AlcoZone: An Adaptive Hypermedia Based Personalized Alcohol Education

Devdutta Bhosale

Thesis submitted to the faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Engineering

Dr. Sandeep Shukla, Chair
Dr. Edward A. Fox, Co-Chair
Dr. Thomas L. Martin

May 8, 2006
Blacksburg, Virginia

Keywords: Adaptive hypermedia, usability, e-learning, alcohol education

AlcoZone: An Adaptive Hypermedia Based Personalized Alcohol Education

Devdutta Bhosale

(ABSTRACT)

In our knowledge based economy, demand for better and effective learning has led to innovative instructional technologies. However, the one-size-fit-all approach taken by many e-Learning systems is not adequate to the different requirements of people who have different goals, preferences, and previous knowledge about a subject. Many e-Learning systems have approached this problem with personalized and customized content. However, many of these systems are closely tied to one particular subject that they are trying to teach; authoring of courses on different subjects using the same framework is a difficult process. Adaptive Hypermedia is an approach in which content presentation and navigation assistance is personalized depending on the requirements of the user. The user requirements are represented using a user model, while the content is represented using a content model. By using a set of algorithms, an Adaptive Hypermedia based system is able to select the most appropriate content to be presented, as the user interacts with the system. The objective of AlcoZone is to educate all of the 5,000 freshman students of Virginia Tech about alcohol education using Adaptive Hypermedia technology, as part of the mandatory university requirement. The course presents different content to different students based on their drinking pattern. AlcoZone integrates Curriculum Sequencing, Multimedia and Interactivity, Alternate Content Explanation, and Navigational Assistance to make the course interesting for students. This research investigates the design & implementation of AlcoZone and its Adaptive Hypermedia based reusable framework for course creation and delivery.

To My Parents,

Thank you for all your help and support

Acknowledgements

This thesis has been the most rewarding experience of my academic life and could not have been accomplished without support from a variety of individuals.

I would like to thank Steven Clarke for his inspiration and patience and for his encouragement in the creation of alcohol education courses for Virginia Tech students.
I would like to thank my advisors: Dr. Sandeep Shukla and Dr. Edward Fox, for guiding me in this effort. I would like to thank Dr. Shukla for being very helpful and considerate and for agreeing to serve as the chair on my committee. I would like to thank Dr. Fox for helping me in making the key technological decisions and for his reassurance.

I am thankful to Dr. Martin for serving on my thesis committee.

Special thanks go to Veena Basavraj for her help in this project. My thanks also go to all the members of College Alcohol Abuse Prevention Center: Kashmira, and Lorianne. I would like to acknowledge the support provided by my friends: Abhijeet, Anant, Aditya, Harshit, and Bharati.

I am grateful to my parents and sister, Jayashree, for their love, support and inspiration throughout my life. And finally, thanks go to all those people who have helped me finish my graduate studies and who contributed to the wonderful time that I have had at Virginia Tech.

Table of Contents

List of Figures

Chapter 2

Chapter 3

Chapter 4

Chapter 5

Chapter 6

List of Tables

Chapter 1: Introduction

In this chapter, I present an overview of AlcoZone and its underlying technologies. Some of the background of AlcoZone is then presented, including its history and evolution. Next, the contributions of this thesis in the field of Adaptive Hypermedia based E-Learning systems are presented. Finally, an outline for the rest of this thesis is presented.

1.1 Overview

E-Learning is now considered to be one of the growing research areas in the application of multimedia computing. And the scope of research in E-Learning is expanding beyond the use of multimedia technologies to deliver content on the web. In fact, in some situations, E-Learning has been shown to be more effective than classroom learning [4]. AlcoZone is a personalized online alcohol education class for Virginia Tech students based on Adaptive Hypermedia technology. In Adaptive Hypermedia, a model of user goals, knowledge, and preferences is built, and it is used throughout the interaction. Using Adaptive Hypermedia, it is possible to present the most appropriate explanation of the same content to a different category of users and to provide navigational assistance in the hyperspace by using link adaptation techniques. Our modified architecture based on AHA [2] allows us to separate the adaptation aspects from the content. Thus, the extended framework is not closely tied to alcohol education and can be used to develop courses in other subject areas. The course integrates questionnaires, interactive exercises, images, audio narration, and text. This creates a true multimedia experience and makes it engaging for learners, while at the same time, provides valuable alcohol educational information. Authors of adaptive content create a high level representation of learning objects using a three level hierarchy of chapters, lessons, and concepts which allows logical separation of content. This high level description includes information about how to adapt the content, and provides navigational assistance in the hyperspace. The actual content to be presented is created using HTML or XHTML files. This separation makes it easier to include interactive elements based on Flash or SMIL content and streaming audio/video. Our extension further separates the content which is part of HTML or

XHMTL file from its rendering. The HTML/XHTML pages include only the textual information and pointers to images and interactive elements. The look and feel of the pages is controlled using the abstraction of 'skins'. The 'skins' functionality makes it easy to change the visual aspects of course layout and colors on the fly. The system provides extensive logging capabilities which log all the answers to questions, concepts visited, and interaction of the user with the system, along with timing information. The logged information analysis enables authors and administrators to understand course usage and student performance, and helps to improve the system.

1.2 History of AlcoZone

AlcoZone originated with the idea of providing personalized, customized, and adaptive alcohol education to thousands of freshman students who start their college at Virginia Tech every year. The first prototype was created and evaluated in Spring 2004. Though the project was launched to aid alcohol education, it was realized that the framework developed should not be tied to alcohol education; creation of courses in other subjects should be possible using the same framework. Many web-based educational systems have been developed that include a list of static content which is the same for everyone. However, people who take online classes come from diverse backgrounds with different knowledge and goals. Content that is interesting for one user may not be engaging for another [2]. The first prototype [24] did include personalized and customized content; however, it was not based on Adaptive Hypermedia technology. It was essentially a rule-based system in which content personalization rules were integrated as part of the source code. The prototype contained mostly sequential textual information with few images and did not provide a high level of interactivity. After the evaluation of the prototype, objectives for the first version were refined and broadened; these are described in Chapter 2. Work on the first version of AlcoZone began in Fall 2004, which consisted of requirements gathering, and technology & tools assessment. AHA [1] (Adaptive Hypermedia for All) was found to be the most suitable tool to build upon because of its open source nature, and flexibility to support the extensions that were identified as part of the requirements gathering process. AHA integrates both the

authoring tools and the adaptation engine. However, most of our modifications in AHA involved modification of the adaptation engine, since the ability to present the adapted content in the most suitable form was considered a priority.

1.3 Thesis Contributions

In this section some of the contributions of this thesis in E-Learning and Adaptive Hypermedia have been highlighted. They will be explored in detail throughout the thesis but are provided as a convenience for the reader.

● AlcoZone is an adaptive educational hypermedia based system which has been designed by modification of a research tool, to be taken by thousands of university students. It integrates features and interactivity found in commercial E-Learning solutions.

● AlcoZone includes curriculum sequencing (i.e., the ability to visit content in sequence) which makes it easier for users to decide what content to visit next.

● Delivery of static HTML pages that do not relate to concepts is possible. This is useful for providing many system features (e.g., skins, audio) that do not interact with higher level representations of content and do not change from one course to another.

● The extended framework supports out of order conditional jumps from one concept to another depending on the user response to questions.

● Hierarchical arrangement of content in the form chapters, lessons, and concepts makes it easy for authors to organize information logically and for learners to avoid confusion.

●The extensive logging support makes it possible to log every aspect of the student interaction with the system. This can be used to predict common learning patterns and correlation between user characteristics.

● AlcoZone makes to possible to change the look and feel of the course on the fly.

● Support for audio narration has been provided.

1.4 Thesis Organization

This thesis is organized as follows. Chapter 2 describes the history of AlcoZone, the first prototype, and its shortcomings. In Chapter 3 we discuss the underlying technologies behind AlcoZone. Chapter 4 describes the architecture and design of AHA, the tool on which AlcoZone has been built. Chapter 5 describes the design and implementation of our extensions. Chapter 6 discusses the evaluation of the developed system. Finally, in Chapter 7, we present the conclusions and summary of this research effort and future work which could be done to augment the system.

Chapter 2: First Prototype

This chapter describes the design of the first prototype of AlcoZone. The prototype has influenced our efforts in the use of an Adaptive Hypermedia based reusable framework. We discuss how the AlcoZone prototype was conceived, as well as the design of the prototype and some of its limitations which led to the development of the current version.

The key requirement in the development of the prototype was the ability to customize content based on the user response to a questionnaire, and ability to customize the user interface as per users' preferences. Using answers to a series of questions, the system classified the user in one of the five drinker categories:

- Abstainer: People in this group do not consume alcohol.
- Occasional: People in this group do not drink more than a certain number of drinks in a month in a social setting or drink only as part of religious activities.
- Low Risk: Low risk drinkers have an upper bound on both the average frequency of consumption and quantity of alcohol consumed in one sitting.
- At Risk: At risk drinkers consume more often than low risk drinkers but less than high risk drinkers.
- High Risk: High risk drinkers consume more than a certain number of drinks (eight for men and six for women) in a sitting.

Teaching objectives for different categories of users were:

Abstainers and Occasional Drinkers: Abstainers and occasional drinkers are presented with materials that they can use to help their peers with drinking problems.

Low-risk Drinkers: Low-risk drinkers do not have a drinking problem but it is important that they stay in their current drinking zone. Content for low-risk users helps them to stay in their current drinking zone.

At-Risk and High-Risk Drinkers: At-risk and high-risk drinkers have a serious drinking problem. Content for these drinkers mainly helps them overcome their drinking problem.

The first step in the development process was to assess the technological feasibility and the user response to online personalized alcohol education. The architecture of the system was defined based on Java servlets and JSPs. There were a series of questions asked in the beginning of the course, answers to which determined what content would be presented to the user. Let us look in detail at the architecture and the system flow of the prototype, as can be seen in Figure 2.1.

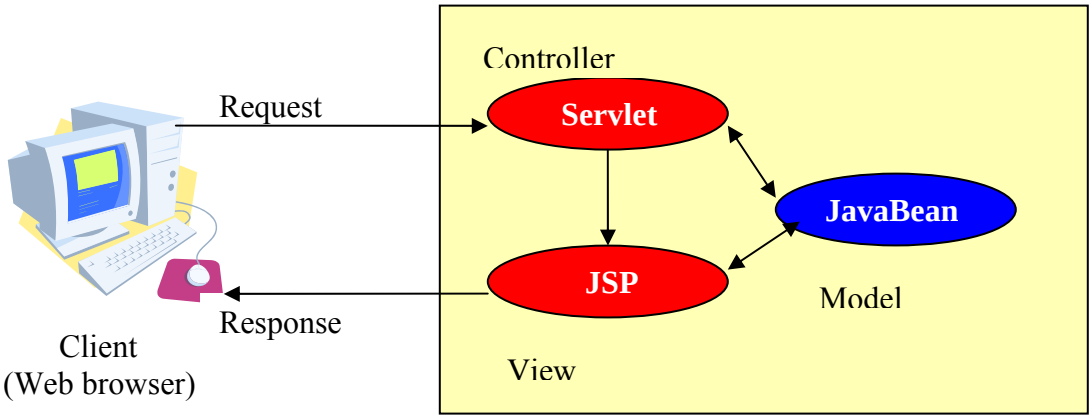


Figure 2.1: Architecture of AlcoZone Prototype [with permission from the author, 24]

The AlcoZone prototype was developed based on the model-2 architecture. This is a hybrid approach to serving dynamic content which uses both JSPs and servlets. Here, the servlet acts as the controller servlet and all user requests are sent to the controller servlet. The servlet is responsible for creation of JavaBeans which are used to pass the data to the JSPs. The JSPs handle the presentation logic and process the JavaBeans to generate dynamic presentation code. Having taken a high level look at the architecture of the prototype, in the next section we shall see the system flow model.

2.1 System Flow of Prototype

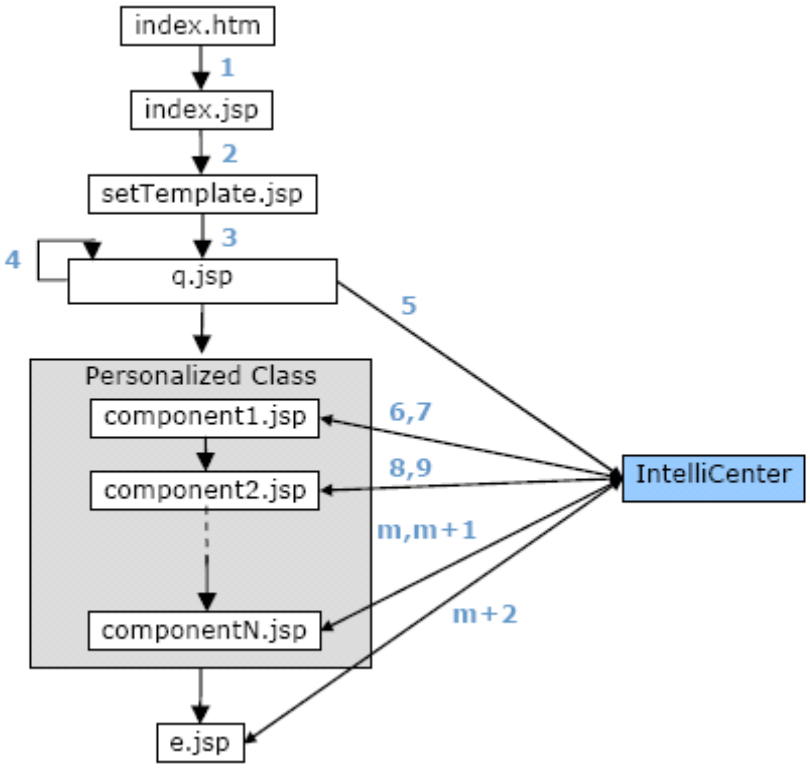


Figure 2.2: System Flow of AlcoZone [with permission from the author, 24]

The system flow [24] starts with index.htm page, which displays the welcome screen, as per Figure 2.2. After this, the system interaction takes place in the following steps:

1. The user is presented with different user interface options to choose from. Upon selection of the interface, this information is stored in the session and the rest of the interaction uses the selected interface.
2. A series of questions are asked to the user to determine his drinking pattern and habits. This questionnaire is handled by a JSP page and the user responses are stored in a JavaBean.

3. Once the user is done with the questionnaire, the centralized (i.e., IntelliCenter) servlet is invoked which processes the responses to questions and creates a FeedBack JavaBean.

4. The servlet also selects the content components that are appropriate for the user.

5. The rest of the interaction involves the presentation of the selected components in series. These components (which are implemented in the form of JSPs) can access the FeebBack JavaBean to personalize content for each user.

2.2 Level of Content Personalization

The prototype supported three levels of content personalization, which are:

1. Calculations

In this personalization, calculations are performed on the questionnaire data submitted by the user and appropriate feedback is provided. The calculations are performed for each user.

2. Personalization of content inside a component

This type of personalization involves decision making, e.g.,
IF (X) THEN ABC ELSE XYZ

As we shall see in the next chapter, this approach is similar to conditional fragment selection, which selects a fragment within a page depending on the result of condition evaluation.

3. Selection of components

This kind of personalization involves selection of educational components for the customized education class. Selection is mainly based on the drinker type.

After the formative and summative evaluation of the prototype [24], it was found that people liked the idea of personalized education and the ability to select user interface. However, some of the drawbacks of the initial prototype were lack of user interactivity, sequential structure of content, and difficulty of using the same system to create other courses etc.

2.3 Drawbacks of AlcoZone Prototype

● The architecture of the prototype was highly monolithic, and generation of personalized content was closely tied in with the JSP and servlet source code. The decision of personalized content generation was taken by one central servlet.

● Any change in the system involved change in this servlet code along with creation of a secondary code base in the form of JSP.

● The system did not provide flexible and separate layers of abstraction like User Model, Content Model and Adaptation Model as found in Adaptive Hypermedia systems or in those systems that evolved from the intelligent tutoring paradigm.

● Close coupling between different system elements limited the extensibility of the system to provide support for collaborative and distributed learning (e.g., discussion board), integration with course management systems (e.g., Sakai, Moodle, etc.), and other feature additions.

● The system did not offer a way to browse all of the course content and allow the learner to learn at his own pace.

● Audio content was not provided.

● The system did not offer interactivity by way of exercises and animations.

● Logging support was not provided which limited the system's ability to analyze student performance and system usage.

● Generation of personalized content depended on answers to the questionnaire asked at the beginning of the course. It was not possible to question users in the middle of the course and adapt accordingly.

Considering the limitations of the initial prototype, the objectives for the first version to be implemented by Fall 2006 were refined and broadened, so that the system developed can be used for the development of other courses as well.

2.4 Requirements for AlcoZone 1.0

1. Create personalized and adaptive alcohol education using technologies that could allow more sophisticated content adaptation than is provided by the integrated rules as in the prototype.
2. Integrate multimedia and interactive learning objects without code changes.
3. Provide content creation and delivery as separate functions.
4. Enable content presentation in a non-linear way and provide a browsable listing of content.
5. Include an integrated logging mechanism that can capture both the application usage and the user performance.
6. Enable the change in the look and feel of the course on the fly.
7. Distribute questions and exercises throughout the length of the course instead of asking them at the beginning.
8. Provide an ability to take the course in multiple sessions.

Using the feedback on the prototype and after following a semester long requirements gathering and technology assessment process, the new design and implementation started from Spring 2005. However, before deciding on the use of Adaptive Hypermedia, a few more technologies were evaluated. Considering the size of the project and time availability, use and modifications of existing tools was given a priority over developing everything from scratch. The following section describes the technological alternatives for content personalization and adaptation.

2.5 Techniques for content personalization and adaptation

Different systems have implemented different techniques for content personalization. Adaptation of content is different from content personalization. In adaptation, a model of the user preferences is used throughout the interaction between the user and the system. On the other hand, in personalization, the system may or may not use the model of the user in every interaction. Many personalization systems use simpler techniques that enable presentation of dynamic content on the web, while more sophisticated personalization systems use rules to determine the presentation of content. The AlcoZone prototype used rules to afford personalized content. However, we have seen the drawbacks of rule based adaptive systems such as the AlcoZone prototype, as discussed in the previous section. Intelligent Tutoring Systems such as ELM-ART [26] can be considered as personalized and adaptive systems. Similar to Adaptive Hypermedia based systems, ELM-ART uses different models throughout the interaction with the user. However, with the development of Adaptive Hypermedia, Intelligent Tutoring Systems have evolved into Adaptive Education Hypermedia systems [32].

Chapter 3: Adaptive Hypermedia

"*Adaptive hypermedia is a direction of research on the crossroads of hypertext (hypermedia) and User Modeling. The goal of this research is to improve the usability of hypermedia. Adaptive hypermedia systems build a model of the goals, preferences and knowledge of the individual user and use this throughout the interaction for adaptation of the hypertext to the needs of that user*" [2].

Thus, an Adaptive Hypermedia System satisfies three criteria [31]:

1. It is a hypertext or hypermedia system.
2. It uses a User Model.
3. It is able to adapt the hypermedia using the User Model.

An Adaptive Hypermedia System (AHS) is especially useful where people with different goals and knowledge are going to use the system and the hyperspace is reasonably big [31]. There are many systems which present personalized content using what the user indicates or selects as his preference or area of interest. An example of such system is online newspapers, which allow customization of news categories based on user's interests. However, many of these systems cannot be considered as AHS, because an AHS uses a much more fine grained User Model and adaptation is done automatically instead of being 'selected' by the user [2]. Similarly, simple sequencing provided in some Learning Management Systems based on the content visited and the test results cannot be considered as Adaptive Hypermedia [3].

3.1 Typical Functions of AHS

De Bra [2] identifies three typical functions performed by an AHS. However, it is possible that AHS may perform more functions:

● The system continuously logs all the user interactions, based on which a User Model is created that includes his knowledge about content (and optionally his preferences and performance in questionnaire).

● The model is used for concept classification such as visited, unvisited, etc. The classified concepts are annotated with appropriate link anchors to guide the user in selecting the most appropriate concept.

● The content to be presented (which may include different levels of explanation for beginners and advanced learners) is divided into fragments (the actual fragment may or may not be in one page). The system conditionally selects the most appropriate content fragment by consulting the User Model and it can show, hide, dim, or highlight the selected fragment.

To better understand the functions performed by an AHS, let us look into the details of higher level representation of AHS as suggested by Brusilovsky [31].

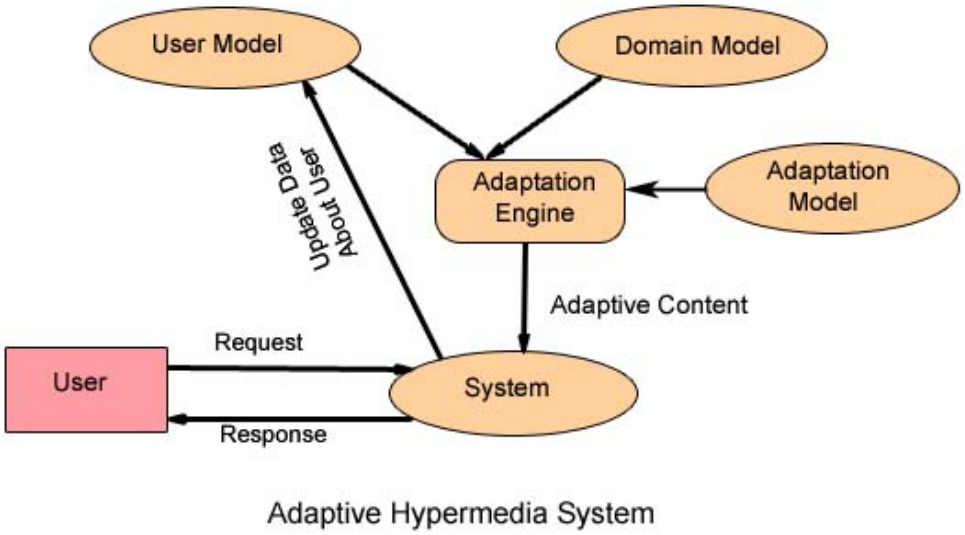


Figure 3.1: Generic Architecture of Adaptive Hypermedia based System [31]

As is shown in Figure 3.1, an AHS system coordinates its activities using an Adaptation Engine (AE), Domain or Content Model (DM), and User Model. A Domain Model describes the properties of content[1] and consists of multiple concepts. A User Model describes the user goals, preferences, and knowledge about the concepts visited. In its simplest form, a User Model consists of attribute value pairs. The Adaptation Engine (AE) is the core module responsible for adaptation effects. The AE engine queries the User Model and the Domain Model to decide on the next content to present for the user. The AE achieves the adaptation effect by using an Adaptation Model (AM) which describes for a given state of the User Model, what content from the Content Model should be selected for presentation. The AM consists of rules which are parsed and interpreted by the AE. In its simplest form, a rule consists of a condition and an action. Upon successful evaluation of a condition, the appropriate action is executed. An example rule is:

If (UserModelAttribute.Value = 'X') Then content = DM.conceptZ

The 'System' block, shown in Figure 3.1, coordinates all of these activities in the AHS. It accepts the user request for a concept and passes the request to the AE. It then waits for the response. Once it receives the adapted content from the AE, it may process it to generate appropriate user interface code (presentation logic). Apart from coordination activities, the system also may handle processing of other desirable features (e.g., usage and navigation log, communication tools like chat, discussion forum, etc.). Having looked at the general architecture of an Adaptive Hypermedia System, in the next section we shall look into the details of types of adaptation techniques.

[1] Depending on the level of abstraction in the system, a domain model may or may not describe the properties of content. In the latter case, the domain model consists of actual content. If the domain model describes content (as in the AHAM reference model), the actual content is separate from the description and can consist of one or fragments, which may or may not be a part of the same page.

3.2 Adaptation Technologies

Adaptation technologies (see Figure 3.2) can be classified in two[1] categories [31]: Adaptive Presentation and Adaptive Navigation Support. In the *hypermedia* terminology a content page is known as a *hyperdocument* or *node*. Each node can consist of some information or a link to another node (or nodes). The information in a node can be in text, image, audio, or video format. In the following section, we shall see the details of the two adaptation technologies and different techniques within those.

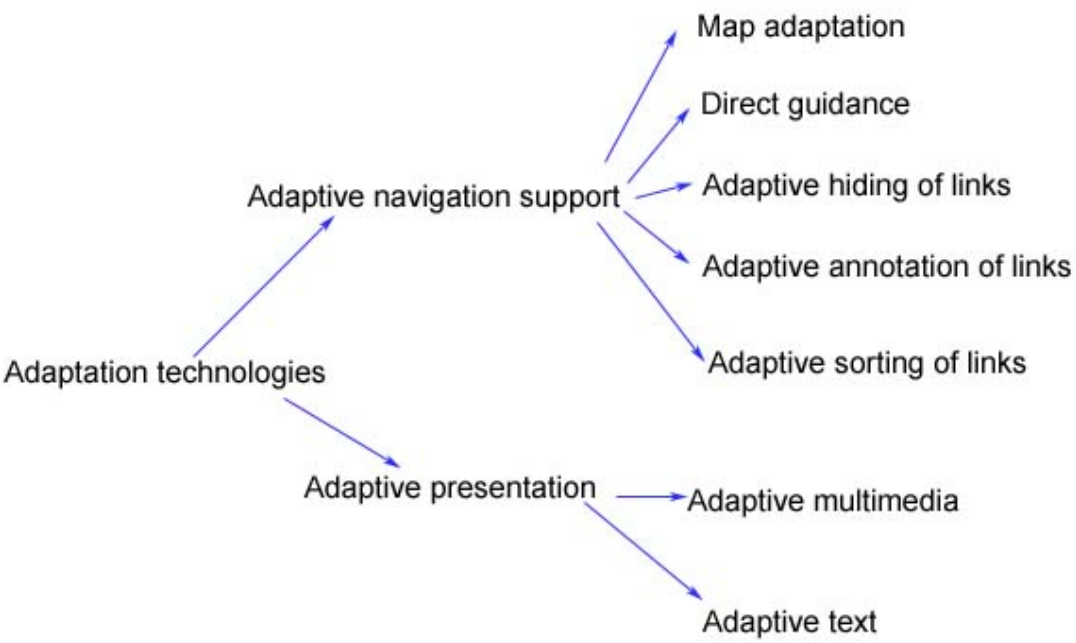


Figure 3.2: Adaptation Technologies [2] [31]

[1] A third class known as Spatial Hypermedia [33] also exists, though Brusilovsky defines only two. In Spatial Hypermedia, spatial properties of objects on a two dimensional plane are adapted depending on the user model state. The HyperMap [33] system demonstrates the use of Spatial Hypermedia.

[2] Image has been included with permission from the author.

3.3 Adaptive Presentation

In Adaptive Presentation, the content of a hyperdocument (which is also referred to as a page for brevity) is adapted. Since the definition of a hyperdocument also includes non-textual content such as images, audio, and videos, this also may be adapted. However, many of the systems developed so far have only used text adaptation. The idea of adaptive presentation was researched in Intelligent Systems [31]. There are two granularity levels within content adaptation: fragment adaptation and page (hypernode) adaptation. In fragment adaptation, a page is divided into multiple fragments. Individual fragment adaptation can be done by addition/removal of fragments, fragment sorting or alteration, and dimming[1] fragments. In the page level adaptation, content to be presented is separated in different pages.

3.4 Adaptive Navigation Support

In a hyperspace environment, all the users may not be able to decide which node to visit next. Adaptive navigation aims to guide the user by restructuring the links so that it becomes easier for the user to decide the next link to visit. This is achieved using different adaptive navigation technologies, which are described below:

● *Direct guidance* is used when the system can decide the next 'best' node to visit depending on the state of the User Model. The system can give visual clues to the user to indicate the best node to visit, or it can be achieved by using Curriculum Sequencing which uses the 'next' link. The use of the latter approach gives authors the flexibility to include the next node which is not directly connected to the current node and is an approach adopted in the AlcoZone system. However, one of the disadvantages of direct guidance is that it leaves no choice for the user but to follow the suggested link. In AlcoZone, we have tried to alleviate this problem by presenting the list of course content. But one limitation of our approach is that the content visited by browsing the content menu is not considered as 'visited'.

[1] In the dimming technique, non-relevant links are shown with dimmed colors so as not to catch a user's attention.

● *Adaptive ordering* is a technique in which all the presentable links are sorted using the User Model state such that the links at the top are more suitable to be visited than the links at the bottom. However adaptive ordering is applicable only for non-contextual[1] links. Brusilovsky also points out the research that shows stable ordering of links is important for novices [31].

● In *Link hiding,* the system decides the links which are not suitable to visit based on the User Model state. So, only the links whose prerequisites have been met are displayed. AHA [1] provides support for link hiding. Link hiding is more transparent to the user than link ordering and is used in many Adaptive Hypermedia based systems.

● *Link annotation* attempts to provide textual or visual cues to differentiate the links with different states, e.g., visited, learned, etc. The system can define as many link states as it thinks are appropriate. In its simplest form, most of the web browsers present today provide support for link annotation by defining two link states: unvisited (blue color) and visited (maroon color). AHA provides support for three link states: suitable to be viewed, unsuitable to be viewed, and visited links.

● *Map Adaptation* involves the adaptation of form of global and local maps presented to the user.

This is the taxonomy of the different adaptation techniques identified by Brusilovsky in [31]. Typically, Adaptive Hypermedia systems use one or more of these techniques depending on the objectives of the system. Use of more techniques may not necessarily improve the user experience and it is necessary to identify those that are most important in a given e-Learning system. The next section describes the study of various tools for authoring adaptive courseware. The objective of this study was to identify a tool that closely matches our requirements.

[1] These links are independent from content and can appear as a set of buttons, a list, or a pop-up menu.

3.5 Review of Adaptive Hypermedia Authoring Tools

Since Adaptive Hypermedia is a fairly new direction of research, there are not many tools available. Out of those which are present (and excluding commercial tools[1]), very few are at a stage where they can be used for the creation of an effective online course. The tools have two components: content authoring and content delivery. Not all of the tools we studied include these two components. Following are some of the objectives of the tools evaluation:

● If the adaptive courseware system includes tools for content authoring, it allows the framework to be reused across many courses, reaching its use beyond one course. This also requires that the AHS separates content authoring and content delivery. Thus one of the requirements was that the tool should include support for both content authoring and delivery.

● Many of the tools in adaptive hypermedia are still in experimental or prototype stage. Though some of the proposed architectures are theoretically better than some other tools, availability of stable source code in an established system was considered more important than a system in its preliminary stage.

● Availability of modular source code was considered to be a high priority.

● The system should support as many features (or it should be relatively easy to add those) as found in commercial e-Learning systems. These features include communication and collaboration tools, search, glossary, etc.

Based on the above requirements a preliminary evaluation was conducted to see which existing systems can be evaluated in detail. Though we found many systems which supported some adaptation techniques which we have seen in the previous section, we

[1] NetCoach [17] is a commercial adaptive hypermedia authoring and delivery tool derived from ELM-ART.

were looking for a system that provided flexibility in adding other features than those provided by the system. Based on the above criteria we evaluated MOT, which is essentially an authoring tool and does not provide delivery capabilities. However, content authored in MOT can be delivered using AHA [1]. Next, we evaluated APeLS which is a more recent effort at providing adaptive courseware authoring and delivery. Finally, we evaluated AHA, the system that we chose for extensions.

3.5.1 My Online Teacher (MOT)

MOT [11] is being developed mainly for the authoring of adaptive courseware. The system has been built on the LOAS [12] model which specifies five layered content authoring and is derived from the AHAM reference model described in Chapter 4. Apart from Domain, User and Adaptation Models, the LAOS model also includes a Goals and Constraints Model which consists of a restricted version of a Domain Model. LAOS also includes a Presentation Model that contains the physical and environmental properties of presentation. The Goals and Constraints Model enables creation of lessons derived from a Domain Model based on educational goals and preferences. For example, it enables creation of different lessons for beginners and advanced users. The Goals and Constraints Model adds one more layer of content description that is similar to a Domain Model.

Figure 3.3: My Online Teacher User Interface

Authoring in MOT is done by starting with the creation of the Domain Model. MOT allows inclusion of various attributes in the Domain Model. This creation process involves specifying the attributes for each concept. Some of the attributes of a concept are: explanation of the concept, exercises, and conclusion (see Figure 3.3). The Goals and Constraints Model is created from the Domain Model by selecting a subset of the attributes. The adaptation effect is achieved by specifying the Adaptation Model whose role is to interpret Domain, Goal, and Presentation models. An Adaptation Model can also update these models. Adaptation is built using a LAG [13] grammar. A LAG grammar consists of three layers. At the highest level, there are adaptation strategies which correspond to different pedagogical strategies. The second level consists of adaptation language and the lowest level consists of adaptation assembly rules such as the IF-THEN rule. Use of a LAG grammar allows use of different pedagogical strategies and abstracts IF-THEN rules.

MOT does not include a tool for delivery of adaptive content (i.e., adaptation engine) but does include conversion tools to convert high level specifications for use with other delivery engines such as AHA and WHURLE[1].

Though MOT offers far more advanced adaptive authoring, particularly suitable for creation of multiple courses, one of its major drawbacks include lack of support for content delivery. Another major problem is that the User Model was still in its implementation stage when we were evaluating different tools. MOT also suffers from the fact that most of its richer semantics are lost in the conversion process, as the delivery is limited by the capabilities of the adaptation engine (which are AHA and WHURLE at present).

3.5.2 APeLS

APeLS (see Figure 3.4) [21] [5] [6] is a promising tool, as it has been developed based on sound pedagogical principles. It uses multiple models and metadata to describe these models. The main models used are the Content Model, Narrative Model, and Learner Model. The Content Model (which is equivalent to the Domain Model in Adaptive Hypermedia terminology) describes the content to be presented at a higher level and does not include actual content. The Content Model metadata describes the content and it adds flexibility and compatibility with other metadata standards such as for IEEE learning objects [21]. The Narrative Model contains a rule set, and is similar to the Adaptation Model. The Narrative Model metadata describes the features of a rule set. The Learner Model is similar to a User Model and represents external factors. One of the strengths of APeLS is: it is not limited to the above models and addition of other appropriate models is possible by including appropriate metadata that describes the model. For example, by including a Device Model that describes the properties of the device (such as PDA or

[1] The WHURLE system is a learning environment and includes some preliminary support for adaptation of fragments. However, it does include link adaptation and presentation adaptation in terms of altering the user interface, which AHA does too.

desktop), content can be customized for that device. Personalized courses are offered as a service instead of a system. This makes it possible to deliver the courses using a Virtual Learning Environment (VLE). Integration with a VLE facilitates use of tools such as tutors, online assessment, and communication. Overall, APeLS scores very high on flexibility and extensibility. One of the drawbacks of APeLS includes limited support for authoring courses. The more important drawback is the developed features are still at a prototype stage and it has not been released as an open source system.

Figure 3.4: Mechanics Course in APeLS

3.5.3 AHA

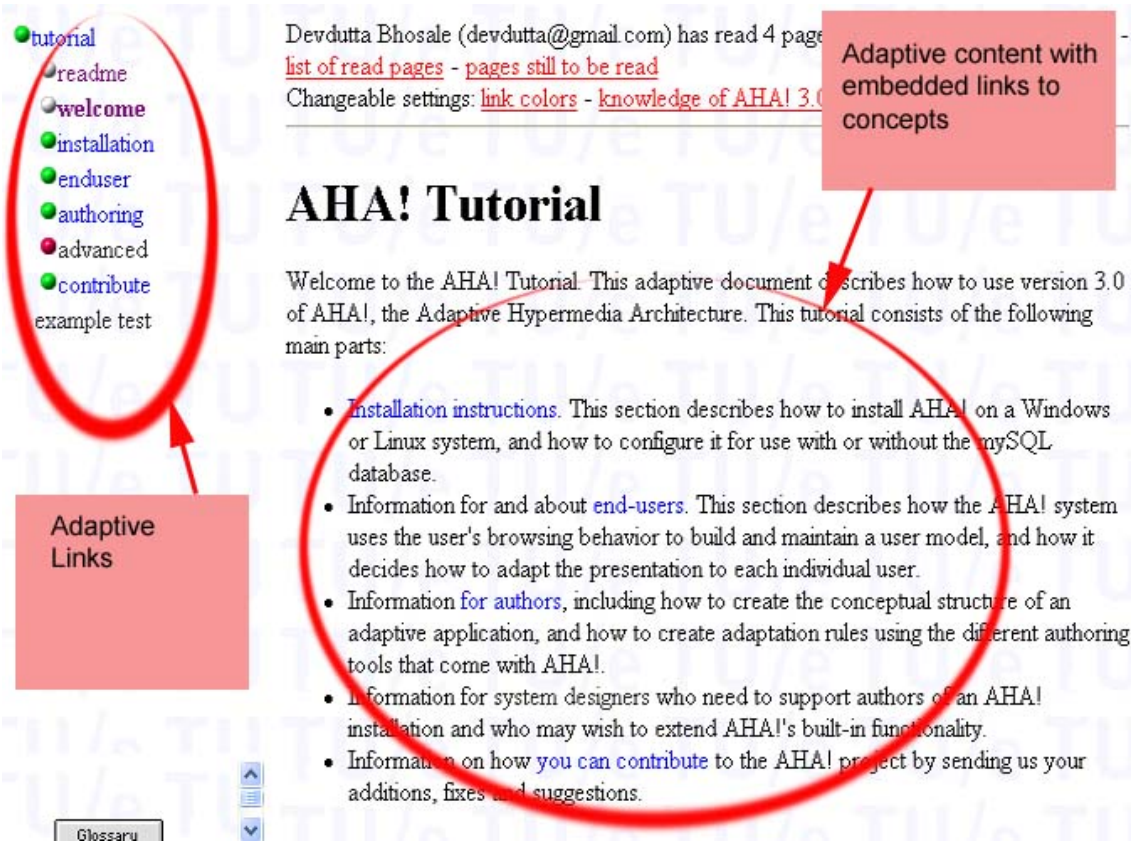


Figure 3.5: Adaptive Course in AHA

The architecture of AHA [1] supports presentation of adaptive content (see Figure 3.5) from a web server (Tomcat). The system consists of a combined Domain & Adaptation Model, and a User Model. Content creators and authors use the authoring tools provided in AHA to describe the properties of content, and create the actual content in HTML/XHTML format. Authors are also responsible for defining the adaptation behavior for the content. AHA provides a separation between Domain Model constructs, which consist of attributes and values for concept characteristics, and the actual content in the form of HTML/XHTML files. This adds flexibility to change the content without changing the content characteristics or vice-versa. The User Model consists of attribute-value pairs and typically defines the knowledge values for different concepts in the Domain Model. Thus, the User Model is an overlay model over the Domain Model.

3.5.3.1 Adaptation in AHA

AHA provides two types of adaptation: content adaptation (or Adaptive Presentation) and link adaptation (or Adaptive Navigation). Adaptive Presentation is implemented by conditional inclusion of objects or fragments. AHA provides two granularity levels in adaptive presentation. An object corresponds to a Domain Model concept while a fragment is a segment within an HTML/XHTML page. For conditional inclusion of fragments or objects, adaptation rules are included in the HTML/XHTML page. It is also possible to present an alternate explanation of a concept in a separate HTML/XHTML page by specifying adaptation rules in the combined Domain/Adaptation Model.

Adaptive navigation is implemented by execution of adaptation rules in the Domain Model which determine the desirability of concept links. If prerequisites for a concept have been met, the link is considered to be good and is displayed in a blue color. If the prerequisites have not been visited, the link is shown in black; otherwise the link is shown in pink[1].

AHA has been entirely written in JAVA as an open source system. Java servlets handle the user request and send the appropriate response. The User Model, Domain Model, and Adaptation Model are built either in XML or a MySQL database. AHA creates an in-memory copy of all of these models to improve the system performance. AHA source code is very modular and easily extensible. The major strength of AHA is a fairly comprehensive, stable, and flexible adaptation engine. Also, the user interface provides for authoring of content, making it easy for authors to create adaptive courses.

[1] Colors values for adaptive links are stored in the user model and can be changed by the user to suit his/her preferences.

3.6 Review of Other Tools

In this section, we briefly describe some of the other Adaptive Hypermedia tools studied as part of this research effort. Though many of these tools have influenced the more recent development efforts in Adaptive Hypermedia, their use in AlcoZone was found inadequate either because of use of previous generation technologies or limited support for adaptation coupled with difficult extensibility.

ELM-ART [19] [26] is one of the first Intelligent Tutorial Systems (ITS), appearing before ITS systems evolved into AIWBES (i.e., Adaptive Intelligent Web Based Educational Systems). ELM-ART includes adaptive navigation support by link annotation and link sorting. It also includes support for adaptive pages (adaptive presentation) and problem solving. ELM-ART is implemented in Common Lisp Hypermedia Server (CL-HTTP) and uses CGI interface to handle user requests. It was one of the pioneering ITSs that showed the use of adaptive navigation. One unique aspect of ELM-ART was that it included curriculum sequencing and shifted the burden of deciding the next concept to visit from the learner to the system. Some of the AlcoZone design aspects have been influenced by this design feature of ELM-ART.

Early development of ELM-ART led to InterBook [18], which is based on the metaphor of *electronic textbook*. It uses Domain and User Models to achieve adaptation. For each Domain Model concept, the User Model stores the user's knowledge about that concept. This is called as overlay model. Glossary is one central idea in InterBook. The glossary provides description about each concept in the Domain Model. It also provides links to all the sections which introduce that concept. We can see that the architecture of AHA has been greatly influenced by InterBook and AHA does provide a conversion tool to convert InterBook content for delivery within AHA [7]. The main adaptation scheme in InterBook is link annotation. A green ball and bold text indicate content is ready to be visited which depends on the visited state of the corresponding prerequisites. A red ball indicates the content that is not yet ready to be visited. Apart from colored balls, InterBook also uses checkmarks to differentiate the visited status of concepts. Similar to

ELM-ART, InterBook also has been developed using CL-HTTP and LISP technologies. The other major limitation of InterBook is that there is a limited support for content adaptation.

Having looked at some of the Adaptive Hypermedia based systems and tools for their application in the development of AlcoZone; we shall compare them on the basis of the evaluation criteria identified below.

Availability of the tool as an open source application was the foremost requirement considering further extensions that were necessary. Modular and extensible system architecture also was considered an important factor. One of the objectives of AlcoZone was to provide different content for each different category of users. For this, content adaptation was a priority than providing navigational assistance. Adaptation Engine design is largely influenced by the design of the Domain, Adaptation, and User Models. Since the design and implementation of a Adaptation Engine is a fairly complex task, availability of an Adaptation Engine was an essential requirement. Some extensions to AHA – namely multiple knowledge levels, data mining and compatibility with SCORM – have been discussed in [22]. However, these were inadequate for the purpose of AlcoZone as can be seen in Section 4.4. Based on the above observations, we now present our evaluation matrix (see Table 3.1) and conclude this section describing why we chose AHA for our extensions.

Metric	**MOT**	**AHA**	**APeLS**
Extensibility support	Little	Yes	Yes
Flexibility	Little	Good	Good
Author support	Little	Excellent	Little
Adaptation Engine	No	YES	YES
Open source Availability	Yes	Yes	NO
Source code quality	Poor	Good	NA

Table 3.1: Comparison of Adaptive Hypermedia tools

Based on the above evaluation criteria, we found MOT did not provide an inherent adaptation engine and used either the AHA or WHURLE engine for content delivery. AHA and APeLS are both good systems with included adaptation engine. However AHA faired really well in terms of availability and quality of source code and support from authors. Also, the AHA system is constantly evolving with incorporation of suggestions by people who evaluate AHA. APeLS offers better architectural advantages as compared to AHA but the developed system is still at a prototype stage and it is not as comprehensive as AHA. All of these factors led to the selection of AHA as the tool for extensions. In the next section we shall look in detail at the architecture of AHA, particularly the design and implementation aspects within AHA.

Chapter 4: Architecture of AHA

The architecture of AHA is based on the AHAM [30] reference model. The AHAM model defines at a conceptually high level how the architecture of an Adaptive Hypermedia system can be defined. The model specifies the architecture using "educational terminology". Some of the key aspects of AHAM are:

● Adaptation is based on three higher level models which are called as Domain (or content) Model, User (or student) Model, and Pedagogical (or adaptation) Model.

● A *concept* and a *page* are two different notions. Many closely coupled Adaptive Hypermedia Systems do not separate these two.

● Development of authoring tools based on the AHAM model is easier compared to those which do not use it.

● In an AHAM based system; it is possible to construct the User Model not only based on a domain overlay but also from external sources such as tests.

In the next section, we shall see how the AHAM reference model is realized in AHA.

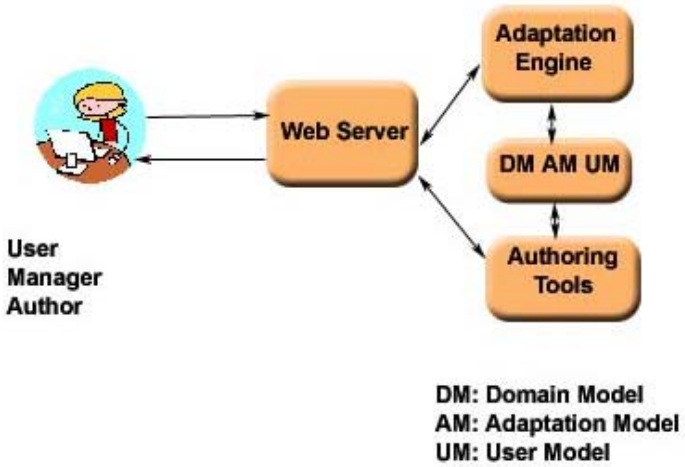


Figure 4.1: Architecture of AHA [1]

As can be seen in Figure 4.1, there are three types of users that can interact with AHA:

● *Authors* are responsible for content creation and defining the adaptation structure. They define the prerequisites for a concept and how much knowledge the individual concepts contribute to the entire course. To define the concept structure and adaptation behavior, authors can use the graphical tools provided by AHA (i.e., Graph Author and Concept Editor). However these tools only define the concept hierarchy, structure, and attributes; the actual content to be presented is separated into HTML or XHTML files. The authors are responsible for the creation of these files.

● *Manager* is responsible for system configuration and making sure that the content authored is delivered using AHA. Manager may work closely with authors to make sure that the content authoring process conforms to AHA guidelines.

● *Users* take the adaptive course developed in AHA. User requests are handled by a web server which hosts AHA. Users can take the course using a web browser.

Architecture of AHA consists of servlets which handle the user requests and send the response using supporting JAVA classes. An author uses the Graph Author tool to create concepts and define relationships between them. Concepts represent topics to be learned in an application domain. A concept has multiple attributes associated with it. An attribute may have meaning for the system (e.g., access, which triggers adaptation rules) or for the author, or for both. It is possible to define custom attributes using the authoring tools. Using the Concept Editor, it is possible to get low level access to adaptation rules. The result of the authoring process is a file that stores all the information abouit concepts and their attributes. This file is interpreted by the authoring tool only and AHA creates individual concept files for each of the concepts. AHA also creates an in-memory copy for each of the concept files. As the user accesses different concepts, the User Model is updated using the adaptation rules which are part of the concept file for that concept (and also present in memory). An adaptation rule consists of two parts: a *condition* and an

action. Condition is a Boolean expression, and if the condition evaluates to *true*, the true action associated with the condition is executed (there may optionally be a false action which is executed if the condition evaluates to false). Execution of actions results in attribute update of the same or another concept, and this may trigger adaptation rules associated with that attribute. The authors of adaptation rules can specify whether they would like to trigger the execution of rules associated with the updated attribute (*isPropagating* flag). The idea behind this is that attribute value changes in a concept typically cause changes in values of attributes in other concepts. For example, if a web based course consists of one or more chapters and each chapter consists of multiple lessons, then knowledge gained by visiting a lesson also constitutes to the gain in knowledge of its parent chapter and to the course.

4.1 Adaptation Techniques in AHA

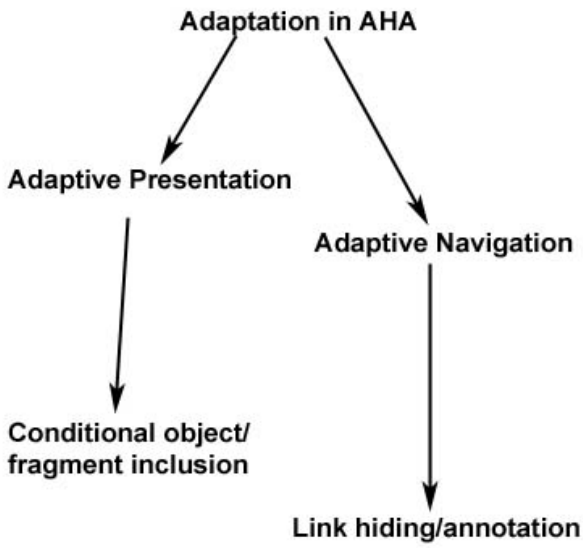


Figure 4.2: Adaptation in AHA [1]

AHA supports two types of adaptation (see Figure 4.2): Adaptive Presentation and Adaptive Navigation. Conditional inclusion of object or fragment is the technique used

for adaptive presentation. Conditional inclusion can be at a page level or at a fragment level within a page. By embodying conditional statements (e.g., IF THEN rules), it is possible to show or hide fragments within a page. The conditional rules are typically based on the state of the User Model (i.e., value of attributes). If a rule evaluates to true, the fragment is shown. Adaptive content presentation is also possible at a page level. Depending on the state of a User Model, different page level content can be presented. All these adaptation rules are part of concept attributes in the combined Domain and User Model.

Link annotation is the main technique used for adaptive navigation. As the user is visiting different concepts in the hyperspace, the User Model is constantly updated to reflect this. Visiting different concepts also contributes to gain in knowledge. Whether a concept is suitable to be visited is determined using the knowledge gained of the prerequisites for that concept. If the knowledge gained is above a certain threshold value, the suitability attribute is changed to 'true'. Depending on the suitability value of a concept, the link colors are changed. It is possible to hide a concept link, if the prerequisites for that concept have not been met.

4.2 Reverse Engineering AHA Adaptation Engine Design

Though AHA's source code is very modular with plenty of room for extensions, lack of source code design documentation led us to create design documents so that we are better able to understand dependency and the flow of control between the various modules. We concentrated our reverse engineering effort on AHA's adaptation engine since the modification of the adaptation engine was more a priority than modifying the authoring tools. In this section we shall see how we constructed the high level design from the source code.

The main idea behind the design construction was to follow the flow of execution and understand how adaptive content is served. Some of the things we tried to understand were:

- How different models (domain/adaptation and user) are represented in memory.

- The execution of adaptation rules and how it updates the User Model.

- The ability of the system to serve multiple courses to different users simultaneously.

AHA's design consists of JAVA servlets which are responsible for processing the user requests (or HTTP requests) and sending the corresponding response. However, most of the processing is done by Java classes which are invoked by the servlets. The JAVA servlets are invoked whenever an author is creating content, a manager is configuring AHA, or a student is taking an adaptive course. After a request is made, the servlets invoke the methods on the supporting classes which generate the resulting HTML stream, which is sent back to the user. We shall focus on the servlets which are invoked when a student is taking an adaptive course. There are two servlets which handle the communication with a student and the adaptation engine: ViewGet and Get. The Get servlet is responsible for execution of adaptation rules in the combined Domain & Adaptation Model, and updates the User Model. After the Get servlet is finished processing the adaptation rules, it sends an HTML stream to the user content, containing one or more requests to the ViewGet servlet. The requests to the ViewGet servlet have a different view name as their parameter. The result of these calls is the generation of HTML/XHTML adaptive content in multiple views. Before we see how the Get and ViewGet servlets work to generate adaptive content, we shall see AHA's presentation and layout capabilities.

4.3 AHA Layout

The content presented to the user is divided in multiple views. For example, the listing of content (presented in different colors as per suitability and labeled StaticTreeView in Figure 4.3) is one view and the textual/image content about a concept is another view (i.e., MainView). In Figure 4.3, the user screen is divided into two sections; the left hand side concept listing is the StaticTreeView, and the detailed concept explanation on the right hand side is the MainView. These views are presented using different HTML frames. For each view, there is a separate class for generation of appropriate HTML content for that view. The number of views for a course and which view is presented using a particular frame is configured using LayoutConfig.xml. Views also are used for presentation of additional course features that are useful in an E-Learning environment, like glossary of concepts, table of contents, etc. Whenever a request for a next concept is made, it is sent to the Get servlet. Get servlet accesses the in-memory Domain/Adaptation Model, and executes the adaptation rules in the combined Domain/Adaptation Model. After this, it updates the User Model. Get servlet then accesses the layout and presentation style to create one or more requests for views defined for the current course. Figure 4.4 explains the flow of control in the Get servlet, and Figure 4.5 is a sequence diagram depicting major AHA classes invoked by Get servlet in the generation of multiple view requests, execution of adaptation rules, and updating the User Model. We have described the execution sequence in the following steps:

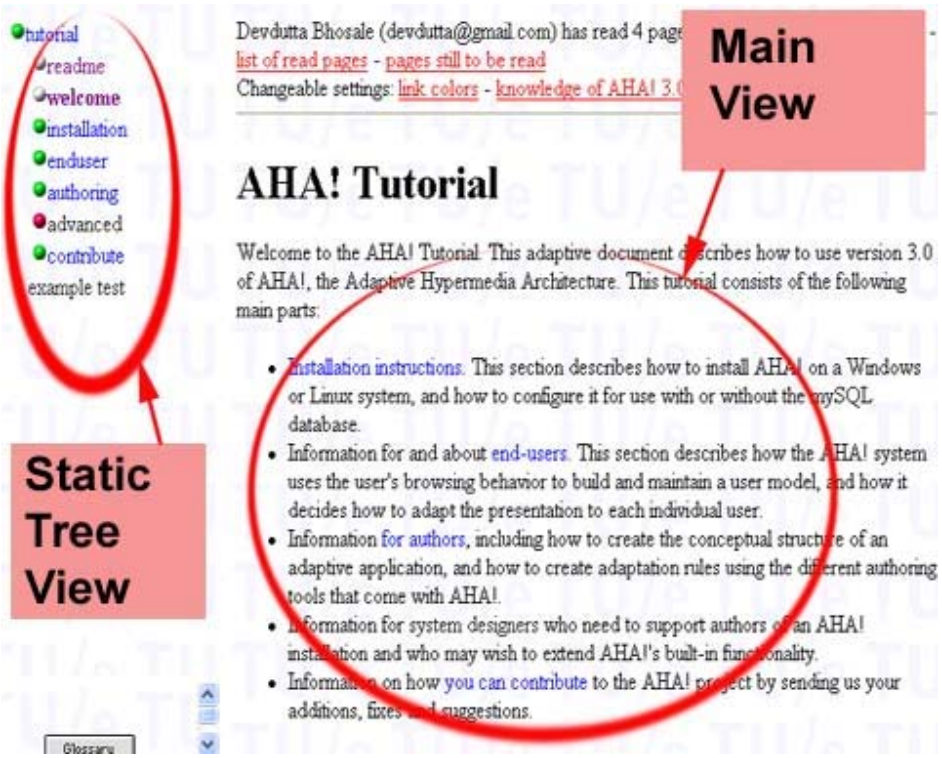


Figure 4.3: Views in AHA

● Whenever a user clicks on a concept name, the request is sent to the Get servlet with the name of the concept to be presented next.

● The Get servlet accesses an in-memory concept structure which has embedded adaptation rules in it. It then invokes the ProfileManager class which manages adaptation rule execution and the User Model update. ProfileManager instantiates the ProfileUpdate class, and starts the execution of adaptation rules starting with the rule associated with the *access* attribute of the current concept.

● Adaptation rule execution results in attribute update of the same or different concept (in case of knowledge propagation). If *isPropagating* flag for an adaptation rule is *true*, it triggers the execution of adaptation rule execution of an updated attribute.

● The result of adaptation rules execution is the updated User Model.

● After the User Model update is processed; the Get servlet processes the layout information for the current course. The layout of a course defines different views for the course (e.g., glossary, main content, table of content, and custom defined views). Using separate HMTL frames, different views are presented to the user. The frame structure for the corresponding view is defined in an external XML file. This makes it easy to change the layout of a course independent of other courses.

● To generate flexible layout code, the Layout class is used. This stores the HTML representation of frame structure with a placeholder for the current concept name. Get servlet retrieves the HTML frame structure representation using the Layout class and replaces the placeholder with the name of the current concept.

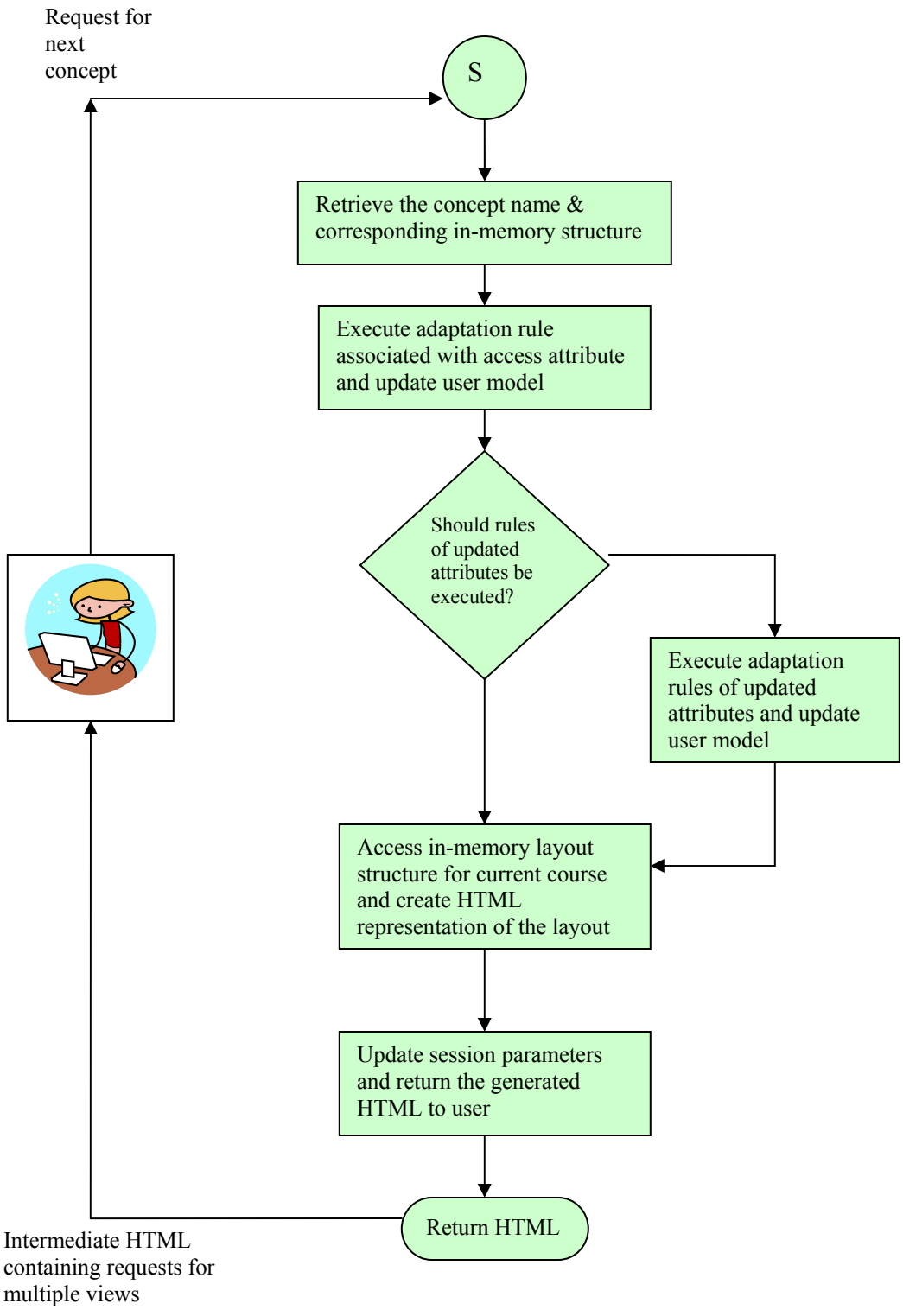


Figure 4.4: Flow of control in Get servlet

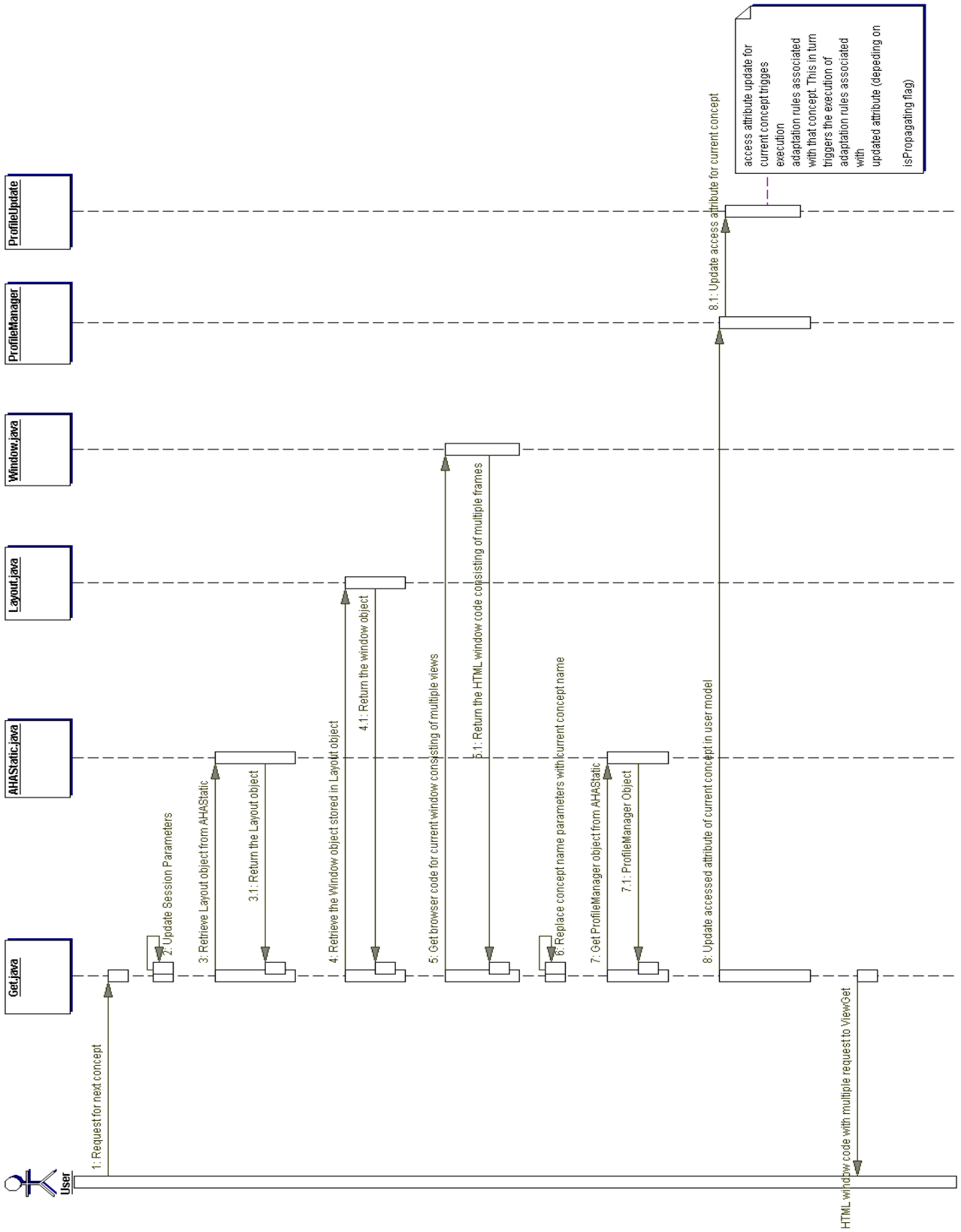


Figure 4.5: Sequence diagram for Get request

So far we have discussed only the first step in the delivery of adaptive content which generates the frame structure. The second step involves the actual delivery of content in appropriate HTML frames defined by the frame structure. For each of the HTML frames, the user's browser sends an HTTP request with the name of the view and the concept name as the parameters. These requests are sent directly to the ViewGet servlet. The ViewGet servlet processes the requests for different views independently. So if a frame structure consists of three frames, there will be three requests to the ViewGet servlet with appropriate parameters. Figure 4.6 explains the flow of control in the ViewGet servlet while Figure 4.7 is a sequence diagram containing all the major classes invoked by ViewGet servlet to generate HTML code. Based on these two diagrams, we delineate execution steps in the ViewGet servlet:

● User's browser sends an HTTP request to the ViewGet servlet with the name of the view and the requested concept name.

● The ViewGet servlet extracts the view name and the requested concept name from the HTTP request. It then retrieves a reference to the View object from the AHAStatic[1] class. The retrieved class handles content generation for a given view.

● The ViewGet servlet then invokes the content generation method on the View object (genBrsCode()).

Before we describe further system flow, we need to note that the implementation of different view objects differ completely from one view to another. The only similarity is that all the view objects implement the *view* interface. Thus, they all implement methods declared in the *view* interface. The sequence diagram in Figure 4.7 describes how

[1] The AHAStatic class is a static class that acts as a container to hold all the objects that are used throughout the application lifetime. Objects held in AHAStatic are instantiated only once when a user logs into AHA. The in-memory objects mainly store all the course information for each course. This includes layout and concept hierarchy of a course.

MainView generates HTML code. The MainView class is responsible for generation of HTML code by processing HTML/XHTML resource files. There are one or more resource files associated with a concept. The resource files are specified as one of the attributes of a concept using AHA authoring tools (Graph Author or Concept Editor). We now continue with the system flow.

● Using the requested concept name, the MainView class gets the in-memory structure associated with the concept.

● It then locates the resource file associated with the given concept with the help of the HandlerManager class. If the author of adaptive content wants to present different content to users depending on the state of the User Model[1], pointers to different files are included as concept attribute (i.e., casegroup structure). In this case, the HandlerManager class executes the rule associated with conditional object inclusion. If a rule evaluates to true, the pointer to that resource is considered. If no rule evaluates to true, a default resource is returned. If the concept does not define conditional resource inclusion then the resource attribute of the concept points to the appropriate resource.

● Depending on the type of resource (i.e., HTML or XHTML), an appropriate resource Handler (XHTMLHandler or HTMLHandler) processes the file. The handler parses the XHTML file looking for certain AHA keywords <object type=”text/aha”> and IF tags for conditional inclusion of fragments. The conditional inclusion rules are parsed and evaluated by the parser and the appropriate fragments are included.

● The generated HTML stream is returned to the ViewGet servlet, which in turn returns it to the user.

[1] In order to present different resources depending on the execution of a condition, the author has to define the concept as of ‘fragment’ type.

The similar process is followed for other ViewGet requests. AHA prerelease 21 uses a frame structure consisting of three views: MainView, StaticTreeView, and ToolBoxView. Requests are made by the user's browser to fetch these three views. Depending on the name of the view, an appropriate class is instantiated. StaticTreeView is an HTML link listing of concepts. The link colors of concepts in the list are changed depending on the suitability of the corresponding concept. The ToolBoxView aggregates the Table of Content and Glossary for the course. Thus, we can see that it is possible to define custom views to customize the look and feel of presentation.

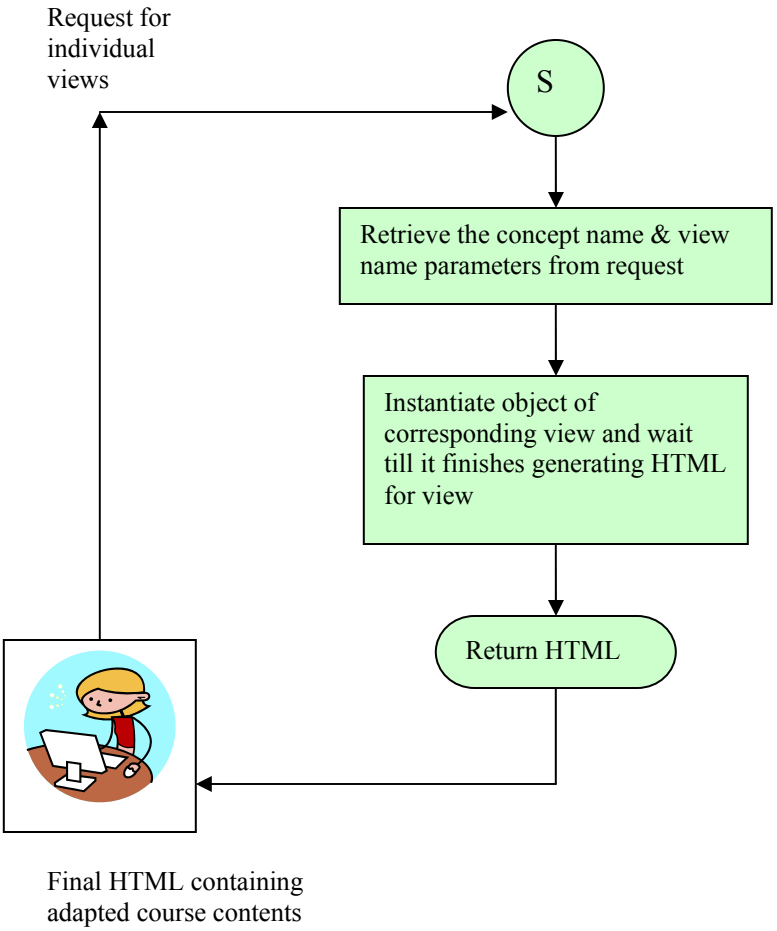


Figure 4.6: Flow of control in ViewGet servlet

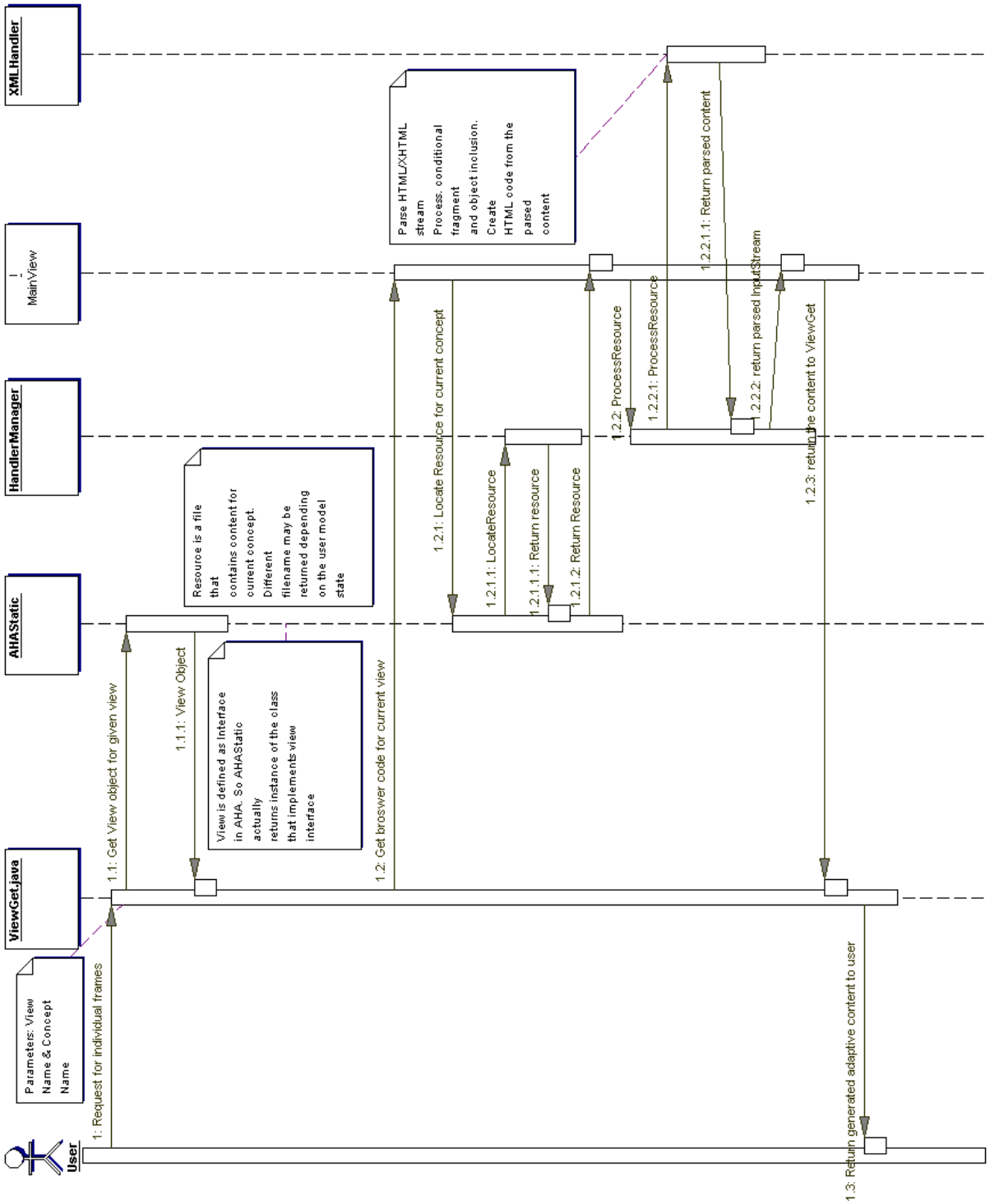


Figure 4.7: Sequence diagram for ViewGet request

The other important aspect of AHA is the in-memory data structures and AHAStatic class, which is the container for all the in-memory objects throughout the application's lifetime. Though the User Model, concept database, and course configuration reside on disk in the form of XML files or in a database, to improve the responsiveness of system, AHA creates an in-memory copy of all these objects. Initialization is done using three different functions (which in turn may invoke methods on other class objects).

The AHA application needs to use many configuration parameters (e.g., base path, course directory, concept database directory). These parameters are read from an XML file and stored in the AHAConfig object. To handle User Model modifications, the ProfileManager class is used. The HandlerManager class handles resource files associated with a concept. ViewManager maintains a ViewList and adds or removes views to this list by reading LayoutConfig.xml. AHADB acts as a database container for concepts, user profiles (or models), and the logging mechanism. Figure 4.8 is a class diagram depicting the major classes and attributes of in-memory data structure.

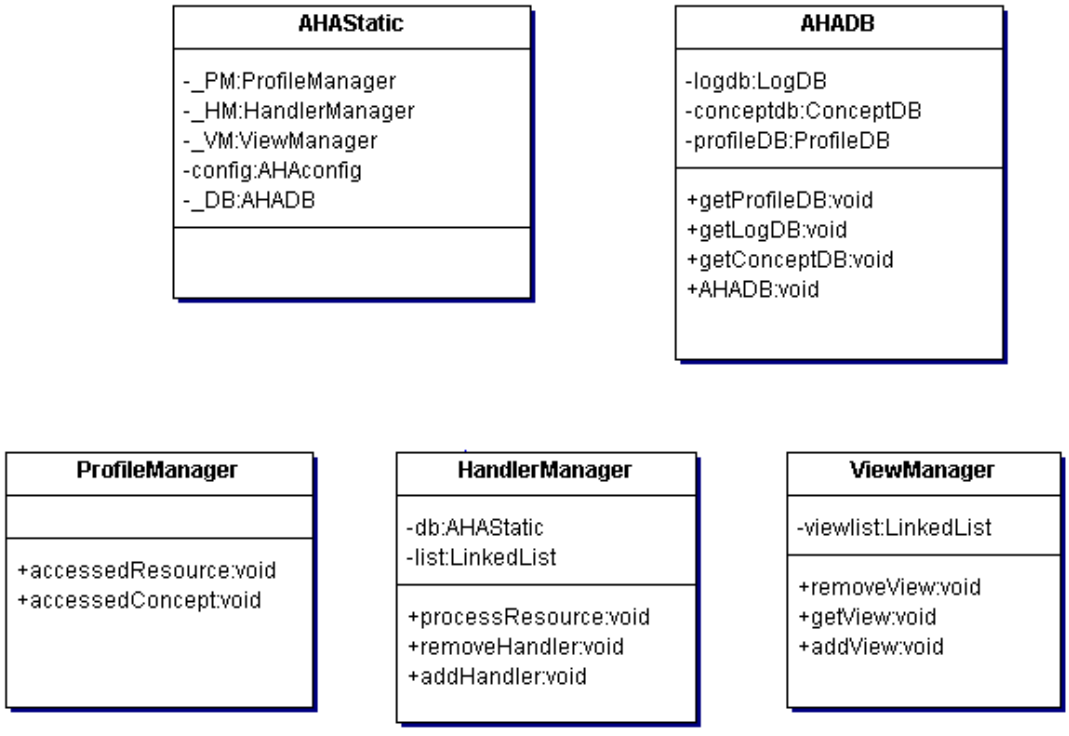


Figure 4.8: Class Diagram of in-memory Data Structures

As can be seen in Figure 4.9, at the root of all data structures is CourseInfoTable, which combines all the courses together. CourseInfoTable is a Hashtable of CourseInfo objects. The CourseInfo object defines all the properties of a course. There are six attributes defined for a course. CourseName uniquely identifies a course. Dependency structure is generated when two concepts are related by a relationship. The outcome and prerequisite of a concept are defined using Dependences. There is a root concept whose name is the same as the name of the course and all other concepts in the course are children of the root concept. This concept hierarchy is arranged in the form of a tree and the Hierarchy structure identifies the parent child relationship between the concepts. The ConceptInfo object defines the name, type, and title of a concept. All the ConceptInfo objects of a course are combined together in the ConceptInfoTable. Concept types are differentiated based on the attributes they can have. For example, an abstract concept does not have a resource associated with it while a page concept does. It is possible to change the presentation of different concept types using a configuration file (ConceptTypeConfig.xml). A ConceptType object holds presentation information for each concept type by processing the ConceptTypeConfig.xml. The Layout object holds information about different views and their presentation. This information is provided on a per-course basis in LayoutConfig.xml file. During the initialization process, LayoutConfig.xml is processed and the information is retrieved in the Layout object. The data structures described in Figure 4.9 bring together all the elements of a course and also provide meta-data about all the concepts as part of a course. However, detailed concept information which includes attributes and adaptation rules is stored in a concept database called the ConceptDB. The AHADB class described in Figure 4.8 defines the methods for attribute retrieval and modification.

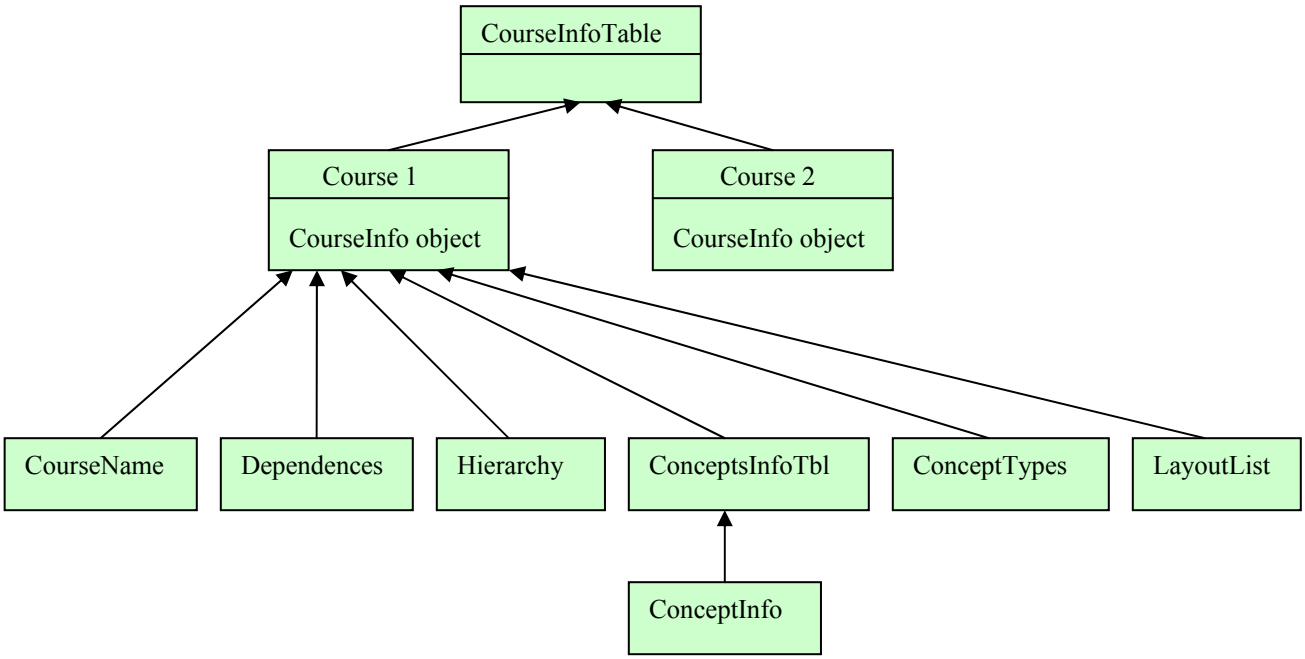


Figure 4.9: AHA Course Hierarchy

4.4 Proposed Modifications in AHA

Though AHA does a nice job of adaptive course authoring and delivery using a framework that is flexible and extensible, it was not adequate in its existing form to realize all the objectives of AlcoZone 1.0 identified in Section 2.4.

In this section we shall describe the requirements of AlcoZone that were not satisfied in AHA, and modifications needed to support them.

4.4.1 Design of Custom Views

AHA's original design encourages users to browse through the hyperspace and finish the course by selecting those concepts that they find appropriate. Assistance is provided to differentiate those concepts whose prerequisites have been met from those whose prerequisites have not been met. In order to accommodate the proposed extensions like curriculum sequencing, etc., it was necessary to redesign views and create classes for presentation of these views. We identified two new views and modified one existing view as part of this process.

4.4.2 Pedagogically Defined Concept Hierarchy

In AHA, an author has the freedom to define the concept hierarchy to suit his or her requirements, and AHA does not put restrictions on it. However, in any learning environment content is typically divided into logical modules like chapters or lessons. This modular division helps both authors and learners by providing clear separation between learning units. AHA did not provide support for content division and it was possible to define a concept hierarchy without restrictions. We have added a three level content hierarchy in which content is divided into chapters, lessons, and concepts. A chapter consists of one or more lessons and a lesson consists of one or more concepts.

4.4.3 Curriculum Sequencing

The Adaptive Navigational support provided in AHA using link annotation gives visual cues about content that is suitable to be visited based on the prerequisites met. However, many times users are not able to choose the best possible alternative to visit out of the available choices. Many E-Learning systems have approached this problem by providing curriculum sequencing. In this technique, the system decides the next best content to be presented to the user. The user interface facilitates navigation to the next content by including HTML link anchors. Using the same technique, it is also possible to visit previously visited content. ELM-ART [26] provides curriculum sequencing at the same time as it provides link annotation. Combining these two approaches allows users with more background knowledge about the domain to select what they want to learn while novice users prefer to take the course by following *next* link [26]. We have tried to provide similar support in AlcoZone where the system is able to decide what should be presented next based on the User Model. At the same time users have freedom of browsing the course contents on their own.

4.4.4 Conditional Jump between Concepts

Conditional jump between concepts is a complimentary feature to curriculum sequencing and works for those concepts which are defined to be of type *option*. The *option* concept has questions associated with it. Whenever a user visits an option concept, his answers to the questions are automatically stored in the User Model. The next concept to be presented to the user depends on the answers to the questions. So even though hierarchical concept structure defines the next logical concept, the actual concept to be presented is dependent on the answers to the current concept.

4.4.5 Questions and Exercises Distributed Throughout the Course

It is possible to create adaptive tests using AHA's Test Editor tool. However, concepts explained as part of the Domain Model are not immediately (or even after visiting a few more concepts) followed by questions related to the learned concepts. Thus, we needed a way in which presentation of questions is not separate from the presentation of concepts, i.e., there are some concepts which have embedded questions in them. It was also necessary to save the questions asked and the corresponding answers in the User Model so that content can be adapted separately for each user. To solve this problem, we have defined two custom concept types known as *option*, which was briefly discussed in the previous section, and *nextfragment.* Using these concept types, it is possible to present certain concepts which have embedded questions in them. Answers to these questions are automatically stored in the User Model, and can be used for adaptation effects.

4.4.6 User Interface Customization

User interface and color appealing to one user may not be interesting for another. Using our customizable user interfaces, users can select the look and feel of the course that is interesting for them. AHA presented web pages using very simple layout and colors which we thought may not be interesting for many. We needed a more sophisticated web design that can be changed on the fly. To support this, we developed an extension which we called *skins.* Using the skins extension, the user HTML code generated by AHA is further processed to change its look and feel. The new presentation HTML code is picked up from an XML file. The code in XML has defined placeholders where adaptive content needs to be inserted. This HTML code is divided in segments and is grouped by view name.

4.4.7 Content Listing Sorted on Visited Status

Curriculum sequencing provides a way for users to see concepts in sequence. However, some users prefer to browse the content and read what they like. To accommodate this we provided a course content menu that is divided into sub-windows. The first window displays content that users have not visited. The second window, located vertically below the first, contains listing of concepts that have been visited. As the user visits different concepts, these listings are dynamically updated to reflect the current state of the User Model. The concepts in the menu are grouped according to the lesson and chapter they belong to.

4.4.8 Delivery of Non-Adaptive Static HTML pages

An effective E-Learning system does more than presentation of adaptive content. Features such as help, glossary, search, communication, and collaboration tools are certainly desirable. AHA content delivery is centered on the presentation of HTML or XHTML resource pages defined as a resource of a concept. However, delivery of XHTML/HTML pages which are not part of a concept structure is necessary to provide additional course features. In a few cases, AHA does present non-concept HTML pages (e.g., changing link colors and knowledge setting in the User Model). However, HTML code for those pages was hard coded in AHA, making delivery of other pages inflexible. Our extension makes it easy to process and present any HTML page which is not part of a concept structure.

4.4.9 Audio Narration for Textual Content

Many users find textual content accompanied by audio narration interesting. To support audio narration, we include a pointer to the audio file containing the narration, in the XHTML file. If a user has indicated that he or she would like audio narration to be played back, then the parsing code in XHTML parser inserts the user interface audio controls

necessary to play back the audio. User preference about audio narration is stored in the User Model so that the playback can be controlled for individual users.

4.4.10 Logging Capability to Log Navigation Behavior

The User Model of AHA persists in an XML file and provides information about concepts visited, knowledge gained, answers to questions, and user preferences. But the User Model does not provide information about the navigation history of a user. This information is useful to determine course usage and usability of the system. Our logging API captures each of the user interactions with the system. The usage log for each user is stored in separate XML files. By combining the User Model with the navigation history it is possible to determine the correlation pattern between user profile and navigation behavior. Also, the data mining of user profiles reveals interesting observations about user characteristics.

Having discussed the specific modifications in AHA to support the extensions for AlcoZone, we shall discuss the design of these modifications in the next chapter.

Chapter 5:
Design and Implementation of Extensions

5.1 Design of Custom Views

One topmost priority of AlcoZone was the user interface redesign, as the existing interface that came with AHA was very simplistic and lacked features that were necessary for AlcoZone. We decided to use AHA's layout capability of using multiple frames and mapping different views in different frames. Though use of HTML frames is not popular these days, we decided to concentrate our effort on extending AHA to support different user interfaces instead of creating a more sophisticated interface. Some of the objectives of our user interface design are:

● Maximum screen area should be available for a view that presents concept explanation. This was necessary because of inclusion of images, audio controls, and interactive flash elements. This view should also include HTML anchors for curriculum sequencing (i.e., Next, Previous, and Repeat)

● The second biggest area is occupied by Adaptive Link Presentation of course contents. Using this menu, users can keep track of the visited concepts and those that still need to be visited, grouped by Chapter, and then by Lesson.

● TopView was created that includes a progress bar (displaying course completion in percentage), option of logging out of the system, and resuming the course.

5.1.1 TopView

Figure 5.1: Top View

TopView, as the name implies, resides at the top of the screen. The view includes a progress bar which tells users how much course they have finished and how much is still remaining. The 'Resume' button lets users return to the last visited concept no matter where they are. As the user is browsing the content in the hyperspace (using TreeView) or visits additional course options such as help and glossary, the displayed page changes. The 'Resume' button lets him continue from the point where he left. The 'Log Out' button lets him log out of the system. Users can still revisit the course and resume from the point where they left. Information about the last visited concept is stored in the User Model.

5.1.2 StaticTreeView

Figure 5.2: Static Tree View

StaticTreeView has been modified and it now consists of two DynamicTree views and acts as a container for four vertical tabs that expand or collapse the user screen space. The content followed by 'Content Not Visited' image in Figure 5.2 is the first DynamicTree view, and content followed by 'Visited Content' is the second DynamicTree view. These two views together sort the concepts based on their visited states. The concepts are grouped together according to the lesson that they belong to. The views can only display lessons within one chapter at a time. By clicking on the chapter number, lessons and concepts which are part of that chapter can be seen. The ball which has an 'L' embossed on it indicates that it is a lesson and the ball with an embossed 'C' denotes a concept.

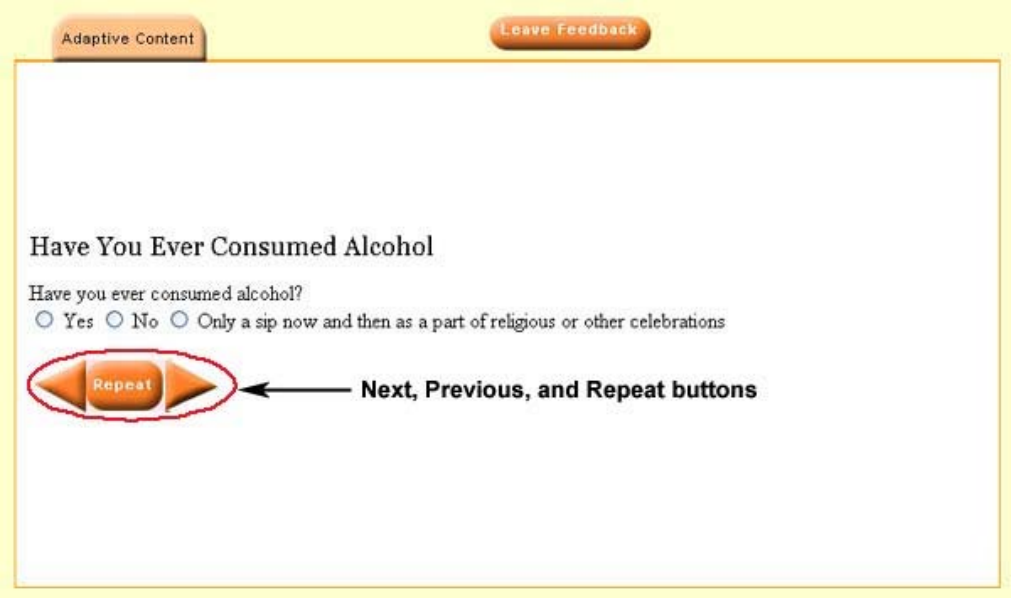


Figure 5.3: Main View

5.1.3 MainView

MainView occupies the biggest screen space and presents concept explanation (see Figure 5.3). The content consists of text, images, and audio. The MainView and StaticTreeView are located side by side horizontally and can expand or collapse depending on user preference. Apart from content, MainView also contains Previous, Next, and Repeat buttons. The 'Next' button points to the next concept (this may point to the actual concept or could be a dummy concept name); the 'Previous' button takes the user to the previously seen concept[1], and the Repeat button presents the current concept again.

In addition to the design of these views, the course needed to provide additional features such as ability to change the look and feel, audio narration controls, glossary, and help. There were more features that we felt could be added in the future. To accommodate this growing list of features on the limited screen space without compromising on the space available for content explanation, we decided to use make use of frames that can be

[1] Currently using the previous button users can go back only one concept.

expanded or collapsed. If the screen is expanded, then it hides the course content menu displaying only MainView horizontally and TopView at the top of the screen. The additional features are grouped in three categories: Course Tools, Preferences, and Help. Course Tools contains a glossary for the course. Using 'Preferences', it is possible to change the user interface and colors, and turn on or turn off audio narration. The Help section provides instructions about how to take the adaptive course and use of other tools. To access these sections, we have provided three vertical tabs. The fourth tab is for the Course Content menu. Clicking on any of these tabs expands the screen if it is not already expanded. Appendix A includes a screen shot of the expanded AlcoZone screen that shows the arrangement of the three views discussed above.

5.2 Pedagogical Concept Hierarchy

Using AHA's authoring tools (Graph Author and Concept Editor), authors have the freedom to define the parent child relationship between the concepts. While this may be useful for many adaptive web applications like Electronic Textbooks and in systems that provide navigational assistance, in E-Learning applications users prefer to learn content that is logically divided in different learning units. To implement this learning unit hierarchy, we have authored AlcoZone content in four levels. The root node (level 0) is the name of the course. 'Level One' nodes are chapters and contain one or more lessons nodes as their children. The lessons aggregate one or more learning concepts. Figure 5.4 explains the three level content hierarchy. The left hand side of the image labeled 'Concept Hierarchy' corresponds to the parent child relationships between concepts. The parent child relationship is created using two pointers: one pointing to the next node which is at the same level as the current node, and the other pointer pointing to the next node which is the child of the current node. For example, the Chapter1 node in Figure 5.4 node is at level one and contains a pointer to the Chapter2 node as its next sibling and another pointer to the Introduction node (which is a level two lesson node) as its first child.

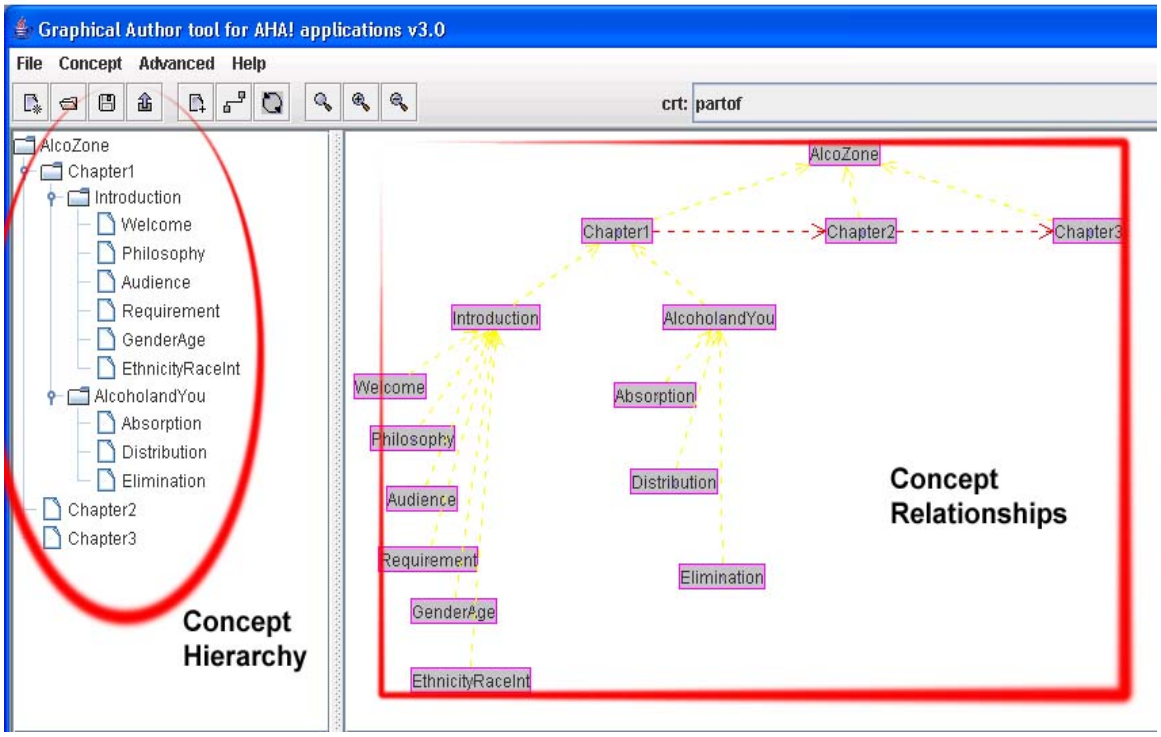


Figure 5.4: Concept Hierarchy in Graph Author

So far we have seen how three-level content can be authored using AHA's authoring tools. To support content hierarchy, one only needs to know the semantics of chapter, lesson, and concept. The delivery of individual concepts in the adaptation engine is not affected because of this authoring process. However, curriculum sequencing and link sorting explained in Section 4.4.3 and in Section 4.4.7 are based on this content division.

The concept relationships, shown on the right hand side in Figure 5.4, describe how two concepts are related. For example, one concept can be a prerequisite for another concept, or knowledge gained from visiting one concept also contributes to gain in knowledge of the related concept. AHA includes certain built-in concept relationships types which define the adaptation rules for knowledge update. In AlcoZone, however, we have not used the built-in relationship types because they attribute the change in knowledge to the visited status of concepts, which may not be applicable in many cases.

Now we explain how AlcoZone content was authored in the three levels we identified above.

AlcoZone 1.0 consists of three main learning units:

● Introduction to the Course: This gives introductory information about the course, including goals, audience, and requirements, and asks a few questions to the user.

● Standard Drinks Unit: This explains what a standard drink is, how it is measured, and some exercises for the user to calculate a standard drink.

● Tolerance Unit: Tolerance is the reduction in effectiveness of a drug. The Tolerance unit describes what alcohol tolerance is, its effects and types, how it is developed, and factors affecting tolerance.

The introduction unit is further divided into two sub-units. The first one gives preliminary information about the course and the second one gives an overview about alcohol absorption, distribution, and elimination. The Standard Drinks Unit consists of two sub-units. The first one consists of a series of questions that determine what type of drinker the user is. The second one describes different types of alcoholic beverages and amount of alcohol in each. Similarly, the Tolerance unit consists of five sub-units.

To support this content organization, we have defined three chapters in AlcoZone for the three main units. The introductory Chapter1 is divided into two lessons: Introduction and AlcoholandYou. Similarly, the second chapter on standard drinks is made up of two lessons: StandardDrinks & SDUContent. The third chapter on Tolerance is made up of five lessons. Each of these lessons consists of one or more concepts. A concept is the smallest learning unit that can be presented to the user. Information presented in a concept consists of text, images, and audio. All the information about a concept is presented on one page. This division of content also reduces the amount of content presented on one page which often causes disinterest of users to read (or listen to) all the information. For details of the AlcoZone Domain Model, please refer to Appendix A that includes a screen shot of the Domain Model in Graph Author. Appendix D includes the listing of concepts for the second and third chapter of AlcoZone.

5.3 Curriculum Sequencing

Providing a curriculum sequencing feature is a fairly easy task in non-adaptive E-Learning courses. In adaptive systems, providing a link to the next concept involves more than simple HTML anchors because the next content presented after the current concept changes from one user to the other and depends on the position of the user in the hyperspace.

There are two ways in which we have solved this problem in AlcoZone. If the name of the concept to be presented after the concept currently being visited can be determined based on the User Model state when the current concept is being processed, a link to the next concept is anchored in the current concept. This solution works for a majority of situations except in those cases where the system has to know a user response to the concept currently being processed before it can determine the next concept. To handle this particular situation, we still provide an HTML anchor to the next concept but it points to a *dummy* next concept with hidden parameters in the HTTP request so that the system can identify such requests. We call this a conditional jump between concepts and explain the details in the next section. Let us see the details of how we provide the link to the next concept.

5.3.1 Link to Next Concept

An anchor to the next concept is shown in the main view of AlcoZone (see Figure 5.3). The HTML request for presentation of MainView is sent by the user's browser to the ViewGet servlet. The details of this process are explained in Section 4.3. The ViewGet servlet gets a reference to the XHTMLHandler (or HTMLHandler) using the HandlerManager class. XHTMLHandler is responsible for parsing the XHTML file which contains textual information and links to images and audio file. When XHTMLHandler is done processing textual information it inserts an anchor tag (which is

wrapped by the *Next* button in Figure 5.3). To determine the concept name to which the anchor tag points, XHTMLHandler calls the *getNextNodeName* method defined in the extended AHA. This method is aware of the three level concept hierarchy and knows the current concept the user has requested. Using this information it traverses the Domain Model concept tree to determine the next node to visit. Let us see how *getNextNodeName* actually determines the name of the next concept to be visited using Figure 5.5.

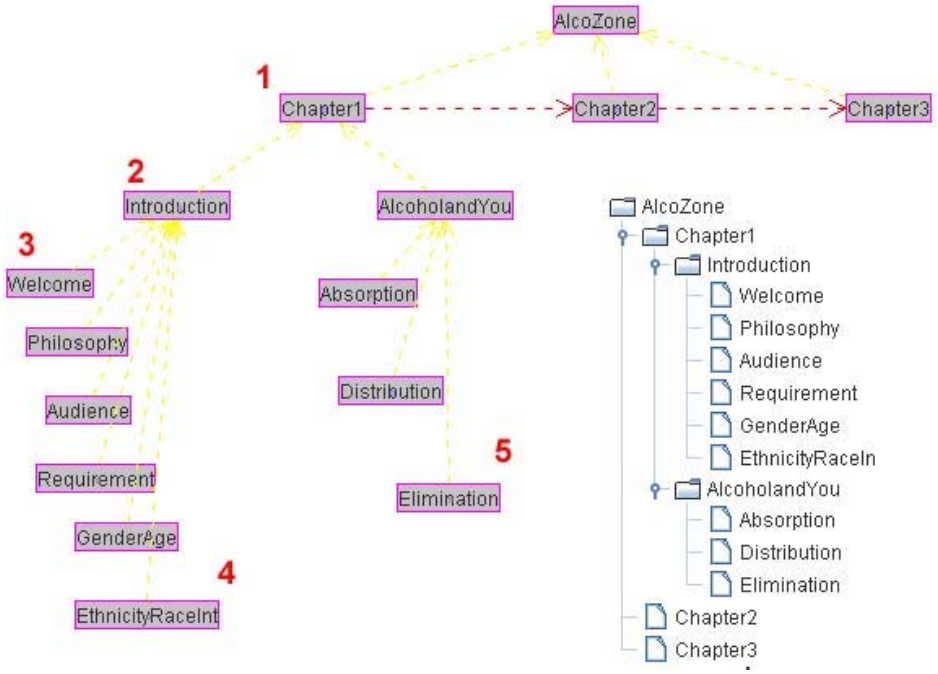


Figure 5.5: Concept Sequencing Scenarios

When a user is visiting the course for the first time, presentation starts with Chapter1. Irrespective of where you are in hierarchy, *getNextNodeName* constructs a path from the current node to the *root* of the hierarchy (AlcoZone node) and determines the level of the current node. The numbers 1 to 5 in Figure 5.5 identify the user's current location in the hyperspace. We shall show the next concept is determined assuming the user is at that point.

1. The system is processing a user request for the ‘Chapter1’ concept. In order to provide a link to the next concept on the ‘Chapter1’ page, getNextNodeName is called within XHTMLHandler. The level of ‘Chapter1’ node is one. The next concept to be presented is the first lesson in ‘Chapter1’, i.e., Introduction.

2. The currently requested concept is ‘Introduction’ which is a level two node (i.e., Lesson). The next concept to be visited is the first child of ‘Introduction’, i.e., ‘Welcome’.

3. The currently requested concept is ‘Welcome’ which is a level three node (i.e., Concept). The next node to be visited is the next sibling of the ‘Welcome’ node, i.e., ‘Philosophy’. Level three nodes do not have children. This sequence continues till the user reaches ‘EthnicityRaceInt’ concept, which is a child of ‘Introduction’.

4. The currently requested node is ‘EthnicityRaceInt’. This node neither has a child or a sibling at the same level. Before visiting any children, the parent node is already visited. So the algorithm determines the next sibling of the parent of the requested node, i.e., sibling of ‘Introduction’ node, which is the next lesson in Chapter 1, i.e., ‘AlcoholandYou’. After AlcoholandYou is visited, all of its children are visited in sequence till the user reaches the ‘Elimination’ node.

5. The current node is ‘Elimination’ which is the last level three node in the ‘AlcoholandYou’ lesson. At this point, all the lessons within ‘Chapter1’ have been visited. The next node to visit is the next chapter, i.e., ‘Chatper2’.

To summarize, concepts are visited in a depth first manner. The pseudo-code of the algorithm is explained in Figure 5.6.

```
Function getNextNodename
Input: user profile, HTTPServeletRequest
Output: name of next concept

Algorithm

1. Determine the level of currently requested node
2. If level =1 i.e. Chapater
      nextnode = first child of current node
      if nextnode = NULL
      nextnode = next sibling of current node
      if nextnode = NULL
      nextnode = currentnode  // this is last node in the course

3. If level = 2 i.e. Lesson
      nextnode = first child of current lesson
      If nextnode = NULL
      nextnode = next sibling of current lesson
      if nextnode = NULL
      nextnode = next sibling of parent of current node
      if nextnode = NULL
      There are no more nodes to visit. Return error code

 4. If level = 3 i.e. concept
      nextnode = next sibling of current concept
       if nextnode = NULL
       nextnode = next sibling of parent of current concept
       if nextnode = NULL
       nextnode = next sibling of grand parent of current concept
                  (i.e. next chapter)
       if nextnode = NULL
       There are no more nodes to visit. Return error code

Return name of nextnode

END
```

Figure 5.6: Concept Sequencing Algorithm

5.4 Conditional Jump between Concepts

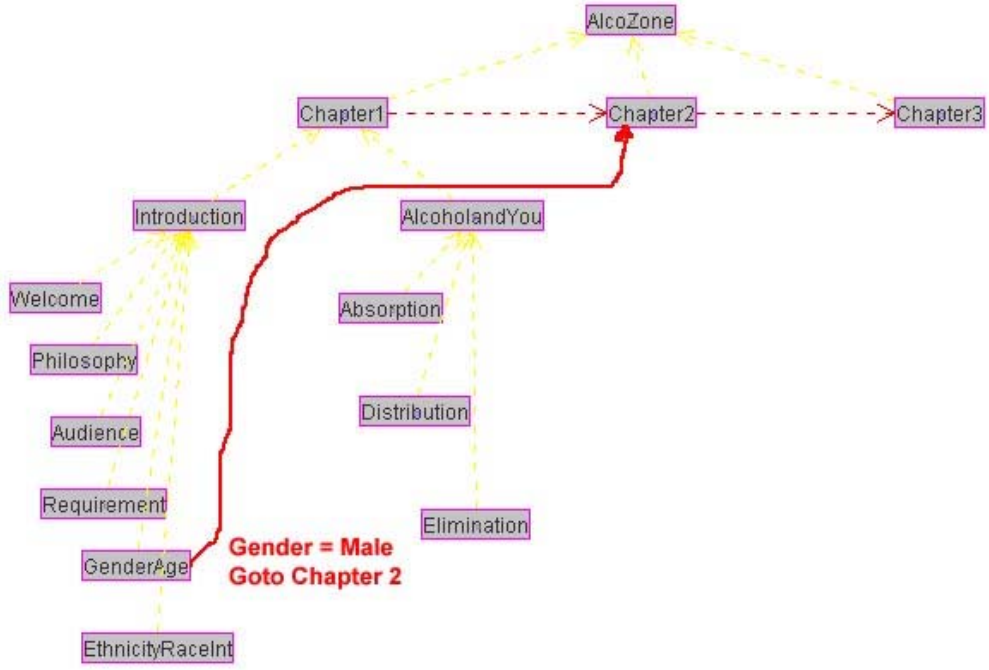


Figure 5.7: Jump between Concepts

The technique explained in the previous section works for all the concepts which do need to know the user response to the current (typically answers to questions) concept. For those concepts which ask questions to users and adapt the next concept to be presented based on the answer, we have used request parameters to differentiate them from other requests. We author these concepts using the custom defined concept types known as *option* concept and *nextfragment* concept. The XHTML file corresponding to these concepts have a hidden parameter *questionconcept* which is set 'true'. During the normal flow of operation, whenever a user clicks on the next button, the request is sent to the Get servlet which executes adaptation rules associated with the request concept, updates the User Model, and sends the response containing three HTTP requests to the ViewGet servlet corresponding to the three views (MainView, StaticTreeView, and TopView). These ViewGet request URLs contain the name of the concept to be presented which is the same as the concept requested originally in the Get request. For Get requests which have the *questionconcept* parameter set to true, the name of the concept is different from the one in the Get request. Whenever the Get servlet receives a request with

questionconcept set to 'true', it also receives questions and a corresponding user response to those questions. Immediately upon receipt of the request, the Get servlet updates the User Model with the response of the user to the questions. The *option* and *nextfragment* concepts have a special attribute *next*, which defines adaptation rules to determine the next concept depending on the user response to questions asked as part of that concept. The Get servlet executes these adaptation rules (using a helper class) and figures out the next concept name.

To execute the adaptation rule defined as part of the *next* attribute, we modified the adaptation rule execution module so that it can understand the modified format of adaptation rules.

The AHA adaptation rule execution module could only execute those rules whose action and condition involved use of valid concept names. We needed a way in which adaptation could be performed based on answers to the questions. Since our questions and answers were not a part of the Domain Model, we stored the answers to the questions in the User Model and assigned a key starting with "personal.answer.NAMEOFQUESTION". For example, a gender entry might look like[1]:

```
<requirement>personal.answer.gender=male</requirement>
      <trueActions>
       <action>
        <conceptName> AlcoZone.genderage</conceptName>
        <attributeName>next</attributeName>
        <expression>"AlcoZone.chapter2"</expression>
       </action>
      </trueActions>
      <falseActions>
           <action>
            <conceptName> AlcoZone.genderage</conceptName>
            <attributeName>next</attributeName>
            <expression>"AlcoZone.EthnicityRaceInt"</expression>
           </action>
      </falseActions>
```

The above adaptation rules are part of the next attribute of concept 'GenderAge'. It says that if the user's gender is male, the next concept to be presented for the user after

[1] This adaptation rule is for illustration purposes only and is not part of the current AlcoZone course.

'GenderAge' is Chapter2. If the gender is not male, then the next concept is 'EthnicityRaceInt'.

To parse and execute such adaptation rules, parsing and execution code of AHA was modified. Once the rules are parsed correctly, we referred to the User Model to determine if the expression evaluates to *true*. Based on the outcome of the evaluation, the appropriate next concept name is returned.

After the adaptation rule is executed, the Get servlet continues its normal operation with the name of the concept determined after the execution of adaptation rules The response sent to the user contains three ViewGet requests containing requests for newly determined concepts and not the one that was previously requested to Get servlet.

Use of the conditional jump technique is complementary to curriculum sequencing. Following the three level concept hierarchy defines a certain path in the hyperspace and does not allow us to skip certain content for some users who may already have knowledge about those concepts. By using conditional jumps, it is possible to present a subset of the entire Domain Model to different users depending on their previous understanding of the subject.

5.5 Questions and Exercises Distributed Throughout the Course

Many personalized E-Learning courses (including the first AlcoZone prototype) take an approach in which users take a pre-course questionnaire. Based on the user response, the system selects the learning objects to be presented to the user during the length of the course. This content selection is a one time process which happens at the beginning. However, this is an inadequate approach for two reasons. First, asking too many questions at one time or one followed by another might create disinterest on a user's part. As a result, some of the responses may not be correct. Second, the questionnaire itself might need to be personalized depending on answers to previous questions. So we a need a system that continuously monitors user response to questions and decides what should

be presented next (which could be explanation content or further questions). By including two custom concepts (option and nextfragment) that automatically update the User Model with the response to questions, it is possible to author certain concepts as question concepts and to distribute those evenly throughout the length of the course. At the same time, execution of adaptation rules associated with question concepts makes it possible to use the responses to questions to decide the next content.

5.6 User Interface Customization

In this section we discuss how the look and feel of the AlcoZone web based course can be dramatically changed on the fly using *skins* functionality. This functionality works in tandem with AHA layout functionality. Using Layout capability, AHA divides the browser space in multiple frames and maps different views in those frames. The HTML code for corresponding views is nothing more than plain HTML hard-coded in class files that are responsible for generating those views. To be able to change generated HTML on the fly, we needed to keep the HTML code separate from the class files. If we include the user interface and navigation code in XHTMLs, we will have to replicate it in all the files and furthermore we will not be able change to it easily. If we include the code in JAVA class files, we will not able to change the interface code on the fly.

To solve this problem, we keep the interface and navigation HTML code in an XML file and it is picked up at run time by parsing the XML file. Depending on which is the currently selected skin, different segments within the XML file are picked up. This gives us the flexibility of changing the user interface without changing the class implementation of views. We have implemented two custom views (TopView and DynamicTree) and modified the MainView and StaticTreeView of AHA in order to support more sophisticated presentation than the one provided by AHA. Design of a custom view has been discussed in Section 5.1. We now describe how skins are implemented.

We shall explain the skins functionality by considering a generic example as the specific details change from one view to another. See Figure 5.8.

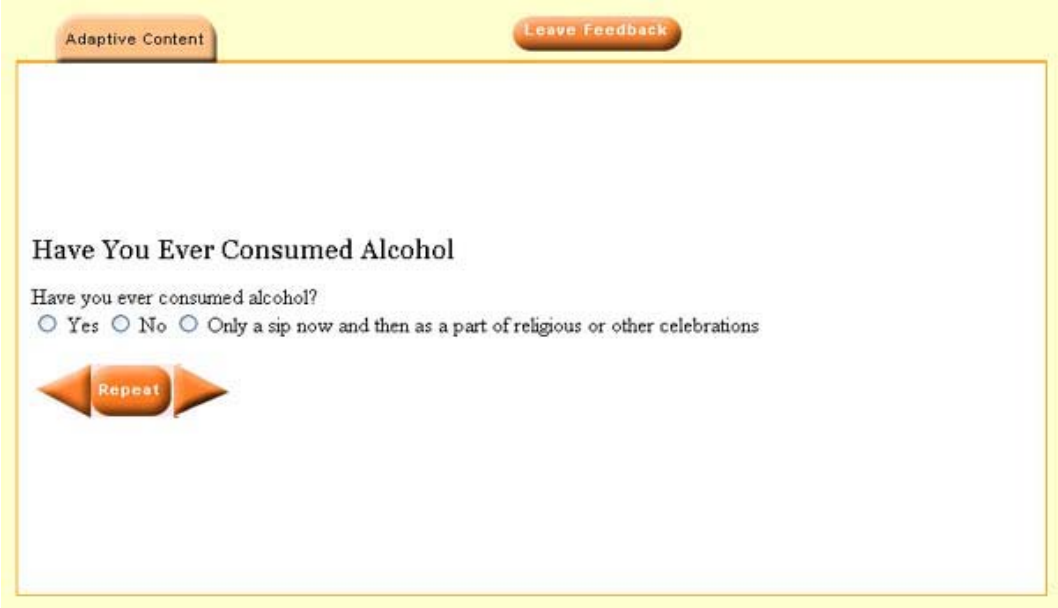


Figure 5.8: MainView generated using skins functionality

The simplified HTML code for the above view is shown in Figure 5.9. The yellow border, rectangles, and the two buttons at the top is the user interface code which is picked up from skins.xml. This code is followed by textual content which comes from the XHTML resource file associated with the concept. This code is shown in blue color in Figure 5.9. After the textual information, navigation code that includes links to the previous, next, and current concept is inserted from skins and shown in a black color in the figure. And finally, we close the inner white rectangle and any open tags (e.g., table, form). This simplified HTML code for the above figure can be divided into four segments. The first segment is the beginning of table layout code with two images: Adaptive Content and Leave Feedback. The second segment consists of textual content that is parsed and presented from the XHTML file. The third segment is the navigational code associated with the buttons. The last segment consists of closing tags for tables and borders. Together, these segments create a Yellow and Orange color combination and thus can be defined as one partial skin consisting of HTML code for MainView. Segments for all the views are combined together which creates one complete skin. It is possible to create markedly different looking skins as long as they contain the same number of segments defined in other skins for that view.

The class that handles HTML code generation for one view is aware about different segments for that view. This class parses the XML file containing the skin definition and inserts the retrieved HTML code at appropriate places in the output HTML stream. Sometimes it is necessary to replace some text within a segment. For example, Navigation Buttons Next, Previous, and Repeat point to different concepts throughout the course. Thus, it is necessary that they point to the current concept depending on the user's location in the hyperspace. The name of the concept to which navigation anchors point has been highlighted in red color in Figure 5.9. The skin representation of navigation code contains placeholders at places where dynamic string replacement is necessary. These placeholders are of the form *%placeholdername%*. The name of the placeholder changes depending on its purpose. After a segment is retrieved, the view generation class replaces the placeholder with a value, which in this is the name of the concept (i.e., survey4.everconsumed). A simplified structure of skins.xml for one view is shown below. This consists of three fragments. This skins.xml does include a fragment named 'second' because this is picked up from the XHTML file. This segment is shown in blue color in Figure 5.9.

Structure of SKINS.XML

```
<skin name="yellow">
      <view name="MainView">
            <fragment name="first"> User Interface Code <table color="FFDDCC"> ..
             </fragment>
             <fragment name="third"> navigation code with placeholders
             </fragment>
             <fragment name="fourth">   ….  </table>
             </fragment>
       </view>
       .
       .
       code for another view
       .
</skin>
```

```
USER INTERFACE SPECIFIC CODE

Content from XHML File
<h1>Have You Ever Consumed Alcohol</h1>
<p> Have you ever consumed alcohol? <br />
 <input name="everconsumed" type="radio" value="yes"> Yes </input>
 <input name="everconsumed" type="radio" value="no"> No </input>
 <input name="everconsumed" type="radio" value="onlyasip">
  Only a sip now and then as a part of religious or other celebrations
 </input>
</p>
XHTML Content ENDS

Navigation Code
<A HREF="/aha/Get?logid=2&concept=survey4.genderage" >
 <IMG SRC=" /aha/xmlroot/skin/yellow/images/backup.jpg"
 NAME="previous">
</A>

<A HREF="/aha/ViewGet/v3?logid=3&treeview=true&
  concept=survey4.everconsumed">
  <IMG SRC="/aha/xmlroot/skin/yellow/images/repeat.jpg"
   NAME="repeat">
</A>

<A HREF="#" javascript="getNextConcept()">
 <IMG SRC=" /aha/xmlroot/skin/yellow/images/frontup.jpg"
  NAME="next">
</A>
Navigation Code ENDS

USER INTERFACE SPECIFIC CODE
```

Figure 5.9: HTML code segments for a view

5.7 Content Listing Sorted on Visited Status

Figure 5.10 is a snapshot of the StaticTreeView that contains the course content menu. This view is implemented using one more view known as DynamicTree. The StaticTreeView includes vertical tabs and HTML anchors for the number of chapters present in the Domain Model. The presentation of concepts preceded by red and green balls in the figure is done by the DynamicTree view. Whenever a user clicks on the

chapter number, it makes a request to DynamicTree and includes a parameter that indicates whether the request is for visited content or unvisited content. Let us see how DynamicTree constructs a concept listing assuming the request is for visited content. When the request is received, the DynamicTree view consults the User Model to find out all the lessons for the selected chapter that have been marked as visited. For each of the selected lessons, it determines all the child concepts that have been marked as visited. It groups these concepts according to the lesson they belong to and creates the HTML presentation code using the interface code defined in the skins.xml. A similar logic is followed for the creation of the unvisited content listing, except in this case DynamicTree creates a listing of those concepts that have not been marked as visited in the User Model.

Figure 5.10: Sorted content listing based on visited status

5.8 Delivery of Non-Adaptive Static HTML pages

To support various dimensions of E-Learning such as collaboration, communication, and distributed learning [23], it is desirable that a web based E-Learning site presents content pages that do not relate to concepts. In AHA, it was possible to deliver two types of content pages: those that are resources for concepts (using the parsing capability of XMLHandler & HTMLHandler), and static web pages that could be directly served using a web server. In some cases, however, it is necessary to display some static web pages with partial customization (e.g., displaying the user name on a page or providing statistical information about a user's performance). Creation of concept using authoring tools is not a good idea in this case because these pages are not semantically related to a Domain Model. And delivering a static HTML page will not allow us partial content customization. To present these web pages, we have used parsing code of XMLHandler and HTMLHandler. By assigning a special URL request parameter, we differentiate request for a static HTML page with partial customization from requests for a concept. Since XMLHandler & HTMLHandler are part of the adaptation engine, they can access the User Model and the combined Domain/Adaptation Model to customize the HTML content. We have used this extension to present vertical tab menu files such as glossary, preferences, and help files.

5.9 Audio Narration for Textual Content

Users prefer to learn using different types of instruction media. Some users like to see text and images while others prefer audio and video. In order to give users the flexibility in choosing an instruction media, we have added audio narration for all textual content. This audio narration can be turned ON or OFF by default. The user preference for audio narration is stored in the User Model. If a user has indicated that he would like to hear audio narration, XHTML parsing code inserts HTML code for audio controls and includes a link to the audio file. In order to avoid high costs of audio recordings, audio narrations are automatically generated from text using a text-to-speech engine. To ensure that audio sounds like human speech, we have used AT&T natural voices [35].

5.10 Logging Navigation History

For each user the system creates two types of log files. The first is the User Model which stores information about each visited concept. This information consists of knowledge gained of the concepts as a result of visiting them, and the user preferences. The second logging function, which we have added, captures the navigation history for a user. It records those user clicks that send requests to the adaptation engine (i.e., the ViewGet and Get servlets). Each of these request URLs contain a log ID using which we can uniquely identify the origin of a request. Once a request for the next page is received, the logging module creates a log entry in the XML file for that user.

The log entries have following format:

```
<record>
  <actionid>1</actionid>
  <accessdate>Mon Apr 17 13:26:05 EDT 2006</accessdate>
  <sessionid>AAD003BB79FF8A374D4A51295778C9D9</sessionid>
  <name>survey4.standarddrinks</name>
  <name>file:/survey4/xml/standarddrinks.xhtml</name>
  <difficulty>medium</difficulty>
</record>
```

The *actionid* uniquely identifies a click on the user screen. In the above example, *actionid* 1 means that the user has clicked on the 'Next' button. The name of the requested concept is '*survey4.standarddrinks*' and the resource associated with this concept is standarddrinks.xhtml. The difficulty level of the requested content is 'medium'[1]. Users can take the AlcoZone course in multiple sessions and *sessionID* identifies the session when the request was generated. Using *accessdate*, it is possible to tell the date and time when the request was made. The number of entries in the access log is roughly equal to the number of user clicks which send request to the adaptation engine.

[1] We have included some preliminary support to present content with different difficulty levels. However, this feature is not completely implemented in the current version.

Chapter 6: Evaluation

AlcoZone started with the idea to create adaptive and personalized alcohol education course using a framework that is not closely tied in with alcohol education. The motivational factor behind the extensions in AHA is to increase the usability of adaptive courses. None of the extensions we have developed are specific to alcohol education, and are applicable for any course developed using the extended AHA framework.

There are three aspects that make an adaptive E-Learning system effective. The first is the content itself and its organization. The second is how effectively the system adapts content for different users and whether the adapted content is suitable for users. The last and the most important aspect is the user friendliness and usability of the system [8] [10]. These aspects of effectiveness can be measured using qualitative and quantitative feedback from users. The quantitative aspect of effectiveness is the improvement in user performance which can be determined using test scores. The qualitative aspect of effectiveness can be measured using users' opinion. Evaluation of AlcoZone consists of qualitative evaluation to determine what users think about adaptive and personalized alcohol educational system. Some of the things that we wanted to know are:

● Is there any difficulty in using the system.

● How users navigate in the course.

● Does the system present the most appropriate content to the users that is suitable for their background.

● What do users think about the extensions provided in AHA.

To answer these questions, we conducted a formative evaluation using a web based Post-Survey. AlcoZone course is to be taken by approximately 5,000 freshman students of Virginia Tech every year. Providing in-person instructions to every individual about how

to take the course in a production environment is beyond the administrative resources available. So we invited the research participants using email and provided them with a web URL to the course. The invitation email contained the information about how to access the course and the system requirements. This situation is similar to how the course will be taken in the production environment. For the formative evaluation, we invited 100 students to take the online course. Students could login to the course using their university ID and password and the course would begin with Chapter one. Students did not have to take the Pre-Course questionnaire as was the case with the first prototype of AlcoZone. There were a total of two evaluations conducted. Feedback received from the first evaluation was used to modify certain features of AlcoZone. Users' reaction to the modified system was tested with the second evaluation. In this section, we discuss the results of the two evaluations.

6.1 First Evaluation

A total of 100 students participated in the first evaluation. In the following sections, we discuss the results of the evaluation.

6.1.1 Overall Evaluation

This area of questionnaire determined the overall rating of AlcoZone which relates to whether the system is successful in giving information about alcohol use that students find helpful (see Figure 6.1). The majority of students agreed that the course was very educative and user friendly (77%) but very few considered it be an excellent resource on alcohol education (7%). 2% of the students said that the system is confusing and 5% believed it is poor.

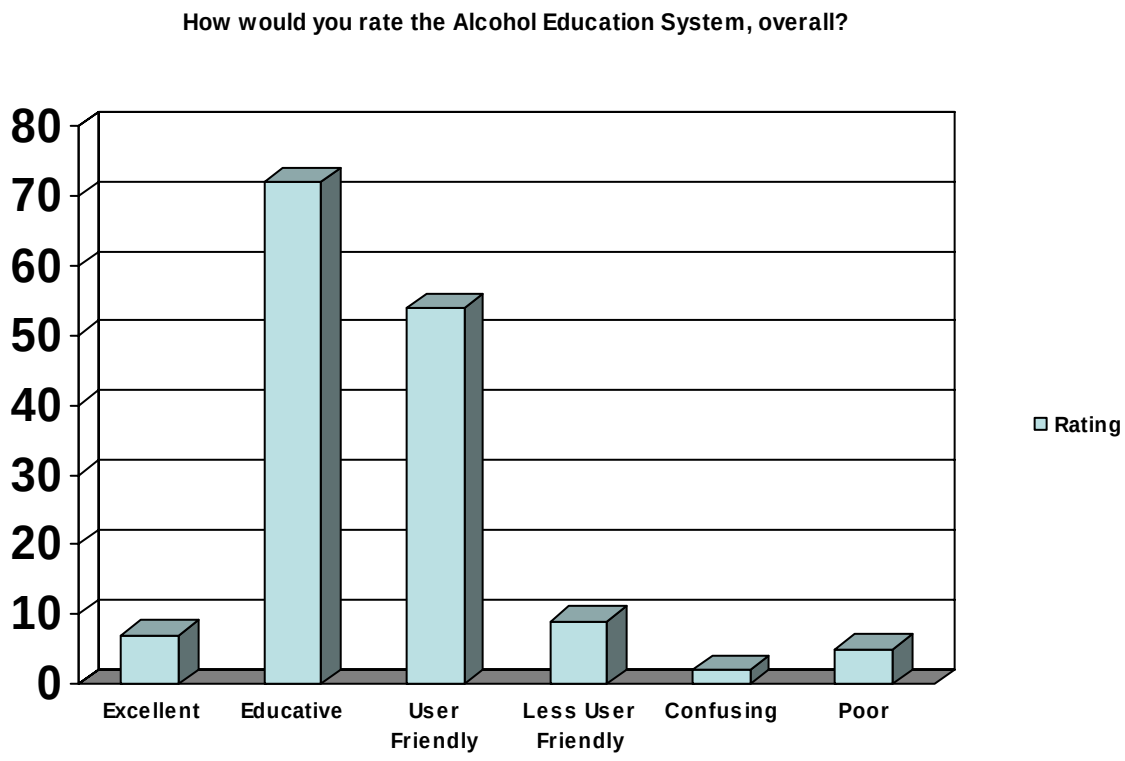


Figure 6.1: Overall rating of AlcoZone

The small number of people who considered AlcoZone as 'excellent' commented that the division and flow of content was very logical and presented using an easy to navigate interface. The users who found AlcoZone less user friendly, confusing, and poor were not satisfied with browser compatibility, a concept page that asks users to enter their drinking log, and content proofreading. The encouraging aspect of positive comments is, the extensions implemented in AHA have largely been successful. This is because users ended up using curriculum sequencing instead of browsing the content and found the division of content in the three level hierarchy logical to follow. The shortcomings either relate to the presentation of content or certain usability aspects.

6.1.2 Content Adaptation

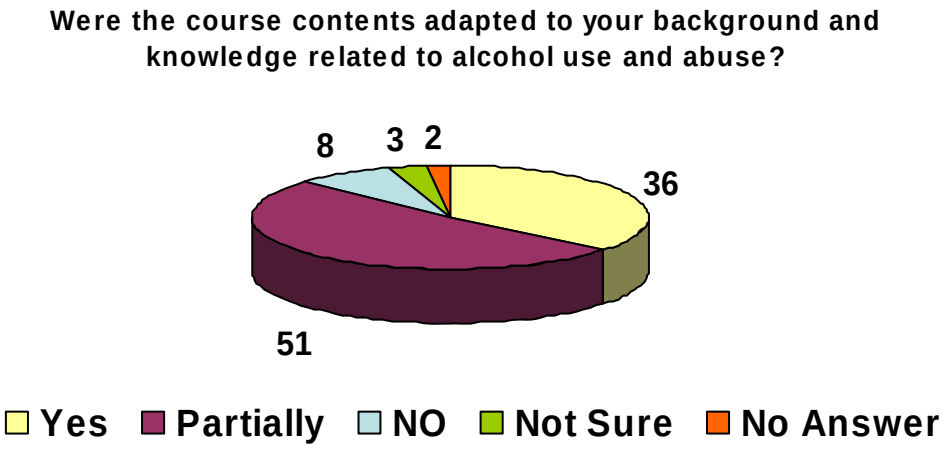


Figure 6.2: User feedback of content adaptation

In Figure 6.2, we have shown the user response to content adaptation. Though users may not directly know that the content is being 'adapted' for them, they can certainly judge whether the content is relevant for them. For example, to high risk drinkers, AlcoZone presents information that will help them reduce their high-risk drinking behavior. And abstainers are presented with information so that they can prevent high risk drinking scenarios. When asked if users think that the presented content was adapted for their background, 87% user agreed that it was. 36% felt clearly that the content was adapted and 51% felt it was partially adapted. 8% users said that the content was not adapted. The difference in the way system determined drinker categorization and user's perception about their own drinking behavior was the main reason why few users felt that the content adaptation was inadequate. For example, if someone drinks only as part of religious celebrations, the system categorizes that person as an 'occasional' drinker[1]. However, these users indicated that they should be categorized as 'abstainers'.

[1] There was an error in the drinker categorization for the first few users in the evaluation study. The system incorrectly categorized a user as a high-risk drinker if he/she drinks only as part of religious activities. This was corrected after 25% of the evaluation was over.

6.1.3 Content Division

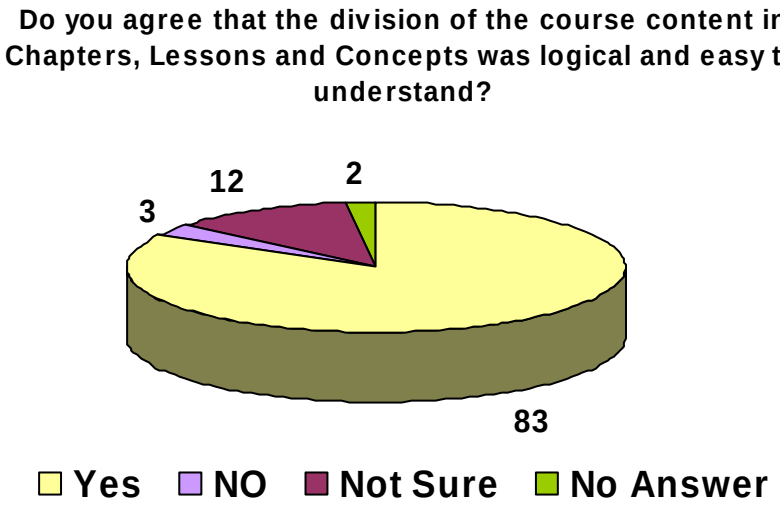


Figure 6.3: Opinion about content division

Figure 6.3 represents the response of users to content division. The majority of users (83%) unanimously agreed that the division of content in different learning units was logical and easy to understand. This confirms our belief that the content organized in pedagogical learning units is easier to follow than the simple link annotation scheme implemented in AHA. Only 3% of the users felt that the content division was not easy to follow. This content division also meant there was less content on a single page, something that many users appreciated. However, some of the content pages contained very little information and it was suggested that such screen space can be occupied with more valuable information for users.

6.1.4 Difficulty Level of Content and Knowledge Gain

The AlcoZone 1.0 content gives the most basic information about alcohol use. Thus, many users found the content easy to understand. 66% users (see Figure 6.4) felt that the difficulty level of the content was easy and 29% found it be of medium level. Only 2%

users found the content difficult to understand. This also implicates that 95% of the users were able to understand the information given in the course. To test the knowledge of users about alcohol use, we asked them to rate their knowledge on a scale of 10, before and after the course. The mean of these values gives us a score of 6.61 for the knowledge before taking the AlcoZone course and a score of 8.31 after the course. This indicates an average increase of 25% in the knowledge of alcohol use.

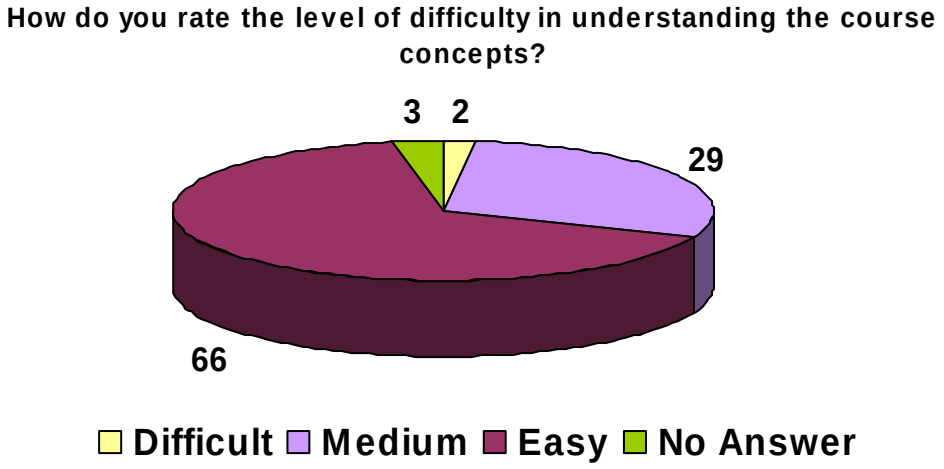


Figure 6.4: Difficulty level of content

6.1.5 Key Findings

There are a number of key findings that arise from the formative evaluation. Content adaptation and personalization is much more useful for students than the delivery of content that is the same for everyone. The use of multiple media is beneficial as users have different preferences for instruction media. However, users do not prefer audio that is computer generated and would like to hear real human voice. Inclusion of interactive exercises makes the system interesting. Logical content division lets users concentrate on smaller learning units at a time. Most of the extensions we have developed are transparent to users. Comments for improvement indicate that it is important that these

extensions are presented in a very user friendly way. For example, the navigation for curriculum sequencing using 'next' and 'previous' buttons was very intuitive. Similarly, the browsable course content listing was not as intuitive. Table 6.1 summarizes the key findings from the evaluation. For a more detailed user feedback about the strengths and weaknesses of AlcoZone, please refer to Appendix B.

What users liked	What can be improved
Ease of understanding	Computer generated voice
Flow of content from one concept to another	Browser compatibility
Division of content in chapter, lesson and concepts	Exercise on drinking log using calendar
Interactive exercises	Grammatical mistakes in textual content
Ease of use	Usability of course content menu
Content adaptivity	Course layout for different screen resolutions

Table 6.1: Strengths and weaknesses of AlcoZone

After the feedback from the formative evaluation, we identified those areas of improvement that were not specific to alcohol education and were inherent in the extended AlcoZone framework. These are the compatibility of AlcoZone on multiple browsers, presentation of the layout for multiple screen resolutions, and the usability of the content menu. The browser compatibility and presentation for multiple resolutions were addressed by changing the HTML code in skins.xml and did not require changes in the modified framework. Apart from these enhancements, we also worked on providing more intuitive option for turning ON or OFF audio narration, which many people found annoying because of machine generated voice. Analysis of navigation log for individual users allowed us to conclude that users are interested in moving forward in the course using the 'next' button and the use of right hand side expand & collapse menu was minimal. To remedy this, we redesigned the link structure and provided images to differentiate between visited and unvisited concepts. In the formative evaluation, the course content menu frame was not expanded by default. This menu window was

expanded whenever a user clicked on any of the vertical tabs located at the right hand side of the screen. This was changed such that the content menu frame is displayed as long as the user does not close it. The style of link anchors to concepts and lessons was changed such that the text for the concept name is underlined. We thought that this will help users identify HTML links which are normally presented using underlined text in web browsers.

6.2 Second Evaluation

After the improvements to AlcoZone described in the previous section, we conducted a second evaluation using the similar process for the first evaluation. However, for the second evaluation only 30 students participated. Following are the results of the evaluation:

6.2.1 Overall Rating

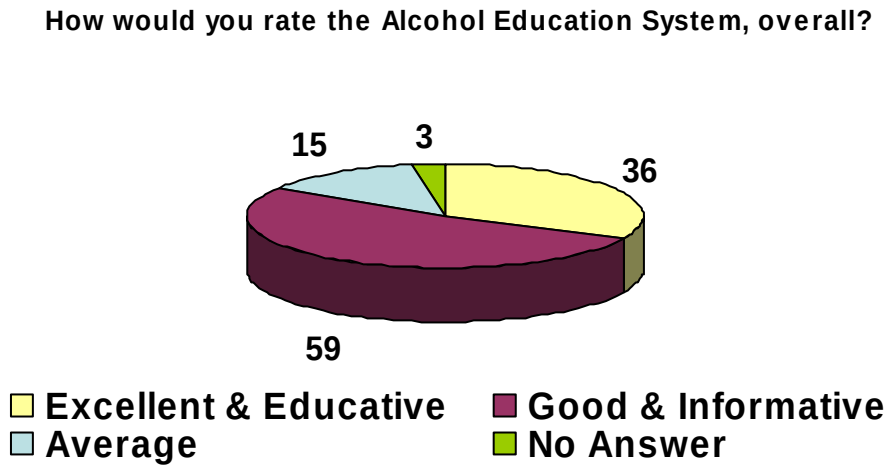


Figure 6.5: Overall rating of AlcoZone in the second evaluation

Figure 6.5 describes the overall rating of AlcoZone. When asked what the users think about AlcoZone, 36% users felt that it is an excellent and educative system and 59% users thought it is good and informative. The interesting observation is none of the users rated AlcoZone as poor or mediocre when these options were present. Another encouraging aspect is that there was an increase in the number of people who gave top rating to AlcoZone compared to the first evaluation. The people who gave top rating commented ease of use, division of content, and questions interspersed between concepts as some things that they liked.

6.2.2 Content Adaptation

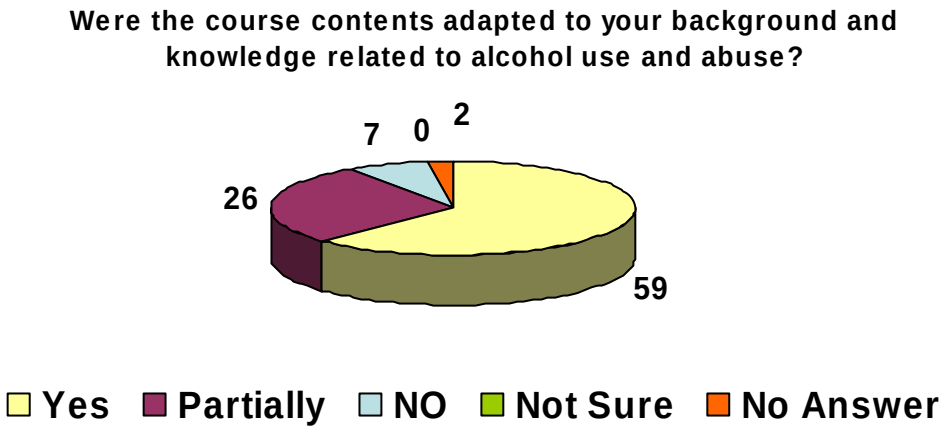


Figure 6.6: Content adaptation in second evaluation

After the second evaluation, the users felt more confident that the content is getting adapted for their background and knowledge. When asked if the content is getting adapted, 59% users said yes and 26% users indicated that the content was partially adapted (see Figure 6.6). The number of users (7%) who felt content was not adapted for their background didn't vary much from the first evaluation (8%). The total number of users who said that the content was adapted has stayed relatively constant at 85% between the two evaluations. However, in the second evaluation, there was a decline in the number of users who felt that the content was partially adapted. This was surprising considering the adaptation rules were not modified for the second evaluation. The only change was in the user interface which always showed the course content listing that was updated based on the 'visited' status, as the user moved in the hyperspace.

6.2.3 Content Division

Users' opinion about the content division in chapters, lessons, and concepts did not vary much between the two evaluations. The user response to content division is represented in Figure 6.7. After the second evaluation, 77% users felt that the content division was logical and easy to follow. The number of users who were not sure about the relevancy of content division remained relatively constant at 13%. However, the figures do indicate a small decline of 6% in the number of users who thought content division was appropriate. But an interesting observation is that there were only 3% users in both the evaluations who felt the content division was not appropriate.

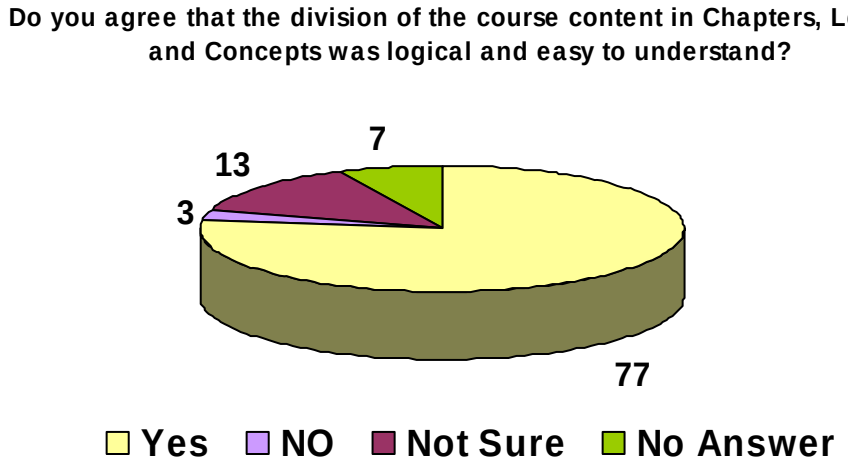


Figure 6.7: Relevancy of content division

6.2.4 Navigation and User Friendliness

User friendliness and navigation is an important aspect of E-Learning system which makes it more usable. The questions about navigation and user friendliness attempt to know if the users are easily able to navigate in the course and do they think that the system is user friendly. 77% users felt that the navigation provided in the course was intuitive (see Figure 6.8). This represents an increase of 12% after the first evaluation. The number of people who rated the navigation as confusing was zero which represents a

decline of 9% over the first evaluation. The number of people who were not sure about the navigation also declined from 23% to 17%. The user friendliness of the system was rated as excellent by 48% of the user and 26% thought that it was good. 15% users considered it be average and 4% rated it as below average.

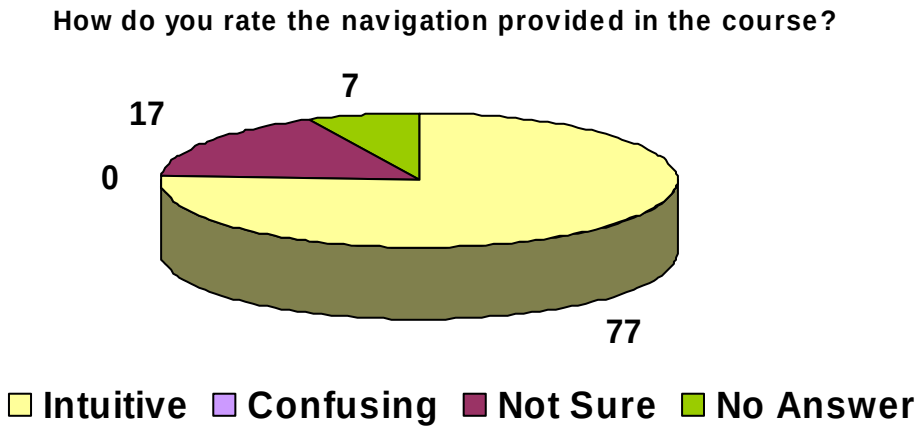


Figure 6.8: Navigation in the course

6.2.5 Key Findings

The second evaluation showed a reasonable increase in the number of users who gave a higher rating to the overall system, content adaptation, and navigation aspects. The majority of comments for improvement were related to the usability of learning objects delivered within the AlcoZone framework. Looking at the browsing pattern of users, it can be concluded that the users want to get done with the course in the shortest possible time. Users still felt that the audio quality was poor. However, our objective behind the inclusion of audio was to test users' receptiveness to the inclusion of audio and

potentially video. We included one Flash exercise where the users had to arrange various alcoholic beverages according to the amount of alcohol they contain. Users had to drag the beverages and arrange them in order from the lowest alcoholic content to the highest. This was a very interactive exercise and many users commented that they liked this exercise and would prefer inclusion of more such interactive exercises. This shows that the interactivity of E-Learning content makes learning interesting for users. However, it will be interesting to see what percentage of the total content that is interactive, is acceptable to achieve the balance between interactivity and learning.

We conclude this section by comparing the results from the two evaluations in Table 6.2. In the next section, we shall see how the course contents and usability of the system can be improved by analyzing user logs.

How often do you drink?	**Every Week**	**Once a month**	**Occasional**	**Do not drink**	**No answer**
First evaluation	29%	5%	36%	27%	3%
Second Evaluation	40%	0%	27%	27%	7%
Were the course contents adapted to your background and knowledge?	**Yes**	**Partially**	**No**	**Not sure**	**No answer**
First evaluation	36%	51%	8%	3%	2%
Second evaluation	53%	33%	7%	0%	7%
How do you rate the level of difficulty in understanding the course concepts?	**Easy**	**Medium**	**Difficult**	**No answer**	
First evaluation	66%	29%	2%	3%	
Second evaluation	67%	27%	0%	7%	
Do you agree that the division of course contents in chapters, lessons, and concepts logical and easy to understand?	**Yes**	**No**	**Not sure**	**No answer**	
First evaluation	83%	3%	12%	2%	
Second evaluation	77%	3%	13%	7%	
How do you rate the navigation in the course?	**Intuitive**	**Confusing**	**Not sure**	**No answer**	
First evaluation	65%	9%	23%	3%	
Second evaluation	77%	0%	17%	7%	
How do you rate the audio quality?	**Excellent**	**Mediocre**	**Poor**	**No answer**	
First evaluation	20%	59%	19%	2%	
Second evaluation	30%	57%	7%	7%	

Table 6.2: A comparison of evaluation results

6.3 Analysis of User Profiles

An analysis of user profiles and navigation logs allows the authors and administrators to understand the usability of the system and discover interesting correlations between user characteristics and performance. The inclusion of data mining support in AHA is discussed in [9] and [22]. As a result of the two evaluations, we collected user preference and performance data in the User Model files. A User Model file contains the user characteristics (e.g., age, gender, drinker type, etc.) and answers to questions. We combined all the user profiles and ran data mining algorithms to determine if there are any correlation patterns between the user characteristics and the performance. An evaluation version of Magnum Opus [34] data mining tool was used to analyze the user profiles. We were particularly interested in the discovery of association rules which determine the associations/correlations in a large data set. For example, a supermarket database includes 1000 transactions. 600 of these transactions include the purchase of 'A' and 600 more transactions include the purchase of 'B' (i.e., 200 transactions include purchase of both). Out of the 200 transactions that include purchase of both A and B, 180 include the purchase of 'C'. The association rule for this can be:

A and B → C

The above rule has a support of 180 transactions (or 18% i.e. 180/1000) and a confidence of 90% (180/200).

We were interested in the discovery of such association rules among the user profile attributes. There were a total of 164 user profile files in the system and each user profile file contained approximately 35 attributes[1]. The attributes in the user profile file can be divided in two categories: personal information about the user and answers to question concepts. We selected age, race, international student status, drinker type, and ethnicity from the personal characteristics. The other personal characteristics gave more detailed information such as the age when first alcoholic drink was consumed and the negative

[1] The number of attributes in the user profile varies depending on the course completion and path followed in the hyperspace.

experiences as a result of alcohol consumption. For the purpose of preliminary data mining, this information was considered too detailed and was not used in the data mining task. In order to know the user performance in questionnaires, we included the number of correct answers in exercise one and exercise two. The selected 11 user attributes are described below:

- Gender: Gender of user.
- Age: Age of user.
- Race: Race of user.
- International: Is user an international student.
- Ethnicity: Ethnicity of user.
- Skins: Selected user interface skin.
- Audio: Audio narration ON/OFF status.
- DrinkerType: The drinker category of user.
- EverConsumed: Has user ever consumed alcohol.
- ExerciseOne: Correct number of answers to exercise one.
- ExerciseTwo: Correct number of answers to exercise two.

After preprocessing the user profiles, an input file for the data mining tool was created which consisted of the eleven attribute values separated by comma for one user profile. Values for different users were separated using a new line character (‘\n’). After running the tool on the data file, 59 association rules were discovered with varying level of support and confidence. In Table 6.3, we describe those rules which we found interesting with the associated support and confidence.

Association Rule	Confidence	Support
2<=ExerciseTwo<=3 is associated with 2<=ExerciseOne<=3	84%	46%
DrinkerType=highrisk is associated with International=no	100%	26%
DrinkerType=occasional is associated with Age<22	93%	18%

Table 6.3: Discovery of association rules from user profiles

The first rule says that if a user has scored a high number of points in the first exercise, his performance in the second exercise is also very good. The second rule says that there is no high-risk drinker who is an international student. The last rule states that most of the 'occasional' drinkers' age is below 22. Reliability and accuracy of association rules increases with the size of input data. One limitation of our data analysis is the relatively small size of input data by the data mining standards. However, as AlcoZone goes into production, a large database of user profiles shall be collected which will enable discovery of more meaningful rules with better support and confidence. Such rules can be used to improve the content and the adaptation effects.

6.4 Summary

This chapter has discussed the evaluation of AlcoZone across various areas. The first section discussed the formative evaluation of AlcoZone and described the strengths and weaknesses inherent in the system. The major shortcoming that was inherent in the AlcoZone framework was in the presentation of course content menu. Browser compatibility and layout for multiple resolutions was remedied using modifications in the HTML code in skins.xml. The feedback specific to the extensions we have developed has largely been encouraging. The majority of the users found curriculum sequencing and the associated navigation (i.e., next, previous, and repeat) intuitive and easy to use. The division of content in chapters, lessons, and concepts resulted in a modular and logical

organization of content. The ability to change the interface was demonstrated by the use of ‘skins’ which makes it possible for users to learn the concepts using a layout that they prefer. Logging capabilities allowed us to observe how users are navigating in the course and this has helped in improving the usability of the system. An analysis of user profile logs has allowed us to discover interesting patterns between user characteristics and performance. The knowledge gained from this has enabled the authors to restructure the Domain Model to make the content more relevant and informative for users. In the next chapter, we shall conclude this thesis, discussing its achievements and future work.

Chapter 7: Conclusion

This thesis has presented the extensions in AHA that were done as part of the AlcoZone course so as to bring Adaptive Hypermedia based courses from research demonstrations to a large number of users. The research demonstrates how the requirements of AlcoZone translated in the use of Adaptive Hypermedia technology, the selection of the research tool to build upon, and the specific extensions implemented. The applicability and usability of the extensions and effectiveness of the course based on Adaptive Hypermedia technology was illustrated from the results of the evaluation study. This chapter summarizes the thesis describing its objectives, achievements, and future work.

7.1 Objectives and Achievements

The driving force behind the development of AlcoZone was the research and implementation of technologies that would allow creation of personalized and adaptive content. Since personalized and adaptive content is more beneficial than static content, applicability of the approach to create courses in other subjects was equally important. The main objectives of AlcoZone were:

- Provide a framework for personalized and adaptive course authoring and delivery.
- Separate the content adaptation from its presentation.
- Create a reusable framework that can be used for creation of other courses.
- Provide more support for adaptive content presentation.
- Present content using an easy to navigate and user friendly interface.
- Integrate questionnaires as part of the course content and distribute the questions throughout the course.
- Make it possible to change the user interface on-the-fly.
- Allow easier integration of audio, video and interactive learning objects such as Flash.
- Allow the users to take the course in multiple sessions.
- Build on a framework that is flexible and extensible so that the addition of E-Learning support tools (e.g., collaboration, communication etc.) is possible.

In order to achieve these objectives in the given timeframe, identification of the most suitable technology and tools was the most crucial task. AHA, based on Adaptive Hypermedia technologies, was found to be the most suitable tool considering its flexible, extensible, and modular architecture. Reusability of the framework, separation of adaptation from content, and ability to take a course in multiple sessions were the inherent benefits in the selection of AHA. However, this thesis demonstrates the relevancy of extensions implemented through successful user evaluations.

Content division in chapters, lessons, and concepts makes the division more logical and when coupled with appropriate content authoring, makes the flow of content easy to understand. When provided with curriculum sequencing support, users tend to navigate in sequence instead of browsing through the hyperspace. Users certainly appreciate content adaptation but perfect adaptation remains illusive considering the content cannot be adapted for all goals, knowledge levels, and backgrounds. This is because the amount of adaptation is restricted to the amount of content available in the Domain Model and the adaptation rules in the Adaptation Model. This was reflected in the user evaluation where a small number of users felt that the content was not adapted for their needs. Thus, adaptive systems need to aim for presentation of content that is 'closest' to the users' needs. Even after the system is able to present the most appropriate content to users, it is important that the content is presented using a user friendly and easy to navigate interface. The user friendliness of available features is far more important than availability of more features. Users have different preferences for instruction media and an option for users to select the instruction media of choice allows them to learn using the most effective media. But content that consists of only text, images, and audio/video may not be able to keep the interest level high all the time. Inclusion of interactive elements can alleviate this problem. Logging capabilities of AlcoZone captured the navigation behavior in the access log and user preferences & performance in the User Model. Interesting observations have been made by analyzing User Models about the user characteristics and performance. The knowledge gained from this has allowed the authors to identify the content that can be explained in an alternate way and given insights to developers to improve the usability of the system.

7.2 Future Work

There are many areas in which AlcoZone can be extended both in terms enhancements to the current framework and inclusion of tools that are useful in an E-Learning environments.

7.2.1 Enhancements to the framework

Currently, the adaptation engine adapts content using the adaptation rules. The adaptation rules consult the User Model to select the best content from the Domain Model. This can be extended to include a Knowledge Repository constructed using data mining techniques that combines the user model and/or access log. This Knowledge Repository can consist of rules that define how users' performance changes for different user characteristics. Depending on the characteristics of the current user, appropriate explanation of concepts can be presented that will help improve the user performance. Secondly, the course content menu can be improved both in terms of usability and adaptation. Currently, AlcoZone does not use AHA's relationships for knowledge update because these rules were based only on the visited status of concepts. This can be changed such that knowledge values are changed depending on users' answers to questionnaires. Based on this, course content menu can display only those concepts for which the user has sufficient prerequisite knowledge. As seen in Section 6.3, an analysis of user profiles and navigation logs can greatly help authors and administrators to improve the course content and usability of the system. Use of data mining techniques on user profiles and navigation logs for better personalization has been shown in [27][28][29] [16]. Our initial work on data mining on user profiles lays the foundation for inclusion of such systems in future. And lastly, AHA's authoring tools can be modified to support our extensions in a more user friendly manner.

7.2.2 Inclusion of External Tools

The other area of future work lies in the inclusion of external tools that have been proven effective in an E-Learning environment. These include support for distributive and collaborative learning as found in PlanetMath [20], and integration with Course Management Systems (CMS) such as Sakai [14] and Moodle [15]. Currently, most of the CMS do not include support for adaptive learning and on the other hand Adaptive Hypermedia Systems lack the features of a CMS [32]. Merger of these two technologies will enhance the learning experience. The integration of APeLS with Moodle has been shown in [25]. A similar integration of AlcoZone with a suitable CMS is one area that needs to be explored.

In summary, this thesis has shown how an Adaptive Hypermedia based educational system can be used for better learner satisfaction and the key things to consider in the development of adaptive courses.

Bibliography

[1] Paul De Bra, Ad Aerts, Bart Berden, Barend de Lange, Brendan Rousseau, Tomi Santic, David Smits, Natalia Stash, ”*AHA! The adaptive hypermedia architecture*”, in Proceedings of the fourteenth ACM conference on Hypertext and hypermedia, Nottingham, UK, August 2003, pp 81-84.

[2] Paul De Bra, Peter Brusilovsky, Geert-Jan Houben,”*Adaptive Hypermedia: From Systems to Framework*”, in ACM Computing Surveys, Volume 31, Issue 4es, December 1999.
< http://www.cs.brown.edu/memex/ACM_HypertextTestbed/papers/25.html >

[3] Nor Aniza Abdullah, Hugh Davis, “*Is Simple Sequencing Adaptive Hypermedia?*”, in Proceedings of the fourteenth ACM conference on Hypertext and hypermedia, Nottingham, UK, 2003, pp 172-173.

[4] Dongsong Zhang Lina Zhou Jay F. Nunamaker, Jr., “*Can e-Learning replace classroom learning?*”, in Communication of the ACM, Volume 47, Issue 5, May 2004, pp 75-79.

[5] Owen Conlan and Vincent Wade, “*Evaluation of APeLS - An Adaptive eLearning Service based on the Multi-model, Metadata-driven Approach***”,** in Third International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems, Eindhoven, The Netherlands, August 23-26, 2004.

[6] Dagger, D., Wade, V., Conlan, O., (2004), "*A Framework for Developing Adaptive Personalized eLearning*", in G. Richards (Ed.), World Conference on E-Learning in Corporate, Government, Healthcare and Higher Education, Chesapeake, VA, 2004, pp 2579-2587.

[7] Ewald Ramp, Paul De Bra, Peter Brusilovsky, "*High-level translation of adaptive hypermedia applications*", in Proceedings of the sixteenth ACM conference on Hypertext and hypermedia, Salzburg, Austria, August 2005, pp 126-128

[8] Carmelo Ardito**,** Rosa Lanzilotti**,** Teresa Roselli**,** Veronica Rossano, Marilena De Marsico**,** Stefano Levialdi**,** Manuela Tersigni, **"***Usability of E-Learning Tools*" in Proceedings of the working conference Advanced Visual Interfaces, Gallipoli, Italy, May 25-28 2004, pp 80-84.
< http://portal.acm.org/citation.cfm?id=989873 >

[9] Cristóbal Romero, Sebastián Ventura, Carlos de Castro, Wendy Hall, and Muan Hong Ng, "*Using Genetic Algorithms for Data Mining in Web based Educational Hypermedia Systems***",** in Workshop on Adaptive Systems for Web-based Education, Malaga, Spain, May 2002.
<http://wwwis.win.tue.nl/~acristea/HTML/Minerva/papers/elearn02RomeroDeBraVenturaDeCastro.pdf >

[10] Mark Notess, "*Usability, user experience, and learner experience*", eLearn Magazine, Volume 2001, Issue 8, 2001, pp 8.

[11] A. Cristea and C. Stewart "*Automatic Authoring of Adaptive Educational hypermedia*", book chapter to appear in "Web-Based Intelligent e-Learning Systems: Technologies and Applications", Zongmin Ma., IDEA publishing group 2005.
< http://wwwis.win.tue.nl/~acristea/WBI/chapter-cristea-stewart27.doc >

[12] Cristea, A.I., De Mooij, A., "*LAOS: Layered WWW AHS Authoring Model and their corresponding Algebraic Operators*", in Proceedings of WWW'03, Budapest, Hungary, ACM, IW3C2, May 2003.
< http://citeseer.ist.psu.edu/656680.html >

[13] A. Cristea, L.Calvi, "*The three Layers of Adaptation Granularity*", User Modelling 2003, Johnstown, USA, June 2003, LNAI 2702, Springer -Verlag, 2003, pp 4-14.

[14] Sakai Course Management System, July 2005. <www.sakaifoundation.org>

[15] Moodle Course Management System, August 2005. < www.moodle.org >

[16] Andrej KRIŠTOFIČ, M. Bieliková (Ed.),"*Recommender System for Adaptive Hypermedia Applications*", IIT.SRC 2005, April 27, 2005, pp. 229-234.
< http://www2.fiit.stuba.sk/iit-src/2005/35-kristofic.pdf >

[17] Gerhard Weber, Hans-Christian Kuhl, and Stephan Weibelzahl, "*Developing Adaptive Internet Based Courses with the Authoring System NetCoach*", In P. DeBra, P. Brusilovsky, and A. Kobsa, Eds., Proceedings of the Third Workshop on Adaptive Hypertext and Hypermedia, Sonthofen, Germany, Technical University Eindhoven, 2001.
< http://wwwis.win.tue.nl/ah2001/papers/GWeber-UM01.pdf >

[18] Brusilovsky, P., Schwarz, E., and Weber, G. *A tool for developing adaptive electronic textbooks on WWW*, In Proceedings of WebNet'96 - World Conference of the Web Society, San Francisco, MA, USA, June 1996, pp 64-69.

[19] Brusilovsky, P., Schwarz, E., & Weber, G.. "*ELM-ART: An intelligent tutoring system on World Wide Web*" in Frasson, C., Gauthier, G., & Lesgold, A. (Ed.), Intelligent Tutoring Systems (Lecture Notes in Computer Science, Vol. 1086) Berlin: Springer Verlag. 1996, pp 261-269.

[20] Aaron Krowne, "*An Architecture for Collaborative Math and Science Digital Libraries*", Master's Thesis, Department of Computer Science, Virginia Tech, 2003.
<http://planetmath.org >

[21] Owen Conlan, "*The Multi-Model, Metadata Driven Approach to Personalised eLearning Services*", Doctoral Thesis, University of Dublin, Trinity College.
< https://www.cs.tcd.ie/Owen.Conlan/ >

[22] Cristóbal Romero Morales, Sebastián Ventura Soto, Cesar Hervás Martínez, Paul de Bra, "*Extending AHA!*", University of Cordoba, Spain and Eindhoven University of Technology, The Netherlands.
< http://www.win.tue.nl/~debra/ichsl2005/ExtendingAHA.pdf >

[23] Stojanovic, L. , Staab, S., and Studer, R. "*Elearning based on the Semantic Web*", in World Conference on the WWW and Internet (WebNet), Orlando, Florida, USA, 2001.
< http://www.aifb.uni-karlsruhe.de/WBS/Publ/2001/WebNet_lstsstrst_2001.pdf >

[24] Abhijat Kanade, "*AlcoZone Project Report*", Independent Study Report, Department of Computer Science, 2004.

[25] Micheál Tiarnaigh, "*Adaptive Moodle*", Final Year Project, Department of Computer Science, Trinity College, Dublin, May 2005.

[26] Weber, G. and Brusilovsky, P. "*ELM-ART: An adaptive versatile system for Web-based instruction",* International Journal of Artificial Intelligence in Education, Volume 12, Issue 4 (2001), pp 351-384.
< http://www2.sis.pitt.edu/~peterb/papers/JAIEDFinal.pdf >

[27] S Perugini, MA Gonçalves, "*Recommendation and Personalization: A Survey",* Department of Computer Science, Virginia Tech, May 2002.
<http://homepages.udayton.edu/~perugisa/papers/access/RSsurvey/RSsurvey-perugin.pdf
>

[28] D. Pierrakos, G. Paliouras, C. Papatheodorou and C.D. Spyropoulos, " *KOINOTITES: A Web Usage Mining Tool for Personalization*" in Proceedings of the

Panhellenic Conference on Human Computer Interaction (PC-HCI), Patras, 2001, pp. 231-236.

[29] Osmar R. Zaiane, "*Building a Recommender Agent for e-Learning Systems*", International Conference on Computers in Education (ICCE'02), volume 1, 2002, pp 55-59.

[30] Wu, H., Houben, G.J., De Bra, P. "*AHAM: A Reference Model to Support Adaptive Hypermedia Authoring*", in Proceedings of the Conference on Information Science, Antwerp, 1998, pp.51-76.

[31] Peter Brusilovsky, "*Methods and techniques of adaptive hypermedia*", User Modeling and User Adapted Interaction, Kluwer, 1996, v6, pp 87-129.

[32] Brusilovsky, P. "*Adaptive hypermedia: From intelligent tutoring systems to Web-based education*", In G. Gauthier, C. Frasson, & K. VanLehn (Eds.), Intelligent tutoring systems. Lecture notes in computer science (Vol. 1839, pp. 1–7). Berlin, Springer Verlag.

[33] A Verhoeven, K Warendorf, "*External Navigation Control and Guidance for Learning with Spatial Hypermedia*", Journal of Interactive Media in Education, v 99, issues 1, March 1999.
<http://www-jime.open.ac.uk>

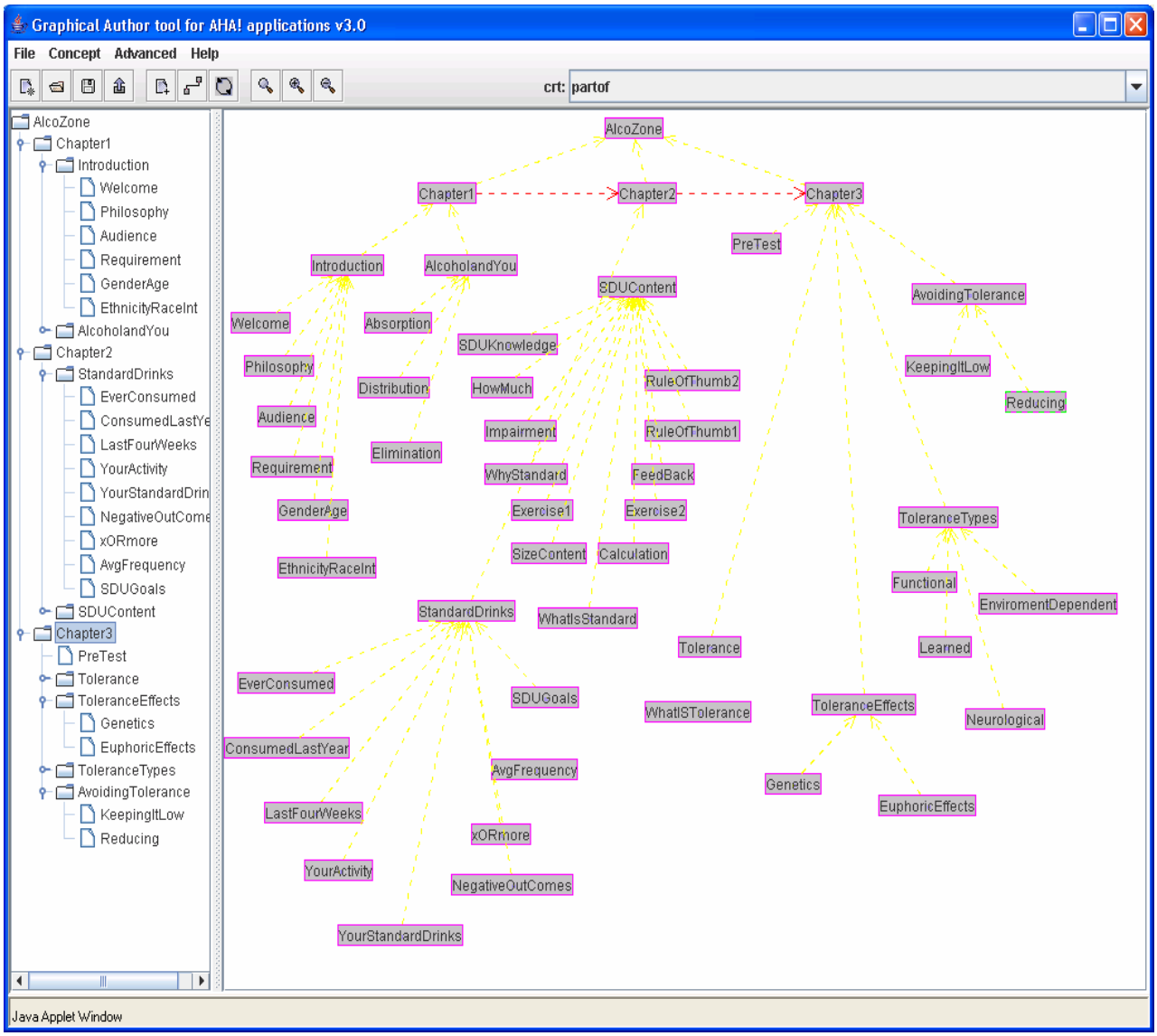
[34] Magnum Opus, April 2006. <http://www.rulequest.com/MagnumOpus-info.html>

[35] AT& T Natural Voices, February 2006. <http://www.naturalvoices.att.com>

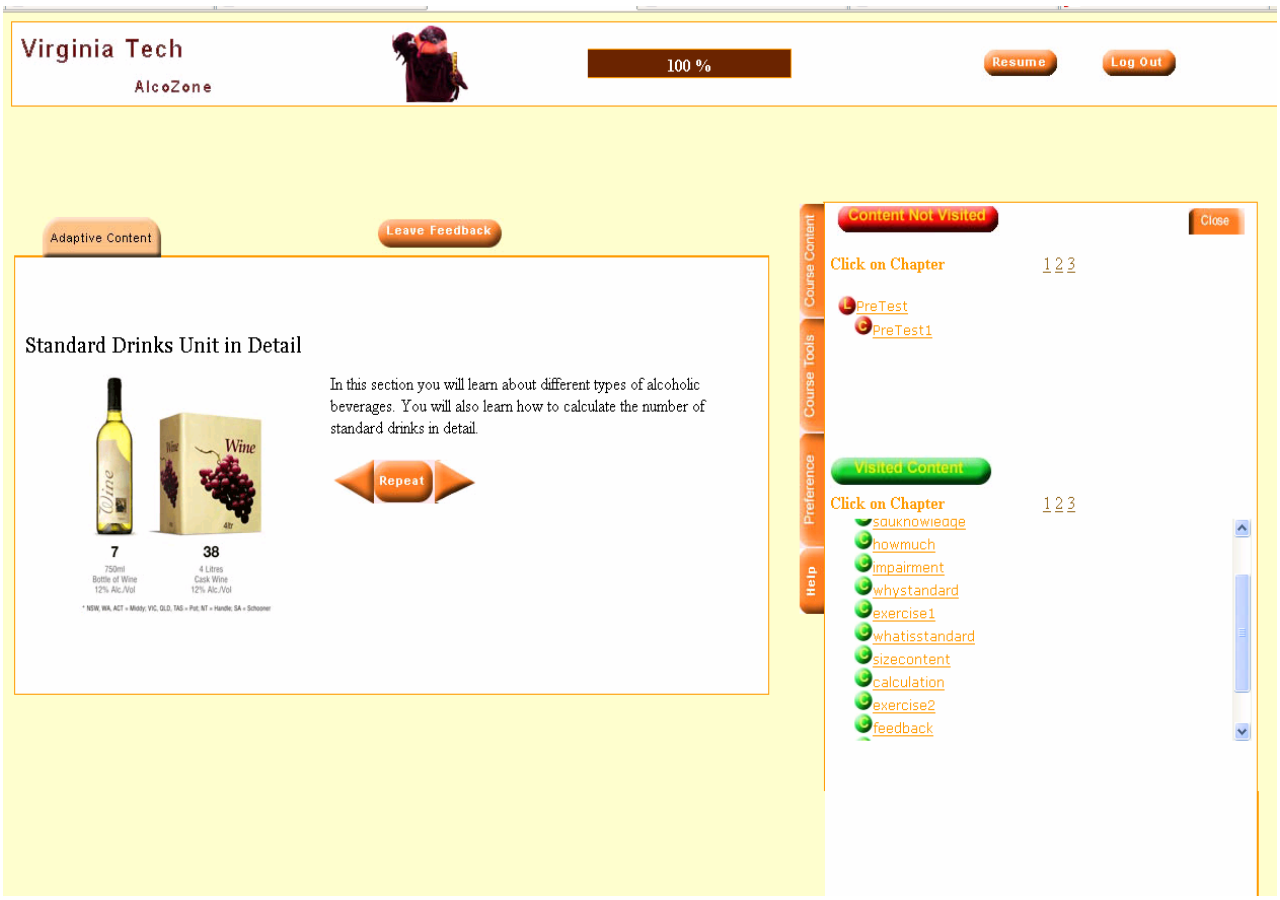
Appendices

Appendix A: Screenshots

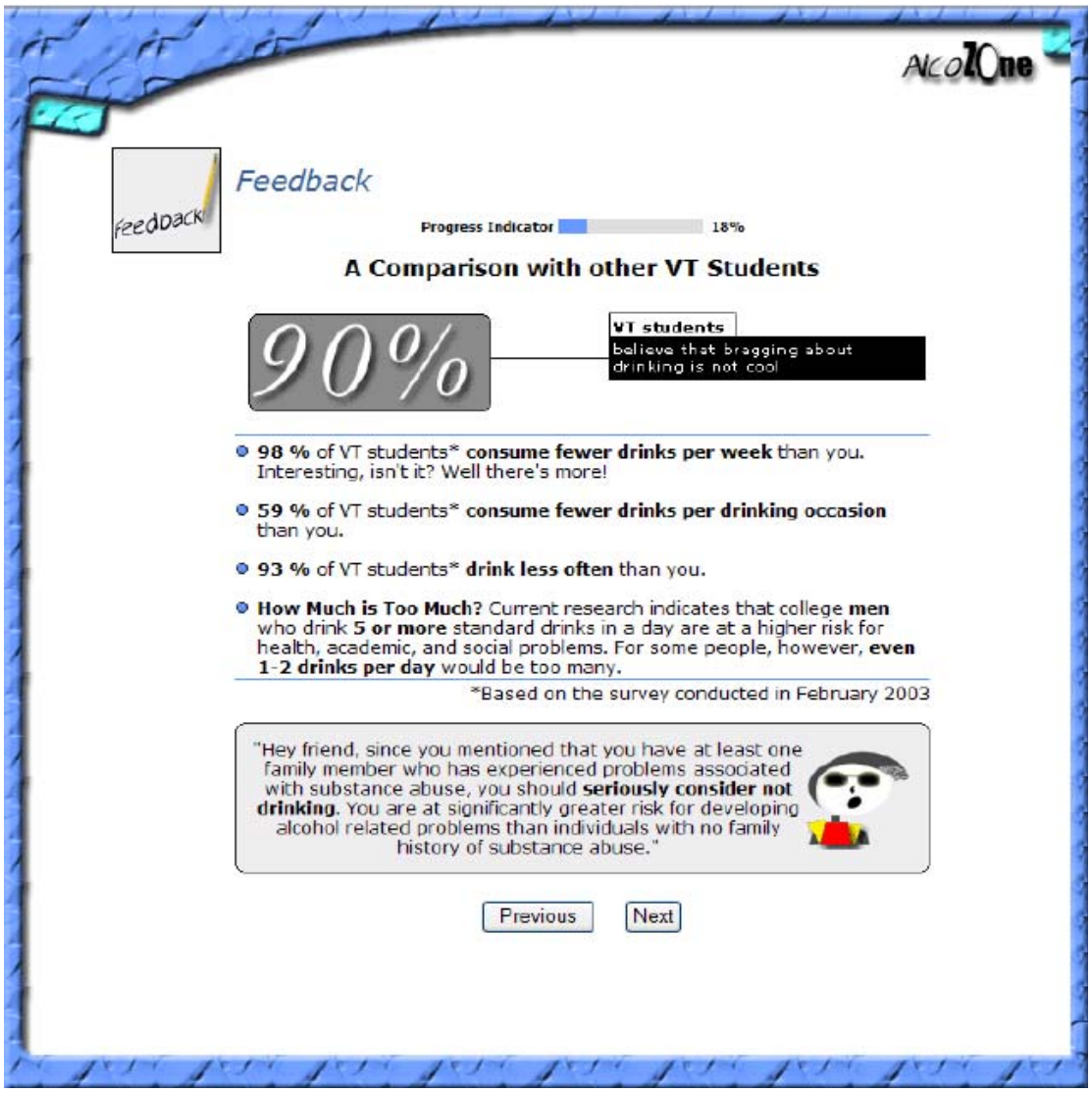
AlcoZone Domain Model



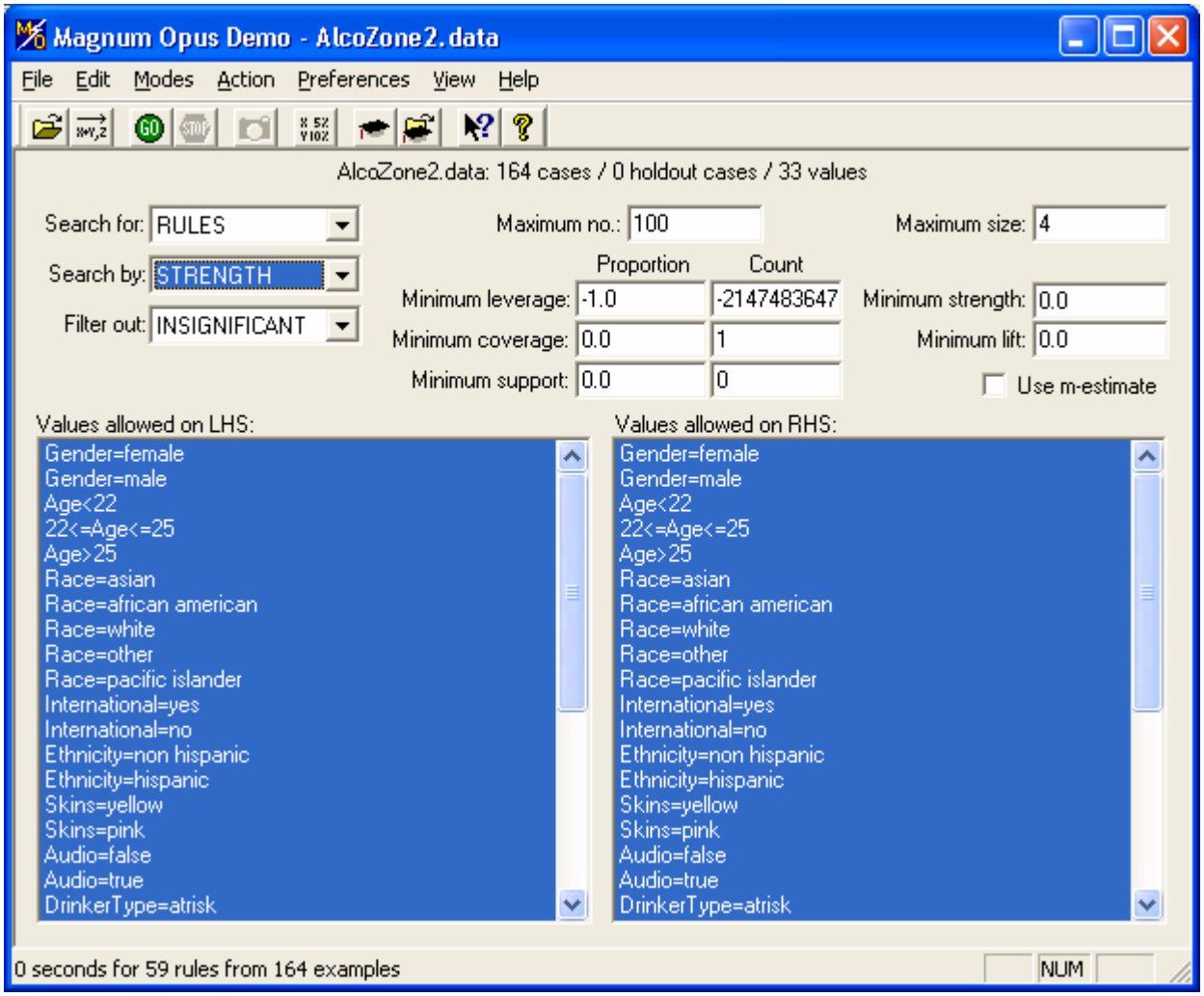
AlcoZone with content menu expanded



AlcoZone prototype



Magnum Opus


Magnum Opus Demo - AlcoZone2.data
File Edit Modes Action Preferences View Help
AlcoZone2.data: 164 cases / 0 holdout cases / 33 values
Search for: RULES
Search by: STRENGTH
Filter out: INSIGNIFICANT
Maximum no.: 100
Maximum size: 4
Proportion
Count
Minimum leverage: -1.0
-2147483647
Minimum strength: 0.0
Minimum coverage: 0.0
1
Minimum lift: 0.0
Minimum support: 0.0
0
Use m-estimate
Values allowed on LHS:
Values allowed on RHS:
Gender=female
Gender=male
Age<22
22<=Age<=25
Age>25
Race=asian
Race=african american
Race=white
Race=other
Race=pacific islander
International=yes
International=no
Ethnicity=non hispanic
Ethnicity=hispanic
Skins=yellow
Skins=pink
Audio=false
Audio=true
DrinkerType=atrisk
0 seconds for 59 rules from 164 examples
NUM

Appendix B: AlcoZone Evaluation

WEB-BASED CONSENT FORM

PURPOSE OF EXPERIMENT: You are invited to participate in a study intending to evaluate a adaptive web-based alcohol education system.

PROCEDURES TO BE FOLLOWED IN THE STUDY: Information from questions and surveys will be used to enhance the adaptivity of course contents to suit different user's cognitive needs and learning styles and improve the overall usability of the system.

ANONYMITY OF SUBJECTS: We will collect no information that could positively identify you.

CONFIDENTIALITY: Data collected for this study will be treated confidentially.

RISKS FROM PARTICIPATING IN THE STUDY: Because we have taken steps to protect your anonymity, there are no known risks to participation in the study.

EXPECTED BENEFITS: Participants get a compensation of $10 for participating in the evaluation study.

FREEDOM TO WITHDRAW: You are free to withdraw your consent and terminate your participation at any time. You are also free to decline to answer any specific items in the post-evaluation questionnaire.

USE OF RESEARCH DATA: The information from this research may be used for scientific or educational purposes. It may be presented at scientific meetings and/or published and reproduced in professional journals or books, or used for any other purpose that Virginia Tech's Department of Psychology considers proper in the interest of education or research. You will not be identified in any presentation or publication of the information collected.

APPROVAL OF RESEARCH: This research project has been approved by the Institutional Review Board of Virginia Tech.

SUBJECT PERMISSION: I have read the above description of the study I hereby acknowledge the above and give my web-based consent for participation in this study. I further understand that if I participate I may withdraw at any time without penalty. I understand that, should I have any questions regarding this research and its conduct, I should contact any of the persons named below:

Principal Investigators:	**Dr Edward Fox, fox@vt.edu**
	Steven Clarke, sclarke@vt.edu
	Devdutta Bhosale, devdutta@vt.edu
	Veena Basavraj veenabs@vt.edu
	Kashmira Phalak kvphalak@vt.edu
Human Subjects Committee Chair:	**David Harrison, Ph.D., 231-4422**
Institutional Review Board Chair:	David Moore, DVM, 231-4991

ACCEPT AND SUBMIT

Post-Evaluation Questionnaire

Gender

- Male
- Female

Age

- <18
- 18-19
- 20-25
- >25

Ethnicity

- Hispanic
- Non-Hispanic

Knowledge of Alcohol Use before taking the Course on a scale of 1-10(with 1 considered Poor and 10 considered Excellent)

Enter number between 1-10

Knowledge of Alcohol after taking the Course on a scale of 1-10 (with 1 considered Poor and 10 considered Excellent)

Enter number between 1-10

How often do you drink?

- Every Week
- Once a Month
- Occasional
- do not drink

Your thoughts and inputs will help us improve the system and suit the needs of the students while they learn about Alcohol Use and Abuse. Please provide your input for the following questions.

How would you rate the Alcohol Education System, overall?

- Excellent
- Educative
- User Friendly
- Less User Friendly
- Confusing

- [ ] Poor

Were the course contents adapted to your background and knowledge related to alcohol use and abuse?

- [ ] Yes
- [ ] Partially
- [ ] No
- [ ] Not Sure

Do you agree that the division of the course content in "Chapters", "Lessons", and "Concepts" was logical and easy to understand?

- [ ] Yes
- [ ] No
- [ ] Not Sure

How do you rate the audio clarity provided by the system?

- [ ] Excellent
- [ ] Mediocre
- [ ] Poor

Would you prefer to see more images and graphical content along with textual information?

- [ ] Yes
- [ ] No
- [ ] Not Sure

Would you prefer to see course contents presented in the form of videos?

- [ ] Yes
- [ ] No
- [ ] Not Sure

Would you prefer to see more detailed explanation and information about certain course concepts?

- [ ] Yes
- [ ] No
- [ ] Not Sure

Would you prefer the system to recommend course contents based on the preferences of other users whose background and knowledge match that of yours?

- [ ] Yes, often
- [ ] Yes, Occasionally
- [ ] No
- [ ] Not Sure

How do you rate the level of difficulty in understanding the course concepts?

- [ ] Difficult
- [ ] Medium
- [ ] Easy

How do you rate the level of interactivity the user has with the system?

- [ ] Sufficient & Useful
- [ ] Not Useful

How do you rate the Navigation provided in the Course

- [ ] Intuitive
- [ ] Confusing
- [ ] Not Sure

How do you rate the questions asked as part of the exercises?

- [ ] Meaningful
- [ ] Challenging
- [ ] Simple
- [ ] Irrelevant

Was the "GLOSSARY" helpful in learning concepts and completing exercises?

- [ ] Yes
- [ ] No
- [ ] Not Sure

Would you prefer the exercises to be made more competitive by including rewards (gift vouchers) for the high scores?

- [ ] Yes
- [ ] No
- [ ] Not Sure

Was the right-hand side expand collapse menu intuitive and easy to use

- [ ] Yes
- [ ] No

Did the "HELP MENU" have sufficient details to guide the user in using the system?

- [ ] Yes
- [ ] No
- [ ] Not Sure

Would you prefer a Virtual Buddy to guide you through the course features rather than the help menu?

- [ ] Yes
- [ ] No
- [ ] Both

What did you like the most about the Alcohol Education System?

If there is anything you would like to suggest as an add-on or as an improvement to the current system features, please specify?

Detailed Evaluation Findings

Gender
Male (77%) Female (21%) no answer (2%)

Age
<18 (1%)
18-19 (35%)
20-25 (58%)
>25 (4%)
no answer (2%)

Ethnicity
Hispanic(2%) Non-Hispanic (93%) no answer (5%)

How often do you drink?
Every Week (29%)
Once a Month (5%)
Occasional (36%)
Do not drink (27%)
no answer (3%)

How would you rate the Alcohol Education System, overall?
Excellent (7%)
Educative (72%)
User Friendly (54%)
Less User Friendly (9%)
Confusing (2%)
Poor (5%)

Were the course contents adapted to your background and knowledge related to alcohol use and abuse?
Yes (36%)
Partially (51%)
No (8%)
Not Sure (3%)
no answer (2%)

Do you agree that the division of the course content in "Chapters", "Lessons" and "Concepts" was logical and easy to understand?
Yes (83%)
No (3%)
Not Sure (12%)

no answer (2%)
How do you rate the audio clarity provided by the system?
Excellent (20%)
Mediocre (59%)
Poor (19%)
no answer (2%)

Would you prefer to see more images and graphical content along with textual information?
Yes (52%)
No (33%)
Not Sure (11%)
no answer (4%)

Would you prefer to see course contents presented in the form of videos?
Yes (23%)
No (59%)
Not Sure (16%)
no answer (2%)

Would you prefer to see more detailed explanation and information about certain course concepts?
Yes (36%)
No (41%)
Not Sure (20%)
no answer (3%)

Would you prefer the system to recommend course contents based on the preferences of other users whose background and knowledge match that of yours?
Yes, often (12%)
Yes, Occasionally (63%)
No (12%)
Not Sure (10%)
no answer (3%)

How do you rate the level of difficulty in understanding the course concepts?
Difficult (2%)
Medium (29%)
Easy (66%)
no answer (3%)

How do you rate the level of interactivity the user has with the system?
Sufficient & Useful (77%)
Not Useful (21%)
no answer (2%)

How do you rate the Navigation provided in the Course

Intuitive	(65%)
Confusing	(9%)
Not Sure	(23%)
no answer	(3%)

How do you rate the questions asked as part of the exercises?

Meaningful	(42%)
Challenging	(6%)
Simple	(68%)
Irrelevant	(7%)

Was the "GLOSSARY" helpful in learning concepts and completing exercises?

Yes	(19%)
No	(21%)
Not Sure	(58%)
no answer	(2%)

Would you prefer the exercises to be made more competitive by including rewards (gift vouchers) for the high scores?

Yes	(50%)
No	(36%)
Not Sure	(10%)
no answer	(4%)

Was the right-hand side expand collapse menu intuitive and easy to use

Yes	(77%)
No	(21%)
no answer	(2%)

Did the "HELP MENU" have sufficient details to guide the user in using the system?

Yes	(20%)
No	(9%)
Not Sure	(68%)
no answer	(3%)

Would you prefer a Virtual Buddy to guide you through the course features rather than the help menu?

Yes	(12%)
No	(79%)
Both	(6%)
no answer	(3%)

User comments about strengths and weaknesses of AlcoZone

	Strengths	Weaknesses
1	I liked the layout of the content and the flow from one topic to another.	I did not understand the calander part of the course. That was not clear what you were supposed to do. Also, if a student wanted to they could just race through this by just clicking next and not actually learn anything.
2	It was straightforward and easy to use.	More sophisticated page design and layout.
3	The media.	The audio was a little distracting but I believe it is necessary. Try to get each section recorded by a human and use that instead.
4	The adaptive content was appeared to work very well, making the second part of the alcohol consumption course more informative.	Proofread. There were MULTIPLE grammatical as well as spelling errors throughout. Make completion voluntary with rewards rather than mandatory.
5	I liked the illustrations the best. It helped me grasp the information better. For example, I really like the impairment with writing illustration, and the "decide which drink is stronger" exercise.	Make it multiple browsers compliant. Binding any sort of educational program to one vendor cuts down on the number of people able to use the system and learn from it
6	I liked the fact that it was adapted to personal use. However, it seems to assume that occasional drinking means binge drinking. It could also mean just having a glass of wine at dinner or champagne on New Years.	There should be movies that are added on to make it more interactive and interesting.
7	The mini games.	More interactive activities could make it more user-friendly and keep the user focused, rather than just reading all the time
8	The questions in between the chapters and information.	In some of the hands on activities, i had trouble moving the drinks down to the bubbles.
9	I liked the fact that we could move through the pages as we wished, and move backwards if we want to look at something again.	It was all good. However, the audio slurred sometimes, but that was the only thing I noticed.
10	it was easy to go through and follow	Not really besides some videos

Appendix C

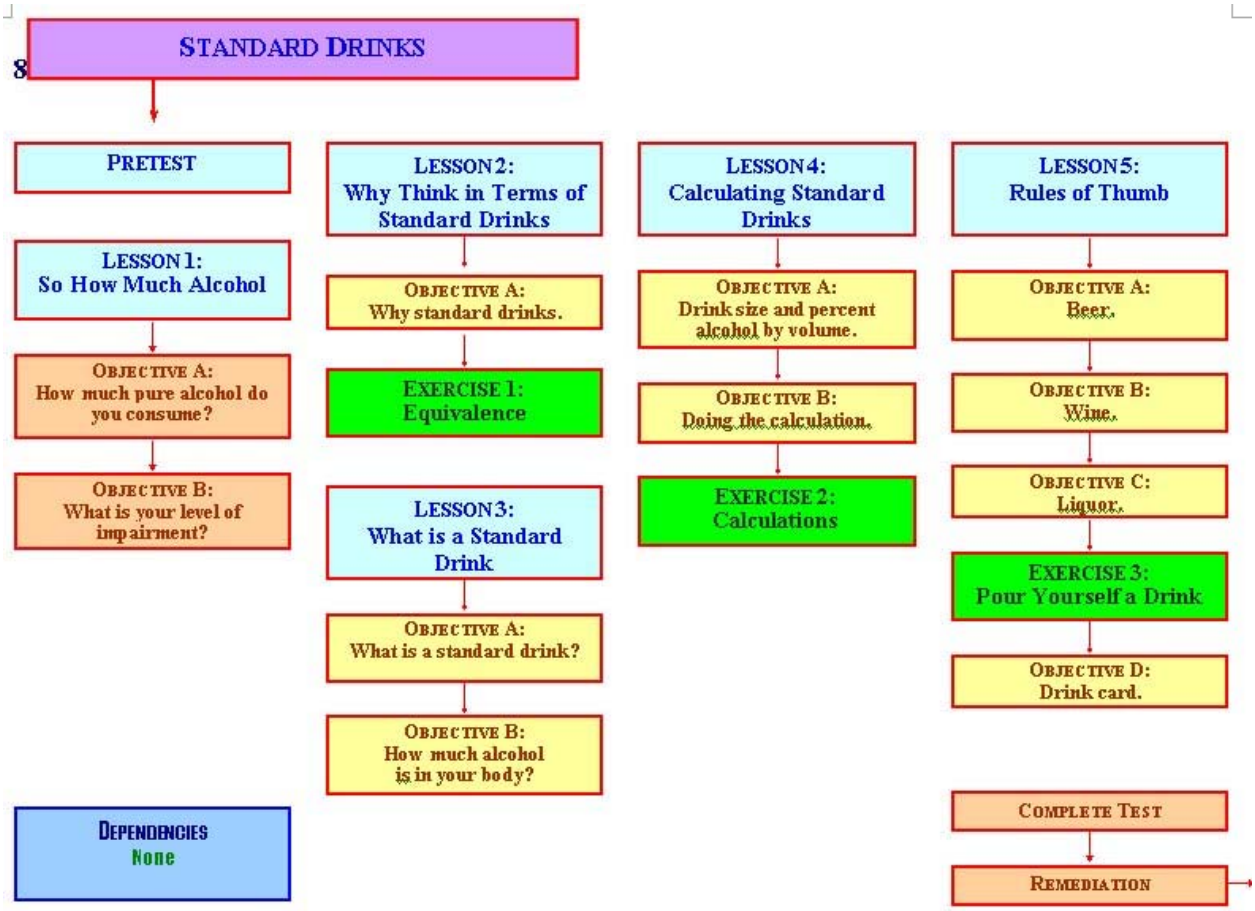
XML representation of a concept

```
<concept>
 <name>survey4.everconsumed</name>
 <description>Survey to record the alcohol consumption frequency</description>
 <resource>file:/survey4/xml/everconsumed.xhtml</resource>
 <stable></stable>
 <concepttype>page</concepttype>
 <title>everconsumed</title>
 <hierarchy>
  <firstchild></firstchild>
  <nextsib>survey4.consumedlastyear</nextsib>
  <parent>survey4.standarddrinks</parent>
 </hierarchy>
 <attribute name="access" type="bool" isPersistent="false" isSystem="true" isChangeable="false">
  <description>triggered by page access</description>
  <default>false</default>
   <generateListItem isPropagating="true" >
    <requirement>! survey4.everconsumed.suitability &amp;amp;&amp;amp; survey4.everconsumed.knowledge
        &amp;lt; 35</requirement>
     <trueActions>
     <action>
      <conceptName>survey4.everconsumed</conceptName>
      <attributeName>knowledge</attributeName>
      <expression>35</expression>
     </action>
    </trueActions>
   </generateListItem>
   <generateListItem isPropagating="true" >
    <requirement>survey4.everconsumed.suitability</requirement>
    <trueActions>
     <action>
      <conceptName>survey4.everconsumed</conceptName>
      <attributeName>knowledge</attributeName>
      <expression>100</expression>
     </action>
    </trueActions>
   </generateListItem>
   <generateListItem isPropagating="true" >
    <requirement>survey4.everconsumed.suitability</requirement>
     <trueActions>
      <action>
       <conceptName>survey4.everconsumed</conceptName>
       <attributeName>visited</attributeName>
       <expression>100</expression>
      </action>
     </trueActions>
   </generateListItem>
 </attribute>
 <attribute name="knowledge" type="int" isPersistent="true" isSystem="false" isChangeable="true">
  <description>knowledge about this concept</description>
  <default>0</default>
```

```
   <generateListItem isPropagating="true" >
    <requirement>true</requirement>
    <trueActions>
     <action>
      <conceptName>survey4.standarddrinks</conceptName>
       <attributeName>knowledge</attributeName>
      <expression>survey4.standarddrinks.knowledge + (0.2 *
                    _survey4.everconsumed.knowledge)</expression>
     </action>
    </trueActions>
  </generateListItem>
 </attribute>
 <attribute name="visited" type="int" isPersistent="true" isSystem="true" isChangeable="false">
  <description>has this page been visited?</description>
  <default>0</default>
 </attribute>
 <attribute name="suitability" type="bool" isPersistent="false" isSystem="false" isChangeable="false">
  <description>the suitability of this page</description>
  <default>true</default>
 </attribute>
 <attribute name="next" type="string" isPersistent="false" isSystem="false" isChangeable="true">
  <description>next concept to be presented</description>
  <default>&amp;quot;&amp;quot;</default>
   <generateListItem isPropagating="false" >
        <requirement>personal.answer.everconsumed=no</requirement>
        <trueActions>
         <action>
          <conceptName>survey4.everconsumed</conceptName>
          <attributeName>next</attributeName>
          <expression>"survey4.sdugoals"</expression>
         </action>
        </trueActions>
        <falseActions>
            <action>
             <conceptName>survey4.everconsumed</conceptName>
             <attributeName>next</attributeName>
             <expression>"survey4.consumedlastyear"</expression>
            </action>
        </falseActions>
   </generateListItem>
 </attribute>
</concept>
```

Appendix D

Concepts in Chapter 2



Concepts in Chapter 3