Dimensions of Formality:
A Case Study for MKM in Software Engineering

Andrea Kohlhase[1] and Michael Kohlhase[2] and Christoph Lange[2]

[1] German Research Center for Artificial Intelligence (DFKI)
Andrea.Kohlhase@dfki.de
[2] Computer Science, Jacobs University Bremen
{m.kohlhase,ch.lange}@jacobs-university.de

Abstract. We study the formalization of a collection of documents created for a Software Engineering project from an MKM perspective. We analyze how document and collection markup formats can cope with an open-ended, multi-dimensional space of primary and secondary classifications and relationships. We show that RDFa-based extensions of MKM formats, employing flexible "metadata" relationships referencing specific vocabularies for distinct dimensions, are well-suited to encode this and to put it into service. This formalized knowledge can be used for enriching interactive document browsing, for enabling multi-dimensional metadata queries over documents and collections, and for exporting Linked Data to the Semantic Web and thus enabling further reuse.

1 Introduction

The field of Mathematical Knowledge Management (MKM) tries to model mathematical objects and their relationships, their creation and publication processes, and their management requirements. In [CF09, 237 ff.] CARETTE and FARMER analyzed "*six major lenses through which researchers view MKM*": the document, library, formal, digital, interactive, and the process lens. Quite obviously, there is a gap between the formal aspects {"library", "formal", "digital"} – related to machine use of mathematical knowledge – and the informal ones {"document", "interactive", "process"} – related to human use.

In the FormalSafe project [For08] at the German Research Center for Artificial Intelligence (DFKI) Bremen a main goal is the integration of project documents into a computer-supported software development process. MKM techniques are used to bridge the gap between informally stated user requirements and formal verification. One of the FormalSafe case studies is based on the documents of the SAMS project ("Sicherungskomponente für Autonome Mobile Systeme [Safety Component for Autonomous Mobile Systems]", see [FHL+08]) at DFKI. The SAMS objective was to develop a safety component for autonomous mobile service robots and to get it certified as SIL-3 standard compliant in the course of three years. On the one hand, certification required the verification of certain safety properties in the code documents with the proof checker Isabelle [NPW02]. On the other hand, it necessitated the software development

process to follow the V-Model (fig. 1). This mandates e. g. that relevant document fragments get justified and linked to corresponding fragments in a successive document refinement process (the arms of the 'V' from the upper left over the bottom to the upper right and between arms in fig. 1).

The collection of SAMS documents (we call it "**SAMSDocs**" [SAM09]) promised an interesting case study for FormalSafe as system development with respect to the V-Model regime resulted in a highly interconnected collection of design documents, certification documents, code, formal specifications, and formal proofs. Furthermore, it was supposed that adding semantics to SAMSDocs would be comparatively easy as it was developed under a strong formalization pressure.
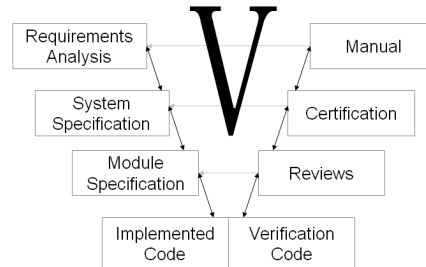

Fig. 1. Documents in the V-Model

In this paper we report on — and draw conclusions from — the SAMSDocs formalization, particularly the formalization of its LaTeX documents. In section 2, we document the process and detect inherent, distinct formality levels and the multi-dimensionality of the formalized structures. Real information needs (drawn from three use cases in the SAMS context) turn out in section 3 to be multi-dimensional. This motivates our exploration of multi-dimensional markup in section 4. Section 5 showcases the feasibility of multi-dimensional services with MKM technology enabled by multi-dimensional structured representations and section 6 concludes the paper.

2   Dimensions of Formality in SAMSDocs

In this paper, we are especially interested in the question *"What should we sensibly formalize in a document collection and can MKM methods cope?"*. Note that we understand "to formalize" as "making implicit knowledge explicit" and not as "to make s.th. fully formal".

The SAMS project was organized as a typical Software Engineering project, its collection of documents SAMSDocs therefore has a prototypical composition

Format	Files	#
LaTeX	*.tex	251
MS Word	*.doc	61
Isabelle	*.thy	33
Misra-C Code	*.c	40

Fig. 2. SAMSDocs

of distinct document types like contract, code, or manual. Thus, SAMSDocs presents a good base for a case study with respect to our question. In fig. 2 we can see the concrete distribution over used document formats in SAMSDocs. Requirements analysis, system and module specifications, reviews, and the final manual were mainly written in LaTeX, only roughly a sixth in MS Word. The implementation in Misra-C contains Isabelle theorem prover calls.

The first stinging, but unsurprising observation was that the *level of formality* of the documents in SAMSDocs varies considerably — because distinct

purposes create distinct formality requirements. For instance, the contract document serves as communication medium between the customer and the contractor. Here, underspecification is an important tool, whereas it is regarded harmful in the fine-granular module specifications and a fatal flaw in input logic for a theorem prover. Since this issue was already present in the set of LaTeX documents, we focused on just these.

For the formalization of this subset in SAMSDocs we used the sTeX system [Koh08], a semantic extension of LaTeX. It offers to both publish documents as high-quality human-readable PDF *and* as formal machine-processable OMDoc [Koh06] via LaTeXML [SKG+10]. Our formalization process revealed early on that previous sTeX applications (based on OMDoc 1.2) were too rigid for a stepwise semantic markup. But fortunately, sTeX also allows for the OMDoc 1.3 scheme of metadata via RDFa [ABMP08] annotations (see [Koh10]). In particular, we could 'invent' our own vocabulary for markup on demand without extending OMDoc. This new vocabulary consists of SAMSDocs-specific metadata properties and relationship types. We call the process of adding this *pre*-formal markup to SAMSDocs **(semantic) preloading**. Concretely, we extended sTeX to sTeX-SD (sTeX for SAMSDocs) by adding LaTeXML bindings for all SAMS specific TeX macros and environments used in SAMSDocs, thus enabling the conservation of the original PDF document layouts at the same time as the generation of meaningful OMDoc.
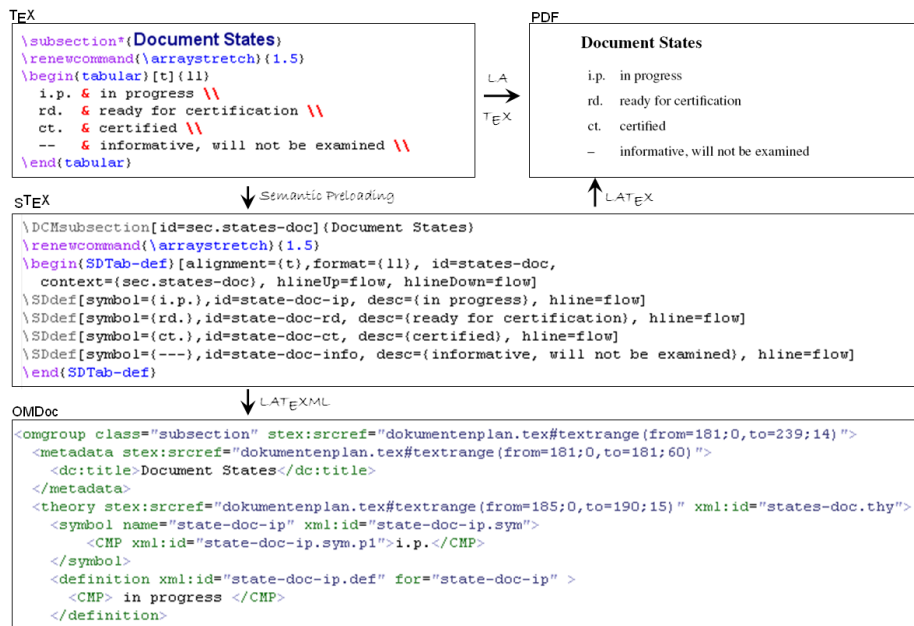

Fig. 3. The Formalization Workflow with sTeX-SD [translated by the authors]

Let us look at an example for such an SᴛᴇX extension within our formalization workflow (see fig. 3). We started out with a TᴇX document (upper left), which compiled to the PDF seen on the upper right. Here, we have a simple, two-dimensional table, which is realized with a LᴀTᴇX environment `tabular`. Semantically, this table contains a list of symbols for document states with their definitions, e. g. "i. B." for "in Bearbeitung [in progress]". As such definition tables were used throughout the project, we developed the environment `SDTab-def` and the macro `SDdef` as SᴛᴇX extensions. We determined the OMDoc output for these to be a symbol together with its definition element (for each use of `SDdef` in place of the resp. table row) and moreover, to group all of them into a theory (via using `SDTab-def`). Preloading the TᴇX table by employing `SDTab-def` and `SDdef` turned it into an SᴛᴇX document (middle of fig. 3) while keeping the original PDF table structure. Using LᴀTᴇXML on this SᴛᴇX document produces the OMDoc output shown in the lower area of fig. 3.

Mathematical, structural relationships have a privileged state in SᴛᴇX: their command sequence/environment syntax is analogous to the native XML element and attribute names in OMDoc. Since many objects and relationships induce formal representations for Isabelle, it seemed possible to semantically mark them up with a logic-inspired structure. But in the formalization process it soon became apparent that (important) knowledge implicit in SAMSDocs did not refer to the 'primary' structure aimed at with the use of SᴛᴇX. Instead, this knowledge was concerned with a space of less formal, 'secondary' classifications and relationships. Thus, our second observation pertains to the substance of formalizations. Even though we wanted to find out *what* we can sensibly formalize, we had assumed this to mean *how much* we can sensibly formalize. Therefore, we were rather surprised to find distinct **formality structures** realized in our SᴛᴇX extension. In the following we want to report on these structures.

We grouped the macros and environments of SᴛᴇX-SD in fig. 5 according to what induced them. Particularly, we distinguished the following triggers:
- "*objects*" — document fragments viewed as autonomous elements — and
- their net of relationships via the *collection*,
- *document*s and
- their *organization*al handling, and
- the *project* itself and thus, its own scheme of meaningful relationships.

For instance, in the system specification we marked a recap of a definition of the braking distance function for straight-ahead driving s_G as an object and referenced it from within the assertion seen in fig. 4. In the module specification s_G was then meticulously specified. This document fragment is connected to the original one via a refinement-relationship from the V-Model, which determined the creation process of the collection. Documents induce layout structures like sections or subsections and they are themselves organized for example under a version management scheme. In the workflow in fig. 3 we already showcased a project-specific element, the definition table, with its meaning. Interestingly, we can-

$$q(t) = s_G(v(t))$$
$$\frac{dq}{dt} = \frac{ds}{dv}\frac{dv}{dt}$$

Fig. 4. s is Braking Distance?

4

not compare formality in one group with the formality in another. For example, we cannot decide whether a document completely marked up with the object-induced structures is more formal than one fully semantically enhanced by the version management markup. As these grouped structures only interact relatively lightly, we can consider them as independent **dimensions of a formality space** that is reified in the formalization process of a document collection.

Concretely, SТEX-SD covers the following dimensions and consists of the listed extension macros/environments (with attributes in [·] where sensible):

Object Structures	Collection Structures
• **SDobject [id]:** Providing a document fragment with an identifier • **SDis [id, cat, for, follows, theory, imports, tab]:** Categorizing an object and relating it to other objects • **SDmore [id, cat, for]:** Overcoming linearity of documents through concatenation • **SDreferences [id, cd, refid]:** Referencing an object - via content dictionary (theory) and identifier - and reifying the document fragment itself • **SDreferencesNoObj [cd, refid]:** Referencing an object • **SDincludes [id, cd, from]:** Including an object as an exact copy	• **SemVMrel[id, rel, cd, refid]:** Relationships within the V-Model
	Project Structures
	• **SDTab-def, SDdef:** Definition tables as described for fig. 3 • **SDTab-paramUse, SDparamUse:** Parameter-Use Tables containing specification for parameters in a project-specific data type • **SDTab-reqs, SDreq:** Safety requirement tables with definitions of safety requirements and their dependencies • **SDTab-FMs, SDFM:** Code error detection tables describing errors, their effects, their detection, and their induced program error state
Organizational Structures • **VMchangelist, VMchange:** Version management • **VMcertification, VMcertified:** Review history management • **SDnode[id, type], SDpath[id, from, to]:** Data flow diagram	

Fig. 5. Formality Dimensions in SТEX-SD

Formalizing object structures is not always obvious, since many of the documents contain recaps or previews of material that is introduced in other documents/parts (e. g. to make them self-contained). Compare for example fig. 4 and fig. 6, which are actually clippings from the system specification "KonzeptBremsmodell.pdf". Note the use of s resp. s_G, *both* pointing in fig. 4 to the braking distance function for straight-ahead driving (which is obvious from the local context), whereas in fig. 6 s represents the general arc length function of a circle, which is different in principle from the braking distance, but coincides here.

$$a = s\,\frac{\sin\frac{\phi}{2}}{\frac{\phi}{2}} = s\,\mathrm{sinc}\,\frac{\phi}{2}$$

Fig. 6. Yet another Braking Distance s?

We also realized that SТEX itself had already integrated another formality dimension besides the logic-inspired one, the one concerned with document layout: A typical document layout is structured into established parts like sections or modules. If we want to keep this grouping information in the formal XML document, we might use SТEX's DCM package for annotating general document

| • DCM package: Annotating general document structures with Dublin Core and similar general-purpose metadata | *Document Structures* |

Fig. 7. The Document Formality Dimension in STEX

structures with Dublin Core (cf. [Dub08]) and similar general-purpose metadata. In the STEX box in fig. 3 we find for example the command `DCMsubsection` with attributes containing the title of the subsection and an identifier that can be used in the usual LATEX referencing scheme.

Finally, we would like to remark that the STEX-SD preloading process was executed as "*in-place formalization*" [SIM99]. It frequently considered several of the above dimensions for the object at hand at the same time. Therefore, the often applied metaphor of "formalization steps" does not mirror the formalization process in our case study. We found that the important aspect of the formalization was not its sequence per se, which we consider particular to the SAMSDocs collection, but the fact that the metaphor of 'steps' only worked within each single dimension of formality. In particular, there is no scale for formalization progress as distinct formality levels in distinct formality dimensions existed in a document at one point in time.

3   Multi-Dimensional Information Needs

We have shown that the formalization of knowledge results in an open-ended, multi-dimensional space of primary and secondary classifications and relationships. But are multi-dimensional document formalizations beneficial for services supporting real users? Concretely, we envision potential questions in the SAMS context and services that retrieve and display answers based on the multi-dimensional markup of SAMSDocs.

Let us first take a **programmer**'s perspective. Her main information source for the programming task will stem from the module specification. But while studying it the following questions might arise:

  (i) What is the definition for a certain (mathematical) symbol?[3]
 (ii) How much of this specification has already been implemented?
(iii) In what state is the proof of a specific equation, has it already been formally verified so that it is safe to ground my implementation on it?
(iv) Whom can I ask for further details?

Assuming multi-dimensional markup an information retrieval system can supply useful responses. For example, it can answer (i) if technical terms in natural language are linked to the respective formal mathematical symbols they represent. For replies to (ii) and (iii) we note that, if all collection links are merged into a graph, their original placement and direction no longer makes a

---

[3] See fig. 4 and 6 for two symbols having the same appearance but different meanings.

difference. So if we have links from the Isabelle formalization to the respective C code and links from this C code to a specification fragment, as realized in the V-Model structure of SAMSDocs, we can follow the graph from the specification through to the state of the according proof. Drawing on the V-Model links combined with the semantic version management or the review logs, the system can deduce the answer for (iv): The code in question connects to a specification document that has authors and reviewers. This service can be as fine-grained as one is willing to formalize the granularity of the version and review management. If we admit further dimensions of markup into the picture, then the system might even find persons with similar interests (e. g. expressed in terms of the FOAF vocabulary), as has been investigated in detail for expert finder systems [SWJL10].

Now, we take a more global perspective, the one of a **project manager**. She might be concerned with the following issues:

(v) *Software Engineering Process*: How much code has been implemented to satisfy a particular requirement from the contract? Has the formal code structure passed a certain static analysis and verification? She does not want to inspect that manually by running Isabelle, thus, she needs high-level figures of, e. g., the number of mathematical statements without a formally verified proof.

(vi) *Certification*: What parts of the specification, e. g. requirements, have changed since the last certification? What other parts does that affect, and thus, what subset of the whole specification has to be re-certified?

(vii) *Human Capital*: Who is in charge of a document? How could an author be replaced if necessary, taking into account colleagues working on the same or on related documents – such as previous revisions of the same document, or its predecessor in terms of the V-Model, i. e. the document that is refined by the current one?

Exploiting the multi-dimensionality of formalized knowledge, it becomes obvious how the issues can be tackled.

Finally, we envision a **certifier**'s information needs. For inspection, she might first be interested in getting an overview, such as a list of all relevant concepts in the contract document. Then, she likes to follow the links to the detailed specification and further on to the actual implementation. For more information, she likes to contact the project investigator instead of the particular author of a code snippet. The certifier also needs to understand what parts of the whole specification are subject to a requested re-certification. Her rejection of a certain part of a document also affects all elements in the collection that depend on it. Again, a system can easily support a certifier's efficiency by combining the formalized information of distinct formality dimensions.

These use scenarios in a Software Engineering project clearly show that multi-dimensional markup is useful, since multi-dimensional queries serve natural information needs. To answer such queries, we need to represent multi-dimensional information in MKM formats.

4 Multi-Dimensional Markup

Structured representations are usually realized as files marked up in formats that reflect the primary formalization intent and markup preferences of the formalizer. In the evaluation of document formats it is thus important to realize that every representation language concentrates on only a subset of possible relationships, which it treats with specific language constructs. Note that therefore the formality space of a semantically enhanced document is very often reduced to this primary dimension. On the formal side, for example, a plethora of system-specific logics exist. Furthermore, formal systems increasingly contain custom modularization infrastructures, ranging from simple facilities for inputting external files to elaborate multi-logic theory graphs [MML07]. Collections of informal documents, on the other side, are often structured by application-specific metadata like the Math Subject Classification [Soc09] or the V-Model relations as in our case study.

No given format can natively capture *all* aspects of the domain via special-purpose markup primitives. It has to relegate some of them to other mechanisms like the sTEX-SD extension for the formalization of SAMSDocs, if more dimensions of the formality space than the primary one are to be covered. In representation formats that support fragment identifiers — e. g. XML-based ones — these relationships can be expressed as stand-off markup in RDF (Resource Description Framework [RDF04]), i. e., as subject-predicate-object triples, where subject and object are URI references to a fragment and the predicate is a reference to a relationship specified in an external vocabulary or ontology[4]. As we have XML-based formats for informal documents (e. g. XHTML+MathML+SVG) and formal specifications (OpenMath or Content MathML), we can in principle already encode multi-dimensional structured representations, if we only supply according metadata vocabularies for their structural relationships. Indeed this is the basic architecture of the "Semantic Web approach" to eScience, and much of the work of MKM can be seen as attempts to come up with good metadata vocabularies for the mathematical/scientific domain.

Since RDF stand-off markup is notoriously difficult to keep up to date, **RDFa** [ABMP08] has been developed: A set of attributes for embedding RDF annotations into XML-based languages, originally XHTML. On the one hand, RDFa serves as an enabling technology for making XML-based languages extensible by inter- and intra-document relationships. On the other hand, RDFa serves as a vehicle for document format interoperability. All relationships from a format F that cannot be natively represented in a format F' can be represented as RDFa triples, where the predicate is from an appropriately designed metadata vocabulary that

---

[4] The difference between "vocabulary" and "ontology" is not sharply defined. Vocabularies are often developed in a bottom-up community effort and tend to have a low degree of formality, whereas ontologies are often designed by a central group of experts and have a higher degree of formality. Here, we use "vocabulary" in its general sense of a set of terms from a particular domain of interest. This subsumes the term "ontology", which we will reserve for cases that require a more formal domain model.

describes the format F. For instance, an OMDoc `<theory>` element can be represented as `<div typeof="http://omdoc.org/ontology#Theory">` in XHTML, using the OMDoc ontology [Lan10]. Support of RDFa relationships make all XML-based formats theoretically equivalent, if they allow fine-grained text structuring with elements like XHTML's `div` or `span` everywhere (so that arbitrary text fragments can be turned into objects). In particular, they become **formats for multi-dimensional markup** as respective other dimensions can always be added via RDFa. We have detailed the necessary extensions for the OMDoc format in [Koh10], so that analogous extensions for any of the XML-based formats used in the MKM community should be rather simple to create.

Note that the pragmatic restriction to XML-based representation formats is not a loss of generality. In the MKM sphere the three classes of non-XML languages used are computational logics, TeX/LaTeX, and PostScript/PDF. We see computational logics as compact front-end formats that are optimized for manual input of formal structured representations; it is our experience that these can be transformed into the XML-based OpenMath, MathML, or OMDoc without loss of information (but with a severe loss of notational conciseness). We consider TeX/LaTeX as analogous for informal structured representations; they can be transformed to XHTML+MathML by the LaTeXML system. The last category of formats are presentation/print-oriented output and archival formats where the situation is more problematic: PostScript (PS) is largely superseded by PDF which allows standard document-level RDF annotations via XMP and the finer-granular annotations we need for structured representations via extensions as in [GMH+07] or [Eri07]. But PS/PDF are usually generated from other formats (mostly office formats or LaTeX), so that alternative generation into XML-based formats like XHTML or OMDoc can be used.

Note as well that a dimension typically corresponds to a vocabulary. In the course of the SAMSDocs case study, most vocabularies have initially been implemented from scratch in a project-specific ad hoc way. But they can be elaborated towards ontologies via sTeX and these can be translated to RDF-based formats that automated reasoners understand [KKL10]. An alternative is reusing existing ontologies. This has the advantage that they are more widely used and thus, reusable services may already have been implemented for them. For instance, there already exists a vocabulary that defines basic properties of persons and organizations: FOAF (Friend of a Friend [BM07]). The widely known Dublin Core element set is also available as an ontology [Dub08]. DCMI Terms [DCM08], a modernized and extended version of the Dublin Core element set, offers a basic vocabulary for revision histories – but not for reviewing and certification. DOAP (Description of a Project [Dub10]) describes software projects, albeit focusing on the top-level structure of public open source projects. LIN et al. have developed an ontology for the requirements-related parts of the V-Model (cf. [LFB96]). HAPPEL and SEEDORF briefly review further ontologies about Software Engineering [HS06]. As, e.g. the SAMSDocs vocabularies can be integrated with existing ontologies by declaring appropriate subclass or equivalence relationships, services can make use of the best of both worlds.

5 Multi-Dimensional Services with MKM Technology

We will now study an avenue towards a concrete implementation of services based on the use cases described in sect. 3 to show how MKM technologies can cope with multi-dimensional information needs demonstrating their feasibility. Concretely, we will study the task of project manager Nora to find a substitute for employee Alice. All required information is contained in the sTeX-SD documents. To abstract from the particulars of sTeX/OMDoc RDFa encoding — e. g. the somewhat arbitrarily chosen direction of the relations or the interaction of metadata relations with the document and the special markup for the mathematical dimension — we extract a uniform RDF representation of the embedded structures, which can then be queried in the SPARQL language [PS08]. Listing 1.1 shows the necessary query in all detail.

Listing 1.1. Finding a Substitute for an Employee via the V-Model

```
# declaration of vocabulary (= dimension) namespace URIs
PREFIX vm:    <http://www.sams-projekt.de/ontologies/VersionManagement#>
PREFIX omdoc: <http://omdoc.org/ontology#>                              # OMDoc
PREFIX semVM: <http://www.sams-projekt.de/ontologies/V-model#>
5PREFIX dc:    <http://purl.org/dc/elements/1.1/>                        # Dublin Core
PREFIX xsd:   <http://www.w3.org/2001/XMLSchema#>               # XML Schema datatypes

SELECT ?potentialSubstituteName WHERE {
   # for each document Alice is responsible for, get all of its parts
10 # i.e. any kind of semantic (sub)object in the document
   ?document vm:responsible <.../employees#Alice> ;
             omdoc:hasPart  ?object .

   # find other objects that are related to each ?object
15 # 1. in that ?object refines them via the V-model
   { ?object semVM:refines ?relatedObject }
   UNION
   # 2. or in that they are other mathematical symbols defined in terms
   #      of ?object (only applies if ?object itself is a symbol)
20 { ?object omdoc:occursInDefinitionOf ?relatedObject }

   # find the document that contains the related object and the person
   # responsible for that document ...
   ?otherDocument omdoc:hasPart  ?relatedObject ;
25                 dc:date        ?date ;
                   vm:responsible ?potentialSubstitute .
   # (only considering documents that are sufficiently up to date)
   FILTER (?otherDocument > "2009-01-01"^^xsd:date)

30 # ... and the real name of that person
   ?potentialSubstitute foaf:name ?potentialSubstituteName .
}
```

In this query we assume that Alice's FOAF profile is a part of our collection, having the URI `.../employees#Alice`. Nora retrieves all documents in the collection for which Alice is known to be the responsible person. For any object O in each of these documents (e. g. the detailed specification of the braking distance function for straight-ahead driving s_G from fig. 4), she selects those objects that are refined by O in terms of the V-Model (e. g. the general braking distance s). Additionally, she considers the mathematical dimension and selects all objects that are related to O by mathematical definition, e. g. the braking function that uses s_G. Of any such related object, Nora finds out to what document it belongs.

10

She is only interested in recent documents and therefore filters them by date. Finally, she determines the responsible persons via the version management links, and gets their names from their FOAF descriptions. The assumption behind this query is that, if, for example, Pierre is responsible for the specification that introduces the general braking distance s, which Alice has refined, Pierre can be considered as a substitute for Alice. Note that getting the answer draws on the collection structures of SAMSDocs (V-Model), on the mathematical structures, as well as on the organizational structures (version management). It is easy to imagine how additional formality dimensions can be employed for increasing precision or recall of the query, or for ranking results. Consider, for example, another filter that only accepts as substitutes employees who have never got a document rejected in any previous certification.

The complexity of the query in listing 1.1 is directly caused by the complexity of the underlying multi-dimensional structures and the non-triviality of answering high-level project management queries from the detailed information in SAMSDocs. As users like Nora would not want to deal with a machine-oriented query language, we have developed a system that integrates versioned storage of semantic document collections with human-oriented presentation with embedded interactive services [DKL⁺10]. Thus, the rendered documents serve as command centers for executing queries and displaying results[5]. They provide access to queries in two ways: Queries with a fixed structure that have to be answered recurringly will be made available right in the (rendered) documents in appropriate places. This is the case with our employeee substitution query: This month, Alice may be ill, whereas next month, Bob may be on holiday. Access to this query can be given wherever an employee or a reference to an employee occurs in a document. Alternatively, non-prefabricated queries can be composed more intuitively on demand using a visual input form.

These examples show that multi-dimensional queries like the ones naturally coming up in Software Engineering scenarios (sect. 3) can be answered with existing MKM technology. Moreover, it illustrates that multi-dimensional markup affords multi-dimensional services. If we interpret our dimensions as distinct contexts, our services become context-sensitive, as dimensions can be filtered in and out. For instance, the context menu of certification documents could be equipped with menu entries for committing an approval or rejection to the server, which would only be displayed to the certifier. The server could then trigger further actions, such as marking the document that contains a rejected object and all dependencies of that object as rejected, too. In general, the more dimensions are formalized in a document, the more context-sensitive services become available.

---

[5] In particular, the rendered XHTML+MathML also preserves the original semantic structure as parallel MathML markup and RDFa annotations, so that a suitable browser plugin can dynamically generate interaction points for semantic services; see [KKL10] for details.

6 Conclusion and Further Work

In this paper we have studied the applicability of MKM technologies in Software Engineering beyond "Formal Methods" (based on the concrete SAMSDocs document collection and its formalization). The initial hypothesis here is that contract documents, design specifications, user manuals, and integration reports can be partially formalized and integrated into a computer-supported software development process. To test this hypothesis we have studied a collection of documents created for the development of a safety zone computation, the formal verification that the braking trajectory always lies in the safety zone, and the SIL3 certification of this fact by a public certification agency. As the project documents contain a wealth of (informal) mathematical content, MKM formats (in this case our OMDoc format) are well-suited for this task. During the formalization of the LaTeX part of the collection, we realized that the documents contain an open-ended, multi-dimensional space of formality that can be used for supporting projects — if made explicit.

We have shown that RDFa-based extensions of MKM formats, employing flexible "metadata" relationships referencing specific vocabularies, can be used to encode this formality space and put it into service. We have pointed out that the "dimensions" of this space can be seen to correspond to different metadata vocabularies. Note that the distinction between data and metadata blurs here as, for example, the OMDoc data model realized by native markup in the OMDoc format can also be seen as OMDoc metadata and could equally be realized by RDFa annotations to some text markup format, where the meaning of the annotations is given by the OMDoc ontology. This "metadata view" is applicable to all MKM formats that mark up informal mathematical texts (e. g. MathDox [CCB06] and MathLang [KWZ08]) as long as they formalize their data model in an ontology. This observation makes decisions about which parts of the formality space to support with native markup a purely pragmatic choice and opens up new possibilities in the design of representation formats. It seems plausible that all MKM formats use native markup for mathematical knowledge structures (we think of them as primary formality structures for MKM) and differ mostly in the secondary ones they internalize. XHTML+MathML+RDFa might even serve as a baseline interchange format for MKM applications[6], since it is minimally committed. Note that if the metadata ontologies are represented in modular formats that admit theory morphisms, then these can be used as crosswalks between secondary metadata for higher levels of interoperability. We leave its development to future work.

The formalized secondary formality structures can be used for enriching interactive document browsing and for enabling multi-dimensional metadata queries over documents and collections. We have shown a set of exemplary multi-dimensional services based on the RDFa-encoded metadata, mostly centered around Linked Data approaches based on RDF-based queries. More services can

---

[6] Indeed, a similar proposal has been made for Semantic Wikis [VO06], which have related concerns but do not primarily involve mathematics.

be obtained by exporting Linked Data to the Semantic Web or a company intranet and thus enabling further reuse. In particular, the multi-dimensionality observed in this paper and its realization via flexible metadata regimes in representation formats allows the knowledge engineers to tailor the level of formality to the intended applications.

In our case study, the metadata vocabularies ranged from project-specific ones that had to be developed (e. g. definition tables) to general ones like the V-Model vocabulary, for which external ontologies could be reused later on. We expect that such a range is generally the case for Software Engineering projects, and that the project-specific vocabularies may stabilize and be standardized in communities and companies, lowering the formalization effort entailed by each individual project. In fact we anticipate that such metadata vocabularies and the software development support services will become part of the strategic knowledge of technical organizations.

In [CF09, 241] CARETTE and FARMER challenge MKM researchers by assessing some of their technologies: "*A lack of requirements analysis very often leads to interesting solutions to problems which did not need solving*". With this paper we hope to have shown that MKM technologies can be extended to cope with "real world concerns" (in Software Engineering). Indeed, industry is becoming more and more aware of and interested in Linked Data (see e. g. [Ser08] and [LDF, Question 14]), which boosts relevance to the multi-dimensional knowledge management methods presented in this paper.

References

[ABMP08] Ben Adida, Mark Birbeck, Shane McCarron, and Steven Pemberton. RDFa in XHTML: Syntax and processing. W3C Recommendation, World Wide Web Consortium (W3C), October 2008.

[BM07]   Dan Brickley and Libby Miller. FOAF vocabulary specification 0.91. Technical report, ILRT Bristol, November 2007.

[CCB06]   A. M. Cohen, H. Cuypers, and E. Reinaldo Barreiro. Mathdox: Mathematical documents on the web. In OMDoc – *An open markup format for mathematical documents [Version 1.2]* [Koh06], chapter 26.7, pages 278–282.

[CF09]   Jacques Carette and William Farmer. A review of mathematical knowledge management. In Jacques Carette, Lucas Dixon, Claudio Sacerdoti Coen, and Stephen M. Watt, editors, *MKM/Calculemus 2009 Proceedings*, number 5625 in LNAI, pages 233–246. Springer Verlag, July 2009.

[DCM08] DCMI Usage Board. DCMI metadata terms. DCMI recommendation, Dublin Core Metadata Initiative, 2008.

[DKL⁺10] Catalin David, Michael Kohlhase, Christoph Lange, Florian Rabe, Nikita Zhiltsov, and Vyacheslav Zholudev. Publishing math lecture notes as linked data. In Lora Aroyo, Grigoris Antoniou, and Eero Hyvönen, editors, *ESWC*, number 6089 in Lecture Notes in Computer Science, pages 370–375. Springer, June 2010.

[Dub08]   Dublin Core metadata element set. DCMI recommendation, Dublin Core Metadata Initiative, 2008.

[Dub10]   Edd Dubmill. DOAP – description of a project. http://trac.usefulinc.com/doap, seen Mar. 2010.

[Eri07]   Henrik Eriksson. The semantic-document approach to combining documents and ontologies. *International Journal of Human-Computer Studies*, 65(7):624–639, 2007.

[FHL⁺08] Udo Frese, Daniel Hausmann, Christoph Lüth, Holger Täubig, and Dennis Walter. The importance of being formal. In Hardi Hungar, editor, *International Workshop on the Certification of Safety-Critical Software Controlled Systems SafeCert'08*, volume 238 of *Electronic Notes in Theoretical Computer Science*, pages 57–70, September 2008.

[For08]   FormalSafe. `http://www.dfki.de/sks/formalsafe/`, 2008. seen March 2010.

[GMH⁺07] Tudor Groza, Knud Möller, Siegfried Handschuh, Diana Trif, and Stefan Decker. SALT: Weaving the claim web. In Karl Aberer, Key-Sun Choi, Natasha Fridman Noy, Dean Allemang, Kyung-Il Lee, Lyndon J. B. Nixon, Jennifer Golbeck, Peter Mika, Diana Maynard, Riichiro Mizoguchi, Guus Schreiber, and Philippe Cudré-Mauroux, editors, *ISWC/ASWC*, number 4825 in Lecture Notes in Computer Science, pages 197–210. Springer, 2007.

[HS06]   Hans-Jörg Happel and Stefan Seedorf. Applications of ontologies in software engineering. In *Proc. 2ⁿᵈ International Workshop on Semantic Web Enabled Software Engineering (SWESE '06)*, 2006.

[KKL10]   Andrea Kohlhase, Michael Kohlhase, and Christoph Lange. sTeX – a system for flexible formalization of linked data. submitted to I-SEMANTICS 2010, 2010.

[Koh06]   Michael Kohlhase. OMDoc – *An open markup format for mathematical documents [Version 1.2]*. Number 4180 in LNAI. Springer Verlag, August 2006.

[Koh08]   Michael Kohlhase. Using LaTeX as a semantic markup format. *Mathematics in Computer Science*, 2(2):279–304, 2008.

[Koh10]   Michael Kohlhase. An open markup format for mathematical documents OMDoc [version 1.3]. Draft Specification, 2010.

[KWZ08]   Fairouz Kamareddine, J. B. Wells, and Christoph Zengler. Computerising mathematical text with mathlang. *Electron. Notes Theor. Comput. Sci.*, 205:5–30, 2008.

[Lan10]   Christoph Lange. The OMDoc document ontology. web page at `http://kwarc.info/projects/docOnto/omdoc.html`, 2010. seen 3/2010.

[LDF]   Linked data FAQ. `http://structureddynamics.com/linked_data.html`.

[LFB96]   Jinxin Lin, Mark S. Fox, and Taner Bilgic. A requirement ontology for engineering design. In *Proceedings of 3ʳᵈ International Conference on Concurrent Engineering*, pages 343–351. Technomic Publishing Company, Inc., August 1996.

[MML07]   Till Mossakowski, Christian Maeder, and Klaus Lüttich. The heterogeneous tool set. In Orna Grumberg and Michael Huth, editors, *Proceedings of the 13ᵗʰ International Conference on Tools and Algorithms for the Construction and Analysis of Systems TACAS-2007*, number 4424 in LNCS, pages 519–522, Berlin, Germany, 2007. Springer Verlag.

[NPW02]   Tobias Nipkow, Lawrence C. Paulson, and Markus Wenzel. *Isabelle/HOL — A Proof Assistant for Higher-Order Logic*. Number 2283 in LNCS. Springer, 2002.

[PS08]   Eric Prud'hommeaux and Andy Seaborne. SPARQL query language for RDF. W3C Recommendation, World Wide Web Consortium (W3C), January 2008.

14

[RDF04]   Resource description framework (RDF). `http://www.w3.org/RDF/`, 2004.

[SAM09]   SAMS. SAMSDocs: The document collection of the SAMS project, 2009. `http://www.sams-projekt.de`.

[Ser08]   François-Paul Servant. Linking enterprise data. In Christian Bizer, Tom Heath, Kingsley Idehen, and Tim Berners-Lee, editors, *Linked Data on the Web (LDOW 2008)*, number 369 in CEUR Workshop Proceedings, April 2008.

[SIM99]   Frank M. Shipman III and Raymond J. McCall. Incremental formalization with the hyper-object substrate. *ACM Trans. Inf. Syst.*, 17(2):199–227, 1999.

[SKG⁺10] Heinrich Stamerjohanns, Michael Kohlhase, Deyan Ginev, Catalin David, and Bruce Miller. Transforming large collections of scientific publications to XML. *Mathematics in Computer Science*, 2010. in press.

[Soc09]   American Mathematical Society. Mathematics Subject Classification MSC2010. `http://www.ams.org/mathscinet/msc/`, 2009.

[SWJL10] Milan Stankovic, Claudia Wagner, Jelena Jovanovic, and Philippe Laublet. Looking for experts? what can linked data do for you? In Christian Bizer, Tom Heath, Tim Berners-Lee, and Michael Hausenblas, editors, *Linked Data on the Web (LDOW 2010)*, CEUR Workshop Proceedings, April 2010.

[VO06]   Max Völkel and Eyal Oren. Towards a Wiki Interchange Format (WIF). In Max Völkel, Sebastian Schaffert, and Stefan Decker, editors, *Proceedings of the 1st Workshop on Semantic Wikis, European Semantic Web Conference 2006*, number 206 in CEUR Workshop Proceedings, Budva, Montenegro, June 2006.