Wissensmodellierung und -nutzung in ActiveMath *

Die ActiveMath Gruppe:
Erica Melis, Georgi Goguadze, Paul Libbrecht, Carsten Ullrich
Universität des Saarlandes, D-66123 Saarbrücken

Zusammenfassung

Die Modellierung und web-fähige Repräsentationen von Lerninhalten gehört zu den Forschungsaufgaben für web-basierte intelligente Lehr- und Lernsysteme. Die Wissensrepräsentation der Lernumgebung ActiveMath orientiert sich an und trägt bei zu den Standards, die für die *semantische* Repräsentation von mathematischem Wissen entwickelt werden und denen, die Lerninhalte didaktisch charakterisieren. Solche Standardisierungen sind erforderlich, damit eine sinnvolle Wiederverwendung der Lerninhalte möglich ist. Über die Repräsentation der Lerninhalte hinaus modelliert und benutzt ActiveMath auch das didaktische Wissen, das beinhaltet, was und wie gelehrt werden soll und welches lokale oder globale Feedback wann gegeben werden soll. Da dies kein Projektbericht ist, liegt der Schwerpunkt des Artikels auf der Modellierung von Wissen und dessen Verwendung in ActiveMath für die benutzer- und kontextadaptive Auswahl und Präsentation von Lerninhalten. Eine Vielzahl von Ideen und Komponenten des Systems werden daher nicht beschrieben.

1 Motivation

ActiveMath ist eine web-basierte, benutzeradaptive Lernumgebung, die derzeit vor allem für Mathematik verwendet wird, jedoch auch für andere Gebiete geeignet ist[1]. Die Modellierung und Repräsentation des zugrundeliegenden Wissens in ActiveMath versucht, die im Folgenden diskutierte allgemeine Problematik für eLearning-Systeme zu lösen.

Die Entwicklung von Lerninhalten für gute Lernsysteme ist weit aufwendiger als die Arbeit für traditionelle Lehrbücher. Insbesondere brauchen benutzer-adaptive Systeme eine in kleinere Einheiten strukturierte Wissensrepräsentation, eine Modellierung von Vorbedingungen für eine Lerneinheit, Wissen über die Bewertung von Übungsaufgaben etc.

Da es sehr aufwendig ist, geeignete Inhalte für Lernsysteme zu produzieren, richtet sich das Augenmerk auf dessen *Wiederverwendbarkeit* in anderen Kontexten. Die Entwicklung von standardisierten Ontologien – Strukturen und Metadaten – für die didaktische Charakterisierung von Lerneinheiten und teilweise für die Inhaltsdomänen ist eine Grundlage, um geeignete Lernmaterialien automatisch zu suchen und hoffentlich passend zusammenstellen zu können. Eine systemunabhängige Wiederverwendung von interaktiven Übungsaufgaben, in denen Problemlösesysteme wie etwa Computeralgebrasysteme oder Statistiksoftware den Lernenden unterstützten oder

*Diese Arbeit wurde teilweise im Rahmen des BMBF-Programms *Neue Medien in der Bildung* finanziert.

[1] Mathematik hat den Vorteil, klar strukturiert zu sein und den Nachteil, zusätzliche Forschung und technischen Aufwand für die Präsentation formal-mathematischer Inhalte zu erfordern.

eine Lösung kontrollieren können, erfordert die *Maschinen-Verstehbarkeit* ihrer Repräsentation. Das heisst beispielsweise, verschiedene Computeralgebrasysteme sollen dieselbe Repräsentation einer Übungsaufgabe verstehen können.

Dies war für uns ein entscheidender Grund, das zunächst für Computeralgenbrasysteme und Beweissysteme entwickelte `OpenMath` und seine Erweiterung `OMDoc` [**?**] zu verwenden, mitzugestalten und um didaktische und andere Informationen zu erweitern.

Die Repräsentation des mathematischen Wissens sollte für Lernumgebungen und andere Systeme, die auf dieses Wissen zugreifen, übereinstimmen, da nur so der kohärente Aufbau und Zugriff auf kollektive Wissensbestände möglich wird. Deshalb entschieden wir uns zugunsten eines allgemeinen Markup Formats für Mathematikdokumente. Dies hat den Vorteil, dass verschiedenste Anwendungen auf dieselbe Repräsentation des Wissens zugreifen können. Dies wiederum hat zur Folge, dass (1) die Wissensrepräsentation von den Funktionalitäten des Systems getrennt werden muss und dass (2) die Wissensrepräsentation modularisiert werden muss, weil die verschiedenartigen Anwendungen zwar gleiches Grundwissen aber teilweise verschiedene Charakterisierung ihrer Elemente und Erweiterungen benötigen.

2 Wissensrepräsentation für Mathematik

Die Wiederverwendbarkeit und Maschinen-Verstehbarkeit von Inhalten im Web sind auch Ziele der weltweiten Forschung zum *semantic Web* [**?**]. Diesen Forschungen ging eine Entwicklung für das Dokumenten-Markup voraus, die von Präsentations-Markup (`HTML`) über strukturelles Markup (`XML`) zu ontologischem Markup (ontologisches `XML`) [**?**] führte. Während `HTML` eine rein syntaktische, präsentations-orientierte Repräsentation hat, lassen sich in `XML`-Sprachen Strukturelemente definieren und in ontologischem `XML` darüber hinaus auch Eigenschaften dieser Elemente und Relationen zwischen ihnen. Diese Sprachen werden jeweils durch eine DTD (document type definition), eine Grammatik, definiert. `XML`-Schemata können die Sprachen noch stärker einschränken.

Semantische `XML`-Sprachen repräsentieren neben Strukturelementen auch die Semantik von Objekten. In `OpenMath` beispielsweise wird die Semantik des mathematischen Objekts 'Additionsfunktion' in einem *content dictionary* festgehalten, auf das sich alle Systeme beziehen können, egal ob sie diese Funktion als '+' oder 'PLUS' präsentieren. Dadurch wird der Inhalt maschinen-verstehbar.

Was liefert ein semantisches Markup für Lernsysteme? Nicht nur können Maschinen die Semantik verstehen (und als ein Nebeneffekt beispielsweise *copy and paste* realisiert werden), die semantische Repräsentation hilft auch, zwei (oder mehr) Dokumente, in denen semantisch gleiche, aber jeweils syntaktisch verschieden präsentierte Objekte (wie + und PLUS) vorkommen, aufeinander zu beziehen und Teile dieser Dokumente gemeinsam konsistent zu präsentieren. Darüberhinaus können verschiedene Präsentationen aus derselben Repräsentation erzeugt werden.

Im Folgenden stellen wir zunächst Standards für Markup-Sprachen für Mathematik vor, an deren Entwicklung sich ActiveMath angeschlossen hat und die wir teilweise weiterentwickelt haben. Diese Orientierung an Standards erlaubt, die weltweit entstehenden Tools, wie Autorenwerkzeuge und Suchmaschinen, zu verwenden.

2.1 Semantik mathematischer Objekte

Für Mathematik gibt es grosse Wissensbestände, die bisher meist in der präsentationsorientierten Sprache LaTeX vorliegen. International entstanden mindestens zwei zunächst voneinander unabhängige Standards für semantische `XML`-Repräsentationen des formalen mathematischen Wissens, `MathML` und `OpenMath`.

MathML [?] definiert zwei Teilsprachen, Presentation-MathML und Content-MathML. Presentation-MathML definiert, wie eine feste Menge von mathematischen Symbolen, die in K-12 (Kindergarten bis Klasse 12) Verwendung finden, präsentiert werden[2]. Content-MathML definiert die Semantik für diese Symbole. Für eine Präsentation muss Content-MathML in Presentation-MathML transformiert werden.

Der OpenMath Standard [?] umfasst kein Präsentations-Markup, aber ermöglicht durch den Mechanismus der *content dictionaries* die Bedeutung (neuer) mathematischer Symbole zu spezifizieren und ist daher nicht auf K-12 beschränkt. In OpenMath werden mathematische Objekte (OMOBJ) aus Anwendungen (OMA), Symbolen (OMS) und Variable (OMV) definiert. Das Beispiel in Abbildung 1 zeigt, wie eine Formel aufgebaut wird aus Symbolen (wie etwa "in" und "eq") und aus Variablen (wie g und G). Das zweite Vorkommen der Formel $g^m = e$ wird repräsentiert durch eine Referenz zu einem früheren Vorkommen. Das Hauptziel von OpenMath war zunächst, einen Standard für Systeme zu schaffen, die mathematische Ausdrücke und Formeln als Eingabe benutzen, so dass diese Standardrepräsentation (mit Hilfe von sogenannten *phrasebooks*) in die Eingabesprachen der verschiedenen Systeme übersetzt werden kann und umgekehrt deren Output in OpenMath. Dafür repräsentiert OpenMath ausschliesslich mathematische Symbole und Formeln.

Content-MathML kann in in einen Teil von OpenMath übersetzt werden und umgekehrt und sie können wechselseitig in die jeweils andere Sprache eingebunden werden.

OpenMath liefert lediglich die Grundlage für eine Ontologie mathematischer Dokumente, denn zu solch einer Ontologie gehören neben den mathematischen Objekten auch Aussagen über die Objekte wie etwa Theoreme und Beweise sowie deren Eigenschaften und Relationen. Um Lernmaterialien, die adaptierbar sein sollen, repräsentieren zu können, bedarf es verschiedener Erweiterungen: strukturelle Elemente und Dublin Core [?] Metadaten wie sie in OMDoc [?] eingeführt werden (siehe §2.2) und diadaktische Elemente, Relationen und Eigenschaften, wie sie in der *education*-Erweiterung von OMDoc spezifiziert sind, siehe §3.

2.2 Ontologie mathematischer Dokumente

OMDoc ist eine Erweiterung von OpenMath und definiert Markup für Einheiten und deren Attribute in mathematischen Dokumenten. Dies ist in Abbildung 1 illustriert, in der die Repräsentation einer Einheit (Definition) gezeigt wird, die einen informellen Anteil (CMP für *commented mathematical property*) und einen formalen Anteil (FMP für *formal mathematical property*) enthält. Aus Platzgründen ist das OMOBJ-Element des FMP weggelassen.

Verschiedene Präsentationen des mathematischen Inhaltes des OMDoc-Elements aus Abbildung 1 sind in den Abbildungen 2 bis 4 zu sehen. Für die unterschiedlichen Präsentationen musste das zugrundeliegende OMDoc-Element selbst nicht geändert werden, es wurden lediglich verschiedene Stylesheets für die Transformation verwendet.

Struktur. Die OMDoc DTD spezifiziert folgende allgemeine Strukturelemente für Mathematik:

- Theorie
- Definition
- Axiom
- Assertion (die mit einem Typ weiter unterschieden werden als Theorem, Lemma, Korollar, Behauptung, Annahme)

[2]Die Browser Mozilla und Amaya unterstützen bereits Presentation-MathML, andere Browser bieten entsprechende Plugins.

```
<definition id="def_order" for="order" type="simple">
   <metadata>
      <title xml:lang="en">
        Definition of the order of a group element
      </title>
        <field use="mathematics"/>
        <abstractness level="neutral"/>
        <difficulty level="easy"/>
        <learning-context use="university_first_cycle"/>
        <relation type="depends_on">
          <ref theory="Th1" name="group"/>
          <ref theory="elementary" name="positive_integer"/>
        </depends-on>
   </metadata>
   <CMP xml:lang="en" verbosity="3"> If
      <OMOBJ><OMV name="G"/> </OMOBJ> is a
        <ref xref="Th1_def_group">group</ref> and
      <OMOBJ><OMA><OMS cd="set1" name="in"/>
            <OMV name="g"/> <OMV name="G"/>
         </OMA></OMOBJ>,
        then the order of <OMOBJ><OMV name="g"/></OMOBJ>
        is the smallest positive integer
        <OMOBJ><OMV name="m"/></OMOBJ> with
      <OMOBJ id="OMOBJ_o1">
            <OMA><OMS cd="relation1" name="eq"/>
            <OMA><OMS cd="Th1" name="power"/>
               <OMV name="g"/>
               <OMV name="m"/>
            </OMA>
            <OMS cd="Th1" name="unit"/>
         </OMA></OMOBJ>.
       If no positive integer
      <OMOBJ><OMV name="m"/></OMOBJ> with
      <OMOBJ xref="OMOBJ_o1"/> exists, we say that the
      order of <OMOBJ><OMV name="g"/></OMOBJ> is infinite.
   </CMP>
    <FMP>...
    </FMP>
</definition>
```

Abbildung 1: `OMDoc` Repräsentation einer Definition von *order of a group element*

- Beweis
- Methode
- Gruppe von Elementen `omgroup` (über Gruppierung in einer Theorie hinaus)
- Text (nicht näher spezifiziert)
- Algorithmus und Code

Metadaten in `OMDoc`. Metadaten und Attribute werden in `XML` verwendet, um Informationen *über* ein Element zu repräsentieren. Für das Kern-`OMDoc` umfassen die

Abbildung 2: PDF-Präsentation des Inhaltes aus Abbildung 1

Abbildung 3: `HTML`-Präsentation des Inhaltes aus Abbildung 1

Metadaten die des Dublin Core, Version 1.1 [3]. Sie sind in Abbildung 1 der Einfachheit halber grösstenteils weggelassen und beschreiben `contributor`, `creator`, `translator`, `subject`, `title`, `description`, `publisher`, `date`, `type`, `format`, `source`, `rights`, `language` und `identifier`.

Weitere Metadaten, die nicht nur für Lerndokumente relevant sind, sind die Informationen über `version` – da ein Repository von Wissen über die Zeit weiterentwickelt werden kann und `depends-on` – das allerdings inhaltlich nicht immer mit 'didaktische Voraussetzungen' übereinstimmt (welches deswegen als `prerequisite-of` repräsentiert wird).

3 Didaktisches Wissen in ActiveMath

In ActiveMath werden mehrere Arten didaktischen Wissens modelliert:

- didaktische Information über die Lerneinheiten (*instructional units*), siehe §3.1
- Information über eine Zusammenstellung von Lerneinheiten (*package*)
- Wissen darüber, was und wie präsentiert werden soll (*pedagogical rules*), siehe §3.2 und
- Modellierung von Lernszenarien, siehe §3.3.

3.1 Erweiterungen von `OMDoc`

Struktur. Nun können Lernmaterialien, neben Definitionen etc. zusätzlich motivierende oder einleitende Texte sowie illustrierende Beispiele, Gegenbeispiele, Abbildungen und Übungsaufgaben enthalten. Zum mathematischen Kern von `OMDoc` gehören diese Elemente nicht, weil sie konzeptuell zu einem Modul gehören, das für *Lern*materialen spezifiziert wird (ein Beweissystem oder ein Computeralgebrasystem benötigt diesen Teil der Wissensrepräsentation nicht). Die DTD der Lern-Erweiterung von `OMDoc` definiert die zusätzlichen Elemente

- Übungsaufgabe
- Beispiel
- Abbildung
- Applet
- verschiedene Textarten:
  - einleitender Text
  - motivierender Text
  - zusammenfassender Text

[3] http://purl.org/dc

Abbildung 4: Variante der HTML-Präsentation des Inhaltes aus Abbildung 1

Diese Elemente, wie auch die unten angegebenen didaktischen Metadaten, entsprechen weitestgehend einer Teilmenge der im LOM-Standard des IEEE Learning Technology Standards Committee (LTSC)[4] und Instructional Management Systems project (IMS)[5] festgelegten Elemente, wenn auch diese Standards nach unserer Erfahrung zu umfangreich für eine konkrete Anwendung sind und manche Metadaten (etwa *level of interactivity*) noch zu ungenau definiert sind, siehe [**?**]. Die Anlehnung an Standards garantiert, dass das System nicht in der Falle einer proprietären Wissensrepräsentation, die von anderen Systemen nicht verwendbar ist, stecken bleibt und dass Tools benutzt werden können, die international für diese Standard-Repräsentationen entwickelt werden.

Metadaten. Die ActiveMath-DTD definiert Metadaten, die insbesondere in Lernanwendungen relevant sind. Einige der Metadaten sind in dem Beispiel in Abbildung 1 enthalten, nämlich `field, abstractness, difficulty, learning-context`.

Die Metadaten, die beliebige Elemente annotieren können, sind

- `difficulty` mit den Werten `easy`, `fair` oder `hard`.
- `abstractness` mit den Werten `concrete`, `neutral` oder `abstract`.
- `learning-context`, welches u.a. erforderlich ist, um Metadaten wie `difficulty` zu relativieren. Dies kann die in IMS und IEEE definierten Werte annehmen: `Primary Education`, `Secondary Education`, `Higher Education`, `University First Cycle`, `University Second Cycle`, `University Postgrade`, `Technical School First Cycle`, `Technical School Second Cycle`, `Professional Formation`, `Continuous Formation`.
- `field` beschreibt das Gebiet des Lernobjektes, `mathematics`, `statistics`, `engineering`, `psycology`, `biology`, `chemistry`, `physics`, `computer_science`, `economy`, `other`.
- `verbosity` für verschieden ausführliche Beschreibungen der Lernmaterialien, etwa ausführliches Skript und weniger ausführliche Folien.
- `related_to` mit den Werten
  - `prerequisite_for`
  - `counterexample_for`
  - `definition_for`
  - `elaboration_for`
  - `example_for`
  - `motivation_for`
  - `proof_for`
  - `similar_to`

Folgende Metadaten charakterisieren derzeit Übungsaufgaben in ActiveMath:

- `pedagogical-level` mit Werten, die durch die Bloomsche Taxonomie von Lernzielen gerechtfertigt sind [**?**]:
  - `knowledge`

[4] URL: http://ltsc.ieee.org/index.html
[5] URL: http://www.imsproject.org/

  - `comprehension`
  - `application`
  - `transfer`

- das Attribut `type` das die geforderte Lernaktivität in einer Übungsaufgabe spezifiziert und zwar mit den möglichen Werten `check`, `calculate`, `give_example`, `prove`, `hypothesize`, `model` und `explore`.

- `system`, dessen Werte spezifizieren, mit welchem System eine interaktive Übungsaufgabe unterstützt wird bzw. ob es sich um eine *multiple choice* oder *fill in* Aufgabe handelt. Es kann derzeit die Werte `maple, mupad, gap, omega, mc, fillin` haben.

Diese durch die Anwendung im Lernen motivierten Metadaten dienen einerseits dem Austausch mit und der Wiederverwendbarkeit in anderen Lernsystemen und sind andererseits erforderlich für die benutzeradaptiven Funktionalitäten von ActiveMath selbst, siehe §4. Beispielsweise kann ActiveMath passend zum `field`-Wert für eine Informatikstudentin anderen Beispiele und Übungsaufgaben auswählen als für einen Psychologiestudenten.

3.2 Pädagogische Regeln

Die Aufgabe des CourseGenerators von ActiveMath ist die dynamische Generierung von Lerndokumenten abhängig sowohl von den Informationen im Studentenmodell als auch von den aktuellen Lernzielen und Szenariowünschen des Lernenden. Ein wichiger Schritt in dieser Generierung ist die Evaluation der sogenannten *pädagogische Regeln*. Diese Regeln kodieren Zusammenhänge zwischen den Einträgen im Benutzermodell, den Lernzielen und -szenario, dem wahrscheinlichen Wissen des Lernenden und dem zu präsentierenden Material (was) und Aufgaben sowie der Art und Weise der Präsentation (wie).

Für diese adaptive Kursgenerierung ist neben den Regeln natürlich die Annotation der Lernobjekte mit didaktischen Metadaten erforderlich. Beispielsweise kann ActiveMath durch Auswertung der Regeln Übungsaufgaben mit dem passenden Metadatum `type` auswählen und so entsprechend dem Lernziel eines Benutzers geeignete Typen von Übungsaufgaben anbieten. Zwei simple (und vereinfacht dargestellte) selbsterklärende Regeln sind

```
IF mastery(C,low) THEN addExcercises(C,difficulty(1,1,2,3))
IF user-field(F) AND field(E,F) THEN preferExample(C,E)
IF not(user_knows(Maple)) THEN discardExamples(ExternalSystem=Maple)
IF scenario(Rehearsal) THEN preferExamples(alreadySeen=true)
```

Eine andere Menge von Regeln modellieren didaktisches Wissen für einen benutzeradaptiven, globalen Vorschlagsmechanismus [**?**].

3.3 Szenarien

Unterschiedliche Lernende verfolgen unterschiedliche Absichten mit dem Lernstoff: die eine möchte sich schnell auf eine Prüfung vorbereiten, der andere sich einen Überblick über ein bestimmtes Thema verschaffen, der nächste gezielt ein Problem lösen. ActiveMath unterstützt diese unterschiedlichen Anforderungen momentan mit Hilfe von acht Szenarien. Durch das Szenario kann der Benutzer oder ein Lehrer das Lernziel des generierten Dokumentes festlegen.

Das Szenario bestimmt, welche Inhalte ausgewählt und in welcher Reihenfolge sie präsentiert werden. Für die Vorbereitung auf eine Prüfung beispielsweise, wird man nicht das gesamte Mathebuch von vorn bis hinten erneut durchlesen, sondern sich

auf die wichtigen Teile beschränken und noch einmal Übungsaufgaben durchgehen. In ActiveMath generiert daher das Szenario `Prüfungsvorbereitung` ein Buch, das nur Konzepte und Übungen für diese Konzepte enthält (die Anzahl der Übungen ist abhängig von Wissenstand des Lernenden, siehe auch die Regel im vorhergehendem Abschnitt). Die entsprechende (vereinfachte) Regel lautet:

```
IF scenario(ExamPrep) THEN present(Concepts,Exercises)
```

Im Szenario `Überblick` werden ebenfalls nur Konzepte und, wiederum abhängig vom Vorwissen des Benutzers, ein oder zwei Beispiele sowie erklärende Texte ausgewählt:

```
IF scenario(Overview) THEN present(Concepts,Examples,Elaborations)
```

Will der Benutzer lernen, wie er Konzepte anwenden kann, erstellt ActiveMath ein sehr ausführliches Buch:

```
IF scenario(GuidedTourApplication)
THEN present(Introduction,Motivation,Concepts,Methods,Examples,
             Exercises,Elaborations,OtherTexts)
```

Der Benutzer ist nicht auf die vom Szenario ausgewählten Inhalte beschränkt, sondern er kann natürlich jederzeit weitere Lerninhalte anfordern.

4 Wissensnutzung in ActiveMath

Wie in vielen KI-Systemen wird das Potenzial des enkodierten Wissens in ActiveMath erst dadurch voll ausgeschöpft, dass die Wissensrepräsentation von den darauf arbeitenden Funktionalitäten (wie Generierung und Präsentation) klar getrennt ist.

ActiveMath generiert Lernmaterial abhängig von den Lernzielen des Benutzers, dem gewählten Lernszenario, weiteren Präferenzen (wie etwa Sprache, Systembenutzung, Hauptlerngebiet (`field`)) und von seinem geschätzten Wissensstand. Ausserdem generiert das System Vorschläge für Navigation, für nächsten Lernstoff und für Übungsaufgaben und zwar abhängig vom Lernfortschritt und von den Aktivitäten des Lernenden [**?**].

Die benutzeradaptive Generierung wird durch die modulare Architektur realisiert, die in Abb. 5 wenig detailliert gezeigt ist – eben nur soweit sie für das Verständnis des Artikels erforderlich ist. Ihre Komponenten leben in verschiedenen (virtuellen) Maschinen. Im Einzelnen sind dies: der Webserver tomcat, der SessionManager (leitet die Anfragen des Benutzers an die zuständigen Komponenten weiter), der Präsentationsprozess einschliesslich der XSLT-Stylesheet-Filter [**?**], die Wissensbank MBase mit Inhalten, deren Repräsentation in §2.2 näher beschrieben wurde, der CourseGenerator, der das Expertensystem Jess[**?**] verwendet, die Sammlung pädagogischer Regeln, das Benutzermodell bestehend aus der Geschichte der Benutzeraktionen, dem Lernerprofil und den geschätzten Kenntnissen des Lernenden. Weiterhin sind Problemlösesysteme eingebunden, die einerseits den Lernenden durch Übernahme bestimmter Schritte unterstützen können und die andererseits die (In)Korrektheit der Lösungsschritte des Benutzers überprüfen [**?**] und dadurch ein interaktives und explorativ es Lernen unterstützen können.

Die Kursgenerierung geschieht folgendermaßen: Der Lernende gibt über eine Auswahlliste seine inhaltlichen Ziele (z.B. *order of a group element*) sowie sein Lernziel (das Szenario) an. Dann werden aus MBase die Zielkonzepte, mit Hilfe der Metadaten diejenigen Konzepte von denen die Zielkonzepte abhängen und weitere Lernmaterialien für die Konzepte (Beispiele, Übungen etc.) extrahiert. Aus dieser grossen Menge an Inhalten werden schließlich die pädagogisch sinnvollen Lernmaterialien unter Verwendung des Benutzermodells und der pädagogischen Regeln des Szenarios ausgewählt. Nach der Linearisierung des Kurses wird die aktuelle Seite nach `HTML` oder PDF transformiert und dem Benutzer angezeigt.

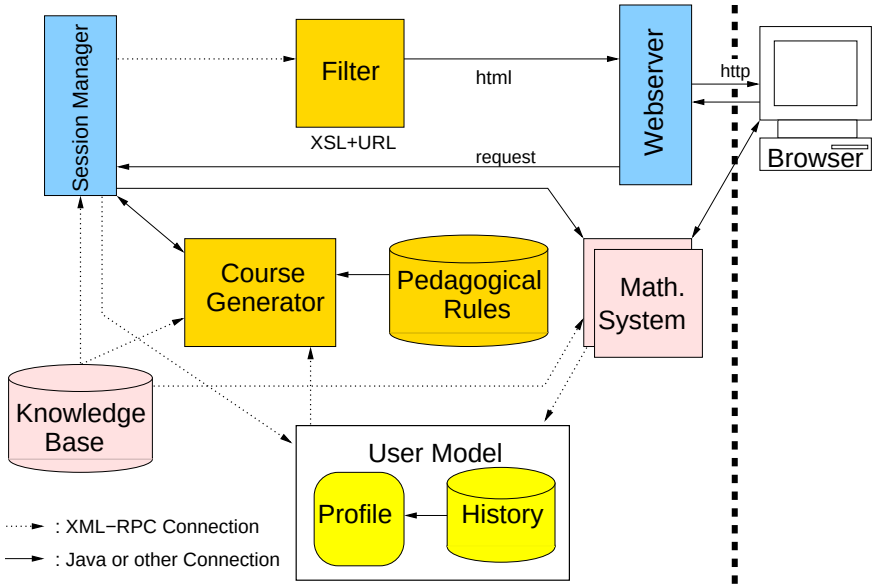

Abbildung 5: Die verteilte Architektur von ActiveMath

Präsentation des Inhaltes durch ActiveMath**.** Von der zugrundeliegenden Wissensrepräsentation sieht der Lernende nichts. Die Ausgabe im Browser als `HTML` und UniCode bzw. zum Drucken als PDF-Datei wird aus der `OMDoc`-Repräsentation in einem mehrstufigen Transformationsprozess erzeugt. Gegenwärtig wird eine Übersetzung von `OMDoc` (K-12) nach Presentation-`MathML` implementiert, so dass in naher Zukunft die Präsentation von Formeln nicht nur mit UniCode sondern auch mit Presentation-`MathML` möglich wird.

Wiederverwendbarkeit. Verschiedene Lernmodule (Inhalte) wiederzuverwenden kann bedeuten:

- Austausch ganzer *packages*, die zusammengesetzt sind aus mehreren Lernobjekten. Eine simple Standardisierung dafür liegt derzeit in IMS content package vor, die im Wesentlichen vergleichbar mit einem Inhaltsverzeichnis ist und demnächst durch EML-Metadaten (*Educational Modeling Language*) erweitert werden soll, die so ein *package* durch Lernprozessdaten charakterisieren [**?**].
- Auswahl und Verwendung einzelner Lernobjekte in verschiedenen Kontexten, wie beispielweise für verschiedene Benutzergruppen und schliesslich
- die konsistente Zusammenstellung und Verwendung von Lernobjekten, die ursprünglich aus verschiedenen Lernmaterialien stammen, möglicherweise auch von verschiedenen Autoren.

Die ersten zwei Arten der Wiederverwendung sind in ActiveMath bereits realisiert. Hierfür sind eine feinkörnige Strukturierung in Lernobjekte sowie deren Annotation mit Metadaten erforderlich. Die dritte Art der Wiederverwendung ist in ActiveMath bisher nur teilweise gelöst, nämlich durch automatisches Verlinken von Inhalten dadurch, dass mathematische Objekte, die zwar von verschiedenen Autoren verschieden bezeichnet wurden aber dieselbe mathematische Semantik haben (siehe 'PLUS' und '+'), in ActiveMath eine einheitliche Präsentation erfahren. Die rein verbalen Bestandteile von `OMDocs` können allerding noch nicht angepasst werden, so dass ein narrativ konsistentes Material entsteht, da sie nicht in abstrakter Form vorliegen.

ActiveMath unterstützt auch die manuelle Zusammenstellung von Lerninhalten. Das bedeutet, dass ein Lehrer ein Inhaltsverzeichnis als Grundlage für die Generierung des Lernmaterials angeben und damit seine eigenes Vorgehen weitgehend bestimmen kann. Erfahrungen zeigen nämlich, dass dies eine wichtige Grundlage für die Akzeptanz der Lernsysteme durch Lehrer ist [**?**].

5 Stand, Ausblick und verwandte Arbeiten

ActiveMath ist eine komplexe, stabil laufende Lernumgebung. Die nichtkommerzielle Verwendung des Systems ist kostenlos. Die frei zugängliche Version im Netz[6] umfasst derzeit Inhalte aus dem deutschsprachigen *Analysis Individuell*, das aus Copyrightgründen derzeit nicht vollständig im Netz verfügbar ist, sowie aus dem englischsprachigen Algebra-Kurs IDA [**?**]. Mehrere Kurse befinden sich gegenwärtig in der Entwicklung, unter anderem Grundlagen der Statistik, Introduction to Topology, Operations Research, Introduction to Computer Science und Sichere Software.

Da der Fokus dieses Artikels auf der Wissensmodellierung und deren Nutzung liegt, fehlen Beschreibungen mancher essentieller Komponenten, wissenschaftlicher Ideen und KI-Aspekte von ActiveMath fast völlig. Das betrifft zum Beispiel das Benutzermodell, Notizfunktionalitäten, die Einbindung von Servicesystemen (die man auch *cognitive tools* nennt), das Design von interaktiven Übungen und deren Feedback, das globale Feedback, das Autorenwerkzeug und vieles andere. Auch jegliche Berichte über erfolgte und geplante empirische Evaluierungen des Systems sind deshalb ausgespart worden (siehe aber [**?**]), was mitnichten bedeutet, dass wir solche Evaluierungen geringschätzen.

Wir verstehen diesen Artikel auch als ein Angebot zur Verwendung von ActiveMath zur Präsentation von Lehr- und Lerninhalten und zur Mitarbeit an den wiederverwendbaren und austauschbaren *open-source* Komponenten von ActiveMath.

Momentan werden einige Komponenten von ActiveMath erweitert: Zum einen wird die Präsentationskomponente, die die interne Wissensrepräsentation in die Ausgabeformate umwandelt, restrukturiert. Sie wird dabei in vier voneinander getrennte Stufen unterteilt, dem Laden des Wissens aus MBase, der Transformation nach `HTML` bzw. LaTeX, der Personalisierung, und schließlich dem Zusammenbau der Seite. Diese Unterteilung erlaubt effektives Cachen von Lernmaterialien auf den verschiedenen Stufen, wodurch die Zeit für die Transformation sinken wird.

Weiterhin wird der CourseGenerator zu einer tutoriellen Komponente erweitert werden, die einen dialog-ähnlichen, interaktiven Zugang zu den Lehrinhalten bietet und somit die Möglichkeiten, die die Wissensrepräsentation bietet, möglichst weitgehend ausnützt.

Verwandte Arbeiten. Innerhalb der *semantic Web* Forschungsrichtung sind ontologische Wissensrepräsentationen entstanden, jedoch unabhängig von Lernanwendungen. Zu ihnen zählt etwa die Sprache DAML+OIL[7].

Weitere deutsche Projekte verfolgen das Ziel, annotierte Repräsentationen für mathematische Lernobjekte zu verwenden, um verschieden zusammengestellte Dokumente zu generieren, nämlich e-stat [**?**] (web-basiertes Trainingsystem für Statistik) und slicing [**?**]. Slicing zerlegt Dokumente, vornehmlich LaTeX-Dokumente, in Einheiten und annotiert sie mit Schlagwörtern. e-stat verwendet eine `XML`-Sprache, um die Lernobjekte zu strukturieren. Diese Sprache stellt Formeln durch Presentation-`MathML` dar, enthält aber keine Semantik der mathematischen Objekte, was eine Wiederverwendung erschwert. Ähnlich dazu generiert das kanadische System Aphid [**?**] nichtmathematische Kurse.

[6] http://www.activemath.org/demo
[7] http://www.daml.org/2001/03/daml+oil-index