Describing and Viewing Large User Models

James Uther
Department of Computer Science
University of Sydney
Sydney, Australia
jimu@cs.usyd.edu.au

Judy Kay
Department of Computer Science
University of Sydney
Sydney, Australia
judy@cs.usyd.edu.au

Abstract

We are developing ways to allow medical students access to user models built on data obtained by online quizzes. A students' individual model can be viewed in a Java applet on a web browser. To provide the model to the the applet, and perhaps to enable exchange of models, we have developed a simple schema for the user model in the Resource Description Format (RDF). The schema and visualisation are described here.

Keywords RDF, User Models, Visualisation

1 The Model

The Graduate Medical Program (GMP) at the University of Sydney uses a problem based curriculum. The lack of formal course outlines in such programs often induces students to over-study. To overcome this the GMP offers an 'online assessment' system in which students can try questions relevant to the course from a bank of around five thousand questions supplied by the course writers [5]. These questions mark the standard of learning required. The students' progress is tracked by the system for feedback to the student, and to question writers.

The model is broken into *learning topics*, the atomic units of the curriculum. In the GMP a learning topic is a unit of study small enough to be summarised with references on two pages. There are approximately 570 learning topics in the first two years of the four year course. Each of these topics has a location (a URL), a title, and is related to other topics by keyword or some other categorisation. Each topic can also have a number of scores associated, calculated from the data collected by the online assessment system. In this trial we included the average mark for the modelled user for each topic, the average mark for each topic of the users' year, and a mark for each topic that the curriculum authors think that the modelled student should be reaching at the time the model is generated.

Proceedings of the Fourth Australasian Document Computing Symposium, Coffs Harbour, Australia, December 3, 1999.

1.1 RDF Representation

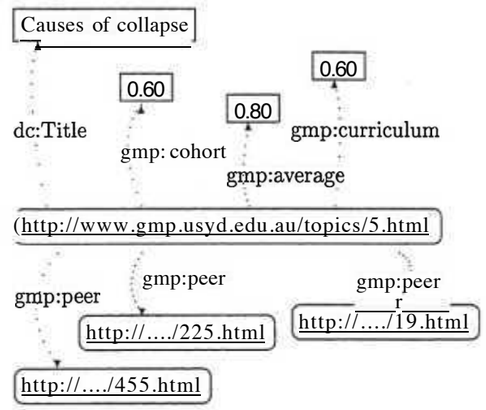

Figure 1: RDF Graph of a model entry.

An RDF [7] graph of a node of the model is shown in Figure 1. The central resource is a learning topic named by its URL. This resource has a title (Causes of collapse), some marks, and some related resources (known as peers in this schema). These peers may then have their own titles, marks, and further peers.

```
<rdf:Description about=
  'http://www.gmp.usyd.edu.au/topics/5.html'>
<dc:Title>Causes of collapse</dc:Title>
<gmp:peer rdf:resource=
  'http://www.gmp.usyd.edu.au/topics/455.html'/>
<gmp:peer rdf:resource=
  'http://www.gmp.usyd.edu.au/topics/225.html'/>
<gmp:peer rdf:resource=
  'http://www.gmp.usyd.edu.au/topics/19.html'/>
<gmp:curriculum>0.60</gmp:curriculum>
<gmp:average>1.0</gmp:average>
<gmp:cohort>1.0</gmp:cohort>
</rdf:Description>
```

Figure 2: A fragment of the XML Serialisation of the RDF Graph in Figure 1, ignoring namespace declarations.

We serialise the graph for the node to XML as in Figure 2. The full model includes relevant XML namespace declarations which are not shown here. Where possible we reuse well known schemas, in particular the Dublin Core [1] for the Title property. The peers are written as multiple properties of the same name. The alternative was to use the *rdf:Bag* construct, but it was felt this would complicate parsing of the file on the client. The scores in the model are named from an internal schema. We are looking at using a more accepted schema that covers user performance like the IMS metadata set [2].

2 Visualisation

The model is described in the RDF schema shown above, but must still be shown to the student in an easily digested form. The visualisation of the model must

- allow the person viewing the model to get a good understanding of the overall state easily.
- allow the person viewing to find detailed information about specific topics.
- allow the student to set a level of performance they are happy with, rather than impose a fixed 'pass' mark.
- allow comparisons between models. For instance a student should be able to find out which on topics they are ahead or behind their year.
- show the relationships between topics.
- allow the viewer to run on a variety of platforms, possibly over modem connections, and on clients with reasonable processors (200Mhz) and XGA (1024x768) screen resolution.
- leave enough space on the screen to display a full web page.

The model viewer is a Java applet placed in one frame of a web page, beside a larger pane that can display a learning topic page, or online assessment questions relating to a learning topic. This leaves us with perhaps 300x550 pixels in which to display the model of 600 items each with additional information and relationships.

Our visualisation can be seen in Figure 3. We show the graph of topics as a vertical list of items reminiscent of DEXTER [9]. The graph is shown as a spanning tree from the selected topic, with depth in the tree represented through brightness, font size, and spacing of the topic titles. The current data set (user average, year average, a comparison, etc) is shown by the hue of the topic titles. Green

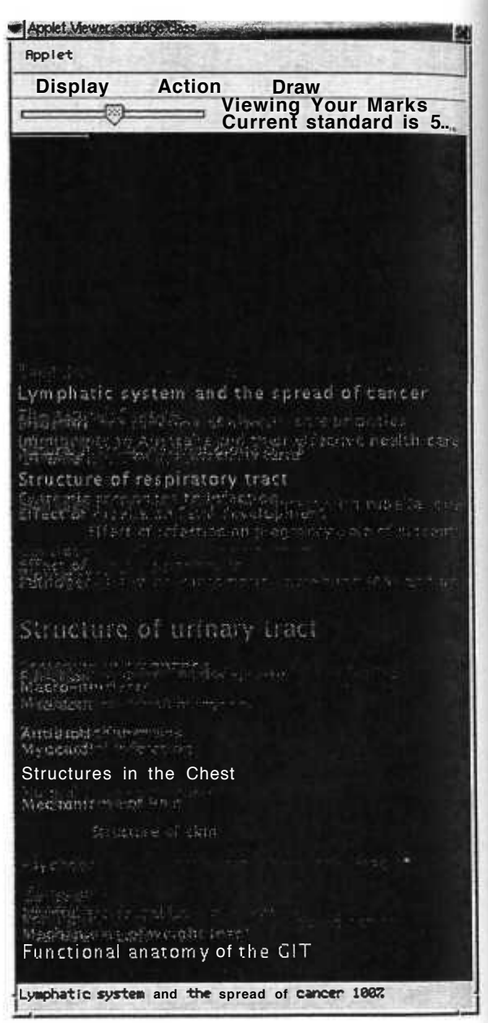

Figure 3: The user model viewer

indicates a higher value than a standard, red a lower, and yellow no information. Degree is shown by colour saturation.

The user interacts with the display by clicking on topic titles, which selects them, calculates a new spanning tree, and updates the display accordingly with an animation. The user may view more detailed information about the selected topic by selecting options from a menu at the top of the viewer, which then requests the appropriate web page and displays it in an adjacent frame of the web browser. Currently the options available are

- view the learning topic itself
- do online assessment questions from that learning topic
- see the evidence for the model on that learning topic

Other menus on the viewer select which data set to show (user average, year average and so forth), and allow some optional interface tweaks to be turned on or off as desired. There is an option to allow the user to drag topics around, thus uncluttering cluttered areas of the display. However, this is initially disabled to avoid confusing new users. Another option (activated in Figure 3) shifts to the right those topics for which there is no current information. This makes it easier to select topics for which there is some information, these being generally more interesting.

2.1 Implementation Experiences

The model RDF file is generated by a small Java program from the online assessment results stored in a RDBMS. This takes several seconds on a lightly loaded machine, making generation on request impractical. Further work on the model storage infrastructure and some optimizations should allow much faster generation.

The client parses the RDF file using a standard XML parser and the SAX parser API. The parser generates objects representing each topic resource. A routine then generates some indexes on the list of topic objects, and *weaves* the topics together by generating references from topic objects to their peers. This weaving process, necessary to be able to find a spanning tree in reasonable time, tends to be the slowest step on average hardware, so a progress indicator is shown.

The visualisation itself is handed a reference to the graph of topics, and spends a small amount of time generating further indexes, mapping topics to screen positions and the like. The animation uses a simple iterative algorithm. First the final position of all topics after the move is calculated. The topics are then all moved half the distance from their current position to the final position and redisplayed. This step repeats six times. Finally all topics are drawn in their final position. The space left around a topic is calculated by taking the inverse of the square of the depth in the tree of the topic, so most space is given to the topic at the top of the tree, and exponentially less space to topics further down the tree.

The implementation has so far proved fast enough to manage the display and animation of seven hundred topics on a modern Java virtual machine and current hardware.

3 Benefits and Drawbacks of RDF

RDF is simply a method of describing internet resources by attaching attributes to a resource by a labeled arc. The resulting directed labelled graph can then be serialised to XML. Most importantly, this XML file can then be restored to exactly the same graph using the published formalism, enabling transfer of a fairly generic representation of the metadata between applications. We have so far only written the visualisation client, but any client that has some knowledge of what our arcs and nodes are (the schema) could consume our file and reason with it [8, 6], or display it differently. There is a proposed way of specifying RDF schemas [4], which we intend to use to formalise our schema when the proposal is accepted as a standard.

Our RDF graph was structured rather simply to avoid difficulty in parsing and building data structures on the client. Recently a number of libraries for handling RDF structures have appeared [10, 11, 3] that try to define generally useful operations on an RDF graph in the same way that the DOM defines generally useful operations on an XML tree. This implementation predates these projects and so has not used them, although we watch the projects with interest. It is also doubtful that these generalised graph structures would be fast enough in their current implementations for our specific purposes, although current work on 'triple databases' within the Mozilla project [3] is interesting.

4 Acknowledgements

Much of this work was supported by the Department of Educational Development and Evaluation, Faculty of Medicine, University of Sydney.

References

[1] Dublin core metadata initiative, http://purl.org/DC.

[2] The ims project, http://www.imsproject.org/.

[3] The mozilla project, http://www.mozilla.org/.

[4] Dan Brickley and R.V. Guha. Resource description framework (rdf) schema specification, http://www.w3.org/TR/PR-rdf-schema/.

[5] Simon Carlile, Stewart Barnet, Ann Sefton, and James Uther. Medical problem based learning supported by intranet technology: a natural student centred approach. *International Journal of Medical Informatics,* 50:225-233, 1998.

[6] Stefan Decker, Dan Brickley, Janne Saarela, and Jürgen Angele. A query and inference service for rdf. In *QL '98 - The Query Languages Workshop,* 1998.