Using Pattern Languages to Design Intelligent Tutoring Systems: A Case Study

Amir Zeid†, Dina Salah††

† Sciences and Engineering Division, American University of Kuwait, Kuwait
††The University of York, UK

Summary
Intelligent tutoring systems (ITS) in the domain of language learning are becoming very common. Software engineering best practices and frameworks should be used to ease the process of developing new intelligent tutoring systems. In this research, we show how a Pattern language for intelligent tutoring systems (PLITS) was used to develop modules of the Arabic Tutor. The Arabic tutor is a web-based ITS for teaching Arabic Grammar. We focus on how PLITS was used to develop both Tutor and interface modules of the Arabic Tutor.

Key words:

Modeling Intelligent Language Tutoring System, Teaching Arabic, Design Patterns, Expert Systems

1. Introduction

Arabic is a Semitic language and its grammar has many similarities with the grammar of other Semitic languages. Arabic is one of the official six languages of the United Nations. According to [16] around 246,000,000 speak Arabic. The demand to learn Arabic is growing. Recently the west started to pay more attention to Arabic language to communicate in more efficient ways with Arabic native speakers.

The design and implementation of Intelligent Tutoring Systems (ITS) is a very complex task, as it involves a variety of organizational, administrative, instructional and technological components. Intelligent Tutoring Systems incorporate built in expert systems in order to monitor the performance of a learner and to personalize instruction on the basis of adaptation to a learner's learning style, current knowledge level, and appropriate teaching strategies [1].

The Arabic Tutor is a web based ITS based on the client server architecture which consists of a Web server, an application server and student interface. The application server performs all tutoring functions. The interface consists of a set of interactive HTML pages; the Web server is responsible for passing the student's actions to the application server. The Arabic Tutor was developed using PLITS [17] as a guiding framework.

Manuscript received September 5, 2010
Manuscript revised September 20, 2010

The rest of this paper is organized as follows: Section 2 contains similar work and brief background about PLITS. In section 3, the architecture and the design of Arabic Tutor is introduced. Section 4 demonstrates how PLITS was used to develop the Arabic Tutor. In section 5, we introduce an assessment for the Arabic Tutor. In section 6, future prospects are discussed.

2. Similar Work and Background

In this section, we will highlight some of the previous similar work. Similar work in Intelligent Language Tutoring Systems (ILTS) can be divided into standalone ILTSs and Web based ILTSs.

Due to space limitations we will just highlight two examples of Web Based Intelligent Language Tutoring Systems. Web Passive Voice Tutor(Web PVT) [7] for English passive voice practice.is a web based ILTS for Grammar practice with four learner levels: Novice, Beginner, Intermediate, Expert. Its features include error specific feedback, individualization of the learning process, Intelligent analysis of students' solutions, and Adaptive navigation support.

The system architecture is composed of:
• The domain knowledge: Each node in the domain knowledge represents a concept category. There are three kinds of link between nodes: part-of, is-a, prerequisite
• Student module that contains student's information.
• Tutor module: That consults the student model in order to provide adaptive navigation support and to individualize the feedback.
•User interface module which controls the sequence of lessons.

Another example of web based ILTS is the German Tutor [6].

2.1 Background

This research uses PLITS [17] in the implementation of an intelligent tutoring system for teaching Arabic. PLITS is not a mere collection of patterns that can be used in the implementation of ITSs; it includes rules and guidelines that explain how and when to apply its patterns to solve a problem which is larger than any individual pattern can solve.

2.1.1 PLITS Pattern Categories

PLITS includes pedagogical, access, instructional, design, adaptive, and interaction patterns. Due to the fact that design patterns used in PLITS deals with low level design issues we will not discuss it in this paper as we aimed to concentrate more on high level design issues. However, we will refer to them briefly. More details on PLITS can be found in [17].

1. Pedagogical Patterns: are concerned with patterns that represent the best practices for teaching and education. The intent of pedagogical patterns is to capture the essence of the practice in a compact form that can be easily communicated to those who need the knowledge and to present this information in a coherent and accessible form [13].

2. Access Patterns: Access Patterns are concerned with the ways that users may access the various ITS resources [17].

3. Instructional Patterns: Instructional Patterns are concerned with the various tasks that tutors perform in order to create and edit courses and learning resources [17]. Both Instructional patterns and Access patterns are used in the Learning Management Systems domain.

4. Design Patterns: follows the original gang of four categories : creational, structural and behavioral[10].

5. Adaptive Patterns: Adaptive instruction can be defined as real-time modification of the instructional curriculum, learning environment to suit different student characteristics [17].

6. Interaction Patterns: Interaction Patterns are focused on solutions to problems that end-users have when interacting with systems. The patterns take an end-user perspective which leads to a format where usability is the essential design quality [17].

3. ARABIC TUTOR ARCHITECTURE AND DESIGN

3.1 The main features of the Arabic Tutor

1) Curriculum Sequencing: Curriculum sequencing provides the student with the most suitable individually planned sequence of knowledge units to learn and sequence of learning tasks (examples, exercises, etc.) to work with. In other words, it helps the student to find an "optimal path" through the learning material.
There are two kinds of curriculum sequencing techniques.

- High-level sequencing or knowledge sequencing determines next concept or topic to be taught.
- Low-level sequencing or task sequencing determines next learning task (problem, example, and test) within current topic.

2) The system displays only one error message at a time, and the student is expected to correct the error (and possibly any others) and resubmit the problem before any more feedback is displayed.

3) Arabic Tutor provides error specific feedback that differs in its level of detail according to student knowledge level per addressed topic.

4) The system has two interfaces one for the students who need to learn a certain material and to take exams. The other is for teachers who want to monitor student progress per topic.

3.2 The major components

Figure 1 shows the major use-cases of Arabic Tutor. It depicts different users of the system and most of the crucial use cases.

Figure 2 depicts the major components of the Arabic tutor:

1. Pedagogical Module (Tutor Module) provides the knowledge infrastructure necessary to tailor the presentation of the teaching material according to the student model.
2. Expert Module which uses Jess [8] (Java Expert System Shell) as a rule-based inference engine.
3. Domain Module contains the knowledge about the actual teaching material.

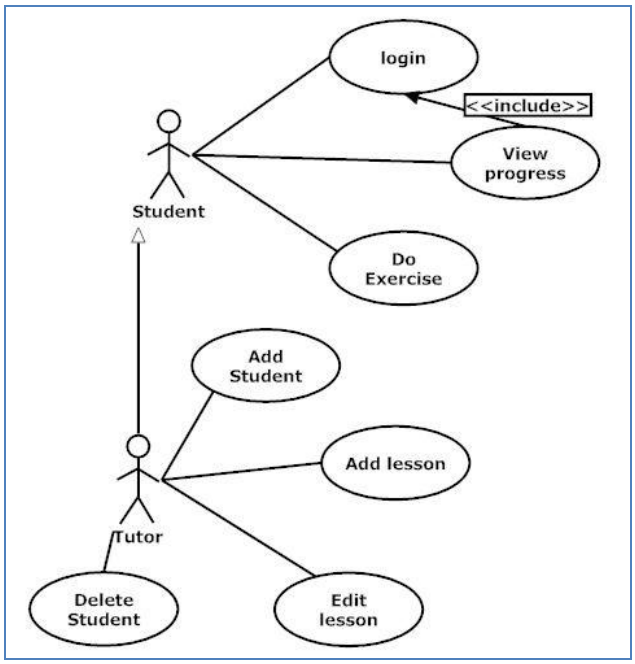

Figure 1 Use case diagram of the Arabic Tutor.

4. Student Model stores details about the student's personal data including name, e-mail, login name and topics that he is interested to learn and performance data which include the current level of mastery of learning material. The student model has the following components:
    a. Student Goals are either Long term or short term goals. Long term goals are learning topics of interest, while short term goals are the appropriate learn items and examples that should be studied to achieve long term goals. Long term goals are determined by students though they are prevented from choosing a topic without choosing its prerequisites and short term goals are automatically achieved through the tutor module.
    b. The Performance model which stores assessment of the student's overall skills and previous knowledge.
    c. The Teaching history model which keeps track of the material presented to the student during the session.

5. GUI Module which is divided into two parts:
    a. Student Interface is a set of dynamically constructed HTML pages whose content is determined by the tutor module.
    b. Teacher Interface which consists of easy interface to create teaching material (domain knowledge) that will be used by the tutor module in addition to the ability to access students' progress reports.

3.3 Used Technologies

- JDeveloper 10g: the development kit that was used. It has a robust testing environment for web applications. It uses the rapid application development (RAD) to build Java applications, JSP pages, applets, and java beans.
- JDK (Java Development Kit) 1.6: Was used as the implementation language. Java is currently very widespread in the internet community for its network flexibility and its innovative solutions.
- JSP 1.2: Was used to develop the dynamic pages for the web application
- Oracle database version 9.2
- Jess [18]: Jess is an interpreter for the Jess rule language. The syntax of the Jess rule language is similar to that of Lisp; it is well-suited to both defining rules and procedural programming. Jess has been used to develop a broad range of commercial software.

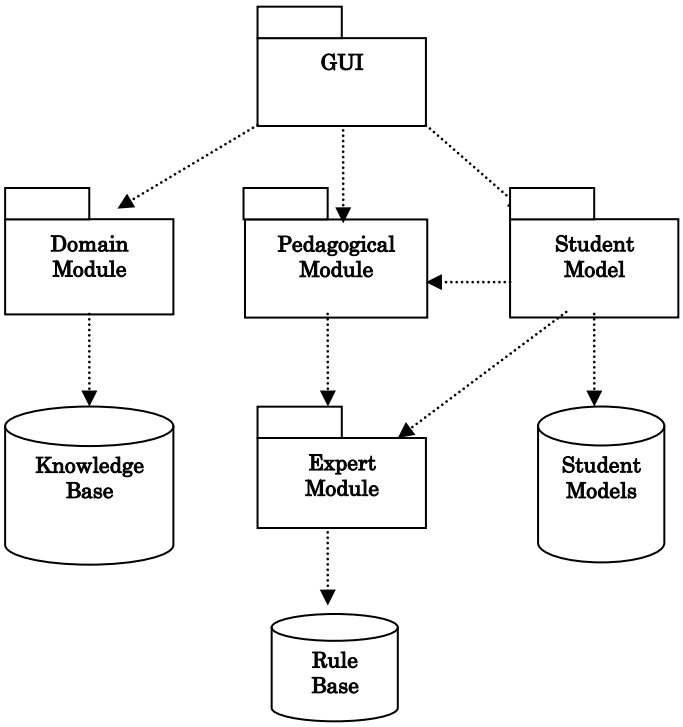

Figure 2 Architecture of Arabic Tutor

4. Using PLITS to develop the Arabic Tutor

The process of building a typical ITS is composed

of four phases; building the Domain Module, Student Model, Tutor Module and the Graphical User Interface Module. However, this is the typical sequence followed, nevertheless, it is not mandatory to strictly follow this order and some ITS developers use a different order for building ITSs.

PLITS is a pattern language for building intelligent tutoring systems [17]. In this section, we briefly talk about how we used the pattern language (PLITS [17]) in some of the components of the Arabic Tutor. Figure 3 depicts each module and the associated design patterns that were suggested by PLITS [17].

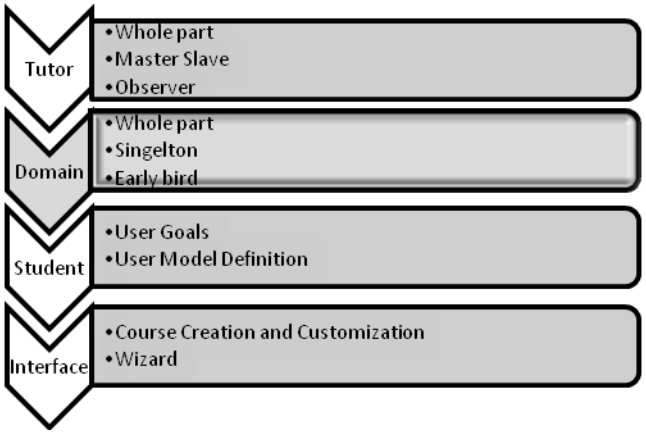

Figure 3 Patterns used in the Arabic Tutor.

In this paper, we will mainly focus at developing the Tutor Module and the Interface Module. As for the other modules, they were presented in previous publications [17],[19].

4.1 Developing the Tutor Module

4.1.1 Whole Part Pattern Usage

The Whole Part Pattern is one of the design patterns from the Pattern Oriented Software Architecture patterns [11]. It helps with the aggregation of components that together form a semantic unit. In ITSs teaching material is composed of a number of topics. The Whole Part Pattern can be applied in building the Domain Module in ITS design in composing topics, composing lessons and composing curricula. This pattern was used in [7] and [4].
In the Arabic Tutor, the pattern was used in Lesson Presentation Planner; we needed to build an agenda of the contents to be presented during the lesson. Each student may need to learn either a complex object (Topic; which can be divided into simple elements represented in a number of Learn Items) or a primitive object (Learn Item).

Exam Generation is another application of the Whole Part Pattern in the Arabic Tutor. This exam generator component utilizes the Whole Part Pattern. The Whole Part Pattern is used to represent aggregate objects that are more than just a mere collection of their parts. In addition it is used to represent a hierarchical relationship between objects [12]. Each Exam is composed of a number of exercises. The structure of the Whole Part Pattern in the Entry Exam of the Arabic Tutor is represented in figure 4:

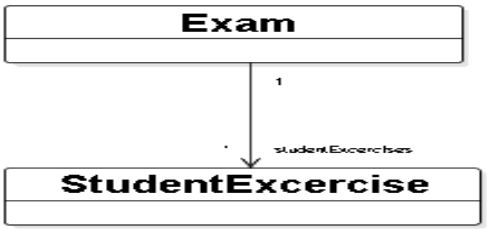

Figure 4 Whole Part Pattern Structure in the Arabic

As illustrated in figure 4, we have two participants:
•Exam Class (Whole Class): Represents a complex object and acts as a container for other primitive objects (StudentExercises). This class is responsible for choosing, organizing questions and computing student score.
•StudentExercise Class (Part Class): Represents the primitive elements. This class is responsible for determining the right answer, student answer, exercise difficulty level and student score per exercise.

Thus we benefited from the Whole Part Pattern in the separation of concerns between the Exam and the StudentExcercise Class. It therefore becomes easier to implement complex strategies by composing them from simpler services than to implement them as monolithic units.

4.1.2 Master Slave Pattern Usage

The Master Slave Pattern is one of the POSA patterns [11]. It is classified as a Design Pattern. It deals with the issue of supporting fault tolerance and computational accuracy. A master component distributes work to identical slave components and computes a final result from the results these slaves return.

The Master Slave pattern was used in the Arabic Tutor ITS to compute the exam results of each student according to their grade in each particular exercise within the exam.

The Master Slave Participants in the Arabic Tutor can be described as follows:

- **Master:** That is represented by the Exam object. Each exam contains a collection of

StudentExcercise. This class is responsible for calculating the overall student score based on scores calculated from StudentExcercise objects.

- Slave: That is represented by a collection of StudentExcercise.

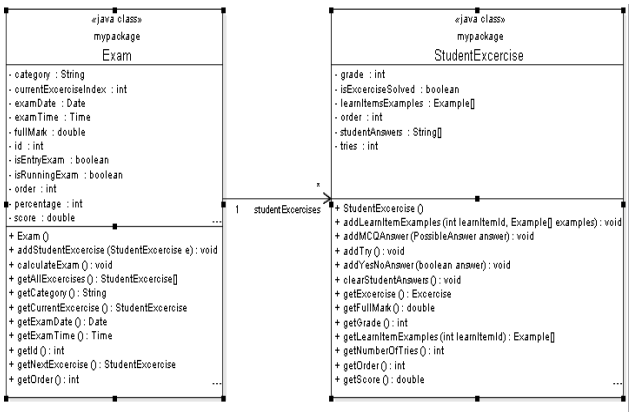

Figure 5 Master Slave Pattern in the Arabic Tutor.

All students are given a score on answering exercises that varies according to the question difficulty level.

The system uses the entry exam to initialize the student model by making a preliminary categorization for student levels per topic and per learning item; this categorization is then refined by observing the student performance in further exams while working with the system.

4.1.3 Observer Pattern Usage in the Arabic Tutor

The Observer Pattern is one of the GOF behavioral Patterns [10]. The observer pattern is used to define a one to many dependencies between objects so that when one object changes state, all its dependents are notified and updated automatically

In the Arabic Tutor, the observer pattern was used in order to observe the status of all students and update the teacher reports with the new student status as he navigates through the learning path. Thus if the teacher is monitoring the status of all students he will always see the up-to-date results of all students even if a student has just taken an exam.

The Observer Pattern Structure in the Arabic Tutor is illustrated in figure 6.

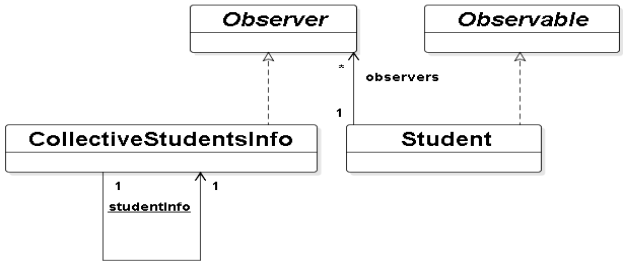

Figure 6 Observer Pattern Structure in the Arabic Tutor

In order to implement the Observer Pattern we followed the following steps:
1. Creating the Subject Participant: This is represented by the Observable interface. That provides an interface for attaching and detaching Observer objects. The Subject Participant knows its Observers. Any number of Observer objects may observe a subject.
2. Creating the Observer Participant. This is represented by the Observer interface. This Participant defines an updating interface for objects that should be notified of changes in a subject.
3. Creating the Concrete Subject Participant. This is represented by the Student Class. This participant stores state of interest to Concrete Observer objects and sends a notification to its Observers when its state changes.
4. Creating the Concrete Observer Participant. This is represented by the CollectiveStudentInfo class. This participant maintains a reference to Concrete Subject object and stores state that should stay consistent with the subject's and implements the Observer updating interface to keep its state consistent with the subject's.

4.2 Developing the Graphical User Interface Module

4.2.1 Course Creation and Customization Pattern Usage

This pattern is used in learning management systems [20] which attempts to answer the question of how can the instructors be assisted in building on-line courses in ITS so that some of the tasks they need to perform can be automated in order to decrease the time and effort of performing those tasks? It provides the instructors with appropriate tools for creating and customizing a course. The creation of courses can be based on design templates with pre-set interfaces, content structure and features.

The Known Uses of the Course Creation and Customization Pattern were discussed in Web-based

distance learning environment [21].

The Arabic Tutor used the Course Creation and Customization Pattern in providing the teachers with a web interface for adding, deleting or editing the course curricula and monitoring the student performance and progress.

4.2.2 Wizard Pattern Usage

The Wizard pattern is an example of User Interface Patterns (Interaction Patterns) [24]. It deals with the issue that the student wants to perform an infrequent complex task consisting of several subtasks which ranges between 3 to 10 tasks where decisions need to be made in each subtask may not be known to the user.

It offers a solution to this issue by taking the user through the entire task one step at the time. Letting the user step through the tasks and showing him which steps exist and which have been completed .When the complex task is started, the user is informed about the goal that will be achieved and the fact that several decisions are needed. The user can go to the next task by using a navigation widget such as a button. If the user cannot start the next task before completing the current one, feedback is provided indicating that the user cannot proceed before completion.
The users are given feedback about the purpose of each task and can see at all times where they are in the sequence and which steps are parts of the sequence.

The Wizard Pattern was used in the Arabic Tutor to allow the student to reach their short term goals by taking them through the entire complex task one step at a time. This was achieved by using a navigation widget represented by a button. The navigation buttons suggest to the students that they are navigating a path with steps. Each task is presented in a consistent fashion enforcing the idea that several steps are taken.

If the student cannot start the next task before completing the current one, feedback is provided indicating that the student cannot proceed before completion; this feedback is given by disabling the navigation widget. The user is also able to revise a decision by navigating back to a previous task. When the complex task is completed, feedback is provided to show the user that the tasks have been completed and optionally results have been processed.

5. Assessment

As mentioned earlier the Arabic Tutor was tested and evaluated on Arabic Language Institute students in the American University in the Arabic Writing Program. The following feedback was collected from both the students and the teachers:

5.1 Arabic Tutor Advantages

- One of the strongest features in the Arabic Tutor is allowing students to provide natural language input by typing a verb with specific features rather than selecting exclusively from among pre-defined answers as is the case in a number of other ITSs.
- Learner controlled pacing.
- Small units of instructional material.
- Providing learners with different versions, different difficulty levels, etc., based on their performance and previous knowledge.
- Immediate feedback that is tailored according to the learner level and error made.
- Online testing and grading.
- The ability to retake tests until mastery is demonstrated.

5.2 Arabic Tutor Disadvantages

- The GUI needs to be improved.
- Audio examples should be used to improve the learning experience.

6. Conclusion and future work

This paper describes how PLITS [17] was used to implement the Tutor and interface modules of the Arabic Tutor. The Arabic Tutor is a web based ITS for teaching a subset of the basic rules of the Arabic language that combines the flexibility and intelligence of ITSs with the availability of the World Wide Web applications. PLITS was very useful in guiding us through the process of design and implementation of the Arabic Tutor. It raised the level of abstraction to patterns rather than classes and gave us chance to focus more on the details of the Arabic Tutor.

Future enhancements can include:

1. Increasing the scope of the Arabic Tutor domain module to include a wider range of topics.
2. Introducing pedagogical agents into the Arabic Tutor implementation.
3. Providing the tool as a set of web-services for teaching Arabic.

References

[1] Liegle, J., and Gyun Woo, H.Developing Adaptive Intelligent Tutoring Systems: A General Framework and Its Implementations. In The Proceedings of ISECON 2000, Vol.17 (Philadelphia).

[2] Heift, T.Intelligent Language Tutoring System for Grammer Practice,2001. http://www.spz.tu-darmstadt.de/projekt_ejournal/jg-06-2/beitrag/heift2.htm. View Date May 2004.

[3] Heift,T.,Toole,J.,McFetridge,p.,Popwich,F.,Tsiplakou,S. An Interactive Course Support System forGreek.Bourdeau,J.&Heller,.(eds).Proceedings of ED-MEDIA 00,World Conference on EducationalMultimedia,Hypermedia&Telecommunications, Charlottesville,VA:AACE:394-399,2000.
[4] Mayo,M.,Mitrovic,A.,McKenzie,J.CAPIT: An Intelligent Tutoring System for Capitalisation and Punctuation. International Workshop on Advanced Learning Technologies,Palmerston North, New Zealand,2000.
[5] FUM ,D.,Giangrandi,P., Tasso,C.Tense Generation in an Intelligent Tutor For Foreign Language Teaching:Some Issues in The design of the Verb Expert.In Proceedings of 4th Conference of the European Chapter of the Association for Computational Linguistics,Pages 124-129,Manchester.Association for Computational Linguistics,1989.
[6] Heift, T. An Interactive Intelligent Tutor over the Internet. Otman, T. & Tomak, I. (eds). Proceedings of ED-MEDIA 1998, World Conference on Educational Multimedia, Hypermedia & Telecommunications, Charlottesville, VA: AACE: 556-560,1998.
[7] M. Virvou & V. Tsiriga .Web Passive Voice Tutor: an Intelligent Computer Assisted Language Learning System over the WWW. IEEE International Conference on Advanced Learning Technologies pp 131 – 134, Wisconsin, USA, 2001.
[8] Ernest Friedman-Hill, Jess in Action, Java Rule-based Systems, Ernest Friedman-Hill, Manning Publications July 2003.
[9] Antonija Mitrovic. An Intelligent SQL Tutor on the Web.International Journal of Artificial Intelligence in Education, 2003.
[10] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides. Design Patterns, Elements of Reusable Object-Oriented Software, Addison Wesley Professional. First Edition. 1995.
[11] Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, Michael Stal, Peter Sommerlad. Pattern-Oriented Software Architecture:A System of Patterns.John Wiley & Sons,2000.
[12] R. Nkambou and Claire IsaBelle. "Cyberphysique: A Web-based distance learning environment". In: Global Education on the Net, Vol. 1, pp.252-256. Springer-Verlag (Berlin) (1998).
[13] Bergin, Joseph. Fourteen Pedagogical Patterns. Proceedings of the Fifth European Conference on Pattern Languages of Programs, July 2000, Irsee, Germany.
[14] Devedzic, V., Jerinic, L., & Radovic, D. The GET-BITS Model of Intelligent Tutoring Systems. Journal of Interactive Learning Research 11(3), pp 411-434, 2001.
[15] Vladan Devedzic, Andreas Harrer. Common Patterns in ITS Architectures. KI 18(3), Pages 17-21, 2004.
[16] http://en.wikipedia.org/wiki/List_of_languages_by_number_of_native_speakers
[17] Dina Salah, Amir Zeid, PLITS:A Pattern Language for Intelligent Tutoring Systems" at 14th European Conference on Pattern Languages of Programs, July 2009, Munich, Germany.
[18] Ernest Friedman-Hill, Jess in Action, Java Rule-based Systems, Ernest Friedman-Hill, Manning Publications July 2003.
[19] Dina Salah, Amir Zeid, Adaptive patterns for Intelligent Tutoring Systems, " at 14th European Conference on Pattern Languages of Programs, July 2009, Munich, Germany.
[20] P.Avgeriou, A.Papasalouros, S.Retalis. Patterns For Designing Learning Management Systems. European Pattern Languages of Programming (EuroPLOP), 25th-29th June 2003, Irsee, Germany.
[21] R. Nkambou and Claire IsaBelle. "Cyberphysique: A Web-based distance learning environment". In: Global Education on the Net, Vol. 1, pp.252-256. Springer-Verlag (Berlin) (1998).
[22] Mayo,M.,Mitrovic,A.,McKenzie,J.CAPIT: An Intelligent Tutoring System for Capitalisation and Punctuation. International Workshop on Advanced Learning Technologies,Palmerston North, New Zealand,2000.
[23] Heift, T., Toole, J., McFetridge, P., Popwich, F., Tsiplakou, S. Learning Greek with an Intelligent and Adaptive Hypermedia System. IMEJ of Computer-Enhanced Learning, Volume 2(2), October 2000.
[24] M. van Welie, H. Trætteberg .Interaction Patterns in User Interfaces.7th. Pattern Languages of Programs Conference, 13-16, Allerton Park Monticello, Illinois, USA, August 2000.

Amir Zeid received his Ph.D. from Carleton University, Canada. He is currently the Program Lead of computer science and information systems at the American University of Kuwait. His research interests include Software Engineering, Computer Education and Cultural issues in Computing. He is a member of ACM

Dina Salah is a Ph.D. candidate at the University of York, UK. She received her M.Sc. from the American University in Cairo. Her main research interests include intelligent tutoring systems and Agile modeling.