Position paper

Ontologies for reusable and scrutable student models

Judy Kay

Basser Dept of Computer Science
Madsen F09
University of Sydney
AUSTRALIA 2006

Ph: +61-2-9351-4502
Fax:+61-2-9351-3838
email: judy@cs.usyd.edu.au

ABSTRACT

We are concerned with building reusable, scrutable student models. Reuse means that different teaching systems are expected to make use of some components of the same student model. Scrutability means that the model can be scrutinised by the learner to determine what it models about them and how that modelling process operated.

Reusable student models demand core ontologies that are common to all uses of the components in a domain. On top of this, various teaching systems can each maintain their own additional frameworks and interpretations on these components.

This paper discusses the division of ontological responsibility between the student model and teaching systems.

1. Introduction

We are concerned with defining ontologies for student models so that they will be reusable and scrutable. We now briefly outline our motivation for each of these goals.

Reusable models

The student model is the system’s representation of the student. Since such a model might be used for a teaching system or for other applications, we will use the term user model to mean the system’s set of beliefs about the user’s knowledge, preferences, goals and attributes. This includes the notion of student model. Once we have a user model, it makes sense to reuse it in a range of applications. For example, consider a user model that can be used to customise a newspaper. This might well model the user’s music
preferences so that it can include news about music. Another system might teach music appreciation and this may well need some of the same information about the user’s music preferences.

Reuse of a user model is sensible from the user’s point of view. Suppose, for example, the user spends time interacting with one application, all the while helping it develop a better user model and hence, an improved ability to interact effectively. It would be very reasonable for the user to expect a second application to use relevant aspects of the same user model. This improves the consistency across applications. It also saves the user from the tedium of training new systems.

Reuse of a model of the user is also defensible from the point of view of implementation. It can be costly to construct and maintain user models for user-adapted interaction. This makes it appealing to amortise some of the costs over different systems, so reducing the cost of the user modelling for a particular interactive system.

This user model reuse can be achieved by maintaining a database of user models that can be accessed by various consumers. Equally, it can be achieved by storing the user model in an external form that can be accessed and interpreted by various programs. For a model to be reusable, it needs an agreed ontology and representation so that it can be understood and used by different user model consumer programs. For powerful individualised systems, it is desirable that the model be reused.

**Scrubtable models**

There are many situations where it is desirable that a computer system be scrutable, by which we mean that the system supports the user in scrutinising the system to understand how it operates.

Scrubtability is important in providing user access to personal information. It also makes it possible for the user to correct and control the student model. It increases programmer accountability. Also since teaching systems have explicit and clearly defined teaching processes, it can be argued that these should be available to the student. Finally, the student model has potential as an aid to reflective learning as argued by various researchers such as (Bull, Brna, and Pain, 1995; Self, 1988; Crawford and Kay, 1993; Dillenbourg and Self, 1990).

**1.1 The um representation**

The um scrutable user modelling shell (Kay, 1990, Cook and Kay, 1994, Kay, 1995) supports construction of student models which enable the student to find answers to questions like these:

- What does the system know about me?
- How did it come to draw conclusions about me?
- What is the meaning of the components of the user model?
• How can I control the user model?
For example, Figure 1 shows an example of an overview display of a student model constructed for the domain of sam, a text editor.

The um form of a user model is a directed acyclic graph. However, it is easy to map this to a tree by repeating parts of the model. This simplifies the display of a pleasantly laid out model without crossing lines and since we expect that most models will, in practice, be trees, the limitation should not be a serious problem for accuracy of presentation.

The node shapes are used to mark the type of the component: squares for knowledge components; diamonds for beliefs; crosses for all other attributes; The merit of distinguishing component types is that it helps identify the parts of the model. For example, most Figure 1 models knowledge components, as indicated by the many squares marking components. The belief-components are noticeable because of their diamond
shape. By contrast, the bottom of Figure 1 has several characteristics-components such as the representation of the user’s typing speed (wpm_info). The display also indicates the value of each component of the model. For all but belief-components, true is indicated by black, false by white. The opposite convention applies for beliefs. This convention ensures that a user who is modelled as having perfect knowledge will have all knowledge components displayed in black. The model for a user with completely incorrect knowledge will have all components displayed as white, indicating that they do not know any of the knowledge components and they do hold all the (incorrect) beliefs modelled. Nested shapes are used to indicate the system could not determine the truth of the component.

2. Core student model ontologies

The user model depicted in Figure 1 was part of work that involved modelling of several hundred users over long periods of time. The same model was used in different coaching experiments. One series of coaches taught about the sam text editor (Gheibi and Kay, 1993) and some of the regular expression parts the same user model were used by a Unix coach (Butler, 1992). Both coaches had quite simple teaching strategies.

Part of the support for the scrutability of the model was an explanation subsystem which enabled the user to access a customised explanation of the meanings of the components in the domain. These explanations were customisable depending upon the user’s level of knowledge and their preference for more or less terse explanations (Cook and Kay, 1994).

Essentially, this ensured that the definition of concepts modelled in the ontology were available to the user. Reusability of the user model means that the user model should be seen as associated with the user rather than any single application system. This means that some core elements of the ontology must be defined in the user model. In the case of the sam work, this ontology consisted of the definitions of components, corresponding to the leaf nodes in Figure 1. In addition, it consisted of the partial models which are shown as the inner nodes of the figure. Finally, there were the relationships between these nodes, which in the case of the sam work were simple specialisation links, with each node closer to the leaf being a specialisation of its parent.

Scrutability requires that these meaning should be available to the student in the form of an explanation of the meaning of each component and that the user be able to see the specialisation structuring of the components.

3. Flexible teaching system ontologies

In the um representation, the individual teaching system has responsibility for interpreting the evidence available about each component. This means that the value of a component may differ across teaching systems. For example, in our early work, one application treated a user as knowing a concept once they accessed the explanation of it. Another application only judged a concept as known if the user’s logs indicated a threshold number of successful uses of it.
In addition, we would like to see relativistic ontologies across teaching systems. So, for example, one teaching system may regard a particular concept as correct knowledge while another might treat it as a misconception. We had such a case in our sam work where this approach would have been preferable to the user model defining the concept as correct knowledge or not. This was the representation of the user belief that a good way to quit the sam editor was by killing the sam window. When we initially built the models, we considered this to be a misconception since there are advantages to using the proper quit command. After some period of studying users, we came to appreciate the merit of the kill-to-quit method for many users. We consider that it is desirable to move this distinction of component type to the individual teaching system.

This may mean that different teaching systems may interpret components differently with one seeing a concept x as a common belief of the user and the teaching system; another teaching system may consider that x is untrue and when the user is deemed to belief it, they are regarded as holding a misconception. Essentially, this means that all user models will be overlays; the differential model or the model of user misconceptions will be a matter of interpretation of the core ontology.

Equally, we consider that the um framework could usefully be restricted to separate the definition of the components of the ontology from the relationships between them. These relationships could be the province of the teaching system since it can then control which components it needs and how it will regard them to be related. This, in turn, will influence its reasoning about the components since it might infer that one collection of true components could be used to infer the value of a generalisation concept. Moving this process from the user model to the teaching system improves the reusability and flexibility of the user model.

This view is similar to the growing move to distinguish an object definition and its role, as for example in the work where an object’s behaviour and role are independent (Sasajima, Kitamura, Ikeda, and Mizoguchi, 1995). In our case, the role of a component of the user model may be different in different teaching systems.

4. Discussion

Since we wish to have scrutatable user models, our proposed split of ontological responsibility between the user model and the teaching system means that the user will need to be able to scrutinise both in order to understand the actual meaning of the user model. Indeed, scrutatability of the user model will simply mean access to explanations of the meanings of the concepts modelled.

Other aspects of the ontology will only be available through the additional structuring and interpretation of a teaching system. So, for example, one teaching system might use the specialisation hierarchy shown in Figure 1. This is based upon the the utility of each concept, with the essentials in the group at the top of the figure. Another might use a similar hierarchy but have only a subset of the concepts. Yet another might impose a quite different structure on its set of concepts such as one, for example, based upon the major functional units of an editor with all concept involving movement within a file.
collected together, regardless of whether they are essential or more advanced or esoteric.

To date, our work has involved quite modest ontologies: the sam domain involved just under 100 concepts in the main experiments. We have also worked with the large and flat but still quite simply structured domain of modelling user preferences of movies and their attributes. Even in these, there is considerable scope to experiment with a range of different structurings of the concepts modelled. We are currently beginning work on the very richer domain of teaching a programming language.

References


