# An enhanced architecture and design for the personalized Content Presentation and Navigation

Ronnie Cheung, Hassan B. Kazemian
London Metropolitan University
United Kingdom
ccheung@acm.org, h.kazemian@londonmet.ac.uk

***ABSTRACT:*** *This paper describes the features of an adaptive e-learning system from the perspectives of content adaptation and generation architecture, which includes three models: the domain model, the learning models and the adaptation model. We have used in our study, the rule base of the personalized e-learning system. We have documented that most students consider the personalized e-learning system useful for learning basic Java programming.*

**Keywords:** E-learning, Ontology, Personalization, Adaptive System

**Received:** 23 December 2011, Revised 19 January 2011, Accepted 28 January 2011



## 1. Introduction

Traditional Web-based leaning systems are based on static contents that are accessed by different learners. Such kind of approach may not be effective for learners with different backgrounds and abilities. The ability to learn a topic is strongly related to an individual's background, learning aptitude and learning aptitude. For example, the courseware for a programming course may include a large number of programming examples and demonstrations. However, lengthy treatments of course contents may not be suitable for advanced learners who want to master the advanced features for the subject matter in a short time. Therefore, it is necessary to design an adaptive framework to implemented e-Learning systems that can cope with the differences in abilities of the learners in terms of background, preferences, learner aptitude, attitude and learner performance with previous activities in the system [2]. In this paper, we describe the development of a personalized e-learning system using an adaptive hypermedia approach. The Personalized E-learning System (PES) is developed as an educational hypermedia system using Semantic Web technologies [13]. It is based on an adaptive framework that is capable of retrieving distributed learning repositories. A personalized adaptive Java course is adopted as the courseware so that students at different educational levels can learn basic Java programming. The initial course level is determined by the student's programming background. The course level of the subsequent chapters would be promoted to a more advanced level, or demote to a less advanced level depending on the chapter test results.

With the rapid development of distance learning and the Web technologies, Webbased learning has now become an important branch in the education technology. One of the benefits for e-learning is that the learning environment can be adapted to the individual's learning process and learning need. In fact, the adaptability and the personalization feature of the e-learning environment is one of the research areas that draw much attention in the e-learning field. The next generation of Semantic Web technologies provides a common framework that allows data to be shared and reused across applications, and it is considered to be a promising technology for implementing e-learning systems.

This paper describes the features of an adaptive e-learning system from the perspectives of content adaptation and generation based on different user models.Personalization and adaption are achieved by implementing an adaptive hypermedia architecture, which includes three models: the domain model, the learning models and the adaptation model. The implementation details of the adaption and personalization features are discussed with reference to the information stored in the domain model, the learner model and the adaptation model. The detailed system design aspects are described with details on how adaptive contents can be implemented and generated using a rules-based reasoning mechanism. Finally, a sample course model "guided study" is used to demonstrate the dynamic course sequencing and presentation processes.

## 2 Literature Review

### 2.1 Semantic Web Technology

The concept of the Semantic Web has emerged with the aim of making Web resources more understandable by machines. It provides semantic-based access to the Internet [4], and provides a common framework that allows data to be shared and reused across applications, enterprise and community boundaries [9]. The main task of the Semantic Web is to "express meanings". In order to achieve this, several layers are required, including the XML layer which represents data, the Resource Description Framework (RDF) layer, which represents the meaning of the data, and the Ontology layer, which represents the formal common agreement about the meaning of data. It also enables intelligent reasoning with meaningful data. The effectiveness of the Semantic Web will increase drastically as more machine-readable Web content and automated services (including other agents) become available. This level of inter-agent communication will require the exchange of "proofs". Two important technologies for developing the Semantic Web are already in place: eXtensible Markup Language (XML) and the Resource Description Framework (RDF).

Ontologies are specifications of the conceptualization and corresponding vocabulary used to describe a domain [5]. They are well suited for describing heterogeneous, distributed and semi-structured information sources that can be found on the Web. By defining shared and common domain theories, ontologies help both people and machines to communicate concisely, supporting the exchange of semantics. It is therefore important that any semantic information for the Web is based on an explicitly specified ontology. By using this approach, consumer and producer agents of the Semantic Web can reach a shared understanding by exchanging ontologies that provide the vocabularies required for processing. Ontologies typically consist of definitions of concepts relevant for the domain, their relations and axioms about these concepts and relationships. Several representation languages and systems are defined, but the more recent language, OWL (Web Ontology Language) is developed to unified different ontology languages. It is also used as a language for describing web resources and supporting inference over those resources.

### 2.2 Personalization and Adaptive Educational Hypermedia Systems

Personalization is an important feature that tailors and customizes learning experience to individual learners, based on the analysis of the learner's objectives, current status of skills and knowledge, and learning style preferences [3][8]. In a personalized elearning system, the learner's interaction, information requests, problem-solving attempts will be recorded such that the system is able to recommend information to the learner or translate the learner's request into a query and send the query to the destination through Web services [11][12].

In the hypermedia or hypertext paradigm, information is interconnected by links; different information items can be accessed by navigating through the link structure. Adaptive Hypermedia (AH) systems merge hypermedia with the user modeling technology, and can be applied in a variety of application area, of which one dominating area is education. In the e-learning area, personalization can be implemented by using Adaptive Hypermedia. Adaptive Educational Hypermedia (AEH) [1] deals with the issue of providing a personalized educational experience. An AEH system aims at providing adaptive presentation of multimedia or text contents according to the learner's needs.

Adaptive Educational Hypermedia Systems overcome the problem of presenting the same content to different users in the same way, regardless of their different interests, needs and backgrounds. AH systems provide two general categories of adaptation. Adaptive content presentation is presenting the content in different ways, according to the domain model (concepts, their relationships, prerequisite information, etc) and information from the user model. The Adaptive navigation modifies the availability and/or appearance of every link that appears on a Web page, in order to show the user whether the link leads to interesting information.

## 3 System Architecture

### 3.1 Design Overview

The personalized e-learning System (PES) is designed to use the Semantic Web technology to provide personalized e-learning experience for the learners based on a hypermedia adaptive framework. The adaptive framework is based on the Adaptive Educational Hypermedia System (AEHS) [10].

Figure 1 shows the use case diagram of the PES system. In this system, the learner is able to view and edit his profile, which contains the learner's programming background and experiences, and other preferences. These data will be used to drive the adaptation parameters. When the learner logs onto the system, a preset course will be loaded. The learner can view the table of content of the course, and access any unit within the course. In the basic Java programming course, a guided study feature is provided for the learners to learn the course sequentially. The sequence of the course and the course content display will be adapted according to the learner's learning aptitude and test results. To make this system more user-friendly, the learner can view his learning progress and his assessment report.

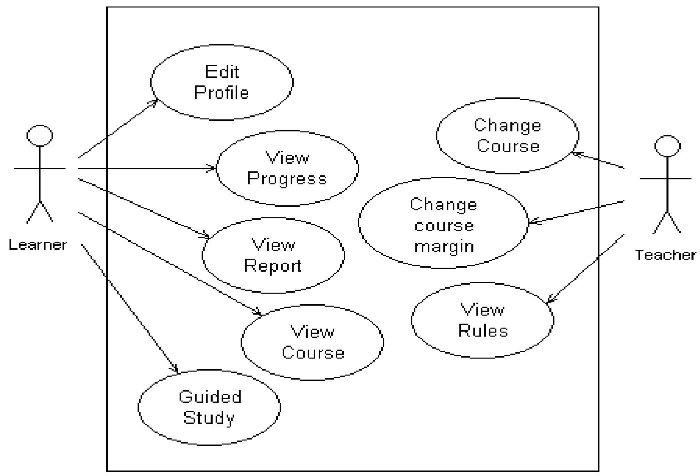


Figure 1. Use case diagram for personalized e-learning system

The teacher can log onto the system and change the course. In the personalized elearning ontology-based system, the course change is apparently equivalent to changing the ontology file of the course. In the course sequencing module, the test results scores are used to determine which course content and course level should be displayed. In our proposed system, we use the Java Expert System Shell (JESS) [6] to implement the adaptive features. The knowledge base, which is a set of Jess rules are stored in a text file to be loaded into the system during system initialization. These rules can be also be edited by the teacher.

### 3.2 System Architecture

The system architecture of the personalized e-learning system is shown in Figure 2. It provides a framework to express the functionality of adaptive hypermedia systems by dividing the storage layer into three parts that specify what should be adapted, according to what features should it be adapted, and how should it be adapted.

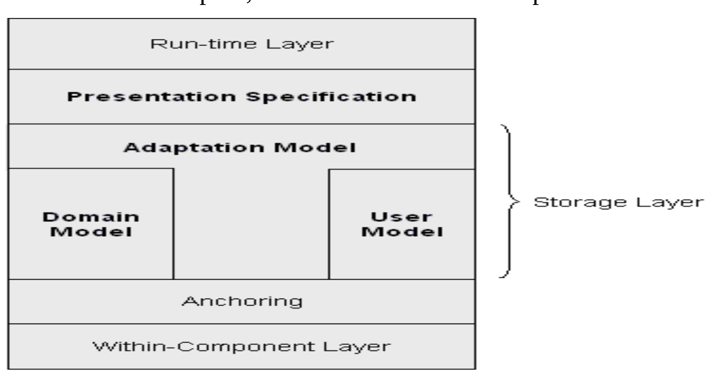


Figure 2. System Architecture of PES

Adaptation to the user's individual characteristics is implemented according to the user model. A user model determines the user's goal, tasks, beliefs and characteristics that are important for adaptation. According to Brusilovsky [1], a hypermedia application can be adapted to the user's knowledge, goals, background and experience, preferences, interests, individual traits and environment. In addition, the following groups of individual user characteristics could be important for the adaptation of elearning systems: personal data (including demographic characteristics such as age, culture), psychological, cognitive and physiological parameters such as user's attention (simple and complex reaction time), memory (verbal working memory, long-term memory), cognitive abilities (spatial arrangement, etc), user's internality/externality, cognitive and learning styles, and personal decision abilities.

The storage layer consists of three models, mainly the domain model, learner model and the adaptation model. The domain model describes how the information content is structured. The user model describes the information about the user. The adaptation model contains the adaptation rules that define how the adaptation is performed. Each model is interrelated and can be accessed by the user.

### 3.2.1 The Domain Model and Ontology

The domain model specifies the conceptual design of the adaptive hypermedia application, i.e. what will be adapted. The design of the domain model is to design the concept hierarchy of the learning objects. In our system, the learning object is the Java course provided through the distributed learning resources. There are three levels of the Java course, with different emphasis on the detail description of the subject depending on the learning preferences and performance of the learner. Each level has its own set of questions. The teacher can choose to display only the questions of the corresponding level, or display questions of all levels to determine the learner's aptitude of the learned course. The course level that is presented to the student is initially determined by the learner's background. Course level of subsequent chapter can be changed according to the learner's test results. If the learner has chosen to take pretest, the pretest result will be taken as the reference, otherwise, the chapter test result of the previous chapter will be used as a reference, assuming that if the user has fully understand the topic, he/she will get a high score in the chapter test. This result is used as a prediction for his future learning. If the learner desires to learn from more examples as his learning preference, more examples on the subject will be displayed.

Figure 3 shows the ontology model for the course relationships for the learning objects. It comprises of Concept and Document classes, with subConceptOf and isPrerequisiteFor relationship between concepts. Each Document class has a subject Concept class to describe the relationship between a concept and the related document. Within the Document class, the isPartOf relationship is used to describe the hierarchical relationship between documents.

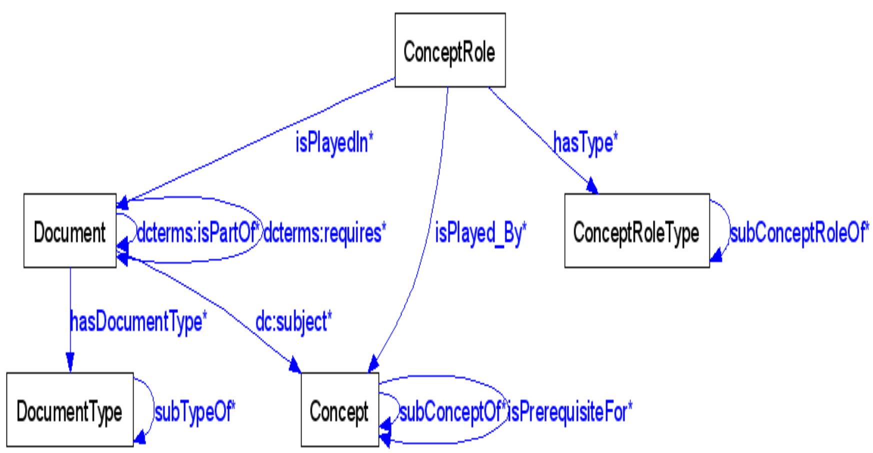


Figure 3. Ontology model for the learning object

Figure 4 shows an excerpt of the concept relationship between the pages in the Java course. If conceptA is a prerequisite for conceptB, the system will display conceptA to the learner who would like to learn conceptB but does not meet the prerequisite requirement.

### 3.2.2 The Learner Model

The learner model is designed based on the general learning scenario of the students. In particular, the learner model described the learner's static information related to the following items:

- The name and student ID of the learner

- The education status, whether he is a high school student, bachelor degree or postgraduate level
- The learning goal, which course he wants to take,
- The motivation of the learner, whether he has high, middle or low level of motivation to learn the subject
- The learner's experience in the chosen subject, whether he has no experience, basic, intermediate or advanced level
- The learning style, whether he prefers principle-oriented or example-oriented.

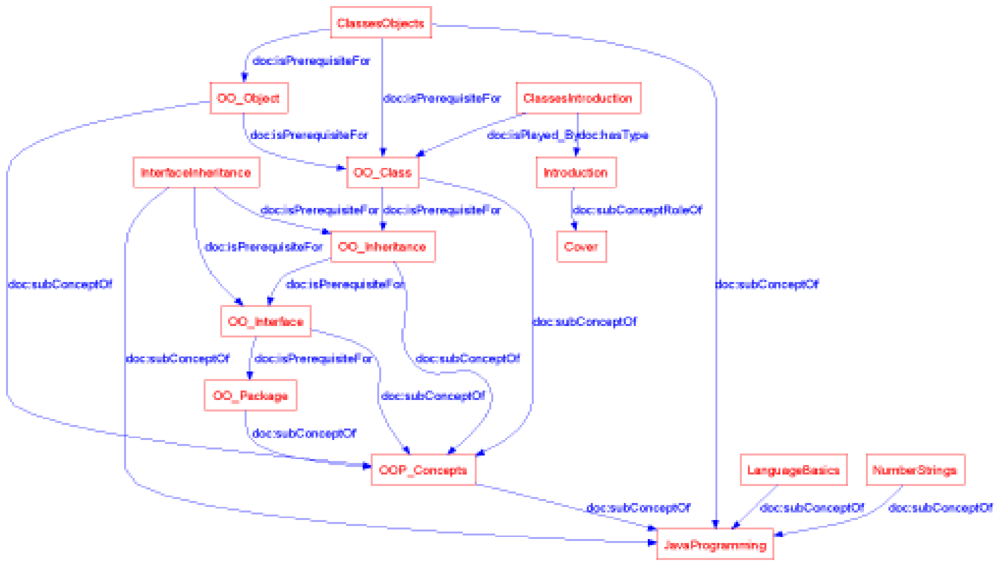


Figure 4. Concept relation of Java Course (an excerpt)

The learner model also describes the learner's learning process and behavior related to the following items:

- What knowledge has the learner mastered? Has he mastered a certain topic in the chosen subject?
- What knowledge has the learner not mastered? What knowledge has he missed?
- At which part of the knowledge is the learner weak/strong?
- What examples were given to the learner?
- What answers did the learner give to a certain question?
- What score did the learner get in the test?
- How many times did the learner try before answering the question correctly?
- What is the rate of success and failure?
- How much time did the learner spend on the questions?
- How well does the learner master the subject?
- How much time did the learner use to master the subject?

The ontology-based learner model describes static and dynamic information that is related to the particular learner. The user preference module describes the static information of the learner that is set by the learner upon registration and the user knowledge module describes the dynamic information that is generated during the learning process. The static information will be created when the learner registered with the system. Registered user can log on to the system to edit the learner profile. The user knowledge is built as a hybrid of an overlay model and a historic model. The key principle of the overlay model is that for each domain model concept, an individual user knowledge model stores some data that can estimate the knowledge level on this concept. The historic model keeps some information about the learner's visit to individual page. In particular, the dynamic information in the learner model includes user visited pages, logged on duration, attempted test, test result, course he has completed, etc.

Figure 5 shows the learner model for the user knowledge module. The Personal class describes the learner's personal information. The Preference class describes the preferences set by the learner. The Goal class describes the learning goal of the learner. The dynamic information of the learner model is described in the Interaction and Performance classes. The Interaction class described the course pages the learner has visited. The learning process, including test results, of the learner will be stored in the Performance class.

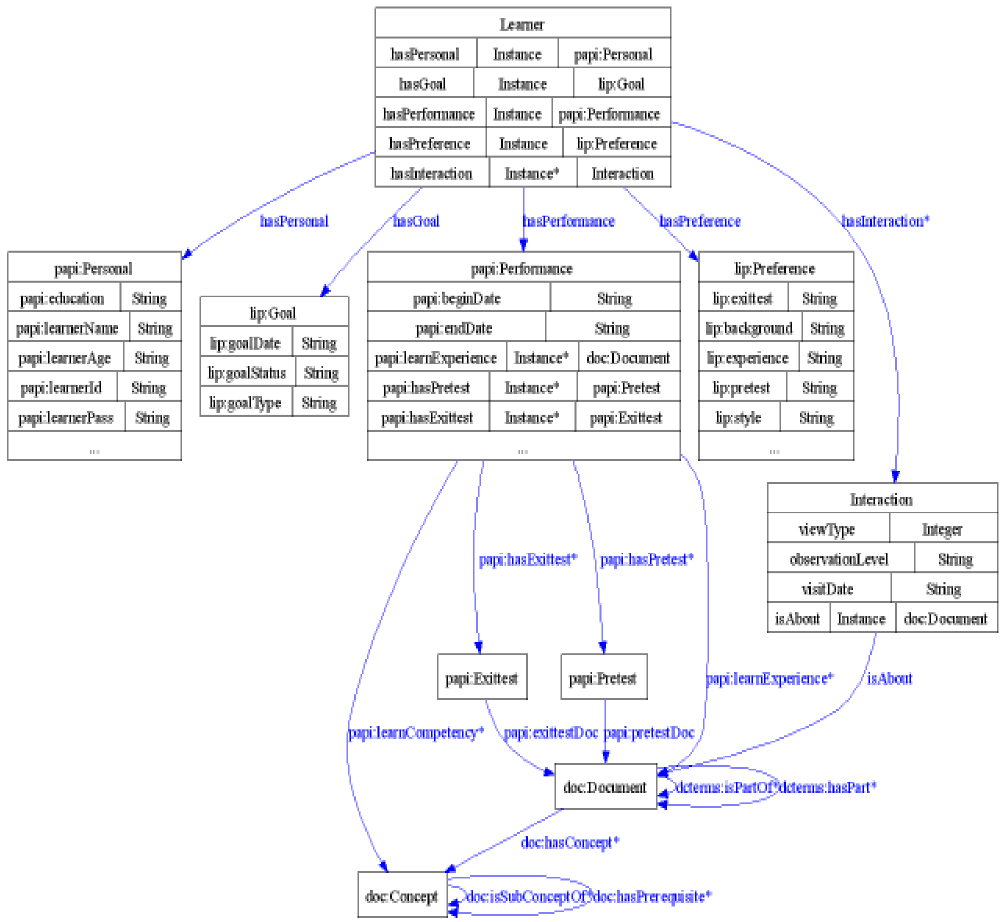


Figure 5. Ontology for the learner

### 3.2.3 The Adaptation Model and the Rule Base

The adaptation model specifies the specific adaptation semantics of the system. The adaptation model consists of adaptive presentation, adaptive navigation support and curriculum sequencing. It is implemented as rules that are stored in a rule base in the personalized e-learning system.

#### 3.2.3.1 Adaptive Presentation

The adaptive presentation module includes the optional pretest generation and course presentation. The pretest can be generated according to the learner's preference, background and the chapter end test (exit test) result of the previous unit. The Jess rule "generatePretest1" and "generatePretest2" define the condition that generates the pretest question list. If the learner's programming background is "novice" and he has chosen to take pretest, the pretest question list will be the same as the last exit test question list. If the learner's programming background is "expert", the pretest questions will consist of both the last exit test questions and the coming chapter end test questions.

```
(defrule generatePretest1
  (learner (background "novice")(ifPretest "has pretest"))
  (exittestList (state "current")(q_id $?el))
 =>
```

```
(assert (pretestList (q_id $?el))))

(defrule generatePretest2
  (learner (background “expert”)(ifPretest “has pretest”))
  (exittestList (state “current” )(q_id $?el1))
  (exittestList (state “next” )(q_id $?el2))
 =>
  (assert (pretestList (q_id $?el1 $?el2))))
```

The adaptive presentation module is also responsible for presenting relevant information to the learner to ensure that he can understand and learn the subject according to his preference and ability. In our system, one of the three different levels of the course will be displayed to the learner according to his learning progress. The default course view level will be set by the learner’s background and programming experience. However, this level can be changed according to the learning progress. This level change can be determined by the chapter end test (exit test) result and if pretest is available, by the pretest result. The Jess rule “recommendNextByPretest1” defines the course view level to be displayed according to the learner’s pretest result. If the pretest results “grade” of a particular document “doc”, which represents a unit or a subunit of a course, is higher than the course margin “pretest_promote” that has been set by the teacher, the system will promote the level of the document to be displayed. This information is stored in a Jess fact “recommend_next”.

```
defrule recommendNextByPretest1
  (levelchange (pretest_promote ?p))
  (pretest (doc_id ?id)(grade ?g))
  (test (>= ?g ?p))
 =>
  (assert (recommend_next (doc_id ?id)(view “promote”))))
```

If the learner has chosen “no pretest” as his learning preference, the course level change will be determined by the result of the exit test. The system will present the next document only if the learner has passed the exit test. The Jess rule “recommendNextByExitTest3” describes the case that if the learner has “visited” the chapter test, the chapter end test score “grade” is lower than the pass grade “exittest_pass” set by the teacher, the system will “demote” the course level, and display the same chapter again. This information is stored in a Jess fact “recommend_next” where the document “id” is the same as that of the “visited” chapter.

```
(defrule recommendNextByExitTest3
  (doc (id ?id)(view ?v)(adapt “visited”))
  (exittest (doc_id ?id)(grade ?g))
  (levelchange (exittest_pass ?ep))
  (test (< ?g ?ep))
=>
  (assert (recommend_next (doc_id ?id)(view “demote”))))
```

**3.2.3.2 Adaptive Navigation**

The adaptive navigation module support guides the learner towards relevant, interesting information related to the learning topic. In our system, we provide additional examples and references hyperlinks to the learner on the particular topic. The learner can increase their knowledge on the particular topic through these additional links. The system will determine the additional links according to the viewed topic and the learner’s preference. The Jess rule “addDocumentHyperlinks” describes the condition when a reference link “doc_ref” is added into the current document “id_parent”. If the reference document “id” has the same concept “concept” of that displayed document, and the learner’s preference has “link” equals “more references”, the system will assert a Jess fact “doc_ref” with the reference document “id” with the document to be added into “id_parent”.

```
(defrule addDocumentHyperlinks
  (learner (link “more references”))
  (reference (id ?id)(concept ?c))
  (doc (id ?id1)(concept ?c))
 =>
  (assert (doc_ref (id ?id)(id_parent ?id1))))
```

The adaptive navigation module also includes a set of recommendation rules to suggest the learners which topic is suitable for him. These recommendations are also used for course sequencing. The system will use the prerequisite of a particular concept to determine if the related document should be recommended to the learner. The Jess rule “recommendByVisit1” describes the next concept to be learned according to the concept prerequisite information. If a document “doc” with concept “concept” has been visited “visited”, and there exists ‘concept_prereq” where the concept is a prerequisite for another concept, the document with this other concept will be recommended and stored as a Jess fact “recommend”. If there is only one “recommend” at one time, the system will be able to perform course sequencing according to this Jess fact. However, if there is more than one “recommend” fact at one time, the system will display the document with a link annotation “recommend” in the “System Suggestion” column on the main course page.

```
(defrule recommendByVisit1
  (doc (id ?id)(concept ?cp)(adapt “visited”))
  (concept_prereq (id ?c)(preId ?cp))
  (doc (id ?i)(concept ?c))
=>
  (assert (recommend (doc_id ?i))))
```

When the learner has finished viewing the selected topic, the system will generate the exit test for the learner to make sure that he has understood the topic. The results will be recorded and updated in the learner’s profile. The assessment item type of our system contains multiple choice questions only. The learner can set his exit test condition to either “no passing grade” or “pass at 50%”. If the learner has failed the exit test, the system will suggest the learner to revisit the topic, assuming that he has not fully understand the topic. The Jess rule “recommendByExitTest1” describes the condition where the document “doc” has been visited, but the exit test “exittest” score “grade” of the corresponding document is below the passing grade “exittest_pass” that is set by the teacher, the system will change the document from “visited” to “to revisit”.

```
(defrule recommendByExitTest1
  ?d1 <- (doc (id ?id)(concept ?c)(adapt “visited”))
  (exittest (doc_id ?id)(grade ?g))
  (levelchange (exittest_pass ?p))
  (test (<= ?g ?p))
=>
  (modify ?d1 (adapt “to revisit”)))
```

**4. Adaptation Rule Parser**

In the PES personalized e-learning system, we use the Java Expert System Shell (Jess) [6] as the adaptation rule parser. The Jess architecture is shown in figure 6. It is designed as a library that can be embedded into many applications. It has an extensive Java API to interact with Jess. The core of the Jess library is the jess.Rete class. An instance of jess.Rete is an instance of Jess. Every jess.Rete object has its own independent working memory, list of rules and set of functions. The jess.Rete class exports methods for adding, finding, and removing facts, rules, functions, and other constructs. The Rete class is a façade for the Jess library.

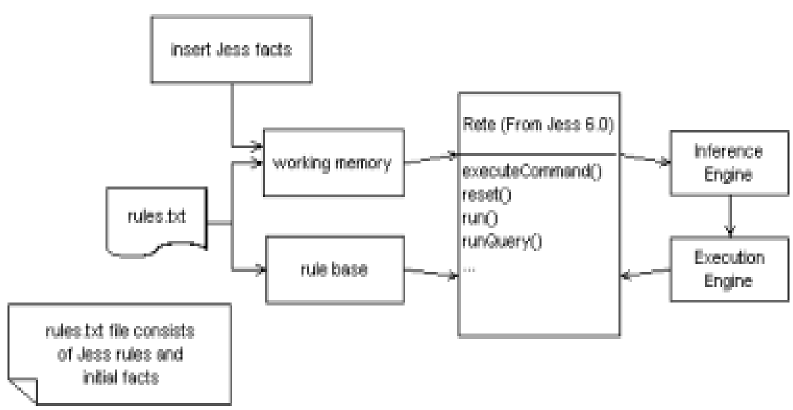


Figure 6. The Java Expert System Shell

The following code extract creates the jess.Rete object when the personalized elearning system starts. When the system starts, jess.Rete object will be created. By calling the executeCommand() to run the command “batch rules.txt”, the rules file “rules.txt” will be read into the system. The file “rules.txt” contains the initial Jess facts and Jess rules of the system. Calling the reset() command will reinitialize the working memory of the Jess’s rule engine. After reinitializing the working memory, all activated rules in the rule base will be fired by calling the function run(). Jess’s Java API reports errors by throwing instances of jess.JessException. Therefore, it is necessary to catch this exception when working with Jess in Java.

```
import jess.*;
try {
    rulesFile = “rules.txt”;
    Rete engine = new Rete();
    engine.executeCommand(“(batch \”” + rulesFile + “\”)”);
    engine.reset();
    engine.run();
} catch (JessException e) {
}
```

In the PES system, we use the forward-chaining rules in Jess. The Jess rules will be fired when the ‘if’ statement is matched. The new facts will be stored in the Rete object. Calling the runQuery() method can retrieve the facts, thus displayed on the web pages. During the learning process, additional Jess facts will be asserted into the working memory and the Jess rules will be executed accordingly. These Jess facts include learner’s visited pages and test results. The inference rules and inference processes are implemented according to the PES system implemented by Cheung et al. [3].

## 5. Detailed System Design

The personalized e-learning system uses the Model-View Controller architecture to separate application logic and page presentation as shown in Figure 7.

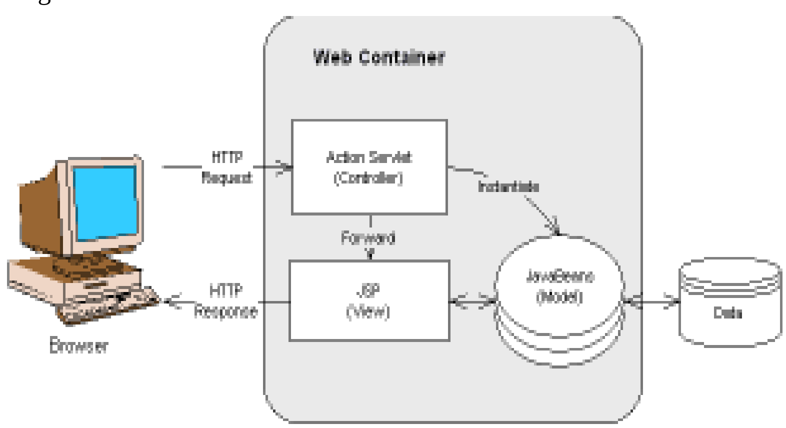


Figure 7. The MVC Model

With this model, the servlets control the flow of the web application and delegate the business logic to the JavaBeans while the JSP pages generate the HTML for the web browsers. In the PES system, the system data consists of the text file that stored the Jess rules and initial Jess facts and an OWL ontology file for each student describing the learner’s profile and his learning progress. Figure 8 shows the relationship between the objects the PES system. For each application session, the session object contains both the course bean and the student bean such that the Java Server Pages will handle the presentation logic and process the Java Beans to generate dynamic presentation code. The course bean contains the LearnerJessInfo object that stores the Rete object from Jess’s Java API [6]. The Rete object stores the initial Jess facts and Jess rules from the text file and Jess facts generated at runtime. The course bean also contains the LearnerOWLInfo object that stores the JenaOWLModel object from the Protégé-OWL API [7]. The JenaOWLModel object stores the learner and course content of the ontology.

## 6. Course Sequencing and Presentation

Figure 8 shows the flowchart for the main logic for course sequencing in the system. The “Guided Study” is an adaptive course sequencing feature for the learner to learn the course unit according to their past interactions. When the learner presses “Guided Study” in the course page, the system will find the latest course level and the current unit from the learner’s ontology instance. The initial course level is determined by the learner’s background and programming experience shown in table 1. The initial current unit will be the first unit of the course. If the learner has started the course, the latest course level and the current unit that has been stored in his last visit will be retrieved from the learner’s ontology instance. Each unit has a unit type. If the unit type is a pretest, the pretest will be displayed. The learner will be requested to answer the questions before he can proceed with the course content. This pretest is optional, and is set in the learner’s profile. The pretest result will be used to determine if the course level has to be promoted or demoted. When the learner has finished the pretest, the system will display the course unit content with the course level either determined by the learner’s profile, or the previous results. The system will also display the course content when the view type of the current unit is “lecture”. When the course unit content is displayed, the system will update the learner “visited” pages, assuming that the learner will read the content of the unit displayed. Upon completion of the course unit content, the learner can press “Next” to proceed to the next page. If the learner prefers “more examples” in his learner profile, an example page will be displayed to further explain the course unit content to the learner. Again, the system can display this page when the learner press “Guided Study” and the current unit view type happens to be an “example”. When the learner press “Next” again, the course unit end test will be displayed. The test questions can be either contains a single level of questions that is coherent with the course level, or a mixture of different levels. The test questions will be determined by the teacher. In the teacher’s page, the teacher can choose between “same”, which is the same level of question as that of the course level, or “mix”, which is a mixture of different levels of questions. In this way, the teacher can change the test questions to meet the individual needs of different learners.

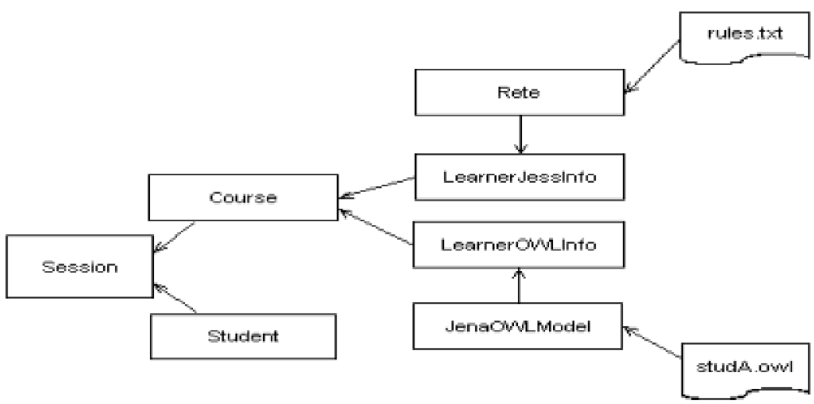


Figure 8. Relationship between the System Objects

When the learner has completed the chapter end test questions, the system will check the results, and will determine the course unit and course level according to the learner’s result. There are four possibilities. If the learner has failed the test (rp), the system will demote the course level; if the learner has passed the course, but the score is lower than the demote score (du), the system will increase the course unit and demote the course level; if the result is above the demote score but under the promote score (ru), the system will increase the course unit only. If the learner test score is beyond the promote score (pu), the system will increase the course unit and promote the course level. The system has assumed that if the learner has a high score in the test, the learner understands the subject well and there is a high probability that the learner is able to learn faster by displaying course content that has less illustrations and examples. This means that the learner will be able to view the next unit in a higher level.

The content presenter is responsible for presenting information to the learner or the teacher according to the user login. If the user is logged in as a learner, the learning environment will be displayed. If he is a teacher, the authoring environment will be displayed. Figure 10 shows an example of a course page in PES, with the details of the pre-tests and exit-test for the learner. Figure 11 shows a course page with learning recommendations.

## 7. System Evaluation

In order to evaluate the effectiveness of the e-learning system, there are three major aspects that are considered in the evaluation: (1) clear content and organization, (2) adaptability for different learners, and (3) user-friendliness and usability. A formal evaluation is conducted by inviting students to take the online Java course. After completing the course, students are

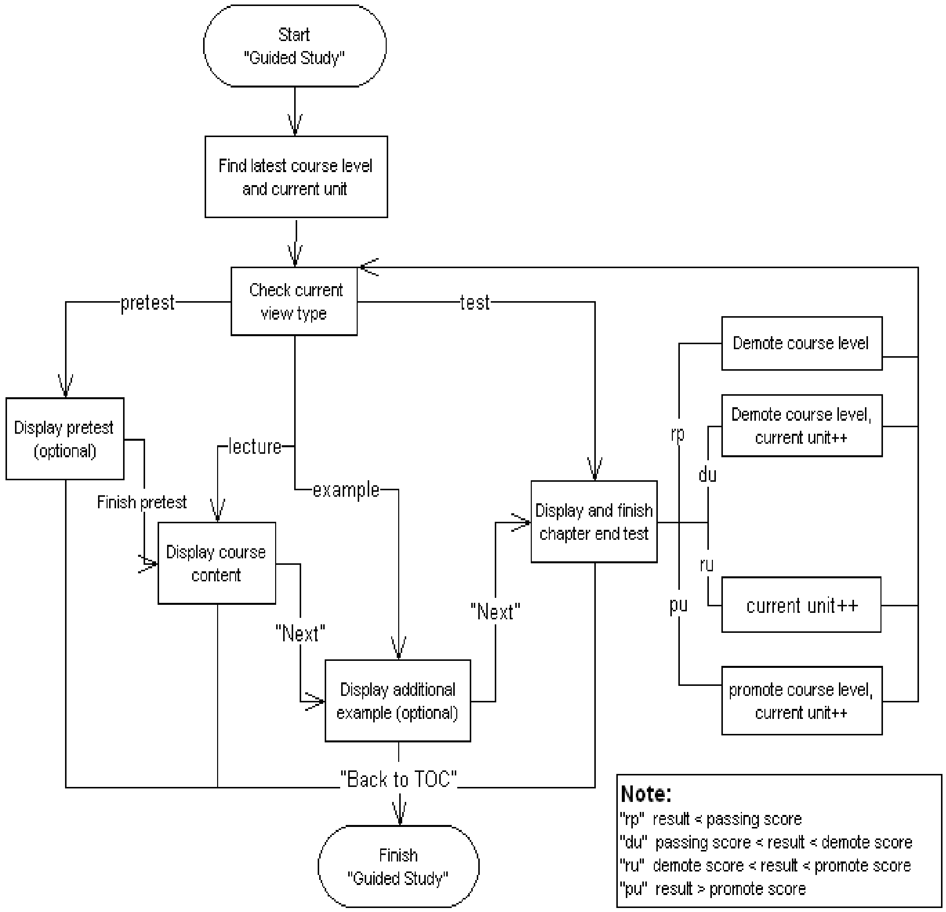


Figure 9. Flow Chart for “Guided Study”

| Programming background | Programming experience | Initial course level |
|---|---|---|
| Novice | Novice | Basic (1) |
| Novice | Intermediate | Basic (1) |
| Novice | Expert | Intermediate (2) |
| Intermediate | Novice | Intermediate (2) |
| Intermediate | Intermediate | Intermediate (2) |
| Intermediate | Expert | Intermediate (2) |
| Advanced | Novice | Intermediate (2) |
| Advanced | Intermediate | Advanced (3) |
| Advanced | Expert | Advanced (3) |

Table 1. The Initial Course Level

required to give the feedback on the system by evaluating the system features based on a 5 point scale ranging from 1 (lowest) to 5 (highest). A total of 64 questionnaires are received from students with education background ranging from high school to graduate level. The programming experiences among these students also varied. Many students have no programming experience, and there are also a significant number of student with extensive experiences in object-oriented programming.

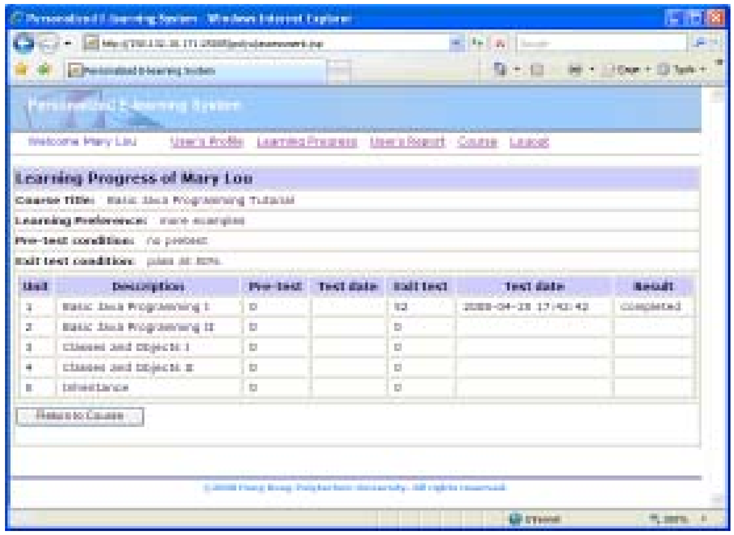


Figure 10. A User Content Page in PES

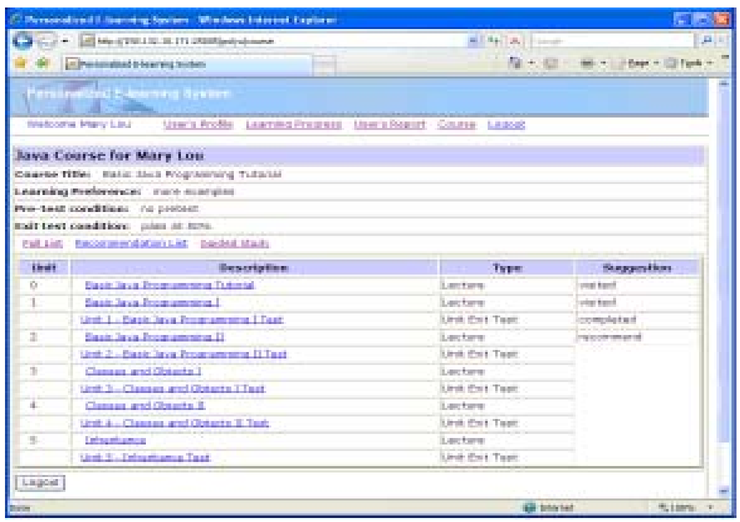


Figure 11. A Course Page with Learning Recommendation

In the system evaluation, respondents having different education levels are selected for the study. Out of the 64 respondents, 42 students are studying at the university at undergraduate or graduate level, 14 students are studying in a community college or associate degree, and 8 of respondents are high school students. Students' Java programming knowledge and online learning experience are also analyzed. Out of the high school students' group, only one respondent knows Java programming at basic level, and the other 87.5% of the students have no programming knowledge. This is significantly different from the college students' group where all of students have programming knowledge; with 28.5% of them having basic Java programming knowledge. As for the university students, all the students have basic programming knowledge.

From the evaluation, most of the high school respondents have no Java programming knowledge and minimal online learning experience. The data from this group of respondents has significantly low responses from the other respondents. In general, the overall rating of the system from this group is 1.75, which is very different from the 3.50 from the college students and 3.79 from the University students. For high school students, the rating for the system adaptation to the students' background is 2.25, compared with 3.57 for the community college students and 3.66 for University students. The t-test is used to test whether the grand mean score for university students is different from high school students. The calculated t-value is 4.48, with a degree of freedom of 48; the computed p-value is 0.000046. The results show that the mean score for university students is different from the score for high school students at 0.001 significance level.

The grand mean score for the college students' group and the university students group are further compared. The t-value is 0.692, with a degree of freedom of 54, the computed p-value is 0.49. The results indicate that the grand mean score for university and associate degree students are the same. Although university and associate degree students have different backgrounds, the results of the evaluation for the system are similar, indicating that PES is effective in adapting to different users. The PES system was designed with a target audience who has different learning experience, and the effectiveness of online learning is significantly improved by the adaptive features that suits the individual needs. However, if the student has no programming language knowledge, using this system may be a nightmare. Table 2 shows the comparison between different student groups. The acceptance of the overall system and the features are the lowest among high school students.

| **A. Overall rating of the system (5=excellent, 1=poor)** | **Uni.** | **Comm. Col.** | **HS** |
|---|---|---|---|
| How would you rate the system? | 3.78 | 3.50 | 1.75 |
| How would you rate the structure of the course? | 3.80 | 3.64 | 2.50 |
| How would you rate the adaptive feature of the course? | 3.76 | 3.50 | 2.12 |
| How well did this course meet your expectations? | 3.90 | 3.71 | 2.37 |
| **B. Features of the system (5=strongly agree, 1=strongly disagree)** | **Uni.** | **Comm. Col.** | **HS** |
| Course contents adapted to your programming background and knowledge | 3.66 | 3.57 | 2.25 |
| Content structure and presentation sequence is relevant | 3.57 | 3.64 | 2.37 |
| Course level change according to the chapter test result is relevant | 3.55 | 3.07 | 2.37 |
| Course level promotion helps you understand the topic faster. | 3.33 | 3.14 | 2.50 |
| Course level demotion helps you understand the topic better with more explanation. | 3.62 | 3.57 | 2.25 |
| Recommendations are helpful | 3.57 | 3.85 | 2.25 |
| Website navigation is clear and easily understood | 3.71 | 4.00 | 2.37 |
| **C. General e-learning experience (5=strongly agree, 1=strongly disagree)** | **Uni.** | **Comm. Col.** | **HS** |
| You have experience in online learning | 3.98 | 3.57 | 2.00 |
| You prefer web-based learning to face-to-face tutorial | 3.98 | 3.64 | 2.37 |
| You have successfully completed an online course in the past. | 3.86 | 4.07 | 2.12 |
| You feel comfortable learning online | 4.02 | 4.07 | 2.50 |

Table 2. Comparison between students with diffent education levels

## 8. Conclusions

Using an adaptive hypermedia framework, we describe the development of a personalized e-learning system (PES) for students to learn basic Java programming. In particular, we have developed a 3-level Java course to enable course content level promotion and demotion to help the student to learn the subject in different levels. The PES system also provides suggestions to the students by system recommendations. Through the system, students can learn the basic Java course with adaptive features according to their progress and performance. In the PES system, a user modeling approach is employed to provide personalized learning. The personalization features are based on both static and dynamic information that are incorporated in the user model. The personalized courseware is initially based on the static information of the learner. As the user interacts with the system, the progress and performance of leaner is updated. The dynamic information is also used to guide the adaptation process for personalized e-learning.

A rule-based approach is used for the implementation of the adaptation model. The system allows teachers to edit the adaptation rules and manage the adaptation behavior of the system according to individual needs. The ontology-based approach is also used to describe the domain model and the learner model. It provides a flexible means for implementing the course contents. It also provides flexibility in maintaining distributed learning resources that can be maintained by a single system. The system evaluation with students shows that the e-learning system is effective for providing personalized e-learning.

## 9. Acknowledgments

The authors would like to thank the Hong Kong Polytechnic University for supporting this research.

## References

[1] Brusilovsky, P.(2001). Adaptive Hypermedia, User Modeling and User Adapted Interaction, 11(1-2) 87-110.

[2] Chen J., Man H., Yen N. Y., Jin Q., Shik K. T.(2010). Dynamic Navigation for Personalized Learning Activities Based on Gradual Adaption Recommendation Model, Lecture notes in Computer Science, Volume 6483, Springer, p 31-40.

[3] Cheung, R., Wan, C., Cheng C., An Ontology-based Framework for Personalized Adaptive Learning, *In*: Proceedings of the 9th International Conference on Web-based Learning. Shanghai, China, December, 2010, p 52-61 (2010).

[4] Dublin Core (2004). Dublin Core website, http://dublincore.org/

[5] Gómez-Pérez, A., Corcho, O.(2002). Ontology Languages for the Semantic Web, IEEE Intelligent Systems, 17(1),54-60

[6] Jess 6.1, Jess, the Rule Engine for the Java Platform, http://herzberg.ca.sandia.gov/

[7] Protégé 3.3.2, The Protégé Ontology Editor and Knowledge Acquisition System, http://protege.stanford.edu/

[8] Sampson, D., Karagiannidis, C (2002). Personalised learning: educational, technological and standarisation perspective, Interactive Educational Multimedia, No. 4, 24-39.

[9] The World Wide Web Consortium, http://www.w3.org/

[10] Weber, G., Brusilovsky, P (2001). ELM-ART: An Adaptive Versatile System for Webbased Instruction, *International Journal of A. I. in Education,* 21, p. 351-384.

[11]De Bra, P., Aerts, A., Berden, B., de Lange, B., Rousseau, B., Santic, T., Smits,D., Stash, N.(2003). AHA! The Adaptive Hypermedia Architecture, *In*: Proceedings of the ACM Hypertext Conference, p. 81-85, Nottingham, UK, August.

[12] Dolog, P., Henze, P., Nejdl, W., Sintek, M.(2004). The Personal Reader: Personalizing and Enriching Learning Resources using Semantic Web Technologies, International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems.

[13] Cheung, R.., Kazemian, H. B.(2011). An Adaptive Framework for Personalized e-Learning, Communications in Computer and Information Science, 136, Springer, p. 293-306.

## Author Biographies

Ronnie Cheung held various posts including lecturer, senior lecturer, assistant professor and CyberU coordinator at the Hong Kong Polytechnic University. Currently, he is also working as a consultant for the Hong Kong Cyber University. He also has extensive consultancy experience for various organizations such as the Hong Kong SAR government and Hewlett-Packard Limited. Previously, he also worked as a consultant for Citibank in Tokyo. He also served as the Chairman of ACM-HK chapter and is a senior member of the association for computing machinery. His publications appeared in reputable international journals from various publishers such as Emerald, Wiley and IEEE.

Hassan B. Kazemian received a B.Sc. in Engineering from Oxford Brookes University, UK, a M.Sc. in Control Systems Engineering from the University of East London, UK, and a Ph.D. in Learning Fuzzy Controllers from Queen Mary University of London, UK, in 1985, 1987 and 1998, respectively. He is currently a full professor at London Metropolitan University, UK. He worked for Ravensbourne College University Sector, UK, as a senior lecturer for eight years. Previous lecturing experience of London, UK, in 1985, 1987 and 1998, respectively. He is currently a full professor at London Metropolitan University, UK. He worked for Ravensbourne College University Sector, UK, as a senior lecturer for eight years. Previous lecturing experience includes the University of East London, UK, University of Northampton, UK, and Newham College, UK. Research interests include fuzzy and neuro-fuzzy control of networks, 2.4 GHz frequency bands, ATM, video streaming and rate control. Prof. Kazemian is a senior member of the Institute of Electrical and Electronics Engineers (SMIEEE), Fellow of the Institution of Engineering and Technology FIET (formerly IEE) UK and Chartered Engineer (C. Eng.) UK.