

Beyond valid and invalid: implementing complex validation in XProc

Andrew Sales

Chief Content Architect
Bloomsbury Publishing Group plc

Balisage, 5th August 2026

Introduction

- Today's topic
- Part technical implementation, part (incomplete) case study...
- Roll-out is in progress

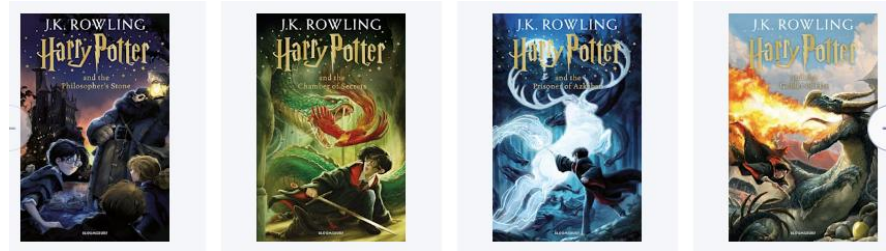


About us

Our company & our team

Bloomsbury Publishing

- Established 1986
- Two main divisions



- Our team works mainly with Acad. & Prof., specifically digital (BDR)



The Content Architecture (CA) team

- XML-first workflow
- Content capture outsourced
- In-house schemas & docs
- Content management & delivery
- Pre-publication QA



The Content Architecture (CA) team

- XML-first workflow
- Content capture outsourced
- In-house schemas & docs
- Content management & delivery
- Pre-publication QA

“SPEC

&

CHECK”

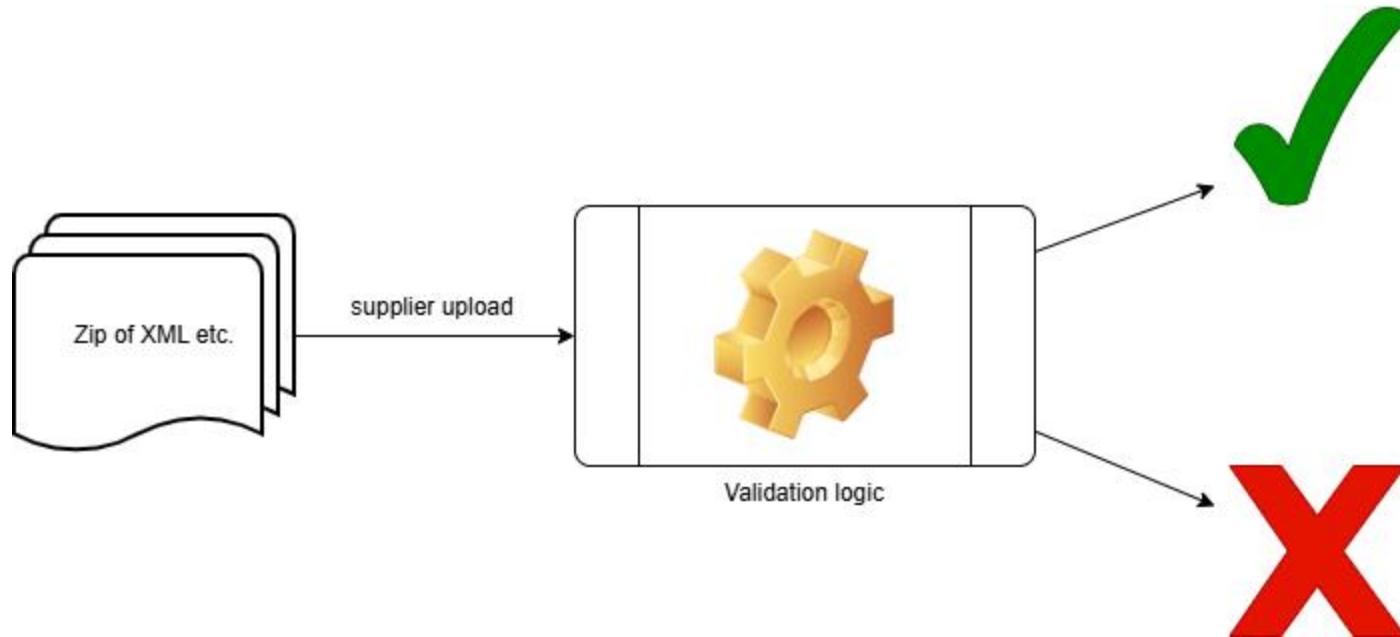


Digital publishing at Bloomsbury

- Began c.2013
- Schematron embraced quickly
- QA team for manual/semi-automated checks
- Gateway for receiving deliveries = automated validation



Validation portal, 1.0



Validation portal 2.0

Enter XProc

Scale up and skill up!

- Growth in output since 2016
- Automation is key
- Team objective to skill up too



XProc 3.x?

- Yes to XProc; No to XProc 1.0
- Maturity of 3.0 standard
- Two solid implementations
- Business case made easier



Design choices (1)

- Version 2.0 functionally equivalent, at base
- Web service executing a pipeline
- XProc selected
- We maintain the pipeline
- Fail-early approach



Design choices (2)

- Track supplier performance
- Reports generated all XML
- Stored in native XML DB (BaseX)



MorganaXProc-III

- Versus XML Calabash
- `p:os-exec` step
- Schematron “skeleton” support
- Performance
- Commercial licensing model
- Calabash still a useful reference!



System architecture

BLOOMSBURY 

Andrew Sales supplier Sign out

NAVIGATION

- Dashboard
- Upload Content

Upload Content

XML Validations

The final package, including XML, chunk PDFs, and image files, should be uploaded as a zip file.

[+ Upload Files](#)

BLOOMSBURY 

Andrew Sales supplier Sign out

NAVIGATION

- Dashboard
- Upload Content

Submitted by me

Submitted By	File Name	Submitted On	Status	Report
No records found.				

BLOOMSBURY 

Andrew Sales Sign out

NAVIGATION

- Dashboard
- User Management
- Upload Content
- DocBook documentation

Documentation - DocBook for Bloomsbury

bloomsbury-mods.rnc (21 KB) 29 July, 2025	→	publishers-51cr.rnc (203.6 KB) 19 July, 2024	→
docbook-mods.sch (750.2 KB) 8 August, 2025	→	Bloomsbury DocBook XML Data Specification v3.1.0.pdf (10.8 MB) 8 August, 2025	→

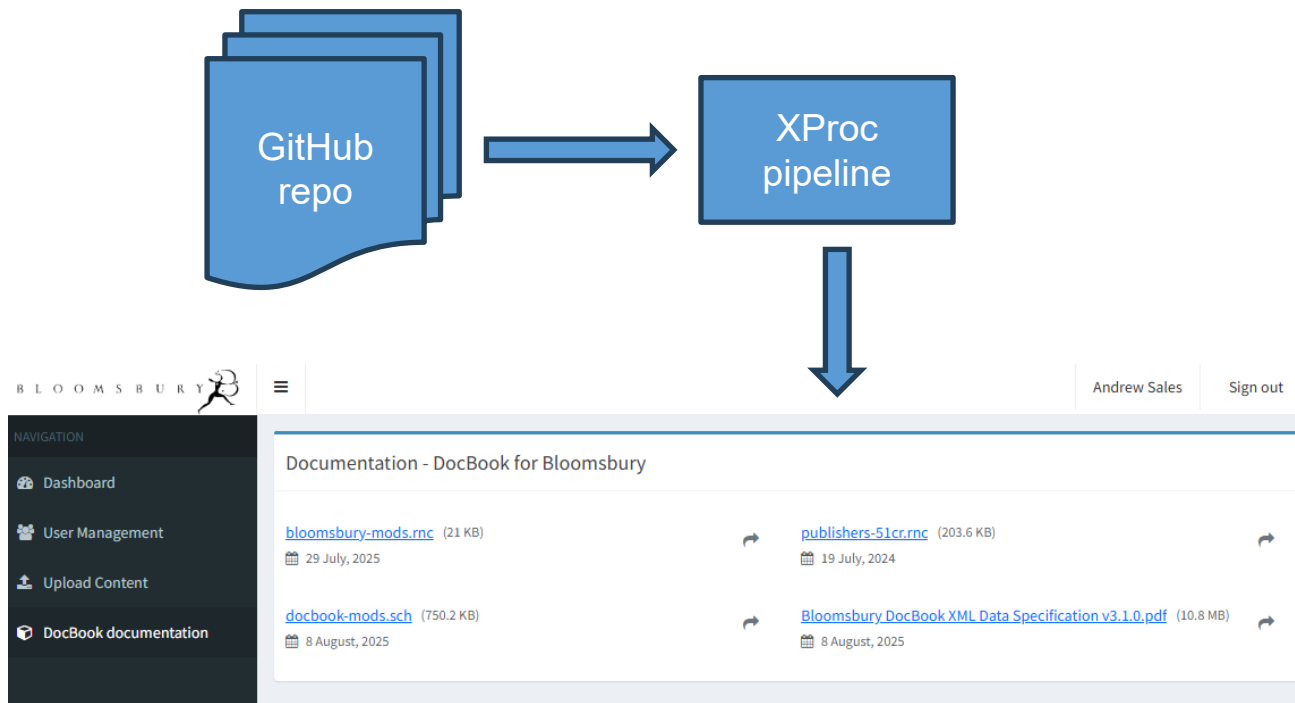


Simplified reports

```
<report
  uri="9781350002876_txt_xml.zip"
  timestamp="2026-02-13T15:39:05.4687649Z" warnings="208" distinctIds="41"
  schematronVersion="[schematron-version-info] schematronVersion = 11"
  workTitle="[title-value] Book title = &#34;Shakespeare and Fun&#34;"
  valid="false">
  <warning id="bib-title-end-quote" count="27">
    <svrl:successful-report test="ends-with(., '') or ends-with(., '"')"
      id="bib-title-end-quote" role="warning"
      location="/book[1].../title[1]">
      <svrl:text>[bib-title-end-quote] title in biblio item ends with
single or double quote</svrl:text>
    </svrl:successful-report>
  </warning>
  [...]
```



Documaintenance



Validation in XProc

Examples

Example: superfluous files

```
<p:variable name='non-primary-xml'  
select="/c:directory/c:file[ends-with(@name, '.xml') and @name ne $primary-xml]"/>  
  
<!-- resolve XIncludes: we explicitly fix up xml:base, so that we can check for  
any extraneous XML -->  
<p:xinclude fixup-xml-base="true">  
  <p:with-input><p:document href="{resolve-uri($primary-xml,  
/c:directory/@xml:base)}"/></p:with-input>  
</p:xinclude>  
  
<!-- all xml:base attributes present after fixup -->  
<p:variable name="xml-base-fixup" select="//@xml:base"/>  
  
<!-- superfluous XML: extracted files not appearing as @xml:base values after  
XInclude resolution -->  
<p:variable name="superfluous-xml"  
select="$non-primary-xml[not(base-uri(.) = $xml-base-fixup)]"/>  
  
<!-- check for superfluous XML in the zip -->  
<p:if test="exists($superfluous-xml)">  
  <p:error code="superfluous-xml">[...]</p:error>  
</p:if>
```



Schemalets

```
for $pi in $doc/processing-instruction(xml-model)
return parse-xml(
  concat('<xml-model ', $pi, '/>')
)
```



```
<xml-model href="my-schemalet.sch"
type="application/xml"
schematypens="http://purl.oclc.org/dsdl/schematron"/>
```



The grey area of the warning...

- An error is:
 - Fault in package structure
 - Schema validation error (RELAX NG or Schematron)
- Acceptance = 0 errors; n warnings
- Inspect and judge vs encode and “score”



Scoring the warnings

```
<xsl:template match="warning" mode="score">
  <xsl:copy>
    <xsl:copy-of select="@*" />
    <xsl:sequence select="(
      bmy:threshold-exceeded(@id, @count),
      bmy:proportion-exceeded(@id, @count,
$node-count-by-id?(@id))
    )" />
    <xsl:copy-of select="*" />
  </xsl:copy>
</xsl:template>
```



ONIX metadata

- Bibliographic metadata discrepancies
- Accuracy is highly desirable
- Store ONIX metadata and compare to content XML on receipt



Testing

- XSpec can now test XProc
- For now, we use BaseX's `%unit:test`

```
declare %unit:test function _:extraneous-media-fail()  
{  
  unit:assert(  
    _:run-pipeline('extraneous-  
media/fail/9781350996526_txt_xml.zip')/c:errors/c:error[@code  
eq 'extraneous-media']  
  )  
};
```



Extensibility

- Recent need to support a new schema variant
- Long, slow & brittle process under portal 1.0
- Just two minor pipeline changes with 2.0



Summary

- Free black-box → low-cost, low-maintenance service
- Validation logic brought in house: adaptive/responsive to quality issues
- Collate and interpret supplier performance data for first time



Thank you!

Any questions?

andrew@andrewsales.com

