E-EVALUATION TO DETECT LEARNING WEAKNESS
An Agent Based Architecture

Francisca Losavio
LaTecS Laboratory
Centro ISYS, Universidad Central de Venezuela

Nicole Levy, Parinaz Davari
Laboratoire PRISM
Universite de Versailles St Quentin

Keywords: E-Evaluation, E-Learning, Agent-Based Software Architecture.

Abstract: A multi-agent architectural framework for evaluation in e-learning situations is proposed. The idea is to enable e-learning students to detect precisely their weaknesses in some goals within a course program. A course is composed of several objectives or goals to be mastered by students; a student can master some, but not necessarily all of them. Each of these objectives are in turn composed of sub-objectives.
The architecture is focused as a tree of intermediate goals, where each node corresponds to a goal to be mastered, as each of its sub-goals. The flexibility of the architecture being a major concern, the goals to be mastered are dynamically defined based on the results of tests passed by the student. The corresponding educational materials are searched on demand and they can be located anywhere on the Web. The quality requirements of the architecture are specified and justified by a standard quality model. Our approach is illustrated with a case study of a computer architecture course.

1 INTRODUCTION

Architectural design is a stepwise process which identifies the key strategies for the large-scale organization of software systems under development (Krutchen, 1999). Functional components (derived from functional requirements) and their crosscutting concerns (components derived from non functional requirements) must comply with the quality goals established by the system requirements (Kiczales et al., 1997) (Sousa et al., 2004). We propose and justify an architectural framework that can be used in multiple domains related with problem solving, in particular, with scientific calculation and auto-evaluation in education. The classical problem solving divide and conquer strategy is transformed into a divide and "reuse" approach. The basic idea is to reuse previous results that are required in a particular problem solution. In an e-learning context, the results of an auto-evaluation situation are the goals that the student has to master with the corresponding educational material and they can be located anywhere on the Web. The flexibility of the architecture is crucial since the results must be dynamically obtained on demand. The quality aspects of the overall architecture are specified using a standard quality model based on ISO/IEC 9126-1 (ISO/IEC, 2001). This work is part of the EEC e-Forminfo project.

The goal is to apply the architecture to an auto-evaluation platform for a continuous formation program in the e-learning domain. Students in this educational program usually have incomplete knowledge on the different subjects they have to master, but they are not aware of these deficiencies. The evaluation platform aims at both detect these weaknesses and provide related e-learning material.

The goals are structured as a tree of intermediate goals, where each node corresponds to a goal to be mastered within an educational program, as each of its sub-goals. Each node corresponding to a goal is composed by the following information:

- An Evaluation to determine whether the specific knowledge related to the node is mastered. This evaluation depends on some required sub-knowledge;
- Educational program for a course, introducing or referencing the sub-goals;
- Educational documents (course literature, exercises, references, etc.);

This document is structured as follows, besides this introduction: Section 2 describes the requirements for an auto evaluation system. Section 3 presents the architecture and its justification. In section 4 a case study on an evaluation example is presented. Some related works are discussed in Section 5. Finally, the conclusion provides final remarks and perspectives.

2 REQUIREMENTS FOR AN AUTO-EVALUATION SYSTEM

2.1 Requirements Specification

The aim of an auto-evaluation system is to help students to detect their knowledge gaps or weaknesses in order to orient them toward parts of the focused course that they should learn more deeply. Such auto-evaluation are to be done in particular, when preparing an examination. Each student is registered in a group with a specific goal. To reach this goal the student has to pass (master) several courses, which in turn have specific goals. Advisors must have defined the goals. The tutors are responsible for providing documental support and the evaluations for each course. The courses are hierarchically ordered.

Students can be connected to Internet in two different modes or situations: prepare for an exam (Training mode) and pass the exam (Evaluation mode).

Training mode: The student logs to the system and selects a course or a specific goal among those offered by the system. The system must propose to the student some e-learning material corresponding to the selected course goals. The student decides to be evaluated either for the complete course or some specific goal within the course. The system generates automatically a corresponding evaluation. The student fulfills it. For each wrong answer, the system points out the non mastered learning goals. There is no time constraint in this mode; the student can spend any time he/she wants to study and revise any goal. He/she must also be able to restart a session in the same state as he/she left.

Evaluation mode: The student passes the exam. The system must check whether he has passed the prerequired courses. The system corrects the exam and adds the results to the student record. There may be time constraints in this mode.

2.2 Non functional Requirements

Non functional requirements for this system are:

- Actors communicate through a network browser.
- Students must be precisely identified.
- Course (and goal) documentation must be available, updated and retrieved on demand.
- Courses (and goals) must be easily modified.
- Evaluations (for goals) must be available, updated and retrieved on demand.
- Students must be offered a legible and customized information on the goal or subgoals required.
- Response-time is critical in the evaluation mode.

3 AN ARCHITECTURE TO EVALUATE THE STUDENT KNOWLEDGE

An architectural solution is proposed to fulfill the system's requirements discussed in the previous section. The proposed architecture models the statical view of a course as a tree of intermediate objectives or goals. Each node corresponds to a goal to be mastered by the student, together with mastering all the sub-objectives required. Some evaluations provided allows the system to detect the weaknesses in the mastering of the precise goals. Lectures documentation or bibliographical information can be retrieved when a goal is not mastered as *e-learning material* (called courseware in (Garro and Palopoli, 2002) and educational content in (Pankratius et al., 2004)).

The basic idea of our architectural framework is the following: each node represents an objective of a course, which is presented in a top-down decomposition, possibly introducing sub-objectives to be learnt in turn. For a course objective, the system will propose one of the evaluations to the student. From the results obtained by the student, the system will deduce the sub-goals not yet mastered. The system will develop the tree introducing sub-objectives associated to the new nodes required.

Since a natural idiom for encoding and executing goals is through software agents (Wooldridge and Jennings, 1995), we propose to use a multi-agent style for this architecture, and this choice will be justified on the basis of the quality requirements for each agent.

3.1 Agents Model

We have identified three kinds of agents:

Course Agent: A Course Agent represents a precise objective (a node) and its decomposition into sub-objectives. It is statically defined (by some tutor). It knows the course name or course program, the required context of the e-learning situation and the reference to an associated Evaluation Agent, all the sub-goals to be mastered represented by their respective Course Agent and references to some e-learning material related to the goal.

Evaluation Agent: An Evaluation Agent represents a precise evaluation method. It is statically defined (by some tutor). It knows the reference to the Course Agent it will evaluate, the required context of the e-learning evaluation situation, a method to generate equivalent evaluations.

Student Course Agent: A student, to evaluate if he/she masters a current course, will create a Student Course Agent. This agent will know all the information concerning his/her mastering situation concerning the course.

3.2 Evaluating Algorithm

When some student wants to evaluate his/her knowledge, he/she first looks for a course objective. The corresponding Student Course Agent will be created. This agent will start its own evaluation process.

1. The Student Course Agent looks for Course Agents corresponding to the objective. The student may then select one among those found.
2. The Student Course Agent asks for an evaluation to the Evaluation Agent associated to the chosen Course Agent.
3. The student fulfills the evaluation.
4. The corresponding Evaluation Agent corrects it and delivers a result.
5. If it is totally correct the objective is said to be mastered.
6. If some parts of the evaluation failed, then sub-goal are not mastered and the Agent will create sub-Student Course Agent associated to their respective course agents.
7. The student will evaluated himself for all the introduced sub-goals according to the same process.
8. The student can re-execute a new evaluation whenever he/she wants, even if the sub-goals are not said mastered. When re-executing an evaluation, a new list of sub-goals may be introduced.

Justification of the architecture: quality model for the problem domain and analysis of the agents quality responsibility

- *Modifiability: changeability* . Both the Course and the Evaluation agents are responsible of this goal
- *Functionality: interoperability, suitability, accuracy, security* . Interoperability is a common property of the agents in a multi-agent architectural style, where data structure play a central role, such as for example the standard XML for data exchange; the suitability or precise execution of the required functions on the node, is accomplished by the Student Course Agent; accuracy is the responsibility of the Evaluation Agent; security is required for the precise identification of the student for the login functionality and will be performed by a service of the multi-agent platform.
- $Usability$: $legibility$ and $operability$. They will be provided by the corresponding GUI component of the agents involved
- *Efficiency: time behavior (response-time)* . It is also provided by the agents communication protocol of the multi-agent platform.

We use an Agent oriented architecture to achieve these non functional properties. Agents hold extensibility and flexibility, meaning that they can be easily created, destroyed and modified.

4 EVALUATION IN COMPUTER ARCHITECTURE

Let us take as example an evaluation in Computer Architecture to evaluate the knowledge about parameter passing in assembly language programming. The important concepts to be mastered are the following:

Parameters are values that you pass to and from a procedure. Two major mechanisms for passing data to and from a procedure are studied: (i) pass by value, (ii) pass by reference.

Let Search be a procedure needing as parameters an array `tab` of at most 100 characters (representing a long string), an integer `n` in [0..100] (representing the size of `tab`) and a character `ch` (representing the one to be searched in `tab`) and as result `res` an integer in [0..101] (representing the address of the first occurrence of `ch` in `tab` or `n+1` if `ch` is not found). Then the following question could be asked:

Give the instructions of the calling program to put these parameters on the stack

The answers must be written in the following table:

	The calling program
a	
b	
c	
d	
	`Call Search`

For each answer to be given by the student, a set of possible answers are defined. For example for answer (a), the student can either give: `push offset tab` or the two instructions `lea bx,tab` and `push bx`.

In the same way, a possible list of errors are defined and for each one, a set of badly understood parts of the course. For example, again for the answer (a), one possible error is ti give as answer `push tab`. Then a bad understanding of the difference between the address denoted by a variable and its value will be detected and it will be recommended to study the definition of a variable, the definition of the keyword `offset` and the definition of the instruction `lea`.

We can imagine that when evaluating a student, if the answer given is not found in one of the possible answers defined, the system will send it to the tutor who will complete the evaluation with a new possible erroneous answer.

5 RELATED WORKS

In the educational problem domain, two main architectural styles emerges as solutions: *services oriented architecture (SOA)* and *multi-agent system (MAS)*.

The SOA systems for Learning Technology are based on the IEEE 1484 Learning Technology Standard Commitee (LTSC) with the participation of the Educom's Instructional Management System (IMS) projet of the Global Learning Consortium, among others (Farance and Tonkel, 1998).

Some platforms such as (Garro and Palopoli, 2002) or (Fernández-Caballero et al., 2003) use MAS. In (Garro and Palopoli, 2002), the agents help enriching, sharing and circulating organization knowledge and make the organization dynamic and flexible. (Fernández-Caballero et al., 2003) introduces three MAS in the architecture: The Interaction MAS, which captures the user preferences. The Learning MAS composes the contents for the user in accordance to the information collected by the Interaction MAS and the Teaching MAS offers recommendations of how to enhance the layout of the Engineering course.

In our case, if we consider roughly an agent as an autonomous software component requiring and providing services, we can establish a correspondence between the SOA (Service Oriented Architecture) style and the multi-agent style: the logical framework properties correspond to the general properties of the multi-agent style which are accomplished internally by the agents composing the evaluation system (Course, Student Course and Evaluation Agents), the data representation is expressed as an exchange facility that all the agents should accomplish to achieve interoperability and the communication and interfaces aspects concern the protocols used by the agents to communicate among themselves and with its environment, which is responsibility of the platform used to implement the style.

6 CONCLUSION

In this paper, we have proposed a multi-agent based architectural framework to evaluate students knowledge weaknesses in an e-learning context. Modifiability (Flexibility to changes) is the main concern of our multi-agent architecture, since it must respond to dynamic changes: evaluation agents must be created on demand for specific goals, a student may ask for an evaluation for any goal in a course and static changes on the proposed evaluations and on the e-learning material can occur at any time. As a consequence, the contextual properties of our agents must be designed emphasizing the flexibility aspect. The usability aspect must be compliant to the standards imposed by educational e-learning systems and could be solved using web services or using a GUI agent. Functionality and efficiency will also be considered within the multi-agent platform implementing the framework. Let us note that currently, the implementation is an ongoing work.

REFERENCES

Farance, F. and Tonkel, J. (1998). Learning technology systems archcitecture (ltsa) specification. In *Version 4.0, draft*, page http://www.edutool.com/ltsa.

Fernández-Caballero, A., López-Jaquero, V., Montero, F., and González, P. (2003). Adaptive interactor multi-agent systems in e-learning/e-teaching on the web. In *ICWE*, pages 144–153.

Garro, A. and Palopoli, L. (2002). An xml multi-agent system for e-learning and skill management.

ISO/IEC (2001). Software engineering - product quality. Technical Report IEC 9126-1, ISO.

Kiczales, G., Lamping, J., Mendhekar, A., Maeda, C., Lopes, C., Longtier, J.-M., and Irwin, J. (1997). Aspect-oriented programming. In Matsouka, M. A. . S., editor, *ECOOP97 Object-Oriented Programming*, pages 220–242. LNCS,vol. 1241.

Krutchen, P. (1999). *The Rational Unified Process*. Addison Wesley, Massachusetts.

Pankratius, V., Sandel, O., and Stucky, W. (2004). Retrieving content with agents in web service e-learning systems. In *The Symposium on Professional Practice in AI, IFIF WG12.5 - First IFIP Conference on Artificial Intelligence and Innovations (AIAI)*, pages 144–153.

Sousa, G., Soares, S., Borba, P., and Castro, J. (2004). Separation of crosscutting concerns from requirements to design: Adapting an use case driven approach. In *proceedings of AOSD-2004*, Lancaster, UK. Early Aspect 2004.

Wooldridge, M. and Jennings, N. (1995). Intelligent agents: Theory and practice. *Knowledge Engineering Review*, 10(2):115–152.