ADAPTATION OF MATHEMATICAL DOCUMENTS
Exploring Document Structures, Metadata, and Context for the Generation of User-specific Documents

Christine Müller
Computer Science Dept., Jacobs University Bremen, Bremen, Germany
BSgroup Technology Innovation AG, Zürich, Switzerland

Keywords: User-specific adaptation, Mathematics, Contexts, Narrative documents, Content planning.

Abstract: The paper proposes a framework that explores document structures, metadata, and context to adapt mathematical documents on three layers - the content, structure, and presentation layer. On the presentation layer mathematical expressions are rendered according to a user context (defining, e.g., the users' preferred convention, language or discipline). On the content layer, appropriate paragraphs of texts are selected that best suit the users' individual information need as well as narrative and semantic context. On the structure layer, these content parts are arranged appropriately, while taking semantic and narrative interdependencies between parts into account. Technically, the framework models documents as dependency graphs, where nodes correspond to any addressable part of a document and edges denote narrative and semantic dependencies. To allow for the adaptation on all three document layers, these dependency graphs are enriched with presentational, content, and structure variants. Finally, the enriched graphs are processed to generate a user-specific document. Users can guide this adaptation process by prioritising their individual constraints, narrative or semantic context.

1 INTRODUCTION

Information technologies have transformed our present era into an information age, in which individuals can freely transfer information and have instant access to data that was formerly difficult or impossible to access. Today, users are surrounded by an ever increasing number of documents, such as product descriptions or advertisements. Using modern web technologies users can create and exchange such documents but seem to lack time to enhance these documents if they are missing details, to filter redundancies and irrelevant content from these documents or to rewrite documents in new contexts.

Why can computers not relieve us from these burdens? For example, consider user manuals. Besides that hardly any user is willing to read a 1000-page long document, users also have different technical skills. Some need more illustrations and others less explanations and details. Alternatively, think of a product description. Wouldn't it be convenient if we could automatically generate advertisements by simply defining a context, e.g., in terms of price and quality constraints? This work aims at providing such kind of services by adapting documents to a user context. **User context** is a commonly known term that defines a set of context parameters like specific preferences, needs, skills, environmental constraints, etc.

The author claims that adaptation can take place on any document layer. On the *content layer* we can substitute document parts, e.g., to adapt a document to different backgrounds. On the *structure layer* we can rearrange document parts, e.g., to adapt documents to different learning styles. One user might prefer to see illustrations first, while another might like to study abstract concepts and generalizations before that. On the *presentation layer* we can change the appearance of documents without changing content and structure. The presentation of documents is relatively solved. For example, standards like CSS help us to specify the layout of web documents in separation to content and structure. This work thus focuses on **content planning services**, which are adaptation services on the content and structure layer of documents.

The most important prerequisite for adaptation services is a representation of documents, which makes the three document layers comprehensible to a computer system. Such machine-processable representations are developed in the scope of the semantic web, which postulates annotations (or metadata)

Müller C..
ADAPTATION OF MATHEMATICAL DOCUMENTS - Exploring Document Structures, Metadata, and Context for the Generation of User-specific Documents.
DOI: 10.5220/0003065601430148
In *Proceedings of the International Conference on Knowledge Management and Information Sharing* (KMIS-2010), pages 143-148
ISBN: 978-989-8425-30-0

to classify and describe resources. Document formats that embed annotations are called markup languages. This work can be applied to any document that is representable in an **XML-based markup language**[1]. The richer the markup language, the better the adaptation results. However, one also needs effective adaptation algorithms, which are one major contribution of this work.

We can distinguish two paradigms to author documents. The **document-centered approach** (or **narrative approach**) creates documents that are well suited to be read by humans but which are also very hard to adapt. They include multiple cross-references and narrative transitions that improve the coherence of the text: all parts are neatly connected and the narrative flow of the text guides the reader through the writing. However, though transitions and cross-references improve the coherence of documents they also hamper us to reuse parts. This is particularly critical for the content planning during which documents have to be modularised in order to substitute parts and to arrange them according to user preferences.

The **topic-oriented paradigm** is based on principles of reuse and modularization and was originally developed to manage technical writings like manuals, product descriptions or software documentations. Topic-oriented documents consists of self-contained units and can be easily modularised. Their adaptation has been well-researched in, e.g., the eLearning domain[2]. However, topic-oriented documents omit narrative transitions and thus lack coherence.

Neither topic-oriented nor document-centered paradigm lead to an adaptation infrastructure, which supports modularization *and* coherence. To address this challenge, this work proposes to bridge the two paradigms: It starts with one way and applies the topic-oriented principles of reuse and modularization to the narrative world.

To illustrate and evaluate the proposed adaptation services, mathematical documents are used. The understanding of mathematical documents helps us to better model documents from other domains. The author thus expects that her findings can be applied to other domains and a wide range of documents.

Nevertheless, having selected mathematical documents requires us to take another aspect into account: the adaptation of mathematical notations, a service on the presentation layer of documents.

[1](Müller, 2010) analyzes a list of document markup languages, such as DITA and DocBook. Here we use the mathematical document format OMDoc (Kohlhase, 2006).

[2](Müller, 2010) analyzes respective adaptation systems.

2 RENDERING NOTATIONS

Mathematics is a mixture of natural language text, symbols, and formulae. Symbols and formulae can be presented with different notations, which underlie cultural conventions and individual preferences. For example, the notations for the binomial coefficient vary in different language: C_n^k is used in French/Russian and $\binom{n}{k}$ in German/English. Alternatively, i is used in mathematics to denote the imaginary unit, while j is used in physics to avoid confusion with the notation I for electronic current. When planning content and structure of documents, notations have to be adapted as well. In particular when applying content planning to multi-authored document collections, adaptation of notations becomes a central issues and helps to avoid notational inconsistencies.

To adapt notations we need a machine-processable representation of mathematical expressions and their notations. Fortunately, we can build on two widely-used standards OpenMath (Buswell et al., 2004) and MathML (Ausbrooks et al., 2008). They provide a markup of the functional structure of mathematical expressions (called **content markup**) and a markup of the two-dimensional layout of notations (called **presentation markup**).

```
<OMA>
 <OMS cd="combinat"
  name="binomial"/>
 <OMV name="n"/>
 <OMV name="k"/>
</OMA>
```

```
<mrow>
 <mo fence="true">(</mo>
 <mfrac linethickness="0">
  <mi>n</mi>
  <mi>k</mi>
 </mfrac>
 <mo fence="true">)</mo>
</mrow>
```

Figure 1: Content/Presentation Markup for Binomial Coeff.

Figure 1 illustrates the respective markups for the binomial coefficient. The content markup to the left specifies the meaning of the expression. The presentation markup to the right the layout of the German notation, which can be interpreted by most web browsers to display $\binom{n}{k}$.

But how do we get from content markup to presentation markup? There has been a long tradition in mathematical knowledge management to support this transformation. For example, we can hard-code the transformation process, which has widely been done[3]. Alternatively, we can encode the notation practice of mathematicians and use it to guide the transformation. This work builds on the specification of **notation definitions** as proposed by (Kohlhase et al., 2008). Notation definitions can be seen as a declarative specifi-

[3](Müller, 2010) analyzes such transformation workflows in proof assistants.

cation of rules that bridge the translation from content markup to presentation markup. The author's contribution is an extension of the initial rendering workflow that allows us to deal with different contexts to, e.g., select appropriate notations for specific languages or disciplines. Dealing with such contexts is the focus of this section, details on how notation definitions work are omitted (Kohlhase et al., 2008).

```
<notation>
 <prototype>
  <om:OMA><om:OMS cd="combinat1" name="binomial" />
  <expr name="arg1"/><expr name="arg2"/></om:OMA>
 </prototype>
 <rendering ic="language:German">
  <m:mrow><m:mo>(</m:mo><m:mfrac linethickness="0">
  <render name="arg1"/><render name="arg2"/>
  </m:mfrac><m:mo>)</m:mo></m:mrow>
 </rendering>
 <rendering ic="language:French">
 <m:msubsup><m:mi>C</m:mi>
  <render name="arg1"/>
  <render name="arg2"/></m:msubsup>
 </rendering>
</notation>
```

Figure 2: XML-Encoding of Transformation rules.

Notation definitions consist of **prototypes** (patterns that are matched against the content markup of expressions) and **renderings** (that are used to construct corresponding presentation markup). Figure 2 presents a notation definition, which prototype matches with the content markup for the binomial coefficient. It includes two renderings. The first one generates the German notation $\binom{n}{k}$ and the second the the French notation C_n^k.

To support a context-aware selection between these two renderings, the author proposes a context model with two components: an **extensional context** and an **intensional context** specification. The extensional context consists of references to documents, notation definitions, and rendering elements, which are resolved to collect notation definitions from various documents. The intensional context is used to guide the adaptation intensionally by providing a set of context parameters. These are matched against the metadata of renderings to prioritize these renderings. In our example, an intensional context is encoded as `ic` attribute. It defines that the content markup should be transformed into a German notation, thus, the first rendering element is selected.

The rendering workflow for the adaptation of mathematical documents has five core components. The *renderer* receives a document (*doc*), a document database (*db*), an extensional context (ec^*), and an intensional context (ic^*). The contexts encode how the user wants to guide the rendering workflow, the database includes all documents referenced by the user, the document includes content markup for all expressions. For each content expression (*expr*) in the document, the renderer first calls the *notation collector* to resolve the extensional context references (ec^*). The notation collector returns the list of notation definitions (ntn^*) that the users wants to consider during the adaptation. The renderer then calls the *context collector* to resolve the intensional context parameters (ic^*) that describe how the expression should be rendered. The context collector returns the effective intensional context (*ic*) for the expression. Finally, the *pattern matcher* is called to select an appropriate rendering (*rnd*). It first matches the content markup with the collected notation definitions (ntn^*) and then passes the rendering elements (rnd^*) from these notation definitions to the *rendering grabber*. The rendering grabber uses the intensional context options (*ic*) to rank the renderings. The first one (*rnd*) is used to generate the presentation markup for the content expression (*expr*). The renderer returns a document (doc_Δ), in which each content markup expression is replaced with a user-preferred presentation markup.

The context-sensitive extension of the initial rendering algorithm has been implemented as central part of the JOMDOC library, the reference implementation of the OMDOC format (JOMDoc, 2010). Having specified and implemented the initial framework, several services can now be provided. Partially these have been implemented by other research projects. For example, the explanation of a formula's structure is supported by the JOBAD project that offers flexible elision and folding of sub-terms (Giceva et al., 2009). The change of notations on-the-fly while reading a document is demonstrated by the document reader *panta rhei* (Müller, 2010).

Several issues remain for the future. One of them is a thorough evaluation whether (and when) adaptation of notations is beneficial. After all authors spend much time and consideration to select appropriate notations. Moreover, when adapting documents with notations we also need to plan its content, which brings us back to the initial intention of this work. The following section extends the notation framework for the content planning of documents. We will see that we can reuse the specification of extensional and intensional contexts as well as the core adaptation mechanisms.

3 CONTENT PLANNING

The previous section introduced the notation service, a service on the presentation layer of mathematical documents. This section focuses on two content planning services: The *substitution* replaces content with alternative, user-specific content (a service on the content layer) and the *reordering* rearranges content (a service on the structure layer). Both services require the adaptability of structure and content layer. This is supported by enriching content with alternative/variant content, by separating reusable from non-reusable content, and by enriching structure with narrative variations.

3.1 Variation on the Content Layer

This work defines two document parts as **variants** if they are interchangeable though different in specific properties. Some relations between document parts like 'translates' or 'formalizes' indicate **variant relations**: They express an *equivalence* and a *difference* between two parts. Other relations like 'more difficult than' solely express a difference. Document parts related in such a relations can become variants if they are essentially equal in certain properties, e.g., if they share the same goal. For example, two proofs are goal equivalent if they prove the same theorem.

Again to process variants, we need to make them understandable to a computer system. For a machine-processable representation of variants we can draw on approaches such as conditional markup in DITA or the markup of formal variants in OPENMATH. Unfortunately, neither approach is general and extensible enough for our purposes. Thus, two new markups are proposed: A grouping of document parts into a `variants` environment and the annotation of variant relations.

```
<variants>
  <proof xml:id="p1" for="#lemma1"
   ic="difficulty:high">...</proof>
  <proof xml:id="p2" for="#lemma1"
  ic="difficulty:medium">...</proof>
</variants>
```

Figure 3: XML-Encoding for Variant Groupings.

Figure 3 illustrates the grouping of three `proof` elements as variants. The `ref` element supports the transclusion of the third proof from another part of the document. The `ic` attribute is used to encode the difference of the three text paragraphs: They have different levels of difficulty.

Figure 4 presents the annotation of variant relations between the three proofs. We use the new metadata syntax for OMDOC (Lange and Kohlhase, 2009),

```
<proof xml:id="p1" for="#lemma1"
  xmlns:cc="http://omdoc.org/var.ctxt?" >
  <metadata>
    <link rel="cc:more_difficult_than" href="#p2"/>
  </metadata> ...
</proof>
<proof xml:id="p2" for="#lemma1">...</proof>
```

Figure 4: XML-Encoding of Variant Relation Markup.

which uses RDFA to mark properties and relations. The namespace declaration introduces the prefix `cc`, which points to a **content dictionary**. In OPENMATH, such content dictionaries are used to define the meaning of mathematical symbols. Here they are used as background ontology for context metadata and variant relations.

3.2 Variation on the Structure Layer

In addition to variants on the content layer, the proposed content planning of mathematical documents requires an adaptability on the structure layer. Having selected mathematical documents turned out to be very beneficial: (1) Mathematical documents are very explicit about relations. (2) Markup formats like OMDOC already thoroughly mark the **semantic context** of documents. Based on this, mathematical documents can be easily modularized into **dependency graphs**, where nodes correspond to addressable document parts and edges denote their dependencies (which are inferred from relations like 'proves' or 'illustrates').

If we solely consider semantic aspects, the topic-oriented approach is very natural for mathematical documents. Respective adaptation routines have been well elaborated. Nevertheless, mathematical formats solely focus on semantic aspects and do not yet consider the narrative flow of texts. For example, paragraph C in Figure 5 has to be placed before E as indicated by the transitional text 'see Pascal's triangle below'. Such transitions improve the coherence of documents but also reduce the reusability of their parts in the adaptation. But we need a variation on content and structure layer to support content planning!

E	Pascal's triangle is a geometric arrangement of the binomial coefficient in a triangle. Row number n contains the numbers $\binom{n}{k}$ for $k = 0, \ldots, n$.
C	The notation $\binom{n}{k}$ was introduced by Andreas von Ettingshausen in 1826, although the numbers were already known centuries before that ***(see Pascal's triangle below)***.

Figure 5: Content of Paragraph E and C.

To address this problem, this work proposes an extension of markup languages with a markup for **narrative variations**. In a first step, authors have to mark all transitional words and phrases to allow machines to separate them from the reusable parts of documents. In a second step, authors have to mark narrative dependencies between their document parts and associate these with the transition texts. Finally, authors can enrich their documents with alternative narrative dependencies and transition texts.

With this extension, we can now distinguish *semantic dependencies* (which are inferred from semantic relations as defined by formats like OMDOC) and *narrative dependencies* (which are added to support narrative variations). In the terminology of the content planning, these dependencies are referred to as **transitions**. They are traversed in order to arrange document parts. Again to process transitions they have to be represented in a machine-processable form (Müller, 2010). With the machine-processable representations of transition texts, such texts no longer reduce the reusability of document parts but can be flexibly hidden or displayed. The **narrative context** of documents, which is formed by coherent transitions between document parts, has been dynamized while preserving the coherence of the adaptation results.

3.3 Modularising Narrative Documents

Most adaptation approaches focus on topic-oriented documents. If we want to apply these approaches to narrative documents, we first need to modularize them into self-contained, independent units. The new markup for transition texts allows machine to distinguish narrative from reusable content and to extract self-contained units. However, even if we could simply decompose narrative documents into topics, a coherent assembly into narrative documents is no longer possible as topics omit transitional words and phrases. Consequently, we need a new adaptation model.

This work proposes to modularize narrative documents into information units (called **infoms**) for which all transition texts are preserved and narrative and semantic dependencies are marked. These are modelled as dependency graphs and processed during the content planning. To increase the variation on the content layer, variant infoms and variant relations are identified. To increase the variation on the structure layer, alternative transitional texts and narrative dependencies are marked.

This work is novel in that it distinguishes the semantic context of document parts, the narrative context of the document, and the user context. Users can prioritize semantic, narrative or their individual constraints to guide the substitution and reordering – the two content planning services addressed by this work. In the following, both workflows are introduced.

3.4 The Substitution Service

The substitution is implemented as two-stage process. In the first stage, the document is abstracted, i.e., certain parts are made adaptable. In particular, these parts are replaced with **holes**. The term 'hole' is used to denote a representation of document parts without substance, which has to be substituted with a concrete, user-specific document part. Thus, holes are the content planning correspondent for content markup in the rendering algorithm.

In the second stage, the abstract document ($d\Box c$) is converted into a user-specific, concrete document (doc_Δ) by substituting each hole with an appropriate document part according to the user's extensional and intensional context specification (ec^* and ic^*).

One contribution of this work is the finding that there is a strong correlation between the rendering workflow for notations and the content planning methods. Consequently, the substitution algorithm is specified as generalization of the rendering algorithm: The notation collector is replaced by the general *infom collector* and the rendering grabber is substituted with a *variant sorter*. Instead of iterating the subroutines for each mathematical expression, the substitution is performed on each hole ($\Box$). The infom collector returns a set of infoms (var^*) rather than notation definitions. These infoms and the effective intensional context (ic) are passed to the variant sorter. The variant sorter ranks these infoms according to how well their metadata matches with the user's context parameters (ic) and eventually returns the most appropriate infom (var), which replaces (or *fills*) the hole. The substitution returns a concrete document (doc_Δ) in which each hole has been substituted with user-specific content.

The substitution workflow has been implemented as abstract document module of the JOMDOC library. The *panta rhei* system has integrated JOMDOC to demonstrate the generation of exams: Teaching assistants can simply point to collections of exercises and specify the context of the new exam (e.g., in terms of language and difficulty level) and initiate the system to generate an exam. An application to a big exercise corpus is currently addressed in the TNTBASE project (Zholudev and Kohlhase, 2010).

3.5 The Reordering Service

The reordering service preserves the grouping of infoms in coarse-grained document parts like chapters, section, subsections, etc. It solely permutes the constituents of these parts.

The reordering service allows users to choose between three ordering strategies. The **context ordering** orders the infoms according to how well their metadata matches with the user context. It reuse the functionality of the variant sorter of the substitution algorithm and only provides a primitive solution that should be used as fallback. Semantic and narrative ordering process the dependency graph, where nodes correspond to the collected constituents of sortable document parts and edges denote their dependencies. The **semantic ordering** arranges infoms according to semantic transitions. The **narrative ordering** considers narrative transitions between infoms.

The reordering algorithm has been implemented in the adaptor library (Adaptor, 2010), which was integrated in the *panta rhei* system to demonstrate the ordering of the lecture notes of a theoretical computer science course. So far the workflow has only been applied to a small corpus, an application to a bigger document collection remains for future research.

4 ADAPTATION IN PRACTICE

The proposed workflows have been implemented and tested in several libraries (JOMDoc, 2010; Adaptor, 2010) and systems (Zholudev and Kohlhase, 2010; Giceva et al., 2009). One of these systems is *panta rhei*, a collaborative document reader and discussion platform that has been used as supplement to a theoretical computer science lecture at Jacobs University. The system integrates the Java library JOMDOC to support the adaptation of mathematical notations as well as the substitution of document parts and the *adaptor* library for the reordering.

5 CONCLUSIONS

This work has addressed the adaptation of narrative documents, by applying the topic-oriented principles of modularization and reuse to the narrative world. Narrative documents are modelled as dependency graphs, which are enriched by narrative variations and content variants. The novelty of this work is the distinction of narrative context, semantic context, and user context.

Mathematics is used as test tube, which required us to specify the rendering of notations. A generalized/extended form of the notation framework now supports the adaptation on all document layers. It empowers users to guide any step of the adaptation:(1) Which document parts should be adapted/remain unchanged, (2) the collection of adaptation objects (notation definitions, infoms, context parameters), and (3) the user-specific selection of the most appropriate object to be applied.

Future research will focus on the thorough evaluation of whether (and when) adaptation is beneficial as well as on the improvements of the adaptation methods, libraries, and systems.

REFERENCES

Adaptor (2010). The Adaptor Library. Retrieved from https://trac.kwarc.info/panta-rhei/wiki/adaptor on February 28, 2010.

Ausbrooks, R. et al. (2008). Mathematical Markup Language 3.0. W3C Working Draft, W3C.

Buswell, S. et al. (2004). The OPENMATH Standard 2.0. Technical report, The Open Math Society.

Giceva, J., Lange, C., and Rabe, F. (2009). Integrating Web Services into Active Mathematical Documents. In Carette, J. et al., editors, *Conference on Intelligent Computer Mathematics*, LNAI 5625, pp. 279–293. Springer.

JOMDoc (2010). JOMDoc — a Java Library for OMDoc documents. Retrieved from http://jomdoc.omdoc.org on May 28, 2010.

Kohlhase, M. (2006). OMDOC – *An open markup format for mathematical documents [Version 1.2]*, LNAI 4180. Springer.

Kohlhase, M., Müller, C., and Rabe, F. (2008). Notations for Living Mathematical Documents. In Autexier, S. et al., editors, *Proceedings of the Conference on Intelligent Computer Mathematics*, LNAI 5144, pp. 504–519. Springer.

Lange, C. and Kohlhase, M. (2009). A Mathematical Approach to Ontology Authoring and Documentation. In Carette, J. et al., editors, *Conference on Intelligent Computer Mathematics*, LNAI 5625, pp. 389–404. Springer.

Müller, C. (2010). *Adaptation of Mathematical Documents*. PhD thesis, Jacobs University Bremen.

Zholudev, V. and Kohlhase, M. (2010). Scripting Documents with XQuery: Virtual Documents in TNTBase. In *Proceedings of Balisage Markup Conference 2010*, Montréal, CA. in press.