A Flexible Architecture for User-Adaptable Visualization

Bastiaan Schönhage
Vrije Universiteit
Dep. of Mathematics and Computer Science
De Boelelaan 1081, 1081 HV Amsterdam
The Netherlands
email: bastiaan@cs.vu.nl

Anton Eliëns
Vrije Universiteit
Dep. of Mathematics and Computer Science
De Boelelaan 1081, 1081 HV Amsterdam
The Netherlands
email: eliens@cs.vu.nl

ASZ Research & Development
Postbus 8300, 1005 CA Amsterdam
The Netherlands
email: schonhage@gak.nl

Abstract

When information visualization is being used to support multiple users, two characteristics are important. First, multiple users with different backgrounds have individual information needs and thus require multiple views on the information. Second, to allow users to experiment with the information and the visualization, the visualization must be adaptable. This paper presents a conceptual framework which decouples the generation and presentation of information by means of an intermediate derived model, allowing users to adapt the visualization to their information needs. Additionally, we will discuss a Web-based software architecture which deploys CORBA to allow for dynamic and interactive visualizations.

Keywords: Software Architecture, Information Visualization, User-Interaction

1 Multi-user visualization: requirements

Visualization, the process of transforming data and information into visual representations, can be deployed for many different purposes. For example, scientific visualization is used to give experts an extra instrument to analyze the enormous amount of information available. Information visualization on the other hand is often used to make complex data and information accessible to non-experts. Additionally, when visualization is combined with simulation, the user may experiment with alternatives to the original situation and evaluate some *what-if* situations with minimal costs.

In this paper, we will look at visualizations to support decision making. For example, a business process re-design or re-engineering (BPR) project is a collective decision making process which can fruitfully be supported by information technology[1]. A BPR project is typically performed by a team of persons with different backgrounds. A team can consist of external BPR-specialists, management of the company and employees who actually do the job. Every individual of the BPR team has his or her own information needs to evaluate the design alternatives. Furthermore, when multiple users have different information requirements, they need a different view (i.e. visualization) of the data. A flexible visualization system must support this by allowing multiple views on the information. For example, the manager will be interested in a more overall view of the process with a focus on the productivity and the quality of service to clients, whereas the representative of the workers will focus on keeping the workload of the employees within limits.

By subjecting different BPR design alternatives to simulation studies of which the results can be displayed using visualization, the participants can increase their insight into the business processes. Additionally, experimenting with alternative situations helps participants understanding the consequences of changes to the business process. Instead of a number of fixed experiments, a user must be able to play with the possible alternatives herself. By making the coupling of generation and visualization of information loosely, we allow for multiple adaptable views on the information.

To stimulate the participants in experimenting with the visualization, and consequently improve the output of the project, it is necessary to have the information including simulation and visualization available at the desktop of the individuals. Furthermore, there is nowadays the preference to integrate the visualization architecture with the World Wide Web. The Web is an excellent medium to provide the context of the visualization consisting of information such as text, pictures, sound and video. By combining the Web and networked visualization we have a general vehicle to support people in making hard decisions.

Concluding, we have three requirements to support multiple users in visualizing information: 1) multiple views or perspectives to support users with different information needs, 2) adaptive visualization to allow for experimentation and 3) a networked or Web-based architecture to support visualization at the user's desktop.

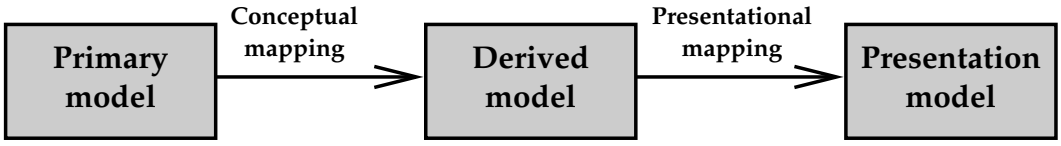

Figure 1: Conceptual architecture

2 The process of visualization: a conceptual architecture

We regard the process of visualization as a transition of data through a sequence of models, starting with the generation of data and ending with the presentation of a visualization. Figure 1 contains three models going from the raw data on the left to a visual representation on the right. The leftmost box is the primary model containing explicitly or implicitly all information that is available. For example, when visualizing simulations, the primary model is the model used by the simulation program. The derived model, represented by the middle box, contains information derived from the primary data. In the transition from the primary to the derived model, information is adapted to the information need of an individual or a group with the same interests. In the case of business process simulations, the manager and the employee can use the same primary model (i.e. simulation) although their derived model might be quite distinct. Finally, the third box contains the presentation model which is used to display the visualization. With 3D visualizations the presentation model is the 3D scene graph. By separating the derived and presentation model, users can share the derived model while their presentation is different. The information content of the visualization is potentially the same although the *style* can differ depending on the user's preferences.

The mapping from the primary to the derived model gives us the flexibility to adapt the data space to our information needs. This is useful because we are only interested in information with some value for our current interest; data only becomes information when it is of use to answer the questions we have. As a consequence, data in the derived model differs from data in the primary model in two ways. First, only data that is useful for the current perspective is selected for the derived model and, second, information derived from primary data is added to the derived model. Consider, for example, the perspectives of the employee and the manager in the BPR project. The employee, interested in his own part of the business process, only needs a small fraction of the available data. The business manager, on the other hand, who is interested in the overall view, wants to visualize statistical information. By allowing the manager to add formulas to the derived model, this statistical information can be calculated.

The mapping from the derived to the presentation model allows us to specify how to present the information available in the derived model. In [2] we have demonstrated that style sheets are a powerful means to specify the presentation of hypermedia and multimedia documents. In line with this approach, we believe that a combination of 1) high-level mappings to application specific visualization primitives and 2) style sheets to fine-tune the visualization are powerful enough for most visualization applications. For example, when visualizing time, we specify in the presentational mapping to use a clock visualization primitive to present the information concept *time*. Whether this clock is analog, digital or even a sand-glass depends on the user's preferences as specified in the style sheet.

3 Diva software architecture

The *distributed visualization architecture* Diva reflects the process of visualization, outlined in the previous section. Diva consists of generic software components that embody the primary, derived and presentation model. These components, written in C++ and Java, can easily be reused to create new visualization applications.

The first component of the architecture, the *generator* represents the primary model and hence generates all data needed for the visualization. In the BPR example, the business process simulation is a generator because it runs the simulation model and generates the source data for the derived model. To allow for an interactive visualization during the simulation run, the generator component must offer an interface to control the generation. This interface allows, for example, to start or stop the simulation or to change some parameters in the simulation model.

The component embodying the derived model must allow for an expressive and adaptive storage of information. As explained in Section 2, the user must be able to adapt the derived model in such a way that it suits their information needs. In our approach we have defined the *shared concept space* which is an information space based on hierarchical conceptual data. The hierarchical nature of the derived model enables the clustering of information in concepts and subconcepts which allows a better separation of useful and superfluous information. The data coming from the generator is stored in the *data properties* of the concepts. Additional (derived) information is added by the *computed properties* which derive new information using a formula and the data available in the shared concept space. At the moment, derived models have to be programmed before they can be employed in a visualization session. However, we have to research the possibilities of dynamically adapting the structure of a shared concept space by user-feedback during a simulation run.

The presentation component, capturing the presentation model, contains a library of visualization primitives (*gadgets*). To specify the outcome of the visualization, concepts in the shared concept space are mapped to these visualization primitives by the user. Additional layout and style preferences can be specified using style sheets. For example, when a participant in a BPR project wants to visualize the time clients have to wait before being served, the user can map the concept containing the information about the waiting queue to a histogram. Additionally, the participant might map the same concept to a different gadget to visualize the length of the queue at a specific time by displaying a row of waiting people. To allow users to change perspective (e.g. from manager to employee perspective), the user can switch to another shared concept space, mapping and style sheet dynamically.

To illustrate the software architecture, let us take a closer look at our BPR example again. Imagine that the project is being done by one manager and two employees who have to evaluate the option that some employees are being replaced by computers. All three persons want to share the generator (i.e. simulation) because they are evaluating the same BPR design alternative. However, the derived model of the manager and the model of the employees are distinct: the manager focuses on the financial pros and cons of the new situation while the employees want to make sure they are not replaced by the computer. The employees who both use the same shared concept space (i.e. derived model) visualize the processes differently according to their personal preferences.

Most Web-based visualizations (see for a number of examples [5]) employ the Virtual Reality Modeling Language (VRML) to display the information. To support interactivity, they employ a combination of hyperlinks, HTTP and cgi-scripts. However, these tools are not powerful enough to support dynamic-updates, user-interaction and user-feedback adequately. Quoting, we state that "... the CGI/HTTP protocol is clumsy, stateless and extremely slow" [4]. This is the main reason we have decided to use the Common Object Request Broker Architecture (CORBA) [4, 7] as the communication architecture. A CORBA-based system consists of distributed objects collected in components, which can be used like local objects, thus allowing powerful communication. The interfaces of the software components are defined using the *interface definition language* IDL and abstract from hardware, operating system and programming language.

To experiment with both the conceptual and the software architecture we have created a CORBA-based prototype. The implementation consists of a business process simulation and a shared concept space, both written in C++, and a presentation component. The prototype allows for simultaneous visualization and simulation of business processes. The user can start or stop the simulator by using a remote control applet. Additionally, he or she is allowed to dynamically change the presentational mapping and style sheet used. Currently, we have two distinct implementations of the presentation component, one Web-browser plug-in based on C++ and the visualization toolkit VTK [6]. The other one is based on VRML 2.0 [3] and Java. Because of the promising features of VRML and Java, such as Web-integration, platform independence and the possibility to create interactive, animated 3D scenes through the *external authoring interface* we will focus on the VRML visualization in the future.

4 Conclusions

This paper presented a framework to support multiple users in visualizing dynamic data. The framework decouples the generation and presentation of information by means of an intermediate derived model, allowing users to adapt the visualization to their information needs. The distributed visualization architecture, which reflects the conceptual framework, consists of CORBA-based components and is the basis for a flexible and interactive visualization application. At the moment, the user can control and change the visualization to a limited extend. However, in the future we will work on user-feedback to allow for more experimentation during a visualization session. As a conclusion, we think that this is a promising step towards a flexible architecture for Web-based information visualization.

References

[1] A. Eliëns, F. Niessink, S.P.C. Schönhage, J.R. van Ossenbruggen, and P. Nash. Support for Business Process Redesign: Simulation, Hypermedia and the Web. In *Euromedia 96: Telematics in a Multimedia Environment, London, United Kingdom*, pages 193–200. The Society for Computer Simulation International, December 1996.

[2] A. Eliëns, J.R. van Ossenbruggen, and S.P.C. Schönhage. Animating the Web — An SGML-based Approach. In Rae Earnshaw and John Vince, editors, *The Internet in 3D — Information, Images and Interaction*. Academic Press, 1997.

[3] International Organization for Standardization. The Virtual Reality Modeling Language, 1997. Draft International Standard ISO/IEC DIS 14772-1.

[4] R. Orfali and D. Harkey. *Client Server Programming with JAVA and CORBA*. John Wiley & Sons, 1997.

[5] R.M. Rohrer and E. Swing. Web-based Information Visualization. *IEEE Computer Graphics & Applications*, 17(4):52–59, 1997.

[6] W. Schroeder, K. Martin, and B. Lorensen. *The Visualization Toolkit: an Object-Oriented Approach to 3D Graphics*. Prentice Hall, 1996.

[7] J. Siegel. *CORBA Fundamentals and Programming*. John Wiley & Sons, 1996.