The Pioneer’s Dilemma
Rethinking and Retooling
What Was Once State-of-the-Art

Bill Kasdorf
VP and Principal Consultant, Apex Content Solutions
Sometimes you need to let go of something you loved and thought couldn’t possibly be improved.
Q: How could a typewriter get any better than this?
A: When it isn’t a typewriter anymore.
Revolutionary. Revelatory. (But still beige.)
What was once the coolest thing starts to look . . . dated.
Anybody remember what this is?
No technology is an island. If it doesn’t work in the ecosystem, it doesn’t work.
Cambridge University Press has been in the vanguard for 500 years.
Cambridge University Press has been in the vanguard for 500 years.

Some recent examples:

SGML.
Cambridge University Press has been in the vanguard for 500 years.

Some recent examples:
- SGML.
- Copyediting with MS Word.
Cambridge University Press has been in the vanguard for 500 years.

Some recent examples:

SGML.

Copyediting with MS Word.

Automation with scripts.
Cambridge University Press has been in the vanguard for 500 years.

Some recent examples:

- SGML.
- Copyediting with MS Word.
- Automation with scripts.
- XML-based workflow.
Cambridge University Press has been in the vanguard for 500 years.

Some recent examples:

- SGML.
- Copyediting with MS Word.
- Automation with scripts.
- XML-based workflow.
- Schematron.
Cambridge University Press has been in the vanguard for 500 years.

Some recent examples:

- SGML.
- Copyediting with MS Word.
- Automation with scripts.
- XML-based workflow.
- Schematron.
- EPUB.

Ummm. . . .
How come our EPUBs are taking so long to get and having so many flaws?

Wasn’t our XML supposed to make EPUBs automatic?

There must be something wrong with our XML.
The Evolution of XML at CUP

Mid 90’s: SGML headers.
2001: Full text CUP SGML.
Mid-2000s: XML as CamML.
CamML begat CJML and CBML, both based on NLM.
CBML is updated frequently as needs are encountered.
2013: CJML becomes CUPJATS 1.0.

What to do about CBML?
The Problems with CBML

Has it gotten too complex?
Frequent updates (every 2–3 months) make it hard to manage.

Files in the repository are inconsistent.

Promised “automatic” EPUBs turn out to be anything but.

Have we over-automated?

Isn’t there a better option out there now?
Here comes XHTML to save the day!

Uh . . . not so fast.

Me last year
The Problems with CBML

Has it gotten too complex?
Frequent updates (every 2–3 months) make it hard to manage.

Promised “automatic” EPUBs turn out to be anything but.

Have we over-automated?
Isn’t there a better option out there now?

*The issues are not all due to the model.*
The Problems with CBML

Has it gotten too complex?

Frequent updates (every 2–3 months) make it hard to manage.
Files in the repository are inconsistent.
Promised “automatic” EPUBs turn out to be anything but.

The model is complex because the books are complex.

Have we over-automated?
Isn’t there a better option out there now?
The Problems with CBML

Has it gotten too complex?
Frequent updates (every 2–3 months) make it hard to manage.

There are ways to make a model adaptable but not disruptive.

Have we over-automated?
Isn’t there a better option out there now?
The Problems with CBML

Has it gotten too complex?
Frequent updates (every 2–3 months) make it hard to manage.

Files in the repository are inconsistent.

Remediation may be called for when the new model is set.

Have we over-automated?
Isn’t there a better option out there now?
The Problems with CBML

Has it gotten too complex?
Frequent updates (every 2–3 months) make it hard to manage.
Files in the repository are inconsistent.

Promised “automatic” EPUBs turn out to be anything but.

With new model, do EPUB 3 with fallbacks rather than “lowest common denominator.”
The Problems with CBML

Has it gotten too complex?
Frequent updates (every 2–3 months) make it hard to manage.
Files in the repository are inconsistent.
Promised “automatic” EPUBs turn out to be anything but.

Have we over-automated?

No! Don’t abandon a great workflow.
The Problems with CBML

Has it gotten too complex?

Frequent updates (every 2–3 months) make it hard to manage.

Files in the repository are inconsistent.

Promised “automatic” EPUBs turn out to be anything but.

Have we over-automated?

Isn’t there a better option out there now?

Yes. BITS 1.0.
Why is BITS Right for CUP?

It’s designed for scholarly/STM books and for publishers that use JATS.
Why is BITS Right for CUP?

It’s designed for scholarly/STM books and for publishers that use JATS.

Below chapter level markup = JATS.
Why is BITS Right for CUP?

It’s designed for scholarly/STM books and for publishers that use JATS.
Below chapter level markup = JATS.
It’s modular and built for adaptation; can keep CUP book semantics & modules.
Why is BITS Right for CUP?

It’s designed for scholarly/STM books and for publishers that use JATS.

Below chapter level markup = JATS.

It’s modular and built for adaptation; can keep CUP book semantics & modules.

It facilitates book/journal interoperability.
Why is BITS Right for CUP?

It’s designed for scholarly/STM books and for publishers that use JATS. Below chapter level markup = JATS.

It’s modular and built for adaptation; can keep CUP book semantics & modules.

It facilitates book/journal interoperability.

BITS structure is better suited to CUP’s automated processes.
Why is BITS Right for CUP?

It’s designed for scholarly/STM books and for publishers that use JATS.

Below chapter level markup = JATS.

It’s modular and built for adaptation; can keep CUP book semantics & modules.

It facilitates book/journal interoperability.

BITS structure is better suited to CUP’s automated processes.

CUP’s comps will be familiar with it.
Lessons Learned

The model that’s right for somebody else may not be right for you.

Moving to XHTML sounded best . . . until we looked at the whole ecosystem at CUP.
Then BITS was the clear choice.
Lessons Learned

The model that’s right for somebody else may not be right for you.

It’s not all about the model.

Most of the issues were not due to the model itself, but how it was used (or not) in various parts of the workflow.
Lessons Learned

The model that’s right for somebody else may not be right for you.
It’s not all about the model.

Parsing ain’t all it was cracked up to be.

The best QA/QC requires all three approaches:
1. Parsing (grammar-based validation)
2. Schematron (rules-based validation)
3. QA CSS (human inspection)
Lessons Learned

The model that’s right for somebody else may not be right for you.
It’s not all about the model.
Parsing ain’t all it was cracked up to be.

Certain things shouldn’t be left to vendors.

You need to “own” essential things:
Your specifications.
Your Word template.
Your adaptation of the model.
Lessons Learned

The “Style Name Boxes” in Word, essential when editing was on paper, caused problems for online editing; now there are better ways to convey the tagging.

Great features can become impediments.
Lessons Learned

The model that's right for somebody else may not be right for you. Validation ain't all it was cracked up to be. Certain things shouldn't be left to vendors. Great features can become impediments. Oversimplification can cause complications.

Leaving elements out of the Word style library made the work unreliable. “Lowest-common-denominator” EPUBs made good e-readers look like bad e-readers.
Thanks!

Bill Kasdorf

bkasdorf@apexcovantage.com
+1 734 904 6252
@BillKasdorf