2003

TuLiP, a Teacher's Tool for Lesson Planning

R. Gabrielle Reed

THE FLORIDA STATE UNIVERSITY

COLLEGE OF ARTS AND SCIENCES

TuLiP, A TEACHER'S TOOL FOR LESSON PLANNING

By

R. GABRIELLE REED

A thesis submitted to the
Department of Computer Science
in partial fulfillment of the
requirements for the degree of
Master of Science

Degree Awarded:
Spring 2003

The members of the committee approve the thesis of R. Gabrielle Reed defended on December 17, 2002.

_____
Lois Wright Hawkes
Professor Directing Thesis

_____
R. C. Lacher
Committee Member

_____
Ian Douglas
Committee Member

Approved:

_____
Sughir Aggarwal, Chair, Department of Computer Science

The Office of Graduate Studies has verified the above named committee members.

TABLE OF CONTENTS

LIST OF TABLES V

LIST OF FIGURES VI

ABSTRACT VII

1 INTRODUCTION 1
1.1 Research Objective 1
1.2 Thesis Outline 3

2 WEB TECHNOLOGY 6
2.1 E-learning and E-commerce 6
2.2 XML and Rapid-Development Frameworks 7

3 TEACHER'S CHALLENGE 10
3.1 Laws Affecting Teacher's Workload 10
3.2 Hurdles to Technology 11
3.3 The Lesson Planning Process 15
3.4 Current Lesson Planning Tools and Internet Resources 17
3.5 Drawbacks of Existing Tools 27

4 MEETING THE CHALLENGE – TULIP 31
4.1 Proposed Lesson Planning Tool's Design 31
4.2 Teacher-Centered Interface 32
4.3 Characteristics 33
4.4 Learning Objects 35
4.5 Fundamental Learning Objects (FLO) and Knowledge Type Templates (KTT) 36
4.6 Learning Environment and Lesson Planning (LEAP) Markup Language 40
4.7 A Rapid-Development Web-Site Platform (Cocoon 2) 49
4.8 Tool Components 63
4.9 TuLiP Development Plan 64

5 TULIP OVERVIEW AND CONCLUSIONS 66
5.1 Benefits of Using TuLiP 66
5.2 Future Work 70

APPENDIX A GLOSSARY OF TERMS 72

APPENDIX B SYSTEM DESIGN DOCUMENT FOR TULIP 74

REFERENCES 78

BIOGRAPHICAL SKETCH 88

LIST OF TABLES

	Page
2.1: Current Web Technology Capabilities Applied by the Proposed TuLiP Tool for Educational Uses.	9
3.1: Selected Perceived Barriers to Teachers to the Use of Technology for Instruction in the Classroom	12
3.2: An Example of the Steps Involved in the Lesson Planning Process.	16
3.3: The Sequence Used by STEPS in the Development of a Lesson Plan	20
3.4: The Instructional Architect's Five-Stage Process Using Learning Objects in Developing an On-line Learning Environment.	27
3.5: Aspects of Commonly Used Lesson Planning Tools	30
4.1: Fundamental Learning Object Objective by Description of Typical Content and Functionality.	39
4.2: A Summary of the Review of Languages Used to Create Instructional Materials, their Strengths and their Drawbacks and the Proposed Solution in LEAP.	47
4.3: Demonstration of Reuse of Parts of a Simplified Lesson Plan for the Generation of Additional Products Designed for Different Audiences.	60
4.4: The URI Naming Scheme for Lesson Plans and Products for Different Target Audiences.	61

LIST OF FIGURES

Page

3.1: Example of the Ohio SchoolNet Lesson Planning Template Form in MS Office. Source http://tlcf.osn.state.oh.us/blueprint/index.html 19

4.1: Illustration of a Simplified TuLiP Tool Interface with Resources for Lesson Planning. 33

4.2: Diagram of the Passau TeachWare Model, Used with the Learning Material Markup Language (LMML) by Christian Süβ. 45

4.3: Cocoon Architecture. Source: xml.apache.org/cocoon 50

4.4: A Code Excerpt from the sitemap.xmap Used by Cocoon to Initiate the Types of Services Needed During an URI Request. Source: xml.apache.org/cocoon 52

4.5: UML Sequence of Events Diagram for Processing a Cocoon HTTP Request. Source: xml.apache.org/cocoon 53

4.6: Demonstration of Statements to Incorporate Logic and Functionality as Separate Elements within a Logicsheet Using XSL 55

4.7: A Cocoon Pipeline. Source: xml.apache.org/Cocoon 56

4.8: The Cocoon Pipeline Process with Components, XML and XSL style sheets using an Aggregator. Source: xml.apache.org/cocoon 58

4.9: Excerpt From TuLiP sitemap.xmap Demonstrating the Use of the wildcard *. 62

5.1: A Conceptual Web Design Diagram of the TuLiP Tool, Fundamental Learning Object Repository and the Products Generated Using the Cocoon2 Framework. 67

ABSTRACT

Teachers are expected to plan the daily learning environment in the classroom, and incorporate technology effectively in the curriculum and instruction. They need to provide information to parents and educational materials to students in accessible and alternative formats. These are requirements of the Federal "Leave No Child Behind" law [PL 107-110, 2002] and the "Individuals with Disabilities Education Act Amendments of 1997" [PL Public Law 105-17, 1997]. Teachers are limited in reaching these goals by the scarcity of easy-to-use tools and resources to meet these challenges.

These problems are addressed by the proposed Teacher's Lesson Planning Tool (TuLiP), a tool that is as simple as a form, but harnesses the power of XML and Java Servlet technology within a Cocoon2 dynamic web-publishing framework. The web-based framework allows widespread access to resources regardless of the teacher's operating system; it also allows for the generation of lesson plans in a variety of formats. It enables access to information by administrators, parents and students, and automates the production of alternative and diverse on-line materials, all from the same content.

The flexibility for reuse and sharing is enabled by the use of the designed XML-based semantic Learning Environment and Planning (LEAP) language. By using LEAP markup, the content is categorized into classes of Fundamental Learning Objects (FLO), arranged and marked according to its instructional use, with Knowledge Type Templates

(KTT). This structure allows storage in a repository (a metadata library), and retrieval using searches on sharable and reusable educational content.

CHAPTER 1

INTRODUCTION

1.1 Research Objective

In 1997, the Presidential Panel of Advisors in Math and Science Education gave a mandate to those in the field of computer science research to develop tools and applications to support the field of education. This is reiterated by President Clinton in the Department of Education call for tools to meet the “National Technology Plan":

"Goal 1: All students and teachers will have access to information technology in their classrooms, schools, communities and homes.

Goal 2: All teachers will use technology effectively to help students achieve high academic standards.

Goal 3: All students will have technology and information literacy skills.

Goal 4: Research and evaluation will improve the next generation of technology applications for teaching and learning.

Goal 5: Digital content and networked applications will transform teaching and learning."

> "e-learning - Putting a World Class Education at the Fingertips of All Children" , 2000, US. Department of Education

In order to comply with Goals 3, 4 and 5, this research encompasses a proposed design and prototype of TuLiP, a teacher's rapid-design Lesson Planning Tool. Lesson planning is the second most time-consuming task of a teacher, after that of actual classroom teaching. Lesson planning is an integral part of teaching, allowing the teacher to review teaching materials and tools and organize the important aspects of the student contact time. This planning is a requirement of all K-12 teachers in that it demonstrates intended coverage of the curriculum, as required by federal, state and district rules. The planning has become critical in preparing materials to integrate technology in the curriculum. This research has discovered that computer-based lesson planners and the Internet are used by only a small percentage of teachers [NCES, 2002]. The most commonly sold lesson planner is a bound paper calendar with supplemental lesson plan forms. A TuLiP lesson planning tool as presented in this paper is designed to be a time saving tool to use the information typically stored in a lesson plan. It allows reuse of the information in a variety of products, not only to produce the required lesson plans, but also to facilitate the use of technology in the classroom, by providing information through the Internet. It does this by storing information once, and automates its display as needed. It allows sharing and reuse of components, files, and templates, and above all, reusable information. This type of tool is needed in light of the rising costs of educational materials, and the additional requirements added to the teacher's existing workload due to the Federal "Leave No Child Behind" law [PL 107-110, 2002] and the "Individuals with Disabilities Education Act Amendments of 1997" [PL Public Law 105-17, 1997]. TuLip is designed to be an easy-to-use tool that provides resources to meet these challenges.

This research confirms a deficiency of computer-based lesson planning applications. The TuLiP tool is a web-based solution proposed by this paper. It uses the same web technology that is beneficial in e-commerce. TuLiP is a web-based rapid-design lesson planning tool. The full design will use a proposed Learning Environments And Planning markup language (LEAP), a compliant Extensible Markup Language (XML). LEAP is designed for the semantic storage of the lesson planning and learning environment content in Fundamental Learning Object (FLO) modules. The FLOs as proposed are complete and functionally independent orthogonal modules based on instructional tasks. The plans use an aggregation of modules. Extensible Scripting Language (XSL) is used for the logic and display of the content. This combination of LEAP and XSL automates the generation of an assortment of supplemental educational materials called products, based on the lesson planning content. The selected framework for the design also allows the selective layering of FLO modules for additional functionality.

The TuLiP lesson planning tool, using the Cocoon 2 framework with an appropriate markup language (LEAP) allows for rapid development of instructional materials. The proposed TuLiP tool is designed to save time by providing a proposed web-based teacher centered interface using the semantics of instruction that minimizes the technological burden of the tool. Timesaving may be found with the use of instructional templates, and reuse of created FLO modules.

1.2 Thesis Outline

This thesis provides the design if an instructional system using free Internet resources already available to teachers. Why are the statistics on the use of technology so low for teachers? From a recent article “Examining 25 years of Technology in U.S. Ed” by

Norris, Soloway and Sullivan, a number of their samples provide interesting results: 42% of the teachers surveyed report that the students uses computers less than 15 minutes per week and 65% report that the students uses the Internet less than 15 minutes per week. To assess what is available for teachers to use, a review of the educational resources on the World Wide Web (Web) was performed. Resources on the Web may make a teacher's job easier. From the sample of tools and instructional content available on the Internet that were used for this thesis, it was quickly realized how many problems there were facing teachers in integrating technology into their classrooms. Instructional content "designed" for re-use was problematic. Many of the instructional design tools were basically tools for software developers. Even HTML editors have many options that can go awry under novice control.

Many of the reviewed e-learning designs were outmoded according to the latest research in education technology or the psychology of learning [Greenagel, 2002]. They reside within the domain of instructivist learning, i.e., "We provide information to students to learn." Because this is an appropriate method for many tasks and processes, it succeeds in many domains. However, if the goal is to produce thinking students, is this now the best way to do it?

In Chapter 2, some of the benefits of e-commerce are proposed for use in e-learning. The power of e-commerce comes from the use of powerful XML and rapid-development web frameworks and services available today.

Chapter 3 looks at the challenges encountered by teachers in their attempt to meet the demands of President Clinton's "National Technology Plan" for Education. This chapter discusses some of the hurdles and problems with existing tools.

Chapter 4 proposes the use of TuLiP, a proposed web-based tool for lesson planning, to open the door to the use of technology and provide some much-needed assistance with distribution of routine information. This chapter also covers the characteristics and components in a full implementation of the Tulip Tool.

Chapter 5 summarizes the benefits of using TuLiP and provides a description of future work needed for this tool.

CHAPTER 2

WEB TECHNOLOGY

2.1 E-learning and E-commerce

There is a wide range of resources ready for purchase and many freely available on the Internet. The resources range from single items, such as images, to fully designed courses in an instructional management system. A number of groups have created Learning Objects (LO) which is a label used for sharable educational materials. Some used structured content, as a rapid development technique for courseware such as Cisco Systems development of their Reusable Learning Objects (RLOs) to train their workforce [Cisco, 2000]. Some use structured data to describe learning objects, to be able to retrieve and share commonly used and taught material such as the Dept of Defense's Advanced Distance Learning (ADL) Initiative Sharable Content Object Reference Model (SCORM) [SCORM, 2001]. Proprietary and open source XML web-development frameworks enable the content to be separate from the style and logic, referred to as the *Separation of Concerns*, allow for easy manipulation of the presentation on WWW pages. Examples of these are Microsoft's DotNet [Microsoft, 2002] and the Apache projects Avalon, and Cocoon [Apache, 2002]. A tool design should take advantage of metadata and separation of Concerns.

Frank L. Greenagel in his recent article “The Illusion of E-learning: Why We Are Missing Out on the Promise of IP Technology” stated that e-learning has not kept pace

with the development of increasingly rich Internet Protocol (IP) based delivery platforms. He states a number of problems with e-learning:

- "Developers don't seem to be aware of how people learn, for they continue to use mostly flawed models.
- … solid measures of effectiveness are infrequently developed or applied.
- The available platform drives the instructional strategy, which may not be appropriate to the learning style of trainees or to the learning objectives.
- … the absence of any commitment to measure effectiveness.
- Effective e-learning experiences are rarely scalable." [Greenagel, 2002]

"Greenagel's complaint that developers aren't necessarily good teachers may indeed explain the existence of some of the less imaginative courses offered. At the very least, it demonstrates that factors like the appropriateness of some learning models to a limited range of competencies and, again, up-front costs have have won out over effectiveness. This is not entirely surprising for a relatively young domain, but it also ought to be a reminder of where we need to be heading next", according to Wilbert Kraan, of the Centre for Educational Technology Interoperability Standards **(**CETIS) [Kraan, 2002]. The TuLiP tool, in the hands of teachers, would be able to correct some of these highlighted problems in e-learning.

2.2 XML and Rapid-Development Frameworks

There have been extensive changes in the way we do business based on the current web technology. This has been fueled by the development of XML and associated Web applications and services.

XML allows semantic content storage and retrieval. This allows a company with a web site, who previously had to hard-code each page of product information, to now use

a template with semantic element tags, with different pages containing information unique to that product. From the same page of information, a catalog entry, a short product description, or more detailed information can be generated dynamically.

XSL allows the presentation of content in a wide variety of output formats. Just a few years ago, the idea of updating the look of a website entailed re-coding each page with the new look tags. The use of style sheets made this process go faster. The particular style sheet can now be edited. XSL takes that one step farther, in that it allows for the style and logic to be designed by semantic element, no matter where it appears.

Portal Technology allows relevant storage, retrieval and community services within a web-based environment. The major producers of software and hardware have sites with the latest downloads, FAQ, and technical support contacts. "Market leaders such as Amazon and America On Line have built integrated models capable of satisfying every user that comes along." according to a report by Datacomm Research [Datacomm, 1999].

Web frameworks such as DotNet and Cocoon 2 allow for the rapid development of web environments that use server technology and combine the use of a variety of web development languages. Cocoon 2 uses XML and has a number of characteristics that will enable the development of an e-learning environment from lesson planning content. A number of other XML parsing and transformation tools are available. Many require the XML source and the translation file to be specifically designated in each web page. Logic and display information is necessarily combined within the XSL file for the content to be transformed in one pass. Web architectures such as Cocoon 2 allow for the separation of style and logic into multiple files, based on functionality. This is one of the strong points of using Cocoon2. Educational content can be stored in a form that is both independent of

how it appears on a webpage and independent of its functionality. Besides rapid development and reuse, Cocoon2 also allows selective use of files. The Cocoon servlet accepts Uniform Resource Identifier (URI) requests and through pattern matching, determines the instructional content, as well as the display and logic XSL files to be used in the transformation. It also has the ability to process URI's that match specific patterns using wildcards, to determine dynamically the source and translation files to be used. This allows the generation of semantic-based instructional content from templates and access to the published web page immediately after uploading to the server.

In summary, the capability of the rapid development architectures that may be used for education is described in Table 2.1. This gives us a powerful architecture to produce a teacher-friendly interface where the teacher deals mostly with educational and not computer terminology.

Table 2.1: Current Web Technology Capabilities Applied by the Proposed TuLiP Tool for Educational Uses.

1. Dynamic selection of files, files formats, logic sheets through a web interface using URI to control the display of content to meet the needs of different audiences,
2. Storage of information by descriptive metadata making it searchable and reusable,
3. Storage of content stored in an XML language based on the needs of the teacher,
4. Web access to resources, databases and files through a web portal, and
5. Web Forms and Services for easy upload to the server.

CHAPTER 3

TEACHER'S CHALLENGE

3.1 Laws Affecting Teacher's Workload

The need for timesaving tools is obvious when you consider the diverse responsibilities of teachers because of current federal laws and mandates:

- "Leave No Child Behind Act" [PL 107-110, 2002],
- "National Education Technology Plan" [e-Learning, 2000], and
- "Individuals with Disabilities Education Act" [IDEA, PL 105-17, 1997].

These following responsibilities are added to existing workload due to the implementation of the above laws:

- integrating technology in the classroom,
- providing accessible information to parents of disadvantaged individuals,
- using scientifically based teaching techniques, and
- accommodating disabilities and student diversity.

These are added to some of the teacher's existing responsibilities:

- writing and submitting lesson plans,
- teaching core curriculum,
- grading papers,
- supervising halls and classrooms,

- assessing disabilities,
- keeping abreast of new teaching strategies, and
- encouraging parental participation.

The full extent of actual responsibilities is typically outlined in the teacher's job description on the district level.

3.2 Hurdles to Technology

From a number of studies on teachers' use of the Internet, it is apparent that few are using this resource extensively. Surveys of the National Center for Education Statistics (NCES) indicate that even though 98 % of the schools are connected to the Internet with computers in essentially all classrooms, less than 10 % of the teachers use the Web a lot for their lesson planning or teaching [NCES 2001-071, NCES 2000-090]. More extensive studies that look at the actual activities associated with teaching with computer or the Internet, and the frequency of use has been proposed by the NCES, but the results are not available at this time. A survey was performed of 400 core curriculum teachers by Quality Educational Data (QED) of Scholastic Inc., to determine the uses of the Internet by teachers. Eighty (80) percent have used the Internet to evaluate curriculum materials. Fifty (50) percent have used it as a presentation tool and for planning. Eighteen (18) percent said that the Internet has not had an impact on their teaching [QED, 2000]. So they may have tried to use it once? But what is the actual usage? From the samples from Norris' August 2002 article, 65% of the teachers report that students use the Internet for less than 15 minutes a week. At least, the teacher had to use the Internet to prepare material to be used. These numbers indicate low usage of computers and Internet.

What do teachers say are the reasons for such low use? Table 3.1 is a combined list of the highest-rated barriers to the use of resources, from the 2000 report "Teacher's Tools for the 21St Century: A Report on Teacher's Use of Technology" [NCES, Pub 2000-102], and some of the hurdles to teachers in integrating technology in the classroom from the 1998 NASA study on web-based instruction and learning [Grabowski et al, 1998]. Lack of time to learn and lack of training appeared in both reports.

Table 3.1: Selected Perceived Barriers to Teachers to the Use of Technology for Instruction in the Classroom

1. Not enough reliable computers
2. Lack of time to learn and use the web resources and the technology tools
3. Lack of training in technology literacy and research skills
4. Lack of time for students to use the computer
5. Lack of easy-to-use tools to integrate the WWW in to the classroom

The use of training, good tools and prepared instructional materials incorporated during the routine planning tasks are recommendations that may facilitate the integration of WWW based resources in the classroom.

In order to eliminate some of the barriers to teachers, the U.S. Federal government has implemented a number of programs. The Technology Literacy Challenge Fund (TLCF) provides more computers to disadvantaged schools. The "Preparing the Teachers of

Tomorrow to use Technology" (PT3) program supports the development of continuing education and training for future teachers. The National Educational Technology Standards (NETS) encourage the use of the Internet in the classroom to increase the use of computers by the students.

The federal government has set up the TLCF to help school districts to purchase technology, to meet their most important educational technology needs, specifically in their disadvantaged schools [LinktoLearn, 2000].

The government has implemented the PT3 program, to provide grants for implementing technology training in teacher training programs. The National Educational Technology Standards (NETS) is described in "e-Learning: Putting a World-Class Education at the Fingertips of All Students" [US DoEd, 2000]. Exemplary teacher-education programs that have incorporated NETS are showcased on the www.PT3.org web site, along with the list of many universities that have been grant recipients. Very little information was found in the education programs about using technology for lesson planning.

An example of training under the PT3 program is the University of Texas at Austin, UTeach Program. They teach the use of office products, multimedia and web authoring tools to produce an HTML teacher portfolio.

Although educational research has many articles pointing out the drawbacks of hypermedia, teaching how to code in HTML is a common approach taken for teaching the preparation of on-line instructional materials at many sites sampled. Some of the HTML problems are disorientation, cognitive overload and discontinuous flow. Discontinuous flow covers narrative flow and conceptual flow, where narrative flow

refers to the didactic flow of the text itself. Conceptual flow refers to the flow of ideas or concepts [Murray, 2000]. Preparing educational materials in HTML may be more expedient than reusable [Wiley, 2001]. From the course descriptions, it was difficult to determine if the web publishing included learner-centered interface design, instructional and presentational design for the web and accessibility issues. All of these needs should be considered in the generation of instructional materials. However, teaching teachers to be web developers is diverting their time and effort away from teaching. Everything that goes into producing good quality educational materials is not quickly taught. Learner-centered design guidelines need to be provided. The other option is to use available web resources to supplement the teaching, but tools are needed to use these materials.

The last hurdle, the lack of easy-to-use tools, provided the motivation to review the difficulties in use of the resources, to determine a design for a useful tool. A 1998 NASA study, "Web-based Instruction and Learning: Analysis and Need Assessment" found that to meet the technology requirements, some teachers are spending a lot of time learning and creating on-line material using tools designed for programmers and web developers. The best way of incorporating technology into teaching, they said, was to incorporate it as an integral part of lesson planning [Grabowski, McCarthy & Koscalka, 1998].

Along with access to working technology and a minimum amount of computer skills, teachers need an easy-to-use web tool that the teacher can use to do his or her required lesson planning duties. This paper proposes and demonstrates that routine lesson planning contains sufficient educational content of value to teachers, students and parents, to provide useful on-line materials. Technology today allows the presentation and accessibility issues to be incorporated in templates that may be used for planning. The

web-based lesson-planning tool was chosen as a likely tool of choice to overcome this hurdle.

3.3. The Lesson Planning Process

To provide an easy-to-use tool on lesson planning, the lesson planning process was analyzed.

Lesson Planning is used to prepare a teacher to teach a lesson. Plans are typically submitted to an administrator to assure conformance to national, state and district curriculum standards. A typical lesson plan includes the objective for the lesson tied to the curriculum, an assessment plan, a list of the teaching tasks and a list of the resources needed to perform the teaching tasks. Additional items may include lecture notes, questions, assignments, and information on common misconceptions. Teachers add whatever else they feel is needed. Comprehensive plans may include:

1. an actual teacher script to be used during teaching,
2. the exact lecture to be given,
3. questions to be asked of the students, and
4. a selection of prompts, hints and answers based on student performance and responses.

The typical steps in the process of preparing a simple lesson plan are included in Table 3.2. [AskERIC, 2002]

Curriculum refers to an adopted set of competency descriptions by subject and grade. The list of teaching tasks provides the sequencing of the presentation of the material to the students. Manipulatives are physical objects used as representations of abstract concepts used by the students to facilitate understanding.

Table 3.2: An Example of the Steps Involved in the Lesson Planning Process.

1. Determine the goal of the lesson
2. Determine the objective of the lesson.
3. Determine that the objective fulfills the curriculum.
4. Develop the Lesson Description: Overall description of the lesson
5. Determine the student profiles.
6. Select teaching methods to match the profiles.
7. Determine any pre-requisites for the lesson.
8. Determine the assessment method based on the objective, the assessment value, content and solutions expected for successful completion of the objective.
9. Determine the materials needed for the lesson: List the resources for the lesson, including equipment and instructions for use, instructional media and manipulatives.
10. Determine the lesson procedure: List the teaching tasks to be performed.
11. Outline lesson content including lecture notes.
12. Develop questions to facilitate cognitive processing of the materials.
13. Address student's most frequent hurdles, with help suggestions, and clarify information for the most common misconceptions.
14. Address the most difficult concepts for students to understand by providing examples, problems and solutions.
15. Identify location and manner for control of each part of the lesson.
16. Determine assignment content and solutions.

The planning document must contain sufficient information to demonstrate coverage of topics in the district, state and national curricula. It needs to be detailed enough that a

substitute teacher could accomplish the lesson for the day, when needed. Lesson Planning is taught in accredited teacher education programs.

> "The primary obligation of the teacher's art is the representation of the subject matter in ways that can be readily learned and understood. TEAC requires evidence that the candidates have learned how to convert their knowledge of a subject matter into compelling lessons that meet the needs of a wide range of pupils and students." [TEAC, 2001-2002]

Some state agencies, school districts and educational institutions have specific recommendations or requirements on the content of a plan.

3.4. Current Lesson Planning Tools and Internet Resources

When designing a useful planning tool, the review of similar tools and their products is needed. The next step is to analyze these for their strengths and weaknesses. Although there are many tools and numerous websites that store lesson plans and learning materials [ERIC, 2002], the time needed for teachers to integrate web resources in their classrooms is in direct competition with their time for the required tasks of lesson planning and teaching. The design of the TuLiP Lesson Planning tool addresses these concerns.

3.4.1 Lesson Planners

The planners can be worksheets or plan books, similar to calendars, with pages to list the activities for each period. Some websites have simple instructions on how to plan. Due to the requirement to implement scientific-based teaching practices, some of the websites and planners come with guidelines to help teachers. A few are based on office

products, such as those used by the Ohio School Net and the Florida Dept. of Education. Some are on-line. This next section discusses the different types of lesson planning tools.

The most common planning assistance is in the form of planning guidelines and existing lesson plans. Directories at on-line educational clearinghouses such as Ask Eric (www.askeric.org), or Florida Information Resource Network (www.FIRN.edu) provide a wide range of information.

The second is forms on paper, in computer applications or on-line. Planning books for handwritten plans are available at bookstores in journal or calendar form. Calendar forms are also available on Personal Digital Assistants (PDA). Many different planning paper forms are available through educational supply houses. Two popular planners are "The Plan Book," published by Scholastic Reference, Inc. and "The Teacher's Daybook" by Jim Burke, published by Heinemann, Inc.

Computer Planning forms are available using a variety of the MS Office or Claris Works Products. For example, the on-line Ohio SchoolNet form (as in Figure 3.1) is in MS Word (.doc) format. It can be completed on the computer and saved by the teacher or printed out and completed by hand. The Curriculum Planning Tool, from the Florida Department of Education, produces files in the Claris proprietary form [ICPT, 2002]. Another lesson planning tool is based on the MS Access database, using the forms and report functionality. These are sharable and usable if the teacher has the same version of production software.

There are some well-constructed on-line and stand-alone lesson planning tools that help teachers plan and tie their lesson to the curriculum. The benefits are evident in that they allow reuse, and modification of the lesson plan. What they do not do, however, is

allow the re-use of content of the lesson plan for any other purposes, such as providing on-line material.

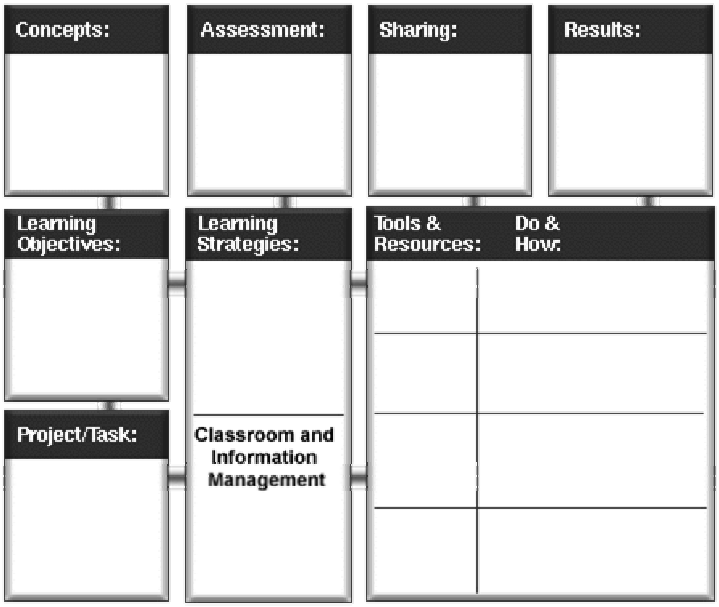

Figure 3.1: Example of the Ohio SchoolNet Lesson Planning Template Form in Microsoft Office. Source http://tlcf.osn.state.oh.us/blueprint/index.html

Examples of on-line lesson planning are the Lesson Planner, at school.discovery.com, and STEPS. STEPS, the Lesson Architect, Version 3.2, is an on-line lesson planning tool, developed by the University of West Florida following the simplified lesson planning process listed in Table 3.3.

Many of the on-line planners have a similar format, with forms for the teacher to complete in order. The Lesson Planner from Discovery.com does allow the teacher to add additional items as a first step. Teachers can save it to an account on the server. It may also be printed out for the teacher to use while teaching.

Table 3.3: The Sequence Used by STEPS in the Development of a Lesson Plan
START: Unit Title and Purpose STEP 1: Lesson Title and Purpose STEP 2: Standards and Benchmarks STEP 3: Objectives STEP 4: Assessment STEP 5: Introduction STEP 6: Lesson Activities STEP 7: Summary STEP 8: Resources

The on-line lesson-planners are supplemented by a search engine that allows the teacher to search for appropriate lesson plans. Lesson plans at these sites are typically a list of activities and are not a fully developed lesson. A teacher could use the ideas as a starting point, but he or she would have to spend additional time to complete the plan in a different application. The on-line tools do not allow the editing of existing plans. They allow for the sharing of plans from a repository on the web, but they do not produce anything more than a printable plan for the teacher to use in the classroom.

Some sites have a suite of teacher tools, one of which may be a lesson planner. At school.discovery.com, they also have a quiz generator developed at the University of

Hawaii that allows for the creation and administration of tests on-line in a number of different formats, such as true/false or multiple choice.

An interesting planner is MIMIC (Multiple Intelligent Mentors Instructing Collaboratively). MIMIC uses multiple Microsoft interface agents in a collaborative process to facilitate the planning, by providing suggestions when requested based on the agent and their incorporated theoretical teaching philosophy. It uses an Artist agent to reflect more creative aspects, an Instructional System Design Agent to reflect problem solving aspects [Dick , and Carey, 1996], Gagne's events of instruction [Daniels, 2002] and an Alternative Agent using constructivist, modern instructional and semiotic theories, cognitive flexibility, Vygotskian and situated cognition [Baylor, 1999]. This last planner demonstrates the type of flexibility needed in instructional design to teach using the different strategies and methods needed for diverse student profiles.

Lesson planners are available for the Personal Digital Assistants (PDA). PDAs are examples of minimal interfaces that may be beneficial in software applications in general. The interface for the planner is a calendar with a text file for each day.

Although there are many lesson planners available from the review, very few have any functionality to generate on-line material in different formats.

3.4.2 Internet Resources

The Internet resources that are available to teachers to supplement a lesson plan include individual multimedia objects, stand-alone applications, applets, web-sites and learning objects. Some of these are web-based and can be used by linking to the site. Others require the use of a proprietary instructional management system.

There are a number of sites that allow teachers to download software for free. Due to the rampant computer virus problems, teachers typically will limit what they use by making sure it comes from a reputable site. From the sample reviewed, it was found that there are many multipurpose applications. With these, the teacher has to take the time to become familiar with the software and then teach the students how to use the software, before the students can use it to complete a task. Simpler interfaces and the limit of scope increase the chances of an application being used.

Applets have a better chance of being used due to their limited purpose. An example of this is "The Analytical Engine On-Line" a website that supplements a textbook [Thompson, 1998]. This provides an activity with a simple interface with instructions on the page. There are a number of applet download sites.

Websites, such as "MarcoPolo" [Worldcom, 2002] provide a wide variety of rich multimedia and extensive support information. These pages exhibit many of the characteristic problems associated with HTML. A student needs a scaffold to know the goal of the exercise, where to go and what to look at or they quickly get lost and distracted from the learning task at hand. These pages are not easily copied and as a result may have a limited lifespan. Sites like these, incorporated into plans constructed today, may not be available tomorrow. WebQuest at the University of California at San Diego limits the activity to a very limited objective focus [WebQuest, 2002]. This helps the students to stay on track. It has a simple interface, which may not meet accessibility standards.

Many single multimedia objects may be used to supplement the lesson plan such as media clips and images. These are rarely accompanied by supporting information in an

easily reusable format. The European Space Agency's caption for each image in PDF is a good example of how information on the images may be bundled for reuse [European Space Agency, 2002].

Learning Objects are considered very promising in providing reusable materials for teachers. They fall under the initiatives for reusable, portable, sharable training materials. Stephen Downes in his review of learning objects demonstrated that there would be savings in terms of time and cost by using the shared common materials [Downes, 2002]. “Learning objects have much potential in making online learning development affordable. For learning objects to be truly effective, pools of RLOs of sufficient size (and quality) are needed (i.e. institutional, national, and international)” [elearnspace.org, 2002]. LO's have been effective in reducing the cost and improving the quality of education [ADL, 2000]. These range in size and content from full courseware to individual objects such as images or video clips [Wiley, 2001]. They typically are developed and produced by instructional designers for use by teachers.

The Learning Objects, however, have varied scope, having been designed for a particular audience. They contain specific teaching methods, presentation style and navigation which can affect its usability. There is a need for commonly used educational content that is devoid of presentation style and instructional strategy [Downes, 2001].

Over the past few years, the number of repositories of learning objects has been increasing in number and they have demonstrated their usefulness [Bruckman,2002].

Learning Objects may be found most easily in libraries/ repositories/ portals/ on-line communities, similar to those operated by the following:

1. The Science, Mathematics, Engineering and Technology Education (SMETE) Open Federation (www.smete.org),
2. The National Science Digital Library (NSDL) (comm.nsdlib.org/),
3. The Educational Object Economy (EOE) Foundation (www.eoe.org),
4. "Latis on-Line" (LOL) (www.latis.net.au/ols/cfptool.htm)

SMETE resources can be located by using a directory to each of the collections, or by using a search facility with keywords stored as metadata about the resource. The information indicates if the resource is a stand-alone application (with operating system information) or on-line content. There is an option for user reviews. The website allows storage on the site for the selected resources in a variety of formats such as PC or Mac applications, on-line web format and Java applets. All of these have their own style, content sequence or navigation design. As an example, some of these learning objects are web-based and contain navigational links, such as links to sponsor's webpages, and presentational style content as well as the educational content. Many of the objects are simulations where the variables can be manipulated. Although these are mostly science and math related, there is a small selection of non-science related materials available, as well.

NSDL is a portal with an on-line digital library based on technology, science, engineering and mathematics.

The EOE website is a community of people who develop and distribute tools to build shared knowledge bases of learning materials. They have a repository of over 2600

different Java Applets for use by educators, sorted by the major divisions in the Dewey Numbering System. Teachers need to link to the resource or know basic html applet tag setup to be able to use these [Spohrer, 1998].

"Latis on-Line" (LOL) has the Curriculum Framework Planning Tool (also known as the e-Tool). This will be an on-line resource that assists teachers in the planning of units of work specific to the Outcomes detailed in the Northern Territory Curriculum Framework. The on-line services include:

- "access to online curriculum framework tools;
- access to electronic student reporting, assessment and tracking mechanisms;
- availability of educational software and associated professional development resources;
- access to classroom resources having sound curriculum value and being digitally housed; and
- the ability for stakeholders to communicate with their peers, locally, nationally and globally" [Latis-onLine, 2002]

Many of the repositories such as EOE and LOL have an on-line community of individuals who can answer questions, as well as a full set of instructions in FAQ (Frequently Asked Questions) pages.

The learning objects may be used as independent activities, but many need to be supplemented with objectives, instruction for use and a summary of the experience when used with children. A web-based tool that uses Learning Objects is the Instructional Architect (IA) [Wiley, 2001]. The tool demonstrates the process of using LOs to supplement education. The IA is an NSF grant-funded project of the Reusability,

Collaboration and Learning Troupe (RCLT) at Utah State University, to develop a tool that uses the existing LOs in the NSF repositories. IA is a simple web-based tool, used by teachers to prepare lessons using the LOs. A search is used to select the LOs based on a catalog of information describing each LO stored as metadata in the repository. It uses repositories such as the Science Mathematics Engineering and Technology Education Content (SMETE) and National Science Digital Library. The characteristics of the IA are:

- simplicity in design;
- access to a repository with search and retrieval capability; and
- use of teacher-centered forms for constructing the learning environment with Learning Objects (LO).

Although some LOs may be used directly, some require additional instruction for students to use it. IA was designed to sequence objects and add supplemental information. The LOs are best preceded by an introduction and instruction to be useful in the classroom. It should explain why the students are doing a particular activity. The IA lets the instructor insert a purpose for the activity, instructions for use, and a concluding statement for each object. IA uses a five (5) stage process in Table 3.4 to generate the instructional content.

When corrections are needed, the teacher is returned to the form in Step 3. Steps 3 and 4 are repeated for each LO that is on a page of HTML.

Table 3.4: The Instructional Architect's Five-Stage Process Using Learning Objects in Developing an On-line Learning Environment.
1. Gather resources. 2. Type in the title and overview on the first page including the list of resources. 3. Add Objects, edit the order in which they appear on the pages, and compose lesson text around the included resources using a web form. 4. Preview the pages. 5. Commit the prepared instruction and receive an URL of its location.

A teacher interested in using Learning Objects would have to find a review of the courseware, or peruse it him/herself. Many titles come with a price tag but with courses given repeatedly, this saves on yearly course development costs. There are course providers listed in Steven Downes' article on "Learning Objects" such as Telecampus at http://courses.telecampus.edu or the Web of Asynchronous Learning Networks (ALN) at http://www.aln.org. The ALN course listings are divided into subject areas, where each subject page contains a list of similar courses offered by different institutions.

3.5 Drawbacks of Existing Tools

From the surveys on the use of technology, most teachers are still doing lesson plans by hand, using a plan book. The teacher lists the activities to be performed and the details for each activity. The simple act of entering the lesson plan on a computer will allow the teacher to reuse or revise it as needed, submit it to administration, or share it with others teaching the same curriculum. Teachers should be able to use the computers and software

currently available. Most computers come with an office suite of programs, but for untrained individuals, these routine office applications can be difficult to use [PT3, 2001]. It may appear that the wide range of available lesson planning and web development tools are useful, but only when teachers have training to use them.

It takes a large amount of initiative on the part of a teacher to create computer experiences for their students. Some of the examples prepared by teachers are websites contained in the WebQuest and MarcoPolo on-line Libraries. Teachers have to undergo training to be able to complete the templates provided by the site.

One common problem encountered by teachers is that the already-created resources on the web are not in an easily used format. It is difficult to extract the exact content you want without causing navigation problems.

Another concern, in the age of copyright protection laws, is that a teacher does not want to use anything that is not clearly permissible. It is difficult to determine the permissible conditions of use of much of the material on the Internet, or to determine the authorship and date of a resource.

Learning Objects are a boon to teachers when the educational content matches the teacher's curriculum and objectives. The metadata used with the object allows for search and use restrictions. However, when resources do not match the teacher's needs, the teacher typically forgoes the use of the object, although the material may contain useful content. Sometimes the teacher reuses the information, using cut and paste techniques into HTML pages, which limits any reusability [Wiley 2001].

Wiley's IA made it easy to include and review LOs. It was during these reviews of LOs, that the weaknesses in the educational material became apparent. An instructional

experience with many LOs constructed in this way is similar in analogy to the experience one might have if they had a meal with each course at a new and different restaurant. Using the restaurant analogy, each Learning Object has its own atmosphere, menu, layout and many escape routes out of the building. It was the lack of instructional control, lack of a uniform navigation and lack of cohesion in the presentation of the learning objects that were the motivating factors in searching for a solution to separate the valuable educational content from the presentation. This separation also would allow the delivery of the content in a consistent style following learner-centered design principles.

The abilities to reuse and share resources are considered important characteristics that will lead to timesavings.

It was discovered that a few changes from the traditional point of view have to be adopted in order to facilitate reusable content. The first change is that instead of the completed lessons that are pervasive in Learning Objects, we need a design for educational content components and a tool to facilitate the use of the components.

The second change is that instead of providing many different program options that are accessible through menus and buttons, we need programs of limited scope with simplified menus, to improve reusability, and ease the incorporation of technology into the classroom. Teaching time should not be wasted on writing instructions for using the on-line tool, or on a navigation scheme that is unique to that application. Students, however, need to be able to find where they must go to be able to start an activity. That requires more one-purpose tools, with easy start and stop options, and an intuitive learner-centered interface with minimal distractions.

Lesson planning tools that help generate web materials, and web development tools that assist in instructional design are rare. What appear to be missing are technologies that provide time-saving tools for the teacher to distribute, share and implement educational materials. A lesson planning tool must be simple to use, deal strictly with educational content, and provide the plan in a number of formats to meet the teacher's requirements.

It is proposed that TuLiP is such a tool, where pure content can be custom assembled by the teacher and presented in a way to meet the needs of any teacher and his or her students. Table 3.5 summarizes the reusability and sharability aspects of the tools sampled, commenting on their strengths and their drawbacks.

Table 3.5: Aspects of Commonly Used Lesson Planning Tools

Description	Examples	Format	Sharability	Reusability
Journal/ Calendar	Scholastic Lesson Planning Book	Weekly/ Daily/ Class Period Paper/ Bound	Limited due to everyday use, may be copied	Used as a reference, but new dates require rewriting.
Planner	Palm Lesson Planner	Palm/ PDA	Files are not designed to be separate from calendar	Digital copy of lessons can be copied to next year calendar
Curriculum Guides and Lesson Plans	Office production software	Applications in proprietary formats	Must have same applications	Easier to update from year to year than paper
Web and Multimedia Designer Software	WebQuest student activities	Website – some html template pages available	Format and navigation must be acceptable for reuse. Unloadable	Due to the scope of site, files are not easily located or bundled. Need HTML editing skills.
Applications	Java applets and programs	No standard format	Packaging and ease of incorporation determines use.	Ease of use determines reusability.
Learning Objects	e-learn (Microsoft), SCORM	No standard format	Directories, catalogs or search engines are needed. May need proprietary applications	Used as is. Size or format of the object may not match need.

CHAPTER 4

MEETING THE CHALLENGE – TULIP

4.1 Proposed Lesson Planning Tool's Design

TuLiP is designed to use a set of subject-independent XML data structures based on a functional analysis of the most common learning and teaching scenarios. It uses the current technologies for web services, adaptive instructional technologies, and collaboration. In use, the teacher would prepare content-based lessons and have a selection of very powerful options for the format of products. It will be possible to generate computer-based teaching materials for use in a wide range of formatted products. These would include printed versions to be used in the classroom, web pages for reinforcement of the lesson material, adaptive lessons for different learning styles, and output to wireless technology for high-tech classrooms and accessibility technology for students with disabilities. A benefit of this particular tool is that it is possible to produce resources that may be shared and reused. As more resources in this format become available, teachers will save planning time.

The tool is designed to leverage the current XML and Java technologies to produce sharable, reusable and searchable content and diverse products. The design is discussed in six parts:

1. A well designed teacher-centered interface for developing lessons
2. Characteristics that remedy drawbacks of educational resources

3. The use of Learning Objects
4. Fundamental Learning Objects (FLO) and Knowledge Type Templates (KTT) to organize instructional content
5. A markup language that describes the Learning Environment and Planning (LEAP) content
6. A rapid-development web-site platform (Cocoon 2), and
7. Components to supplement and facilitate use.

4.2 Teacher-Centered Interface

Teachers need tools with a simple teacher-centered interface, with sufficient functionality to assist in performing their responsibilities. A well-designed teacher-centered interface should minimize the training burden, and a set of simple and easily-reused objects should allow lessons to be designed quickly and easily. An interface that may be used is shown in Figure 4.1.

The TuLiP tool is designed with a flexible, minimal but sufficient teacher-centered Web interface to reduce the technological burden of instructional design tools. The items in the interface were determined from tools commonly used for planning.

The most frequently seen format is a form that is completed. The TuLiP interface needs to allow customization of the planning form, so the teacher can select what parts are needed in his/her lesson plan. A teacher's default lesson plan form can be saved and retrieved when needed. The tool determines the type of FLO needed for each part, and stores each part in a reusable file.

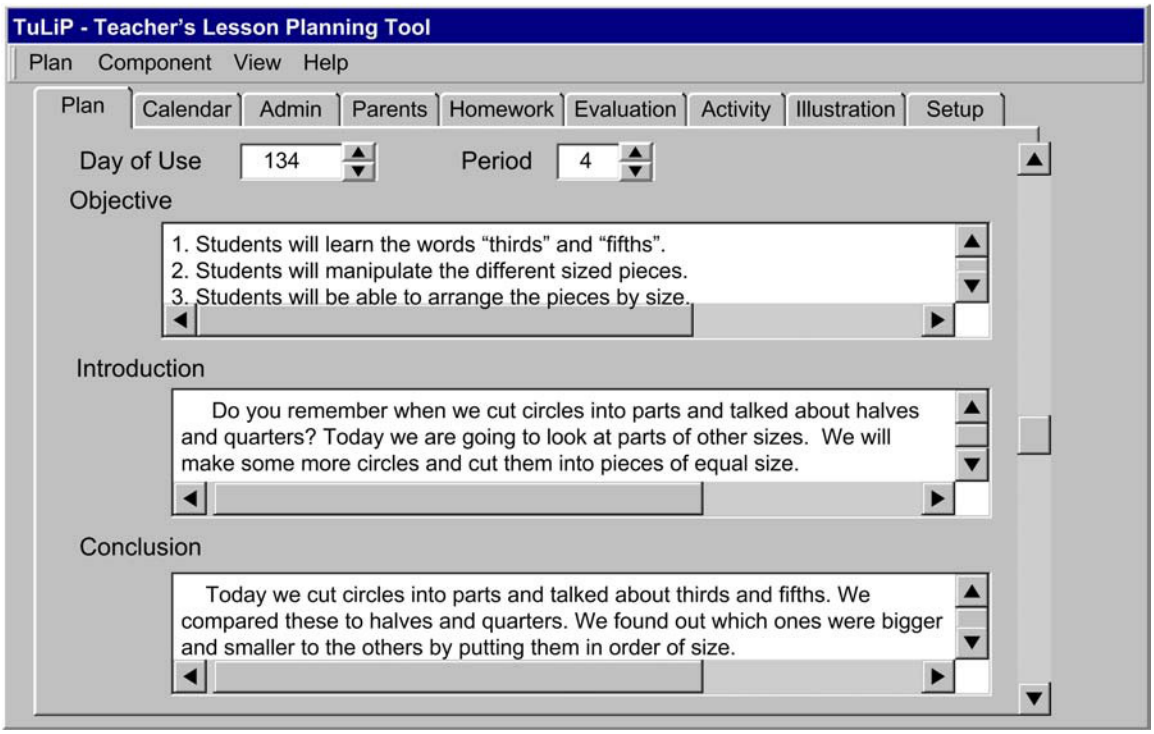

Figure 4.1: Illustration of a Simplified TuLiP Tool Interface with Resources for Lesson Planning.

4.3 Characteristics

With the time constraints, teachers will use technology when it is perceived to be beneficial. If the resources are difficult to use, a teacher will do without or create from scratch rather than struggle with already prepared material. It was evident from tools on the web, that easy to use tools are needed [Gabrowski, et al, 1998] with accessible, portable, sharable and reusable educational content [Wiley, 2002]. The following is a list of general characteristics from the sample of tools available to teachers on the Internet reviewed in chapter 3. A tool, that is to be used by teachers to save time, needs to meet the following criteria:

- Limited in scope and functionality: it should be no more complicated than completing a form, as in the Instructional Architect and STEPS.
- Resources need to be readily accessible, with instructional content for teachers by domain and age group, similar to the directories and repositories of WebQuest and SMETE.
- Resources need to be free of charge and easy to review. Examples are the Gutenberg Project [Project Gutenberg, 2001] and the Open Directory [Open Directory Project, 2002]. Teachers lose a lot of time in software review and preparing purchase requisitions.
- Information is stored in a format that can be bundled or "cut and pasted, " such as the images from the European Space Agency [ESA, 2002]. Complicated learning objects and websites are limited in their reuse [Wiley, 2002].
- Creation, and the edit and reuse of existing information are easily done. XML is viewed as the standard way information will be exchanged in environments that do not share common platforms [XML.org, 2002].
- Sharing and distribution is easily accomplished using the Internet. The World Wide Web Consortium (W3C) develops interoperable technologies (specifications, guidelines, software, and tools) to lead the Web to its full potential [W3C.org].
- Help needs to be readily available with the tool, graduated to level of the user, from demonstrations and FAQ, to a community of users, such as the help provided at the Learnativity [Learnativity, 2002] and EOE [EOE, 2002] portals.

4.4 Learning Objects

The use of structured objects appears to be beneficial for our proposed tool. The characteristics of the components need to be determined.

There are a number of initiatives to produce reusable, portable, sharable training materials known as Learning Objects (LO). The LO is a new way of looking at educational content. The University of Wisconsin Center for International Education Online Resource Center has a definition for the Learning Objects. From their definition, LOs should have these characteristics:

1. be smaller than lessons,
2. self contained,
3. reusable,
4. can be aggregated, and
5. tagged with metadata.

Many things referred to as LO's vary from this definition.

The benefit of using reusable educational materials is demonstrated with the Department of Defense (DoD) Advanced Distributed Learning (ADL) Shareable Content Object Reference Model initiative (SCORM) for military training materials. Under the Advanced Distance Learning initiative, the Department of Defense has reduced the cost and improved the quality of training by using sharable, reusable educational materials [ADL, 2002]. Another example is the Open School Project [Open Learning Agency, 2001] for core high school curriculum in Canada. These projects produce courseware for a core curriculum that does not change on a year-to-year basis. The reusability of these is

possible when you wish to teach a specific course, with the particular teaching strategy and teaching methods incorporated in the courseware.

Learning Objects could be easier to use if the most common educational content was available without overly-specific presentation style or instructional strategy [Downes, 2001]. Separating the content from the logic, and/or the presentation is an example of a term known as the *Separation of Concerns*. This is an important characteristic of easily reusable educational materials. Instructional materials not only have information content, but also have presentation style and lesson control logic. It will help to differentiate the types of objects used in education according to Wiley in his discussion on granularity of learning objects in "Connecting Learning Objects to Instructional Design Theory. "This type of thinking manifests itself as people equate learning objects with 'content objects' to the exclusion of 'logic objects' and 'application objects', for example." [Wiley, 2001]

Most Learning Objects contain the style, and logic along with the content as an integrated object. A problem arises when the format does not match the teacher's needs. The tool needs to either break the content down to usable pieces, or create it that way, and allow control of the display and navigation.

4.5 Fundamental Learning Objects (FLO) and Knowledge Type Templates (KTT)

This section discusses the development of the fundamental learning objects (FLOs) used in this design. A task analysis was performed to come up with terms used to describe the components of a lesson plan, the functionality of lesson-planning tools and the components of on-line learning environments. It is expanded to cover learning events by including interface, interactive, and agent components as well. In addition, these should facilitate the generation of an interactive environment. A sample of teaching

scenarios was analyzed, treating the learning environment as a functional model and the execution of a lesson as an event model. Gibbons performed analysis of the learning environment in a similar way. He describes a number of functions used within a learning/teaching environment [Gibbons, 2001].

In order to come up with classes of components that could fit together, the next step was to divide the tasks and events into objects with characteristics of *orthogonality*. *Orthogonality* in this paper is defined as the principle that only one class of object contains a particular functionality or content attribute. *Completeness*, as defined within, is the principle that each class has one basic objective and that it contains all the information needed to meet that objective. The attributes and functions needed for completeness were determined by the information needed to make it usable as a learning object. This breakdown of types of information and the necessary tasks are described in Table 4.1. The proposed separation of tasks was tested for sequencing, to see if they could follow the teaching events described by Gagne [Daniels, 2001]. FLOs used in combination had the structure to contain lesson plans that would satisfy the objectives presented in Bloom's taxonomy for cognitive and behavioral outcomes [Clark, 2002]. The taxonomy of teaching strategies by Park [Park, 1996] was checked to see if the design had the breadth it needed to encompass different strategies. The additional objects to handle the objectives in the psychomotor domain will be considered in future designs.

To understand some of the uses of the Fundamental Learning Objects, some examples of possible uses is presented. An example of an *Informative* FLO used for informative materials might be an on-line dictionary that provides definitions in a form that has a term and the definition marked, with a topic for the different definitions associated with a

term, wrapped with the reference name, publisher and date. Each definition can be used in the fully-expounded format like a dictionary entry. It can also be used as an entry in a glossary list, by term, or in a list of terms by topic. It has the following uses in educational materials:

A teacher is preparing a lesson plan and has a glossary of terms to include. A search can be made for a definition for each term matching the topic area. Depending on how the information will be used, it may be presented in a number of formats. The list can be presented as a printed resource to go to when the student is stuck. It can be a vocabulary list, with definitions appearing when you hover your pointer over the term, on a computer screen. The information may be used on a page of all the terms with definitions to be used in the teacher's notes. A teacher may want just one particular definition for a term, to appear on a page when a student needs the definition during an on-line review session. Children's pages need not show the reference, but middle and higher levels need the reference included with the display, for copyright awareness.

The other types of FLO objects are *Illustrative, Evaluative, Cognitive, Communicative, Collaborative* and *Adaptive*. Each one addresses a different aspect of teaching. The *Illustrative* and *Informative* make up expository modules where information is introduced. The *Evaluative* modules may be used for teaching using the Socratic Method, for practice and assessment. *Cognitive* FLOs encompass the use of mind tools and learning objects. “Mind tools” is a term applied to applications that may be used with instructions to facilitate cognitive processes, beyond the knowledge required to use the application. *Communicative* FLO initiates the agents used to start client software, voice interface or narration for content. The *Collaborative* facilitates peer-to-

peer work with communication and collaboration services. The *Adaptive* FLO, covers methods such as affective and interactive experiences such as those received in tutoring, sets up the additional information and instructional structure needed for independent learning experiences.

Table 4.1: Fundamental Learning Object Objective by Description of Typical Content and Functionality.

FLO Objective	**Description of Instructional Content/ Functionality Contained In FLO Class**
Informative	Descriptive information, abstracts, introductions, summaries, definitions, reference material, document excerpts
Illustrative	Pictures, illustrations, tables, graphs, figures, examples, diagrams
Inquisitive/ Evaluative	Questions, type of question, selections, prompts, hints, examples, solutions, evaluation, point values, total value; This requires logic options to be added for interactivity.
Cognitive/ Experience Expanding	Structure for video, sound files, Java applets, hypermedia activities that require players or runtime environments, client-based, server based applications and agents
Adaptive/ Affective Interactive	Structure for graduated help scheme and student monitoring of progress in completing a task; incorporates a four phase learning cycle sequencing with student monitoring
Communicative/ Instructive Pedagogical Directive	Structure for interface agent as an expert, an instructor, a mentor, a reader, or assistant
Collaborative/ Peer to Peer	Structure for web services for peer to peer through chat, instructor by chat or email, threaded discussion, interface agent as peer or instructor

These small components would be used in aggregations to construct a lesson plan or a learning environment. This called for the development of a set of templates that already contained a representative set of the FLOs to meet the different teaching objectives.

The types of teaching objectives that could be covered by prepared templates are:

1. Fact
2. Event
3. Skill
4. Process
5. Experience
6. Analysis
7. Experimentation
8. Cognitive Process

Excellent examples are available on strategies and methods to learn facts, skills and processes [Cisco, 2000]. Advances in Historical and Event Markup Language [HEML, 2002] implemented with Cocoon2 can be used to see what aspects may be desirable in teaching events. These can be developed from the best practices determined from research in each of the subject domains.

All that is left in creating the objects is to locate an appropriate XML language document type definition. Using a predefined definition would facilitate reuse by others. A review of available markup languages was performed. It was discovered that there is no language with sufficient breadth to meet the needs of the FLO to be used in the TuLiP tool. The discussion of the design considerations of the needed markup language to code the content is in the following section.

4.6 Learning Environment and Lesson Planning (LEAP) Markup Language

Initially, the use of an existing markup language was considered for TuLiP. A sampling was performed to determine the languages that were used for learning, educational and teaching materials on the Internet. By far, the most common is HTML with multi-media and hyperlinks. Dynamic scripting languages such as JSP, ASP, PHP

and Python are also used for interactive material. Although these are powerful, they perpetuate the current state of affairs, generating educational materials with one-type-of-use.

The standards for learning technology were reviewed for applicability. The IEEE Learning Technology Standards Committee (LTSC) study groups cover many topics. One group is developing the Learning Object Model (LOM). This standard proposes metadata to describe a learning object. The metadata allows it to be located, as a card catalog helps to locate a book at a library. Due to its intent to describe all learning objects, it does not have recommendations for content. The metadata elements pertain to publishing concerns and very little of educational concern for a K-12 teacher. For educational purposes, the description element is usable, but a uniform format of the description needs to include information of interest to teachers. It appeared that the teacher's lesson planning content needed additional metadata elements.

Existing markup languages were considered. Many good characteristics and drawbacks, as well, were determined during the survey of markup languages. It was discovered that there are very few markup languages used to document the aspects of lesson plans. Most are for the development of instructional content or the product of instructional planning, i.e., the end product. There are markup languages used to construct and define structured on-line learning environments. The design stages, i.e. lesson planning stages, are not described by the current set of available markup languages. Two options were considered. The first was to take an existing markup language and create a compliment of the language, so when combined, both could be used to describe the lesson plan to learning environment concepts. The second option was

to base the markup language on the Fundamental Learning Object attributes and functionality using educational terminology for the elements, using the best characteristics of existing markup for appropriate FLOs.

The main references used to determine the best characteristics are the descriptions of the discipline-based markup languages on the "XML Cover Pages" at xml.coverpages.org, hosted by Organization for the Advancement of Structured Information Systems (OASIS) [Cover Pages, 2002]. This site has an introduction to each markup language and a set of links to its documentation. The list has reached over 450 in number. A few markup languages have been adopted by the World Wide Web Consortium (W3C), such as MathML [W3C.org, 2002]. Although there are many markup languages for different domains, only seven (7) of the 450 referred to learning, teaching or education!

Schools Interoperability Framework (SIF) and PESC refer to data exchange between schools and government agencies, vendors and financial organizations. The Instructional Management Systems (IMS) metadata project, ADL Sharable Content Object Reference Model (SCORM) and Universal Learning Format (also called Learning Content Format) deal primarily with the metadata to describe the learning objects. Only two (2) refer to the educational content directly. These are the Learning Material Markup Language (LMML) by Christian Süβ [Süβ, 2000] and the University of Bristol's Tutorial Markup Language (TML) [TML V.4, 2002]. Two (2) additional languages were located as well. These are SGML/XML used by the Open Learning Agency and Open School Project (OLA/OSP) [Klassen et al, 1999], and the RLO/RIO concept from Cisco Systems [Cisco Systems, 2000]. These four were reviewed for use in the TuLiP Tool.

Tutorial Markup Language (TML) has content written in XML, processed by Perl scripts to generate the webpages for NetQuest, at the Institute for Learning and Research Technology at the University of Bristol [NetQuest, 1995]. The software has been developed both for authoring and for displaying TML documents. It is designed to provide the structure for learning experiences taking the form of expository information, followed by practice questions, a drill or an evaluation, and a conclusion. This is appropriate for fact-based knowledge.

SGML/XML, as described by Klassen et al at OLA/OSP, presents a structured framework to describe core curriculum courses [Klassen et al, 1999]. Canada has National and Provincial curriculum standards. Material, meeting the standards, developed at one school, may be used at another. The OS Project in British Columbia, Canada, included instructors, developers and administrators, who worked together to develop educational materials to be used in K-12 education throughout Canada. The first hierarchy used was to build the structure of the curriculum by course, by sections within the courses, and by individual lessons. The lessons take the form of an informative exposition, followed by activities and assessments in varied formats, to address the needs of different student profiles. The advantages of a structured approach are discussed in Paille's conference paper "The effect of using structured documents (SGML) in instructional design" at the North American Web (NAWeb) Conference in 1999 [Paille, Klassen, and Maxwell, 1999]. The ability to generate alternative materials easily was demonstrated by using the structured language. It uses templates for units, so it is easy to produce courses in similar formats. This language uses the major formatting and elements

used in the printing industry. It does not define the content to a size smaller than a lesson unit that would allow easy capture and reuse of content.

The LMML language was developed by Christian Süß (Suess) in 2000 [Süß, 2000]. It is based on modular XML, with data definitions for the major parts of on-line learning and instructional environments generated as HTML. It appears that the logic is incorporated in the content. Modular XML includes only those files that it needs to perform the transformations. LMML contains document type definitions (DTDs) with the elements designed for specific domains of learning materials. The DTD element labels mix domain-based with instructional terminology along with the technical term for the data contained in the elements. Examples of these are the following: some elements use subject/jargon terms such as Proof and Equation as types of SpecifiedObjects; some elements are educational terms, such as ConceptualUnit and ContentUnit; and some, such as FloatingText, are technological terms that refer to content and not to use of the information. This may be seen in Figure 4.2, which is the Passau TeachWare Model, Used with the Learning Material Markup Language (LMML).

The products from this model are primarily for process, skill or factual knowledge accumulation. It does not appear to incorporate the flexibility to be used in other domains with other teaching strategies.

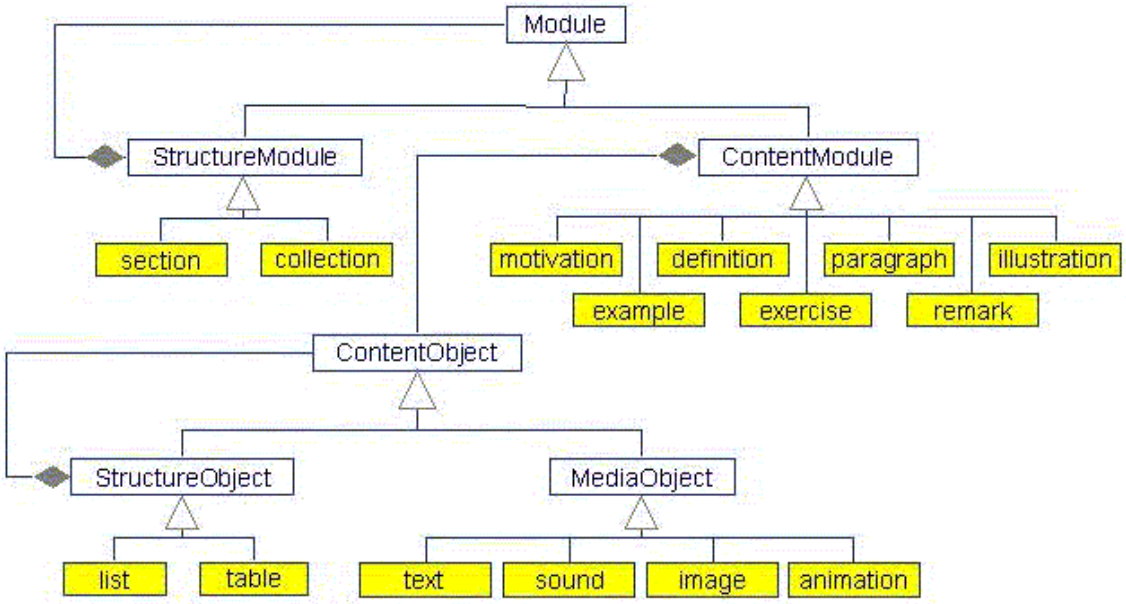

Figure 4.2: Diagram of the Passau TeachWare Model, Used with the Learning Material Markup Language (LMML) by Christian SüB.
Source: http://www.lmml.de/index?text=22

The Reusable Learning Object strategy (RLO) from Cisco Systems [Cisco Systems, 2000] is used to train people to use particular products and perform particular processes. It is a strategy of using smaller pieces called Reusable Informational Objects (RIOs) that vary in content based on the information they will contain (such as concept, fact, process, principle, or procedure information). Each RIO is composed of content, practice and assessment material. Each type of RIO has guidelines for the minimum information and activity requirements needed to cover the information. One example is that of a "procedure" RIO, which requires the inclusion of a procedure table and a decision table in the content section. The RIO can be considered a Block or topic. A bunch of these are

assembled into a Reusable Learning Object (RLO) based on a single objective, also referred to as a Lesson.

In summary, some of the markup languages reviewed support development of learning material with the granularity of a lesson, while others mimic textbooks on-line in producing products in HTML. The most common structure for a learning experience took the form of expository information, followed by evaluation and/or activity (as in the case of RIO and TML). This sequence and content is appropriate for computer-based training (CBT), and ideal for training processes or skills. LMML uses a different structure for each discipline (such as one structure to teach database theory, and another to teach opera) with potential conflicts for multi-discipline topics. If we use the same markup language that is used for the textbook printing industry, SGML, with marking specifically for formatting, the best we could generate is on-line textbooks. The current selection of markup languages does not have the breadth needed to implement FLOs. A summary of the findings of the review of markup languages is outlined in Table 4.2. Therefore, it is clear that a new markup language needs to be developed for TuLiP that is capable of constructing FLOs and providing the structure for reusable objects. It needs to be designed to use education based terminology, to build a structure for the storage of information for instructional content, instructional events, system functionality, and sequencing of the learning experience from lesson planning to learning environment. The proposed Learning Environment and Planning Markup Language (LEAP) is designed to meet these requirements. Table 4.2 describes the strengths and drawbacks of the current languages used in creating learning materials. It lists proposed solutions to the drawbacks and the beneficial features to be incorporated into the LEAP design.

Table 4.2: The Summary of a Review of Languages Used to Create Instructional Materials, their Features, Drawbacks and the Proposed Solution in LEAP.

Language	Features	Drawbacks	LEAP Solution
LOM/SCORM	Metadata	Not used for educational content description	Include instruction and teacher-specific metadata.
HTML	Static web pages; easy to use in the Classroom, independent of platform	Not easily reused; Incorporates navigation and lesson control	Output product in HTML, but store content, style and logic in different structures.
JSP, ASP, PHP	Dynamic functionality with "types of pages" and variable content	Typically a one-pass process with logic and style together	Develop the logic and style to be applied independently. Produce "types of objects" consistent with the FLO attributes and functionality.
RIO/RLO LMML TML	Standardized Content	Includes element labels from extraneous domains in DTDs; Limited to teaching facts, skills and processes; Size too large for easy reuse	Use lesson planning and teaching terminology, consistent with the FLO design. Allow plan templates to be parsed into usable sizes of objects for reuse. Develop structure for experimentation, exploration and experiences.
SGML	Course or Unit templates used for consistent design; Built-in alternative materials	Many of the terms based mostly on presentation style information. Incorporates logic with style	Use templates of the most common set of FLO (teaching task) and KTT(teaching objective). Develop FLOs to use agent, web services and servlet technologies. Incorporate the alternative materials structure.

The LEAP summary of the design is described as follows:

1. The DTD define the data structure using semantic element labels to describe information. It should have the following features:
    a. Include metadata of instruction and teacher-specific information for searching and retrieval.
    b. Use lesson planning and teaching terminology.
    c. Incorporates the alternative materials structure.
    d. Describe classes of Fundamental Learning Objects with attributes and functionality to accomplish a teaching task. Describe the content, logic and style for each FLO (teaching task). Describe classes of FLOs that use agent, web services and servlet technologies as teaching tasks.
    e. Describe classes of Knowledge Type Objects as aggregations of teaching tasks to accomplish a specific teaching objective. Describe the KTO for teaching objectives, such as experimentation, exploration and experiences.
2. Language should support the implementation of functionality, to be administrated by XML services framework:
    a. Select style information to generate desired product in HTML, PDF and other formats.
    b. Allow completed plan templates to be parsed into objects of usable sizes for reuse.
    c. Store content, style and logic in different structures.
3. Language should support the use of templates for lesson plans:

a. Derive use templates of the most common type of lesson plan to be parsed into the set of FLO (teaching task) and KTO (teaching objective).

4.7 A Rapid-Development Web-Site Platform (Cocoon 2)

4.7.1 Introduction to Cocoon2

Cocoon started as a simple Java servlet, for the eXtensible Scripting Language (XSL) styling of Extensible Markup Language (XML) documents. The Cocoon Project under Apache was begun in January of 1999 by Stephano Mazzocchi. Over the past three years it has evolved to become a web site framework used for web development and management, based on the Avalon framework using Java Servlet technology.

It allows for the generation of diverse output products using Wireless Application Protocol (WAP) and Hypertext Transfer Protocol (HTTP). It allows the production of a variety of printed formats by using XSL Transformations (XSLT) to convert XML documents to Formatted Objects (FO), and using existing FO technologies to generate Portable Document Format (PDF) documents [Mazzocchi, 2002].

All components used by Cocoon adopted development standards and philosophy used in the Avalon framework. A framework is a set of guiding principles and a set of components that function consistent with the principles. This philosophy incorporates a number of concepts such as the *Separation of Concerns*, which requires the use of separate components for each task, and pattern matching for logical branching and configuration. The components encompass a variety of classes of objects such as sitemaps, servlets and components, where each component has one and only one function. The Architecture of Cocoon2 is composed of layers in the Framework. These are shown in Figure 4.3. The outermost layer contains the content and user-developed

components. These are manipulated by components in the lower levels to perform specific functions. The built-in logic sheets are used by components to process content.

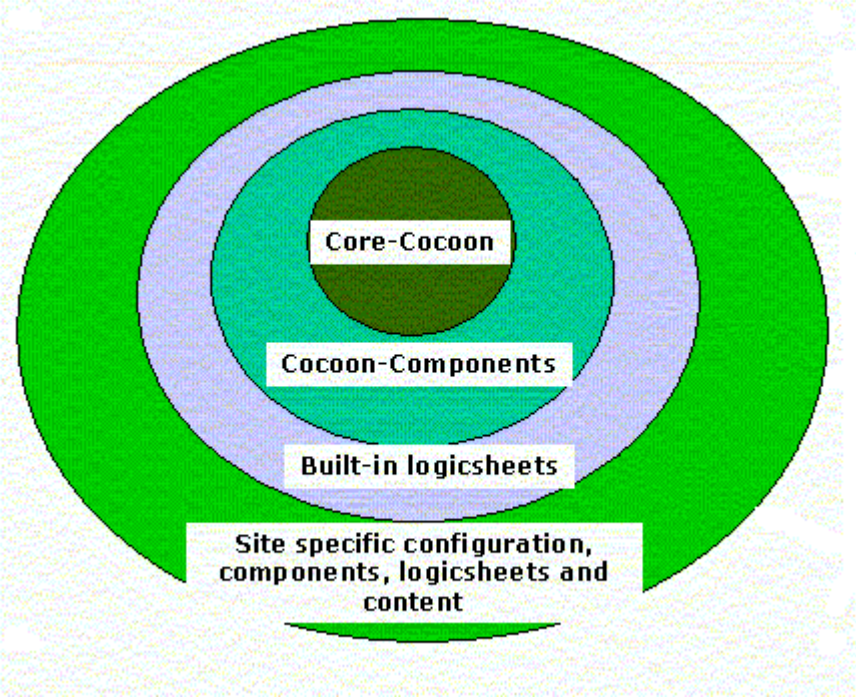

Figure 4.3: Cocoon Architecture. Source: xml.apache.org/cocoon

Core-Cocoon contains the Avalon framework for logging, configuring and threading, caching, pipeline handling, generating, compiling, loading and executing programs, and the base classes for generators, transformers, serializers and components.

Cocoon-components are specific single-use generators, transformers, matchers and serializers.

Built in logicsheets in XSL facilitate the processing of the sitemap, incorporating XSP logic into the pipeline, handling SQL or HTTP requests and generating an HTTP

response. Site-specific configuration, components, logicsheets and content follow the framework philosophy.

4.7.2 Processing of a Request to a Cocoon URI

The Cocoon servlet is initiated by a Java Servlet engine whenever there is a request to a Cocoon Uniform Resource Identifier (URI). A request is handled by the Servlet engine that passes the request to the Cocoon servlet. The request is processed, and a response is passed back from Cocoon to the Servlet engine that passes the response to the client. To process a request, Cocoon initiates the Sitemap to find a pipeline contained in the sitemap.xmap file. This is the location for the declared components and the location for the defined pipelines using the declared components. The sitemap.xmap, as in Figure 4.4, is an XML file at the site's base directory that contains scripts to be performed based on the URI. The pipeline patterns may be exact name matches, regular expressions, or wildcards.

Based on an HTTP request, Cocoon initiates the Sitemap that selects the pipeline to be used at runtime, and compiles any modified components. An action is a component that manipulates the runtime parameters used in the sitemap. This is where form processing and dynamic navigation is handled. The pipeline that is selected matches the pattern in the URI. For example, in the case of a URI "lesson.xml," the Sitemap finds the first pipeline definition that matches, which could be a pipeline with the wildcard "*.xml." The Sitemap initiates the components to be used based on the pattern in the URI request. The sitemap, similar to a configuration file, declares identifiers and assigns source locations for any components needed for processing the XML files of the site.

```
<map:sitemap
xmlns:map="http://apache.org/cocoon/sitemap/1.0">
      <map:components>
            <!-- component declarations -->
            <map:generators/>
            <map:readers/>
            <map:transformers/>
            <map:actions/>
            <map:serializers/>
            <map:actions/>
            <map:matchers/>
            <map:selectors/>
      </map:components>

      <map:pipelines>
            <!-- pipeline definitions -->
      </map:pipelines>
</map:sitemap>
```

Figure 4.4: A Code Excerpt from the 'sitemap.xmap' Used by Cocoon to Initiate the Types of Services Required by an URI Request. Source: xml.apache.org/cocoon

The sitemap contains the identifiers, runtime parameters and match patterns for the different pipelines that contain the possible processes. At runtime, Cocoon Sitemap selects the pipeline, compiles any needed components and uses a component called an Action to process any run time parameters. The Sitemap can pass the parameters and the source attribute to the Action, and the Action can return new values based on the HTTP request or other runtime conditions. The pipeline components act on the input files and process the request, with the Serializer producing the final output that is returned as an HTTP response.

The Cocoon steps that occur as a result of an http request are as follows:

1. Accept a request from a user.

2. Determine the correct pipeline using a matcher to interpret this request and produce a response.

3. Construct the pipeline from specified components.

4. Instruct the pipeline to service the request.

5. Return the response generated by the pipeline to the user retaining results for later use.

Figure 4.5 is a Uniform Modeling Language (UML) Sequence Diagram that illustrates the sequence of events in processing Cocoon requests by the Cocoon Servlet from the Client.

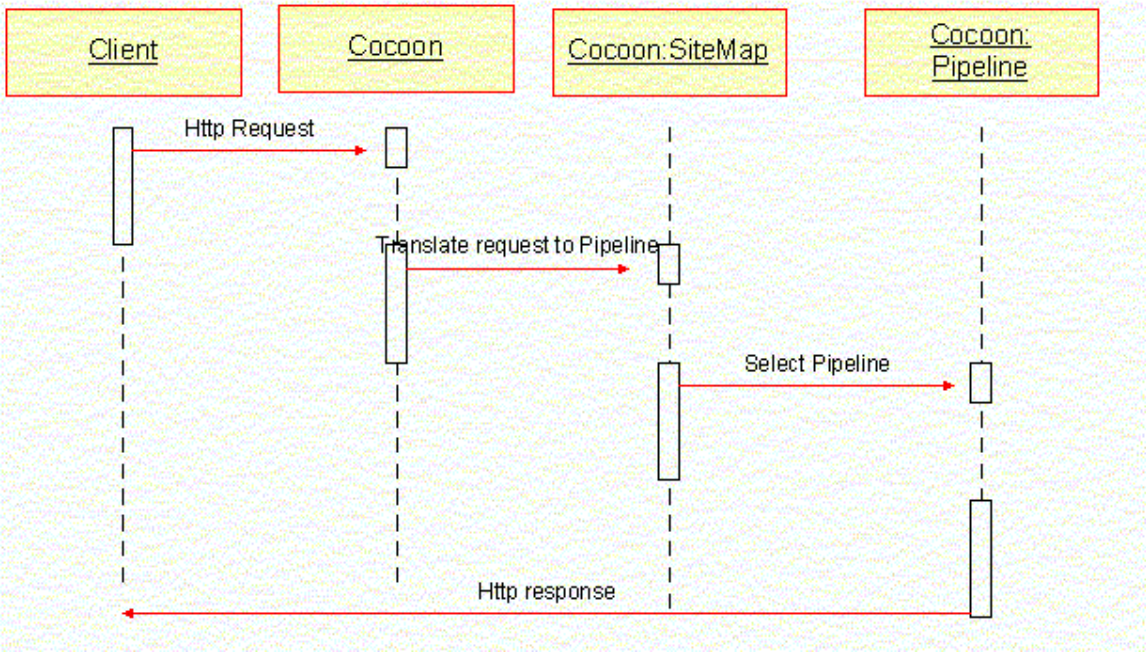

Figure 4.5: UML Sequence of Events Diagram for Processing a Cocoon HTTP Request. Source: xml.apache.org/cocoon

4.7.3 Separation of Concerns

The *Separation of Concerns* is a concept first incorporated in the Avalon framework. It is one approach to address the need for multiple processing of an XML file. The

extensible scripting Language (XSL) has syntax for formatting and logic. Both of these together are better than HTML, which also contains the informational content as well. Early in the use of XML, developers using an XSL scripting file would pass it between the graphic layout designers and the logic programmers. There was always the potential of damaging what the others had coded, and wasting time in repairing, coordinating and testing the file. [Apache, 2000]

If the logic is scripted in eXtensible Server Pages (XSP) pages (also written in XSL), and the style is in an XSL file, then parallel development is possible. However, a servlet to process the XML file along with the XSP and XSL files at the same time presented a programmer's challenge. Typical dynamic pages are coded in scripting languages, such as JSP or ASP. The actions are performed by the servlets and servers that process them. They are typically "a one pass process" where the input file is processed and the product is in the desired display format. An alternative method, possible with Cocoon2, is demonstrated in Figure 4.6.

4.7.4 Cocoon Pipeline Process

The Avalon framework uses pipelines, containing applications, each of which performs one task, then passes the information to the next application to do its task. This is similar to the pipe, which is the symbol "|," used in UNIX. It is placed between commands, to redirect the output of the command before the pipe, to the input for the command following the pipe. A "pipeline" can be constructed of Java applications,

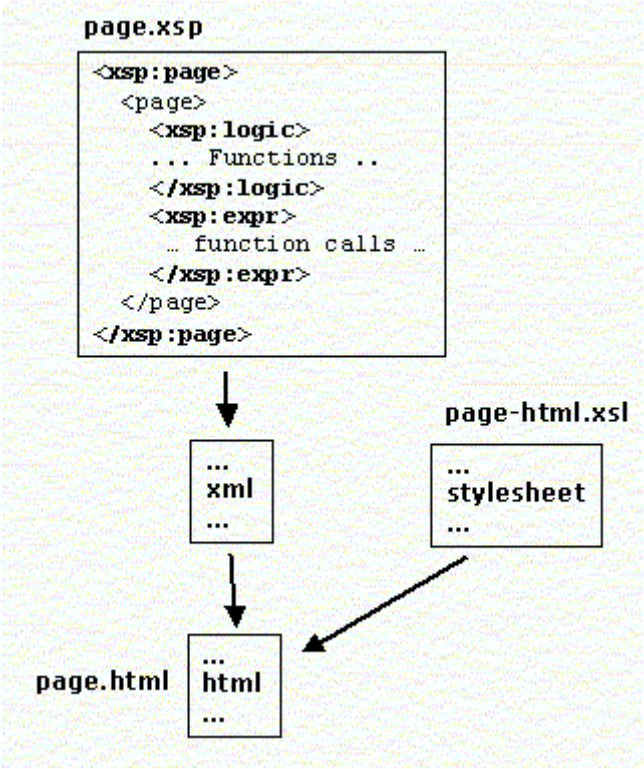

Figure 4.6: Demonstration of Statements to Incorporate Logic and Functionality as Separate Elements within a Logicsheet Using XSL. Source: xml.apache.org/cocoon

compiled in the order to be performed, along with the necessary parameters and the source location for each input file.

The information is passed between the applications in the pipeline using a Simple API for XML (SAX) events. This presents the need for three different types of applications as shown in Figure 4.7. The initial one in the pipeline, called a generator, starts with a content file such as the XML source file and generates the SAX events signaling the start or end of each element tag. Multiple numbers of XSL transformers follow in the pipeline converting the event from one format to another. Each one performs one type of transformation, such as adding logic based on the element tag, or adding style to the

events. The last application in the pipeline, a serializer, processes the SAX events according to a format definition file (in XSL) and produces the serialized product. Cocoon incorporates a framework, which allows the separation of content, style, logic and management functions in XML-based web sites and web services.

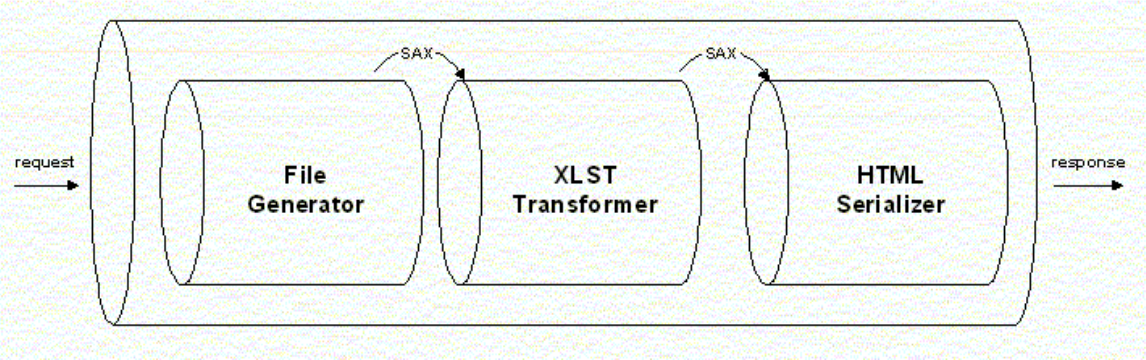

Figure 4.7: A Cocoon Pipeline. This illustration represents the pipeline and components defined in a sitemap.xmap with the input files entering the pipeline on the left and the output from the pipeline leaving it on the right. The communication between the components occurs with the passing of a SAX event that initiates the processing of the next component in the pipeline. Source: xml.apache.org/Cocoon

The options that use of a pipeline and the various components can make available to a teacher is considerable. With appropriate pipelines developed, a file stored in XML can be output in a large number of formats, only by giving it a different extension on the name. A file "myplan.xml" can be displayed on a mobile device by requesting "myplan.wml." From one xml file, pipelines can be designed to provide both a static text page and an interactive environment with the difference in the two pipelines is the addition of the logic page. A person with a text reader would prefer using the static text page, to the interactive environment. The ability to add or leave out a logicsheet will

provide the flexibility for the teacher to control the navigation and sequencing, and consequently have more instructional control. Using different logic sheets with the same content can change a question and answer review session into an on-line examination.

4.7.5 Components in the Pipeline

A generator is used to create an XML structure from an input source such as a file, directory or stream. Generators act as parsers that generate SAX events based on the generated XML structure. The parser goes through the XML structure once and generates different events as it detects a beginning element tag <tag>, an empty tag <tag/> or an ending tag </tag>. It passes the element name and all the attribute information and continues to parse the content. Generators may have a number of input types such as relational database, logic, objects and content. The generator, in that case, produces the corresponding XML and generates the SAX events.

A transformer is used to map input XML structure into another XML structure. The transformer uses an XSL file to determine the transformations to be performed to the information in an incoming SAX event. It determines the element label, and searches the XSL file for a matching pattern for the specific beginning element tag. Branching logic may be written in XSL, with the action performed based on the element's attributes within the tag. An example of a transformation is changing from a simple content-based XML file to an XML Formatted Object (FO) format, which incorporates style format information. It generates SAX events that are passed to the next application in the pipeline.

Aggregation of separate files into one product is also enabled by the use of the pipeline. This is the ability to use multiple input XML files and combine them as needed to provide a unified page format without having an XML document containing all the content, file imports or file includes. This allows the use of the XML files separately or in combination. This function is performed by a component called an aggregator. An example of use is Figure 4.8.

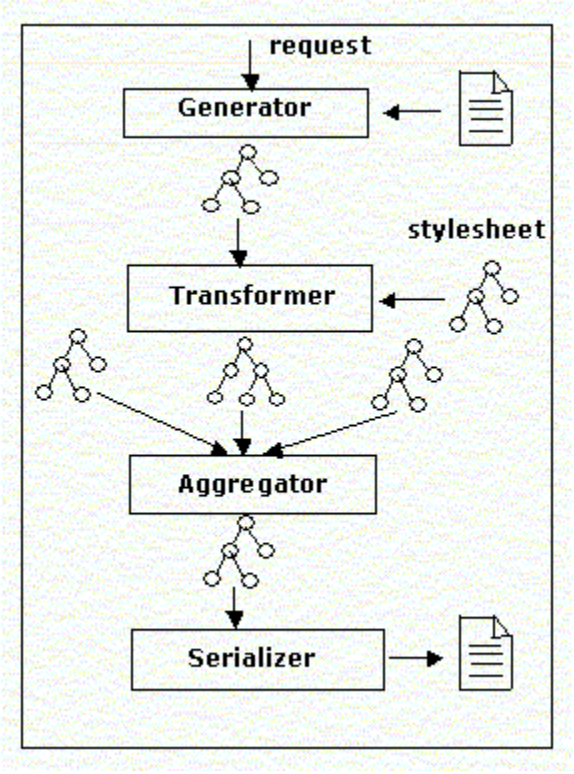

Figure 4.8: The Cocoon Pipeline Process with Components, XML and XSL style sheets using an Aggregator. This illustration is of the entire process showing the different components and input files. The input and output are represented as pages. The style sheets, data definition files and XML files are displayed as hierarchical tree structures. Each type of component is labeled as to its function, such as the Generator, Transformer, Aggregator, and the Serializer. Source: xml.apache.org/Cocoon

Serializers are the last type of application used in a pipeline and are used to render the XML. The serializer receives the SAX events and writes to an output stream such as a file, a port or standard output. The output stream as an HTTP response will be displayed in the browser. A commonly used serializer is an XML to HTML Serializer. This takes an XSL file that describes how each element in the XML is to be displayed in HTML, and writes the corresponding HTML to standard output to the HTTP response. When viewed with a browser, the page appears as an HTML page. A variety of Serializers are already available in the Cocoon2 framework, such as PDF, WML, VOX ML and SVG.

4.7.6 Multiple Formats from the Same Content

One of the most powerful features of the Cocoon2 framework is the sitemap, which uses Uniform Resource Identifier (URI). With an appropriate naming scheme in TuLiP, Cocoon allows the parents, administrator or student to check on the web for information specific for them, which had been stored in one plan. Table 4.3 demonstrates the parts of a Lesson Plan that may be reused for the generation of additional products designed for different audiences.

Administration must be assured the teacher is following the curriculum and appropriate assessment methods are used. The student needs sufficient information to complete or review the lesson, assignment or evaluation with minimal assistance. The parent needs to know what the child is learning and where to find the resources to help the child.

Table 4.3: Demonstration of Reuse of Parts of a Simplified Lesson Plan for the Generation of Additional Products Designed for Different Audiences.

Teacher's Lesson Plan	Administration	Student on-line Resources	Parent on-line Resources
Metadata Construction Detail Curriculum ID Objective ID Reference of Content	All	None	Construction & References
Review Resource List and Locations		All	Resource List
Instruction Objective Evaluation Method	All	None	All
Task List Sequence plan	All	None	Task List
Information List Introduction Notes Outline Lecture Text Conclusion Resources List Glossary	Information List Resources List	Introduction Notes Outline Lecture Text Conclusion Glossary	Information List Resources List
Illustration List Description Points of Interest Player/Equipment Resource Location	Illustration List	Description Points of Interest Illustration	Illustration List
Activity List Objective Directions Tool Use instructions Resource location Examples FAQ Solution	Activity List Objective Tool Resource location	Directions Tool Use Instructions Examples FAQ Solution	Activity List Objective FAQ
Evaluation List Questions Prompts Hints Value Answers	Evaluation List	Questions Prompts Hints	Evaluation List
Post-instruction Main Points Explore List Next Lesson Introduction		Main Points Explore List Next Lesson	Main Points Explore List Next Lesson
Assignment Activity List Information List Resource List		Activity List Information List Resource List Resource Location	Activity List Information List Resource List

A potential naming scheme to take advantage of the strength of Cocoon to automatically produce different products for a number of audiences is shown in Table 4.4. A teacher's plan may be saved as nnnpsss.ktt, an arbitrary default-naming scheme for the filename for a lesson plan. The nnnpsss is with the teacher initials represented as nnn, the period as p, and the particular school day as sss. Each audience group would be given the naming convention for each day and use their particular extension at the end to access their information. The files listed under Style would be default style sheets to be used with all plans to display only the information needed for that audience. They can be modified during setup. Note that the actual naming scheme of the files is arbitrary with few requirements. The requirements are that it differentiates between teachers, classes and identify each lesson plan uniquely. It just as easily could be a teacher id, followed by subject, followed by the unit number.

Table 4.4: The URI Naming Scheme for Lesson Plans and Products for Different Target Audiences.

URI: http://the.school.cnty.st.us/nnnpsss.*			
Source Lesson Plan in XML stored as: nnnpsss.ktt			
Audience	URI Extension (replaces the * in the name)	URI example	Style Sheets for Products
Teacher	.plan	Nnnpsss.plan	HTML/PDF
Administrator	.adm	Nnnpsss.adm	ExtractAdminInfo.xsl
Parents	.par	Nnnpsss.par	ExtractParentInfo.xsl
Students -default	.go	Nnnpsss.go	Lesson.xsl
Students -default	.do	Nnnpsss.do	ExtractAssignment.xsl
Students -visual	.v.go	Nnnpsss.v.go	AddVisuals.xsl
Students -aural	.a.go	Nnnpsss.a.go	AddAural.xsl
Students - k-3	.k-3.go	Nnnpsss.k-3.go	Lesson.k-3.xsl

In order for Cocoon to provide the audience specific information, each of the uses are described in a pipeline with a match to a wildcard with the extension used in the URI. There will be a pipeline for *.adm and *.par for example. Whatever string is placed in the wildcard location it is saved to a variable{1}, that is used to find the corresponding ktt, or xml file with the matching file name, but with the ktt extension. The teacher plan and administrator pipelines are demonstrated in the code sample from the TuLiP Demonstration page, in Figure 4.9. In order to display specific information for one audience and not another, style sheets need to be designed with display instructions for each element.

```
<map:pipelines>
	<map:pipeline>
		<map:match pattern="*.plan">
			<map:generate src="docs/plans/{1}.ktt"/>
			<map:transform src="stylesheets/lessonPlan2html.xsl"/>
			<map:serialize type="html"/>
		</map:match>
	</map:pipeline>

	<map:pipeline>
		<map:match pattern="*.adm">
			<map:generate src="docs/plans/{1}.ktt"/>
			<map:transform src="stylesheets/ExtractAdminInfo.xsl"/>
			<map:serialize type="html"/>
		</map:match>
	</map:pipeline>
</map:pipelines>
```

Figure 4.9: Excerpt From TuLiP sitemap.xmap Demonstrating the Use of the wildcard *. This is used to select the lesson plan *.ktt, with content displayed with different style sheets to direct the transformation based on the extension of the URI.

4.8 Tool Components

From the samples and analysis of resources discussed in this thesis, the following sections list the desirable components and functionality of a fully implemented TuLiP tool design.

4.8.1 Lesson Planning

1. View facility for LP, KTO and FLO, administration requirements and parent information, and student learning environment;
2. Fundamental Learning Objects Template (FLO) Forms selected by Teaching Task;
3. Basic Lesson Plan (LP) Form;
4. Knowledge Type Object Template (KTO) Forms selected by Learning Objective;
5. Save and retrieve from all forms; All work is saved in progress at scheduled times and states of developing the objects. This is accomplished by a database to assist in the construction.
6. Lesson Planning Instructions, Help, Frequently Asked Questions (FAQ), and On-line demonstration of TuLiP,
7. Set up options to customize forms, and
8. Source of tools, components and resources for Lesson Planning.

4.8.2 Portals, Databases, and Repositories to Support Retrieval and Sharing of Resources

1. Repository of stored Plans, FLOs, and KTOs,
2. Options for upload, retrieval and list of the repository content,
3. Catalogs of KTT, FLO's and Plans available by Grade Level, by Topic, and by National Curriculum,

4. Search available by Plan, KTT or FLO, metadata, or specific text,
5. Community of members,
6. Communication services,
7. Shared workspaces and collaborative services, and
8. Links to assortment of educational resources.

4.8.3 Learning Environment

1. Alternative formats of materials; Students can choose to have a narration of the content to speakers, text on the screen, or an illustration with explanations.
2. Materials for students of diverse cultures; Using the text based XML structured pages, translations may be added to the text document for a student where English is a second language. Students can have a selection of languages.
3. Web-based Homepage with portals for each audience; This can automate the access to the particular page for the day. Cookies can retain user information.
4. Session and login for monitoring use, and
5. Secure student data repository.

4.8.4 Administration

1. Administration of site, Style and Tools settings,
2. "Behind the Scene" Demonstration of Cocoon, and
3. TuLiP Project Staff, Contact information.

4.9 TuLiP Development Plan

The prototype design is residing on the Department of Computer Science's webserv2 Apache webserver. The hardware requirements are outlined in the System Design Document, included in APPENDIX B.

The development of the software components has progressed to the current prototype of TuLiP consisting of three (3) components:

1. TuLiP demonstration website of the Cocoon2 framework functionality to produce diverse products from one lesson plan.
2. Examples of pages using the LEAP Markup Language, several Fundamental Learning Objects and Knowledge Type Objects and Lesson Plans stored in structured form.
3. Samples of proposed interfaces and results of user testing.

Development of the tool, the interface and services are planned in the following sequence:

1. Implementation of components in Section 4.8.1 Lesson Planning, in order.
2. Implementation of components items one (1) and two (2), in Section 4.8.3 Learning Environment.
3. Implementation of the Repository, with functionality, search and catalog options as described in items 1-4 of Section 4.8.2, in order.
4. The remaining items will be implemented, as needed and warranted by user needs assessment and testing.

CHAPTER 5

TULIP OVERVIEW AND CONCLUSIONS

5.1 Benefits of Using TuLiP

Why should we construct Teacher Lesson Plans using XML Fundamental Learning Objects and Cocoon2? This thesis presents the design, as in Figure 5.1, of a web-based teacher-centered lesson-planning tool that produces functionally-based, sharable, reusable XML components called Fundamental Learning Objects FLO, constructed from the LEAP Markup Language. These are used in combination in Knowledge Templates (KT) based on the teacher's objective. Even though the KT template would be designed from research findings for teaching different types of knowledge, it may be modified to incorporate the teacher preferences. Both the FLO and the KT may be used to generate a wide range of generated output products enabled by Cocoon scripting language scheme and web framework. By using XSL with a wide range of style sheets, it would be possible to generate web pages that already incorporate learner-centered design principles.

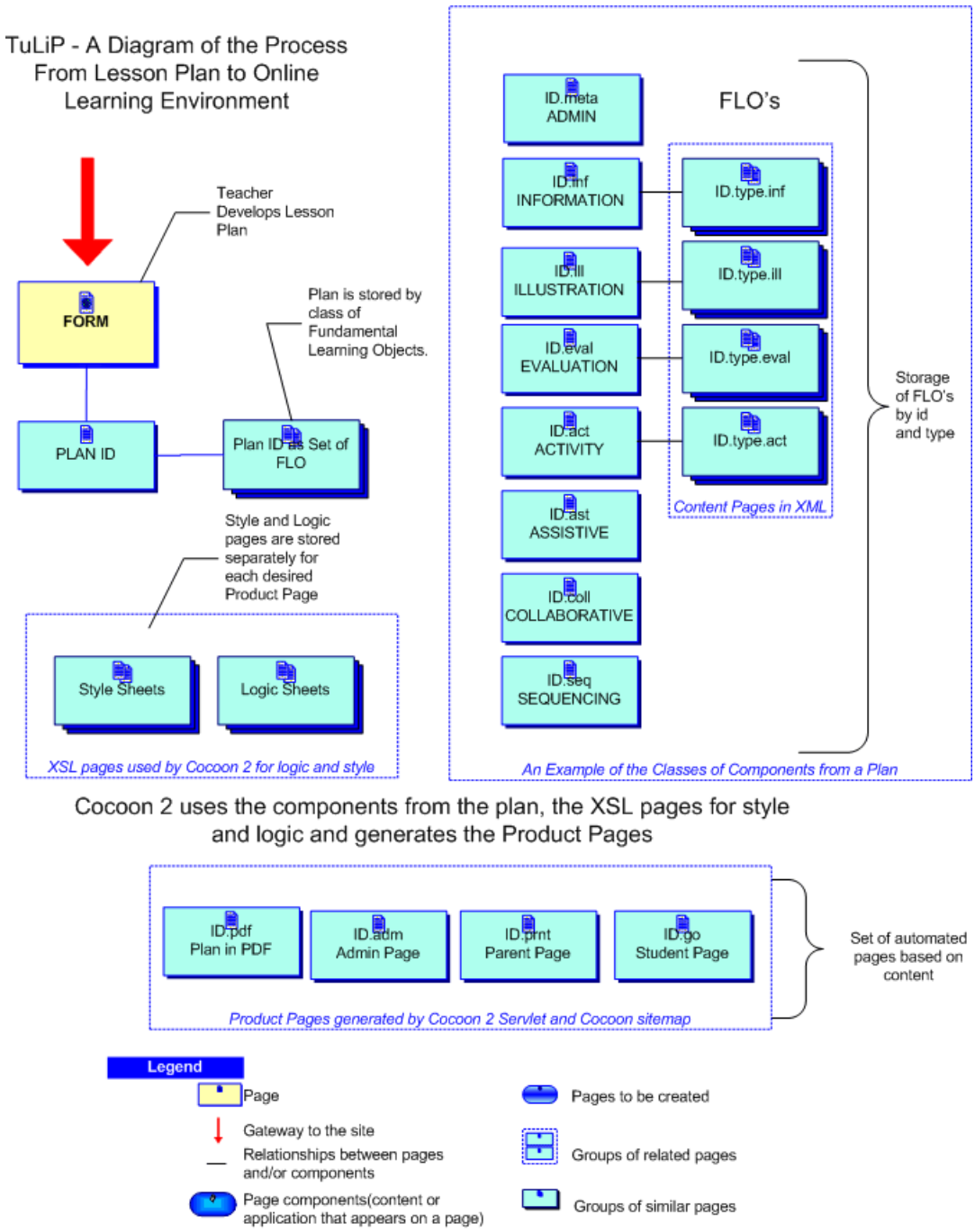

Figure 5.1: A Conceptual Web Design Diagram of the TuLiP Tool, Fundamental Learning Object Repository and the Products Generated Using the Cocoon2 Framework.

But there are a number of problems with this design. Lesson Plans are typically prepared by date. Now, teachers need to plan by objective. Research has shown that different objectives require different teaching strategies. If they plan their class around objectives then the KT can help them formulate good plans to accomplish that one objective. A number of objectives may be done in one day by aggregation of components. It is not the intent for the tool to handle all possible teaching experiences, but the most common types.

Why create another lesson-planning tool? Current tools do not have the output flexibility that will be available using the TuLiP tool. Lesson planning generates useful material that may be easily reused and shared if it was only in the right format. This tool will save it in a portable, sharable and searchable form.

So, teachers will be able to do their lesson plans, and press a button to produce a web site? It cannot generate more content than what is given, but even a web page with just a list of the activities for the day is useful for other teachers teaching the same material, and for the parents to reinforce the material at home.

So what does a teacher need to provide in this format to generate something useful to the student? Many teachers already provide enough information to generate supplemental information. A lesson plan for a lecture to a class may include:

1. An introduction to material,
2. illustrations,
3. explanations of the illustration,
4. lists of thought-provoking questions to see if they comprehend the material and the answers that are expected,

5. a description of in-class activities to see if they can implement the material, and
6. summaries of the material.

Besides being given as a lecture to the class, the content can now be used as an interactive experience for the student. A proposed web presentation may include the narration of the lecture, and an evaluation of their understanding, for which they respond by typing in an answer.

A lesson plan for a lecture in class can now be a one-to-one experience. Because of the component nature of the design, questions used in lecture can be used to produce quizzes or tests on-line.

Why not use other schemes of XML such as LOM? LOM uses an XML scheme to describe the information about existing learning material and does not describe the content. The content may be programmed in any number of different formats, and contains learning materials described as courses, lessons, or blocks of lessons. It does not facilitate the use of the material except in its entirety.

If the components are coded separately we have greater flexibility. By using the components, then the material may be more easily reused. An example of this is "instead of linking the entire encyclopedia, let’s provide just the topics for the lesson."

Well, what about efforts of different disciplines to come up with their own Learning Material Markup Language? There are subject/area-based "learning material markup languages" available. These have elements that are common to that subject area, using field-of-study-based terms to describe the content. This prevents use of these by interdisciplinary fields, with a problem of potentially conflicting data definitions.
We can eliminate this problem and generate the most useful educational content by using

subject-independent markup called Learning Environment And Planning Markup Language (LEAP). The language is based on the educational activities that are performed in the process of planning, teaching and learning. This is not intended to be exclusive, or cover all fields, but it would produce a readily sharable and reusable set of educational resources useful to many fields of study.

Well, how will this save teachers time? It is envisioned that many teachers would contribute to a repository of usable parts. Teachers within a district, working within a particular subject and grade could develop different parts and use the material jointly. If resources on the web used the smaller complete objects, they could easily be incorporated. Format for the pages would be pre-designed, so teachers would not lose time in publishing details. Teachers, however, must relinquish the need to control every detail in the final fixed published product, in exchange for generating more flexible content. Templates for the most common teaching techniques can be readily on hand and used to complete the plans.

5.2 Future Work

The following is a list of items needed to fully implement the TuLiP tool.

1. A comprehensive survey of educational materials is needed.
2. Survey of Teachers' Needs - A wider survey needs to be done to assess teachers' needs to finalize the simplified teacher-centered interface.
3. User Studies of Prototype - An extensive study of the prototype needs to be performed for potential enhancements.
4. Complete Development of LEAP – Language needs to be completed with respect to grammar and classification of classes.

5. Completion of the Fundamental Learning Object Set - The FLO set needs to be completed, and may need to be expanded to include new technologies or to provide better functionality when combined. The need for such changes should become apparent in the use of the prototype.
6. Completion of the Knowledge Type Template Set - The KTT set needs to be completed and may need to be expanded to include new combinations of the FLOs.
7. Use of TuLiP as a Research Tool - With the modular design of TuLiP, interface research could be performed quickly. With the same content, researchers can make modifications to the presentation and layout instead of re-creating equivalent material.
8. Extensions to the Teacher Planning Tool – The portal and a database of curriculum and planning resources should be accessible from the tool. This may include external and/or internal databases.

APPENDIX A

GLOSSARY OF TERMS

Cocoon2: Java-based Web publishing framework under the Apache/Jakarta project

Completeness of a Fundamental Learning Objects: The educational intent that one basic objective is contained in one object and the information needed for that objective is included

Components: Services and aspects performed by other applications besides the Cocoon2 Framework.

Knowledge Type Templates (KTT): The structure of aggregation of FLOs to meet an objective

Fundamental Learning Objects (FLO) templates: The structure for complete learning objects

Learning Objects: Educational materials in various formats

LMML: Learning Material Markup Language

Learning Object Model (LOM): Standard of the IEEE for Learning Object metadata

Metadata: Structured Information to describe an object

Object: Self-contained archive file with markup pages, resources and metadata

Orthogonality: Functionality of a particular type is contained in only one type of Object

Planning Templates: The structure of teacher planning information

Product: Automatically generated Web pages

Repository: Location for storage and retrieval of Objects, Plans and Teacher Materials

Reusability: The object can be easily used or incorporated into an existing learning environment

Reusable Learning Objects/ Reusable Information Object (RLO/RIO) – structure of training materials used in the CISCO training manuals

Simple API for XML (SAX) events: Java Interface for the transfer of information between servlets.

Sharability: Sufficient information is provided for confirmation of validity and allows use.

SGML: Standard Graphical Markup Language, the printing industry standard for document printing.

Template: Empty markup page, may be used for planning, FLOs or KTTs

TML: Tutorial Markup Language

XML: Extensible Markup Language

XSL: Extensible Scripting Language

APPENDIX B

SYSTEM DESIGN DOCUMENT FOR TULIP

Introduction

With the current demands on teachers to use and integrate technology into the classroom, the need for a lesson planning tool becomes apparent.

Purpose of the system

The purpose of the Tulip tool is to provide an easy-to-use Teacher Lesson Planning tool that facilitates the development of a lesson plan. From this information, the tool automates the generation of information that needs to be distributed by the teacher to audiences of interest as required. The lesson plan may be made of smaller parts, such as objectives, or teaching tasks. The lesson plan is stored in a structured format that facilitates sharing and searching for the information. The data structure lends itself to a variety of display and cataloging options for use.

Design Goals

1. Web-based, independent of user computer operating system
2. Simple user interface to minimize technological burden of a user
3. Automatic generation of products from structured information from a stored lesson plan
4. Easy storage and retrieval of stored information with the use of XML based language.

Definitions, acronyms, and abbreviations

FLO - Fundamental Learning Objects - a set of classes of instructional objects with distinct attributes and functionality

KTT - Knowledge type templates - a set of objects combined to meet a teaching objective

LEAP - Markup language containing the elements, logic sheets and style sheets to structure the lesson planning information, and generate the required products.

Lesson Planning Form - A guided template to facilitate preparation of a lesson.

References

Lesson Planning Procedure - AskEric.org

Cocoon2 documentation - xml.apache.org/cocoon

XML specification - W3C.org

FLO, KTT, LEAP - ww2.cs.fsu.edu:8080/cocoon/reed/

Proposed Architecture

- **Overview**

  Providing a web-based tool requires a 24/7 webserver, connections to the Internet and the software to drive the application.

- **Subsystem decomposition**

1. A webserver and connection to the Internet.
2. Client device capable of receiving HTML, WML, or VoxML.
3. Server drivers and equipment for intended client used products.
4. A database server.
5. Permanent data/file storage.

- **Hardware/software mapping**
    1. Server system is running an Apache webserver, a Tomcat java servlet engine and Cocoon2 components, allowing for the use of Web forms using scripting language and servlet to process information.
    2. The database server runs ODBC driver and Database administration software to allow for the SQL queries from Cocoon Servlet.
    3. The hardware to support network protocol, including network card driver, NIC and TCP/IP.
    4. Client needs web browser software that allows the submission of forms.
    5. The File server runs the software for the organization and indexing of the data storage.
- **Persistent data management**
    1. A database interfaced with Cocoon using SQL enables the storage of information in the process of lesson planning.
    2. A set of files is generated and stored in file storage at the end of the process.
    3. A database is used to maintain the repository information of the generated files.
    4. File archives are used to package dependent files.

Access control and security

Cocoon components allow web-based session control and controlled access through login forms based on password system using hash tables.

Boundary conditions

1. Webserver, database administration and network may be set up to start automatically at boot up.
2. Services will be lost if the web server, the servlet engine, or the Cocoon2 servlet is not running or accepting requests from the Web. Mirror systems may be necessary.
3. Service will be discontinued if the network providing access to the Web is down.

Subsystems services

Cocoon2 Services, including components to perform XML parsing to SAX events, XSL transformations, portal, database and generation of products.

REFERENCES

"Abbreviations in e-learning." Seminar in University Teaching. Carleton University. 12 Oct. 2002, <http://www.carleton.ca/sut/hightech_Acronymns_and_References.htm>.

ACENET. "More College Freshmen Report Disabilities." 17 Jan. 2000. Vol. 49, No. 1, with excerpts from American Council on Education (ACE), "College Freshmen with Disabilities: A Biennial Statistical Profile" <www.heath-resource-center.org>.

ADL, Inc. "Advanced Distributed Learning." 23 Nov. 2002 <http://www.adlnet.org/index.cfm>.

ADL, Inc. "SCORM Course and Tools." 13 Mar. 2002 <http://www.scorm.tamucc.edu/>.

APACHE. 20 Nov. 2002 <xml.apache.org>.

AskERIC “Write-a-lesson Plan Guide.” Dept. of Instructional Design, Development and Evaluation at Syracuse University. 30 Jan. 2002 <http://www.askeric.org/virtual/lessons/guide.shtml>.

Baylor, Amy. "MIMIC: Multiple Intelligent Mentors Instructing Collaboratively: An Intelligent Agent-based Learning Environment for Learning Instructional Design." Third International Cognitive Technology Conference, CT’99. Aug. 1999 <http://www.cogtech.org/CT99/Baylor.htm>.

Bonharme, Eric. "HCI Usability Evaluation Techniques." Napier University. 30 Apr. 1996 <http://www.dcs.napier.ac.uk/marble/Usability/Evaluation.html>.

Bowen, Jonathan. “Internet: Portals and e-commerce.” 4 Aug. 2002 <http://forums.museophile.net/story/2002/8/4/155237/2125>.

Brickley, Dan. "TML Version 4” and “Towards an Open Question-interchange Framework." 12 Jul. 1996 <http://www.ilrt.bris.ac.uk/netquest/about/lang/motivation.html>.

Bruckman, Amy. "The Future of e-Learning Communities." Communications of the ACM. Vol. 45, No. 4, 2002. Association of Computing Machinery. Apr. 2002.

Brusilovsky, Peter, Elmar Schwarz and Gerhard Weber. “Electronic Textbooks on WWW: From Static Hypertext to Interactivity and Adaptivity.” Chapter 30, Web-based Instruction, Khan, Badrul H., Ed., Educational Technology Publications, 1997.

“CanCore: Canadian Core Learning Resource Metadata Application Profile.” 17 Apr. 2002 <http://www.cancore.ca/>.

"CAREO" “Learning Commons” Universities of Alberta and Calgary, 2 Dec. 2002 <http://careo.netera.ca/cgi-bin/WebObjects/Repository>.

Cisco Systems, Inc. "Reusable Learning Objects Strategy." V3.1. April, 2000.

Clark, Don. "Learning Domains, or Bloom's Taxonomy: Three Types of Learning." 24 May 2002 <http://www.nwlink.com/~donclark/hrd/bloom.html>.

"Cocoon2." Apache Foundation. 23 May 2002 <http://xml.apache.org/cocoon/cocoon-links.html>.

"Cocoon2 Project." Apache Foundation. 23 May 2002 < http://xml.apache.org>.

Cole, T., T. Habing, D. Hillmann, J. Hunter, P. Johnston, C. Lagoze, A. Powell. “Dublin Core.” Recommendations for XML Schema for Qualified Dublin Core, Proposal to DC Architecture Working Group. 14 Jul 2002.

"Cover Pages", hosted by Organization for the Advancement of Structured Information Systems (OASIS), 4 Dec. 2002 <http://www.oasis-open.org/cover/>

Daniels, Harold Lee. "Gagne and Learning." 24 May 2002 <http://coe.etsu.edu/departments/cuai/danielsh/5507/aids/gagne.htm>.

Datacomm, "Portals to Profit: E-Commerce Business Models and Enabling Technologies." Datacomm Research Company's Report. 20 Nov. 2002 <http://www.igigroup.com/st/pages/portalupdate.html>.

Dick, W., L. Carey. The systematic design of instruction. Addison-Wesley Longman. 1996.

Downes, Stephen. “The Learning Object Economy.” Draft, 5 Aug. 2002.

Downes, Stephen. “Learning Objects: Resources for Distance Education Worldwide.” International Review of Research in Open and Distance Learning 2001, 2, 1 <http://www.icaap.org/iuicode?149.2.1.2>.

Downes, Stephen. "MarcoPolo." The Technology Source. 2000, July/August. 26 May 2002 <http://horizon.unc.edu/TS/default.asp?show=article&id=792>.

EOE, "Educational Object Economy Foundation." Home Page. 3 Dec. 2002 <http://www.eoe.org>.

"Educational Software Components of Tomorrow." Home Page. SRI International. 2001 <http://www.escot.org/>.

ERIC. "The Educational Resources Information Center (ERIC)." U.S. Department of Education's Office of Educational Research and Improvement, administered by the National Library of Education (NLE). 21 May 2002 <http://www.eric.ed.gov/>.

ESRC Research Programme in Cognitive Engineering, "Multimedia, Education and Narrative Organisation (MENO)", 1998, <http://meno.open.ac.uk/>.

European Space Agency (ESA) 2002 <http://www.esrin.esa.it/export/esaCP/index.html>.

"Explorations in Learning & Instruction." The Theory into Practice Database, Encyclopedia of Psychology. <http://tip.psychology.org/>.

Florida Information Resource Network (FIRN). Home Page. 21 May 2001 <http://www.firn.edu/>.

"Glossary of Teaching Strategies," University of Idaho, 24 May 2002 <http://ivc.uidaho.edu/mod/models/taba/glossary.html>.

Gibbons, Andrew S., " The Nature and Origin of Instructional Objects", 2001.<http://www.reusability.org/read/chapters/gibbons.doc>.

Grabowski, B., M. McCarthy and T. Koscalka. "Web-based Instruction and Learning: Analysis and Needs Assessment." NASA/TR-1998-206547, 1998.

Greenagel, Frank L. "The Illusion of E-learning: Why We are Missing Out on the Promise of IP Technology." 31 July 2002 <http://www.elearningmag.com/elearning/article/articleDetail.jsp?id=26850>.

"HEML, an Historical and Event Markup Language." 8 June 2002 <http://heml.mta.ca/heml-cocoon/site/heml-description.html>.

Hill, Tom. "Learning Object Metadata Framework." 19 Nov. 1997 <http://www.geocities.com/ResearchTriangle/6568/metadata.htm>.

Horton, Sarah. "Practical Accessibility." Dartmouth College. 30 Sep. 2001 <http://www.dartmouth.edu/~webteach/articles/access.html>.

IBM. "Open Document." 30 Sep. 2002 <http://www-105.ibm.com/developerworks/education.nsf/dw/xml-onlinecourse-bytitle?OpenDocument&Count=500>.

ICPT. “Florida’s Interdisciplinary Curriculum Planning Tool.” 2000. Florida Department of Education. 2002 <http://www.firn.edu/doe/curric/prek12/ecpt.htm>.

Institute of Electrical and Electronics Engineers, Inc. (IEEE). "LTSC Mission, Working and Study Groups." 13 May 2002 <http://ltsc.ieee.org/>.

IEEE, "The Mission of IEEE Learning Technology Standards Committee (LTSC) Working Groups." 13 May 2002 <http://xml.coverpages.org/ieeeLTSC.html>.

IEEE, P1484.20, Competency Definitions Work Group. "Draft Competency Definitions." 13 May 2002 <http://ltsc.ieee.org/>.

IEEE, P1484.3/D3. "Draft Standard for Information Technology -- Learning Technology – Glossary." 9 Mar. 2001 <http://ltsc.ieee.org/>.

IMSGlobal Learning Consortium, Inc. 23 May 2002 <http://www.imsproject.org/metadata>.

INT Media Group. "Generating Web Content with Cocoon." 2002 <http://www.webreference.com/xml/column52/>.

International Society for Technology in Education (ISTE), Inc. Home Page. <http://www.iste.org >.

ISTE. "National Education Technology Standards." <http://www.iste.org>.

ISTE. "Preparing Tomorrow’s Teachers to use Technology (PT3) program." <http://www.iste.org/research/index.html>.

KangaX. "KangaX, a secure XML Publishing tool." May 2002 <http://www.kangax.com/client/4/home.html>.

Kaye, H. S. "Computer and Internet Use Among People with Disabilities." Disability Statistics Report (13). Washington, DC: U.S. Department of Education, National Institute on Disability and Rehabilitation Research. Mar. 2000 <http://www.dsc.ucsf.edu/UCSF/pdf/REPORT13.pdf>.

Klassen, Prescott, John Maxwell and Solvig Norman. “Structured Information and Course Development: An SGML/XML Framework for Open Learning.” EdMedia99, pages. 20 Nov. 2002 < http://www.merlin.ubc.ca/people/Jmax/Misc/EdMedia99-1910.pdf>.

“Latis On-Line”, “Latis On-line Services: Curriculum Framework Planning Tool” and “Latis Portal.” Australian Department of Employment Education and Training, GPO Box 4821, Darwin, NT 0801, Australia. 25 Oct. 2002 <http://www.latis.net.au/ols/cfptool.htm>.

Learnativity. “Learning Objects and Standards Resources.” 25 Nov. 2002 <http://www.learnativity.com/standresources.html>.

“Learning Material Markup Language (LMML).” 31 July 2000 <http://www.oasis-open.org/cover/lmml.html>.

"Learning Objects." University of Wisconsin at Milwaukee. 20 Nov. 2002 <http://www.uwm.edu/Dept/CIE/AOP/LO_what.html>.

LinkToLearn. “Technology Literacy Challenge Fund.” State of Pennsylvania. 2002 <http://www.l2l.org/tlcf/>.

MathStar. "Rubric for Evaluating Unit Plans." New Mexico State University. 24 June 2002 <http://mathstar.nmsu.edu/exploration1/unit/unit_rubric.html>.

Mayer, R. E. Multimedia Learning. New York: Cambridge University Press, 2001.

Mazzocchi, Stefano. “Introducing Cocoon 2.0.” 13 Feb. 2002. O'Reilly's XML.com <http://www.xml.com/pub/a/2002/02/13/cocoon2.html>.

Merrill, M. D. "Instructional Transaction Theory: Instructional design based on knowledge objects." In C. M. Reigeluth (Ed.), Instructional Design Theories and Models: A New Paradigm of Instructional Theory (pp 397 - 424), Hildale, N.J: Lawrence Erlbaum Associates, 1999.

Merrill, M. David. "Knowledge Objects." CBT Solutions. Mar/Apr 1998.

Microsoft Corporation. “.Net.” 2002 < http://www.microsoft.com/net/>.

Microsoft Corporation, “Inside the .NET Framework,” Module 02.4, Visual Studio.NET Training Tour, 2001.

Moreno, R., R.E Mayer, H. A Spires and J.C. Lester (2001). "The case for social agency in computer-based teaching: Do students learn more deeply when they interact with animated pedagogical agents?" Cognition & Instruction, 2001, 19(2), pp 177-213.

Murray, T., J. Piemonte, S. Khan, T. Shen and C. Condit. "Evaluating the Need for Intelligence in an Adaptive Hypermedia System" in Proceedings of ITS-2000. Montreal, Quebec, Canada, June 2000.

Murray, Thomas. "Knowledge types for ITS design", excerpt from "Facilitating Teacher Participation in Intelligent Tutor Design", Dissertation, 1992.

National Oceanographic and Atmospheric Administration (NOAA). "Assistive Technology News." 6 June 2002 <http://www.hpcc.noaa.gov/access/technews.htm>.

"NetQuest." 12 Dec 1995. Institute for Learning and Research Technology at the University of Bristol. <http://www.ilrt.bris.ac.uk/netquest/liveserver/TML_INSTALL/doc/tml_user.html>.

Nilsen, Erik. "HCI and the Web." CHI 96 Workshop, 1996. 24 May 2001 <http://www.acm.org/sigchi/web/chi96workshop/papers/nilsen.html>.

Norris, Cathleen, Elliot Soloway and Terry Sullivan. “Examining 25 years of Technology in U.S. Education.” Communications of the ACM, 20 Aug. 2002, vol. 45, No. 8, pp 15-18.

Ohio Schoolnet. “Lesson planning template.” 30 May 2001 <http://tlcf.osn.state.oh.us/blueprint/index.html>.

OLA (Open Learning Agency). “The Open Learning Agency." Home Page. 13 May 2002 <http://www.ola.bc.ca/>.

OLA/OSP. "Open School Project." Open Learning Agency, 2001. 13 May 2002 <http://www.openschool.bc.ca>.

Open Directory Project. "List of XML Markup Languages." 4 Jan. 2002 <http://dmoz.org/Computers/Data_Formats/Markup_Languages/XML/>.

Open Directory Project. “The Open Directory Project." Netscape. 2002 <http://dmoz.org/about.html>.

PageBox, "Cocoon Support / Soap Publishing Framework." 4 Dec. 2002 <http://pagebox.net/cocoonSupport.htm>.

Paille, G., S. Norman, P. Klassen and J. Maxwell. "The effect of using structured documents (SGML) in instructional design." In NAWeb Conference Proceedings, 1999. 20 Nov. 2002 <http://naweb.unb.ca/proceedings/1999/paille/paille.html>.

PaKMaS. "Learning Materials Markup Language (LMML)." 8 Apr. 2002 <http://daisy.fmi.uni-passau.de/projects/PaKMaS/LM2L/>.

Park, Ok-choon, “Adaptive Instructional Systems, Chapter 22” of Soft Technologies: Instructional and Informational Design Research, Handbook of Research for Educational Communications and Technology, Jonassen, D. H. (Ed.), Association for Educational Communications and Technology (AECT), Simon and Schuster © 1996.

"Project Gutenberg." Home Page. 6 Aug. 2001 <http://promo.net/pg/>.

Punte, Steve. “Getting Started with Cocoon 2.” 10 July 2002 <http://www.xml.com/pub/a/2002/07/10/cocoon2.html>.

QED (Quality Education Data, Inc.) “Quality Education Data (QED) Releases Internet Usage Report of K-12 Public Schools.” 18 Oct. 2000 <http://www.qeddata.com/iups_pr.htm>.

Roschelle, J. and J. Kaput. "Educational Software Architecture and Systemic Impact: The Promise of Component Software." Journal of Educational Computing Research, 14(3), pp 217-228. 7 Oct. 1998 <http://www-jime.open.ac.uk/98/6/roschelle-01.html>.

Siemens, George. “Learning Objects.” Red River College, Winnipeg, Manitoba, Canada. 20 Oct. 2002 <http://www.elearnspace.org/learningobjects.htm>.

Siirtola, H. and T. Heimonen. “Scalable Support for Work Groups and Groupwork” Poster presented at Mobile HCI 2001: Third International Workshop on Human Computer Interaction with Mobile Devices, IHM-HCI 2001. Lille, France. Sep. 2001 <http://www.cs.uta.fi/hci/leco/publications.html>.

Slowinski, Joseph. “Workforce Literacy in an Information Age: Policy Recommendations for Developing an Equitable High-Tech Skills Workforce.” First Monday, volume 5, number 7. July 2000 <http://firstmonday.org/issues/issue5_7/slowinski/index.html>.

Spohrer, Jim, Tamara Sumner and Simon Buckingham Shum. "Educational Authoring Tools and the Educational Object Economy: Introduction to the Special Issue from the East/West Group." Journal of Interactive Media in Education, 98(10). 20 Oct. 1998 <http://www-jime.open.ac.uk/98/10/spohrer-98-10.pdf>.

STEPS. "Steps and the Lesson Architect." University of West Florida, Office of Educator Performance. 2001 <http://www.ibinder.uwf.edu/Steps/>.

"SunBankPortal." 8 June 2002 <http://sunshine.sundn.de/sunshine/sunbankportal>.

Süß, Christian, Rudolf Kammerl and Burkhard Freitag. “A TeachWare Management Framework for Multiple Teaching Strategies.” Proceedings ED-MEDIA 2000, World Conference on Educational Multimedia, Hypermedia & Telecommunications. Montreal, Quebec, Canada. 2000 <http://hdf.ncsa.uiuc.edu/Parallel_HDF/hdf5_doc/v1_2/doc/html/XML_DTD.html>.

TEAC (Teacher Education Accreditation Council). Home Page. 20 Nov. 2002 <http://teac.org/>.

TECFA. "Cocoon Pointers." 12 Feb. 2002 <http://tecfa.unige.ch/guides/xml/cocoon-pointers.html>.

ThinkQuest, Inc. "Teaching Teachers." 2 July 2002 <http://thinkquest.org/> and <http://t3.thinkquest.org/>.

Thompson, Brooks/Cole Publishers, PWS, “The Analytical Engine On-Line," 1998 <www.pws.com/aeonline.html>.

“Tutorial Markup Language (TML) Version 4.” 3 Dec. 2002 <http://www.ilrt.bris.ac.uk/netquest/about/lang/>.

U.S. Department of Education, "Condition of Education 2002", Pub 2002-025, Washington, DC., June 2002.

U.S. Department of Education, National Center for Education Statistics. “Beyond School Level Internet Access: Support for Instructional Use of Technology." Pub 2002-029, Washington, DC, pages. April 2002 <http://nces.ed.gov/pubs2002/2002029.pdf>.

U.S. Department of Education, National Center for Education Statistics. "Internet Access in U.S. Public Schools and Classrooms: 1994-2000." Pub 2001-071, Washington, DC, pages. 2001 <http://nces.ed.gov/pubs2001/2001071.pdf>.

U.S. Department of Education, National Center for Education Statistics. "Teacher Use of the Internet in Public Schools." Pub 2000-090, Washington, DC, pages. 2000 <http://nces.ed.gov/pubs2000/2000090.pdf>.

U.S. Department of Education, National Center for Education Statistics. "Teacher’s Tools for the 21st Century: A Report on Teacher's Use of Technology." Pub 2000-102, Washington, DC, pages. April 2000 <http://nces.ed.gov/pubs2000/2000102.pdf>.

U.S. Department of Education, Office of Educational Technology. "National Educational Technology Goals." Excerpt from eLearning: Putting a World-Class Education at the Fingertips of All Students. December, 2000 <http://www.ed.gov/Technology/elearning/index.html>.

U.S. Department of Education, Office of Special Education and Rehabilitative Services, IDEA General Information: “Overview”, 1997.

U.S. Department of Education. "Preparing Tomorrow's Teachers to Use Technology (PT3) program." <http://www.pt3.org/>.

US Public Law 101-336. "Americans with Disabilities Act (ADA) of 1990." Washington, DC. 26 July, 1990 <http://www.access-board.gov/about/ADA%20Text.htm>.

US Public Law 105-17. "Individuals with Disabilities Education Act (IDEA) Amendments of 1997." Washington, DC, pages. 1997 <http://www.ed.gov/offices/OSERS/Policy/IDEA/IDEA.pdf>.

US Public Law 105-220, Section 508. "Workforce Investment Act of 1998." Washington, DC, pages. 1988 <http://www.usdoj.gov/crt/508/508law.pdf>.

US Public Law 107-110. "No Child Left Behind Act of 2001." Washington, DC, pages. 8 Jan. 2002 < http://www.ed.gov/legislation/ESEA02/107-110.pdf>.

W3C (World Wide Web Consortium). Home Page. 2002 <http://www.w3c.org>.

W3C. Guide to the W3C XML Specification ("XMLspec"). Document Type Definition (DTD) Version 2.1. 26 May 2002 <http://www.w3c.org/XML/1998/06/xmlspec-report-v21.htm>.

"WebQuest." San Diego State University. 2 July 2002 <http://webquest.sdsu.edu/webquest.html>.

"WebQuest Training Materials." San Diego State University, Educational Technology Department. 2 July 2002 <http://webquest.sdsu.edu/materials.htm>.

Welsch, Edward. "SCORM: Clarity or Calamity?" Online Learning Magazine. Aug. 2002 <http://www.onlinelearningmag.com/onlinelearning/magazine/article_display.jsp?vnu_content_id=1526769>.

Wenger, E., Artificial Intelligence and Tutoring Systems: Computational and Cognitive Approaches to the Communication of Knowledge, Los Altos, CA: Morgan Kaufmann Publishers, Inc., 1987.

Wiley, D. A., "Connecting Learning Objects to Instructional Design Theory: A Definition, a Metaphor, and a Taxonomy", in D. A. Wiley (Ed.), The Instructional Use of Learning Objects, Bloomington, IN: Association for Educational Communications and Technology, 2001.

Wiley, D. A., Ed. The Instructional Use of Learning Objects. A joint publication of the Agency for Instructional Technology and the Association for Educational Communications and Technology. 2002 <http://www.reusability.org/read/>.

Wiley, D. A. "Learning Objects Need Instructional Design Theory." In A. Rossett (Ed.) The 2001/2002 ASTD Distance Learning Yearbook. New York: McGraw-Hill, pages. 2001 <http://wiley.ed.usu.edu/docs/astd.pdf>.

Wiley, D. A. "Learning Objects." In A. Kovalchick & K. Dawson (Eds.), Educational Technology: An Encyclopedia. Santa Barbara: ABC-CLIO, pages. 2001 <http://wiley.ed.usu.edu/docs/encyc.pdf>.

William, C., "Internet Access in U.S. Public Schools and Classrooms: 1994-1999" (NCES 2000-086), U.S. Department of Education, Washington, DC, National Center for Education Statistics, 2000.

Wilson, Scott. "ADL - New Sequencing specification imminent - Open Source tools to follow." The Centre for Educational Technology Interoperability Standards (CETIS). 3 May 2002 <http://www.cetis.ac.uk/content/20020503152933>.

WorldCom, Inc. "MarcoPolo." 11 Nov. 2002 <http://marcopolo.worldcom.com>.

WorldCom, Inc. "MarcoPolo | Teacher Training Kit: downloads." 2002 <http://marcopolo.worldcom.com/trdownload.shtml>.

XML.org. Organization for the Advancement of Structured Information Systems (OASIS). 2002 < http://www.xml.org/xml/aboutxml.shtml>.

BIOGRAPHICAL SKETCH

R. Gabrielle Reed has 8 years experience as an instructor, including 4 years of teaching Computer Science at the college level

R. Gabrielle Reed has a goal of providing high quality educational materials and learning experiences to the students of the world. She was the recipient of private education at Bishop Moore High School in Orlando, Fl., through the sacrifice of her parents, Robert and Julia Reed. She also received academic and financial need scholarships from Rollins College in Winter Park, Fl., along with the Florida State Assistance Grant and the US Pell Grant. The value of this education became apparent during her 4 years of teaching college students. Many of the students were not as well prepared to handle the challenges of college due to simple lack of learning, study, and problem solving skills. The lack of interest in learning was pervasive. What could kill the desire to learn that seems to be innate in early childhood? Many of these seemed to be children that had fallen through the cracks of the education system. Gabrielle's lifetime research goal is to tutor the world, by using combinations of intelligent tutoring systems, multimedia and animated interface agents to help stimulate interest and desire to learn.

R. Gabrielle Reed has a Masters of Science in Physics from Florida State University and worked a number of years in the fields of health and medical physics. She worked as a computer professional in the Fl. Dept. of Education for three years before returning to school to work toward a Master's degree in Computer Science at Florida State University.