How can users edit and control their models in Ubicomp environments?

Judy Kay, Andrew Lum, James Uther
School of Information Technologies
University of Sydney, Australia 2006
{judy,alum,jimu}@it.usyd.edu.au

Abstract

We describe a visualisation for user models that could be applied to ubiquitous computing environments. The visualisation allows users to explore their user model and scrutinise the components in models with hundreds, perhaps even thousands of components.

Keywords scrutability, ubiquitous computing, user modelling, visualisation.

1. Introduction

Ubiquitous computing environments are characterised by the range of sensors, of varying reliability, with information about the user. Typically, sensors have limited computational power and limited and potentially unreliable communication with centralised services. They may also operate in quite diverse ways. This means that information for user modelling in particularly prone to noise, uncertainty and unreliability.

At the same time, there may be similar problems of power, reliability and bandwidth into the devices which can interact with the user in the ubiquitous environment. On the other hand, there may be possibilities for rich interaction using gesture, projected images that the user can interact with, and interactive wall displays of other types. It is more common to consider the need for interaction with restricted devices such as PDAs and phones. When, and if, the user wants to understand the personalisation in such an environment, they will need to be able to scrutinise their user models with the aid of the devices available.

If a user model is very simple and small, there may be many possible ways to provide access to the model and allow the user to alter it. It may be that quite small parts of the user model might be identified as relevant for the action of the personalised part of the ubiquitous environment.

However, if the model is large, it becomes quite challenging to display relevant parts of the model effectively. One appealing approach is to use a visualisation tool so that the user can readily see and interact with a large number of user model elements.

In the next section, we describe one approach to this problem, a visualisation designed to display user models. In the subsequent section, we discuss its applicability for ubiquitous computing environments, both within them and in conjunction with them followed by a discussion and conclusion.

2. VlUM visualisation

Visualisation of Large User Models (VlUM) is a program written to display large user models in web-based systems (Uther, 2001). It was inspired primarily by Murtagh's work on automated storytelling systems (Murtagh, 1996). VlUM is implemented as a Java Applet, which occupies a frame on one side of the browser window. It utilises perspective distortion to allow users to navigate the user model. Users can see the whole user model at once. Colour is used to show the values of the components of the user model. The user can see the structure of the model because it is reflected in the font size, spacing and brightness of the components.

VlUM also facilitates the viewing of multiple datasets. The visualisation enables a user to see their user model in absolute terms or in terms of another standard. It also supports visualisation of the user model in comparative terms. For example, in the learning domain of its initial development, students could see how their user

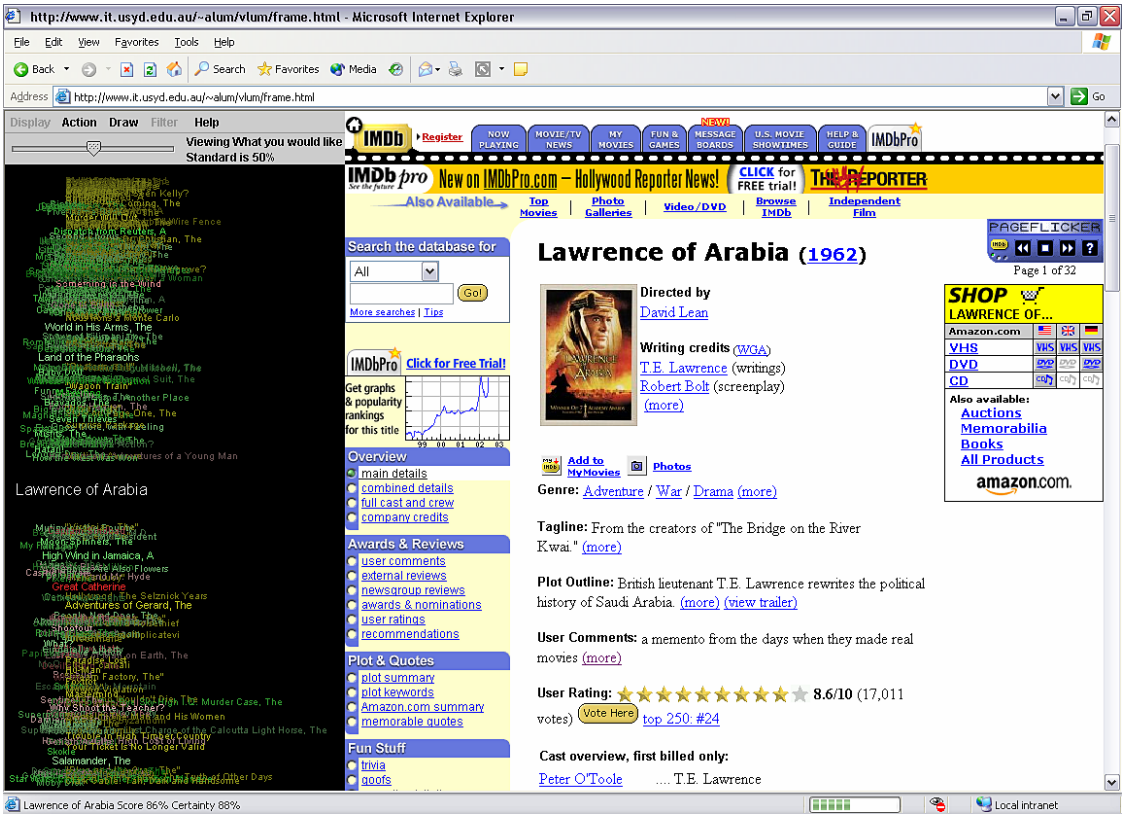

Figure 1: VlUM displaying a user model from a movie recommendation service. There are over 300 terms on the display. *Lawrence of Arabia* is currently selected. The system is fairly certain the user will enjoy this movie.

model compared with that of the teacher's expectations or the whole class.

VlUM user model components have a score and certainty. The score represents a heuristic value for the domain. The certainty value shows the system's confidence that the score is accurate. VlUM uses colour to provide an indication of a component's score through a slider that adjusts the viewing *standard*. Any components with a score less than the standard will have a red hue. Ones greater than the standard will have a green hue. The further a score is away from the standard, the more saturated the colour will become.

Clicking on a component with the mouse will select it and put it into focus, and the display will change so that the selected component will then have the largest font. Other components will have increased sizes the more relevant they are to the selected one. Components that are not relevant are dimmed and shown in a small font.

In Figure 1, we show an example of a VlUM display for a user's movie preferences. Note that the actual display is larger than the one above and the use of colour and animation is quite important for the effectiveness of the user interface that VlUM presents. In this figure, we can see that *Lawrence of Arabia* is currently selected, which is why it is larger than other movie titles and has more space around it. It is the most visible component of the user model. Its most closely related movies are the next most visible, being smaller and having less space than *Lawrence of Arabia*, but larger and with more space than the less closely related movies. These include *The World in His Arms* which is about a quarter of the way from the top of the display, *Land of the Pharaohs* which is a little below it and *The Salamander*, which is quite near the bottom of the display.

VlUM addresses Shneiderman's classic visualisation tasks (Shneiderman, 1996) in the following way:

- **Overview**: the entire user model is shown on the display.
- **Zoom**: users can focus on a particular element in the graph.
- **Filter**: non-relevant items are hidden with a smaller font and darker colour.
- **Details-on-demand**: can find out more about the focused item from the menu.
- **Relate**: related items are shown in a progressively larger font the more relevant they are.
- **History**: users can step back through previously focused items.
- **Extract**: a search function is provided, that puts in focus items matching the search.

VlUM is distinctive in that it was designed explicitly for the purpose of displaying user models and supporting visualisation of a large number of components of a user model. VlUM was extensively evaluated to assess how large a user model could be navigated effectively by users. It proved to perform well with up to the largest data set in the evaluation, consisting of 700 concepts.

Any user modelling representation that can be mapped to a graph has a natural representation in VLUM. This means that it should be able to handle such diverse representations as Bayesian models or simple overlay models. The usability evaluations on VlUM were essentially overlay models of user preferences for movies.

3. Related Work

The VlUM work appears to be the only work to date on providing an overview of substantial sized user models. Moreover, it is among the quite small set of visualizations which has been carefully evaluated to assess whether users can perform the goal tasks and to determine how this performance is affected by the number of elements displayed.

There are tools for visualising user models that could possibly be adapted for use in ubiquitous computing. For example, qv (Kay, 1999) also gives an overview interface for um user models. The interface displays the user model as a tree hierarchy, with the root node on the left of the screen. Subsequent nodes can be expanded or collapsed. The user model represented user knowledge about editors such as EMACS, SAM, and vi. Branches that are not relevant to the user's current level of knowledge are initially collapsed to reduce cognitive overload when they use the interface to scrutinise their user model. Different shapes indicated beliefs (diamonds), knowledge (squares), and non-leaf nodes (circle). The fill colour of the shapes represents component values. Although qv was designed to give an overview of a user model, it uses a very different approach and was explicitly designed to make quite modest numbers of elements visible at any one time. Users can only access the finest details of information by navigating down the hierarchy.

From the perspective of providing a user model overview, there are some very important differences between qv and VlUM. Consider the case where the use is currently focusing on a concept such the modeled preference for the movie qv. Figure 1 shows the VlUM display. The corresponding display for qv would show an hierarchical tree expanded to show this component. This works well where there is a natural hierarchical structure as in the case of knowledge of text editors, the main domain of the qv work. However, it is less suited to a largely non-hierarchical domain such as movie preferences. Even more significant is the differences in the way these overview tools enable the user to see outliers in the user model. In the case of qv, the collapsed parts of the model are displayed with a value that summarises the average of the values under that node. If there are many nodes with just one having a different value, the user would be unable to distinguish this from the case where all the subsumed nodes were of the same value. By contrast, in VlUM, suppose that almost all of a model is one value, say true. Then almost all the components are displayed in green. Even a single false component will stand out in red. In practice, this may be extremely important, especially in the types of context that apply in ubiquitous computing environments.

VISNET (Zapata-Rivera and Greer, 2000, Zapata-Rivera, Neufeld et al., 1999) is a visualisation designed to help people understand Bayesian Belief Networks, which can be naturally represented as network structures. Arrows between nodes represent cause and the nodes are effects. A tree is displayed with distance between nodes indicating the strength of the causal relation, and the nodes size and sizes colour representing the belief values.

4. Applying VlUM in ubiquitous computing environments

Within the current experimental ubiquitous environments, with very restricted I/O devices, the current VlUM would be difficult to use. This still leaves the possibility that the user could interact with the visualisation on a conventional terminal. The possibility of doing this may be sufficient to make the user confident about allowing the model to be used for personalisation within ubiquitous environments.

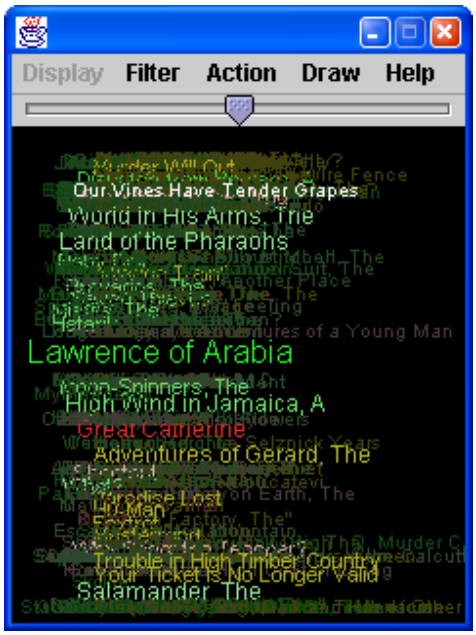

Figure 2: VlUM modified for small displays showing the same model as Figure 1. A depth cut-off value of 2 has been used to reduce the amount of visible text in the visualisation.

Even within the constraints of PDAs and other small devices, it may be possible to provide a visualisation in the near term. We have experimented with rendering the visualisation in smaller resolutions. Figure 2 shows the same applet rendered at 320x240 pixels, which is the similar to many PDA resolutions. A limit to the amount of expansion from a component has been imposed on the spanning algorithm to reduce the number of visible words on the display. The size of the current applet is less than 400k.

5. Discussion and Conclusions

There is a need for users to be able to control their user models. The European Union Data Protection Directive (European Union, 1995) states that a user's personal data should be accessible to them and be correctable if there is incorrect data.

In a ubiquitous computing environment it is especially important that the personal data that constitutes user models should be scrutable (Kay, 1999) since much of the information will be collected invisibly, even surreptitiously. These user models are likely to be very large, perhaps consisting of several sub-models representing different contexts or services. The VlUM tool allows users to see an overview of their complete user model. This is a starting point to navigating through the model, and scrutinising the finer details.

This is especially important when users move into a new environment and context. At that point, the user may become aware of the personalisation if it changes with the new context. This may motivate them to see if the information in their user model is relevant or, perhaps, whether it is correct. The user may also want to check whether aspects of the user model should be available to that part of the environment, before it is released and shared with the environment. Users may also want to propagate information to each other in everyday tasks. For example, users may wish to share task lists with each other to help coordinate work and become more efficient as a group (Schneider, Kortuem et al., 2000).

Previous work on user model visualisation tools has involved conventional interfaces with large screen, keyboard and mouse. We would expect that these will need to be adapted to ubiquitous computing environments. This follows work such as the exploration of adaptation of core interaction objects, such as menus, to smaller displays such as PDAs (Rose, Stegmaiser et al., 2003).

We have described a novel interface that supports visualisation of a user model. This serves as a foundation for users to gain an overview of their user model and to navigate through this so that they can edit and control that user model. Effective user model visualisations give users the understanding to control their user model. Since this model is the core of control of the personalisation, such interfaces are critical to empowering users to control the personalisation of their ubiquitous computing experience.

6. References

European Union, *EU Directive 95/46/EC: Protection of Individuals with Regard to the Processing of Personal Data and on the Free Movement of such Data.* Official Journal of the European Communities. 1995.

Kay, J., *A scrutable user modelling shell for user-adapted interaction*, 1999, University of Sydney.

Murtagh, M., *The Automatist Storytelling System: Putting the Editor's Knowledge in Software*, Masters, 1996, Massachusetts Institute of Technology.

Rose, D., et al. *Non-invasive Adaptation of Black-box User Interfaces*. In: R. Biddle and B. Thomas, Editors. *Fourth Australian User Interface Conference*; 2003.

Schneider, J., et al. *Disseminating Trust Information in Wearable Communities*. In: *2nd International Symposium on Handheld and Ubiquitious Computing*; 2000.

Shneiderman, B. *The eyes have it: A task by data type taxonomy for information visualizations*. In: *1996 IEEE Conference on Visual Languages*; 1996, p. 336-343.

Uther, J., *On the Visualisation of Large User Model in Web Based Systems*, PhD Thesis, 2001, University of Sydney.

Zapata-Rivera, J.D. and Greer, J., *Inspecting and Visualizing Distributed Bayesian Student Models*, in G. Gauthier, C. Frasson, and K. VanLehn, Editors, *Intelligent Tutoring Systems*. 2000. p. 544-553.

Zapata-Rivera, J.D., Neufeld, E., and Greer, J. *Visualization of Bayesian Belief Networks*. In: *IEEE Visualization 1999*; 1999.

Acknowledgements

We would like to thank Hewlett-Packard for funding parts of this research, and Mr. Michael Avery for proof-reading.