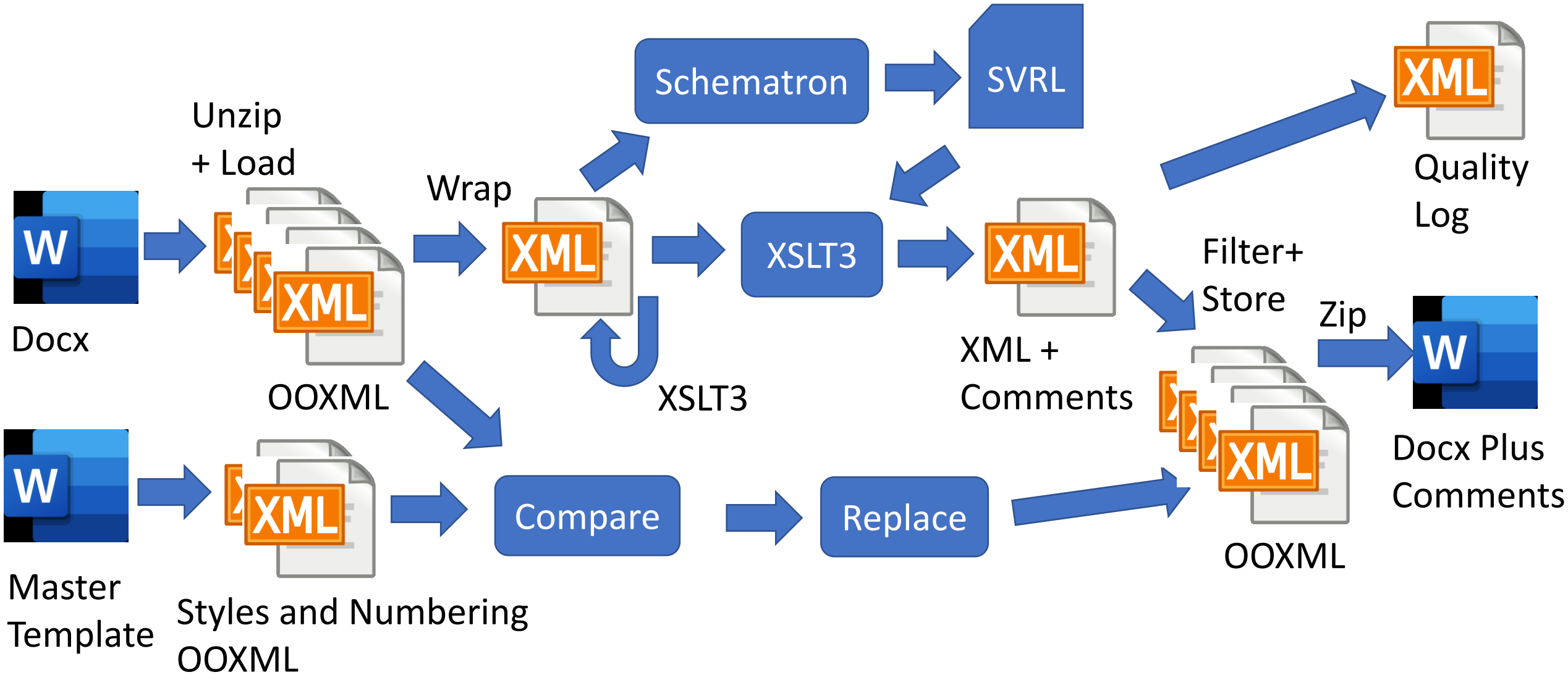
DEMO – Validation



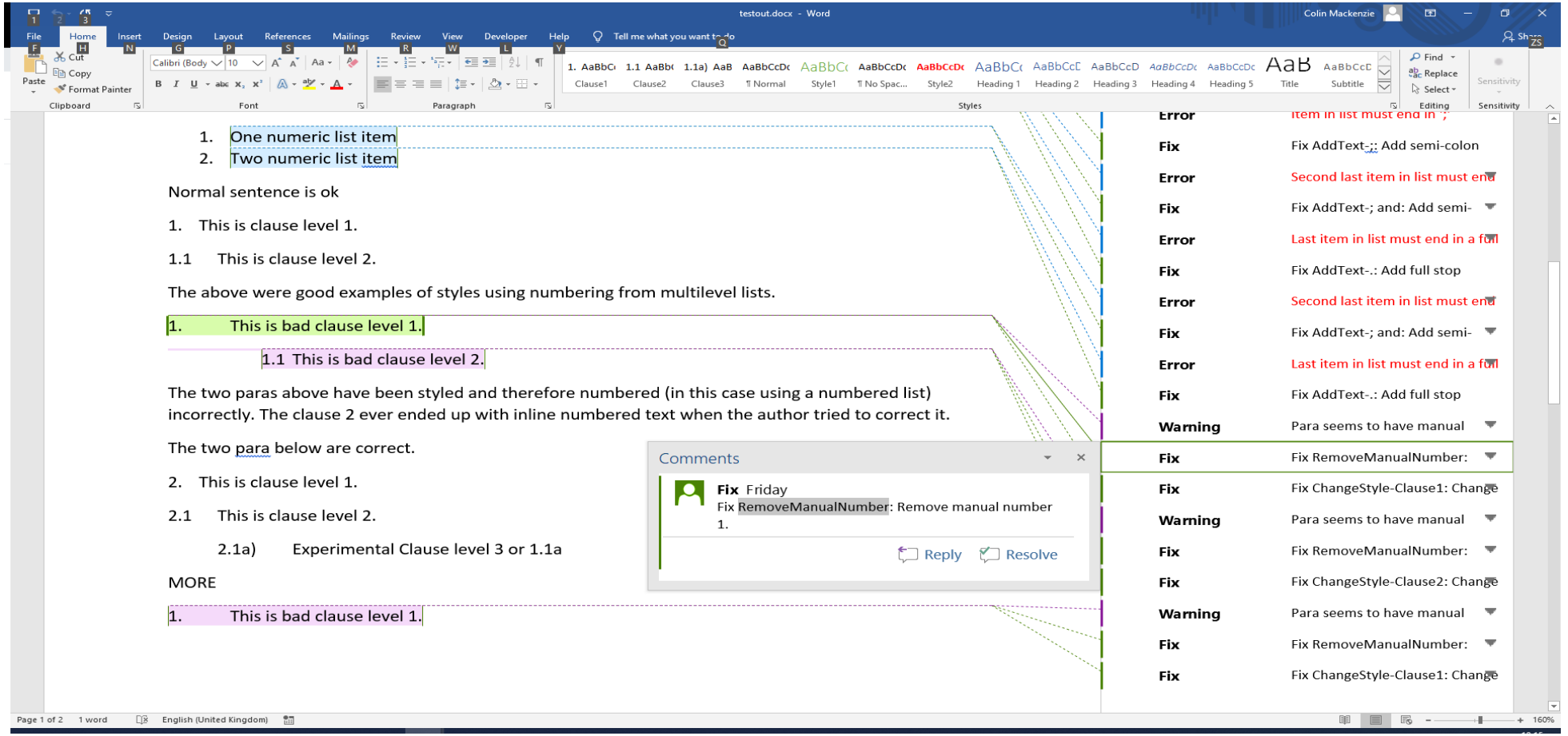
A sample rule

```
<sch:pattern>
    <sch:rule context="w:p">
    <sch:let name="vNumStr" value="ms:getManualNumber(.)"/>
    <sch:let name="vSuggestedStyleNames"
value="ms:getSuggestedStyleNames(.,$vNumStr)"/>
    <sch:report test="$vNumStr" id="ManualNumber1"
role="warning">Para seems to have manual number '<sch:value-of
select="$vNumStr"/>': consider replacing using styles <sch:value-
of select="$vSuggestedStyleNames"/></sch:report>
    </sch:rule>
</sch:pattern>
```

XProc Implementation



DEMO – Fixes following validation



A sample rule with a fix

```
<xsl:for-each select="$vSuggestedStyleNames">
   <cmqf:fix id="ChangeStyle-{.}">
      <cmqf:description>
         <cmqf:title>Change style to <xsl:value-of
select="."/></cmqf:title>
      </cmqf:description>
   </cmqf:fix>
</xsl:for-each>
```

ng

```
      </xsl:for-each>
   </cmqf:fixes>
  </sch:report>
 </sch:rule>
</sch:pattern>
```

Lessons learnt

- Saxon PE/EE needed for xsl:evaluate
- Functions can only be successfully called dynamically if marked as visibility=“public” (although function-available returns true no matter how it is set
- Named parameters (as opposed to XPaths) to dynamically called functions need to be specified

```
<xsl:evaluate xpath="my:function(., xpath , $varname)" context= "." >
    <xsl:with-param name= "varname" select="$varname" />
</xsl:evaluate>
```

- Ensure that injection attacks are avoided
- XProc annoyances and debugging – XProc3 please!
- OOXML annoyances

Conclusion

- There is no technical reason why tools like XProc, XSLT3 and Schematron cannot be used to solve corporate Word use cases
- Limitations will be around willingness of technical staff to learn and use the technologies
- While toolsets and libraries of functions like the ones I prototyped can reduce the technical burden, this may still be a challenge

Old dogs CAN do new tricks!

```
<xsl:sequence
        select="let $f := function-lookup(xs:QName($vFinalFunctionName),$vCountArgs)
    return if (exists($f)) then $f($pOriginalElement, ., $vFunctionArgs)
                                    else ()"/>
```

Thanks!

Any questions?

Colin Mackenzie

colin@mackenziesolutions.co.uk

XML Consultant