

Jacobs University Bremen  
School of Engineering and Science

Research Proposal for a Ph.D. Thesis

# Towards a Semantic Wiki for Science

Christoph Lange <[ch.lange@iu-bremen.de](mailto:ch.lange@iu-bremen.de)>

March 3, 2007

Advisor: Prof. Dr. Michael Kohlhase

## Abstract

Collaborative work environments for scientific knowledge have many applications in research as well as in education. Such systems already exist (e.g. *Wikipedia* and *PlanetMath*), but they do not offer all the services users would like to have. Starting from semantic markup of knowledge as a prerequisite for services like dependency management, intelligent navigation and search, and for knowledge reuse by web services, I work out why a semantic wiki is an appropriate platform for integrating these services and how to entice authors to contribute their knowledge into such a wiki by sharing the benefits of the services that exploit the knowledge with them.

While most semantic wikis allow users to create hypertext with typed pages and links, the types being defined ad hoc or imported from arbitrary external ontologies, my approach is to tailor the wiki to scientific domains by incorporating both domain-specific semantic markup languages and their system ontologies and specifying services in terms of these ontologies. The SWIM prototype that supports the mathematical markup language OMDOC will be extended to a semantic work environment that offers interactive services operating on knowledge from multiple sciences. The resulting SWIM<sup>+</sup> system will be evaluated in a research and an education case study.

## Contents

<b>1</b>	<b>Motivation</b>	<b>3</b>
1.1	A Collaborative Work Environment for Scientific Knowledge Management . .	3
1.2	The Need for Structural Semantic Markup in Science . . . . .	4
1.3	Semantic Social Software as a Forerunner of the Future Web . . . . .	7
1.4	The Semantic Author’s Dilemma – Motivating and Gratifying Users . . . . .	7
1.5	Towards a Semantic Wiki for Science . . . . .	8
<b>2</b>	<b>State of the Art</b>	<b>8</b>
2.1	Semantic Wikis . . . . .	8
2.2	Knowledge Representation and Management in Mathematics and Science . .	10
<b>3</b>	<b>Starting Point: The SWiM Prototype</b>	<b>11</b>
3.1	Editing OMDoc in SWiM . . . . .	11
3.2	Presenting OMDoc in SWiM . . . . .	12
3.3	Extracting and Using Knowledge from OMDoc . . . . .	12
3.4	OMDoc’s System Ontology . . . . .	14
<b>4</b>	<b>Work Plan</b>	<b>15</b>
4.1	WP-P: Technical Preparation of the Prototype . . . . .	15
4.2	WP-R: Get a Reasoner Running . . . . .	15
4.3	WP-OL: System Ontologies of the Markup Languages . . . . .	15
4.4	WP-M: Mapping from XML to Ontologies and Back . . . . .	16
4.5	WP-OS: An Ontology of Collaborative Environments . . . . .	17
4.6	WP-AV: Planning Services using Added-Value Analysis . . . . .	17
4.7	WP-SI: A Service Programming Interface . . . . .	18
4.8	WP-SA: A Service for Auto-Completion . . . . .	19
4.9	WP-SD: A Dependency Graph Navigation Service . . . . .	19
4.10	WP-SM: A Dependency Management Service . . . . .	21
4.11	WP-SE: An Edit-in-Place Service . . . . .	22
4.12	WP-SP: Scientific Publishing with SWiM+ . . . . .	23
4.13	WP-SO: Other Services . . . . .	23
4.14	WP-CS: Case Study as a Tool for Scientists . . . . .	23
4.15	WP-CE: Case Study in Education . . . . .	24
4.16	Preliminary Schedule . . . . .	24
<b>5</b>	<b>Conclusion and Outlook</b>	<b>24</b>
	<b>References</b>	<b>26</b>

# 1 Motivation

The core problem of this thesis is: How can open, collaborative work environments like *Wikipedia* [Wik06b] be improved in a way that they serve the needs of scientists and learners better? I argue that this can be done by integrating semantic markup languages into a semantic wiki and pose the second core question: How can users be motivated to contribute content in such an environment? My objective is to implement a system, provisionally named SWIM<sup>+</sup> (semantic wiki for mathematics and more) that answers these questions.

## 1.1 A Collaborative Work Environment for Scientific Knowledge Management

For the purpose of this thesis, I want to focus on the following three applications of a collaborative work environment for scientific knowledge:

1. It should serve scientists as a powerful tool for jointly developing theories from their hypotheses and publishing them on the web.
2. It should enable scientists and scholars to publish a comprehensive knowledge collection as a work of reference and a source for education.
3. It should serve as a collection of knowledge units that are reusable by humans and (automated) web services.

For the first application, the SWIM<sup>+</sup> system should support formalisations of ideas and management of dependencies. If, for example, one scientist, named E. IN. STEIN, wants to elaborate on his hypotheses about “relativity”, he should be able to search for existing theories he can build on (e.g. the theory of gravitation developed by his colleague N. EW. TON), and SWIM<sup>+</sup> should support him in formalising his ideas using previously defined terms from other theories. However, these “other” theories he build on, may still be under development. If N. EW. TON changes some basic assumptions on gravitation, E. IN. STEIN’s considerations building on them might become invalid and thus needs to be adapted.

For the second application, SWIM<sup>+</sup> should adapt to the previous knowledge and preferences of its users: If the user S. CH. OLAR wants to learn more about a topic that has many prerequisites (e.g. calculus, which presupposes knowledge about sequences and limits), SWIM<sup>+</sup> should ask him whether he has already heard about these topics. If not, he should be offered to read about them first. S. CH. OLAR should also be able to search for topics in a flexible way: by navigation along the relations among the topics in the knowledge collection, by entering keywords, and by entering objects of the respective area of science (e.g. formulae in mathematics or chemistry). Furthermore, he should be able to tailor the presentation to his preferred language and notation and to submit comments to course notes.

For the third application, SWIM<sup>+</sup> should be able to integrate services from other web sites or applications on the user’s computer (e.g. visualising tools or simulators), where it does not support these services by itself. The knowledge units collected by the users of the environment should be made accessible to web services so that automated agents (e.g. intelligent search engines) can reuse them.

All of the above applications require a comprehensive *knowledge management*<sup>1</sup>. Scientific knowledge is usually represented in *documents*. One advantage of documents — compared to

---

<sup>1</sup>In this thesis, I understand knowledge management in the same way as the community around the conference on Mathematical Knowledge Management (MKM) does — as “an emerging interdisciplinary field of research

databases, for example, — is that they can have a *narrative* structure, which is most suitable for didactics [Koh06c, chapter 8]. While databases (in a general sense, not just relational databases) are most suitable for computing tasks like automated deduction, databases can still be generated or fed from well-structured documents by only extracting the logical structure. On the other hand, a mere content or document management system is not sufficient to solve the problems I introduced, as it understands too little of the *structures* of scientific knowledge, such as the dependencies among theories and the meaning of the symbols in a formula.

### 1.1.1 Particular Use Cases for Mathematical Knowledge Management

While the above examples apply to science in general, or, at least, to multiple areas of science, more detailed use cases for the above-mentioned three applications can be found in specific sciences — for example, mathematics, being a particularly well-structured and well-conceptualised subject. The following use cases are inspired by M. KOHLHASE’s outlook towards a “Semantic Web for mathematics” [Koh06c, chapter 3.4]:

1. As a tool for scientists, SWIM<sup>+</sup> could automatically check published proofs, if they are sufficiently detailed and structured. It could support semantic search for mathematical concepts rather than keywords (“Are there any objects with the group property out there?”) and data mining for representation theorems (“Are there undiscovered groups out there?”). A similar case is classification: given a concrete mathematical structure, is there a general theory for it?
2. As a reference for education, SWIM<sup>+</sup> could explain mathematical proofs, e.g. by specialising them to an easy-to-understand example for a specific case. A use case for navigation in mathematics is the need to learn what a certain symbol in a formula (e.g. the “!” in  $e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!}$ ) means. Furthermore, as many notational conventions are specific to certain communities of practice<sup>2</sup>, it should be possible to customise the presentation of formulae accordingly.
3. In a collection of reusable knowledge, semantic search for formulae should be supported: If a user searches for  $(x + y) \cdot z$ ,  $a \cdot c + b \cdot c$  may qualify as a result, depending on the definition of the operators and variables used<sup>3</sup>. Cut and paste of mathematical objects on the level of computation is likewise desirable: One could take one search result and paste it into a computer algebra system.

## 1.2 The Need for Structural Semantic Markup in Science

Large computer-based collections not only of scientific knowledge created in a collaborative effort have emerged in recent years. The encyclopedia *Wikipedia* [Wik06b] with its more

---

in the intersection of mathematics, computer science, library science, and scientific publishing.” [BF06] that “provid[es] new techniques for managing the enormous volume of mathematical knowledge available in current mathematical sources and making it available through the new developments in information technology” [Koh05a]. I extend this definition of knowledge management from mathematics to science in general.

<sup>2</sup>Consider, for example, binomial coefficients:  $\binom{n}{k}$  is one standard notation, but  $C_k^n$  is used by French mathematicians, while Russians write  $C_n^k$  [KK06].

<sup>3</sup>In this case, a law of distributivity must hold for the operators  $+$  and  $\cdot$ , and the variables  $x/a$ ,  $y/b$  and  $z/c$  must be free or have the same values.

than  $5 \cdot 10^6$  articles in more than a hundred languages is the most successful one for general knowledge, including science. *PlanetMath*<sup>4</sup> with entries on more than 10,000 mathematical topics is one of the largest online compilations of mathematical knowledge. *Wikipedia* contains human-readable text with tables, graphics and formulae, which is merely grouped into categories and searchable in full-text, whereas *PlanetMath* has a fixed set of metadata associated with each article, including its type (definition, theorem, etc.), parent topic, Mathematics Subject Classification, synonyms and keywords, but the content itself is written in presentational-only L<sup>A</sup>T<sub>E</sub>X<sup>5</sup>.

Neither *Wikipedia* nor *PlanetMath* support the use cases outlined in section 1.1: As the *structural semantics* of the scientific knowledge is not made explicit, machines cannot understand the structures and thus cannot exploit them. Both projects are thus not part of the *Semantic Web*, a “web of data” that “provides a common framework that allows data to be shared and reused across application, enterprise and community boundaries” [W3C06].

Exposing structural semantics means to explicitly mark the structural divisions of documents, to declare the dependencies among these divisions and the documents, to link symbols in formulae to their definitions, to declare the order of operations in a formula, etc.? To expose the structural semantics of scientific knowledge, many languages have been developed:

- **Content MathML**<sup>6</sup> [ABC<sup>+</sup>03] and **OpenMath** [BCC<sup>+</sup>04] for mathematical formulae
- **OMDoc** (Open Mathematical Documents) [Koh06c], which embeds the former ones for formulae, but allows for expressing larger structures, that is, statements and theories
- **PhysML** [HKS06], an OMDOC derivative for physics
- **CML** (Chemical Markup Language) [CML07] for chemical concepts like molecules and reactions
- **CNXML** [HG05], a lightweight markup language for structuring general educational content (but not structures of specific scientific domains). It is used by *Connexions*<sup>7</sup>, a community-driven collection of free educational content.

All of the above languages are applications of XML (eXtensible *Markup* Language) [BPSM<sup>+</sup>04]. Actually, most newly-developed document formats are XML applications nowadays, because XML can easily represent hierarchical structures through containment of child elements and relations via URI references. Furthermore, it is an open W3C standard, and it is easy to process due to its strict syntax and therefore supported by many tools and libraries.

The variety of applications that are already available for the above-mentioned languages demonstrate that the problems stated in section 1.1 can actually be tackled using semantic markup: For OMDOC, for instance, there are (separate) applications for searching formulae, publishing (including community-specific notations), generating user-specific textbooks and verifying proofs [Koh06c] (see also section 2.2).

<sup>4</sup><http://www.planetmath.org>, see also [Kro03]

<sup>5</sup>L<sup>A</sup>T<sub>E</sub>X supports presentation-oriented markup out of the box. As T<sub>E</sub>X is programmable, one can, however, extend it by structural semantic markup, as has been done with sT<sub>E</sub>X (see page 24).

<sup>6</sup>... the structural semantic twin of *Presentation MathML*, which is dedicated to presenting mathematical formulae in a human-readable way

<sup>7</sup><http://cnx.org>

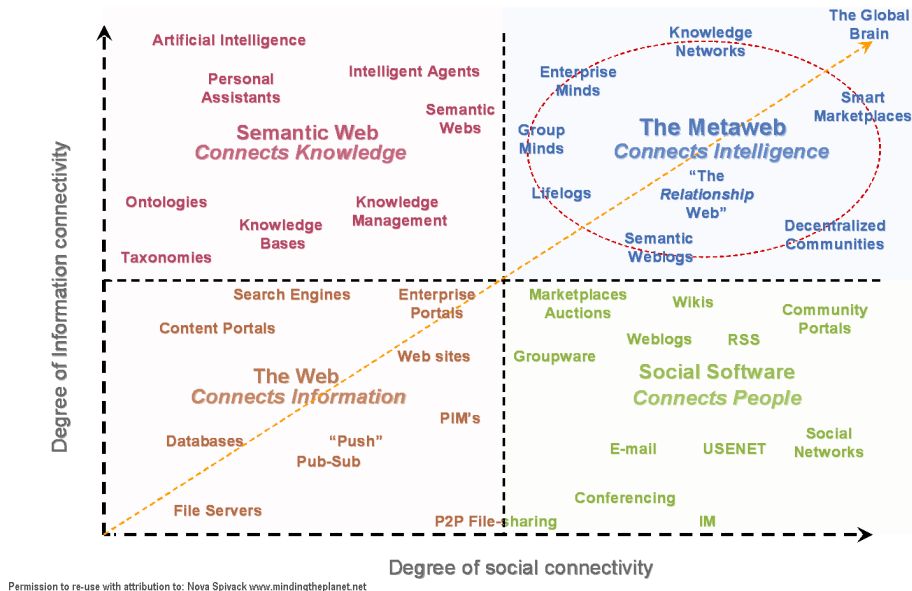


Figure 1: Graph on the integration of social software and semantic web [Spi06]

### 1.2.1 Semantic Markup as a Special Kind of Semantic Annotation

In the context of the Semantic Web, both the term “annotation” and the term “markup” are used for similar things. Annotations are metadata (data about data) that use terms from an ontology to describe the meaning (i.e. the semantics) of certain data [UCI<sup>+</sup>06]. In general, an annotation is “a note of explanation or comment added to a text or diagram” [McK05]; the etymology of the word (from Latin “annotare”, where “ad-” means “to” and “nota” is a mark [McK05]) indicates that annotations are optional. On the Semantic Web, they actually need not be part of that data; they can also be stored externally and point to the original data via URI references. Markup is defined as “a set of tags assigned to elements of a text to indicate their structural or logical relation to the rest of the text” [McK05], where a tag is “a character or set of characters appended to or enclosing an item of data in order to identify it” [McK05]. In the case of XML, where document formats tend to have strictly defined document models<sup>8</sup>, markup is mandatory to a large extent.

In the following, I will regard semantic markup as semantic annotations embedded into the document they describe. A typical annotation workflow for structural semantic markup in science is refining the markup in a stepwise formalisation process from human-readable text to a representation suitable for the Semantic Web to a full formalisation that can be verified by a theorem prover in the case of mathematics [Koh06c, chapter 4].

### 1.3 Semantic Social Software as a Forerunner of the Future Web

A collaborative environment for semantically marked-up knowledge is an example of “semantic social software” — social software combined with Semantic Web technologies. Where social software connects people (here: scientists and scholars) and the Semantic Web formalises web content, semantic social software is the connection of both [Sch06b]. SCHAFFERT identifies the perspective of a “convergence of the Web, Social Software and the Semantic Web” [Spi06] in many predictions about the future web; SPIVACK, for example, speaks about a “metaweb” as “a new higher-order whole” instead of “just a bunch of interacting parts” [Spi06] (see figure 1).

According to SCHAFFERT, semantic social software has two perspectives that are two sides of the same coin; it improves both:

1. the usage of social software by adding semantic annotations, which are then exploited by sophisticated services (“semantically enabled social software”), and
2. the process of creating Semantic Web metadata by using social software (“socially enabled semantic web”) [Sch06b].

As one application of semantic social software, he introduces semantic wikis (see section 2.1) as “community-authored knowledge models” where informal natural language descriptions created by domain experts are formalised in collaboration with knowledge engineers [Sch06b]. As one of my main focuses is a tool for scientists who jointly author knowledge (see section 1.1) and as M. KOHLHASE identifies the formalisation workflow supported by wikis as necessary for scientific markup (see section 1.2.1), I believe that a semantic wiki matches the picture of a collaborative environment for scientific knowledge management best. The SWIM<sup>+</sup> system projected in this thesis, although initially motivated via the first aspect of semantic social software (see sections 1.1, 1.2) can also be seen from the second point of view: Many huge scientific knowledge bases, including the above-mentioned projects *Wikipedia* and *PlanetMath*, were created in a joint effort of many people — supported by social software.

### 1.4 The Semantic Author’s Dilemma – Motivating and Gratifying Users

The advantages of the semantic markup languages mentioned in section 1.2 are that they are easily processible by software and that they expose the structural semantics of the scientific knowledge they encode. The downside is that they are hard to author for humans because they require formalising meaning, which demands time and effort from the authors. According to A. KOHLHASE and N. MÜLLER, allowing users not only to consume content but also to produce it is attractive in many settings, but scientific communities have experienced that the mere prospect of great applications that become possible with scientific knowledge management (the “macro-perspective”) once enough content is already available is not sufficient to entice potential authors to overcome the hurdles of authoring semantically rich content. From their individual “micro-perspectives”, they perceive drawbacks that argue against taking action: the expectation of a high initial investment proportional to the depth of the markup and the uncertainty about the other authors’ will to collaborate and the benefit

---

<sup>8</sup>There are many languages to define an such a document model, which is also called an “XML schema”; one of them is named “XML Schema”, but there are others, like DTD and RelaxNG. In the following, I will continue to use the term “document model” to avoid confusion.

the individual authors will gain from their own markup [KM07]<sup>9</sup>. Therefore, A. KOHLHASE and N. MÜLLER argue that a semantic work environment must offer “elaborate, semantic added-value services for the concrete situation the user is in” [KM07] to motivate users into action and propose a method for designing and analysing services (see section 4.6).

The user need not even be aware of the macro-perspective to be motivated, as A. KOHLHASE argues elsewhere: Many people primarily appreciate social tagging systems (e.g. *del.icio.us* or *flickr*) as *personal* knowledge management systems before they realise that they are contributing to a shared knowledge base [Koh06a].

Apart from *motivating* users into action, the approach of *instantly gratifying* users for their contributions also prevails in semantic social software [Sch06b]. *Mangrove* was the first semantic work environment with instant gratification as its central design goal. Authors can semantically annotate their web pages and publish them to the system, which notifies registered services (a central calendar, for example) that instantly harvest the annotations and return feedback to the user about how and where his annotations have been utilised [MEG<sup>+</sup>03]. *WikSAR*, one of the first semantic wikis (see also section 2.1), is inspired by this approach and provides instant gratification by instantly improving navigation: Not only are incoming and outgoing links displayed for each page, grouped by their types, but also links to other semantically related pages are inferred [AA05]. In fact, this is a core service in most semantic wikis.

## 1.5 Towards a Semantic Wiki for Science

Following the justifications given in the previous sections, my further work will focus on realising the collaborative work environment for scientific knowledge management sketched in section 1.1 as a semantic wiki (see section 1.3) whose pages are written in scientific markup languages (see section 1.2), keeping the question how to motivate users contribute in mind (see section 1.4). First, I give an overview of the state of the art, both in semantic wikis (see section 2.1) and in scientific knowledge management (see section 2.2); then, I discuss the already-existing SWIM prototype (see section 3) for mathematical knowledge and establish a work plan how to improve and extend it towards science in general (see section 4).

# 2 State of the Art

## 2.1 Semantic Wikis

A semantic wiki is a wiki with an “underlying model of the knowledge described in its pages” [Wik06c], but there are many different ways to gather, store, and utilise that knowledge<sup>10</sup>. Common semantic wikis, most of which are still research prototypes, combine the wiki principle with Semantic Web technologies; thus, they are semantic social software. Most semantic wikis utilise formal languages like RDF [LS99] or OWL [MvH04] for annotating pages and links with semantic information. *SweetWiki* [BCG<sup>+</sup>06], for example, features a wiki system ontology with classes like “page” or “user” and lets users tag pages with concepts from several OWL domain ontologies. Other well-known semantic wikis include *Semantic*

---

<sup>9</sup>A. KOHLHASE and N. MÜLLER compare this situation to the “prisoner’s dilemma” from game theory.

<sup>10</sup>For an overview, see [Lan07, section 1.4]. Also note [ODM<sup>+</sup>06], which focuses on annotation and navigation, and the comprehensive brainstorming on [http://ontoworld.org/wiki/Semantic\\_Wiki\\_State\\_of\\_The\\_Art\\_Paper](http://ontoworld.org/wiki/Semantic_Wiki_State_of_The_Art_Paper).

*MediaWiki* [VKV<sup>+</sup>06]<sup>11</sup>, an attempt to integrate the *Wikipedia* into the Semantic Web, and *IkeWiki*<sup>12</sup>, which is intended as a social tool for knowledge engineering<sup>13</sup>. In most semantic wikis, including, *IkeWiki* and *Semantic MediaWiki*, one page describes one real-world concept. The page itself and its links to other pages are *typed* using terms from ontologies. Importing and exporting RDF from and to the semantic web is possible in most semantic wikis, and the knowledge base can be queried. Some semantic wikis employ a reasoning engine to infer additional knowledge from the explicit annotations that users made. Recent trends in wikis in general include:

- improved user experience through WYSIWYG editing — as, for example, in *SweetWiki* and *IkeWiki*,
- integration with other social software like Weblogs — *SnipSnap*<sup>14</sup> is an example of a wiki with blogging functionality and some semantic features (“Labels”)
- a standardised web service interface and data exchange format for automated clients and distributed systems [V<sup>+</sup>07], as well as
- better spam and vandalism protection (esp. on large, public wiki sites like *Wikipedia*<sup>15</sup>).

Another important aspect is bringing semantic wikis to non-scientists. The by far most successful software in this regard is *Semantic MediaWiki*, which, despite its intention, is not yet used by *Wikipedia* (mainly for performance and scalability reasons), but by numerous other sites [Ont07b]. The trends mentioned for general wikis also prevail in the area of semantic wikis, with the addition of the following trends that apply to SWIM<sup>+</sup>:

**Reasoning:** One aspect of current research is formalising the knowledge in a semantic wiki more stringently — from ad-hoc database queries to description logic (DL) reasoning, the most common logic formalism on the Semantic Web, as well as other ways of reasoning. SCHAFFERT mentions tolerance about inconsistencies in an evolving knowledge model, versioning of annotations, coping with the tradeoff between soundness and performance improvements, as well as providing justifications for decisions based on reasoning to the user as major challenges for reasoning in semantic wikis [Sch06b, VK06].

**Visualisation** of the RDF graph of the knowledge inside the wiki [ER06] and incorporation of visual mapping techniques [HKV06]

Most semantic wikis serve general purposes; as of February 2007, there is only one more semantic wiki dedicated to mathematical knowledge management apart from SWIM: *se(ma)<sup>2</sup>wi* [Zin06] is an installation of *Semantic MediaWiki*. OMDOC content from the ACTIVEMATH learning environment [MGP<sup>+</sup>06] has been automatically converted to *MediaWiki*’s syntax via XSL transformations and imported to the wiki. The pages are categorised by their OMDOC statement types (e.g. *definition*) and annotated with learning metadata from ACTIVEMATH. The

<sup>11</sup>See also the home page [http://ontoworld.org/wiki/Semantic\\_MediaWiki](http://ontoworld.org/wiki/Semantic_MediaWiki)

<sup>12</sup><http://ikewiki.salzburgresearch.at>, see also [Sch06a]

<sup>13</sup>The first well-known semantic wikis are *Platypus* [TCC04] and *WikSAR* [AA05], both of 2004, but they are not under active development any more.

<sup>14</sup><http://snipsnap.org>

<sup>15</sup><http://en.wikipedia.org/wiki/Wikipedia:Vandalism>

other semantic information from OMDOC has, however, got lost in the conversion: The formulae are given in presentational-only  $\text{\LaTeX}$ , and the links between wiki pages that represent mathematical statements, for example a link from a theorem to its proof, are not typed, as they are in SWIM.

## 2.2 Knowledge Representation and Management in Mathematics and Science

All sciences rely on mathematical foundations for modeling and formalisation. To represent scientific knowledge, I follow the three-layered structure model of mathematics and science that M. KOHLHASE proposed and realised with the OMDOC markup language [Koh05b]:

**Object level:** “represents objects such as complex numbers, derivatives, etc. for mathematics, [molecules in chemistry] or observables for physics.” [Koh05b]

**Statement level:** “sciences are concerned with modeling our environment [using] statements about the objects in it. [Statements include] model assumptions, their consequences, hypotheses, and measurement results. All of them [...] state relationships between scientific objects and have to be verified or falsified in theories or experiments. [They] have a conventionalised structure, e.g. *exercise, definition, theorem, proof*, and [there is] a standardised set of relationships among them. For instance, a model is fully determined by its assumptions (also called *axioms*); all consequences are deductively derived from them (via *theorems* and *proofs*), and therefore their experimental falsification uncovers false assumptions of the model.” [Koh05b]

**Theory/Context level:** “Representations always depend on the [...] context; even the meaning of a single symbol<sup>16</sup> is determined by its context, and depending on the current assumptions, a statement can be true or false. Therefore, the sciences [...] fix and describe the situation of a statement. [...] For instance, in mathematical logic, a theory is the deductive closure of a set of axioms, i.e. the (in general infinite) set of logical consequences of the model assumptions.” [Koh05b]

The OMDOC markup language supports the above-mentioned three levels in a mathematical setting, but it was easy to extend it towards physics [HKS06]. The resulting PHYSML language left the three-layered structure intact, only adding a few concepts (*observable, system* and *experiment*) on the statement level, and its authors estimate that an extension towards chemistry will be possible in a similar way [HKS06]. With new representation formalisms, new search capabilities are needed to replace full-text search, which does not utilise the above-mentioned relationships. Similarly, formula-centered software tools like computer algebra systems are not sufficient, as they only rely on the object level, but not on the statement and theory level.

### 2.2.1 Theory Management

Theory management has first been implemented in the domain of automated software verification, but it starts expanding to other scientific domains. Large, modular collections of interdependent theories form *development graphs* [AH02, MAH06], which need to be managed and verified “in-the-large”. In a development graph, each node represents a theory with

---

<sup>16</sup>e.g. the glyph  $h$  as the height of a triangle or Planck’s quantum of action

its axioms and theorems. Edges represent *theory inclusions* [FGT92]<sup>17</sup>: If axioms from the source theory  $\mathcal{S}$  can be found in the target theory  $\mathcal{T}$ , possibly after applying a signature morphism (i.e. remapping symbols), or proven as theorems following from  $\mathcal{T}$ 's axioms, theorems from  $\mathcal{S}$  can be reused in  $\mathcal{T}$ . A theory inclusion can be decomposed into edges from individual axioms of  $\mathcal{S}$  to  $\mathcal{T}$ , each of which must be proved “in-the-small”, that is, within the theory  $\mathcal{T}$ . In mathematics, proof assistance systems such as proof verifiers or automated theorem provers can be used for that. Note, however, that automated theorem proving is expensive (therefore theory management started in a domain where safety and security is essential), and that many applications therefore do not use it. OMDOC, which only cares about structural requirements, is in fact being successfully applied in many of the settings presented in section 1.1, although its model “only cares about structural aspects [of proofs]”: Theorems, for example, are considered “true” if a proof exists, however natured. “[W]hether the author chooses to prove [a statement] or indeed whether the statement is true at all is left to other levels of modeling.” [Koh06c]

### 2.2.2 Semantic Search

Searching interrelated statements and theories is possible if the relations are, for example, available in a relational database or in a graph representation such as RDF — which can be generated from a markup language. Searching for mathematical or chemical formulae is more involved; for mathematics, M. KOHLHASE and ŞUCAN identified the following to approaches [Kc06]: The simple-minded one is to convert the formulae to text representations and have them indexed by a full-text search engine. Different formulae can, however, be equivalent modulo  $\alpha$ -equivalence (renaming of variables, consider  $\int_a^b x dx$  and  $\int_p^q y dy$ ) or modulo some properties of operators (for example commutativity:  $a + b = b + a$ ). Structural indexing is slower, but yields better results: The formula is given in a content markup format, and its tree structure is leveraged.  $\alpha$ -equivalence becomes manageable that way, and other semantic equivalences can also be taken into account.

## 3 Starting Point: The SWiM Prototype

My implementation of the depicted work environment does not start from scratch; instead, I will evolve the SWiM (Semantic Wiki for Mathematical Knowledge Management) prototype, developed during my diploma thesis. It is a fork of the Java-based *IkeWiki* (see section 2.1) with OMDOC as its page format [Lan07], modeled in an object-oriented, extensible way (see figure 2), which makes the incorporation of additional document formats feasible. Even the original wiki text markup has been retained, which is still suitable for overview pages or unstructured notes.

### 3.1 Editing OMDoc in SWiM

So far, the source code of an OMDOC document can be edited in SWiM. The OMDOC syntax has been slightly adapted to fit the requirements of a wiki, which include easy linking and small pages. Small pages improve the effectivity of wiki usage, as they facilitate editing and re-use by linking and allow for a better overview through lists of recent changes and

---

<sup>17</sup>They are called “theory interpretations” there.

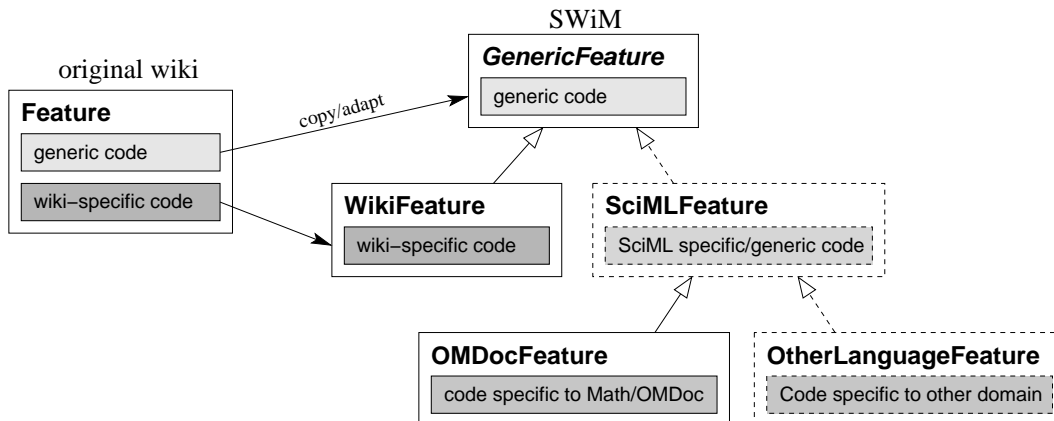


Figure 2: Refactoring of wiki classes. Dashed boxes stand for the projected extension towards general scientific markup languages (see section 4.1).

other automatically generated index pages. Therefore, link targets need not be full URI references, but they can be abbreviated: A statement on a theory page can be referenced as `theory#statement`. While OMDOC only allows statements that are not constitutive for a theory<sup>18</sup> to live in their own documents, the SWiM-extended OMDOC document model allows any kind of statement to be rolled out to its own page.

### 3.2 Presenting OMDoc in SWiM

For presentation (see figure 3), SWiM uses a slight adaptation of the multi-level XSL Transformation workflow to XHTML+MathML that has earlier been developed for OMDOC [Koh06c, chapter 25.1]. The links from symbols in formulae to their definitions, which are generated during this transformation, improve the navigability of the wiki. Additionally, a hyperlinked source view is available. It is particularly useful for browsing complex OMDOC documents, as not all kinds of links between statements or theories have been mapped to RDF triples and thus semantic links navigable in the wiki (see below).

### 3.3 Extracting and Using Knowledge from OMDoc

*IkeWiki*, as most semantic wikis, considers each wiki page to represent one real-world concept. As for OMDOC, I considered small theories<sup>19</sup> and statements appropriate page-level concepts, but not sub-statement structures such as proof steps, as the latter would make it difficult to overlook complex statements.

Relations between theories and statements can be expressed in OMDOC either through containment of child elements within parent elements — a theory can have statements as its children, for example, — and via URI references — for instance, from an *example* element to

<sup>18</sup>Symbols, definitions, and axioms are indispensable for the meaning of a theory (“constitutive”), but assertions, their proofs, alternative definitions of concepts already defined, and examples are not.

<sup>19</sup>According to M. KOHLHASE, OMDOC advises to follow a “little theories approach” [FGT92], where theories introduce as few new concepts as possible. A theory may introduce more than one concept, if they are interdependent, e.g. to introduce the natural numbers via the Peano Axioms, we need to introduce the set of natural numbers, the number zero and the successor function at the same time.

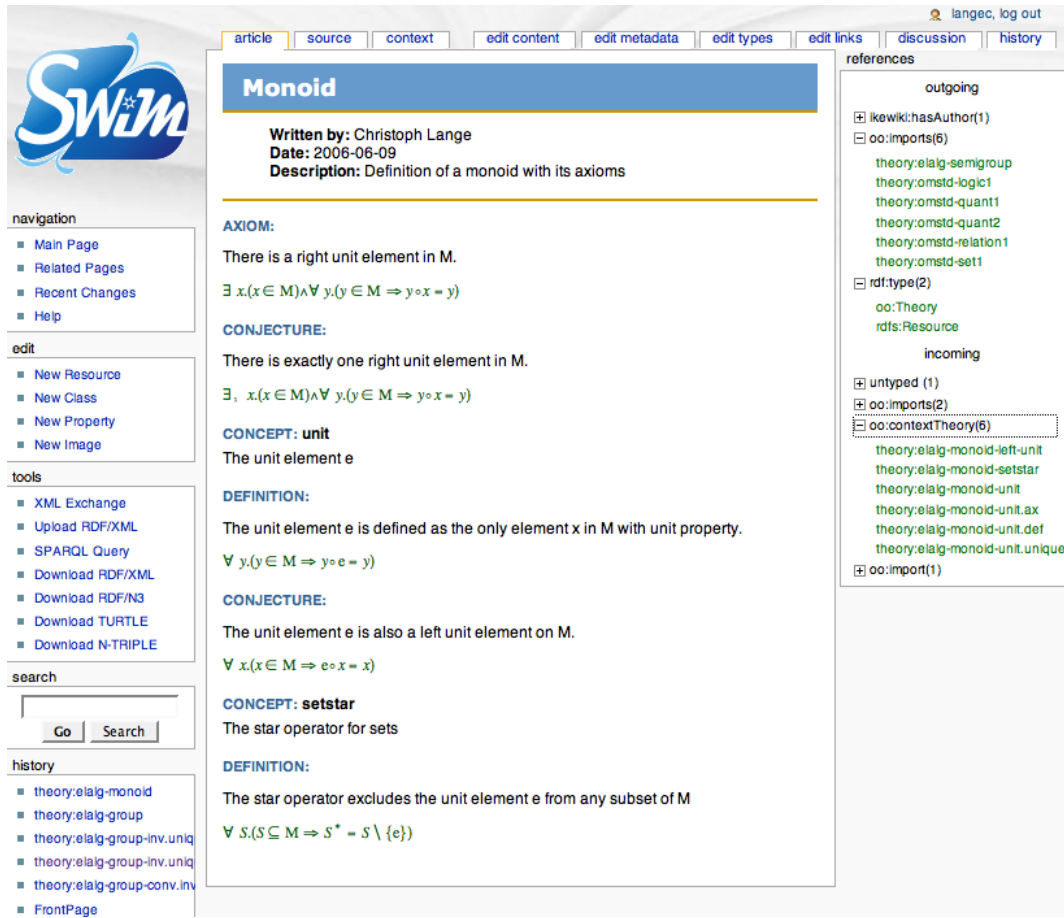


Figure 3: A rendered page in the SWIM prototype

the assertion or definition it explains. To enable *IkeWiki* and other Semantic Web applications to reason on these relations and to exploit them for navigation and interactive queries, the information about the concept instances and their interrelations must be made accessible in a Semantic Web standard format, that is, as RDF subject–predicate–object triples.

While the vocabulary for encoding knowledge about mathematics in OMDOC has been fixed in a RelaxNG schema [Koh06c, appendix D], there was no such vocabulary — called *ontology* in the context of the Semantic Web<sup>20</sup> — for encoding knowledge about mathematics in RDF. Therefore, I modeled a subset of the system ontology of OMDOC (see section 3.4), representing most concepts and relations of OMDOC’s statement module [Koh06c, chapter 15]. Given that, it remained to *extract* those parts of the knowledge that could be represented in terms of that ontology from OMDOC to a more explicit RDF representation; after all, the knowledge is not available as separate, handy annotation in scientific markup languages, but rather buried in the markup. For the SWIM prototype, a simple RDF extraction procedure based on XPath expressions with a hard-coded mapping from XML elements to concepts of the ontology has been implemented; it extracts information about the types of theories and statements as well

<sup>20</sup>Ontologies are formal, explicit specifications of a conceptualisation; they describe a specific domain and emerge as a result of a shared understanding among experts in that domain. On the Semantic Web, the most common formalism for ontologies is description logic.

as the links between them. *IkeWiki* can directly use these links to improve navigation: They are displayed in a special “references” box, grouped by type, as can be seen on the right side of figure 3.

### 3.4 OMDoc’s System Ontology

As the above-mentioned ontology represents knowledge *about* mathematical knowledge (e.g. that a proof proves a theorem), whereas OMDoc in itself is an ontology language for mathematical knowledge, it could be called a “meta”-ontology. Still, the term *system ontology* is more common in our group; in [Mül06], it is defined as “an ontology describing the data model of a system or the representation language the system and its applications are based on — independently of their respective syntactical realisation”. Note that this is different from an ontology that directly represents mathematical concepts. Relations between mathematical concepts, such as “all differentiable functions are continuous” cannot be expressed directly in OMDoc; as OMDoc captures how scientists communicate about mathematics, they must be wrapped into mathematical statements — assertions in this case —, but can nevertheless be extracted if a DL representation is required<sup>21</sup>.

SWiM differs from other semantic wikis in that it comes with a pre-bundled system ontology of a domain-specific data format. Most general-purpose semantic wikis allow for importing or modeling ontologies for arbitrary domains, but due to the great differences between those ontologies, they cannot utilise them further than for displaying navigation links, some editing assistance, and semantic search. As the OMDoc system ontology contains uniform terms for semantic relations between mathematical statements and theories — like “dependency” or “containment” —, services built on top of SWiM can be formally specified in terms of the system ontology (see section 4.7).

So far, a subset of this system ontology, which is implicitly given by the OMDoc specification in a human-readable but not machine-understandable way, has been modeled in OWL-DL (the description logic sub-language of the Web Ontology Language [MvH04]), covering most of the statement module and a small part of the proof module of OMDoc (see [Koh06c, chapters 15 and 17]). Figure 4 shows the most important concepts (in OWL: classes) and relations between them (object properties) that have been modeled: theories, a hierarchy of several statement classes and a generic transitive “concept–depends-on–concept” that subsumes the transitive import relation between theories, the containment relation between a statement and the home theory that fixes its context, and several relations between statements, where one statement further specifies another one: “symbol–has–definition”, “proof–proves–assertion”, and “example–exemplifies–statement(s)”.

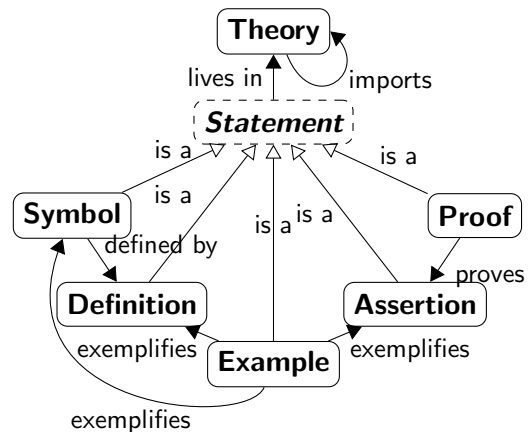


Figure 4: A subset of OMDoc’s system ontology

<sup>21</sup>The *proof* of this theorem about functions cannot be expressed in DL, though, as it requires higher-order logic. The latter is, however, disliked in the context of the Semantic Web, as it is not decidable.

## 4 Work Plan

The development of the collaborative work environment SWiM<sup>+</sup>, which shall offer services for knowledge from many sciences, can be split up into several work packages depending upon each other, mainly packages concerning implementational issues, the architecture of the system, detailed specifications of some individual services, as well case studies: preparation of the prototype (WP-P), putting the reasoning engine into operation (WP-R), modeling the system ontologies of the scientific markup languages (WP-OL), an XML $\leftrightarrow$ ontology mapping (WP-M), modeling the system ontology of the collaborative work environment (WP-OS), planning services using added-value analysis (WP-AV), designing a service programming interface (WP-SI), an auto-completion service (WP-SA), a dependency graph navigation service (WP-SD), a dependency management service (WP-SM), an edit-in-place service (WP-SE), scientific publishing (WP-SP), other services not yet specified in detail (WP-SO), a case study of the applicability as a tool for scientists (WP-CS), and a case study in education (WP-CE).

### 4.1 WP-P: Technical Preparation of the Prototype

The first, pragmatic step of my work will be the release of the existing SWiM prototype with the most urgent bug fixes. (Detailed bug reports are available at <https://trac.kwarc.info/swim/>.) This is intended to attract prospective users and lead to feedback, which is particularly valuable for the added-value analysis (see section 4.6). As the prototype is still based on an outdated version of *IkeWiki*, the next step will be the upgrade to its current version 1.99, which, among many bug fixes, features a completely redesigned user interface. Then, I will apply the refactoring depicted in dashed lines in figure 2 in order to set the stage for wiki pages in scientific markup languages other than OMDOC.

### 4.2 WP-R: Get a Reasoner Running

Some of the projected services (see the work packages WP-SD in section 4.9 and WP-SM in section 4.9) require computing transitive closures of the dependency relations. *IkeWiki* includes the *Pellet* DL reasoner [SPG<sup>+</sup>06], as well as an experimental rule-based reasoner implemented in Prolog, which are both capable of doing that. As regards the so far unsolved problem with reasoning despite inconsistencies in semantic wikis (see section 2.1), I will try to find a pragmatic solution to get most out of the planned services.

### 4.3 WP-OL: System Ontologies of the Markup Languages

To integrate a scientific markup language into SWiM<sup>+</sup>, knowledge from the document markup must be made accessible to services. In the prototype, it has proven to be practicable to extract knowledge required by services to an explicit representation as RDF triples in terms of the system ontology of the respective markup language.

Following the assumptions of the developers of PHYSML [HKS06] that knowledge is structured similarly in many sciences (see section 2.2), I plan to model the system ontologies of different scientific markup languages uniformly as specialisations of an abstract “upper system ontology” (USO) — a variation on the term “upper ontology”, which the IEEE Standard Upper Ontology Working Group defines as an ontology “limited to concepts that are meta, generic, abstract and philosophical, and therefore are general enough to address (at a high

level) a broad range of domain areas” [iee06]. Many services (see the respective work packages) are not specific to a certain scientific domain. If there are traits common to all scientific system ontologies, they can be modeled in the USO, and one and the same service can utilise knowledge from all scientific markup formats by accessing it in terms of the USO<sup>22</sup>. Such common traits will be identified during modeling the individual system ontologies; so far, I assume that they include:

- The basic concepts of the three-level stack described in section 2.2: objects, statements, and theories
- A generic “containment” relation: Theories contain sub-theories and statements, statements contain sub-statements (e.g. proof steps that are proofs in itself) and objects.
- A generic “dependency” relation between theories and statements, together with some generic refinements: the import relation between theories, the relation of an example to the statement it illustrates, and the relation between an evidence and the assumption it substantiates<sup>23</sup>.

To model the system ontologies, I will learn more about ontology design, starting with theoretical foundations [BCM+03] and practical courses (for example [CO-05]). The system ontology of OMDOC needs to be completed to cover the whole specification of the language: abstract data types (module *ADT* [Koh06c, chapter 16]), complex theories and development graphs (modules *CTH* and *DG* [chapter 18]), as well as containment relations below statement level are missing in the current implementation of the ontology. This needs not be done first, though, as there are already several statement types and specialisations of the dependency relation that can be used by services. Therefore, I will start to model the system ontology of a second markup language, most likely CNXML, in an early phase of the overall development. Both the CNXML and the OMDOC system ontology will be modeled as specialisations of the USO, whose development I will start in parallel. CNXML, though not targeted at a specific science, qualifies as the second format to be supported in SWIM<sup>+</sup>, because it features objects (e.g. equations) and statements (no theories, though), as well as a containment and dependency relation<sup>24</sup>, but at the same time it is small and manageable, as it is less formal than OMDOC. There is already a full system ontology, modeled by CH. MÜLLER and N. MÜLLER of our work group<sup>25</sup>.

#### 4.4 WP-M: Mapping from XML to Ontologies and Back

The prototype’s method of mapping from OMDOC’s document model to its system ontology and thus from XML elements and attributes to RDF triples using terms from the system ontology (see section 3.3) lacks scalability in the context of supporting more than one markup language. Instead of expressing the mapping in a Turing-complete programming language — Java or XSLT, for example, — a formal approach with less expressivity but stronger

<sup>22</sup>A query for all “statements” in a knowledge base about physics and mathematics would then return physical experiments as well as mathematical proofs.

<sup>23</sup>Laws in natural sciences cannot be proven formally; they can only be corroborated by observations [HKS06]. In mathematics, this relation would, of course, be refined to the “proof–proves–theorem” relation.

<sup>24</sup>expressed via the *cnxn* (connection) element

<sup>25</sup>It is not yet implemented in OWL-DL, but in UML (Unified Modeling Language), which has a better tool support. UML can, however, be mapped to OWL [Obj06].

constraints seems suitable for our case, where a rather strict document model (as opposed to rich text XML formats, for instance) constrains the input of such a conversion and a well-defined ontology constrains its output. A standard solution to that problem is not yet available, but there is research towards it: *WEESA* [RGJ05], a powerful declarative language that defines a flexible mapping from XML to RDF using classes and properties from a given OWL ontology and XPath expressions for extraction, looks promising and will be evaluated for integration into the core of SWiM<sup>+</sup>. It is also desirable to reverse this mapping: Editing services can use it on the ontology-to-XML level to find out which XML elements represent a certain relation, and import assistants can use it to construct scaffolds of documents from imported RDF triples.

Furthermore, extending the system ontologies by sub-statements and objects requires extracting explicit knowledge not only on page level, but from sub-structures of pages, and allowing fragments of pages (represented as URI with a fragment identifier in XML, e.g. `statement-page#object`) as subjects or objects of RDF triples.

#### 4.5 WP-OS: An Ontology of Collaborative Environments

Not only will services access knowledge from documents, but also knowledge about user interaction with the collaborative system and other users. If, for example, a learning ontology like LOM<sup>26</sup> is employed, which allows to express the difficulty of a statement, the latter could be estimated by counting the number of comments annotated as “questions” to the page containing that statement. The terms “page”, “comment”, as well as “user”, “edit”, and “version” are independent from the scientific markup languages and their system ontologies; they are part of the system ontology of the collaborative environment. Therefore, I will extend the wiki ontology that is already available in *IkeWiki* and enable services to reason over the union of both the documents’ and the environment’s system ontology.

#### 4.6 WP-AV: Planning Services using Added-Value Analysis

A. KOHLHASE and N. MÜLLER point out that the “added value” of a service is relative to a *core* solution to a specific *core* problem [KM07]. According to their definitions, the SWiM prototype offers a solution to the core problem of making semantically marked up documents editable in a collaborative environment. From a user’s micro-perspective, there is still a lack of motivation: The sacrifices the user has to make need to be identified and utilised as triggers for specifying requirements for added-value services.

To discover potential added-value services, A. KOHLHASE and N. MÜLLER propose the following design method called “added-value analysis” [KM07]:

1. Identify a core problem (e.g. editing content) from the “micro-perspective” of an individual user
2. Give a solution (i.e. an added-value service) to that problem.
3. Analyse the solution: Identify its benefits and sacrifices.
4. Both benefits and sacrifices may spawn new core problems and thus ideas for further added-value services; go to step 1.

---

<sup>26</sup>Learning Objects Metadata; see <http://ltsc.ieee.org/wg12/>

SWiM	Trigger	Core Problem	Core Solution	Value	
				Benefits	Sacrifices
Core	(initial)	collaborative work environment ...	social (web) software	<ul style="list-style-type: none"> <li>• joint development</li> <li>• easy publishing</li> </ul>	<ul style="list-style-type: none"> <li>• keeping track of others' contributions</li> </ul>
Core		for scientific knowledge ...	semantic markup	<ul style="list-style-type: none"> <li>• <b>stepwise formalisation</b></li> <li>• can be narrative</li> <li>• machine-understandable</li> </ul>	<ul style="list-style-type: none"> <li>• <b>hard to write</b></li> <li>• <b>not directly usable for automated deduction</b></li> </ul>
Core		... management	semantic work environment	<ul style="list-style-type: none"> <li>• reusable knowledge "atoms"</li> <li>• <b>explicit dependency links</b></li> </ul>	<ul style="list-style-type: none"> <li>• making knowledge explicit</li> </ul>
Core	<b>stepwise formalisation</b>	incremental editing based on division of labour	wiki	<ul style="list-style-type: none"> <li>• <b>easy linking</b></li> </ul>	<ul style="list-style-type: none"> <li>• danger of spam</li> </ul>
AVS <sup>a</sup>	<b>easy linking</b>	find possible targets	<ul style="list-style-type: none"> <li>• naïve auto-completion</li> </ul>	no more link typos	<ul style="list-style-type: none"> <li>• <b>too many inappropriate targets</b></li> </ul>
AVS	<b>too many inappropriate targets</b>	recommend allowed targets only	ontology-based auto-completion	<ul style="list-style-type: none"> <li>• concise list of appropriate targets</li> </ul>	
AVS	<b>explicit dependency links</b>	preserve dependencies upon editing	change management	<ul style="list-style-type: none"> <li>• less conflicts</li> <li>• improved consistency</li> </ul>	
AVS	<b>hard to write</b>	support the user's preferred editor	external editor interface	...	
AVS	<b>not directly usable for aut'd deduction</b>	extract logical structure	...		

<sup>a</sup>(potential) added-value service

Table 1: An incomplete added-value analysis of SWiM<sup>+</sup>

This method can be applied at any stage of the development<sup>27</sup>. If it is used for designing new services (instead of improving existing ones), identifying the benefits of the solution to a core problem rather means suggesting a solution and establishing requirements for it. I will use this method to scrutinise the introductory argumentation (“Why do we need OMDoc in a semantic wiki to manage scientific knowledge collaboratively?”), to develop new services for motivation and instant gratification of users (see section 1.4), but also to adapt existing services provided by external applications (see section 4.13) to the needs of individual authors when connecting them to SWiM<sup>+</sup>. To learn about as many micro-perspectives as possible, I will conduct a user survey.

While some services had already been specified before the method of added-value analysis came to my knowledge at the end of January 2007 (see the following work packages), I will also use the method to scrutinise these specifications. As an example, the beginning of an added-value analysis, starting from the original core problem of my work, is given in table 1.

#### 4.7 WP-SI: A Service Programming Interface

To facilitate the implementation of services after specifying them, a programming interface will be designed that provides a uniform way to access the documents in the wiki as well as the explicit knowledge extracted from them in terms of the upper system ontology, the individual system ontologies, and the wiki system ontology. The requirements for the interface

<sup>27</sup>personal communication with A. KOHLHASE and N. MÜLLER

will be determined by analysing the specifications of the planned services (see section 4.6) with regard to the following aspects:

- What kind of added value does it provide to individual users (“micro-perspective”) and to the community (“macro-perspective”), resp.?
- How much knowledge does it use? Just the text of the current page, semantic information about the current page, or (explicit or inferred) information about relations between multiple documents?
- Does it only depend on the system ontology behind the markup language that is used, or does it rely on a specific notation (e.g. XML)?
- Does it offer a one-step workflow, or does it require additional user feedback before further steps are executed?
- Should it be entirely implemented as a part of SWIM, or does it make sense to integrate an external tool instead?

#### 4.8 WP-SA: A Service for Auto-Completion

To assist the user in finding a target for a link, an auto-completion service for the source editor is planned (see also table 1): When the user enters a link like `<proof for="ring-`  (a link from a proof to the assertion it proves in OMDOC), where `_` indicates the cursor position, the relation represented by this XML attribute is determined from the XML-to-ontology mapping (see section 4.4). The ontology is queried for the range of the relation “*Proof-proves-?*”, leading to the answer *Assertion*, and then a pop-up will be displayed that lists all known instances of the class *Assertion* (i.e. all wiki pages that hold assertions) starting with “ring-”. The input for the service is the current XML element, or more exactly, the element name *e*, the link attribute name *a* and the (possibly incomplete) attribute value *v*. The output is a list of all possible (w.r.t. the ontology) link targets.

Most likely, this service will be the first one to be implemented on top of the evolving programming interface: It is fully ontology-based but works with any relation in a system ontology; thus it is suitable for testing the system ontologies while they are still under development. It does not use transitive closures of relations and thus does not require DL reasoning (see section 4.2). Furthermore, it is fairly easy to implement (except maybe for the pop-ups), and it is easy to evaluate its contribution to enhanced productivity in isolation: A small SWIM<sup>+</sup> installation with a few pages would be presented to two volunteers, who have to create links between several existing pages — with auto-completion enabled or disabled, respectively, — and the time difference would be measured.

#### 4.9 WP-SD: A Dependency Graph Navigation Service

The user shall not only be able to navigate along the graph given by the transitive closure of the dependency relation using dynamic navigation links, he shall also be able to explore dependencies interactively — especially in an educational setting. Suppose that the user is currently reading a page about the (mathematical) theory *ring*, which depends (via import) on *group* and *monoid*, with *group* depending on *monoid* and *monoid* in turn depending on *semigroup* (see figure 5). In this case the wiki shall not only display navigation links to the

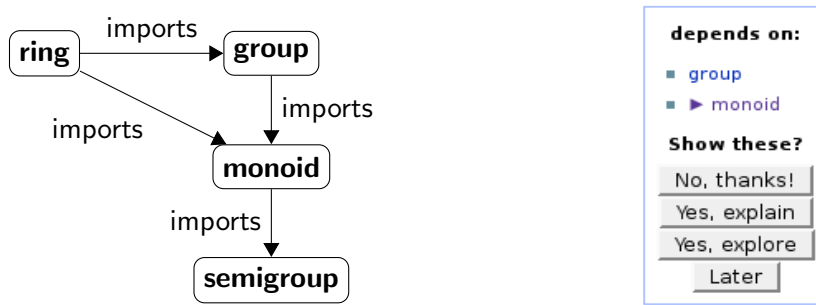


Figure 5: A dependency graph (left); navigation buttons (right)

direct dependencies *group* and *monoid*, but it shall also provide unobtrusive buttons that allow the user to give one of the following acknowledgments:

**No, thanks!** “I already know *group* and *monoid*, please let me just read about *ring*.”

**Explain** “Please show me *group* and *monoid* so that I can learn about *ring*’s prerequisites.” — *group* and *monoid* will be displayed.

**Explore** “Please show me *all* prerequisites for *ring*.” — In our example, these are *group*, *monoid*, and *semigroup*, which could be opened in separate windows or serialised into one page.

**Suspend** “I want to know about *group* and *monoid*, but only later.” — The system keeps a notice in the user’s profile that she wants to read *group* and *monoid* sometime. Reminder links to suspended concepts are shown on a separate navigation bar.

Not only the last case should be recorded — the others are interesting as well for *social bookmarking*. For example, if many users requested a concept *c* to be explained, the system could default to display not only the direct dependencies but also the level-two dependencies, for it seems that *c* is too difficult for only being explained shallowly.

Furthermore, the system will not only keep track of which concepts the user wants to be explained, but also which concepts the user has already learned. For each concept, a button will be offered for telling the system “I have learned this”. Links to concepts learned can then be disabled or displayed in a more unobtrusive colour than links to concepts that are new to the user. Concretely, this service comprises two steps:

1. The first step (making suggestions, presenting the action buttons) requires as input the currently viewed concept *c*, a reference to the dependency graph, and, possibly later, the user’s preferences and background knowledge. Its output is a tree *t* of concepts that *c* depends on. DL reasoning is needed in this step because of transitivity.
2. The second step (displaying wanted pages, bookmarking) requires as input *c* and *t* from step 1, and the information which button the user pressed. It returns the desired pages to the user and keeps a bookmarking record in the knowledge base.

## 4.10 WP-SM: A Dependency Management Service

If there are explicit dependency links between wiki pages, changes to the concept depended upon can render the other concept invalid. If, for example, in SWIM<sup>+</sup>, a mathematical theory  $t$  depends on a theory  $t'$ , we cannot exclude that modifying a definition in  $t'$  breaks  $t$  because some proof in  $t$  relies on it. While the fact that knowledge models in semantic social software “slowly and dynamically evolve over time”, leading to “inaccuracies and even inconsistencies” [Sch06b], has been observed by SCHAFFERT and others, there is not yet a solution for preserving dependencies in semantic wikis.

In a collection of scientific knowledge, I do, however, consider dependency between statement and theories one of the most important relations — in accordance with current research on theory management (see section 2.2.1). In mathematics, for example, known results are often generalised to find the “real” dependencies, mathematical theories and books are rewritten to make dependencies minimal. Note that it is impossible to *formally* verify all dependencies, though. In OMDOC, for example, it is sufficient for proofs of theory inclusions to fulfil certain structural properties; therefore, a dependency could formally be broken but seem intact to the system anyway. In natural sciences, there are no proofs altogether; successful experiments can only give evidence of an assumption. Therefore, SWIM<sup>+</sup> services can only rely on the dependency links asserted in the semantic markup to help users to maintain dependencies.

As OMDOC and other scientific markup languages are both structured data formats and ontology languages in their respective domains (see section 3.4), results of research on change management in these areas can be applied to SWIM<sup>+</sup>: N. MÜLLER of our work group aims at realising a practicable ontology-based change management for OMDOC and other structured document formats on top of informal document engineering processes [Mül06]. Change management for generic structured documents with semantic interrelations has been realised for MMiSSI<sup>A</sup>TEX [KBLL<sup>+</sup>04], a structured L<sup>A</sup>T<sub>E</sub>X-based format whose system ontology served me as an inspiring example for modeling OMDOC’s system ontology. Change management for mathematical and logic-based documents with strong dependencies modeled by development graphs (see section 2.2.1) has been formally investigated by HUTTER [Hut04], and change management for OWL ontologies is described by KLEIN et al., who differentiate “between version relations [between the definitions of concepts . . . in the original version of the ontology and those in the new version] and conceptual relations inside an ontology” [KFKO02].

As long as a ready-to-use change management solution for integration into SWIM<sup>+</sup> is not available, though, I aim at providing an incomplete but usable change management *assistance* service. To that end, I assume that those hyperlinks that represent dependencies (as specified in the ontology-to-XML mapping; see section 4.4) do not point to concepts in general, but to certain versions of them<sup>28</sup>. When an author enters a link to a theorem `thm`, for example, this reference will be stored internally as `thm/n`, where  $n$  is the number of the latest version. If the service encounters a versioned link that no longer points to the most recent version, the user is asked whether to have it updated.

1. The input to the first step is the currently viewed concept  $c$ . For each concept  $d_i$  linked from  $c$ , the version number  $v_i$  in the link is compared to the most recent version number  $v'_i$ . If they differ,  $(d_i, v'_i)$  is appended to the output. The user interface is the usual editing box, with indicators next to each updatable link  $d_i$ .

---

<sup>28</sup>Old versions of a page are available for free in a wiki, and the concept of a “version” will be an integral part of the wiki system ontology.

2. If the user clicks on such an indicator, a pop-up dialog with links to  $d_i/v_i$  (i.e. the old version of page  $d_i$ ),  $d_i/v'_i$  and a comparison page  $\text{diff}(d_i/v_i, d_i/v'_i)$  as well as two buttons labeled “keep” and “update” appears<sup>29</sup>. The user’s reaction to this dialog serves as input for the third step.
3. If the user wants to have the link  $d_i/v_i$  updated, its source code is replaced by  $d_i/v'_i$ .

Manual confirmation of an update should only be demanded when the semantics of  $d_i$  has changed from version  $v_i$  to  $v'_i$ . As this is non-trivial to decide, the user who makes a change should have the possibility to classify his change<sup>30</sup>.

A second service handles the opposite case: It should warn the user upon changing a concept  $e$  that many other concepts  $f_i$  depend on, that changing the semantics of  $e$  might affect the integrity of the  $f_i$ :

1. The only input argument to the service is the current concept  $e$ . The service queries the knowledge base for all concepts  $f_i$  depending on  $e$  — possibly also transitively, if DL reasoning is employed.
2. The output is the list  $f_{ii}$ . To the user, this might be presented as a warning like “ $n$  concepts depend on the one you are editing”. If the user makes a major change (w.r.t. the classification mentioned above) that affects many dependents, SWiM<sup>+</sup> could offer copying  $e$  to a new concept  $e'$ , which is to be specified as another service.

#### 4.11 WP-SE: An Edit-in-Place Service

Users can already split large OMDOC documents into their individual concepts in the SWiM prototype, but in some cases, even a single concept can be too large to edit — consider a complex proof with many steps or case distinctions. Some wikis, such as *MediaWiki*, allow for editing individual paragraphs of pages [Wik06a]. The Edit-in-place interface [The06] of *Rhaptos* (the software run by *Connexions*; cf. page 5) — is even more user-friendly; it is particularly suitable for editing or inserting sections in large XML documents with shallow structures. It is optimised for the CNXML format, but the idea can also be adapted to other formats whose structures are not too deeply-nested<sup>31</sup>. The documents are displayed in a near-WYSIWYG view, but clicking on a section replaces its view by a text area containing its XML source. The section can be deleted or edited and saved by an asynchronous request to the server.

The user interface to the edit-in-place service will be embedded into the presentation mode: If the user has the permission to edit a page, an “edit” button is added to every section. The user interacts with the service in three steps:

1. In the first step, the current page is structured section-wise. This is implemented as a part of the transformation of the page for presentation. To determine which XML elements denote sections, the service first queries the system ontology of the language of

<sup>29</sup>An alternative user interface could allow the user to select multiple links with check boxes and have all of them updated at once.

<sup>30</sup>Many wikis distinguish between “major” and “minor” changes. More sophisticated classifications of changes in the context of ontologies are discussed in [KFKO02, PT05, Mül06].

<sup>31</sup>While OMDOC can be deeply nested, SWiM<sup>+</sup> will not advise users to do so, but rather to split documents into several small, reusable wiki pages.

the current page for subclasses of the “section” class from the upper system ontology; then it queries the ontology-to-XML mapping (section 4.4) for those XML elements that represent them. The output is a presentational view of the page, with each section marked (e.g. by a frame) and prepared to be edited upon user’s request via embedded JavaScript code.

2. The user triggers the second step by selecting a section for editing. The service is provided an identifier of the section (e.g. its *xml:id* or an XPath expression). The presentation view of the respective section is replaced by an editing box with the source code of the section’s text and buttons for saving and canceling the edit as well as deleting the section. “Save” and “Delete” requests are sent to the server, while “Cancel” can be handled on the client by just replacing the editing box with the cached presentation view.
3. If the user requests the section to be saved or deleted, the third step is triggered. The source code of the page, with the respective section replaced or deleted, is stored to the page database. RDF triples whose subjects are fragments within the section (see section 4.4) are re-generated. While it is desirable to have only the replaced section re-rendered for presentation — instead of the whole page —, it is open whether that is possible using the available transformations of the respective scientific markup language to XHTML.

#### **4.12 WP-SP: Scientific Publishing with SWiM+**

One application of scientific markup is preparing documents for publishing. For example, narrative OMDOC documents are ready for transformation to XHTML+MathML or L<sup>A</sup>T<sub>E</sub>X (and to PDF from there) for publishing on the web or on paper [Koh06c]. This applies to the intended uses of SWiM<sup>+</sup> in education as well as research. Therefore, I will establish requirements for SWiM<sup>+</sup> as a publishing tool, following BERGHEL [Ber99] and others, and integrate existing publishing workflows as services.

#### **4.13 WP-SO: Other Services**

This work package serves as a placeholder for other services. The services specified so far do not yet cover all intended use cases of SWiM<sup>+</sup> given in section 1.1. I expect solutions for the remaining use cases as a result of an added-value analysis (see section 4.6). These solutions need not necessarily be services that will be implemented within SWiM<sup>+</sup>, but connecting external web services or applications to SWiM<sup>+</sup> could also make sense in cases where these have already been developed; consider, for example, semantic search for mathematical formulae, which is already provided by *MathWebSearch* [Kc06], or WYSIWYG editing for OMDOC, provided by the *Sentido Mozilla* extension [Pal06].

#### **4.14 WP-CS: Case Study as a Tool for Scientists**

Applying the results achieved with OMDOC for mathematical knowledge management to other sciences is a major concern of our work group, in cooperation with other groups from physics, geosciences and chemistry. That gives me the opportunity to evaluate SWiM<sup>+</sup> as a tool that supports scientists developing theories (see section 1.1) in an interdisciplinary

setting. In particular, I will concentrate on evaluating the feasibility of integrating common traits of scientific knowledge in the upper system ontology and on general services relying on that ontology — such as the dependency management (see section 4.10) — across the borders of individual sciences.

#### 4.15 WP-CE: Case Study in Education

$\text{s}\text{T}\text{E}\text{X}$  is a package for structural semantic markup of  $\text{L}\text{A}\text{T}\text{E}\text{X}$  documents, which can then be converted to OMDOC [Koh06b]. M. KOHLHASE maintains his slides and lecture notes for the “General Computer Science I & II” course at Jacobs University Bremen in this format; so far, there are more than 1,000  $\text{s}\text{T}\text{E}\text{X}$  modules, which correspond to (small) OMDOC theories. This corpus will be imported into a SWiM<sup>+</sup> installation accessible to the students; in particular, the dependency graph navigation service (see section 4.9) and facilities that allow students to interact with the instructor, such as the commenting feature proposed in section 4.5, will be evaluated.

#### 4.16 Preliminary Schedule

Figure 6 shows the preliminary schedule for completing my Ph.D. thesis within three years. Most of the work packages are divided into a “core” and an “additional” part. While the major work will be done in the core part, the second period is scheduled for extending services by additional features or extending the system ontologies by additional concepts required by the evolving services. Furthermore, I expect feedback for improving the services from the case studies.

Regarding the publishing of my results, I plan to present and to discuss the original idea and first experiences at relevant workshops. I will publish later results as workshop and conference papers and finally compile the Ph.D. thesis from my overall results.

### 5 Conclusion and Outlook

At the end of my work, the case study WP-CS will have shown whether SWiM<sup>+</sup> can serve scientists of several areas as a powerful tool for developing and publishing new theories – supported by dependency management and other services. The case study WP-CE will have shown whether SWiM<sup>+</sup> improves education by supporting self-directed learning and by allowing instructors and students to interact about the syllabus of a course. Furthermore, the reusability of the knowledge managed within the SWiM<sup>+</sup> environment on the Semantic Web will have been evaluated; a concrete case study has not yet been drafted.

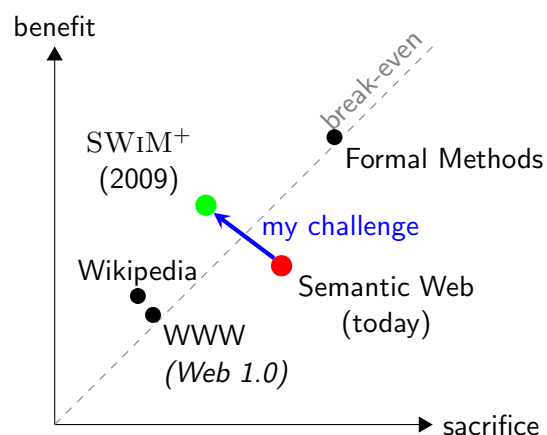
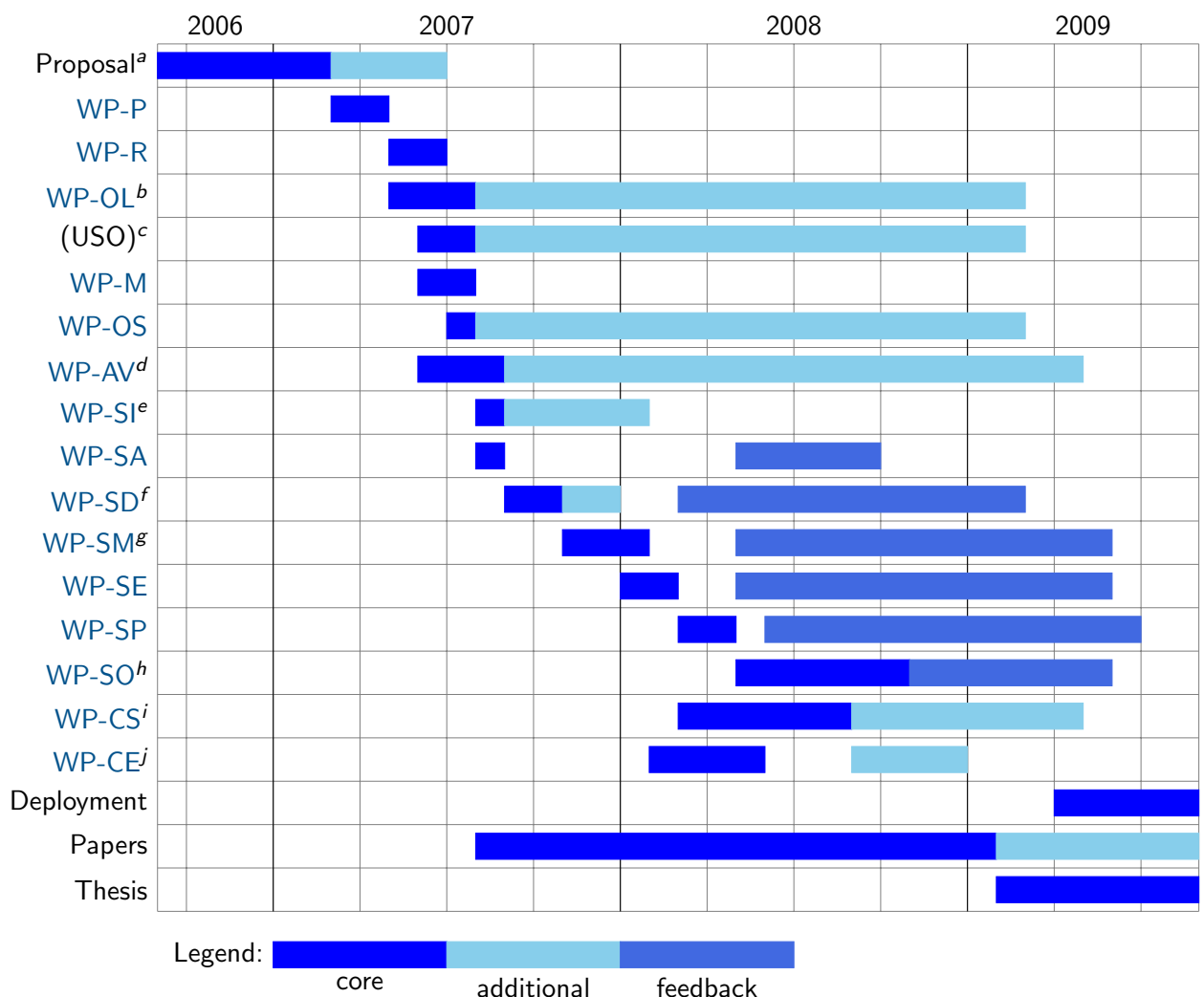


Figure 7: The Semantic Web Challenge



<sup>a</sup>... and other preparations. The additional time comprises workshop contributions derived off the proposal and first experiences gained.

<sup>b</sup>The ontology modeling process is needed for every language that is integrated into SWIM<sup>+</sup>.

<sup>c</sup>The upper system ontology evolves with the individual system ontologies.

<sup>d</sup>While one added-value analysis is needed in the beginning, the services will be evaluated in subsequent analyses.

<sup>e</sup>I expect that the service programming interface needs to be extended while the first services are implemented to meet their requirements.

<sup>f</sup>Implementing the navigation features beyond the basic requirements may take some more time.

<sup>g</sup>These plans may change once N. MÜLLER's management of change solution *locutor* is ready for integration into SWIM<sup>+</sup>.

<sup>h</sup>The real effort needed for this placeholder package is not yet known, as the "other" services have not been planned yet.

<sup>i</sup>The schedule for the projects with our cooperation partners is not yet definitive.

<sup>j</sup>The main part of this study will be conducted during the spring semester 2007; a second run will take place in the fall semester.

Figure 6: Preliminary Schedule

I am convinced that SWiM<sup>+</sup> will demonstrate how the Semantic Web can be improved – by providing a higher benefit to the users through various services while reducing the investment the users have to make (see figure 7). The achievements made by services running in the “testbed” of SWiM<sup>+</sup> can then be transferred to the “large” Semantic Web, following an idea by SCHAFFERT [Sch06b]. If my approach of supporting interoperability using system ontologies — both of markup languages and of the work environment — as an abstraction layer succeeds, porting the SWiM<sup>+</sup> services to other semantic work environments will become possible by creating an upper system ontology for wikis and other environments.

## References

- [AA05] David Aumüller and Sören Auer. Towards a semantic wiki experience – desktop integration and interactivity in WikSAR. In *Proc. of 1st Workshop on The Semantic Desktop – Next Generation Personal Information Management and Collaboration Infrastructure, Galway, Ireland, Nov. 6th, 2005*, 2005.
- [ABC<sup>+</sup>03] Ron Ausbrooks, Stephen Buswell, David Carlisle, Stéphane Dalmas, Stan Devitt, Angel Diaz, Max Froumentin, Roger Hunter, Patrick Ion, Michael Kohlhase, Robert Miner, Nico Poppelier, Bruce Smith, Neil Soiffer, Robert Sutor, and Stephen Watt. Mathematical Markup Language (MathML) version 2.0 (second edition). W3C recommendation, World Wide Web Consortium, 2003. Available at <http://www.w3.org/TR/MathML2>.
- [AH02] Serge Autexier and Dieter Hutter. Maintenance of Formal Software Developments by Stratified Verification. In *Proceedings 9th International Conference on Logic for Programming Artificial Intelligence and Reasoning*, LNAI. Springer-Verlag, 2002.
- [BCC<sup>+</sup>04] Stephen Buswell, Olga Caprotti, David P. Carlisle, Michael C. Dewar, Marc Gaetano, and Michael Kohlhase. The Open Math standard, version 2.0. Technical report, The Open Math Society, 2004. <http://www.openmath.org/standard/om20>.
- [BCG<sup>+</sup>06] Michel Buffa, Gaël Crova, Fabien Gandon, Claire Lecompte, and Jeremy Passeron. SweetWiki: Semantic WEb Enabled Technologies in Wiki. In Völkel et al. [VSD06].
- [BCM<sup>+</sup>03] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.
- [Ber99] Hal Berghel. Digital village: Value-added publishing. *Communications of the ACM*, 42(1):19–23, 1999.
- [BF06] Jon Borwein and William M. Farmer, editors. *Mathematical Knowledge Management, MKM’06*, number 4108 in LNAI. Springer Verlag, 2006.
- [BPSM<sup>+</sup>04] Tim Bray, Jean Paoli, C. M. Sperberg-McQueen, Eve Maler, and François Yergeau. Extensible Markup Language (XML) 1.0 (Third Edition). W3C

- Recommendation, World Wide Web Consortium, February 2004. Available at <http://www.w3.org/TR/2004/REC-xml-20040204>.
- [CML07] Chemical markup language (CML). <http://cml.sourceforge.net/>, seen January 2007.
- [CO-05] Ontology design patterns and problems (iswc 2005 tutorial). <http://www.co-ode.org/events/tutorials/iswc2005/>, 2005.
- [ER06] Juhani Eronen and Juha Rönig. Graphingwiki – a semantic wiki extension for visualising and inferring protocol dependency. In Völkel et al. [VSD06].
- [FGT92] William Farmer, Josuah Guttman, and Xavier Thayer. Little theories. In D. Kapur, editor, *Proceedings of the 11<sup>th</sup> Conference on Automated Deduction*, volume 607 of *LNCS*, pages 467–581, Saratoga Springs, NY, USA, 1992. Springer Verlag.
- [HG05] Brent Hendricks and Adan Galvan. The Connexions Markup Language (CNXML). <http://cnx.org/technology/cnxml/>, 2005. Seen July 2006.
- [HKS06] Eberhard Hilf, Michael Kohlhase, and Heinrich Stamerjohanns. Capturing the content of physics: Systems, observables, and experiments. In Borwein and Farmer [BF06].
- [HKV06] Heiko Haller, Felix Kugel, and Max Völkel. iMapping wikis – towards a graphical environment for semantic knowledge management. In Völkel et al. [VSD06].
- [Hut04] Dieter Hutter. Towards a generic management of change. In Christoph Benzmüller and Wolfgang Windsteiger, editors, *Computer-Supported Mathematical Theory Development*, number 04-14 in RISC Report Series, pages 7–18. RISC Institute, University of Linz, 2004. Proceedings of the first “Workshop on Computer-Supported Mathematical Theory Development” held in the frame of IJCAR’04 in Cork, Ireland, July 5, 2004. ISBN 3-902276-04-5. Available at <http://www.risc.uni-linz.ac.at/about/conferences/IJCAR-WS7/>.
- [iee06] IEEE Standard Upper Ontology Working Group. <http://suo.ieee.org/>, seen December 2006.
- [KBLL<sup>+</sup>04] Bernd Krieg-Brückner, Arne Lindow, Christoph Lüth, Achim Mahnke, and George Russell. Semantic interrelation of documents via an ontology. In G. Engels and S. Seehusen, editors, *DeLFI 2004, Tagungsband der 2. e-Learning Fachtagung Informatik, 6.-8. September 2004, Paderborn, Germany*, volume P-52 of *Lecture Notes in Informatics*, pages 271–282. Springer-Verlag; D-69121 Heidelberg, Germany; <http://www.springer.de>, 2004.
- [Kc06] Michael Kohlhase and Ioan Şucan. A search engine for mathematical formulae. In Tetsuo Ida, Jacques Calmet, and Dongming Wang, editors, *Proceedings of Artificial Intelligence and Symbolic Computation, AISC’2006*, number 4120 in *LNAI*, pages 241–253. Springer Verlag, 2006.

- [KFKO02] Michel C. A. Klein, Dieter Fensel, Atanas Kiryakov, and Damyan Ognyanov. Ontology versioning and change detection on the web. In Asunción Gómez-Pérez and V. Richard Benjamins, editors, *EKAW*, volume 2473 of *Lecture Notes in Computer Science*, pages 197–212. Springer, 2002.
- [KK06] Andrea Kohlhase and Michael Kohlhase. Communities of practice in mkm: An extensional model. In Borwein and Farmer [BF06].
- [KM07] Andrea Kohlhase and Normen Müller. Added-value: Getting people into semantic work environments. In *Emerging Technologies for Semantic Work Environments: Techniques, Methods, and Applications*. Idea Group, 2007. To appear; chapters under review.
- [Koh05a] Michael Kohlhase, editor. *Mathematical Knowledge Management, MKM’05*, number 3863 in LNAI. Springer Verlag, 2005.
- [Koh05b] Michael Kohlhase. OMDoc: Open Mathematical Documents. In Fred de Vries, Graham Attwell, Raymond Elferink, and Alexandra Tödt, editors, *Open Source for Education in Europe: Research and Practise*, pages 137–143, Heerlen, The Netherlands, November 2005. Open Universiteit Nederland, Open Universiteit Nederland. Proceedings at <http://hdl.handle.net/1820/483>.
- [Koh06a] Andrea Kohlhase. The User as Prisoner: How the Dilemma Might Dissolve. In Martin Memmel, Eric Ras, and Stephan Weibelzahl, editors, *2nd Workshop on Learner Oriented Knowledge Management & KM Oriented e-Learning*, number 2, pages 26–31, 2006. Online Proceedings at <http://cnm.open.ac.uk/projects/ectel06/pdfs/ECTEL06WS68d.pdf>.
- [Koh06b] Michael Kohlhase. s<sub>TEX</sub>: A L<sup>A</sup>T<sub>E</sub>X-based workflow for OMDoc. In *OMDOC – An open markup format for mathematical documents [Version 1.2]* [Koh06c], chapter 26.15.
- [Koh06c] Michael Kohlhase. *OMDOC – An open markup format for mathematical documents [Version 1.2]*. Number 4180 in LNAI. Springer Verlag, 2006.
- [Kro03] Aaron Krowne. An architecture for collaborative math and science digital libraries. Master’s thesis, Virginia Tech, 2003. Available at <http://scholar.lib.vt.edu/theses/available/etd-09022003-150851/>.
- [Lan07] Christoph Lange. SWiM – a semantic wiki for mathematical knowledge management. Technical report, International University Bremen, 2007.
- [LS99] Ora Lassila and Ralph R. Swick. Resource description framework (RDF) model and syntax specification. W3c recommendation, World Wide Web Consortium (W3C), 1999. <http://www.w3.org/TR/1999/REC-rdf-syntax>.
- [MAH06] Till Mossakowski, Serge Autexier, and Dieter Hutter. Development graphs – proof management for structured specifications. *Journal of Logic and Algebraic Programming*, 67(1–2):114–145, 2006.
- [McK05] Erin McKean, editor. *The New Oxford American Dictionary, Second Edition*. Oxford University Press, May 2005.

- [MEG<sup>+</sup>03] Luke McDowell, Oren Etzioni, Steven D. Gribble, Alon Halevy, Henry Levy, William Pentney, Deepak Verma, and Stani Vlasseva. Mangrove: Enticing ordinary people onto the semantic web via instant gratification. In *3<sup>rd</sup> International Semantic Web Conference (ISWC 2003)*, 2003.
- [MGP<sup>+</sup>06] Erica Melis, George Goguadze, Alberto González Palomo, Adrian Frischauf, Martin Homik, Paul Libbrecht, and Carsten Ullrich. OMDoc in ActiveMath. In *OMDOC – An open markup format for mathematical documents [Version 1.2]* [Koh06c], chapter 26.8.
- [Mül06] Normen Müller. An Ontology-Driven Management of Change. In *Wissens- und Erfahrungsmanagement LWA (Lernen, Wissensentdeckung, Aktivität) conference proceedings*, 2006.
- [MvH04] Deborah L. McGuinness and Frank van Harmelen. OWL web ontology language overview. W3c recommendation, W3C, February 2004. Available at <http://www.w3.org/TR/2004/REC-owl-features-20040210/>.
- [Obj06] Object Modeling Group. *Ontology Definition Metamodel, Sixth Revised Submission to OMG/ RFP ad/2003-03-40*, May 2006.
- [ODM<sup>+</sup>06] Eyal Oren, Renaud Delbru, Knud Möller, Max Völkel, and Siegfried Handschuh. Annotation and navigation in semantic wikis. In Völkel et al. [VSD06].
- [Ont07a] Ontoworld.org wiki. <http://ontoworld.org>, seen January 2007.
- [Ont07b] Sites using Semantic MediaWiki – ontoworld.org. [http://ontoworld.org/index.php?title=Semantic\\_MediaWiki&oldid=25046](http://ontoworld.org/index.php?title=Semantic_MediaWiki&oldid=25046), seen January 2007.
- [Pal06] Alberto González Palomo. Sentido: an authoring environment for OMDoc. In *OMDOC – An open markup format for mathematical documents [Version 1.2]* [Koh06c], chapter 26.3.
- [PT05] Peter Plessers and Olga De Troyer. Ontology change detection using a version log. In Yolanda Gil, Enrico Motta, V. Richard Benjamins, and Mark A. Musen, editors, *International Semantic Web Conference*, volume 3729 of *Lecture Notes in Computer Science*, pages 578–592. Springer, November 2005.
- [RGJ05] Gerald Reif, Harald Gall, and Mehdi Jazayeri. WEESA: Web engineering for semantic web applications. In Allan Ellis and Tatsuya Hagino, editors, *WWW*, pages 722–729. ACM, May 2005.
- [Sch06a] Sebastian Schaffert. IkeWiki: A semantic wiki for collaborative knowledge management. Technical report, Salzburg Research Forschungsgesellschaft, 2006. Available at [http://www.wast1.net/download/paper/schaffert06\\_ikewiki.pdf](http://www.wast1.net/download/paper/schaffert06_ikewiki.pdf).
- [Sch06b] Sebastian Schaffert. Semantic social software – semantically enabled social software or socially enabled semantic web? In Sure and Schaffert [SS06].

- [SPG<sup>+</sup>06] Evren Sirin, Bijan Parsia, Bernardo Cuenca Grau, Aditya Kalyanpur, and Yarden Katz. Pellet: A practical OWL-DL reasoner. *Journal of Web Semantics*, 2006. To appear; draft available at <http://www.mindswap.org/papers/PelletJWS.pdf>.
- [Spi06] Nova Spivack. The metaweb. [http://novaspivack.typepad.com/nova\\_spivacks\\_weblog/the\\_metaweb/](http://novaspivack.typepad.com/nova_spivacks_weblog/the_metaweb/), 2003–2006. Seen March 2006.
- [SS06] York Sure and Sebastian Schaffert, editors. *Semantics 2006: From Visions to Applications: Semantics – The New Paradigm Shift in IT, Vienna, Austria, November 2006*, November 2006.
- [TCC04] Roberto Tazzoli, Paolo Castagna, and Stefano Emilio Campanini. Towards a Semantic WikiWikiWeb. In *3<sup>rd</sup> International Semantic Web Conference (ISWC 2004)*, Hiroshima, Japan, 2004.
- [The06] The Connexions Team. Connexions: Help on editing modules. <http://cnx.org/help/EditingModules>, 2006. Seen March 2006.
- [UCI<sup>+</sup>06] Victoria Uren, Philipp Cimiano, Jose Iria, Siegfried Handschuh, Maria Vargas-Vera, Enrico Motta, and Fabio Ciravegna. Semantic annotation for knowledge management: Requirements and a survey of the state of the art. *Web Semantics: Science, Services and Agents on the World Wide Web*, 4(1):14–28, January 2006.
- [V<sup>+</sup>07] Max Völkel et al. Wiki interchange format – ontoworld.org. [http://ontoworld.org/wiki/Wiki\\_Interchange\\_Format](http://ontoworld.org/wiki/Wiki_Interchange_Format), seen February 2007.
- [VK06] Denny Vrandečić and Markus Krötzsch. Reusing ontological background knowledge in semantic wikis. In Völkel et al. [VSD06].
- [VKV<sup>+</sup>06] Max Völkel, Markus Krötzsch, Denny Vrandečić, Heiko Haller, and Rudi Studer. Semantic Wikipedia. In *Proceedings of the 15<sup>th</sup> international conference on World Wide Web, WWW 2006, Edinburgh, Scotland, May 23–26, 2006*, May 2006. Available at <http://www.aifb.uni-karlsruhe.de/WBS/hha/papers/SemanticWikipedia.pdf>.
- [VSD06] Max Völkel, Sebastian Schaffert, and Stefan Decker, editors. *1st Workshop on Semantic Wikis*, volume 206 of *CEUR Workshop Proceedings*, Budva, Montenegro, June 2006.
- [W3C06] W3C semantic web. <http://www.w3.org/2001/sw/>, 2001–2006. Seen August 2006.
- [Wik06a] Section (from Wikimedia meta-wiki). <http://meta.wikimedia.org/w/index.php?title=Help:Section&oldid=480808>, December 2006.
- [Wik06b] Wikipedia, the free encyclopedia. <http://www.wikipedia.org>, 2001–2006.
- [Wik06c] Semantic Wiki (from Wikipedia, the free encyclopedia). [http://en.wikipedia.org/w/index.php?title=Semantic\\_Wiki&oldid=92323454](http://en.wikipedia.org/w/index.php?title=Semantic_Wiki&oldid=92323454), December 2006.
- [Zin06] Claus Zinn. Bootstrapping a semantic wiki application for learning mathematics. In Sure and Schaffert [SS06].