Implementation of Efficient Proactive Computing Using Lazy Evaluation in a Learning Management System

Denis Zampunieris, University of Luxembourg, Luxembourg

ABSTRACT

In Zampunieris (2006) we proposed a new kind of learning management system, proactive LMS, designed to help users to better interact online by providing programmable, automatic, and continuous analyses of the users’ actions, augmented with appropriate actions initiated by the LMS itself. The proactive part of our LMS is based on a dynamic rules-based system. However, the main algorithm we proposed in order to implement the rules-running system suffers some efficiency problems. In this article, we propose a new version of the main rules-running algorithm that is based on lazy evaluation in order to avoid unnecessary and time-costly requests to the LMS database when a rule is not activated, that is, when its actions part will not be performed because preliminary check(s) failed.

Keywords: electronic learning (e-learning); proactive computing; technological innovations; Web-based learning systems

INTRODUCTION

Learning management systems (LMSs), or e-learning platforms, are dedicated software tools intended to offer a virtual educational and/or training environment online. Despite a large number of functions covering a large number of user needs for a variety of different users acting in specific roles in these environments, current LMSs are fundamentally limited tools. Indeed, they are only reactive software, developed like classical, user-action-oriented software. These tools wait for an instruction, most likely given through a graphical user interface, and then react to the user request.

Students using these online systems could imagine and hope for more help and assistance tools based on an intelligent analysis of their (lack of) actions. LMSs should tend to offer some personal, immediate, and appropriate support like teachers do in classrooms.

Moreover, some particular users like e-tutors have to peruse lots of data in order to try to efficiently manage specific users’ needs and would expect some highlighting (where to
search and what to look for) from the system instead of a static database.

In Zampunieris (2006) we proposed a new kind of learning management system, proactive LMS, designed to help users better interact online by providing programmable, automatic, and continuous analyses of the users’ interactions, augmented with appropriate actions initiated by the LMS itself.

Proactive systems (see, e.g., Tennenhouse, 2000; Salovaara & Oulasvirta, 2004) adhere to two premises: working on behalf of, or for, the user, and acting on their own initiative without a user’s explicit command. Proactive behaviours are intended to cause changes rather than just react to changes. This is a major change from interactive computing, in which we lock a system into operating at exactly the same frequency as we do.

Our proactive LMS can automatically and continuously take care of e-students with respect to previously defined procedure rules, and even notify an e-tutor if something wrong is detected in some e-learner’s behaviour; it can also automatically check some awaited behaviours of e-students and react if these actions did not happen. Automatic and user-specific checks of generic access conditions (prerequisites) to e-learning modules can be implemented using dynamic rules in the proactive system. Finally, some automatic management processing of the LMS can also be performed by using the proactive part of the system.

The proactive part of our LMS is based on a dynamic rules-based system. However, the algorithm we proposed in order to implement the rules-running system suffers some efficiency problems mainly due to lots of database requests when running the rules, some of them being superfluous.

In this article, we propose a new version of the main rules-running algorithm that is based on lazy evaluation in order to avoid unnecessary and time-costly requests to the LMS database when a rule is not activated, that is, when its actions part will not be performed because preliminary checks failed. In computer programming, lazy evaluation is a technique that attempts to delay the computation of expressions until the results of the computation are known to be needed.

USER INTERFACE

Several recent works also propose to improve current Web-based educational systems by adding intelligence in these systems, but these add-on modules are as static as the initial LMS was. Indeed, they still need a click or an action from the user to activate it. Our goal was to design and develop an LMS that is able to analyze a situation and to act spontaneously with respect to the situation, without queries from its environment.

In our system, the user receives information, help, or hints sent by the proactive system at any time and with no actions needed from him or her. As these messages should not disturb his or her current work (like pop-up windows do, for example), the user interface has been thought of in such a way that the information is viewable at any time and in any context in the LMS in a small screen area.

A message zone has been dedicated in the header (see Figure 1). This alert zone is a Flash application that is able to display the server messages in real time. Messages follow each other vertically and may have different colours according to their importance.

By clicking on a message, the user opens the message manager (see Figure 2) and then can read more details on alerts and can decide to save them or not for later sessions.

DYNAMIC RULES-RUNNING SYSTEM

The set of rules is stored by the proactive system into a FIFO list: The oldest generated rule is at the beginning of the list and will be run first. Two parameters influence the behaviour of the rules-running system: $F$ is the time frequency of its activation periods, and $N$ is the (maximum) number of rules it runs during an activation period. These parameters are set by the system manager and can be changed at run time. The LMS activates (starts) the rules-running system
5 more pages are available in the full version of this document, which may be purchased using the "Add to Cart" button on the product’s webpage:

www.igi-global.com/article/implementatio-nefficien-proactive-computing-using/3004?camid=4v1


www.igi-global.com/e-resources/library-recommendation/?id=2

Related Content

Balancing Tradeoffs in Designing, Deploying, and Authoring Interactive Web-Based Learn-By-Doing Environments
www.igi-global.com/chapter/balancing-tradeoffs-designing-deploying-authoring/41349?camid=4v1a

Animated Pedagogical Agents: The Effect of Visual Information on a Historical Figure Application
R. Heller and M. Procter (2009). International Journal of Web-Based Learning and Teaching Technologies (pp. 54-65).
www.igi-global.com/article/animated-pedagogical-agents/3022?camid=4v1a

Bridging the Gap between Instructional Design and Double-Loop Learning
www.igi-global.com/chapter/bridging-gap-between-instructional-design/40546?camid=4v1a