

# Dynamic Assembly of Learning Objects

Robert Farrell  
IBM Research  
30 Saw Mill River Rd  
Hawthorne, NY 10532  
+01(914)-784-7432  
[robfare@us.ibm.com](mailto:robfare@us.ibm.com)

Soyini D. Liburd  
MIT  
77 Mass. Ave.  
Cambridge, MA 02139  
[soyini@mit.edu](mailto:soyini@mit.edu)

John C. Thomas  
IBM Research  
30 Saw Mill River Rd  
Hawthorne, NY 10532  
+01(914)-784-7561  
[jcthomas@us.ibm.com](mailto:jcthomas@us.ibm.com)

## ABSTRACT

This paper describes one solution to the problem of how to select sequence, and link Web resources into a coherent, focused organization for instruction that addresses a user's immediate and focused learning need. A system is described that automatically generates individualized learning paths from a repository of XML Web resources. Each Web resource has an XML Learning Object Metadata (LOM) description consisting of General, Educational, and Classification metadata. Dynamic assembly of these learning objects is based on the relative match of the learning object content and metadata to the learner's needs, preferences, context, and constraints. Learning objects are connected into coherent paths based on their LOM topic classifications and the proximity of these topics in a Resource Description Framework (RDF) graph. An instructional sequencing policy specifies how to arrange the objects on the path into a particular learning sequence. The system has been deployed and evaluated within a corporate setting.

## Categories and Subject Descriptors

K.3.1 [Computer Uses in Education] Computer-Managed Instruction

## General Terms

Algorithms, Documentation, Design, Experimentation, Human Factors, Standardization, Languages.

## Keywords

Learning Object, Metadata, LOM, Semantic Web, RDF, Instruction, Content Management, Data Retrieval, Information Retrieval, Assembly, Organization, Linking.

## 1. INTRODUCTION

There is a movement in the education and training community to promote the development of Web-based applications in a more modular fashion. Related media resources are grouped together into aggregates called "learning objects" that can be developed relatively independently from one another and made accessible on the WWW or a large content repository. Learning objects can then be reused across courses, disciplines, and institutions. Ideally, Web-based course developers could quickly assemble learning objects into coherent and effective Web-based learning experiences [1]. However, current Web-based courses are still developed largely manually.

This paper focuses on the problem of how to automatically assemble learning objects into simple, short, focused, Web-based "custom courses". This process, which we call "Dynamic Assembly", includes the process of connecting relevant search results into a learning path, sequencing the selected learning objects on the path, and linking the selected learning objects into an organized structure. Dynamic Assembly is based upon parameters that are available only when a learning session starts, such as the learner's keyword query, desired level of detail, and the amount of time they have available to learn. The query is typically based upon a task focus, professional development opportunity, or specific interest.

We have developed a software component, the Dynamic Assembly Engine, to assemble standards-based learning object content. This component is integrated into the Custom Course System, a web-based e-learning system we have deployed within IBM on a pilot basis for employee training on a range of information technology topics [2].

### 1.1 Motivation

People under time pressure and high demands on their productivity are often motivated to educate themselves on new topics, but do not necessarily have the time to take a full course of instruction. In corporations, for example, technical professionals need to learn new things in the context of their job tasks, but rarely have the time to take full e-learning courses or attend class. These practitioners often search the Web or company databases, or scan through technical material from a variety of sources to quickly learn what they need to know [3][4]. For employees new to a subject, information from these sources is often difficult to find and organize for effective learning. Knowledge acquired is often disconnected, forgotten, or not effectively integrated into practice. These observations are not unique to the corporate environment. Learners in colleges and universities are motivated differently, but also spend considerable effort searching for information to gain knowledge and skills. In all cases, information on the web is often not effectively organized and learners spend considerable time in unproductive interactions and may not properly integrate information to address their immediate learning need. A more flexible approach is needed that is sensitive to each learner's unique needs and context, but also provides focused and structured learning.

### 1.2 Approach

To address these challenges, we developed an approach to web-based learning based upon three principles. First, modularize; create small learning objects that can be sufficiently independent and de-contextualized to serve as "building blocks" while describing them with sufficient metadata to enable their broader use. Second, customize; assemble together a small number of these learning objects into a coherent and logically sequenced

learning path customized for the individual. Third, empower; let learners drive the assembly of the learning path from their own needs. The resulting assembled paths can be archived, shared between learners, or disseminated throughout an organization.

This approach has many advantages. First, self-directed learners are often more motivated [11]. Creating their own learning paths engages learners in thinking about what they need to know and do. Second, learning paths created dynamically can be tailored to address a particular learning gap, to the extent that the learner is aware of that gap. Lastly, learners are focused on a small set of resources relevant to their task, context, need, or interest [7]. Learning paths in our system are displayed as short “Custom Courses”. Courses provide a familiar structure for learning and set of cues to enable learners to easily navigate from one learning object to another.

## 2. System

We developed the Custom Course System over a period of 18 months at IBM Research [2]. We have made the system available for four months, 24 hours a day and seven days a week on a trial basis. Employees working in IBM’s information technology services businesses have been able to access the system from three different web sites to learn about the WebSphere product [15] and related technologies. The system has had over 300 users from around the world. This section describes the user experience, system architecture, data formats, process flow, and some details of implementation.

### 2.1 User Experience

Users login to the system and are asked to use the system to fulfill their learning needs. A *Home page* describes the system and provides hyperlinks to the *Course assembly page* and the *My courses* page. On the *Course assembly page* (see Figure 1) users enter topic keywords (e.g., ‘wsdl’), optional desired course duration, and an optional Search Scope and then press the “Assemble” button (or select the Assemble course menu entry) to create a custom course. Course duration ranges start with “1 to 2 minutes” and go up to “90 to 120 minutes”. A Search Scope of “overview” explores related topics, while a Search Scope of “indepth” focuses primarily on a single topic. Advanced query options allow users to restrict the search to learning objects to particular resource types, levels of difficulty, and other preferences.

A Manual assembly option allows users to search and select learning objects. Users can additionally select learning objects recommended by the Dynamic Assembly Engine.

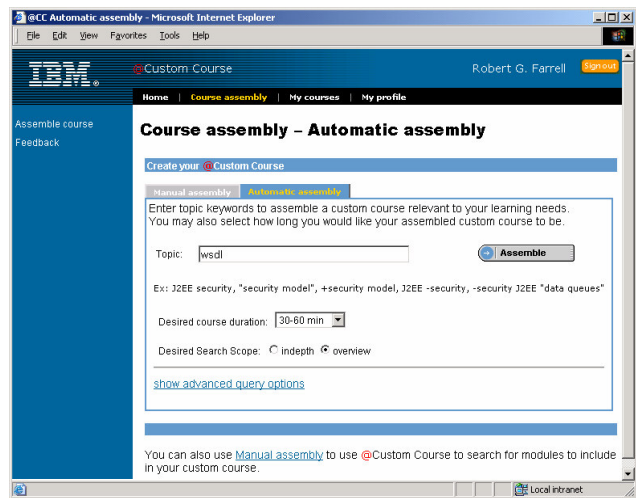


Figure 1: The Course Assembly page

The system returns with a Custom Course outline page that shows learning objects as a sequence of numbered “lessons”. Each lesson title is a hyperlink to launch the course in the Course Player starting at the given learning object. Alternatively, users can select the “Play course” menu entry or “Play” button (not shown) to start at the first learning object. To the right of each lesson title is the role of the learning object for instruction (e.g., “Introduction”); and its associated topic (e.g., “WebSphere”). If the user does not like the sequence offered by the Custom Course System, he or she can drag and drop lessons within the browser to reorder. Next to each lesson is its typical learning time. Under each lesson title is a listing of the lesson’s educational objectives.

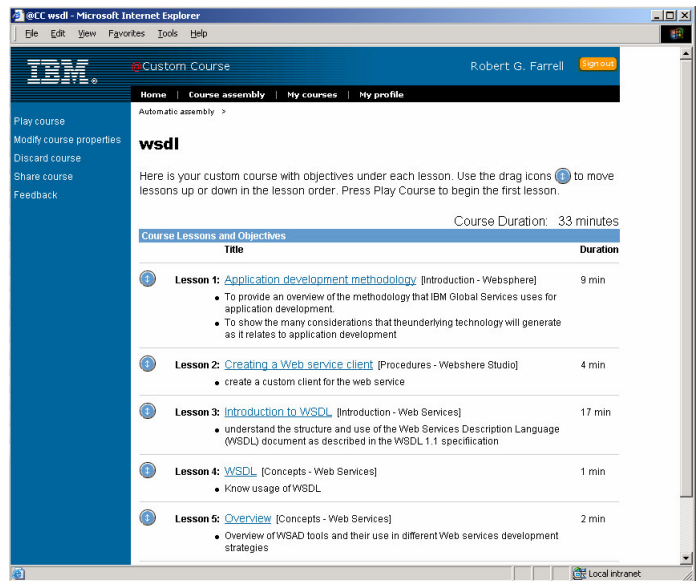


Figure 2: Custom Course outline page

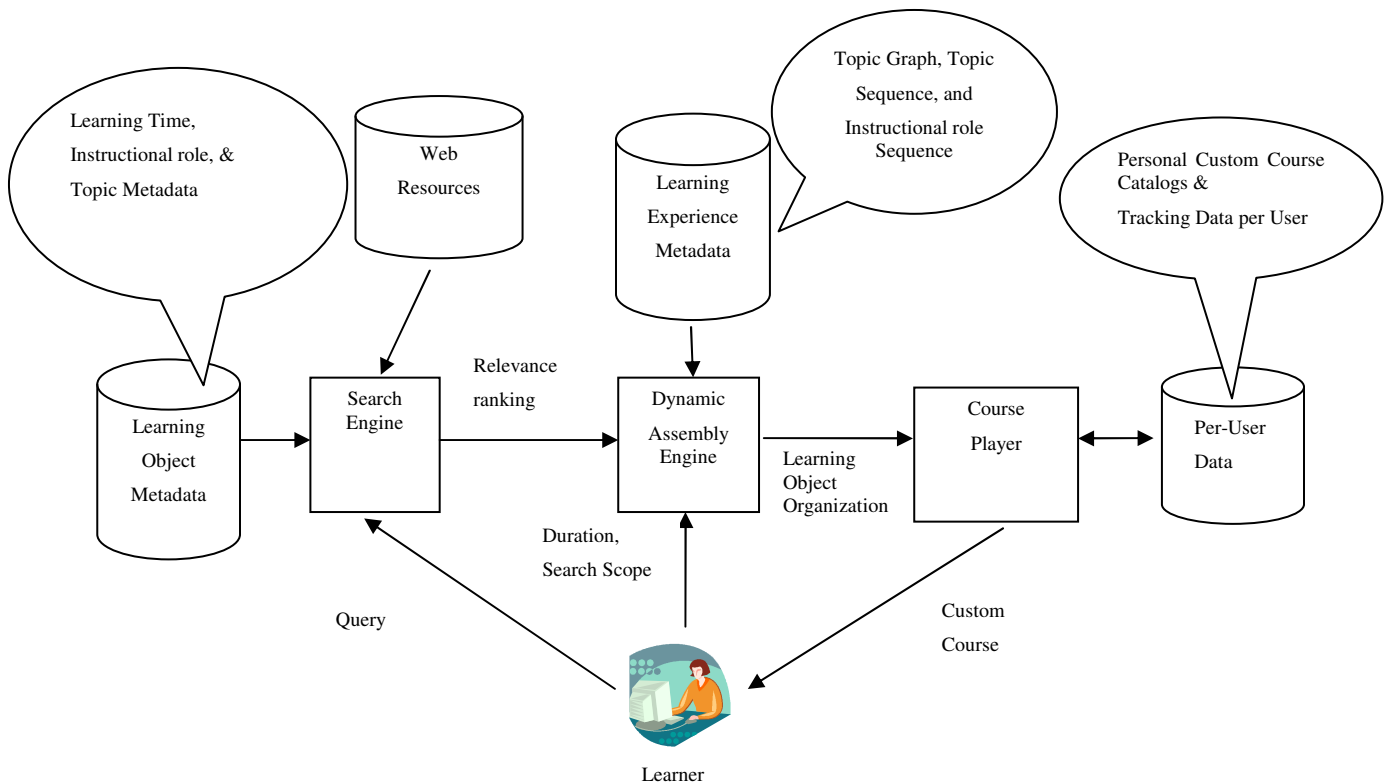
Lessons listed in the custom course outline are first sequenced by topic. The topic order shown is WebSphere, WebSphere Studio, and then Web services. Within a topic, lessons are ordered by their instructional role. For example, “Introduction to WSDL” has an instructional role of “Introduction” and is thus listed before the next two lessons, which are “Concepts”. When lessons have the same instructional role and topic, such as the “WSDL” and “Overview” lessons, they may be ordered by other criteria, such as their position within the original source materials.

Assembled courses are archived and stored in the learner's personal course catalog. Custom courses have an initial title that is copied from the query used to produce the course, but this title and other course properties can be modified by the learner. Courses can be shared with other learners by specifying an e-mail address. Users must register and then they can immediately start viewing the shared course's Custom Course Outline page. The identity and e-mail address of the user offering the course is displayed along with the offer date. Users can provide feedback on the system, a custom course, or an individual lesson using the Feedback feature.

## 2.2 System Architecture

The Custom Course System consists of a Search Engine, the Dynamic Assembly Engine, and Course Player (see Figure 1). The

Search Engine accesses a full-text index created from a combination of the Web resource XML file and the metadata file and returns a relevance ranking of search results. The Dynamic Assembly Engine maps search results to topic categories, maps objects from search results to one or more categories in a graph, computes statistics for each category based upon the mapped objects, and then uses both the statistics for each category and the relationships between categories encoded as the edges of the graph to find a best path through the graph. The path is then linearized into an XML form that can then be played in the Course Player.



**Figure 3: Custom Course System architecture**

These system modules consist of several major services:

- The *Search Engine* consists of:
  - Search Engine Service – combines metadata XML and content XML, indexes the combined files, and orders references to the XML content in order of relevance to the query to generate search results.
- The *Dynamic Assembly Engine* consists of:
  - Path Generation Service – collects search results that are closely related according to their Learning Object Metadata into a coherent path, as defined by relationships between metadata values appearing in the Learning Experience Metadata, respecting the constraints of Duration and Search Scope.
  - Sequencing Service – sorts the objects on the path, respecting the Topic Sequence and the Instructional role Sequence. Accesses the Course Manager Service to add each of the

objects returned from the Sequencing Service to the learning object organization.

- The *Course Player* consists of:
  - Learner Authentication Service - verifies the user name and password and registers users with the system.
  - Learner Profile Service - allows users to modify personal attributes and updates Tracking Data such as the number of times a learning object has been viewed outside a course, assembled into a course, bookmarked, or played as part of a course.
  - Course Navigation Service – fits the learning object organization into the IMS Manifest structure, displays the Custom Course to the user, processes navigation requests, including requests to bookmark lessons and suspend the course, and communicates with the Learner Profile Service to store tracking data.

These components use a common data management layer consisting of a Metadata Manager Service that provides access to the Learning Object Metadata associated with each Learning Object and a Course Manger Service that builds simple linear “Custom Course” structures from learning paths and provides access to the learner’s Personal Custom Course Catalog.

## 2.3 Data Model

The Dynamic Assembly Engine uses three standards as the basis for its interoperability with other applications: IEEE LOM, IMS Content Packaging, and W3C RDF. This section describes each of these standards and their use in the system.

The IEEE LOM and IMS Content Packaging are currently two major components of the Shareable Content Object Reference Model (SCORM) proposed by the ADL (Advanced Distributed Learning) organization to enable interoperability between content and learning management systems.

RDF is an emerging WWW standard and is used by the W3C’s Semantic Web activity. While the Dynamic Assembly Engine works with the standard LOM format. Learning objects must have certain metadata elements filled in with values an extended “instructional role” vocabulary. These metadata values also appear in the Instructional role Sequence. Thus, while the approach is applicable to all learning objects, good Dynamic Assembly results can only be achieved by providing metadata from a specific extended vocabulary.

### 2.3.1 IEEE Learning Object Metadata

The IEEE Learning Object Metadata (LOM) standard [19] provides an information model that defines the structure of a metadata instance for a learning object. A metadata instance describes relevant characteristics of the learning objects grouped into general, life cycle, meta-metadata, educational, technical, educational, rights, relation, annotation, and classification categories. Dynamic Assembly depends only upon the General, Educational, and Classification metadata.

The following LOM metadata elements of Learning Objects are used by the Dynamic Assembly Engine:

1. Identifier [*1.1 General*] – A globally unique identifier for this learning object encoded as a URI.
2. Title [*1.2 General*] – The name given to this learning object. This is encoded as a “LangString” data type.
3. Instructional role, *added as an additional Learning Resource Type [5.2 Educational]* –We extended the vocabulary to add a set of “instructional role” types. These types represent potential roles for the learning object in future assemblies. It is used to order resources within classifications using the Instructional role Sequence.
4. Typical Learning Time [*5.9 Educational*] – Approximate or typical time it takes to work with or through the learning object for the typical intended target audience. Encoded as a Duration datatype that is mapped to an absolute number of seconds for comparison with the desired course duration input by the user.
5. Topic, *added as an additional Taxon Path [9.2 Classification]* – The taxonomic path with a particular classification system. The Id of the “leaf” node in this taxonomic path [ 9.2.2.1 Classification, Taxon Path, Taxon] must be a topic URI. This URI can then be related to other topics in the Topic Graph.

### 2.3.2 IMS Content Package

The IMS Content Packaging Specification [20] provides the functionality to describe and package learning materials, such as an individual course or a collection of courses, into interoperable, distributable packages. Content Packaging addresses the description, structure, and location of online learning materials. An IMS content package is created from a manifest file containing metadata, an organizations structure, a set of references to resources, and a collection of supporting files.

The Dynamic Assembly Engine assigns learning objects as resources along with a reference to their associated metadata description. Resources are referenced as item elements within an organization container. Our implementation of Custom courses creates a default linear organization of the items [24]. The organization is placed in an organizations structure within the manifest file. The Dynamic Assembly Engine also assigns metadata to the entire custom course, including a Title based upon the user’s query and Typical Learning Time duration from the combined Typical Learning Time values from constituent learning objects. Multiple custom courses are combined as submanifests within a larger manifest that represents the personal Course Custom Course Catalog.

### 2.3.3 RDF

Resource Description Framework (RDF) is a specification for processing metadata about Web resources that is based on a model of entities and properties. The entities in RDF graphs are Web resources and the properties are characteristics, attributes, aspects, or relations to other entities. The ability to describe relations between Web resources is important for Dynamic Assembly.

### 2.3.3.1 Topic Graph

The Topic Graph includes nodes for topics and edges for topic relationships, encoded as RDF entities and properties, respectively. We provide a list of 10 different semantic relations that are sufficient to describe products, processes using those products, and technologies to facilitate both. We anticipate that the list of relationships will have to be expanded to describe the relationships between topics in various classifications typically used by learning objects.

A sample Topic Graph is shown in Figure 4. Entities are references to IBM products, such as “WebSphere Portal” and “WebSphere Studio”, and technologies such as “J2EE” and “Web Services”. Technologies are connected to products through a Used-by relationship, while products sold in a family are connected to main product family through a Has-part relationship. Relationships are directed arcs in the graph, but may have an inverse (e.g., “Has-part” and “Part-of”).

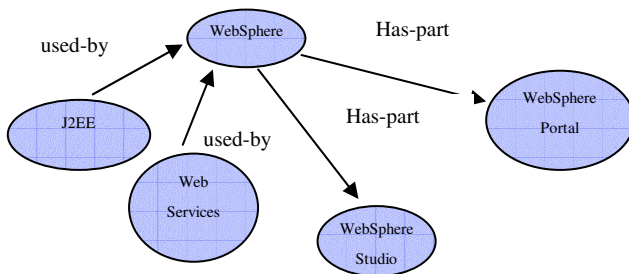


Figure 4: A Topic Graph

A total ordering of topics is stored in the Topic Sequence used by the Sequencing Service.

### 2.3.3.2 Instructional role Sequence

The instructional role sequence is an ordered list of particular learning resource types that represent how learning objects will be ordered within topics. We have extended the learning object metadata to include a special learning resource type called “instructional role”. These instructional role types fall into three categories: rhetorical, cognitive, and domain-specific. The rhetorical category defines elements of discourse (e.g., introduction, motivation, and conclusion). These rhetorical elements provide proper structuring aimed at increasing comprehension or instilling motivation. The cognitive category encodes Merrill’s taxonomy for types of knowledge (facts, principles, concepts, processes, procedures) [9]. The domain-specific category includes resource types specific to the target domain. In our case, we encoded information technology resource type vocabulary, such as system, architecture, and code listing.

The Instructional role Sequence encodes a particular linear order across all three of the instructional role categories. The particular vocabulary used can be modified as long as the Learning Resource Types stored in the LOM correspond to the values in the Instructional role Sequence. For example, one could include Bloom’s taxonomy of educational objectives [10] or other taxonomies of learning outcomes. We find that the level of description offered by our learning objects is at a finer grain than the typical executable learning object defined in SCORM. Thus, the Custom Course may correspond to a dynamic ordering of

## 2.4 Process Flow

Given the user’s query, the Search Engine Service returns a ranked list of relevant search results. The query may include parameters other than keywords, including difficulty level, media, and other parameters based on learning object metadata. The learner can also provide a list of search results from this list to serve as “targets” when assembling a coherent subgraph connecting learning objects through related topics. If no targets are selected, the system tries to connect the highest ranking search results.

The process flow for the Path Generation Service is given in Figure 5.

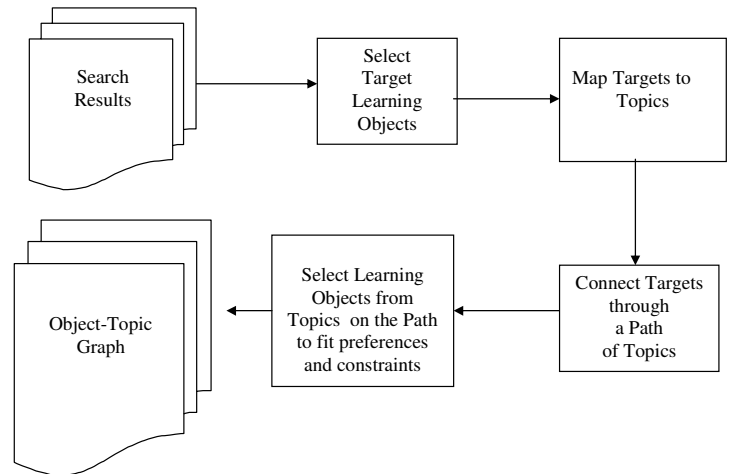


Figure 5: Process flow for the Path Generation Service

The Path Generation Service calls the Metadata Manager Service to map each Learning Object in the search results to topic entities in the Topic Graph using the topic classification in its Learning Object Metadata file. The process of mapping includes computing statistics for each topic based upon the metadata and relevance score of the objects contained within topic categories.

The Path Generation Service attempts to connect related learning objects into coherent Paths using a graph search algorithm. The graph traversal terminates when all target learning objects have been reached or a depth limit has been met. If all the target learning objects are reached, the best path is a minimum spanning tree. However, if some learning objects are beyond the depth limit, the Path Generation Service traverses the graph repeatedly, creating proximal “islands” around the topics containing the selected targets. The Path Generation Service evaluates otherwise equivalent paths using a measure of coherence. The most coherent path has the fewest “breaks” with few relevant objects mapped to topics on the paths.

Given the best path, the Path Generation Service selects learning objects mapped to topics on the path based upon parameters provided in the query and on the statistics collected for each topic by the mapping step. An overview method selects the most relevant objects from each topic on the best path. An indepth method chooses the most relevant learning objects from a given topic before including objects from other topics. The choice of method is controlled by a search scope parameter that is part of

the query. Learning objects are added to the output Object-Topic Graph as long as the cumulative duration of these learning objects does not exceed the maximum desired duration

The Sequencing Service orders the topics on the Object-Topic Path according to the Topic Sequence. This enables custom courses to be generated that put more basic information first. For example, it may be helpful to learn some aspects of J2EE before attempting to learn aspects of the WebSphere product utilizing J2EE.

Next, the Sequencing Service sorts learning objects within each topic on in the Object-Topic Graph. Objects are sorted by the position of the instructional role of each object in the Instructional role Sequence (e.g., introduction, concepts procedures, and conclusion). This provides a logical instructional sequence within each topic.

We have identified an Instructional role Sequence for information technology topics that has worked well for our pilot data. We derived this Instructional role Sequence by looking at the most common patterns how-to materials and tutorials. The consistency of the instructional roles provides structure to the learning experience,

The Sequencing Service calls the Course Manager Service to add each learning object in order to the learner's individual "Custom Course". Custom courses are usually between 1 and 10 learning objects with each learning object taking 5 to 20 minutes.

The combination of the focus provided by the user's query, the coherence provided by path connecting learning objects through the topic graph, the selection of additional objects to improve topic transitions according to the user's available time, and the consistent ordering of instructional roles within topics results in focused, coherent, learning paths without requiring manual linking between individual learning objects. In a controlled study, users who constructed their own custom courses using our system performed at a significantly higher level on a design task than other users with similar prior knowledge who used a search engine alone.

## 2.5 Implementation

The system is implemented entirely in Java and runs on web application servers compatible with the Servlet 2.0 specification. Web pages are created using JSPs and Learning Object content in XML is transformed to XHTML using the XSLT style sheet processor. We have used the IBM XML search engine Juru developed at the IBM Haifa Lab [12] to index both the Learning Object content and the Learning Object Metadata.

A team of over 27 subject matter experts and instructional designers used our content development tools to extract over 500 learning objects from IBM Redbooks, essentially how-to books on IBM products [14]. Books are transformed into DocBook XML [16] to use a common set of elements to indicate hierarchical structure. The documents are then cleaved into chapters, sections, or subsections and processed to create values for some of the basic metadata elements, such as the Identifier, Title, and Typical Learning Time as required by the dynamic assemble engine. All Learning object Metadata is stored using the XML binding for the LOM standard, submitted by our IBM team to the IEEE

LTSC (Learning Technology Standards Committee) [18]. Java objects are created for the learning object metadata as needed by the Metadata Manager. Custom Courses are created as Java objects that are then persisted in the IMS Content Packaging manifest format.

Relations between topic entities are declared using the RDF Description element. While not strictly an OWL ontology [17], we utilize OWL's properties about relationships. For example, the property "uses" and its inverse "used-by" are declared using the OWL ObjectProperty and owl:inverseOf elements. In addition, each property has a label from the rdfs namespace [22]. RDF files are parsed into Java objects at startup.

## 3. Evaluation

Our research team ran a pilot study with an early version of the system with 114 users from across IBM for one month[2]. This version of the system sequenced learning objects selected by users into custom courses. 84 users accessed the system and used it for at least an hour. 73 users filled out evaluation forms at the conclusion of the test period. 81% of users answered with a positive (4 or 5 out of 5) rating to the question "What is your overall satisfaction with this method of learning"? 81% reported that this system would enhance their knowledge/skills. 52% said they would prefer this method of learning over others. 90% found creating and navigating custom courses easy to do. During the pilot, users provided 12 positive (e.g., "a great time saver and productivity boost") and 2 negative comments (e.g., "I found it very difficult to use the query method to build my unique course."). A log analysis revealed that in 69 instances, 37 unique users selected, assembled, and played custom courses. We are now doing an experimental comparison between the Dynamic Assembly Engine and a search engine alone to see if performance on a design task can be significantly improved. We expect that learners will spend more time learning and less time searching and browsing using our system, thus improving their performance on the task.

The Dynamic Assembly approach requires learning objects that may be combined into courses in many different ways. Validating with the learner population that the many possible sequences of learning objects make sense and are educationally effective is a challenging task. One approach is to collect topic queries from users and then have instructional designers rate assembled courses with a variety of settings for input parameters. Another approach is to compare various custom courses on a topic against a hand-generated course on the same topic.

We tried to minimize the amount of metadata that needed to be entered manually. In our experience, automatically derived metadata (e.g., the duration and description of learning objects) was sufficient for users trying to decide whether to select or skip particular learning objects. Manually entered metadata (e.g., difficulty) was hard for experts to estimate without seeing the other learning objects on similar topics and was prone to disagreement, despite careful instructions by the research team. However, subject matter experts and instructional designers must be involved in their design, development, and maintenance or the topic graph and instructional role sequencing policy as new learning objects are added.

The current system is highly dependent on the search engine. A search engine with higher precision is probably preferable to one with higher recall because extraneous learning objects are less

easily ignored when assembled into custom courses. We have addressed this issue by allowing users to optionally select a relevant set of focus learning objects manually from the search results. Using this method, the custom course engine then recommends additional learning objects to add coherence to the learner's selections.

#### 4. Discussion

Learners spend a great deal of time on the Web searching and browsing for information to "amplify" their intelligence[27][28]. They gather just enough information about a topic to be able to complete a task or carry on a discussion. This use of the Web is ubiquitous and yet has not been supported adequately by existing web-based learning systems.

Web-based courseware is often designed for a general audience and is thus not responsive to individual learners. Courses miss the learner's knowledge gap by spending too much time building from basics or by addressing concepts beyond their immediate need. Our Dynamic Assembly approach holds promise because it offers prerequisite information only if it can connect that information to the learner's topic within their available time. It allows learners to focus their cognitive effort on information they need to know in order to do what they want to do.

A good instructor often provides some type of overview of the material to be learned. Research studies have shown that advanced organizers can help improve comprehension of complex material [8]. Studies of learning from hypertext indicate that a table of contents is helpful to low-knowledge users because the outline text provides a guide to the relationships between linked sections and indicates the overall structure [6]. Our Dynamic Assembly Engine creates a course outline that serves as both an advance organizer and navigation aid. It gives cues to the learner about what needs to be learned (topic), how (instructional role), and in what order.

We found the IEEE Learning Object Metadata alone insufficient for instructional sequencing across diverse materials. The LOM provides a Relations category, but learning object metadata relations in LOM can only be between individual learning objects. We found that relating individual learning objects consumed too much content development time. We required a more flexible scheme that would allow us to relate the metadata vocabulary used within Learning Object Metadata files. This additional semantics is particularly important for extensions to the LOM vocabulary, such as our instructional role vocabulary (stored in the Learning Resource Type element of LOM). While there does exist a proposal for an RDF binding for LOM that recognizes a need for this meta-level of description [23], the RDF binding intends to be an alternate encoding of the LOM information model, not a way to enhance learning object metadata vocabulary with additional semantics.

Other researchers have suggested using semantic web technologies for e-learning (see [25]). However, few practical systems have been built. We found it impractical, for example, for learning object developers to richly link individual learning objects to one another because learning objects were being developed independently across a large number of source materials and topics. Instead, we developed a topic classification and topic relationship graph, collected learning object metadata independently and in parallel, assigned learning objects to topics, and then reviewed the implications across the repository.

While there has been considerable research on adaptive hypermedia, much of this work is based upon generation of learning paths by allowing learner traversal of only certain hyperlinks between related web resources [29]. according to a user model. In De Bra et. al. [30], a user model is used to adaptively control both the links and the contents of a Web page. However, the assumption in much of adaptive hypermedia work is that web resources are already richly linked and the job of the adaptive system is to show or hide various links. In generative approaches, a domain model is often used that is too expensive to build because it requires creating a full ontology or knowledge base. Dynamic Assembly achieves higher levels of scalability than typical generative hypermedia approaches, yet is less expensive to develop. Each learning object is tagged independently and relationships and sequencing are encoded between metadata values instead of between individual resources.

Looking ahead, we would like to investigate additional methods for learners to specify what and how they need to learn. We would like to offer self-assessments on topics related to user queries, so that the Dynamic Assembly Engine can prefer learning objects that build on topics that the user knows well and suggest simpler topics when the user is interested in topics where they have self-assessed low on prerequisite knowledge.

#### 5. Conclusion

People learning naturally are often driven by their interests, anticipated needs, or immediate tasks. We have developed an approach to web-based education and training, "Dynamic Assembly", which empowers learners to generate their own learning paths from modular learning objects as needed.

Dynamic Assembly uses both learning object metadata XML and cross-object relationships expressed in RDF to assemble modular learning objects into coherent, focused, and individualized learning paths. We have implemented the approach in a Dynamic Assembly Engine that has been deployed as part of the Custom Course System and proven useful in a pilot study. This work demonstrates how the Resource Description Framework (RDF) can be used in conjunction with the Learning Object Metadata (LOM) standard to improve information retrieval and organization for web-based education and training applications.

#### 6. ACKNOWLEDGMENTS

We would like to acknowledge the contributions of the Learning Object Framework team at IBM that helped develop the Custom Course System, especially Bill Rubin and Steve Levy. Special thanks also goes to Sam Dooley, now at Integre Technical Publishing, who did the technical work on the LOM XML binding while on the team. Thanks also to Danny Oppenheim for his work on the graphical user interface and Amy Katriel for technical support. David Carmel and Yosi Mass at IBM Haifa provided us with the Juru XML search engine. We would also like to thank Ray O'Donnell and Ernie Fuller for their work on the IBM Redbooks pilot project and Yael Ravin, Laretta Jones, Juerg von Kaenel, Steve Rolando, and Peter Fairweather for their encouragement and support.

#### 7. REFERENCES

- [1] [Wiley] Wiley, D.A. (ed.) The Instructional Use of Learning Objects. Agency for Instructional Technology, Bloomington, IN. 2002. <http://reusability.org/read/>

- [2] [Farrell] Farrell, R. et.al. *Learner-driven Assembly of Web-based Courseware*". Proceedings of E-Learn 2003 (Phoenix AZ, Nov 2003).
- [3] [Rieman] Rieman, J. A Field Study of Exploratory Learning Strategies. *TOCHI* 3(3): 189-218 (1996)
- [4] [Carroll] Carroll, J.M. *The Nurnberg Funnel: designing minimalist instruction for practical computer skill*. The MIT Press, Cambridge, Massachusetts, 1990.
- [5] [Charney] Charney, D. The Impact of Hypertext on Processes of Reading and Writing. In Literacy and Computers Hilligoss and Selfe (Eds.) New York: Modern Language Association, 1994. pp. 238-263.
- [6] [Foltz] Foltz, P.W. Comprehension, coherence, and strategies in hypertext and linear text. In J. Levonnen, J.F. Rouet, A. Dillon & R. Spiro (Eds.) *Hypertext and Cognition*. Lawrence Erlbaum, Mahwah, NJ. 1996.
- [7] [Hammond] Hammond, N., & Allinson, L.. Extending hypertext for learning: An investigation of access and guidance tools. In *People and Computers V*, Nottingham, UK.1989.
- [8] [Mayer] Mayer, R.E. (1979). Can advance organizers influence meaningful learning? *Review of Educational Research*, 49, 371-383.
- [9] Merrill, M.D. Component Display Theory. In C. Reigeluth (ed.), *Instructional Design Theories and Models*. Hillsdale, NJ: Erlbaum Associates, 1983.
- [10] [Bloom] Bloom, B.S. (ed.) *Taxonomy of Educational Objectives*. The Classification of Educational Goals. Handbook I, Cognitive Domain. New York: Longman. 1956.
- [11] [Brookfield] Brookfield, S., D. (1986) *Understanding and Facilitating Adult Learning*. San Francisco: Jossey-Bass.
- [12][Carmel] Carmel, D., Maarek, Y.S., Mass, Y., Efraty, N., and Landau, G.M. (2002), An Extension of the Vector Space Model for Querying XML Documents via XML Fragment, *The 25<sup>th</sup> Annual ACM Special Interest Group in Information Retrieval (SIGIR) Conference*, Tempere, Finland.
- [13]Farrell, R., et.al. *Implementing and Extending Learning Object Metadata For Learning-directed Assembly of Computer-based Training*. In "Learning Technology Newsletter" of the IEEE Computer Society, Learning Technology Task Force (LTTF), January, 2003. ISSN 1438-0625, [http://lttf.ieee.org/learn\\_tech/issues/january2003/#5](http://lttf.ieee.org/learn_tech/issues/january2003/#5)
- [14]IBM Redbooks Web site, <http://www.redbooks.ibm.com>.
- [15]IBM Websphere software platform, <http://www.ibm.com/websphere>.
- [16]DocBook XML <http://www.oasis-open.org/docbook/xml/>
- [17]OWL Web Ontology Language overview, <http://www.w3.org/TR/2003/CR-owl-features-20030818/>.
- [18]IEEE P1484.12.3/D1, 2002-12-15 Draft Standard for XML Binding for Learning Object Metadata Data Model, <http://ltsc.ieee.org/wg12/>.
- [19][LOM] IEEE 1484.12.1-2002 Learning Object v1 Metadata Final Draft, [http://ltsc.ieee.org/doc/wg12/LOM\\_1484\\_12\\_1\\_v1\\_Final\\_Draft.pdf](http://ltsc.ieee.org/doc/wg12/LOM_1484_12_1_v1_Final_Draft.pdf)
- [20][IMS] IMS Content Packaging specification v1.1.3 final <http://www.imsglobal.org/content/packaging/index.cfm>
- [21][RDF] Resource Description Framework (RDF) Model and Syntax Specification. <http://www.w3.org/TR/1999/REC-rdf-syntax-19990222>, W3C Recommendation. February 22, 1999.
- [22][RDFS] RDF Vocabulary Description Language: RDF Schema 1.0, <http://www.w3.org/TR/rdf-schema/>
- [23][RDFLOM] RDF binding of LOM Metadata. <http://kmr.nada.kth.se/el/ims/metadata.html>, IEEE draft proposal, August 26, 2002.
- [24][Dumais] Dumais, S.T., Cutrell, E. and Chen, H. (2001) Bringing order to the web: Optimizing search by showing results in context. In *Proceedings of CHI'01, Human Factors in Computing Systems*, April 2001, pp. 277-283.
- [25][Stoj] Stojanovic, L. , Staab, S., and Studer, R. Elearning based on the Semantic Web. World Conference on the WWW and Internet (WebNet), Orlando, Florida, USA, 2001.
- [26].Goh, S. Bressan, S. Madnick, M. Siegel, Context interchange: representing and reasoning about data semantics in heterogeneous systems, Sloan School Working Paper #3928, Sloan School of Management, MIT, 50 Memorial Drive, Cambridge, MA 02139, Oct. 1996.
- [27]Andersson, M. Person Plus Web – Samples From Everyday Life. Word Conference on e-Learning (e-Learn), Phoenix, AZ., 2003.
- [28]Bruner, J., *Actual Minds, Possible Words* (Cambridge, MA: Harvard University Press), 1986.
- [29]Brusilovsky, P., Kobsa, A., and Vassileva, J. (eds.) (1998) *Adaptive Hypertext and Hypermedia*.
- [30]De Bra, P., Aerts, A., Berden, B., De Lange, B., Rousseau, B., Santic, T., Smits, D., Stash, N., [AHA! The Adaptive Hypermedia Architecture](http://www.aha.acm.org/). Proceedings of the ACM Hypertext Conference, Nottingham, UK, August 2003.

