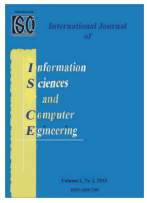
Information Sciences and Computer Engineering, Vol. 1, No. 1, (2010) 9–15

PHARUS

International Journal of
Information Sciences and Computer Engineering

journal homepage: http://www.ijisce.org



Modelling Time Control in Evaluation Process of E - learning Platforms by Automata Formalism

Mirna El-Hajj-Barbar[a], Youssef Monsef[b], Kablan Barbar[c,*], Bilal Chebaro[c]

[a] *Faculty of Economics, Lebanese University, lebanon*
[b] *Faculty of Engineering, Lebanese University, lebanon*
[c] *Faculty of Sciences, Lebanese University, lebanon*

***Abstract*– This paper shows that automata formalism is applied as data structure and running system models to express time control in the evaluation process. Therefore, at the implementation level, new questionnaires description language and evaluation system architecture are introduced for time control over questionnaires, in translating the automata structure, respectively, in the XML and client-server environments. Moreover, the automata formalism is extended and enriched with the concept of attributes in order to evaluate time at each automata state (or test).**

Keyword: Evaluation, assessment, tests, automata, time control, attributes, XML

1. Introduction

In this paper, we show that time control can be integrated in the automata formalism designed for an educational evaluation process. We introduce new questionnaire description language and evaluation system architecture in translating the automata structure, respectively, in the XML and client-server environments.

Most of evaluation platforms ([1, 2, 3, 4] and [5]), introduce new tests description languages that are presentation oriented. They offer many tools to create and present tests that are not based on a conceptual model. Therefore, evaluation characteristics like scoring and time control are partially integrated in the tests languages without any formal proofs. On the other hand, the XML environment is the best to implement educational tests ([6, 7], and [8]) because it offers the techniques of XSL style sheets to present XML documents and the DOM interface to manipulate XML documents in Web pages. For this reason, we are interested to define a conceptual model for the evaluation process, to study the integration of the evaluation characteristics at the conceptual level and finally to implement it in the XML environment. Our conceptual model for the evaluation process is based on the automata formalism.

We recall that automata structures contain several components; different states, a transition function, and an output function. We make use of this structure to model an educational evaluation process which is based on tests and walking strategies over those tests. In [9], tests are translated into states, and walking strategies into transition functions. Now, time control on tests and walking strategies can be integrated in an automata structure in different ways. It can be attributed to either the states, or the transition function, or the output function. In some cases, time control has to be described in terms of attributes attached to states, similar to attributes associated with nonterminals in the concept of attributed grammar, introduced by [10] for automatic generation of compilers for programming languages. On the pedagogical level, time control is defined by a) associating duration either with a question or with an entire questionnaire, and by b) comparing the duration the learner takes to answer a question with the duration of tests or of the questionnaire.

In the first case, only the duration value, which is static information, is integrated into the test structure. A test therefore becomes a triplet constituted by a question, a set of responses and a duration. Time control is then used to check whether the learner exceeds the defined duration of a given test or not – what is taken into consideration in the determination of the next test. In order to be able to do that, we have considered three temporal operators: *before*, *after* and *any*. Additionally, these three temporal operators have been integrated into the alphabet of the automata which is the Cartesian product of the power set of responses and the set of the temporal operators. This means that time control on each test is associated with the transition function of the automata.

In the second case, when duration is associated with the entire questionnaire, the problem can be solved by introducing the new concept of attributed automata; a variable attribute called "duration" is associated with each *state* of the automata, and the learner's answer duration is attributed to the automata's *output function*. Therefore, at each transition from state t to state t', the value of the duration attribute of the state t' is calculated in terms of the duration attribute of the state t and the output function.

We note that there are some conceptual models for assessment in general like in ([11, 12, 13] & [14]), but as we know, there are no studies in the field of modeling time control in assessment process.

*Corresponding author:
Email address: kbarbar@ul.edu.lb, Ph: +96 13432455

This paper is organized in the following way. Section 2 illustrates the approach by using a quiz example. Section 3 studies the problem of time control by setting a duration for each test, while section 4 deals with the problem of setting a duration for the entire questionnaire. Section 5 proposes an implementation for our automata model with integrated time control for both cases. The conclusion finally discusses the obtained results and the advantages of the approach.

2. Modeling Questionnaires by Automata

In E-learning domain, automata are used to describe the questionnaire structure. The states structure implements the tests components whereas the graph of tests is represented by the transition function. Here, automata play the role of data structure in comparison of the abstract machine role assumed by automata in compiler theory. Now, we are going to give the automata formal description of the questionnaire structure. Let Q be the set of questions and R the set of responses. We define the set of tests T by $Q \times P(R)$ where P(R) is the power set of R and $\times$ is the Cartesian product. Then, an automaton for questionnaire is defined by:

$A=< \Sigma, T, t_0, F, \delta >$ where

- Σ =P(R), is the input alphabet,
- $T = Q \times P(R)$ is the set of states,
- t_0 is an element of T and the initial state,
- F is the set of terminal states and is a subset of T,
- δ is the transition function: $\delta : T \times P(R) \rightarrow T$.

Example 1: We consider the example of questionnaire on the comprehension of the concept of "pointer" in the programming language given in [15]. The list of tests comprises:

Tests	Questions	Answers
t_0	q_0 : what is the best definition for a pointer ?	r_1 : a pointer is a variable r_2 : a pointer is an address r_3 : a pointer is a function
t_1	q_1 : what is the type of a pointer variable?	r_4 : a memory address r_5 : it depends on the pointed object type
t_2	q_2 : can we use a variable by giving its location in memory?	r_6 : true r_7 : false
t_3	q_3 : can we store one variable into another?	r_8 : true r_9 : false
t_f	final :	

We consider a walking strategy that works in the following way: if the learner's answer is wrong, he or she is sent into a loop of questions (questions t_2, t_3 and t_0), given a chance (question t_1) to advance if his answer is quite correct. This strategy is represented by the graph in figure 1 and the corresponding automaton is $A=< \{t_0, t_1, t_2, t_3, t_f\}, \{ r_1, \ldots, r_9\}, t_0, \{ t_f\}, \delta >$ where δ is given in table 1.

Table 1. The transition function.

Function δ	t_0	t_1	t_2	t_3	t_f
r_1	t_f				
r_2	t_1				
r_3	t_2				
r_4		t_2			
r_5		t_f			
r_6			t_3		
r_7			t_3		
r_8				t_0	
r_9				t_0	

3. Local Time Control

In the local time control, we fix a duration by test and we try to implement it in the automata model. We consider time control as a component of the evaluation process. Time control must therefore intervene in the transition function which determines the advancement from one test to the next.

Let $t=(q,\{r_1, \ldots, r_q\})$ be a test where q is the question and r_1, $\ldots$, r_q are some possible responses to the question q, and m its authorized duration. There are three possible cases when we present the test t to a learner:

- Case 1: the learner gives the correct answer r_i *before* the expiration of the duration m.
- Case 2: the learner gives the correct answer r_i *after* the expiration of the duration m.
- Case 3: the learner gives the wrong answer r_i *before or after* the expiration of the duration m.

Let us consider the example of the above quiz by giving a transition for each of the above three cases:

- Case 1: the learner gives the correct answer r_1 for the test t_0 *before* the expiration of the authorized duration m_0 (associated with t_0), then the system goes to the test t_f.
- Case 2: the learner gives the correct answer r_1 for the test t_0 *after* the expiration of the authorized duration m_0 (associated with t_0), then the system goes to the test t_1.
- Case3: the learner gives the wrong answer r_3 for the test t_0 *before or after* the expiration of the duration m_0 (associated with t_0), then the system goes to the test t_2.

On the formal level, we have three temporal operators *before*, *after* and *any* that correspond respectively to case1, case2, and case3. These operators express the duration of the learner's answer compared to the authorized duration.

The solution for implementing time control is to integrate :

- the duration in the *state (or test) structure* and,
- the temporal operators in the *transition function* of the automata.

Let Q be the set of questions, R the set of responses and M be the set of durations such that to each question q of Q there an associated duration m in M. Then, $T=Q\times P(R)\times M$ is the set of tests. Let and θ = {before, after, any} is the set of temporal operators. Therefore, the automaton for local time control becomes:

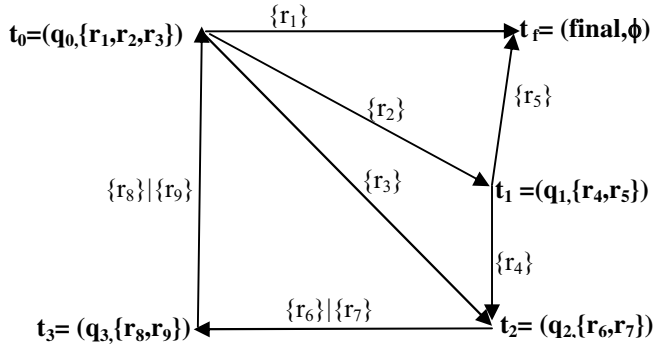


Fig. 1. The graph of tests.

Table 2. Automata transitions.

Input / State → ↓	t_0	t_1	t_2	t_3	t_f
(r_2,before)	t_1				
(r_4,before)		t_2			
(r_7,after)			t_3		
(r_8,before)				t_0	
(r_1,before)	t_f				

$$A=< Q \times P(R) \times \boldsymbol{M}, P(R) \times \theta, t_0, F, \delta > .$$

We notice that the input of the automata is an element of the alphabet $P(R) \times \theta$ that is a pair of a subset of responses and a temporal operator. This means that learners are judged not only by their answers but also by the time they take to answer the tests.

Example 2: Let us consider the quiz definition in example 1 with M={60,40,. . . ,40,0} as a set of durations. The automaton $A = < \sum, T, t_0, T_f, \delta >$ will be the following:

- $\sum = P(\{r_0, \ldots, r_9\}) \times \{before, after, any\}$ is the alphabet
- $T = \{t_0, t_1, t_2, t_3, t_f\}$ is the set of states
- $t_0$is the initial state
- $T_f = \{ t_f\}$ is the set of terminal states
- δ is given by the graph in figure 2.

During an evaluation session, the learner advances through the tests in the following order **:** t_0, t_1, t_2, t_3, t_0, t_f(figure 2), and answers respectively with r_2,r_4, r_7, r_8, and r_1; the duration for $r_2$is 50, 35 for r_4, 50 for r_7, 35 for $r_8$and 50 for r_1. This provides the following input values for the automaton (r_2,before), (r_4,before), (r_7,after), (r_8,before) and (r_1,before). The path from the first test to the last one is expressed by automata transitions, and when using temporal operators we get the array in table 2.

4. Global Time Control

In global time control, a total duration is set for the entire questionnaire that concerns all tests. When the learner answers a given test, this total duration will be decremented by the time that he or she has taken to answer the test. To do this calculation to the level of every test, one must keep information on the remaining time during the transitions between tests. This information is calculated at every transition after the learner gave his answer to the current test. Then, one proposes a solution that answers two questions:

where to place information on the remaining time?

how to calculate the remaining time in link with the transitions of the automaton?

The solution consists in:

- associating an *attribute* to every test (or state of the automaton),
- using the *output function* to express the remaining time at each transition,
- defining a *mechanism of calculation* that works with transitions (see figure 3),
- starting the calculation after *initialization of the attribute of the initial test*. The initial value is the global duration of the questionnaire.

In order to describe the global time control, we need to give an extension of the automata formalism, applied to the evaluation domain, by introducing the new concept of attributes. In the new formalism, we describe, at the same time, the local and the global time control. Let be the following:

- Q is a set of questions,
- R is a set of responses,
- M is a set of local durations associated with tests,
- θ is a set of temporal operators,
- *globalDuration* is the duration associated with the whole questionnaire.

An *attributed automaton* is an 8-uplet :

$A = < \sum, T, t_0, T_f, \delta, \Lambda, \lambda, ATT, E >$ where:

- $\sum = P(R) \times \theta$ is the alphabet,
- $T = Q \times P(R) \times M$ is the set of states (or tests),
- $t_0$is the initial state,
- T_f is the set of terminal states,
- δ is the transition function $\delta : T \times \sum \rightarrow T$,
- λ is the output function, $\lambda : T \times \sum \rightarrow \{reals\}$
- $\lambda(t,(r, \alpha)) = d$ is the time taken by a learner to give the response r to the test t,
- ATT = {rTime} is the set of attributes, and *rTime* represents the remaining time at the level of tests,
- E is a set of attributes equations defined by :

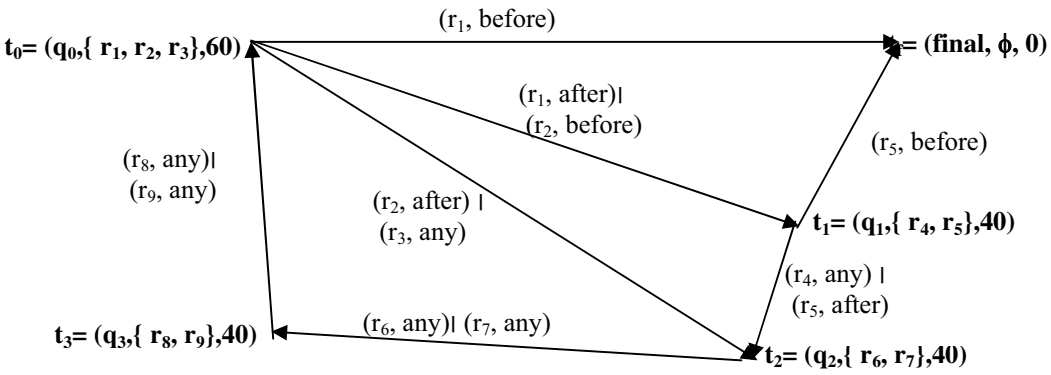


Fig. 2. Graph of tests.

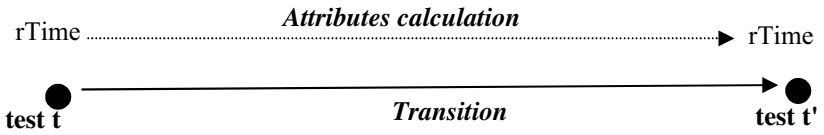


Fig. 3. A transition with attributes calculation.

- t_0.rTime= globalDuration, is the remaining time associated with t_0,
- for a transition δ(t, (r, α)) = t' and λ (t,(r, α))=d, the remaining time at the test t' is given by :

  t'. rTime = t. rTime – d où

  t'. rTime and t. rTime represent, respectively, the remaining time of the tests t and t'.

On the implementation level, the function λ runs as a chronometer. It starts when the system presents a new test to the learner and stops or determines the value *d* when the learner sends the answer *r* to the test *t*.

Note that this formalism is rather general, and that one can set a duration for each test and a total duration for the entire questionnaire at the same time. The following example illustrates the concept behind the formalism.

Example: Let us consider the same evaluation session as in section 3. Suppose that the total duration for the entire questionnaire is 300. That will be the initial value for the attribute d of the test t_0. The values of the function λ are $\lambda(t_0,(r_2,before)) = 50$, $\lambda(t_1,(r_4,before)) = 35$, $\lambda(t_2,(r_7,after)) = 50$, $\lambda(t_3,(r_8,before)) = 35$ and $\lambda(t_0,(r_1,before)) = 50$. Following the same path, the calculation of the attribute d is given by the table 2.

5. The Implementation

The attributed automata formalism is a model for questionnaires structure and evaluation systems architecture. Therefore, it is translated in two different environments. The attributed automata formalism is translated into a DTD in the XML environment in order to introduce a new questionnaires description language, while the transition function is implemented in the client-server environment in terms of an evaluation system.

First, we want to show the syntax of the *questionnaires description language*. Also, we explain the implementation of the durations in local and global time controls. The automaton attribute d (representing duration for a test) attached to a state can be considered as an attribute of an element "state" when translated into XML [9]. On the other hand, the total duration can be considered as an attribute of the root element "< automata> ". In that way, the XML implementation is reduced to a general DTD that covers both the case of attributing durations to single tests and the case of assigning a total duration for the entire questionnaire. The durations are represented as components of tests, and the temporal operators as parameters in the transition function. The new questionnaires description language, called Automata Markup Language or AML, is given by the following DTD:

```
< !– The automata –>
  < !ELEMENT automata(states, initialState, terminalStates, delta, lambda>
  < !ATTLIST automata globalDuration (#PCDATA)>
  < !ELEMENT states (state+)>
  < !ELEMENT state (question, responses, duration)>
  < !ATTLIST state id ID #IMPLIED>
  < !ATTLIST state rTime (#PCDATA)>
  < !ELEMENT question (#PCDATA)>
  < !ATTLIST question id ID #IMPLIED>
  < !ELEMENT responses (response+)>
  < !ELEMENT response (#PCDATA)>
  < !ATTLIST response id ID #IMPLIED>
  < !ELEMENT duration (#PCDATA)>
  < !ELEMENT refState (EMPTY)>
  < !ATTLIST refstate idState IDREF #IMPLIED>
  < !ELEMENT refResponse (EMPTY)>
  < !ATTLIST refResponse idResponse IDREF #IMPLIED>
< !– The initial state –>
  < !ELEMENT initialState (refState)>
< !– The set of terminal states –>
  < !ELEMENT terminalStates (refState+)>
```

Table 3. Automaton transitions and attributes calculation.

State → **Attribute→** **Input** ↓	t_0 t_0.rTime=300	t_1 t_1.rTime	t_2 t_2.rTime	t_3 t_3.rTime	t_f t_f.rTime
(r_2,before)	t_1 t_1.rTime=300-50=250				
(r_4,before)		t_2 t_2.rTime=250-35=215			
(r_7,after)			t_3 t_3.rTime=215-50=165		
(r_8,before)				t_0 t_0.rTime=165-35=130	
(r_1,before)	t_f t_f.rTime=130-50=80				

< **!– The transition function delta–**>
< !ELEMENT delta (transition+)>
< !ELEMENT transition (orignState, (label+, targetState)*)>
< !ELEMENT orignState (refState)>
< !ELEMENT targetState (refState)>
< !ELEMENT label (refResponse+, temporalOp)>
< !ENTITY % E_temporalOp “before | after | any” >
< !ELEMENT temporalOp (%E_temporalOp;)>
< **!– The output function lambda–**>
< !ELEMENT lambda (l_transition+)>
< !ELEMENT l_transition (originState, (refResponse+,time)*>
< !ELEMENT time (#PCDATA)>

The document corresponding to the automata given in sections 3 and 4, with respect to the DTD, is:

< **!– The automata –**>
< automata globalDuration=”300”>
< **!– The set of states –**>
< states>
< state id=”t0” rTime=””>
< question id=”q1”>
what is the best definition for a pointer?
< /question>
< responses>
< response id=”r1”> a pointer is a variable < /response>
< response id=”r2”> a pointer is an address< /response>
< response id=”r3”> a pointer is a function< /response>
< /responses>
< duration> 60< /duration>
< /state>
...
< state id=”tf” rTime =””> < question> < /question>
< responses> < /responses> < duration> 0< /duration>
< /state>
< /states>
< **!– the walking strategy: initial state, set of terminal states and transition function –**>
< **!– The initial state –**>
< initialState> < refState idState=”t0” /> < /initialState>
< **!– The set of terminal states –**>
< terminalStates> < refState idState=”tf” /> < /terminalStates>
< **!– The function delta –**>
< delta>
< transition> < originState> < refState idState=”t0”/>
< /originState>
< label>
< refResponse idResponse=”r1”/>
< timeOp> before< /timeOp>
< /label>
< targetState> < refState idState=”tf”/> < /targetState>
< /transition>
...
< /delta>
< **!– The function lambda –**>
< lambda> < l_transition> < originState>
< refState idState=”t0”/> < /originState> < /l_transition>
...
< /lambda>
< /automata>

Now, we give the design of an *evaluation system* by implementing the *automata transition function* in the client-server environment. Figure 4 shows the correspondence between the automata formalism and the learner system interaction.

This evaluation system manages an AML document for a questionnaire, extracts the initial test, presents it to the learner, takes the learner’s answers, makes a transition from the current test to a next one depending on the answers and calculates the attribute ’*rTime*’. Then, the evaluation system is composed of two important server modules, called initialization and transitions (see Figure 5).

The ’*initialization*’ module realizes the following operations:

- loads the questionnaire into the server memory,
- set the attribute ’rTime’ of the initial test ’t_0’ to the value of the attribute ’globalDuration’ of the questionnaire,
- presents the initial test ’t_0’ to the learner.

The ’*Transitions*’ module makes the following operations:

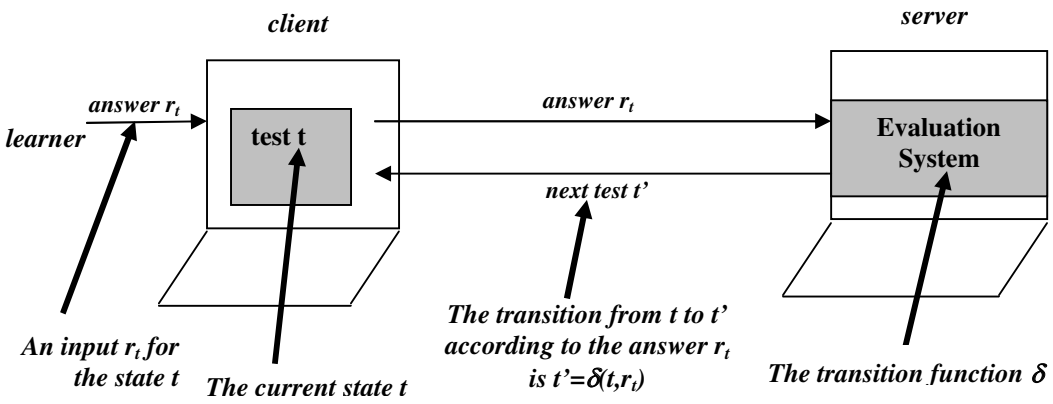


Fig. 4. Correspondence between evaluation system and automata.

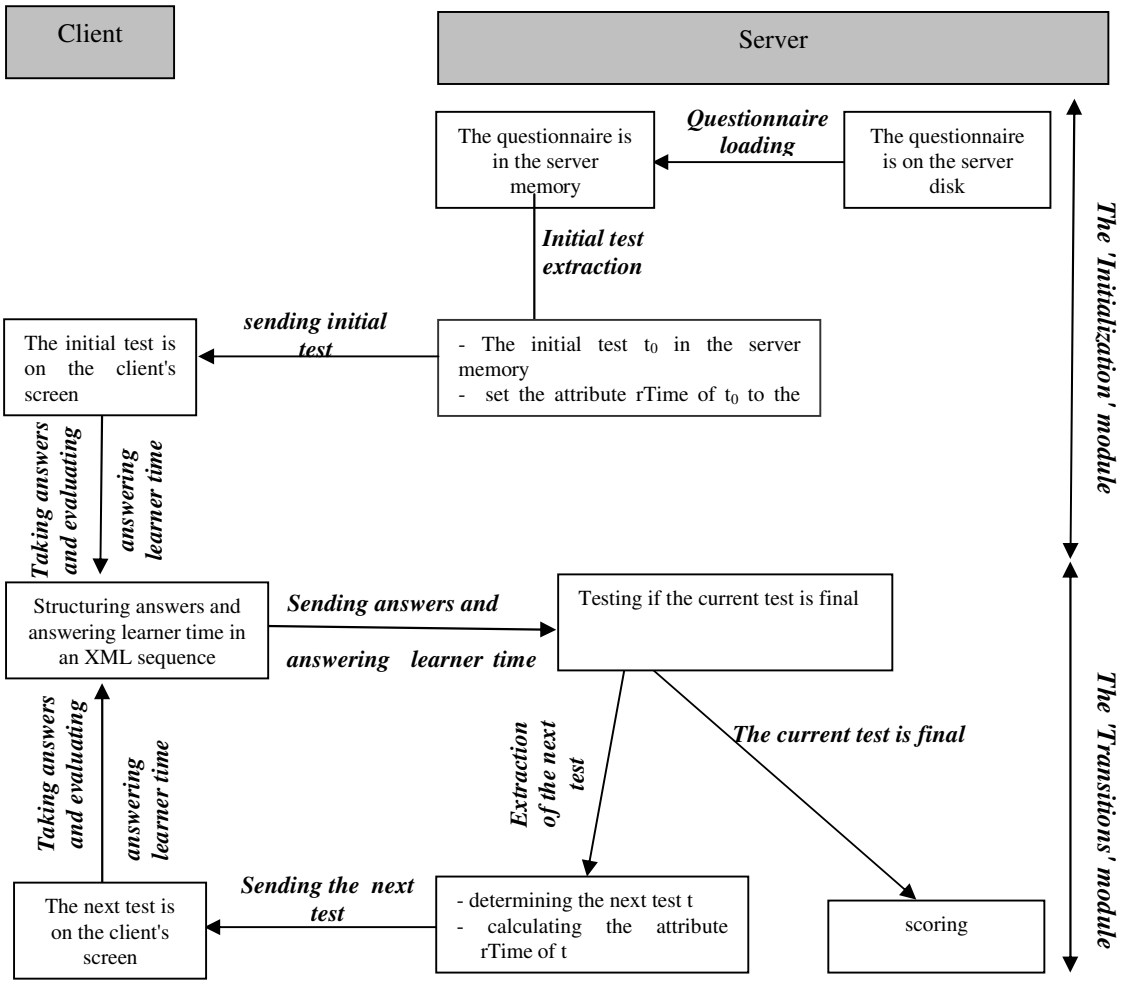


Fig. 5. The evaluation system

- determines the next test according to the current one and the learner answers
- evaluates the attribute 'rTime' of the next test.

We have realized an automaton based evaluation system under the ASP and XML technologies. The two server modules, 'Initialization' and 'Transitions', are written in ASP and VBScript and use the DOM interface for manipulating of the questionnaires documents. The tests presentations are made by applying XSL style sheets to the XML sequences of tests. The generated HTML documents for tests integrate a timing function that realizes the automata output function.

6. Conclusion

In this paper, we have presented an automaton model to describe evaluation process and to integrate time control in a formal way. This formalism is used as a data structure model for questionnaires structure and as an evaluation systems architecture. It can be viewed as a standard for questionnaires structures that is complementary to the IMS standard for tests structure.

We think that automata model can be applied to other modules of an E-learning platform, like LCMS (Learning and Content management system), and then allows conceiving a whole E-learning platform running as an automata transition function.

References

[1] "Hot potatoes tutorial and examples."

[2] Netquiz. http://www.ccdmd.qc.ca/ri/netquiz/.

[3] "Webct."

[4] M. Fejtová, J. Fejt, and L. L. L. Sedlácek, *E-leaning system multiples*. IADIS International Conference e-Society, 2003.

[5] J. Williams, P. Browning, and H. &. M. D. Brickley, "The netquest project: question over the web using tml," *CHEER*, vol. 2, p. 11, 1197.

[6] S. Benadi, J. Y. Ramel, G., and Beuchot, "Using xml for the structuration of e-learning documents," in *The EAEEIE 13th Annual, International Conference on Engineering Education*, (York, Angleterre), pp. 8–10, April 2002.

[7] M. E.-H. Barbar, K. Barbar, Y. Monsef, and I. &. Saleh, *Standard implementation of educational tests in XML*. July 2005.

[8] I. Question and T. I. I. Model. version 2.0 final specification.

[9] M. E.-H. Barbar, K. Barbar, Y. Monsef, and I. &. Saleh, *A three-level model for educational process in a client/server environment*. June 2004.

[10] D. Knuth, "Semantics of context-free languages," *Math. Systems Theory*, vol. 5, pp. 127–145, 1968.

[11] R. Barchino, L. D. M. Henares, and J. M. Guttierrez, "An interoperable assessment language proposal," *ACM SIGCSE Bulletin Archive*, vol. 40, September 2008.

[12] B. Barchino, L. D. Marcos, J. M. Gutiérrez, S. Otón, L. Jiménez, J. A. Gutiérrez, and a. J. J. M. J. R. José Hilera, "Assessment design: A step towards interoperability," *Computer Applications in Engineering Education*, may 2009.

[13] L. Barbosa, H. Rego, T. Moreira, and F. &. G. P. nalvo, "A model for online assessment in adaptive e-learning platform," in *International conference on Multimedia and ICT in Education*, (Lisbon), April 2009.

[14] L. Y. Por and A. B. Zaitun, "An adaptive user assessment model for e-learning," *WSEAS Transactions on Advances in Engineering Education*, 2008.

[15] F. Issac and O. &. Hû, "Un formalisme de description de questionnaires pour l'évaluation," *TICE'2000, Technologie de l'Information et de la Communication dans l'Enseignement Troyes*, pp. 18–20, Octobre 2000.

Mirna El-Hajj Barbar is currently Information Technology Officer in the Information and Communication Technology Division of the Economic and Social Commission for Western Asia - United Nations in Beirut. Ms. Barbar is undertaking research, carrying out analytical studies and performing regional analysis in the area of ICT. She has over 10 years of working experience in ICT. With a PhD in Computer Science from University of Paris 8, she had taught at the Lebanese University and conducted research in the area of e-learning, software engineering, system analysis and design, project management and other fields. Ms. Barbar had served also for many years as ICT expert at the Lebanese Ministry of Social Affairs.

Youssef Monsef, Civil Engineer from Ecole Superieure d'Ingenieurs de Beyrouth (Lebanon), Electronical Engineer from Ecole Superieure Electricite (France), PhD from University Paris-Sud (Orsay), was involved in many simulation projects in Nuclear and Electronical Fields at French Atomic Energy (C.E.A.), Framatome (French-American Atome) and Thomson Companies. He is currently full professor at the Faculty of Engineering of the Lebanese University. His research interests include the Methodology Design in Simulation, Cognitive Man-Machine Interfaces, Computer Aided Instruction on Complex Systems and E-learning.

Kablan Barbar holds Ph.D in Computer Sciences from the University of Bordeaux I. He is a full professor at the Faculty of Sciences of the Lebanese University and director of the Lebanese University's law center. His current research interests include attributed grammars, compiling of markup languages and automatic generation of web applications.

Bilal Chebaro received the PhD degrees from the University of Paul Sabatier in Toulouse, France. He is currently Associate Professor at the Faculty of Sciences of the Lebanese University. His research interests are: image processing, computer vision, multimedia indexing, information retrieval and E-learning.