Usability Metrics in Adaptive Agent-based Tutoring Systems

Víctor López-Jaquero, Francisco Montero,
Antonio Fernández-Caballero, & María D. Lozano

Laboratory of User Interaction and Software Engineering
Computer Science Research Institute
University of Castilla-La Mancha, Albacete (Spain)
{ victor, fmontero, caballer, mlozano }@info-ab.uclm.es

Abstract

Human-computer interaction in traditional application development is focused on the interaction between tasks and a single user interface designed for a single kind of user. A logical evolution should lead interaction to a development model where user skills and preferences are taken into account. In this paper, we introduce preference metrics and performance metrics as parameters that enable user interface adaptation in tutoring systems. The proposal includes a practical example for learning/teaching of an engineering course.

1 Introduction

The ultimate goal for Human-Computer Interaction (HCI) must be the creation of user interfaces based on each individual user preferences (López-Jaquero, Montero, Fernández-Caballero & Lozano, 2003). Those preferences can be captured initially, to a certain extent, in analysis development stages. Using those captured data user profiles can be created in concordance with the identified user stereotypes. However, the user advances in his knowledge, and his preferences change. We need to understand the way the user "uses" the application, and that is where usability metrics can do the job for us.

We introduce in this paper how usability metrics applied to intelligent tutoring systems can lead to the definition of high adaptive learning/teaching systems. These systems will be able to adjust to a specific student and even they will be able to recommend to teachers how to improve the course. Finally, to achieve our goal to create a highly adaptive teaching environment it will take some artificial intelligent techniques that will be modelled by means of multi-agent systems (MAS). An Intelligent Tutoring System is proposed, in which usability metrics are used to achieve intelligent behaviour of the system to improve learning.

2 Usability Metrics

Usability metrics are software quality metrics with a long history of successful application in software engineering (Card &Glass, 1990; Gilb, 1977; Henderson-Sellers, 1996). But, metrics also carry risks (Constantine & Lockwood, 1999). No simple number can completely represent anything as subtle and complex as the usability of a software system, but numbers can sometimes create the illusion of understanding.

Usability metrics have a number of uses, but mostly from the designer's point of view. Metrics for usability can be thought of as falling into three broad categories: preference metrics, which quantify the subjective evaluations and preferences of users, performance metrics, which measure the actual use of working software, and predictive metrics, or design metrics, which assess the quality of designs and prototypes. We shall focus on preference and performance metrics.

One of the most popular ways to assess usability is to use preference metrics. User satisfaction is a component of usability and also an important factor in success in the marketplace. One good example of a standardized set of preference metrics is the Software usability Measurement Inventory (SUMI) developed as part of the ESPRIT project (Porteous, Kirakowski & Corbett, 1994). SUMI is a 50-item questionnaire that includes five subscales measuring different subjective aspects of usability: affect, efficiency, helpfulness, control, and learn ability. Another approach is the Subjective Usability Scales for Software (SUSS) questionnaire, which measures six key elements of user interface designs affecting usability: valence, aesthetics, organization, interpretation, acquisition, and facility. Preference metrics are one of the pillars for user interface customization. However, because of their intrinsic characteristics, they are difficult to assess at run time. There are some preference metrics, such as the manipulation artefact used when commanding tasks (keyboard, menus, and toolbars) that can become especially useful for capturing user preferences.

Performance metrics are indices of various aspects of how users perform during actual or simulated work. Measurement studies form the basis of much traditional research on human factors. User performance is almost always measured by having a group of test users perform a predefined set of test tasks while collecting time and error data (Nielsen, 1993). Typical quantifiable usability measurements include: the time users take to complete a task; the number of tasks of various kinds that can be completed within a given time limit; the ratio between successful interactions and errors; the time spent recovering from errors; the number of user errors; and so on (Nielsen, 1993). Of course, only a subset of these measurements would be collected during any particular study. Performance metrics are especially useful for assessing overall usability. One important point for this kind of metrics is that most of them can be evaluated at run time in a simple manner. Performance metrics are one more input parameter to advance towards user interfaces adapted to the user. Our proposal is to leave behind user interfaces where the user must adapt to a given and fixed interface.

3 An Adaptive Agent-based Tutoring System

The architecture proposed so far is being tested in e-learning system as an Intelligent Tutoring System (ITS) for an Engineering course taught at the Polytechnic Superior School of Albacete, University of Castilla-La Mancha. One of the main goals is that the alumni learn more and better, that is to say, to be able to structure learning matter in such a way to facilitate the learning facilities. One characteristic to take into account in learning is the rhythm the student is able to learn. Thus, an ITS has to adapt rhythm it introduces the concepts to the learning rhythm of each student (for instance, to show more or less exercises, to show more or less tests, etc.). Another aspect widely considered in learning theory is reinforcement by rewarding a correct answer and penalizing the errors (by means of messages, sounds, etc.). Another goal in our environment is to enhance teaching as well as learning. One of the main problems a professor faces when teaching is that he does not know the skills of his alumni. Our proposal leads to conclusions that "teach how to teach".

In our teaching system (see figure 1) there are three multi-agent systems: (1) The *Interaction MAS*, which takes care of the user. It captures user preferences by means of usability metrics to build profiles. The contents shown to the user will be created according to the preferences and skills captured to improve learning experience. (2) The *E-Learning MAS* composes contents for the user. Contents are made of three different parts: theory, exercises and tests. All three parts are composed according to the information captured by the *Interaction MAS*. (3) The *E-Teaching MAS* is one of the most important parts according to our experience. It makes recommendations for improving our day-to-day classes for that course.

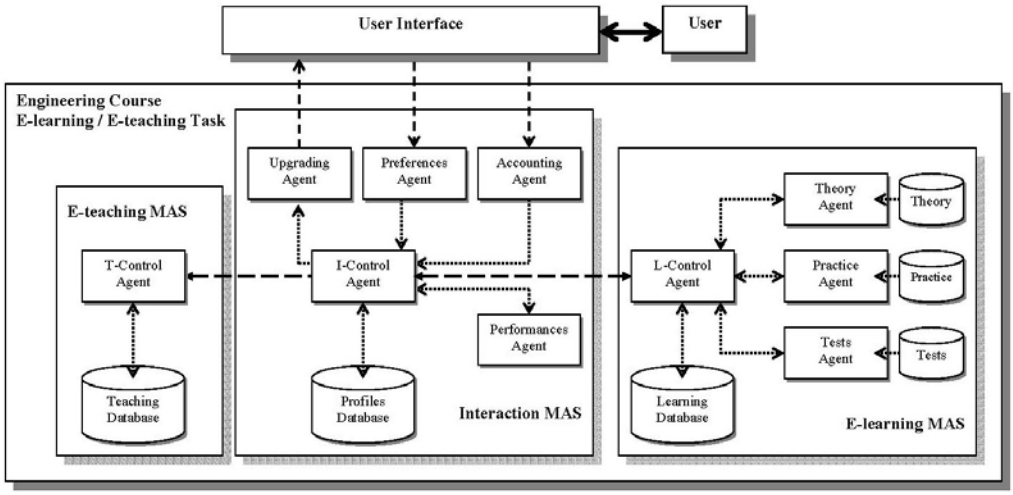

Figure 1: Adaptive agent-based tutoring system architecture.

The user (the student) is in front of the user interface. From the interaction of both entities, modelled by the *Interaction MAS*, different metrics that are stored in a *Profiles Knowledge Database (KDB)* are collected. This database contains the different profiles as a result of the use of the system by different students, with different aptitudes, motivations, etc. The multi-agent system for the learning (*E-Learning MAS*), gets data obtained from the profiles (analysis of the distinct metrics captured) and adequates the contents shown to the concrete student that accesses the Web site. On the other hand, the multi-agent system for teaching (*E-Teaching MAS*) obtains measures that permit to get recommendations to enhance the course. Finally, to offer a good learning of the course, the latter has been decomposed into theory, exercises and tests.

3.1 E-Learning MAS

The *Learning MAS* appears from the general goal to maximize the course learning. The *learning control agent* communicates bi-directionally (asks for and receives information) with the *theory agent*, the *exercises agent*, the *tests agent* and with the *interaction control agent* (*Interaction MAS*). This agent asks for/receives Theory Web Pages to/from the *theory agent*, asks for/receives Exercises Web Pages to/from the *exercises agent*, asks for/receives Tests Web Pages to/from the *tests agent* and communicates (through the *interaction control agent*) with the *performance agent* to record the performance of the student in order to decide if he needs a reinforcement. If the student needs some kind of reinforcement the *learning control agent* will elaborate a plan with the material that has to be shown to the student. In order to determine if the student needs

reinforcement the *performance agent* will have access to a *KDB* where the minimum requisites for each subject are stored (quantity of exercises to be initially shown to the student, how many exercises the student has to answer correctly, and in how much time, maximum time to correctly answer an exercise, etc.).

The *theory agent* is constantly waiting for the *learning control agent* to ask for a Theory Web Page. When this occurs, it looks for the proper theory page and sends it to the *learning control agent*.

The *exercises agent* is autonomous as it controls its proper actions in some degree. The agent by its own means (pro-active) selects the set of exercises to be proposed in the subject studied by the student and adds to each exercise the links to the theory pages that explain the concepts related to the exercise. It sends to the *learning control agent* a Web page containing the exercises proposed.
The *tests agent* is continuously listening to the *learning control agent* until it is asked for Tests Web Pages. The agent by its own means (pro-active) goes on designing a set of tests for the subject the student is engaged in. These tests will be shown to the student in form of a Web.

3.2 E-Teaching MAS

The *Teaching MAS* is the result of the second general goal fixed, namely, to maximize the teaching capacity of the course. The *Teaching MAS* will be collecting the goodness or badness of the parameters defined for the learning system. The *Teaching MAS* is pro-active in the sense that it will be providing recommendations to the teacher on those parameters.

3.3 Interaction MAS

The *Interaction MAS* has been conceived to facilitate the adaptive communication between the system and the user. The *interaction control agent* tells the *upgrading agent* what the user preferences are, as obtained by the *preference agent*, and which has to be the next Web page to be shown (*learning control agent* of *Learning MAS*). When speaking about the preferences of the student, we mean the type of letter, the colour, the icons, etc., the user prefers. The information collected is stored in the *Profiles KDB*. All information concerning time-related parameters and some of the user's behaviours are obtained through the *performance agent* and the *accounting agent*.

The *preference agent* perceives the interaction of the user with the user interface and acts when the user changes his tastes. The *preference agent* is continually running to know the student's preferences at any time.

The *performance agent* calculates the performance metrics when the student leaves the system (at the end of a working session) y goes evaluating everything the student does in order to know if he needs reinforcement. It is autonomous and pro-active; as it may calculate metrics at the same time the student performs other tasks. Some of the metrics the *performance agent* handles are: for each Theory Web Page, the mean time alumni spend there; for each exercise Web page, the mean punctuation obtained by the alumni, as well as the time spent to get the correct answer; for each Tests Web Page, the mean time spent to answer all questions, and the mean punctuation obtained in the tests.

The *accounting agent* perceives the interaction between the student and the user interface and acts (gets information) when the student changes to another Web page, scrolls up and/or down, performs an exercise or a test, etc.

Finally, the *upgrading agent* is constantly waiting for the *interaction control agent* to ask to update the user interface with the new information to be shown to the student (to show another Web page or to show the same Web page but changed to the new tastes of the student).

4 Conclusions

User interface generation has become a software engineering branch of increasing interest. This is probably due to the great amount of money, time and effort spent to develop user interfaces, and the increasing level of exigency of user requirements for usability and accessibility ("W3C", 2002) compliances. Besides it, users engaged in HCI are becoming more and more heterogeneous, and that is a fact we cannot ignore.

In this paper we have proposed an architecture that considers the high diversity of users' skills and preferences: a user-centred and adaptive interaction multi-agent system. This architecture is inspired in usability metrics.

Acknowledgements

This work is supported in part by the Spanish CICYT TIC 2000-1673-C06-06 and CICYT TIC 2000-1106-C02-02 grants.

References

Card, D., & Glass, R. (1990). Measuring Software Design Quality. Prentice-Hall.

Constantine.L.L., & Lockwood L.A.D. (1999). Software for Use: A Practical Guide to the Models and Methods of Usage-Centered Design. Addison-Wesley.

Gilb,T. (1977). Software Metrics. Winthrop Publishers, Inc., Cambridge MA.

Henderson-Sellers, B. (1996). Object-Oriented Metrics: Measures of Complexity. Prentice-Hall.

López-Jaquero, V., Montero, F., Fernández-Caballero, A., & Lozano, M.D. (2003). Towards adaptive user interfaces generation: One step closer to people. Proceedings of International Conference on Enterprise Information Systems 2003.

Nielsen, J. (1993). Usability Engineering. Academic Press.

Porteous, M., Kirakowski, J., & Corbett, M. (1993). SUMI User Handbook. University College Cork, Ireland.

W3C. (2002). http://www.w3.org/WAI/