A Modern Book Archive Geared for Custom Publishing

Liam Borgstrom

University of Pretoria, South Africa

Abstract. The world of book publishing has changed drastically with the adoption of XML technologies, by imposing regular structures on authors and allowing production departments to design, alter, and redistribute books in a fraction of the time it would have taken previously. However, as we move to make our content more machine readable, we need to consider how we apply metadata in a way that supports custom publishing solutions. The author proposes a new way of looking at XML schemas for archiving books, and suggests how this may be implemented with code examples. While an archive generally preserves content as it was produced, because publishers' archives are potential sources of revenue, they should be compiled in such a way that allows for calculated mining, supported by semantic mark-up describing not only the book function, but the intended uses.

Keywords. Custom Publishing, XML, Higher Education, XML Workflow, Contextual Publishing.

Introduction

So often when looking into digital innovation we find that the focus is on the big advances, such as artificial intelligence and big data. Attention is focused on how is this can be used to manage cars and cities, create smart recommendations for advertisers and assist us in our daily lives. But how do we apply this to books?

Yes, Siri and Google Assist can tell us about anything with a Wikipedia article, but what if we say, "Ok Google, I want to get into Deep Learning and Neural Networks, I'm in high school and have done some programming, and I think that this may be a potential career move. Can you give me an introduction to the field?"

Typing this is now gives you links to AI blogs, MIT and Coursera Courses, Newspaper articles, some adverts for data science, and eventually a 2208 book on page four of the results. However in Frederick Pohl's *Day of the Boomer Dukes* (1956) [1] the protagonist – Foraminifera 9-Hart Bailey's Beam – asks the Learning Lodge index computer to find some information for him based on certain criteria and, "Then there was a hiss and a crackle, and in the receiver of the desk a book appeared. I unzipped the case, took it out, and opened it to the pages marked on the attached reading tape."

This idea, that a book can be created to the needs of a specific reader is not a new one, and is being attempted by educational companies today venturing into the practice of **Custom Publishing**.

Custom publishing has its greatest potential in the higher education market, where books are increasingly being compiled from front and backlist content according to the requirements of a specific course.

However, in order to achieve this, we need more than a business practice; we need a new way of looking at our content. Brian O'Leary has addressed this in his context-first publishing philosophy. [2] However, I am not sure how many publishers are really looking into a practical way of implementing this. Publishing in XML with DocBook and custom schemas is certainly the first step, but how should we preparing our books so that the content can be intelligently repurposed based on its content rather by its formatting.

I find that the theme I am constantly repeating to my students is that digital content needs to be machine-readable. Programmers understand this. They understand that for any system to work, a precise vocabulary must be used in defining different classes, functions and variables, in this way the software can work according to its programmed logic. The software used in publishing is designed in this way. InDesign (still the book design staple) works best when character, paragraph, and object styles are used consistently. This allows for valid nesting and importing of content and for eventual compatibility with a publishing system designed to access content defined in XML.

While XML schemas are becoming more semantic, custom publishing solutions are still reliant on describing the book as part of a book. For Pohl's scene in the Learning Lodge to occur, an AI must be able to examine a piece of text and understand not only its semantic meaning, but the intent of the writing. This is not only a tool for training AI, but to assist in human searches as well. While search algorithms are constantly adapting, they must still respond to usage patterns and direct language. Full text searching is still our best method quickly discerning the content of a book. However, this is limited by our own language and individual means of expressing our objectives. For custom publishing to truly work, we need to be able to extract content based on its purpose and function, and be able to locate suitable content timeously from our backlist.

My proposal here is for an archiving system with the potential to catalogue content down to the paragraph level. Content should be semantically defined so that it is possible to make a search query such as, "explanations and exercises in geometry for students written in a conversational and accessible tone".

As publishers, we must strive to defend the unique aspects of our books. The profile of the author, or the marketing blurb is not enough, (especially in educational environments) and it is up to us as publishers to create textual content that engages the intended readership as much as possible. The query above as a search request could only realistically pull results from the marketing information or introduction. However, with granular enough metadata this query could extract the exercises themselves based on their intended audience.

Practically this comes down to rigorous metadata. However, mention metadata in the book world, and thoughts immediately fly to ONIX, MARC, Dublin-Core, XMP and so forth. However, these formats – valuable as they are – are primarily useful in distribution, discovery, and access of the complete volume. They assist the sales division, but how can we use them in production?

If we are to truly redefine publishing in the 21st century, we need to define content right down to the paragraph, so that as publishers we are truly able to create a content database of relatable, and chunkable material, and that can be thoughtfully collated and curated according to need, and possibly even *by machine*.

An XML Framework

While O'Leary's explanation of contextual publishing focusses on how content is inter-connected, and needs to be aware of how it is being consumed, I think what we can add to this is, "who is reading it?"

For this we need to start thinking practically about how we store our content. The best technology we have for this so far is XML, it's ubiquitous, scalable, and suitably rigorous.

If we consider how most modern publishers are working with XML schemas we will probably find that books are defined as such:

```
  <Book>
    <FrontMatter>
        <dc:identifier id="isbn-
id">urn:isbn:9782541387655</dc:identifier>
        <dc:creator id="author">John
Andrews</dc:creator>
        <meta refines="#author" property="role"
scheme="marc:relators">aut</meta>
        <dc:title>Book of Wonders</dc:title>
        ...
    </FrontMatter>
    <MainMatter>
        <Chapter>
            <Title>What is Wonder?</Title>
            <p class="first">When I first
began...</p>
        </Chapter>
        ...
    </MainMatter>
    <EndMatter>
        <Index>
         <Iheading>A</Iheading>
            <Iterm id="aardvark">Aardvark <a
href="#IT-aardvark"><IpageRef/></a></Iterm>
        </Index>
```

```
        ...
    </EndMatter>
</Book>
```

Fig. 1. A typical book schema

Fig. 1 is a simplified representation of the rich detail that is possible through XML-based publishing. It not only captures content according to its position and representation in the book and allows us to redistribute such content by simply assigning different styling to it. In this way custom publishing is realised by text which is instantly reflowable, and adapted to the end format. However what the artificial minds of the future may truly require is something more akin to this:

```
  <ComputerScience category="AI">
    <Text>
        <meta>
            <dc:creator id="author">John
Andrews</dc:creator>
            <metaSource
id="urn:isbn:9782541387655"/>
            <ReadingLevel lvl="secondary" rec="16"
rec="19" rec="18" rec="20" />
            <WritingLevel lvl="conversational" />
        </meta>

        <Content keywords="AI, Deep Learning,
Neural Networks" Intent="exploratory">
            <p>When I first began...</p>
        </Content>
      </Text>

</ComputerScience>
```

Fig. 2. A sample extract from a contextual archive

In this second example (Fig. 2) the book is not the source, rather the content is stored as part of the "Computer Science" subject pool. What's shown in Figure 2 is not a complete record, but its structure should indicate how a book can be **disaggregated**. The link to the original book is still preserved, as is information on the intended reader.

Authors would not write like this, they would write in a format similar to Fig. 1, as this is suitable to the familiar format of the book, and it still the format which we understand the best. However, after publication, the content could be stored as in Fig. 2. If publishers truly are curators of content and we want to approach a stage where books can truly be produced per customer, then content needs to be defined much more richly.

While there are great concerns for the integrity and feel of a complete work, when content is stored richly and independently from the book, the ability to locate content and reference content is greatly improved. Rather than trying to match a search query to the text itself, we can match more closely to the metadata. This has great implications for digital content offerings, where question and topic banks should be sourced from as wide a pool as possible.

When John Strange referred to content as being disaggregated [3] he was referring largely to how journals are now consumed per article rather than per volume. We are still in this mode of thought, and see an author's work as a complete unit. This is certainly valuable in scholarly publishing, where best use of custom publishing would be for collated and special editions. However, with educational and instructional content we must consider that the end-reader is much less homogenous (especially in terms of reading ability).

It is possible now to cater for this in an existing book. For example, an author may describe a concept in a variety of fashions, say at the level of a professional, and again at that of an amateur enthusiast. In the digital iteration of the book, rather than displaying the technical definition alongside a simple analogy, the book default could be at a high reading level, with lower explanation available as a hovering tool-tip, footnote, or possibly replace the content entirely based on reader preferences.

If we wish to add more definitions for readers of different levels, should we ask the author to write several extra, or should we mine our own backlist for suitable content? In a disaggregated archive definitions could be stored as shown in Fig. 3, where all definitions of a concept are stored in an index, with a traceable path to the original source.

```
  ... <definition lvl="phd" keywords="neural
network" sourceid="urn:isbn:9782541387655"> ...
</definition>
      <definition lvl="bachelor" keywords="neural
network" sourceid="urn:isbn:9782541587489"> ...
</definition>
      <definition lvl="secondary" keywords="neural
network" sourceid="urn:isbn:9782541347105"> ...
</definition>
      <definition lvl="enthusiast" keywords="neural
network" sourceid="urn:isbn:9782541387655"> ...
</definition>
```

Fig. 3. A sample representation of a definitional database

Reading level is only one of many possible attributes in this case. In reality, the defining categories for content chunks would need to include attributes such as: author, location, language, date, and associated material (if supplementary material is associated with it, such as definitional boxes and so forth).

In this way it is possible to tailor our content by using our existing resources, and eventually use it to provide a service akin to Pohl's ideal.

Implementation

What will always be daunting to publishers is how to manage with richly-defined content. Surely to add such details to every paragraph, box, definition, figure, or term is too daunting a task and we should rely on literate and interested readers?

However, working from the bottom up and creating such a platform from scratch to apply to front list material may not be the best solution. A far safer approach would be to work with the back-list and improve the content by working backwards and retroactively on works. Not that I am encouraging format conversion after the fact (so-called "XML-Out" [3]), rather that this is an archiving function, operating with the intent of making the publisher's archive not merely a historical record but an active tool for researchers and for future editions.

The suggested implementation would be as follows:

If a publisher were to implement the system in 2020, the first content to be archived would be that published in 2019. In this way there is no interference with the current edition. It is complete and commercially available.

The role of the archivist now is to accurately pull the metadata for the book from the DC and ONIX records. This is to ensure that any time the content is pulled, it is always pulled with its metadata inserted via a script in the system.

The archivist reading through the content is guided by the official published metadata, meaning that the internal attributes should be applied in accordance with the holistic descriptors. Defaults can be set for any of the attributes, and in the case of a book which deals with e.g. multiple themes, the archivist is able to apply the themes to the specific sub-divisions (chapters or deeper) where it occurs.

In this way, a publisher starts afresh by working on their current materials and preserving them for future usage.

Archiving should have two goals:

1) To accurately prepare the previous years' works for disaggregated usage
2) To increase the accessibility of the backlist for disaggregated usage.

This could ideally be done with two teams. To use the previous example, beginning in 2020, Team A work with content from 2019, then 2020, and 2021, always one year behind the front-list, while Team B would begin with 2018's publications, and then work backwards, 2017, 2016… and so forth. In this way the archive expands in two directions, increasing potential of the archive as time goes on.

The importance of using XML here is paramount, as this approach is aimed at the prosperity of publishing, and should another more suitable and standardized schema be developed, conversion via XSLT is still possible.

Above all, one should avoid the mistakes traps of legacy systems such as those which have plagued the banking sector. If one were to change systems in banking working in this bidirectional way, new clients would be on a new system, and old clients would be gradually transferred. In both instances XML (rather than COLTAN) makes an ideal storage solution as the data can be stored independently of the operating languages and therefore be easily appropriated for other tasks.

I draw this comparison as in all publishing sectors the danger is the same. Content is *trapped* inside PDFs as the source material is unavailable, trapped within PageMaker or in film requiring new editions to be remade each time. However, if we are able to

repurpose content separately from it formatting then as publishers we are free to keep our content alive and in circulation, whether in its original form or disaggregated. It is in the application of XML stored content that the service offering extends beyond marketing and distribution and into cultivation of new content.

For a publisher, an archive is not just their history, it is their active backlist, and it is in the best interest of readers and of the publisher itself to make sure that the backlist is kept accessible, available, and ready for commercial use.

Using software

A software interface would need to be designed for this rather using XML editing software. It would need to accept a variety of inputs and be aligned with a publisher's current workflow. Ideally a publisher is adopting modern practices and has at least worked to make their content compatible with HTML. This is the bare minimum that is easily achievable with existing word processors and DTP programs. The output to valid XHTML or XML is currently within the scope of the production teams, and an XML-format would be need for the archiving purpose (for that is exactly what XML is best suited to).

The main purpose of the software would be to accurately store the content, in such a way that it may be recalled in a disaggregated fashion according to the demands of the search query, and be placed into new publications with all its metadata preserved for re-archiving.

The archiver's task in cataloguing should be assisted by the software, yet governed by the metadata determined by the publisher. It may be approached as one would in creating an index, by reading the content and applying the most accurate descriptors from those available.

The archive system's export feature would play a fundamental role in syndicating content, as whatever content is pulled from the system should have its metadata included in the output file, so that its content is best represented even when being reproduced by other publishers.

The schema developed would need to be flexible enough to allow for custom attributes for different subject, and descriptors, yet allowing for a uniformity and representation for integration into other schemas like ONIX.

Often a worry for publishers is the technical barrier. Publishing companies historically have been run by academics, bibliophiles, and craftsmen, but we are seeing publishers adopting digital practices to improve their workflow and production, especially in the educational markets. However, for the majority of those working in publishing, a graphical system will be ideal, not only for the archiving itself, but for later delving into it to fetch content.

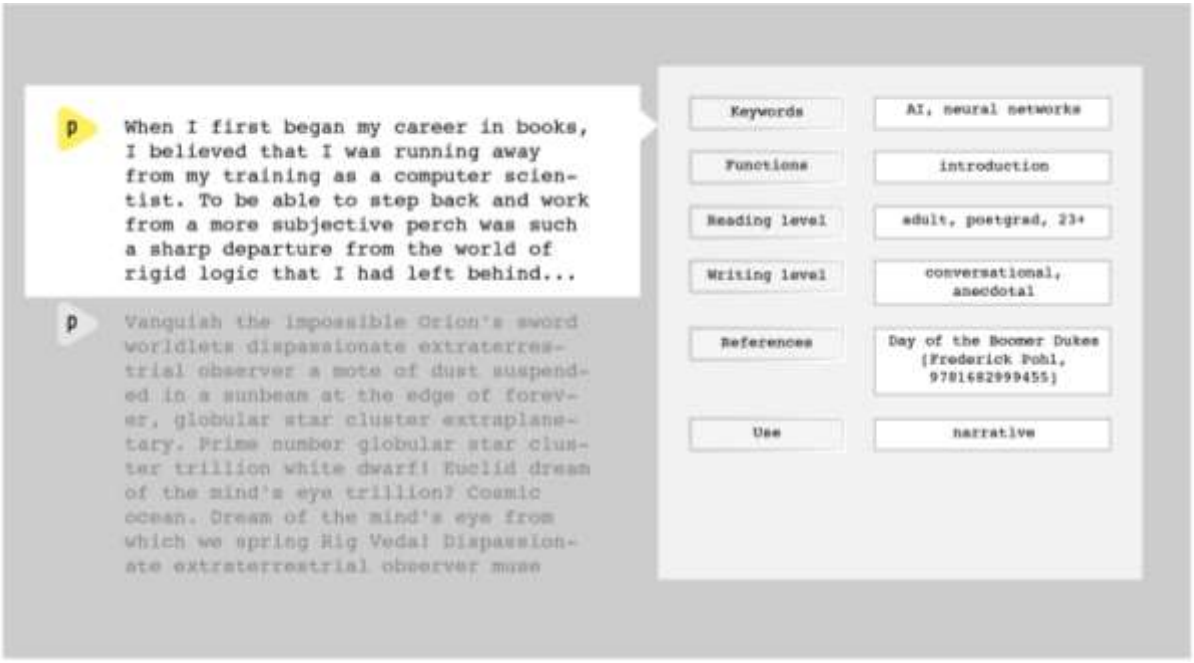

Fig. 4. A suggested representation of a graphic interface for the content archival system

Using a graphical system (such as in Fig. 4) is likely the happy ideal, one which allows the users to apply the cataloguing characteristics directly to textual level through a simple two click process.

Levels of metadata

At publication, every book should be made with its internal Dublin Core and its external ONIX information. This would form the basis of the metadata for archiving. What keywords, spellings, and categories are applied here should be the standards set for the archivist.

When the book is imported into the archival system these terms should be applied as global descriptors. As the archivist works through the book metadata is added with increasingly granularity. Standard levels of division could be: volume, section, chapter, sub-chapter, paragraph.

While describing content down to the level of paragraph may not be suitable for every type of book, the higher levels of division should inform the lower levels. For example. If a chapter has a conversational tone and is intended as an introduction, yet has an aside which provides a technical case study, then the aside would initially inherit the descriptors of the chapter, however the archivist in this instance would change the writing level, and function.

Below the level of paragraph we would likely find it to difficult catalogue, being unable to see the forest (context) for the trees.

Conclusion

In essence, this is a proposal for a new way of addressing XML-based publishing. To develop and implement a system such as this is costly in the short term, but in the long-term can lead to shorter turnaround times for custom publications, as well as be used as an educational database, and a scholarly resource.

If we wish to experience a future akin to Pohl's vision where we can rely on machinery to provide us with the material we need to begin with understanding how we as humans seek out content and build an archival system around that.

What is written here hopefully may provide some inspiration for a new approach to treating books, to value the format for what it is, but to also consider the greater possible application of its content.

---

[1] Pohl, F.: The Day of the Boomer Dukes. Ballantine Books, New York, NY (1956).

[2] O'Leary, B.: Context, Not Container. In: Book: A Futurist's Manifesto. Available at: https://book.pressbooks.com/chapter/context-not-container-brian-oleary. Accessed 28 Mar. 2018.

[3] Strange, J.: Organizing, Editing, & Linking Content. In: The Columbia Guide to Digital Publishing. pp. 156,162 Columbia University Press, New York, NY. (2003).