Proactive e-Learning Management System

Denis Zampuniéris
CICeL – Faculty of Sciences, Technology and Communication
University of Luxembourg
denis.zampunieris@uni.lu

Abstract

This paper introduces a new kind of e-Learning Management System: proactive LMS. These e-learning platforms are designed to improve their users' online interactions by providing appropriate actions initiated by the LMS itself.

1. Introduction

Learning Management Systems or e-learning platforms are dedicated software tools intended to offer a virtual educational and/or training environment online. Although there are already a large number of functions in these environments, current LMS are fundamentally limited tools. Indeed, they are only reactive software developed like classical, user-action oriented software. These tools wait for an instruction and then react to the user request. One could imagine and hope for more help and assistance tools.

We introduce a new kind of Learning Management System: proactive LMS, designed to improve the users' online interactions by providing programmable, automatic and continuous analyses of users (inter-) actions augmented with appropriate actions initiated by the LMS itself.

Proactive systems (see e.g. [1, 2]) adhere to two premises: working on behalf of, or pro, the user, and acting on their own initiative, without user's explicit command. Proactive behaviors are intended to cause changes, rather than just to react to changes.

Our proactive LMS can, for example, automatically and continuously help and take care of e-users with respect to previously defined procedures rules, and even flag other users, like an e-tutor, if something "wrong" is detected in their behaviors; it can also automatically verify that awaited behaviors of e-users have been carried out, and it can react if these actions did not happen.

2. Overview of the proactive components

The proactive system is implemented as a dynamic rules-based system, and is added next to the initial LMS (see Figure 1). They both use the same database as their source of information on the users, their activities, the available resources and the current state of the whole system.

These ideas were first introduced in [5], and then refined in [6]. This paper consolidates and develops these foundations, thanks to experiments feedbacks.

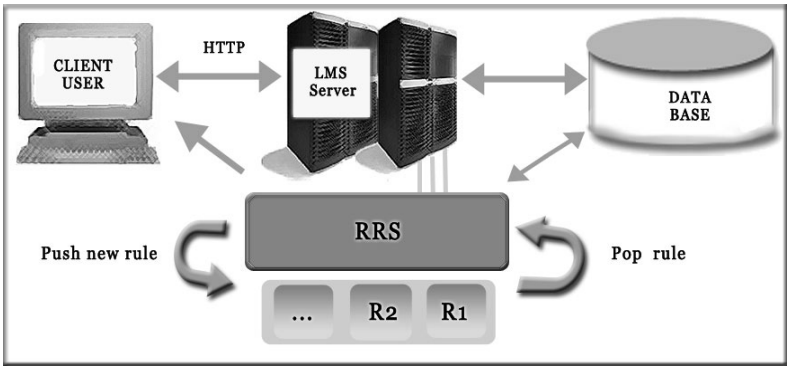

Figure 1. The proactive part of the LMS is implemented by the rules running system (RRS).

The set of rules is stored by the proactive system into a FIFO list. Two parameters influence the behavior of the rules running system (RRS): F = the time frequency of its activation periods, and N = the

Seventh IEEE International Conference on Advanced Learning Technologies (ICALT 2007)
0-7695-2916-X/07 $25.00 © 2007 IEEE

(maximum) number of rules it runs during an activation period. Once activated, the RRS runs the N first rules of the FIFO list (if available), one at a time with respect to their ranks, using the algorithm shown hereunder. A rule is made of five parts: data acquisition, pre-condition, condition, action and rule generation.

```
i. repeat for each data acquisition request DA
     perform DA; if error then raise exception and go to
     step vii; else create new variable initialized to result
ii. create new local Boolean variables "pre-condition"
    and "condition", both initialized to false
iii. repeat for each pre-condition test PC
     evaluate PC; if result == false then go to step vi;
     else if PC == last pre-condition test then set pre-
     condition to true
iv. repeat for each conditions test C
     evaluate C; if result == false then go to step vi;
     else if C == last condition test then condition = true
v. repeat for each action instruction A
     perform A; if error then raise exception on system
     manager console and go to step vii
vi. repeat for each rule generation R
     perform R
vii. delete all local variables
viii. discard rule from the system
```

In our first prototype, we faced an efficiency problem with the continuous evaluation of large sets of rules. Two optimizations were designed with respect to that issue, in order to speed up activations runtimes of the rules running system (RRS): the system implements, on one hand, lazy evaluation (a technique that attempts to delay computation of expressions until the results of the computation are known to be needed [3]) which is an optimization local to the run of one rule, and on the other hand, dynamic programming based on memoization (a technique used primarily to speed up computer programs by storing the results of function calls for later reuse, rather than recomputing them at each invocation of the function [4]) which is an optimization global to all the rules processed in an activation of the RRS.

3. Example of rule for proactive behavior

The idea of the following (very simple) proactive behavior is to give some welcome and recommendation words to an e-student (id = S) registered to an e-course (id = C) under the coaching of an e-tutor (id = T), the first time she/he connects to that e-course. Note that this rule could automatically be added to the system when the student is registered, even if this rule will be activated only weeks later.

```
data acquisition
   es = get_user(S)
   ec = get_course(C)
   et = get_user(T)
pre-condition
   es.isConnectedToCourse(ec) == true
condition ∅
action
   showMessageBox(es.session, "Welcome to the
     course " + ec.name + ", your tutor is " + et.name)
   showMessageBox(es.session, "Do not forget to take
     a look at the forum dedicated to the course " +
     ec.name + " called " + ec.forum)
rule generation
   if (pre-condition == false) then cloneRule(self)
```

4. Conclusion and future work

We introduced a new kind of Learning Management System: proactive LMS, and we showed how to implement it on the basis of a dynamic rules-based software system. Future work includes the design and the implementation of sets of rules (packages) dedicated to common users needs.

5. References

[1] A. Salovaara, and A. Oulasvirta, "Six modes of proactive resource management: A user-centric typology for proactive behaviors", Proc. of the Nordic conference on human-computer interaction, *ACM international conference proceedings series*, 82, 2004, pp. 57-60.

[2] D. Tennenhouse, "Proactive Computing", *Communi-cations of the ACM*, 43 (5), 2000, pp. 43-50.

[3] Wikipedia, http://en.wikipedia.org/wiki/Lazy_evaluation

[4] Wikipedia, http://en.wikipedia.org/wiki/Memoization .

[5] D. Zampuniéris, "Implementation of a Proactive Learning Management System", Proc. of E-Learn World Conference on E-Learning in Corporate, Government, Healthcare & Higher Education, Eds. Th. Reeves & Sh. Yamashita, AACE, October 2006, pp. 3145-3151.

[6] D. Zampuniéris, "Implementation of efficient proactive computing using lazy evaluation in a learning management system", Proc. of m-ICTE International Conference on Multimedia and Information & Communication Technologies in Education, Eds. A. Mendez-Vilas, A. Solano Martin, J. Mesa Gonzales, J. Mesa Gonzalez, FORMATEX, November 2006, pp. 886-890.