

# CONNEXIONS: MATHML AND COLLABORATIVE CURRICULUM DEVELOPMENT IN ENGINEERING

Version 2.5: 2003/02/06

Brent Hendricks  
Ross Reedstrom

*Produced by The Connexions Project*

This work is licensed under the Creative Commons Attribution License\*

## Abstract

The Connexions Project uses a collaborative, community-driven approach to content creation, organization, and dissemination. We provide a set of open source tools and an open repository for the publication and exchange of knowledge modules. These XML-based modules allow instructors to compose customized courses, providing students with new opportunities to explore the connections between different ideas and domains.

## Connexions: MathML and Collaborative Curriculum Development in Engineering

### 1 Introduction

The Connexions Project<sup>1</sup> is a community-driven collaborative knowledge creation and dissemination project at Rice University. Many different authors contribute small nuggets of information that we call **knowledge modules** to a repository. Each module covers a narrowly defined topic and is crosslinked to other modules, forming a sea of knowledge. Course instructors access this repository to construct a touring itinerary for their classes to follow, adding in their own content as well. Students use standard browsers to navigate through their instructor's course as well as the modules that crosslink to it, helping them visualize the relationships between concepts throughout the curriculum.

A team of faculty, staff, and students has been developing and using Connexions since 1999 to great success. Over six hundred modules now form the basis for six undergraduate courses at Rice and the University of Illinois, spanning the departments of electrical and computer engineering, computer science, and applied mathematics. Faculty members at other institutions worldwide (including University of Michigan, Ohio State University, Georgia Institute of Technology, Polytechnic University, Ecole Normale Supérieure in France, Ecole Polytechnique Fédéral de Lausanne in Switzerland, and Stavanger College in Norway) are forming author and instructor communities to develop thousands more modules.

\*<http://creativecommons.org/licenses/by/1.0>

<sup>1</sup><http://cnx.rice.edu/>

## 2 Connexions Philosophy

The two key ideas behind the Connexions approach are the modularization of the content, and the open licensing of both the software and the content. These provide a number of benefits for authors, instructors and students.

Modularity reduces the barrier to entry for authorship. Rather than committing three to five years to write a traditional textbook, authors may write individual, focused modules within their areas of interest, and publish them through the repository. This drops their time commitment from years down to perhaps only a few evenings. The open, public nature of the repository then allows for the rapid, iterative, collaborative improvement of their contribution, combining it with writings by other members of the Connexions community to generate larger bodies of work.

Once the repository has achieved a "critical mass" of modules in a particular topic area, instructors can assemble courses that follow their own particular didactic and pedagogical approach, adapting the text to their course, rather than the converse. The open licensing of the content allows instructors to combine existing modules with their ideas, rather than duplicate existing work. Note that the existence of the large body of modules allows the instructor to designate alternative approaches to key concepts in a straightforward way, as well as providing simple access to supplementary enrichment or prerequisite materials.

The repository includes modules and courses that intersect with the student's current course. Since these cross-course and cross-curriculum linkages are directly accessible to students, they can gain an increased understanding of the non-linear interconnectedness of various subjects. With traditional materials, students often do not gain this kind of integrated knowledge until late in their undergraduate career, or even later, in post-graduate studies or on the job.

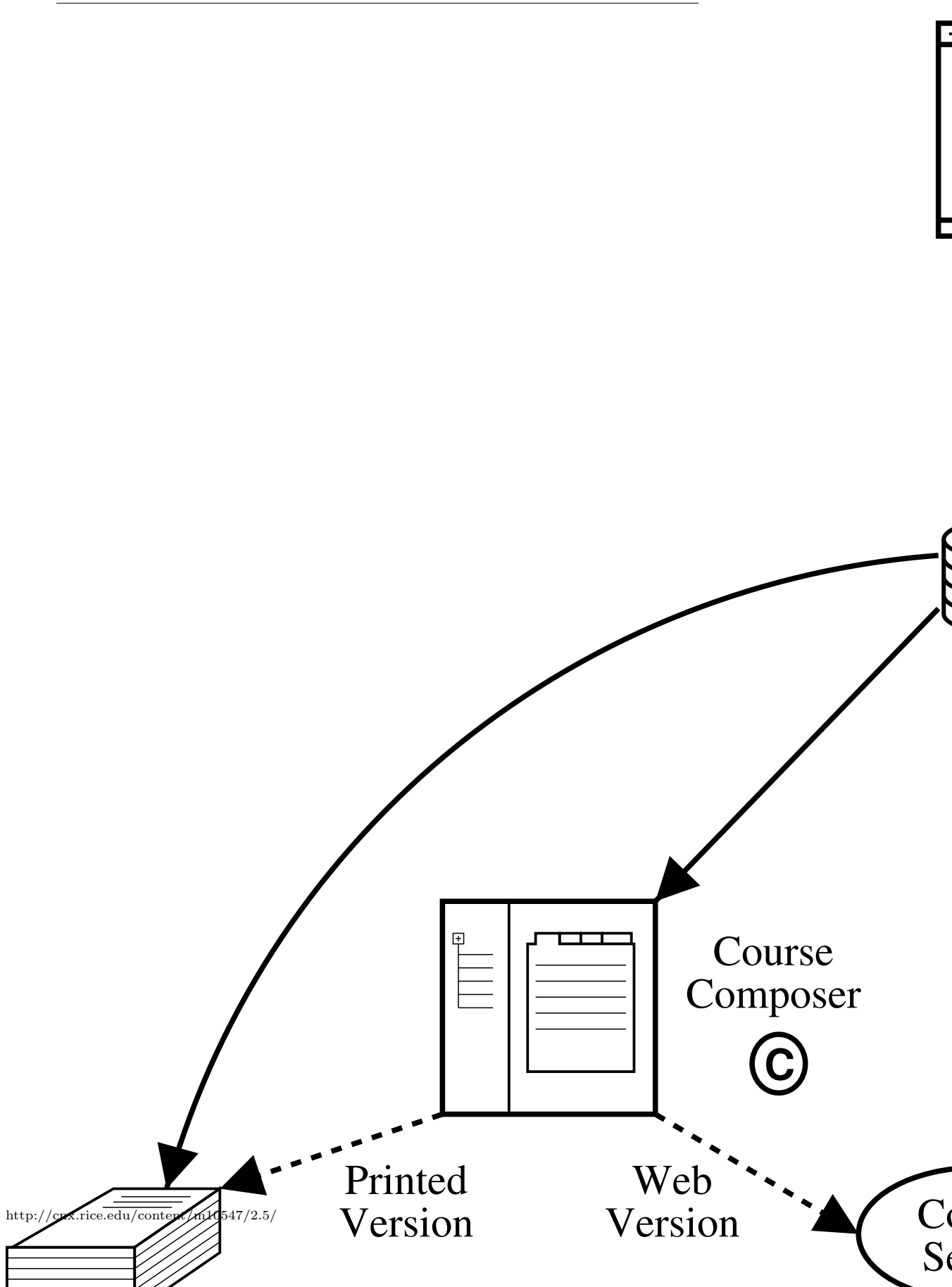
## 3 Implementation

At the heart of the Connexions System are the modules stored in our central repository. The repository stores the modules' text in the Concurrent Versions System (CVS)pg ??, a version control software package that provides revision history and allows for concurrent editing. A relational database (specifically, PostgreSQLpg ??) houses the metadata associated with each module for easy search and retrieval. Each of our three user communities interacts with the repository via specialized tools as depicted in Figure 1. The author community generates high-quality content modules and makes them available by submitting them to the repository. Members of the instructor community weave these modules together into customized courses and place them in the repository or an independent course server. Students then use the provided course paths as they navigate the repository.

### 3.1 Storing Modules: Content vs. Presentation

Due to the distributed, collaborative nature of this project, it is critical to capture the semantics of the authors' work, rather than presentation-specific syntax. This encourages authors to concern themselves with the meaning and structure of their content, rather than the notation and how it will look. Towards this end, we have adopted the eXtensible Markup Language (XML)pg ?? for our material in general, and specifically content MathMLpg ?? for mathematics. We have developed a lightweight document markup language called CNXMLpg ?? for storing the basic document structure, incorporating content MathML via the XML namespace mechanismpg ?? to mark up any mathematics present in the module.

The use of XML has several advantages. The content may be stored in a single source format and styled for multiple output modalities: web, print, ebook, audio, or even future



formats not yet defined. This flexibility is key in freeing us to devise new means of visualizing information without being concerned with compatibility with particular browsers, or designing to the lowest common denominator functionality of older browsers. Adapting our content to a new visualization capability requires developing a new stylesheet, rather than than recoding and transforming documents en masse.

This separation of content and style also allows for a coherent presentation of materials from many authors, including mathematical notation. If multiple authors were to write modules using their own preferred mathematical notation, instructors would be constrained to assembling courses with internal notational inconsistencies. Such discontinuities are very disruptive for students, causing cognitive dissonance and dramatically slowing learning and their attainment of educational goals. By using content MathML and the XML Stylesheet Language for Transformations (XSLT)pg ??, we allow instructors to create courses with consistent notation throughout, independent of the notational preferences of the various original authors.

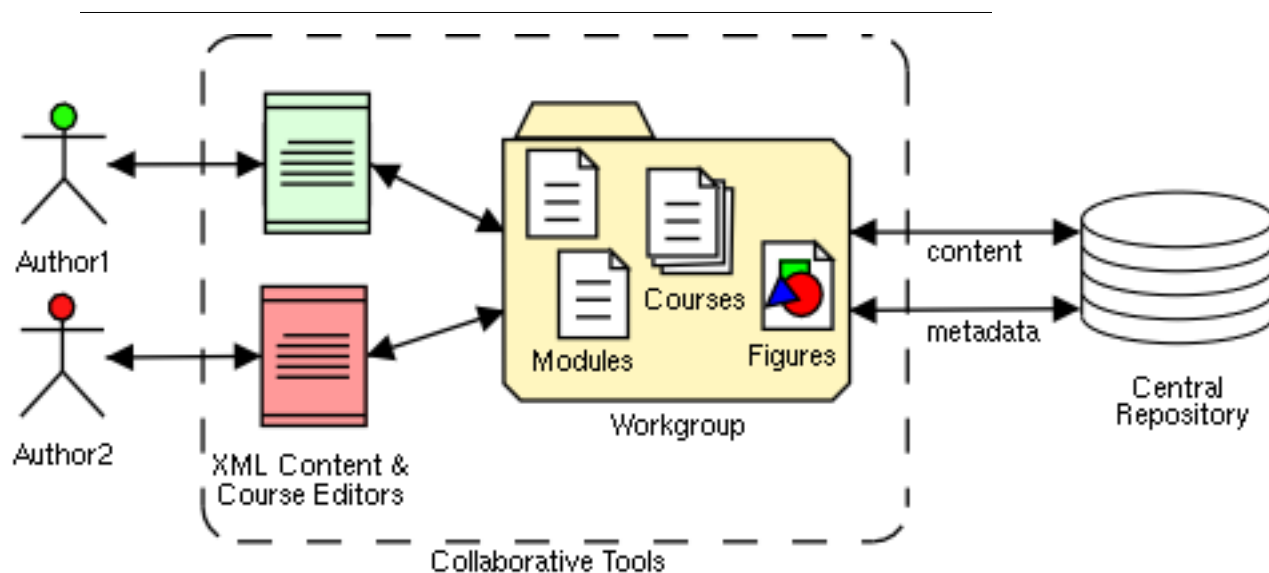
### 3.2 Authors

Connexions module authors take advantage of our web-based collaborative editing system to create, share, and maintain modules in the repository. This system is based on Zopepg ??, a web application server platform. Using the editing system, authors form workgroups to collaborate on modules and courses, sharing ideas and swapping successive versions until they feel the content is ready to be submitted to the repository (See Figure 2.) The interface provides facilities for entering module metadata as well as for editing the text of the module itself. It also allows authors to list related materials to be presented as auxiliary links (See students (Section 7) for details.) The interface also provides a download/upload feature for the module text so those who dislike editing in a browser window may use an external editor. Since XML is an open standard, we need not require any particular editing software, and authors may use the editor of their choice. With XML's wide industry support, several editors have emerged. Many of these are geared mainly toward tree-structured data entry, but some do provide a fair document markup interface (Morphonpg ??, XXEpg ??) These are still unacceptable for handling MathML, however, where a specialized editor is required.

We have explored several avenues for producing content MathML. The text editor Emacs has an XML editing modepg ?? that many of our staff use, although we wouldn't recommend this for non-technical users. Many mathematics manipulation packages (Mathematicapg ??, Maplepg ??, MathCADpg ??) provide MathML export capabilities, but often the output is only available as presentation MathML. We have looked at two dedicated math-entry programs: WebEQpg ?? and EZMathpg ?? . WebEQ's input is more geared towards presentation, but it does attempt to generate content MathML, interpreting the the author's intent and providing some visual feedback when the meaning is unclear. EZMath takes a novel approach, requiring input in a natural-language style instead of a using a graphical interface. Since it retains the semantic content, it has no difficulty outputting content MathML, although it does require the user to learn a new syntax. In addition to improved content MathML support, we would like to see two features in MathML editors. The first is better extensibility, including support for `csymbol` tags and OpenMathpg ?? definitions. The second is better integration with other general-purpose XML editors, perhaps as a plugin.

### 3.3 Instructors

An instructor's role in the Connexions system is to serve as a guide for their students through the sea of modules. They do this by using our **Course Composer** software to build



**Figure 2:** Authoring Work Flow

touring itineraries or "coursepaths". Instructors build up chapters and courses by combining modules written by other authors with their own material. In addition, Course Composer lets instructors provide additional links on the modules in their course, beyond what the original authors have specified. These "instructor-imposed" links are course-specific, and stored without modifying the original modules. Each imposed link belongs to an instructor-specified category (such as "supplemental" or "prerequisite") and can be assigned a number from one to five indicating the strength or relatedness of the link. These links are then presented to the students to help them understand the relationships between materials and to encourage individual exploration. The Course Composer creates a course description file out of the imposed links along with the selected course path. The repository stores the course description file in the Resource Description Framework (RDF) format for student access.

One potential hurdle in assembling materials from different authors is mismatched notation. Connexions solves this problem by utilizing content MathML. The repository stores only semantic information, leaving notation and presentation specifics up to the instructor. Instructors are allowed (and encouraged) to specify both cosmetic and notational parameters for displaying their course (eg. background color, or the use of  $j$  vs.  $i$  for  $\sqrt{-1}$ .) Course Composer stores these choices in the course description file. They are then used to select an XSLT stylesheet (and possibly a set of passed parameters) to transform the course for display. In this way the instructor can customize the the course to his or her taste, achieving a consistent presentation. This is a limited implementation of the techniques described by Naylor and Watt.

In addition to electronic display on the web, instructors may wish to provide a printed version of the material. This can easily be done from the same source, thanks to the separation of content from presentation. We use XSLT stylesheets to transform the RDF course description file and the course modules (stored in CNXML and content MathML) into a single file consisting of XSLFO and presentation MathML. This is then fed to

passivetexpg ??, creating a PDF file that can be sent to Kinko's or a university press or printshop for printing and distribution to the students. Documents produced this way achieve high quality mathematics layout thanks to the TeXpg ?? rendering engine, and can have autogenerated tables of contents and figures, as well as an index based on keywords and vocabulary terms.

### 3.4 Students

To access courses, students use our **Roadmap** navigational software. The Roadmap queries the repository for a list of available courses (stored as RDF) and presents this list to the student. The student then selects his or her course, and the roadmap downloads the corresponding course description file. Once the file is downloaded, the Roadmap presents the student with an outline of the course (See the left hand side of Figure 3) and the set of imposed links for the current module (Right hand side of Figure 3.) The course view gives students a high-level view of the course materials, allowing them to track their progress. The link view lets them explore relationships and experience how the materials are related. Students may also choose to gain a different perspective by switching link views from the instructor-imposed links to the auxiliary links provided by the module authors. The Roadmap is currently implemented as a browser add-on using Netscape's eXtensible User-Interface Language (XUL)pg ?? and is only available for Netscape 6 or the Mozilla browser. We are working on a similar plugin for Microsoft's Internet Explorer, but in the meantime the system falls back to generating HTML webpages that achieve a similar effect using frames.

With or without the Roadmap, when a student retrieves a module for viewing, XSLT stylesheets transform the content-based XML (CNXML and content MathML) into appropriate presentation dialects (XHTML and Presentation MathML.) The particular stylesheets used are determined by the Connexions server, based on the student's specific browser and the parameters specified by the course instructor (if the student is accessing the module as part of a course.) In this way students see the series of modules as a stylistically coherent text using their instructor's preferred notation. These transformations are largely done server-side, although we are investigating the use of client-side transformations for the future. A notable exception to this is our client-side use of the MathML stylesheet provided by the W3C MathML working grouppg ??, enabling the same source to be used by different rendering environments. This allows the same module to be viewed by browsers with a wide range of functionality. We currently support Mozilla with built-in MathML support and Internet Explorer 6 with Design Science's MathPlayer pluginpg ?? (Internet Explorer 5.5 is also supported with an upgrade to to version 3 or greater of Microsoft's MSXML parser.) In the future we hope to support other browsers by providing a transformation to bare-bones HTML with images for the math.

## 4 Future Directions

For future research, we are investigating novel methods of content creation that more directly capture experts' knowledge than traditional, linearly-structured papers (such as this one.) One such approach to collaborative, web-based authoring that tends to generate highly interconnected content is the Wikipg ??. We are investigating conversion tools to enable the use of Wikis as outlining and rough-draft tools to generate interlinked Connexions modules. Another approach we are investigating is the **concept map**pg ?? pg ?? both as an assessment tool, and for creating courses. Instructors could create a graphical "map" of the material and its relationships, which would then be converted into a course description file.

The screenshot shows the Connexions project interface for the course 'Signals and Systems' at Rice University. The main content area is titled 'Linear Algebra: The Basics'. It includes a summary paragraph, a detailed introductory paragraph, and a section on 'Linear Independence' with a definition and an example of two non-linearly independent vectors.

**the connexions project**  
Rice University

Signals and Systems Previous Next

Course

- Signals and Systems
  - Cover Page
  - Introduction
  - Signals and Systems: A First Look
  - Continuous-Time Systems in Time Domain
  - Linear Algebra Overview
    - Linear Algebra: The Basics**
      - Eigenvectors and Eigenvalues
      - Matrix Diagonalization
      - Eigen-stuff in a Nutshell
      - Eigenfunctions of LTI Systems
    - Fourier Series and Orthogonal Expansions
    - Fourier Transform
    - Sampling Theorem
    - Laplace Transform and System Design
    - Z-Transform and Digital Filtering

## Linear Algebra: The Basics

**Summary:** This module will give a very brief tutorial on some of the basic terms and ideas of linear algebra. These will include linear independence, span, and basis.

This brief tutorial on some key terms in linear algebra is not meant to replace or be very helpful to those of you trying to gain a deep insight into linear algebra. Rather, this brief introduction to some of the terms and ideas of linear algebra is meant to provide a little background to those trying to get a better understanding or learn about eigenvectors and eigenfunctions, which play a big role in deriving a few important ideas on Signals and Systems. The goal of these concepts will be to provide a background for signal decomposition and to lead up to the derivation of the [FOURIER SERIES](#).

### Linear Independence

A set of vectors  $\{x_1, x_2, \dots, x_n\}$ ,  $x_i \in \mathbb{C}^n$  are **linearly independent** if none of them can be written as a linear combination of the others.

**DEFINITION 1: Linearly Independent**

1. For a given set of vectors,  $\{x_1, x_2, \dots, x_n\}$ , they are linearly independent if

$$c_1x_1 + c_2x_2 + \dots + c_nx_n = 0$$

only when  $c_1 = c_2 = \dots = c_n = 0$

**EXAMPLE:** We are given the following two vectors:

$$x_1 = \begin{pmatrix} 3 \\ 2 \end{pmatrix}$$

$$x_2 = \begin{pmatrix} -6 \\ -4 \end{pmatrix}$$

These are *not linearly independent* as proven by the following statement, which, by inspection, can be seen to not adhere to the definition of linear independence stated above.

Figure 3: The Connexions Roadmap

Similarly, on the client side we would like to develop a more visually-oriented Roadmap tool, by representing modules and links graphically. This would truly allow students to see the relationships between concepts as they navigate the interlinking web of modules.

Since most other document formats are not as semantically driven as CNXML, auto-conversion from, for example, Microsoft Word documents or PowerPoint presentations is not possible. However, the creation of "wizards" that guide an author through the process of adding semantic information while converting markup is certainly possible, and is an avenue we are interested in pursuing.

The W3C's Annotepg ?? project provides a method of annotating web pages without having to modify the source. Instead, the annotations are stored on an annotation server using RDF and XPointerpg ?? . We plan to incorporate this technology into our software as well, allowing instructors to add their own comments onto course modules. This could be used, for example to call attention to important concepts or refer back to points made in class lectures. It could also be useful for enabling students to create personal "margin notes" while they are reading the materials.

Widespread adoption of the Connexions system will require expansion beyond a single, central repository. We are investigating the technologies required to support different levels of distributed repositories, ranging from closely cooperating, federated repositories to independent access of multiple repositories whose only commonality is standards compliance and a common toolset.

As the size and richness of the content pool increases, the need for more advanced searching and discriminating tools will become critical. We are addressing this problem in two ways. Our first approach incorporates the use of external metadata stores, and advanced indexing technologies. The second approach is perhaps more interesting from a community point of view. We are investigating the requirements to support third-party "lenses", which will focus a user's view of the repository on selected content. These could serve as content guides for instructors and authors, much in the way courses guide students.

## 5 Conclusion

Connexions would not have been possible without the development and maturation of XML and related technologies such as MathML and XSLT that fulfill the promise of separation of content and presentation. This separation, combined with a sufficiently fine-grained modularization allows authors to create easily-maintainable and shareable content. Publication and exchange of this content through an open repository enables instructors to pull together disparate components to create customized courses with a coherent mathematical notation. We are currently working with education assessment professionals to determine the effectiveness of this approach from the student perspective. Our experience to date using Connexions has taught us that both students and faculty appreciate the availability of high-quality materials in both web and print formats. We expect to gain widespread adoption of our tools and approach as we expand the content pool and incorporate new technologies.

## 6 References

1. Annotea Project<sup>2</sup>, World Wide Web Consortium
2. "Assessing Science Understanding: A Human Constructivist View". Mintzes, J.J., Wandersee, J.H. & Novak, J.D. (2000) San Diego: Academic Press

---

<sup>2</sup><http://www.w3.org/2001/Annotea/>

3. CNXML 0.4 Language Specification<sup>3</sup>, Coppin, S., *et al.*, 12 March. 2002
4. Concurrent Versions System (CVS)<sup>4</sup>, SourceGear Corporation
5. "Connexions: An Architecture for Web-based Educational Materials", Brent Hendricks, MS Thesis, Rice University, Jan. 2001
6. Extensible Markup Language (XML) 1.0 (Second Edition)<sup>5</sup>, Bray, T., *et al.*, 6 Dec. 2000
7. Extensible Stylesheet Language (XSL) Version 1.0<sup>6</sup>, Adler, S. *et al.* 15 October 2001
8. "Learning How to Learn". Novak, J.D. and Gowin, D. B. (1984). New York and Cambridge, UK: Cambridge University Press
9. MathCAD<sup>7</sup>, MathSoft Engineering and Education
10. Mathematica<sup>8</sup>, Wolfram Research, Inc.
11. Mathematical Markup Language (MathML) Version 2.0<sup>9</sup>, Ausbrooks, R. *et al.*, 13 Nov. 2000
12. Mathematics on the Web: The EzMath notation<sup>10</sup>, Ragget, D. and Batsalle, Davy 27 Nov. 1997
13. MathPlayer<sup>11</sup>, Design Sciences
14. Meta Style Sheets for the Conversion of Mathematical Documents into Multiple Forms<sup>12</sup>, Naylor, B. and Stephen Watt, S., (2001) First International Workshop on Mathematical Knowledge Management: MKM'2001 RISC, Hagenberg
15. Morphon XML-Editor Suite<sup>13</sup>, Morphon Technologies
16. Namespaces in XML<sup>14</sup>, Bray, T. *et al.* 14 Jan. 1999
17. The OpenMath Standard (Draft)<sup>15</sup>, The ESPRIT OpenMath Project, Numerical Algorithms Group, Ltd. Oxford, UK
18. PassiveTeX<sup>16</sup>, Rahtz, S. Feb 2002
19. PostgreSQL Data Base Management System<sup>17</sup>, The PostgreSQL Global Development Group
20. PSGML: A GNU Emacs mode for SGML files<sup>18</sup>, Staffin, L.

---

<sup>3</sup><http://cnx.rice.edu/cnxml/0.4/spec/>

<sup>4</sup><http://www.cvshome.org>

<sup>5</sup><http://www.w3.org/TR/2000/REC-xml-20001006>

<sup>6</sup><http://www.w3.org/TR/2001/REC-xsl-20011015/>

<sup>7</sup><http://www.mathsoft.com>

<sup>8</sup><http://www.wolfram.com/products/mathematica/>

<sup>9</sup><http://www.w3.org/TR/2000/CR-MathML2-20001113>

<sup>10</sup><http://www.w3.org/People/Raggett/EzMath/EzMathPaper.html>

<sup>11</sup><http://www.dessci.com/webmath/mathplayer/>

<sup>12</sup><http://www.risc.uni-linz.ac.at/institute/conferences/MKM2001/Proceedings/naylor.pdf>

<sup>13</sup><http://www.morphon.com/xmleditor/index.shtml>

<sup>14</sup><http://www.w3.org/TR/1999/REC-xml-names-19990114>

<sup>15</sup><http://www.openmath.org/>

<sup>16</sup><http://www.hcu.ox.ac.uk/TEI/Software/passivetex/>

<sup>17</sup><http://www.postgresql.org>

<sup>18</sup><http://www.lysator.liu.se/projects/about-psgml.html/>

21. Putting mathematics on the Web with MathML<sup>19</sup>, The W3C Math Working Group, Mar. 2002
22. Resource Description Framework (RDF) Model and Syntax Specification<sup>20</sup>, Lassila, O. and Swick, R. 22 Feb. 1999
23. The TeXbook, Knuth, D. Addison-Wesley Publishing Co. Jun. 1986
24. Waterloo Maple<sup>21</sup>, Waterloo Maple Software
25. WebEQ Developer's Suite<sup>22</sup>, Design Science Incorporated
26. Wiki Wiki Web<sup>23</sup>, Cunningham, W.
27. XML Pointer Language (XPointer) Version 1.0<sup>24</sup>, DeRose, S. *et al.* 11 Sep. 2001
28. XMLMind XML Editor (XXE)<sup>25</sup>, Pixware
29. XSL Transformations (XSLT) Version 1.0<sup>26</sup>, Clark, J., 16 Nov. 1999
30. XUL Programmer's Reference Manual<sup>27</sup>, The Mozilla Project
31. Zope<sup>28</sup>, Zope Corporation

---

<sup>19</sup><http://www.w3.org/Math/XSL/>

<sup>20</sup><http://www.w3.org/TR/1999/REC-rdf-syntax-19990222>

<sup>21</sup><http://www.maplesoft.com>

<sup>22</sup><http://www.dessci.com/>

<sup>23</sup><http://c2.com/cgi-bin/wiki?WikiWikiWeb>

<sup>24</sup><http://www.w3.org/XML/Linking/>

<sup>25</sup><http://www.xmlmind.com/xmleditor/>

<sup>26</sup><http://www.w3.org/TR/1999/REC-xslt-19991116>

<sup>27</sup><http://www.mozilla.org/xpfe/xulref/>

<sup>28</sup><http://www.zope.com>