Bachelor Degree Project

UNIVERSITY OF SKÖVDE

Data models for interactive web based Textbooks

Investigating how well DocBook 5.0 supports interactive and multimedia content

Bachelor Degree Project in Computer Science
C-Level 30 ECTS
Spring term 2015

Sigurður R. Ragnarsson

Supervisor: Mikael Berndtsson
Examiner: Björn Lundell

In Memoriam
Jesus Martinez Barja
22.12.1959 - 27.06.2011

"Deyr fé,
deyja frændr,
deyr sjalfr it sama,
en orðstírr
deyr aldregi,
hveim er sér góðan getr."

Abstract

Much attention is focused on electronic publishing of books and textbooks. At present most of these eBooks are read on specific reading devices and published in different standards that are not necessarily compatible.
Advances in web technologies and the emergence of HTML5 and relevant technologies has greatly improved the possibilities of moving textbook publishing over to the web and use it as the preferred presentation layer. This also means that textbooks can be made much more interactive and it becomes much easier to add advanced multimedia features.
The DocBook document standard has been around for a long time and is used by variety of publishers in many types of publishing. On the eve of a new era in publishing it was interesting to investigate how well DocBook 5.0 would support these features made possible by advances in web technology.
This project shows that despite a sophisticated technical specification of DocBook 5.0, it offers only limited support for interactive and multimedia features.

Acknowledgments

First of all I would like to thank my instructor Mikael Berndtsson for his enormous support, professionalism and patience. To the people at the university of Skövde who gave their support on so many different levels. Last but not least I would like to thank my wife Bjarney Bergsdottir for her never ending support through the often difficult periods experienced during the work on this final project.

Table of Contents

1. Introduction

At present, eBooks are published in various formats. The most notable of those are: DocBook, PDF, ePub, Mobi and AZW, where PDF and ePub formats are supported by most reading devices. In most cases eBooks are designated to be read on special readers which are in turn proprietary devices such as Kindle, Pocket Book and Kobo Touch. Some of these vendor specific eBook readers support more than one format while other are more focused on a single format solution. Amazon's Kindle reading device is a good example of the former as a support for the PDF- format and text document(TXT) is provided as well as the Kindle format AZW, KF8.

These formats deal almost exclusively with the presentation of eBooks as electronic versions of their printed counterpart. The focus is on supporting the original layout structure and adapting it to the screen size of the reader's device. In other words commercial eBooks are merely books printed on an electronic screen instead of paper Shaffer(2012).

The reading device may be fitted with “helper” functionality like zooming and internal navigation. The same situation seems to be the case when looking at textbooks. The added structures found in many printed textbooks(assignments, quizzes etc) are not made interactive for the possible benefit of the reader, but are a replica of their printed version.

It is worth noting here that there is a vast number of standards and formats on offer when it comes to publishing documents and eBooks. For example There are many versions of PDF (e.g. PDF/A for archiving and PDF/E for engineering in addition to the ISO 32000 which is based on PDF 1.7). Further, PDF/A is released in several different versions (PDF/A-1, PDF/A-2 and both these are recognized by ISO). There are also PDF/X, PDF/VT and PDF/UA that serve special purposes. Similarly, ePub is a format that has been released under different versions.

Electronic publishing(ePublishing) is a young practice compared to the history of its traditional paper based counterpart. The first ePublications started in the 1980's as plain text subscription emails, moving on to electronic Journals in 1994-1995(Pettenati 2001). In 1999 the so called electronic book(eBook) started to gain more importance(Živković 2008).

An electronic book(eBook) can be classified as the digital version of a traditionally printed book or magazine designed to be read digitally on a personal computer, hand-held computer, personal digital assistant (PDA) or a dedicated e-book reader Reitz(2004). It has also been noticed that an eBook can be defined among others as: a text in digital form; as a book converted to digital form; and as a digital reading material. Rocci et al(2010).
Very early in the history of eBooks and electronic publishing it was realized that standards and openness were essential. Lee et al(2002) identified four critical issues that should be taken into consideration as well functional and technical issues:

- *Interoperability*
    - All parties involved should be able to exchange eBooks independently. To this end a non proprietary standard should be developed. Interoperability is seen as most important to achieve wide spread success of eBooks.
- *Extensibility*
    - eBook standard should be able to extend as to include new functionality like multimedia and user interaction.
- *Applicability*
    - An eBook format should be applicable to various kinds of related fields. Different types of style sheets should be used to compensate for various screen sizes.
- *Openness*

- eBook standards should be vendor independent. It must thus be an open standard that is accessible freely.

These four issues provide arguments for the way eBooks should be published in terms of type and structure. They still provide a good rationale for these publications to be moved entirely over to the internet where each of these these issues can be implemented in variety of ways perhaps more easily than in vendor specific solutions.

Recent developments in computer technology and the latest advances in web design provide a good opportunity to move eBook publishing from a device specific reading experience over to the internet(web). Instead of vendor specific standards and closed format reading mediums, electronic books could be published as (responsive)web applications that would be readable on all devices capable of displaying web pages. Furthermore the use of web as the preferred publication methodology opens up the possibility of interactive and perhaps other features previously not yet found in eBooks or e Textbooks. This lines up well with the conclusion that eBooks should provide innovative functionality that supersede traditional paper books(Shen, Koch 2011).

This method of publishing electronic books will most likely call for more stringent separation of presentation and data since using responsive web design means the layout changes dynamically according to the device's screen size on which the eBook is being read. These arguments for web publishing seem to be coming more important as information technology is becoming an essential presentational component of the curriculum(Wang and Towey 2012).

When a textbook is intended for publishing on the web one idea is to use an open standard providing a data model that retains structural integrity. While this standard would be similar to the current formats / standards used in eBook publishing it would probably need an addition to address the issue of interactivity and dynamics provided by the latest web technology. For example those who publish textbooks on the web should be able to take advantage of a standard just as those that publish for other formats.

Perhaps this publishing practice is made more complicated by the fact that at present there exists no eBook standard. eBooks are published by a way of various formats, some of which have become ISO standards. For example the ePub format is an ISO standard(ISO/IEC TS 30135). Similarly an eBook specific format does not exists, neither as a standardized format nor as a technical specification. As a result of this there is no overseeing body or committee concerned solely with such a format/standard.

A question now arises concerning the move of eTextbook publishing over to the web. Might one or more of the current eBook publishing practices on digital devices have the capability to handle such a transition over to the web. The first thing to do in answering this is to select a publication standard that is currently used in ePublishing and inspect it. The standard chosen in this work is the DocBook(5.0).

DocBook 5.0 is an OASIS standard. It is a collection of standards and tools for technical publishing(Stayton 2007) and also a major player in the eBook Publishing industry. Furthermore it is based on XML and therefore known XML methods and technologies can be applied with relative ease. DocBook also facilitates the separation of content and data and provides for additions of personal preferences. These facts make DocBook an ideal candidate for supplying an open standard for electronic Textbooks published on the web in accordance with the issues discussed above. Part of achieving this, is to investigate how well DocBook currently is able to include dynamic and interactive structures such as quizzes, activities, videos, sounds, dynamic help etc.

Choosing the right candidate for this project was complicated by the fact that there is no single standard(s) or format(s) that can be chosen. To assist in the selection a list of criteria was constructed. The purpose of such a list is to list as many relevant arguments as possible and

through that list, offer a clearer view of the formats or standards involved. This list can be found in appendix E. As an anchor for this list were the four critical issues identified by Lee et al(2002).

The motivation to select DocBook for this project was due to it's good conformance to the list of criteria:

1. It is currently in use.
2. It is a standard(OASIS standard), It is based XML that is itself an open W3C standard.
3. It's is used in textbook publishing.
4. It is developed and maintained by OASIS DocBook Technical Committee.
5. The versioning of DocBook is straightforward. There are no special parallel versions.
6. Textbooks using the DocBook format are both published on paper and as eBooks.
7. There is no need to include DRM in DocBook in order for it to work.

Besides these arguments, DocBook can be converted to various formats like PDF and ePub through dedicated software.
The selection of DocBook does by no means imply that other formats or standards used in publishing of electronic textbooks are in any way inferior or lacking functionality in any way. It is simply necessary to sharpen the focus on one single solution.

This project aims to answer the question of how well DocBook supports:

a) interactivity.
b) content responsiveness.

The project concerns itself in particular with textbooks that provide good feature candidates through quizzes, activities, tests, explanatory graphics, et al.

In order to answer this a systematic method needs to be developed and applied on the current DocBook(5.0). Through this method and its application the question of weather a feature is supported or not supported. The results should provide conclusions about the capability DocBook(5.0) to support web publishing of textbooks.

2. Background

2.1 DocBook 5.0

2.1.1 A brief history of DocBook

DocBook is a collection of standards and tools for technical publishing Stayton(2007). It's development started in 1991 as a joint project between the Hal Computer Systems and O'Reilly & Associates. It's origin are in SGML(Standard Generalized Markup Language) and it was defined with a set of markup declarations known as DTD Walsh(2010).

Upon the release of DocBook 1.1 a forum was created for maintenance and development. This forum was named the Davenport Group and in 1994 it became the officially responsible for the maintenance of the standard. Under this management, the standard grew in scope and it´s audience continued to grow. But eventually, development slowed down and in 1998 the standards activities were moved away from the Davenport Group to OASIS(Organization for the Advancement of Structured Information Standards). OASIS carried out and continues the work that was started by the Davenport Group and the standard is still maintained by the organization.

The version that forms the basis of this work is version 5 (sometimes referred to as 5.0).

2.1.2 The structure of DocBook

DocBook is a semantic language based on XML and therefore a DocBook file is an XML file and those rules that apply to XML also apply to DocBook. Older versions of DocBook used Document Type Definition (DTD) and XMLSchema as it's main schema languages and a lot of DocBook validations is still done using these two.
However the normative schema for DocBook 5.0 is the RELAX NG grammar with it's Schematron annotations Walsh(2010). Furthermore as of version 5.0 DocBook is contained within it's own name space which is new to version 5.0.

One aspect of the DocBook 5.0 structure is the logical division of it's elements into categories. Walsh(2010) broadly divides these categories as follows:

- *Sets*
  - Used for treating a set of DocBook files as a single one. Useful in a series of books.
- *Books*
  - DocBook book elements are defined in a most general manner. This is due to to sheer variety of books described by DocBook on a worldwide scale. The book author is given a free space to operate. If need be customizations can be used for more strictness in structure.
- *Divisions, which divide books*
  - This category has two elements: part and references. In the DocBook hierarchy these elements are directly below the book element. This means that they are intended for a logical division of the book it self into bigger sections(parts).
- *Components, which divide books or divisions*
  - These elements are designed to be the chapter- elements of DocBook. Whereas division elements are not always used, these elements can be placed as a direct sub set of the book- element.
- *Sections, which subdivide components*
  - Sections consist of various elements for breaking DocBook into sections. Most of these elements are nested and some have (nesting)numbering: sect1,sect2 etc.
- *Meta-information elements*
  - The info element is a wrapper around information about the content(meta-information). An info element can contain a number of other elements, designed to

provide the possibility for a more detailed information.

- *Block elements*
  - These elements like inline elements deal with what can be called a geographical dispersion(space) of DocBook text. These elements occupy multiple lines of page space. Block elements are usually presented with a paragraph break before and after them Walsh(2010).
- *Inline elements*
  - Whereas block elements can be seen logically as a block of elements, inline elements on the other hand are designed to deal with single lines of DocBook text.

All the elements of DocBook describe a book/article data structure. They do at no time dictate how elements are to be treated visually. This emphasizes is on the separation of presentation of data.

2.1.3 RELAX NG

DocBook 5.0 structure is defined by the RELAX NG schema language(Regular Language for XML Next Generation. It was developed in 2001 and 2002 by the OASIS RELAX NG technical committee and is described by ISO/IEC 19757-2:2008.

According to Stayton(2007) the benefits of RELAX NG as the official DocBook schema include the following:

- It handles name spaces.
- It allows the element's content model to vary according to it´s context.
- It is relatively easy to read in it's compact form.
- It is quite easy to customize in order to extend or subset the DocBook schema.

A RELAX NG schema specifies patterns for the structure and content of an XML document. It comes with two forms of syntaxes: an XML syntax and a so called compact syntax(RNC) which is the syntax used by default in the DocBook schema. These two formats have the same expressive power; it is possible to transform between them with no loss of information Walsh(2010).
RNC notation is an expression syntax and is in general similar to the syntax definition of context-free grammar found in BNF(Backus-Naur form) Shipman(2011). See *appendix D(table d.1) for an example*.

Shipman(2011) divides RNC into three categories by their role in the schema structure:

Named Patterns – Used for reduction of verbosity in schemas due to repetitive patterns. The way this is accomplished is to identify identical attributes on different element types and then from those, create a named composite element that is then referenced from within the corresponding elements.

Content Patterns – Used to describe the patterns by applying constraints such as “zero or more”, “one or more”, “optional” and more. Each patterns has a corresponding operator.

Definition Patterns – Used to define the overall structure of a RNC schema. These patterns include amongst others the following definitions:
name – names a pattern
element name{(type)constraint} – defines an element and it's matching constraint
attribute name{(type)constraint} - used inside an element to specify that it must have an attribute called name.
In addition to patterns the RNC uses a predefined set of standard data types that are an important part of XSchema. Those are contained within the xsd- namespace.

2.1.4 Scematron

Schematron is a language for making assertions about patterns found in XML documents Walsh(2011). It is not based on language grammar like RELAX NG and XMLSchema but is focused on finding patterns within documents and can be used to test assertions of arbitrary complexity about those specific patterns.
Schematron is used in DocBook to provide validations that go beyond validating the document structure. Generally for those constraints that can't be expressed in RELAX NG a Schematron validation schema is used.
For example, a Schematron rule is added to prevent a sidebar element from containing another sidebar.
For Schematron code example see appendix E

2.1.5 Customization

Customization of the DocBook schema basically involves adding new elements, attributes and/or rules. It also involves the removal or modification of existing language structures. These customizations are done through the RELAX NG schema since it offers better support for modification than DTDs Walsh(2007). Customizing DocBook is facilitated through the so called customization layers. A customization layer is when a modification is layered on top of the existing DocBook schema. Creating such a custom schema is very similar to creating a customization layer for XSL as the new layer is a new RELAX NG schema containing the changes as well as including the standard DocBook schema. Customization layers come in various stages of complexity but they tend to have similarities with other layers. Most designs include the complete DocBook standard and then add the modifications.
The process of modifying DocBook has several aspects that should be taken into consideration.

General considerations - is the question of if the customization includes a *subset* of DocBook or *extension* to the standard. A subset that adheres to DocBook is a valid DocBook instance where as an extension is not. This is mainly due to the license agreement that under which DocBook is distributed that states that alterations to the standard can not be called DocBook.

Namespace considerations – is related to the change in DocBook namespace and it's version. If a DocBook schema is modified this is to be reflected in the namespace declaration. The namespace as such is left unchanged but an alternative version- part must be supplied. According to Walsh(2011) t he DocBook technical committee recommends the following format:

base_version-(subset | extension | variant) (name[-version])+

It is worth noting here that the variant keyword is used if a characterization of the modifications made to the document should not be stated!

Structure considerations – RELAX NG is a collection of patterns distributed in DocBook as a single file. It is however possible to distribute the schema in a collection of files. The RELAX NG schema that defines DocBook is broken into several modules by using named patterns for logical grouping of related elements and attributes. These pattern can be used as a handle to the part of the schema that is to be customized. For example, *db.*.attribute* is *a* convention that defines a single attribute. An example of usage is *db.depth.attribute* which in turn is a pattern that matches the depth attribute where ever it appears(in every element that contains it).

Usability considerations – Deciding what to change in the DocBook schema is of great importance since it affects the intended structure and behavior of the elements in question. For example adding an element may include numerous patterns that could be changed. Choosing from those patterns and applying the modifications is what determines the behavior of the intended customization. An important consideration is to administer thorough tests of the changes made and validate the results as well.

Adding and Removing considerations - Due to it's size it may be a good idea to remove elements that are not needed for a given purpose. This is a straightforward process where an element is simply redefines as not allowed. A situation can occur where the complete removal of an element is not the desired case but that an element should be removed from a certain context. This sort of removal calls for a redesign of those patterns that allow the element within the targeted context. The process of removing attributes is quite similar.
Addition of elements is a relatively uncomplicated process: it is created and added to the desired pattern using an operator. There are however situations when an element is a part of complex structures. This may involve a more complex addition. Attributes as usually added to the add list pattern of the element that it's being added to.

2.1.6 DocBook Transformation

The DocBook data model is transformed to different presentation formats through the use of XSL(Extensible Stylesheet Language) and XSL Transformations(XSLT). This allows for a variety document types to be generated.
XSL stylesheets are available for DocBook to use in publishing of formatted content. These stylesheets are now an open-source project maintained on SourceForge Stayton(2007). These are in divided into a collection of stylesheets capable of generating HTML output(both as a single file and multiple files), files for printing, XHTML output, HTML help output and JavaHelp output. To increase flexibility in transformation using DocBook two options are available, using the DocBook stylesheets and customizing the XSL.

Stylesheet Options:
Stylesheet parameters – Named parameters that that can be assigned a value. In this way the DocBook Stylesheet formatting can be changed without rewriting the transformation entirely. These options can be split roughly in two: HTML output options and print output options.

HTML output – A single file output or multiple files output can be generated from a DocBook XML file. Generating a single HTML file is done by the *docbook.xsl*. The xml file is transformed as a whole into an **.html* file. There is also a possibility of processing a part of a document and the DocBook stylesheets have a parameter that makes this possible Stayton(2007). There are however conditions to use this feature.
DocBook stylesheets support dividing a (large) document int several HTML files. This process is called *chunking* and the individual files created are called *chunks* Stayton(2007). This option contains several ways to control how this chunking is done to affect the resulting HTML files.

Printed output – Printed output is generated by processing DocBook xml files to create a **.fo* file and then processing it with a XSL- FO processor. The process involves the control over various parts of the document intended for printing including parts like:

1. Page layout
2. Typography
3. Chapter numbering
4. Page breaking
5. Cross reference page numbers

Customizing DocBook XSL:
Methods – Sometimes there is a need to change a DocBook XSL but there are no parameters that can be used. In such a case it is necessary to customize the XSL stylesheet(s). A general approach is to create a customization layer for the changes needed and continue to use the DocBook stylesheets wherever possible. Stayton(2007) describes various customization methods:

1. Creating a customization layer
2. Setting parameter values

3. Modify attribute-sets
4. Filling in placeholder templates
5. Customizing generated text
6. Replacing stylesheet templates
7. Adding new Templates
8. Creating custom processing instructions
9. Generate customized title page templates.

Customizing both HTML and FO(print) – Some customizations that are made to DocBook XSL are applicable to both HTML output and Print output(FO). This is due to the fact that some code and processing styles are shared Stayton (2007). The customization that works on common parts for these output styles are put in a separate file and then included separately.

Customizing HTML – Sometimes neither CSS stylesheets nor stylesheet parameters are quite sufficient to make the modifications needed. In such cases a customization of the DocBook XSL is the necessary action to take. These customizations could include the following Stayton(2007):

1. Generate custom class values
2. HTML headers and footers
3. Server side includes
4. Inserting external HTML code
5. HTML head elements
6. Body attributes
7. Chunking customizations
8. Return to top
9. Customized hrefs

Customizing print output – Modifying and customizing printed output of a DocBook file can unsurprisingly require more work than is necessary for it's HTML counterpart. This is in part due to the fact that style properties in a DocBook XSL need to be specified for FO output which is then processed for printing(PDF / PostScript). Following is a list of various document parts that can be affected by customization Stayton (2007):

1. Document level properties
2. Title fonts and sizes
3. Custom page design
4. Print TOC control
5. Running headers and footers
6. Borders and background shading
7. Customizing inline text
8. Customizing admonitions
9. Side-by-side formatting
10. Side floats
11. Multi-columns and spans
12. Adding a font
13. Numbering paragraphs
14. Adding line breaks

2.2 OASIS and DocBook maintenance.

DocBook is developed and maintained by Organization for the Advancement of Structured Information Standards (OASIS). It is a non-profit consortium that works on developing and distribute open standards. It was formed in 1993 under the name SGML Open. Five years later the original name was changed to OASIS in tandem with the move from SGML to XML.

The governance and operating procedures are transparent and the technical agenda is set by the

members themselves. The decision making process in carried out in open ballots.
OASIS is divided into (technical) committees that oversee individual standards development and maintenance. One of these committees is dedicated to the DocBook standard.

OASIS DocBook Technical Committee:
This committee, founded in 1998 develops and maintains the DocBook specification and it's subsets such as extension modules and simplified DocBook.
The committee retains a charter that states it's purpose and defines it's scope of work. This charter is modified to reflect the changes and developments of the DocBook standard. For communications and community interaction the committee relies on mailing-lists where members can share information and feedback can be provided by the public. There is also a OASIS DocBook Wiki on the web(URL: https://wiki.oasis-open.org/DocBook). In addition to this the committee contains two subcommittees:

Publishers – This sub committee is chartered to develop and maintain official variants of DocBook in support of the publishing industry. It's focus is on schema and stylesheet developments in support of book publishing along with other forms of appropriate publishing (oasis web taken 10.06.2013).

eLearning – The OASIS DocBook eLearning Subcommittee develops and maintains official variants of DocBook in support of eLearning content. The focus of this subcommittee is the schema and customization that is needed to structure eLearning content(oasis web taken 10.06.2013).

OASIS DocBook Standard:
The classification of DocBook V5.0 as an OASIS standard is published on the OASIS web (Standards | OASIS 2015)

The DocBook Technical Committee is chartered to develop and maintain the DocBook family of specifications. (OASIS DocBook TC | OASIS 2015)

Intellectual Property Rights is an important aspect of an open standard and OASIS states their IPR policy on their the web (Intellectual Property 2015).

DocBook version used in this work:
The following technical specification was referenced during the DocBook investigation:
The DocBook Schema as published by OASIS. (The DocBook Schema 2015).

The following normative references were also used during this work:
Extensible Markup Language (XML) 1.0 (Fifth Edition). (The DocBook Schema 2015)
DocBook 5.0: The Definitive Guide. (The DocBook Schema 2015)

3. Problem description

3.1 General considerations

DocBook is richly equipped with elements, attributes, rules and flexibility that make it a powerful data modeling tool for eBooks. Furthermore it has been around since 1993 and enjoys a considerable support within the IT industry. One of it's strong point is it's availability as an open standard and therefore accessible to anyone without any costs.
DocBook has it roots in structuring technical documents that where then printed on paper. Since today's eBooks are often than not books printed on an electronic screen instead of paper Shaffer(2012) the standard has prevailed in both printed and electronic publishing.

Only in recent times have eBooks started to become more interactive to the extend of providing more than hyperlinks to resources on the web. Now they allow the reader to highlight text, add notes and bookmark pages. Newer interactive features include embedded audio, video, slide shows and image galleries Fenwick et al. (2013).
These new features and emerging technologies are however driven by proprietary reading devices and vendor specific technologies. Much less attention seems to be given to open standards and vendor independent solutions.
As a result of this, it becomes interesting to investigate if such interactive content could be standardized to some extent and made accessible through the DocBook 5.0.
This project focuses on developing methods by which the DocBook can be investigated as to identify possible support of multimedia and interactive content of eBooks and in particular eTextbooks.
Defining appropriate extensions involves a close inspection of standard DocBook elements that could support the modeling of such content.
Criteria for modifying these elements must be set as not to alter the original role and create ambiguity in the specification. This criteria should also indicate when new elements/attributes should be created. For each added or modified element in DocBook there should be a definition in RELAX NG and the new or modified structure must be validated. Consideration should be given to the methodology of defining more complex rules in Schematron.

DocBook is very much about data modeling, and the role of this work still is to investigate if the current parts of the DocBook standard provide enough information to be transformed to HTML5 or other languages that support responsive web design. XLS transformation must be considered as an example to assess the quality of the markup.

The aim of this project is to review the DocBook data model in order to establish current support for interactive eTextbook content.
The current elements of the standard will be evaluated against a criteria representing an interactive alternative.

3.2 Related Work considerations

Related work i.e work focusing on data models or data structures for eBooks published on the web was not found in CS literature during the time frame of this work. In fact no articles were found about eBook data models at all that could be used for comparison and further discussions of this work.
Since this work is aimed at looking at the DocBook 5.0 model for current support of interactivity and dynamics as a part of publishing textbooks on the web, gathering material regarding web publishing suggests interests among researches for publishing on the web using web 2.0 techniques and designs. While not providing direct proof of the actuality of this work, it could provide least some indirect support and help justifying its existence.

This support is mainly deducted from the fact that in order to publish textbooks(or any book) using current web HTML5 based technologies, the separation of representation and data must be utilized and therefore it can be assumes that such a web would benefit considerably from a data model.

It has been pointed out that a book page structure along with metadata, interactive objects and relations between book objects are in fact pieces of the *book data* and need to be well defined to enable various types of reader software. Gregorief et al. (2011)
This identification of book data and the need for clear definitions, gives support to the validity of researching book data and possible data models. It therefore seems, if indirectly, to support an investigation into the DocBook 5.0 model.

Table 3.1: Support of relevance

Level	Level description	Support
L1	Papers discussing data models for eBooks published on the web.	Strong
L2	Papers discussing various web publishing techniques, ideas and forms.	Indirect
L3	General discussions about ePublishing often involving vendor specific solutions.	Vague

Since no papers in Level 1 from the table were found and the discussions offered in Level 3 were too vague, the supporting relevance was taken from three papers that fall into L2 as classified in table 3.1.

Shaffer et al. (2011) point out several subsystems that would have to be in place for such a project. While in their work the focus is put primarily on the task of a community taking part in writing an eBook, they identify the importance of *data structure support* and point out the awareness that contributors must have of this and the other subsystems.
Furthermore it's found that the only technology robust enough to implement the dynamic and interactive components, is HTML5 along with JavaScript and CSS. Shaffer et al. (2011)
In particular the identification of data structure support pints out an important part of content creation for the web, can be seen as another supporting claim favoring a research of possible eBook data models and existing ones. It seems to add further support to the general idea that a responsive web design is almost impossible without some underlying model for the respective data.

Beer et al. (2011) identify the following requirements for their system:

- Must offer possibility of multimedia content.
- Support for device independent representation.
- Simple and standard delivery through already established protocols on or off line.
- Context awareness.
- Offline Mode.

Again the focus is put on multimedia content and device independent presentation. While this requirement list does not add anything directly related to some underlying data model it is addressing the same or similar issues of web publishing as the other works recited here. It is added here for the purpose of demonstrating how similar the tasks and considerations are in this seemingly emerging field of book publishing.

3.3 Aim

The aim of this project is to review the DocBook 5.0 data model in order to establish current support for interactive eTextbook content.

3.4 Objectives.

- Develop a set of criteria for interactive content.
- Develop an evaluation method.
- Apply the method using the given criteria.
- Evaluate the results based on the application of the method.

4. Method

As the goal of this project is to investigate how well DocBook 5.0 supports interactivity and multimedia a systematic method must be developed in order to safely be able to make a reliable comparison between hose features and the DocBook 5.0 specification to reveal if a certain feature is supported or not supported.

As there are no standards for writing a textbook or the structure of it's contents the approach taken here is to select simple, straightforward features that an author could include in a textbook designed to be published on the web.

On each of these selections a scenario is constructed providing a usage overview that is then broken down into use-case style parts for a more detailed description. These use cases are then used to create a requirements list that serves the purpose of providing a list of XML elements and attributes that are needed for the final comparison with the DocBook 5.0 elements. This approach relies heavily on UML software design methods and has its root in user oriented software design.

4.1 Selected Candidates

The features that in the end were chosen for this approach were a *video* that could be embedded in a chapter and a simple *quiz* containing single choice *questions* that could also be embedded in a book chapter. The quiz and questions are of cause closely related but are split up here for the sake of simplification.

As mentioned above, there are no standards that can aid in choosing a candidate and therefore the basic idea is to focus on features that could greatly benefit a web published textbook. In the case of an embedded video this is quite widespread in open learning systems structure known as MOOC(Massive Open Online Course) where lectures are prerecorded and added to the course content. The quiz is then chosen to represent a candidate that can be easily implemented in a web context and could also be used in a non-interactive versions of a textbook. There is also the factor of time constraints of the project is self that affects the choice of candidates and limits the overall complexity that is manageable in a project of this magnitude.

4.2 User Scenarios

To start extracting possible XML elements and attributes the approach taken in this work is to create user scenarios describing the intended usage of each candidate as closely as possible. These scenarios provide the first clues of how the respective functionality could/should me marked up in the DocBook 5.0 standard. The creation of user scenarios follow these two set of considerations:

Scenario structure:
- *Reference number*
- *Scenario name*
- *Scenario description*

Description narrative:
- *Include location(where or in what circumstances does the scenario take place).*
- *Describe a fully working system.*
- *Follow a straightforward flow in the description.*
- *Focus on what actions must be taken as opposed to how they are carried out.*
- *The narrative is a textual description and should be written as such.*
- *Iterate through the description with respect to:*
    - *Logical flow*

    - *Unnecessary repetition*
    - *Relevance*
    - *Accuracy*
- *Document any changes made during the iteration process.*

Example 4.1: A simple user scenario

Scenario 1: Author adds a video to a textbook chapter

The author has decided that a part of a chapter should be a video that the reader can watch. The author has already chosen the appropriate file that contains the video and has located the intended chapter and the appropriate place in which the designated video should be placed. The author locates the file and carries out the necessary commands to „insert" the video at the right location within the chapter.

A list of all scenarios used in this work is given in appendix A.

4.3 Use Cases

While the user scenarios provide an overall structure of how a feature is used it is still too general to provide document structure elements. What this means is that a single scenario needs to be broken down further. This is done by creating use cases from each description. This process identifies possible features that extract information from the scenario in a structured way that will later be used in the construction of a requirements list.
There are five aspects to be emphasized during this breakdown phase. Each individual part of a use case provides structured information on the scenario in question.

Action – A scenario can include a number of actions. A single action forms the basis for the use-case style analysis of a scenario. As an independent entry the *action* helps to establish an understanding of the scenario itself by emphasizing the individual parts the actions derived from the scenario.

Prerequisites – Everything that must be in place so that the corresponding action can take place. This may provide information about constructs that should be defined in the extended DocBook schema.

Constraints – Any limitations (constraints) that should be applied to the action, such as legal range of numbers, data types, set of values etc. is declared in this section. When transforming these individual parts into a requirements list the constraint section should provide a good ground for attributes and rules.

Expected result – States the intended results of a certain action. This provides an important part of the transition from a scenario to a requirements list as this result can be matched to a possible set of DocBook extension constructs and aid in determine the "best fit".

Notes – If necessary notes should be taken down with the intent to aid in the transformation to the requirement list. No restrictions should be applied here.

Additionally there is an analysis segment number that can be referenced.

Example 4.2: Use case (a scenario may require more than one)

Number	1
Action	Author adds video to chapter
Prerequisites	Chapter exists (identifier). Page (in the chapter) is identifiable
Constraints	Video: must be identifiable. Each must be assigned a unique identifier
Expected result	The video file information is stored and the video becomes a part of the chapter.
Notes	This is done when an author has decided to include a video in a chapter

A list of all the scenarios used in this work can be found in appendix B.

4.4 Requirements list

When the use cases have been constructed the final step of the method is to create a list of requirements that more closely resembles descriptions of DocBook elements and attributes. The role of this list is to name possible elements needed for a specific feature and in doing so make a comparison with DocBook easier and more reliable.

The requirements list is divided in two halves, *requirements* and *preliminaries*. This emphasizes the role that the list should play in the capture of requirements from a scenario use case. The list should also help to provide a way to identify these requirements as suggestions for the final evaluation of DocBook elements. The Requirements list is structured in the following way:

NO – Requirement number. This is a unique number for each requirement. More than one requirements can be extracted from a single scenario analysis segment. Each one

Requirement – This is a short sentence describing the respective requirement.

RN – Reference number. Provides a handle to a scenario analysis. If more than one requirements are found in a scenario segment they all share the same value for RN.

Type – Based on the context in the scenario being broken down, a type may be identified as being either an element or an attribute.

Rule – If an attribute value is subject to some constraint or rule it is marked here by an identifier.

Name – Both elements and attributes are assigned names. If the requirement has been identified as an element the name in put in <*element name*> wrapper.

TagRef – In a DocBook hierarchy each element has a parent element until the top of the document is reached. The list offers the possibility of locate the potential tag within the anticipated structure.

Datatype – The last preliminary in the requirements list is the suggested data type. If the requirement is judged to be an element the data type is set to *tag*. Furthermore if a rule for the respective requirement has been identified the data type is set to the rule identifier for the element in question.

Example 4.3: A simple requirement list demo.

Requirements			Preliminaries				
NO	**Requirement**	**RN**	**Type**	**Rule**	**Name**	**TagRef**	**Datatype**
1	Chapter has a unique identifier	1	A	NO	id	<chapter>	int
2	Video is identifiable	2	E	NO	<video>	<chapter>	tag
3	Video has a unique identifier	3	A	NO	id	<video>	int
4	Video has interactivity level	n	A	YES	ial	<video>	R1

A full requirements list used in this project can be found in appendix C.

4.5 Other approaches

As has been made clear there are no standards for writing a textbook neither for traditional printed versions nor the digital counterpart. This opens up a variety of possible approaches and angles of attack as to how to set up a sensible method to fulfill the aim of this thesis.

One way might be to inspect one or more of the many Learning Management System(LMS) that are in use and choose interactive and multimedia features from the system in question. Then break those features down to evaluate against the DocBook Specification.

Another way might be to design a database supported “WebBook” editor software using object oriented design and then analyze both the Class Diagram(s) and the Entity Relationship Diagram(s) to extract the features for evaluation.

The third way might rely on similar methods as is used in this work but choose totally different set of features to work from and doing so with a different set of arguments.

5. Investigation

After the user scenarios have been constructed, broken down into use cases, transposed into a requirements list for the necessary new elements a systematic search of DocBook 5.0 elements and their description was performed. The result of this search was then compared with the elements found in the requirements list

5.1 User scenarios

As stated in 4.2, The evaluation process is started by creating scenarios. For this project a total of three scenarios were created. The scenario creation followed the guidelines provided in 4.2 and each scenario was iterated through three times. The focus when applying these guidelines is to create a clear narrative of the work flow, simplifying the process of breaking the scenario down into use cases.
A part of scenario text:

"Even though a single choice question and a multiple choice question have a lot in common the system starts by asking the author which type he/she wants to create:
The first step for the author is to connect the question to a certain material(s) in the eTextbook. This step is optional but the question can be tied to multiple parts of the book.
As the next step, the author does is to add the question text itself. Then comes the decision on how many answer possibilities there should be.
When the author has chosen how many answers there are, the system opens up three input fields for each question. "

From this example description it was possible to write up a list of actions:

- Author creates a (single choice)question
- Author connects the question to a material
- Author adds question textbook
- Author chooses number of answer to the question
- System displays input fields

From this type of lists use cases were created.

The three complete scenarios developed for this project are displayed in appendix A

5.2 Use cases

In 4.3 the construction of use cases from the scenarios was described. A total of 47 use cases were constructed from these scenarios.
As stated in 5.1 the first step in breaking down the scenarios was to derive a list of actions and possibly actors as well. When considering each item in the list, use cases were only created for those actions that would directly result in an element and or attribute. In other words would a structural element(XML) be necessary as a result of this action being carried out.

An example of a use case construction from a scenario via an action list:

Number	29
Action	The author chooses a single choice question:
Prerequisites	None
Constraints	The question gets a unique identifier (ID)
Expected result	A new single choice question is now ready to be constructed.
Notes	This is basically the same as number 38 and serves the purpose to emphasize the type of the question.

The action name of the use case may vary slightly from the action list. The scenario was scanned for a possible hint of prerequisites that would affect the action to be carried through. Since it was deemed necessary to equip each question with a unique identifier a constraint was incorporated. In order to increase the accuracy of the use case each one has an expected result description that in as short a text as possible states what changes will have taken place when that use case is carried out. This is a common practice with use cases in software development but it had the importance here that if the result did not imply any sort of data needing modeling, the use case was simply dropped.

For example the action: “System displays input fields” did not make it to become a use case simply because no data would be generated by this action!

Full list of completed use cases in in appendix B.

5.3 Requirements list

In accordance with 4.4 a list of typical requirements for elements and attributes is derived from the use cases. In total 51 requirements were created from the use cases.
As mentioned in 5.2 use cases are constructed only if it's action resulted in data being modeled. The requirements list act as a kind of design template for that modeling. It's role is considered to be a draft description of the XML data structure created by the use case.

As an example of requirements derived from the use case example given in 5.2.

NO	Requirement	RN	Type	Rule	Name	TagRef	Datatype
37	Question is an element	29	E	NO	<question>	<chapter>	mixed
38	Question has type	29	A	NO	type	<question>	string
39	Question has unique identifier	29	A	NO	Id	<question>	integer

Since the use case stated that a single choice question was being created that fact was treated as a type. The identifier became id and then it was clear that a question was an element.

It is important to notice here that a single entry in the requirements list was not necessary built from a single use case. There may be more than one involved in the decision making as well as consulting the scenarios.

A complete requirements list is in appendix C.

5.4 Candidate elements

Going through the DocBook elements in search of possible candidates, two approaches were made:

1. Searching by element *name*.
    - The names in the requirements list were used as a reference

2. Searching by *context*.
    - Is feature supported by element of a different name.
    - A look was taken at DocBook's elements.
        - Common attributes
        - Link attributes
        - Processing expectations

The context search yielded no results. The results of the name search are in appendix D.

5.5 Evaluation

At the core of DocBook 5.0 are the elements that contain attributes, other elements and to a lesser extent, complex rules and other parts. Walsh(2010) gives an overview of the DocBook 5.0 elements and in his text the description of each element is roughly divided into the following sections:

Element name
- Short comment

Synopsis
- Content Model(*described under synopsis*)
- Attributes
    - Additional attributes
- Additional Constraints
- Description
    - Processing expectations
    - Future Changes

Attributes
See Also
Examples

This is the "evaluation template" that was used for the final step in applying the method. Not finding any element candidates when searching by context the task of evaluating the remaining DocBook elements was somewhat simplified.

Three types of outcome were expected when applying the evaluation method:

1. *Feature is supported*:
    - An element in DocBook can be used without any modification.
2. *Feature is not supported*:
    - An element feature in DocBook must be modified to fit the new context(partly supported)
    - An element does not exist and needs to be created.

Example 5.1: A evaluation template designed to clarify element differences / similarities

Element description(based on Walsh(2011)	Use case and Preliminaries
Name	TagRef
Short comment	Notes?
Synopsis	Rule
Attributes	[DocBook common attributes]
Additional Attributes	Type + Name
Additional constraints	Prerequisites + Constraints + Rule.
Description	Requirement
Processing expectations	Expected Result
Future changes	
Attributes	Description of Type + Name
See also	Notes?
Examples	

Example 5.2.1 Resulting XML code for a task having consulted the evaluation template.

Task	Embedding a (standalone) video in a paragraph that is situated within a chapter.
Requirements	<chapter> <para> <video path='' interaction='zero'> <title></title> </video> </para> </chapter>
DocBook	<chapter> <para> <title></title> <videoobject> <videodata fileref='' /> </videoobject> </video> </para> </chapter>
Consulted requirements	3,6,12,
Consulted use cases	2, 8
Notes	DocBook 5.0 uses videodata embedded in a videoobject. The fileref attribute is interchangeable with the path attribute from the requirements list
Conclusion	DocBook 5.0 supports without any adjustments the placement of a (standalone) video within a chapters paragraph.

Example 5.2.2 Resulting XML code for a task having consulted the evaluation template.

Task	Embedding an interactive video in a paragraph that is situated within a chapter. The video is to halt after 2:17 minutes of playing time offering some internal assignment(task) to be performed. Upon the user finishing the respective task the video continues.
Requirements	<chapter> <para> <video path='' interaction='internal'> <title></title> <stops> <stopat>2:17</stopat> <stops> </video> </para> </chapter>
DocBook	<chapter> <para> <title></title> <videoobject> <videodata fileref='' *missing interaction attribute* /> *missing stop time features* </videoobject> </video> </para> </chapter>
Consulted requirements	3,6,12,20,21
Consulted use cases	2, 8,13,14
Notes	DocBook 5.0 lacks elements and attributes
Conclusion	DocBook 5.0 does not support this feature since it lacks elements and attributes specifically related to the task to be performed. For example by not being able to reveal the intended stop time for a specific assignment, external applications such as JavaScript, used at the presentation level may not be able to add interactivity.

The evaluation was not based solely on the requirements list but the use cases and even scenarios were consulted. As the number of elements to be evaluated was only five the template shown in example 5.1 was not needed.

Table 5.1 Evaluation summary

Feature	1st evaluation	2nd evaluation
Video with interaction	Not supported	Not supported
Video(stand alone, embedded)	Supported	Supported
Question with answers	Supported	Supported
Single choice question with answers	Partly supported	Not supported
Quiz with single choice questions	Partly supported	Not supported

First evaluation corresponds with a preliminary outcome and the second one is the final result were there the feature evaluated must be supported without any modifications on existing DocBook elements.

5.6 Result analysis

The results provided in 5.5 were to be expected as DocBook 5.0 has primarily been the backbone for printed documents, books and their electronic counterpart. This may support the claim that the web publishing of textbooks is really in it's infancy.

The fact that these results do not reveal any anomalies or inherent limitations in the DocBook elements being inspected leads to the conclusion that the method chosen is sound and could be applied to evaluation of other features.

A strong position was taken as to what would suffice as a “feature supported”. This has of cause affected the evaluation results. The choice of features to use in this project may have had an effect on the results as well.

6. Conclusions

6.1 Contributions

The main contribution of this work is the construction of a method to answer the question whether an interactive and/or multimedia feature is supported or not in the DocBook 5.0 standard.

Another contribution is establishing the fact the DocBook 5.0 standard does not as it stands support those features that were developed as a test case.
What this means it that there are no tags in the DocBook semantics that can be interpreted directly to embed the functionality required by the scenarios used in the test case. Since DocBook is a set of standards and tools for technical publishing and was originally created as a standard for computer documentation(Streyton 2007). This lack of support does however not exclude that such support can be added to the features of DocBook.

6.2 Problems

The biggest challenge was the fact that there exists no standard nor best practices for writing a textbook on any subject. In addition there does not seem to exists any work into eBook data models. This may indeed provide certain amount of freedom but there are inherent problems. The author can not rely on or reference the work of others and thus an important quality control is lost.
Another potential problem is that the choice and development of the method and the application of that method are the sole work of one person that does not benefit from the work of others.
Unconsciously parts might have been missed, or misinterpreted. Test of the various parts of the method would benefit greatly were they carried out by others or at least had some reference to rely on. The lack of related work could even mean that the subject is considered unimportant and even insignificant.

6.3 General discussion

Web book publishing has just been boosted by the advent of new technologies. It will probably lead to textbook publishing on/through the web bypassing the current formats. In order to distinguish a textbook from a normal web site a structure or a data model must be put in place. It would be of great benefit were this data model to be separate from the presentation layer and not at any point mix the book data and presentation. Furthermore a web book should be tested against this model and verified.
It was interesting to find out if one of the publishing standards currently used was capable of handling the almost inevitable demand for more interactive and feature rich books that follows publishing on the web.
The American publisher O'Reilly uses DocBook as the companies publishing format and such a firm with an enormous catalog of textbooks surely could benefit if the document model was ready to support a new way of publishing. Even if a ePublishing standard did not include support for these features it could be argued that adding to or augmenting the DocBook OASIS standard so that a support were established would be of value. One reason being to enrich a relative static books with multimedia features such as videos with interaction, responsive assignments and complex quizzes.

The basic aim of this project was to investigate if the DocBook 5.0 standard supported interactive and multimedia features. Comparison with the chosen features showed very little support and the

conclusions were that in this particular test it dot not have a support for those features.
What this means is that perhaps it's time to start preparing for at least some transition of textbook publishing over to the web and begin to equip the current standards with means to tackle features such as multiple question type exams, interactive videos and shareable notes.
These are just a few examples. If using responsive web design is added to this there is a distribution technology that is reachable and if the standards are kept open and are freely available the benefits for learners and educators are clear.

7. Future Work

One idea for future work is to develop an open and free standard for web published textbooks(WebBook). This would include a separate data model for book content that could manage both interactive and multimedia features supported by HTML5 and related technologies or some descendants thereof.
This data model would support the structure and constraints necessary for a book to have. This model could be based on XML(DocBook for that matter), JSon or any other model that the book in question could be verified by.

An open source web software to write WebBooks is another idea. This software would support the author(s) in two main ways:

a) creating the data model automatically.
b) Providing visual aids for the many display configurations supported.

The third idea would be to develop an open source and free web framework for displaying and managing eBook content. This could framework would support the book presentation in different cultures and for different learning demands and specifications.

A fourth idea is a development of a structured way of transforming books and eBooks of various formats to the web and enhance them with interactive and multimedia content.

Research should be carried out in the following:

Investigating business models for web published textbooks.
Usage of eLearning material available as a web published textbooks.
Evaluate the learning gain from solutions such as suggested here.
The existence or disappearance of DRM in textbooks published on the web.

References

Beer, W., Wagner A.,2011.Smart books: adding context-awareness and interaction to electronic books. In MoMM '11: Proceedings of the 9th International Conference on Advances in Mobile Computing and Multimedia, Hue City, Viet Nam — December 05 – 08, pp 218 – 222.

Fenwick Jr, J.B., Phillips, R. , Kurtz, B.L.,Weidner, A. , Meznar, P. 2013 Developing a Highly Interactive eBook for CS Instruction. In SIGCSE '13: Proceeding of the 44th ACM technical symposium on Computer Science Education, Denver, CO, USA — March 06 – 09, pp 135-140.

Gregorgiev, K., Matelan, N., Pandeff, L.,Willis, H. 2011 Sophie 2.0 and HTML5: DIY Publishing to Mobile Devices. In ELPUB2011. Digital Publishing and Mobile Technologies, 15th International Conference on Electronic Publishing, Istanbul Turkey, June 22-24, pp. 20-27.

Intellectual Property Rights (IPR) Policy | OASIS. Available from <https://www.oasis-open.org/policies-guidelines/ipr>. [22 November 2015]

Luppicini, R., Haghi, A.K. 2010. *Cases on Digital Technologies in Higher Education: Issues and challenges.* Information Science Reference Hershey New York USA.

Lee, K. H., Guttenberg, N., McCrary, V. 2002. Standardization aspects of eBook content formats. *Elsevier Computer Standards & Interfaces*, Volume 24, Issue 3, July, pp 227 – 239.

OASIS DocBook TC | OASIS. Available from <https://www.oasis- 0pen.org/committees /tc_home.php?wg_abbrev=docbook>. [22 November 2015]

Pettenati, C. 2001. Electronic publishing at the end of 2001. In Advanced Technology and Particle Physics, Proceedings of the 7th International Conference on ICATPP-7, Como Italy, October 15 – 19.

Reitz, J. 2004. *Dictionary for Library and Information Science*. Libraries Unlimited Santa Barbara, California U.S.A.

Shaffer, C. A. 2012. Active eTextbooks for CS: what should they be? In SIGCSE '12 Proceedings of the 43rd ACM technical symposium on Computer Science Education, Raleigh NC U.S.A., February 29 – March 3, pp 680–680.

Shaffer, C.A., Karavitra, V., Korhonen, A., Naps, T.L. 2011. OpenDSA: beginning a community active-eBook project. In Koli Calling '11: Proceedings of the 11th Koli Calling International Conference on Computing Education Research, TBA, Finland, November 17 - 20, pp 112-117.

Shen, W., Koch, U. 2011. eBooks in the Cloud: Desirable Features and Current Challenges for a Cloud-based Academic eBook Infrastructure. In ELPUB2011. Digital Publishing and Mobile Technologies, 15th International Conference on Electronic Publishing, Istanbul, Turkey, June 22-24, pp. 80-86.

Shipman, J.W. 2011. *Relax NG Compact Syntax(RNC),* New Mexico Tech Computer Center NM USA.

Standards | OASIS. Available from: <https://www.oasis-open.org/standards#dbv5.0>. [22 November 2015]

Strayton, B. 2007. *DocBook XSL The Complete Guide*. 4rd edition, Sagehill Enterprises Santa Cruz CA USA.

The DocBook Schema. Available from <http://docs.oasis-open.org/docbook/specs/docbook-5.0-spec-os.html>. [22 November 2015]

The DocBook Schema. Available from <http://docs.oasis-open.org/docbook/specs/docbook-5.0-spec-os.html#normative.refs>. [22 November 2015]

Walsh, N. 2010. *DocBook 5 The Definitive Guide.* O'Reilly Media Inc Sebastopol CA USA.

Wang, T., Towey, D. 2012. A Rethinking of Digital Learning Device Projects. In IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE2012), Hong Kong, August 20 – 23, pp W2C-1 – W2C-5.

Živković, D. 2008 The Electronic Book: Evolution or Revolution?. *Bilgi Dünyası*, Vol. 9, No. 1. (2008), pp. 1-20.

Appendix A – User scenarios

User Scenario 1: Author creates a question.
The scenario takes place at an editor that is used to create an electronic textbook to be published on the web. The author has opted to create a new question. As the author works through the task at hand he/she offers explanation and knowledge of the system.

Even though a single choice question and a multiple choice question have a lot in common the system starts by asking the author which type he/she wants to create:
• The author opts to create a single choice question:
  • The first step for the author is to connect the question to a certain material(s) in the eTextbook. This step is optional but the question can be tied to multiple parts of the book.
  • As the next step, the author does is to add the question text itself. Then comes the decision on how many answer possibilities there should be.
  • When the author has chosen how many answers there are, the system opens up three input fields for each question.
  • In the first field an answer is created. This is a mandatory input.
  • In the second field a response text can be added that is relative to the answer given.
  • In the third field(radio button) the author marks if this specific answer is correct. The system checks this option since one and only must be marked.
  • Having added all the answers the author has an option of grading the question's overall difficulty. The difficulty varies (in numbers) from 1 – 10. This is not a mandatory action.
  • The last thing the author does before creating the question is to add his/her name to it.
  • Doing so the author clicks on *create Question* and the new question is created. Since it is possible to add a new question without linking it to a certain quiz the system keeps track of which option is being used and adds the necessary connection.
• The author opts to create a multiple choice question:
  • This is almost the same as creating a single choice question. The difference is that instead of a mandatory one single answer to be correct, and all the others to be incorrect, such limits are not posed here. All answers can be correct but at least one is minimum. All the author has to do is to mark the correct answers. Should however the author want all options to be incorrect he has to do that using the answer's text and mark that option as correct!
  • After this the author supplies his/her name to the question.
  • Doing so the author clicks on *create Question* and the new question is created. Since it is possible to add a new question without linking it to a certain quiz the system keeps track of which option is being used and adds the necessary connection.

User Scenario 2: Author adds a Quiz to a textbook chapter

The scenario takes place at an editor that is used to create an electronic textbook to be published on the web. The author has opted to create a new question. As the author works through the task at hand he/she offers explanation and knowledge of the system.

The author locates a correct place for the quiz within the chapter and then chooses an *add Quiz* option and the system opens a quiz window that encapsulates all necessary functions and options for managing questions and quizzes.

- The next thing the author does is to give the quiz a title. After that it is possible to write a (short) description of the quiz being created.
- Before the creation/insertion of questions starts the author can choose to put a weight in percentage on this particular quiz. This is done if the author plans to use the quiz is a part of a group of quizzes that provide a collective grade through all the chapters of the book being written.
- Having finished these three tasks the author indicates how many questions he or she will be using. The system prompts with a list of the necessary editing options for each individual question with their newly assigned ID visible for the orientation of the author. For each question the author is asked if a new question should be created or if an existing question shall be used:

- The author opts for creating a new question
    - The system opens up a *new Question* option(Scenario no 3)
    - If the author wishes a weight in percentages can be added here. This percentage is valid only within the quiz being created.
- The author opts to use an existing question
    - The system opens up a question overview window for the author to use.
    - The author locates the question he wants to use and clicks on *add to Quiz* button.
    - The system prompts the author if he/she wishes to add weight to the question.

- Having added all the question to the quiz the author can opt for adding a response(text) if the reader has failed the quiz. This is done by the author adding a "fail" percentage and creating a response to that failure. This system supports only response to a fail grade but lacks the ability to create response to various degrees of success(in percentages).
- At last the author adds his / her name to the quiz.

Having undertaken these tasks described, the author adds the quiz itself to the chapter by clicking on *save Quiz* button.

It should be noted here that the questions are displayed in the order they are added to the quiz.

User Scenario 3: Author adds a video to a textbook chapter

The scenario takes place at an editor that is used to create an electronic textbook to be published on the web. The author has opted to create a new question. As the author works through the task at hand he/she offers explanation and knowledge of the system.

The author seeks the desired location(paragraph) within the chapter and indicates the insertion of a video by adding a necessary marker for the video placement. The system then responds by displaying the appropriate options to the author.
First the video is given an ID by either the system or by the author. This identifies the video to be within the corresponding chapter. After the ID has been set the author adds a title to the video that is intended to offer some information on the content it self. There is however no restriction as to what is set as the title but rather it is an option that the author can choose to use or not to use.
If extra textual info is to be provided outside of the title the author can use the caption to do so. Like with the title this is optional. The next inputs from the author relate to the video file it self:

- The author is asked to supply the file type of the video. The author selects the correct file type from a list of supported system file types.
- Then the author is asked to supply the video file size.
- Then the author supplies the running time of the video. This is mandatory.
- The last meta data the author is asked to provide is the path to the file itself. This path is then used by the system that transforms the book data into a usable format.

Having finished supplying the file meta data the author can now provide a substitute information that can be displayed in case the video itself is not displayable. The author chooses between a textual information and an image that lead to two different paths of editing:

- The author chooses text:
    - The system displays an editor window for the creation text messages. Visible in this editor is the video ID that was given at the start of the process. This tells the author that this message will be exclusively linked to this video at this location within this chapter. The author creates the message and clicks on a save button.
- The author chooses an image:
    The system displays an image adding window from which the author chooses the specified image file. As is the case with the text option the video ID is visible to the author having the same meaning. Then the author supplies a name to the image and a caption. Both are optional and in the event of the author not opting to use those, the file is given a name by the system. The author finishes the insertion of the image by providing this necessary information and finally clicks a save button.

The last task the author has to complete is to set the level of interaction that the chosen video is supposed to support. There are three interaction levels supported: *zero*, *mixed* and *internal*. These levels indicate how the author intends the reader to interact while the video is playing. The level can be broken down in the following manner:

- The interaction level is *zero*:
    - If the author has intended the video to be without any interaction while it's running then this is the option. No further actions from the author regarding the interactivity are needed.
- The interaction level is *mixed*:
    - If the author has chosen this option a new input window opens giving the author access to the required editing features. When using a mixed interaction level the author intends the video to be paused at certain times and the reader is given a “task” to finish before continuing watching the video. The first step for the author is to provide the stop times in a timely order. This is done before any further steps are taken.
    - For each stop time the system assigns a chronologically unique ID and the system checks if any given time exceeds the supplied running time of the video, rejecting it if necessary. When all the stop times are in place the author proceeds to adding “assignments” to each stop.

        - In this version only two kinds of assignments are allowed. They are in the form of a textual assignment description or a single question. The Question can be multiple choice or a single choice.
        - The author opts for the textual description:
            - An editor opens for the author to type in his description of the assignment. The video ID and the stop time are clearly visible to indicate where the author working. There are no other steps to be taken at this stage since there is no evaluation of the readers response. The user simply clicks an OK button and the assignment is connected to the right stop time.
        - The author opts for a multiple choice question:
            - An editor opens for the inclusion of the question. The author can choose between opening an existing question or create a new one. If an existing question is used the author selects it, clicks OK and the question is linked to this stop time assignment. Otherwise the author fills in the necessary information in a question editor(see the process user scenario 2), clicks on OK and the new question is linked to the assignment.
        - The author opts for a single choice question:
            - The process is identical to the process described above.
    - The interaction level is *internal*:
        - The video itself contains interaction with the reader. No further action needs to be taken by the author.

The author has now successfully completes the necessary steps of inserting a video into a eTextBook. This is indicated by the system that offers a preview of the inserted video. The author can accept or decline and clicks the finish button. The video is now a part of the designated chapter.

Appendix B – Scenario Analysis

Analysis of scenario 1:

Number	1
Action	The author seeks the desired location(paragraph) within the chapter
Prerequisites	A chapter has been created and at least one paragraph exists
Constraints	Only one paragraph(ID) can be located each time.
Expected result	The ID of the chosen paragraph has been made “available”
Notes	Since a paragraph ID is only unique to a chapter the chapter ID must be known.

Number	2
Action	The author indicates the insertion of a video(links a video to the specific location).
Prerequisites	Video exists(ID). Location(ID) exists
Constraints	
Expected result	A video file has been linked to a paragraph ID
Notes	The added “extras” to each paragraph should probably get an order number 1 = first etc.

Number	3
Action	author adds a title to the video
Prerequisites	The video ID is available
Constraints	Only one title can be given to a video in a paragraph.
Expected result	The video “inserted” into the paragraph has now both ID and a name
Notes	The name of the video probably belongs to the video and not the video in the paragraph

Number	4
Action	author sets the caption that is to be used with the video
Prerequisites	The video ID is available
Constraints	Only one caption can be set for a single video in a paragraph(location)
Expected result	The video has a caption that relates to the location.
Notes	The caption can be different for the same video if it's used in more than one place.

Number	5
Action	The author supplies the file type of the video
Prerequisites	The video ID is available and the video file itself is accessible
Constraints	A video can have only one file type
Expected result	As a part of a metadata the file type has now been made available
Notes	This data belongs to the video irrespective of where it's used

Number	6
Action	Then the author is asked to supply the video file size.
Prerequisites	The video ID is available and the video file itself is accessible
Constraints	A video file has one size only
Expected result	As a part of a metadata the file size has now been made available
Notes	This data belongs to the video irrespective of where it's used

Number	7
Action	The author supplies the running time of the video.
Prerequisites	The video ID is available and the video file itself is accessible
Constraints	A video file has only one running time
Expected result	As a part of a metadata the running time of the video has now been made available
Notes	This data belongs to the video irrespective of where it's used

Number	8
Action	The author provides the path to the video file itself
Prerequisites	The video file exists
Constraints	The video file van only be in one location (unique path)
Expected result	A video file has been linked to the video metadata and the video ID
Notes	This data belongs to the video irrespective of where it's used

Number	9
Action	The author chooses text as a video substitute.
Prerequisites	The video ID is available as well as the paragraph ID.
Constraints	The substitute can be: a) text b) image.
Expected result	Textual info has been supplied as a substitute for the video file.
Notes	This is most likely a fail save measure geared towards the presentation layer.

Number	10
Action	The author chooses an image as a video substitute.
Prerequisites	The video ID is available as well as the paragraph ID.
Constraints	The substitute can be: a) text. b) image.
Expected result	An image has been supplied as a substitute for the video file.
Notes	This is also most likely a fail save measure geared towards the presentation layer.

Number	11
Action	The author supplies a name to the image.
Prerequisites	An image ID is available.
Constraints	Only a single name is allowed to each individual image.
Expected result	The image in question has a name(title).
Notes	The name is probably unique to the image itself irrespective of it's usage.

Number	12
Action	The author supplies a caption to the image.
Prerequisites	An image ID is available.
Constraints	Only a single caption is allowed to each individual image within each usage location.
Expected result	The image in question has a caption.
Notes	This caption can vary relative to where it's being used.

Number	13
Action	The author sets the level of interaction for a video.
Prerequisites	The video ID is available.
Constraints	Three levels of interaction can be set: none, mixed, internal.
Expected result	An interaction level has been set for this video and can be used.
Notes	The mixed option of the interaction is the one that needs further analysis.

Number	14
Action	The author provides stop times for the video.
Prerequisites	The video ID is available. Interactive option is: mixed.
Constraints	Each assigned stop time must be unique. No time must exceed the running time of the video.
Expected result	The system has now registered the intended stop times for the video in mixed interactive mode.
Notes	This involves that the video is to stop at certain times in the narrative. This can probably be done in two ways: manually and automatically.
Belongs to	13

Number	15
Action	The author adds text description to a stop time.
Prerequisites	A stop time is available.
Constraints	An (interactive) assignment can be: a) a text description. b) a single question (single or multiple choice)
Expected result	The assignment has been described and is connected to the relative stop time.
Notes	The reader has the responsibility to resume with the video. No time constraints on the assignment are imposed.
Belongs to	13

Number	16
Action	The author adds a single question to a stop time.
Prerequisites	A stop time is available.
Constraints	An (interactive) assignment can be: a) a text description. b) a single question (single or multiple choice).
Expected result	The assignment has been set as a single question and is connected to the relative stop time.
Notes	The reader has the responsibility to resume with the video. No time constraints on the assignment are imposed.
Belongs to	13

Number	17
Action	The author seeks the desired location(paragraph) within the chapter.
Prerequisites	A chapter has been created and at least one paragraph exists.
Constraints	Only one paragraph(ID) can be located each time.
Expected result	The ID of the chosen paragraph has been made “available”.
Notes	Since a paragraph ID is only unique to a chapter the chapter ID must be known.

Number	18
Action	The author indicates the insertion of a quiz at the selected location.
Prerequisites	Chapter(ID) exists. Location(ID) exists.
Constraints	There can be only one quiz per paragraph.
Expected result	A quiz has been linked to a paragraph ID.
Notes	If there are more than one quiz in a chapter each quiz should be embedded a paragraph

Number	19
Action	The author gives quiz a title.
Prerequisites	Quiz(ID) exists.
Constraints	One quiz can have only one title.
Expected result	The quiz in question has a title.
Notes	Title of a quiz should probably belong to the quiz itself rather than a quiz in a paragraph.

Number	20
Action	The author writes a description of the quiz.
Prerequisites	Quiz(ID) exists.
Constraints	Only one description is allowed in each quiz.
Expected result	The quiz in question has a description.
Notes	The description of a quiz should probably belong to the quiz itself.

Number	21
Action	The author puts a weight(in percentage) on a quiz.
Prerequisites	Quiz(ID) exists. Chapter(ID) exists.
Constraints	The quiz's weight must be an integer between 1 and 100 inclusive. One quiz = one weight.
Expected result	The quiz now has a weight in percentages.
Notes	None at this time (23.07.2013 12:17:15).

Number	22
Action	The author selects the number of questions in the quiz.
Prerequisites	Quiz(ID) exists.
Constraints	The number entered is a positive integer.
Expected result	The number of questions for a particular quiz has been determined.
Notes	This is some sort of metadata for the quiz.

Number	23
Action	The author creates a new question.
Prerequisites	Quiz(ID) exists.
Constraints	
Expected result	A new question is added to a question database and to the “working” quiz.
Notes	See also: 29 -

Number	24
Action	The author locates a question
Prerequisites	The question(ID) exists
Constraints	
Expected result	The ID of the respective question is available for usage
Notes	This involves querying a database of questions already in the “system” for reuse purposes. As in 25 the author can add weight to this question as it is within a quiz.

Number	25
Action	The author adds a weight in percentages to a question
Prerequisites	Quiz(ID) exists. Question(ID) exists
Constraints	The weight is an integer between 1 and 100 inclusive.
Expected result	A question has been added a weight in the quiz context
Notes	This percentage given to a question is valid only within the quiz being created.

Number	26
Action	The author adds a failed grade limit.
Prerequisites	Quiz(ID) exists.
Constraints	Only one fail grade can be set for each quiz. The grade is an integer between 1 and 100(percentage)
Expected result	The quiz contains the intended fail threshold as intended by the author
Notes	No further constraints can be imposed on the percentage an author can input.

Number	27
Action	The author adds a response(text)
Prerequisites	Quiz(ID) exists. Fail grade has been set.
Constraints	The response can only be a text and only one response per quiz
Expected result	Connected to the failing grade the quiz incorporates a response to quiz failure
Notes	May be implemented as quiz completion response at some later stage

Number	28
Action	The author adds author name to quiz
Prerequisites	Quiz(ID) exists.
Constraints	A quiz can have multiple authors
Expected result	Author(s) of a particular quiz known
Notes	Only the author(s) name is intended here. No further info about him/them.

Number	29
Action	The author chooses a single choice question:
Prerequisites	None
Constraints	The question gets a unique identifier (ID)
Expected result	A new single choice question is now ready to be constructed.
Notes	This is basically the same as number 38 and serves the purpose to emphasize the type of the question.

Number	30
Action	The author connects question to specific book content(material) elements.
Prerequisites	The question(ID) exists.
Constraints	The content connector can be books, chapters, paragraphs, images, videos. One question can be connected to multiple content.
Expected result	The question has been linked to a chosen content elements
Notes	The question of link to multiple books needs to be addressed during iteration the process.

Number	31
Action	The author adds question text.
Prerequisites	The question(ID) exists.
Constraints	Only one version of the text can exist per question
Expected result	The question contains textual description of the problem
Notes	Variations of the question text are not allowed. They might be added later.

Number	32
Action	The author adds number of answers for a chosen question.
Prerequisites	The question(ID) exists.
Constraints	The number must be a positive integer.
Expected result	The question contains a number indicating how many answer(possibilities) there exist.
Notes	Maybe only relevant to the editor used to create the question. To be looked at during iteration.

Number	33
Action	The author adds answer to chosen question.
Prerequisites	The question(ID) exists.
Constraints	
Expected result	An answer has been added to a chosen question. An ID has been created for the answer
Notes	As not to defy logical thinking a minimum of two answers is set as mandatory minimum.

Number	34
Action	The author adds a response text to answer
Prerequisites	The answer(ID) exists.
Constraints	Only one response per answer is allowed
Expected result	The chosen answer contains a response text
Notes	The response is mainly to be used should user choose the relevant answer to a question.

Number	35
Action	The author marks the correctness of a chosen answer.
Prerequisites	The answer(ID) exists.
Constraints	Correctness can have only two values: true = answer is correct. false = answer is incorrect. Only one answer can be correct.
Expected result	The answer now “knows” it's correctness.
Notes	33, 34 and 35 are repeated for the number of answers the author selected in 32

Number	36
Action	The author adds the difficulty level of a question.
Prerequisites	The question(ID) exists.
Constraints	Level is a positive integer between 1 and 10
Expected result	Question contains difficulty level
Notes	This is an estimated level by the author and is open to interpretation

Number	37
Action	The author adds author name to question.
Prerequisites	The question(ID) exists.
Constraints	There can be multiple authors of a single question.
Expected result	The question contains the names of all authors
Notes	None

Number	38
Action	The author creates a multiple choice question:
Prerequisites	None
Constraints	The question gets a unique identifier (ID)
Expected result	A new multiple choice question is now ready to be constructed.
Notes	This is the same as number 29 and serves the purpose to emphasize the type of the question.

Number	39
Action	The author connects question to specific book content(material) elements.
Prerequisites	The question(ID) exists.
Constraints	The content connector can be books, chapters, paragraphs, images, videos. One question can be connected to multiple content.
Expected result	The question has been linked to a chosen content elements
Notes	The question of link to multiple books needs to be addressed during iteration the process.

Number	40
Action	The author adds question text.
Prerequisites	The question(ID) exists.
Constraints	Only one version of the text can exist per question
Expected result	The question contains textual description of the problem
Notes	Variations of the question text are not allowed. They might be added later.

Number	41
Action	The author adds number of answers for a chosen question.
Prerequisites	The question(ID) exists.
Constraints	The number must be a positive integer.
Expected result	The question contains a number indicating how many answer(possibilities) there exist.
Notes	Maybe only relevant to the editor used to create the question. To be looked at during iteration.

Number	42
Action	The author adds answer to chosen question.
Prerequisites	The question(ID) exists.
Constraints	
Expected result	An answer has been added to a chosen question. An ID has been created for the answer
Notes	As not to defy logical thinking a minimum of two answers is set as mandatory minimum.

Number	43
Action	The author marks the correctness of a chosen answer.
Prerequisites	The answer(ID) exists.
Constraints	Correctness can have only two values: true = answer is correct. false = answer is incorrect. Multiple true / false answers can exist
Expected result	The answer now “knows” it's correctness.
Notes	

Number	44
Action	The author adds a response text to question(multiple choice)
Prerequisites	The question(ID) exists. The question is of type multiple choice.
Constraints	The author supplies zero or three responses to each multiple choice question. Each response is of the type: a) correct. b) mixed. c) incorrect.
Expected result	The question now contains a threefold response to match possible user choices.
Notes	In multiple choice mode the response is not tied to a single answer but three combinations are identified and used.

Number	45
Action	The author adds the difficulty level of a question.
Prerequisites	The question(ID) exists.
Constraints	Level is a positive integer between 1 and 10
Expected result	Question contains difficulty level
Notes	This is an estimated level by the author and is open to interpretation

Number	46
Action	The author adds author name to question.
Prerequisites	The question(ID) exists.
Constraints	There can be multiple authors of a single question.
Expected result	The question contains the names of all authors
Notes	None

Number	47
Action	The system adds unique identifiers
Prerequisites	An entity needs to be assigned an identifier.
Constraints	Unique identifiers are positive integers
Expected result	The respective entity is now represented with a ID
Notes	The DocBook extension does not care who does it as long as an ID is created.

Appendix C – Requirement list

Requirements			Preliminaries				
NO	**Requirement**	**RN**	**Type**	**Rule**	**Name**	**TagRef**	**Datatype**
1	Chapter has a unique identifier	1	A	NO	id	<chapter>	integer
2	Paragraph has a unique identifier	1	A	NO	Id	<p>	integer
3	Video is an element	2	E	NO	<video>	<chapter><para>	mixed
4	Video has a unique identifier	2	A	NO	Id	<video>	integer
5	Child elements of chapter can have order	2	A	NO	order	multi	integer
6	Video has title element	3	E	NO	<title>	<video>	element
7	Video has caption element	4	E	NO	<caption>	<video>	element
8	Video has a file type	5	A	NO	type	<video>	string
9	Video has file size	6	A	NO	size	<video>	float
10	Video has a metadata element	6	E	NO	<metadata>	<video>	mixed
11	Video has a running time	7	A	NO	time	<video>	float
12	Video has a file path	8	A	NO	path	<video>	string
13	Video has a substitute element	9	E	NO	<substitute>	<video>	mixed
14	Substitute has a text element	9	E	NO	<text>	<substitute>	string
15	Substitute has an image element	10	E	NO	<image>	<substitute>	image
16	Image has name	11	E	NO	<name>	<image>	string
17	Image has unique identifier	11	A	NO	id	<image>	integer
18	Image has a caption	12	E	NO	<caption>	<image>	string
19	Video has interactivity level	13	A	NO	<interaction>	<video>	string
20	Video has stop times	14	E	YES	<break>	<video>	mixed
21	Break has times	14	E	YES	<stopat>	<break>	time
22	Break has assignment(task)	15	E	NO	<task>	<break>	mixed
23	Assignment(task) has description	15	E	NO	<description>	<task>	string
24	Assignment(task) has question	16	E	NO	<question>	<task>	mixed
25	Quiz is an element	18	E	NO	<quiz>	<chapter>	mixed
26	Quiz has unique identifier	18	A	NO	Id	<quiz>	integer
27	Quiz has a title	19	E	NO	<title>	<quiz>	string
28	Quiz has description	20	E	NO	<description>	<quiz>	string
29	Quiz has weight in percentages	21	A	NO	weight	<quiz>	integer
30	Quiz has question count	22	A	NO	numquest	<quiz>	integer
31	Quiz has questions element	22	E	NO	<questions>	<quiz>	mixed
32	Question in a quiz has a weight	25	A	NO	weight	<question>	mixed
33	Quiz has fail grade	26	A	NO	fgrade	<quiz>	integer
34	Quiz has response element	27	E	YES	<response>	<quiz>	string
35	Quiz has authors element	28	E	NO	<authors>	<quiz>	mixed
36	Authors element has author elements	28	E	NO	<author>	<authors>	string
37	Question is an element	29	E	NO	<question>	<chapter>	mixed
38	Question has type	29	A	NO	type	<question>	string

39	Question has unique identifier	29	A	NO	Id	<question>	integer
40	Question has text	31	E	NO	<text>	<question>	string
41	Question has answer count	32	A	NO	numans	<question>	integer
42	Question has answers	33	E	NO	<answers>	<question>	mixed
43	Question has answer	33	E	NO	<answer>	<answers>	string
44	Answer has unique identifier	33	A	NO	id	<answer>	integer
45	Answer has response	34	E	NO	<response>	<answer>	string
46	Answer has correctness	35	A	NO	correct	<answer>	boolean
47	Question has difficulty level	36	A	NO	level	<question>	integer
48	Question has authors	37	E	NO	<authors>	<question>	mixed
49	Question has author	37	E	NO	<author>	<authors>	string
50	Question has response	44	E	YES	<response>	<question>	string
A1	Answer has text	*34*	E	NO	<text>	<answer>	string

NO: Number of the requirement.

Requirement: What is the intended use of this requirement.

RN: Reference Number. Refers to the use case the requirement “belongs” to.

Type: Element or attribute.

Rule: Indicates if developing a further rule to this requirement might be beneficiary.

Name: Name of element or attribute.

TagRef: Intended parent elements(s).

Datatype: What sort of data should be used.

Appendix D – Candidate DocBook elements

Video element DocBook 5.0 candidates		
<videodata>	Used as a pointer to an external video file	Relevance through name
<videoobject>	Wraps <videodata> and associates	Relevance through name
<mediaobject>	Contains set of alternative media objects	Relevance through <videoobject>
<inlinemediaobject>	Contains set of alternative graphical objects	Relevance through <videoobject>

Question element DocBook 5.0 candidates		
<question>	States a problem answered by <answer>	Relevance through name
<qandaentry>	Wraps <videodata> and associates	Relevance through <question>
<qandadiv>	A section of <qandaset>	Relevance through <qandaset>
<qandaset>	List consisting of questions and answers	Relevance through context
<answer>	Optional within <qandaset><question>	Relevance through name /context

Quiz element DocBook 5.0 candidates
None found

Appendix E – Selection process arguments

Currently in use	The fact that a format or standard is currently in use can imply some level of publishing success and therefore if it is an open format can be studied if needed.
Open standard and/or format	Preferably there should be an open standard or format used in the publishing process. The need for openness here is to allow for a relatively direct access to the underlying specification.
Used in textbook publishing	The focus of this work centers around textbook publishing on the web and it could substantially aid the selection process if the standard/format to be studied would be used in publications of textbooks.
Development and maintenance	The existence of a community or vendor that is supporting a format or standard by development and maintenance indicates that there will not be an eminent depreciation.
Incremental versioning	A simple incremental versioning is an important support for positioning the work through the appropriate choice of version. If many “spinoff” versions are available it complicates the selection and can even result in a selection error.
Publication formats	Should the same format or standard be used to publish in more than one format(i.e books on paper, eBooks) it could give an indication that conversion to other formats is possible indicating strength and flexibility.
DRM	The format, standard should preferably not be required to include or embed A Digital Rights Management code of any kind as it might lead to complications of the inspection itself.

Appendix F – Code examples

Listing e.1: Example of the two RELAX NG syntax formats.

XML format	*Compact format*
<oneOrMore> <element name="author"> <attribute name="id"/> <element name="name"> <text/> </element> <optional> <element name="born"> <text/> </element> </optional> <optional> <element name="died"> <text/> </element> </optional> </element> </oneOrMore>	element author { attribute id { text }, element name { text }, element born { text }?, element died { text }? }+

Listing e.2: A named pattern “common” declared and used in two different elements

```
common =
  attribute id { text },
  element name { text },
  element email { text }?

element admin {
  common-content,
  element access { xsd:integer}?
}

element user {
  common-content,
  element department { text }?
}
```

Listing e-3: A rule on numbered being set by Schematron (based on Walsh(2011, p. 64-65)).

```
namespace db = "http://docbook.org/ns/docbook"
namespace s = "http://www.ascc.net/xml/schematron"

default namespace = "http://docbook.org/ns/docbook"

include "docbook.rnc"{
  db.section =
          [
                  s:pattern [
                          name = "limit depth of sections"
                          s:rule [
                            content = db:section"
                            s:assert [
                              test = "count(ancestor::db:section) < 2"
                                      "Sections can be no more than three levels deep"
                            ]
                          ]
                  ]
          ]
          element section {
            db.section.attlist,
            db.section.info,
            db.section.blocks.or.sections,
            db.navigation.components*
          }
}
```