Towards a Mizar Mathematical Library in OMDoc Format

Grzegorz Bancerek[i] and Michael Kohlhase[ii]

[i] Technical University Bialystok, bancerek@mizar.org
[ii] Jacobs University Bremen, m.kohlhase@iu-bremen.de

Abstract. Mizar is one of largest libraries of formalized mathematics. The language of the library is highly optimized for authoring by humans. Like in natural languages, the meaning of an expression is influenced by its (mathematical) context in a way that is natural to humans, but hard to specify for machine manipulation. From this point of view, it may be considered as locked up in an arcane file format. Indeed, the Mizar system itself is currently the only system that can reliably operate on the Mizar library.

This paper presents an experiment of using the Mizar system to transform the Mizar library into the OMDoc format (**O**pen **M**athematical **Doc**uments), an XML-based representation format for mathematical knowledge that is geared towards making formula structure and context dependencies explicit.

We expect the result of this experiment: an OMDoc version of the Mizar library to enhance system support for formal mathematical libraries.

1 Introduction

In the last years we have seen the birth of a new research area: "Mathematical Knowledge Management" (MKM), which is concerned with representation formalisms for mathematical knowledge, such as MathML [2], OpenMath [9] or OMDoc [16], mathematical content management systems [12, 1, 5], search engines [24, 23, 19, 15, 10], as well as publication and education systems [18, 20] for mathematics. The perceived interest in the domain of general knowledge management tools applied to mathematics is that mathematics is a very well-structured and well-conceptualized subject. The main focus of the MKM techniques is to recover the content/semantics of mathematical knowledge and exploit it for the application of automated knowledge management techniques, with an emphasis on web-based and distributed access to the knowledge.

Unfortunately, one of the largest resources of formalized mathematics, the Mizar library is locked up in an arcane file format that is highly optimized for authoring by humans, but not for MKM techniques other than verification of proofs.

ISBN 978-83-7431-128-1 ISSN 0860–150X

1.1 The MIZAR Mathematical Library

MIZAR is a representation format for mathematics that is close to mathematical vernacular used in publications and a deduction system for verifying proofs in the MIZAR language. The continual development of the MIZAR system has resulted a centrally maintained library of mathematics (the MIZAR mathematical library MML). The MML is a collection of MIZAR articles: text-files that contain theorems and definitions, and proofs. Currently the MML (version 4.76.959) contains 959 articles with 43149 theorems and 8185 definitions. Introductory information on MIZAR and the MML can be found in [22, 27, 28, 32]. For the rest of this paper, we assume that the reader is at least superficially familiar with these basic texts.

The MIZAR language is based on Tarski-Grothendieck set theory [29], it is essentially a first-order logic[1] with an extremely expressive type systems that features dependent types as well as predicate restrictions, [6]. This MIZAR language — in particular the type system and the input syntax — are highly optimized for authoring by humans. Consider for instance the following theorem:

```
1  for A being set holds
     A is finite iff ex f being Function st rng f = A & dom f in omega
```

For a skilled mathematician this can be almost read and understood without MIZAR-specific training. Like in natural languages, the meaning of an expression is influenced by its (mathematical) context in a way that is natural to humans, but hard to specify for machine manipulation. For example, MIZAR allows the following types:

```
reflexive transitive antisymmetric Relation
normal Subgroup of G
3  onto Element of MonoSeq(A)
```

As we see, MIZAR types consist of 2 parts: a cluster of adjectives (possibly empty) and a radix type. The cluster of adjective for the last type listed above consists of one adjective `onto` and `Element of MonoSeq(A)` is the radix type of it. In internal presentation, the adjective `onto` is recognized as an adjective with two arguments: the set `NAT` of natural numbers and the finite set `A` and the base type of the adjective is `Relation of NAT, A`. This reconstruction is done from the context. The set `MonoSeq(A)` is a subset of `Funcs(NAT, A)` and then the `Element of MonoSeq(A)` is also an element of `Funcs(NAT, A)`. Further, elements of `Funcs(NAT, A)` are `Function of NAT, A` which widens in sequence to `Relation of NAT, A`. The last type fits to the definition of the adjective `onto` and therefore the arguments of the last type are taken as the arguments of the adjective.

The MIZAR system itself is currently the only system that can reliably operate on the MIZAR library.

This paper presents a series of experiments of using the MIZAR system to transform the MIZAR library into the OMDOC format (**O**pen **M**athematical **Doc**uments [16, 26]), an XML-based representation format for mathematical knowledge that is geared towards making formula structure and context dependencies explicit.

[1] Technically, the Fraenkel Operator of MIZAR slightly transcends first-order expressivity, but the language is first-order in style.

We expect the result of this experiment: an OMDoc version of the Mizar library to enhance system support for formal mathematical libraries.

1.2 OMDoc in a Nutshell (three levels of modeling)

The OMDoc (Open Mathematical Documents) format is a content markup scheme for (collections of) mathematical documents including articles, textbooks, interactive books, and courses. To achieve content and context markup for mathematical knowledge, OMDoc uses three levels of modeling corresponding to the concerns raised previously. We have visualized this architecture in Figure 1.

Mathematical Formulae At the lowest level of mathematical formulae, OMDoc uses the established standards OpenMath [9] and Content-MathML [2]. These provide content markup for the structure of mathematical formulae and context markup in the form of URI references in the symbol representations

Mathematical Statements OMDoc provides an original markup scheme for making the structure of mathematical statements explicit. Again, we have content and context markup aspects. For instance the definition in the second row of Figure 1 contains an informal description of the definition as a first child and a formal description in the two recursive equations in the second and third children supported by the type attribute, which states that this is a recursive definition. The context markup in this example is simple: it states that this piece of markup pertains to a symbol declaration for the symbol plus in the current theory (presumably the theory arith1).

Mathematical Theories At this level, OMDoc supplies original markup for clustering sets of statements into theories, and for specifying relations between theories by morphisms. By using this scheme, mathematical knowledge can be structured into reusable chunks. Theories also serve as the primary notion of context in OMDoc, they are the natural target for the context aspect of formula and statement markup.

All levels are augmented by markup for various auxiliary information that is present in mathematical documents, e.g. notation declarations, exercises, experimental data, program code, etc.

2 Translating Mizar

The motivations for a translation of the Mizar library are not particular to the OMDoc format, and it is therefore not very surprising that the work reported on in this paper is not the first attempt to translate the Mizar library. In fact there have been many translation experiments:

The Mizar project produces a hyper-linked, pretty-printed version of the assertions of a Mizar article for publication in the journal *Formalized Mathematics* [7] with its electronic counterpart the *Journal of Formalized Mathematics* [14]. The translation generates a human-oriented presentation of Mizar articles, where formulae are presented in mathematical notation LaTeX and parts of the Mizar logical language are verbalized.

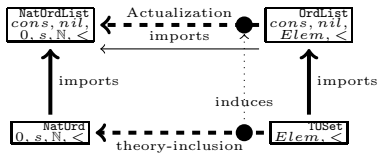
Level of Representation	OMDoc Example
Theory Level: Development Graph – Inheritance via symbol-mapping – Theory inclusion via proof-obligations – Local (one-step) local link vs. global links	
Statement Level: – Axiom, definition, theorem, proof, example,... – Structure explicit in statement forms and references	<definition for="#plus" type="recursive"> <CMP>Addition is defined by recursion on the second argument </CMP> <FMP>$X + 0 = 0$</FMP> <FMP>$X + s(Y) = s(X + Y)$</FMP> </definition>
Object Level: OpenMath/MathML – Objects as logical formulae – Semantics by pointing to theory level	<OMA> <OMS cd="arith1" name="plus"/> <OMV name="X"/> <OMS cd="nat" name="zero"/> </OMA>

Fig. 1. OMDoc in a Nutshell (the three levels of modeling)

There have been various hand translations of selected Mizar articles for benchmarking automated theorem provers (see e.g. [11]). Of course this approach does not really scale to the whole MML. In 1997/8 Czeslaw Bylinski and Ingo Dahn translated parts of the MML into a PROLOG syntax by extending the Mizar system with a PROLOG generator.

Josef Urban has built an extension of the Mizar parser (now part of the Mizar) that allows to export the internal, disambiguated form of Mizar articles used by the Mizar system to other formats. In the first stage, Urban uses this to generate a first-order version of the MML to test automated theorem provers and independently verify (parts of) the MML. In this translation, the Mizar type system was relativized to obtain standard first-order formulae; the Fraenkel operator and Mizar schemas were not treated, since they are second-order constructs. In current stage, Urban's extension reflects the internal form of Mizar article without relativization and, actually, is incorporated into the Mizar processor.

The first author uses the Mizar database and Urban's extension for information retrieval purposes. The MML Query system provides a web interface to and a query language the MML Query database; see [10, 8] for details. Since this query interface is primarily intended for Mizar authors and developers, great care has been taken to ensure that no relevant structure of the Mizar language has been lost. In particular, the Mizar types have not been relativized away. Currently, the MML Query database contains small amount of data about proofs, since this was considered secondary for information retrieval purposes.

2.1 General Issues in the Translation

When processing MIZAR text we have the choice between two levels of language available in the MIZAR (see [30, 10, 8]): The **pattern-level** language is the rich human-oriented input syntax in which MIZAR articles are written, and the **constructor-level**, which is the unique machine-internal representation used in the MIZAR proving engine.

The MIZAR language is a language designed for the practical formalization of mathematics. To enable it MIZAR allows synonyms for an overloading of symbols and patterns, MIZAR allows homonyms, and MIZAR allows also unambiguous tokenization dependent on lexical context (see *three hindrances in text based searching* in [8]). These features were used by different researchers to develop different parts of mathematics in MML. In the result, MML is full of notation conflicts on Pattern-level. To have adequate translation (mirroring the meaning of MIZAR text), we should omitted such conflicts as, in general, the syntactic similarity doesn't implies semantic correlations. Eventually, we decided to use the unambiguous internal representation (constructor-level) in MML QUERY format as a source for the translation.

2.2 The Translation of MML Data to OMDOC

Since version 7.2 the MIZAR system produces quite detailed XML-based semantic description of MIZAR articles at the constructor level [31]. We build on this format for our translation[2], using XSLT style sheets [33]. To understand the process, let us look at a simple example from the article `boole`:

Listing 1.1. A MML theorem

```
theorem
  for X being set holds X \/ {} = X
proof
  let X be set;
  thus X \/ {} c= X
  proof
    let x be set; assume x in X \/ {};
    then x in X or x in {} by XBOOLE_0:def 2;
    hence thesis by XBOOLE_0:def 1;
  end;
  let x be set;
  assume x in X;
  hence thesis by XBOOLE_0:def 2;
end;
```

Note that the identifiers in the MML text above are translated into pointers to internal array at the constructor level. For instance, `set` corresponds to the first constant in this array, witnessed by the attribute `nr="1"` in line four of Listing 1.2. Therefore, the first step is to add absolute MML addresses using Josef Urban's accommodation style sheets `addabsrefs.xsl`, the result of this is given in Listing 1.2

[2] The work reported here extends an earlier translation experiment reported at the "30 years of Mizar" workshop at MKM 2004 in Bialowieza, Poland 2004. That was based on the MML QUERY database and specified the translations as special MML QUERY templates, which filled in the necessary information via MML QUERY queries [10].

below. The style sheet has added the `aid` and `absnr` attributes in line four. This identifies the identifier `set` as the first constant in article `HIDDEN`.

Listing 1.2. The XML representation of Definition 1.1

```
1  <JustifiedTheorem plevel="" aid="BOOLE" kind="T" nr="1">
     <Proposition line="27" col="11" plevel="" propnr="1">
       <For pid="0" vid="3">
         <Typ kind="M" nr="1" pid="1" aid="HIDDEN" absnr="1"><Cluster/><Cluster/></Typ>
         <Pred kind="R" nr="1" pid="7" aid="HIDDEN" absnr="1">
6          <Func kind="K" nr="6" pid="5" aid="XBOOLE_0" absnr="2">
             <Var nr="1"/>
             <Func kind="K" nr="5" pid="4" aid="XBOOLE_0" absnr="1"/>
           </Func>
           <Var nr="1"/>
11       </Pred>
       </For>
     </Proposition>
     <Proof line="28" col="5" plevel="" newlevel="1">...</Proof>
   </JustifiedTheorem>
```

The absolute addresses are needed to translate MML constructors into OpenMath symbols which need a name and a content dictionary for referencing: We interpret Mizar articles as OMDoc content dictionaries, and thus the identifier `set` is translated to the symbol `<OMS cd="HIDDEN" name="c1"/>`; as the surface identifiers like `set` are overloaded, we simply generate symbol names from their numbers.

Note that Mizar language constructs like the Pred element on line five of Listing 1.2 are translated into OpenMath applications (via the OMA element) where the first child is the applied function and the rest are the arguments. Similarly, Mizar quantifiers are translated into OpenMath bindings (via OMBIND) elements with the appropriate binding symbol as the first child. In the case of the For element on line three of Listing 1.2, we use the universal quantifier from first-order logic (see Section 3 for a discussion). The bound variable needed as the second child of the is not represented in Mizar-XML, so we invent one, but the type can be translated from the second child of the For; it is represented in the OpenMath attribution lines 7-13 below.

Listing 1.3. The OMDoc representation of Theorem 1.1

```
<assertion xml:id="BOOLE.th1" type="theorem" just-by="BOOLE.pf1">
  <CMP>for X being set holds X \/ {} = X</CMP>
  <FMP logic="mizar">
    <OMOBJ xmlns="http://www.openmath.org/OpenMath">
5     <OMBIND><OMS cd="pl1" name="forall"/>
        <OMBVAR>
          <OMATTR>
            <OMATP>
              <OMS cd="simpletypes" name="type"/>
10            <OMS cd="HIDDEN" name="c1"/>
            </OMATP>
            <OMV name="X"/>
          </OMATTR>
        </OMBVAR>
15      <OMA><OMS cd="HIDDEN" name="c1"/>
          <OMA><OMS cd="XBOOLE_0" name="c2"/>
            <OMV name="X"/>
            <OMS cd="XBOOLE_0" name="c1"/>
          </OMA>
20        <OMV name="X"/>
        </OMA>
      </OMBIND>
```

```
    </OMOBJ>
  </FMP>
</assertion>
```

The discourse-level constructs in Mizar are translated to the appropriate statement-level ones in OMDoc, for instance the assertion element in line one above. The `just-by` attribute points to the translated proof, whose OMDoc representation is given in Listing 1.4; as we have already discussed the formula-level translation, we will just gloss them mathematical notation in the boxes.

Listing 1.4. The OMDoc representation of the proof in Listing 1.1

```
<proof xml:id="BOOLE.pf1" for="BOOLE.thm1">
  <symbol name="X"><type system="mizar">set</type></symbol>
  <derive>
    <FMP>X ∨ {} ⊆ X</FMP>
    <method xref="by">
      <proof>
        <symbol name="x"><type system="mizar">set</type></symbol>
        <hypothesis><CMP>x ∈ X ∨ {}</CMP></hypothesis>
        <derive>
          <CMP>x ∈ X ∨ x ∈ {}</CMP>
          <method xref="by"><premise xref="XBOOLE_0.omdoc#XBOOLE_0.def2" /></method>
        </derive>
        <derive type="conclusion">
          <method xref="by"><premise xref="XBOOLE_0.omdoc#XBOOLE_0.def1" /></method>
        </derive>
      </proof>
    </method>
  </derive>
  <symbol name="x"><type system="mizar">set</type></symbol>
  <hypothesis><CMP>x ∈ X</CMP></hypothesis>
  <derive type="conclusion">
    <method xref="by"><premise xref="XBOOLE_0.omdoc#XBOOLE_0.def2" /></method>
  </derive>
</proof>
```

Finally, `environment` declarations in Mizar articles can directly be mapped to `imports` elements in OMDoc, since they have the same function: providing vocabulary and theorems in the current environment (implicitly given by the Mizar article/OMDoc theory).

3 OMDoc Content Dictionaries for Mizar

As the OpenMath formula language underlying the OMDoc format is logic-independent, the "reserved words" of the Mizar language are expressed as OpenMath symbols and need to be documented in content dictionaries. We have developed two OMDoc content dictionaries: `mizar` and `mizar-types`, building on an already existing set of content dictionaries for logical systems presented in [17]; see Figure 2 for details.

The "Mizar-types" theory for the Mizar language defines the following symbols for constructing types, all others can be inherited from the content dictionaries already present.

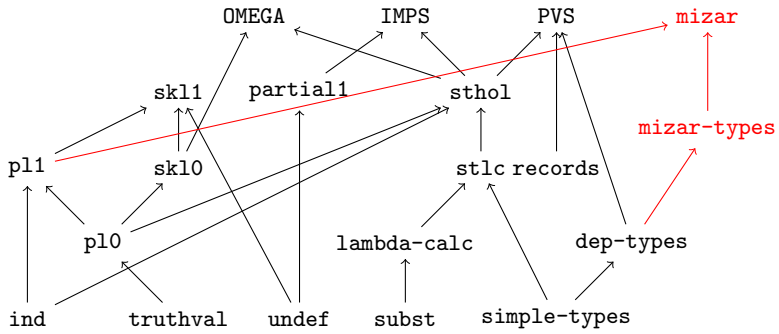


Fig. 2. Integrating the MIZAR language in to a Hierarchy of Logical Systems

type-expression The type expression constructor. The first argument is a radix type and the rest is a list of adjectives that modify it.

adjective-cluster The adjective composition operator. It makes an adjective cluster out of a sequence of simpler ones.

type-expression-nonempty The operator that expresses the fact that a type is legal, since it is non-empty.

adjective-inclusion The operator that expresses the fact that an adjective includes another on a radix type. The first argument is the radix type, the next two are adjective clusters, the first cluster entails the second one on the radix.

For the meanings of these please consult [6]. The term level of the MIZAR language similarly profits from existing content dictionaries, we only need to specify symbols that are not present in regular first-order logic. Thus we only need to provide symbols for dealing with the MIZAR type system and the weakly second-order construct of Fränkel sets. Concretely, we provide the symbols:

is The "is of type" operator in MIZAR. It expresses that its first argument has the type that is expressed as its second argument.

possesses The operator the expresses that a term possesses certain properties. Its first argument is the term and the rest are adjectives that express the properties.

adjective-holds The operator that expresses the fact that an adjective always holds on a radix type. The first argument is the radix type, the second one is an adjective cluster.

fraenkel-bind/set A Fränkel set is a set with typed bound variables in MIZAR, e.g. $\{x^n - 1 \mid \sqrt[n]{x} > 1 \text{where } x \in \mathbb{R} \text{ and } n \in \mathbb{N}\}$. Given the `mizar` content dictionary, this can be represented in OPENMATH as

```
1  <OMBIND><OMS cd="mizar" name="fraenkel−bind"/>
     <OMBVAR>x : ℝ n : ℕ</OMBVAR>
     <OMA><OMS cd="mizar" name="fraenkel−set"/>x^n − 1 ∛x > 1</OMA>
   </OMBIND>
```

where the `fraenkel-bind` acts as a binding operator binding the variables x and n and the fraenkel-set constructor combines the set expression schema (in these bound variables) and the restriction.

These two content dictionaries act as an explicit context representations for OpenMath representations of Mizar terms. As we have seen above, the OMDoc representation uses numbered constant names for representing constants. For presentation to the human user, the original Mizar identifiers can be regained via the OMDoc presentation language (see [16, chapter 25]). For instance, our translation generates the following `presentation` element in the `HIDDEN` content dictionary.

```
1 <presentation for="c1"><use format="default">set</format></presentation>
```

Thus presenting the generated OMDoc version of a Mizar article will lead to a similar user experience (e.g. cross-linked identifiers) as the presentation generated from Mizar XML via the Mizar presentation stylesheet in the Mizar distribution. Similar presentation elements for the Mizar language symbols allow to extend this presentation mechanism to them as well, linking the generated presentation to the relevant parts of the Mizar content dictionaries making them an integrated language documentation.

But this is not the only added value we obtain from the translation: other OMDoc-based tools become applicable as well, e.g. the MathWebSearch engine [21, 15], a highly efficient search engine for mathematical formulae in OpenMath format.

4 Conclusion

We have presented a structure-preserving, statement-level translation from the Mizar mathematical library to the OMDoc format. This translation currently produces valid OMDoc documents. Since the OpenMath formula language underlying the OMDoc format is logic-independent, the "reserved words" of the Mizar language are also expressed as OpenMath symbols, which are documented in the OpenMath content dictionary lstinline[basicstyle=`]mizar.omdoc`, which builds on and is integrated with the logic hierarchy presented at [17].

In the future we plan to experiment with OMDoc-based tools on the transformed Mizar library, such as the MBase mathematical knowledge base [12] or the Maya development graph manager [4], which offers semantics-informed process of change management. Furthermore, we plan to evaluate Immanuel Normann's theory morphism detector [25] to augment the semantic structure of the theory graph of the library, which is currently simply based on the simple inheritance structure of Mizar articles. We expect to find a theory structure that is based on the "semantic topology" of library content rather than on the historical coincidence that largely governs the structure of the library today.

Finally, we plan to develop an inverse translation from OMDoc to Mizar to allow round-tripping and checking for completeness of the transformation. If the Mizar translation of the OMDoc translation of the MML still passes the Mizar verifier, then we can be reasonably sure that no (essential) information has been lost in the translation processes.

References

1. Stuart Allen, Mark Bickford, Robert Constable, Richard Eaton, Christoph Kreitz, and Lori Lorigo. FDL: A prototype formal digital library – description and draft reference manual. Technical report, Computer Science, Cornell, 2002. `http://www.cs.cornell.edu/Info/Projects/NuPrl/html/FDLProject/02cucs-fdl.pdf`.
2. 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.
3. Andrea Asperti, Bruno Buchberber, and James Harold Davenport, editors. *Mathematical Knowledge Management, MKM'03*, number 2594 in LNCS. Springer Verlag, 2003.
4. Serge Autexier, Dieter Hutter, Till Mossakowski, and Axel Schairer. The development graph manager MAYA (system description). In Hélene Kirchner, editor, *Proceedings of 9th International Conference on Algebraic Methodology And Software Technology (AMAST'02)*. Springer Verlag, 2002.
5. Andrea Asperti, Luca Padovani, Claudio Sacerdoti Coen, and Irene Schena. HELM and the semantic math-web. In Richard. J. Boulton and Paul B. Jackson, editors, *Theorem Proving in Higher Order Logics: TPHOLs'01*, volume 2152 of *LNCS*, pages 59–74. Springer Verlag, 2001.
6. Grzegorz Bancerek. On the structure of Mizar types. *Electronic Notes in Theoretical Computer Science*, 85(7), 2003.
7. Grzegorz Bancerek. Automatic translation in Formalized Mathematics. *Mechanized Mathematics and Its Applications*, 5(2):19–31, 2006.
8. Grzegorz Bancerek. Information retrieval and rendering with MML Query. In Jon Borwein and William M. Farmer, editors, *Mathematical Knowledge Management, MKM'06*, number 4108 in LNAI, pages 266–279. Springer Verlag, 2006.
9. 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`.
10. Grzegorz Bancerek and Piotr Rudnicki. Information retrieval in MML. In Asperti et al. [3], pages 119–131.
11. Ingo Dahn and Christoph Wernhard. First order proof problems extracted from an article in the Mizar Mathematical Library. In Ulrich Furbach and Maria Paola Bonacina, editors, *Proceedings of the International Workshop on First order Theorem Proving*, number 97-50 in RISC-Linz Report Series, pages 58–62. Johannes Kepler Universität Linz, 1997.
12. Andreas Franke and Michael Kohlhase. System description: MBase, an open mathematical knowledge base. In David McAllester, editor, *Automated Deduction – CADE-17*, number 1831 in LNAI, pages 455–459. Springer Verlag, 2000.
13. Tetsuo Ida, Jacques Calmet, and Dongming Wang, editors. *Proceedings of Artificial Intelligence and Symbolic Computation, AISC'2006*, number 4120 in LNAI. Springer Verlag, 2006.
14. Journal of Formalized Mathematics. `http://www.mizar.org/JFM`.
15. Michael Kohlhase and Ioan Şucan. A search engine for mathematical formulae. In Ida et al. [13], pages 241–253.
16. Michael Kohlhase. OMDoc – *An open markup format for mathematical documents [Version 1.2]*. Number 4180 in LNAI. Springer Verlag, 2006.

17. Michael Kohlhase. Standardizing context in system interoperability. In OMDoc – *An open markup format for mathematical documents [Version 1.2]* [16].
18. Christoph Lange and Michael Kohlhase. A semantic wiki for mathematical knowledge management. In Max Völkel, Sebastian Schaffert, and Stefan Decker, editors, *Proceedings of the 1st Workshop on Semantic Wikis, European Semantic Web Conference 2006*, volume 206 of *CEUR Workshop Proceedings*, Budva, Montenegro, June 2006.
19. Paul Libbrecht and Erica Melis. Methods for Access and Retrieval of Mathematical Content in ActiveMath. In N. Takayama and A. Iglesias, editors, *Proceedings of ICMS-2006*, number 4151 in LNAI. Springer Verlag, 2006. `http://www.activemath.org/publications/Libbrecht-Melis-Access-and-Retrieval-ActiveMath-ICMS-2006.pdf`.
20. E. Melis, J. Buedenbender E. Andres, Adrian Frischauf, G. Goguadze, P. Libbrecht, M. Pollet, and C. Ullrich. The Activemath Learning Environment. *Artificial Intelligence and Education*, 12(4), 2001.
21. Math web search, web page at `http://kwarc.info/projects/mws/`, seen Jan 2007.
22. Mizar manuals. `http://mizar.org/project/bibliography.html`.
23. Rajesh Munavalli and Robert Miner. Mathfind: a math-aware search engine. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 735–735, New York, NY, USA, 2006. ACM Press.
24. Bruce R. Miller and Abdou Youssef. Technical aspects of the digital library of mathematical functions. *Annals of Mathematics and Artificial Intelligence*, 38(1-3):121–136, 2003.
25. Immanuel Normann. Enhanced theorem reuse by partial theory inclusions. In Ida et al. [13].
26. The OMDoc repository, web page at `http://www.mathweb.org/omdoc`.
27. Piotr Rudnicki, Christoph Schwarzweller, and Andrzej Trybulec. Commutative algebra in the Mizar system. *Journal of Symbolic Computation*, 32:143–169, 2001.
28. Andrzej Trybulec and Piotr Rudnicki. On equivalents of well-foundedness. *Journal of Automated Reasoning*, 23(3-4):197–234, 1999.
29. Andrzej Trybulec. Tarski Grothendieck set theory. *Formalized Mathematics*, 1(**1**):9–11, 1990.
30. Josef Urban. Translating Mizar for first-order theorem provers. In Asperti et al. [3], pages 203–215.
31. Josef Urban. XML-izing Mizar: making semantic processing and presentation of MML easy. In Michael Kohlhase, editor, *Mathematical Knowledge Management, MKM'05*, number 3863 in LNAI, pages 346 – 360. Springer Verlag, 2005.
32. Freek Wiedijk. Mizar: An impression, 1999. `http://www.cs.kun.nl/~freek/notes`.
33. Xsl transformations (xslt) version 1.0. W3c recommendation, W3C, 1999.