prefuse: a toolkit for interactive information visualization

Jeffrey Heer
Group for User Interface Research
Computer Science Division
University of California, Berkeley
Berkeley, CA 94720-1776, USA
jheer@cs.berkeley.edu

Stuart K. Card
User Interface Research Group
Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, CA 94301, USA
card@parc.com

James A. Landay
DUB Group
Computer Science & Engineering
University of Washington
Seattle, WA 98195-2350, USA
landay@cs.washington.edu

ABSTRACT
In this demonstration we present prefuse, an extensible user interface toolkit for building interactive information visualization applications, including node-link diagrams, containment diagrams, and visualizations of unstructured (edge-free) data such as scatter plots and timelines. prefuse features a set of high level abstractions for mapping abstract data into visual forms and then manipulating visual data in aggregate, including layout, animation, and distortion routines. The result is a platform for creating scalable, highly-interactive visualizations of large data sets in a modular and principled fashion. We have used prefuse to implement both novel and existing visualizations, validating the toolkit's power and expressiveness.

Keywords: Information visualization, user interfaces, trees, graphs, toolkits, interaction techniques, navigation, Java

INTRODUCTION
To support the design and implementation of novel visualizations, we have built prefuse, an extensible toolkit for crafting interactive information visualization applications. The basic formalism of a graph — a set of entities and relations between them—is used as the toolkit's fundamental data structure, enabling a broad class of visualizations. This comprises node-link diagrams, containment diagrams, and visualizations of unstructured (edge-free) data such as scatter plots and timelines. prefuse introduces abstractions for *filtering* source data into visualized content and using composable *actions* to perform batch processing of this content, allowing developers to manipulate content in aggregate. prefuse also includes a rich library of layout algorithms, navigation and interaction techniques, and integrated search facilities. prefuse is written in Java using the Java2D graphics library.

TOOLKIT ARCHITECTURE
To provide a modular toolkit in a principled manner, prefuse implements existing theoretical frameworks for information visualization [1,2]. These reference models decompose design into a process of representing abstract data, mapping data into an intermediate, visualizable form, and then using these visual items to provide interactive displays (Figure 2).

UIST '04, October 24--27, 2004, Santa Fe, New Mexico, USA
ACM 1-58113-962-4/04/0010

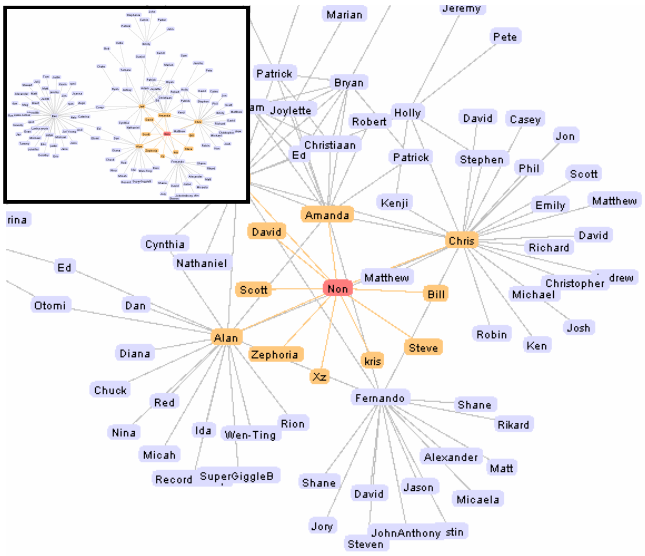

Figure 1. Sample prefuse visualization: A zoomable force-directed layout of an online social network, including an overview display.

Prior work has validated the model's expressiveness, providing a comprehensive taxonomy of visualization techniques [2]. prefuse implements this model by providing abstract data structures, transforming source data into visualizable content through *filtering*, manipulating visual data using composable *actions*, and mapping visual data into interactive views through a configurable rendering system.

Informed by a review of existing applications, our own years of experience designing novel visualizations, and earlier toolkit evaluations, the prefuse toolkit offers:

- Data structures and I/O libraries (including database connectivity) for unstructured, graph, and tree data
- Multiple visualizations and multiple views of data
- Scalability to thousands of on-screen items, and to backing data sources with millions of elements
- Batch processing of data using composable modules
- A library of provided layout and distortion techniques
- Animation and time-based processing
- Graphics transforms, including panning and zooming
- A force simulator for physics-based interfaces
- Interactor components for common interactions
- Integrated color maps and search functionality
- Event logging to support visualization evaluation

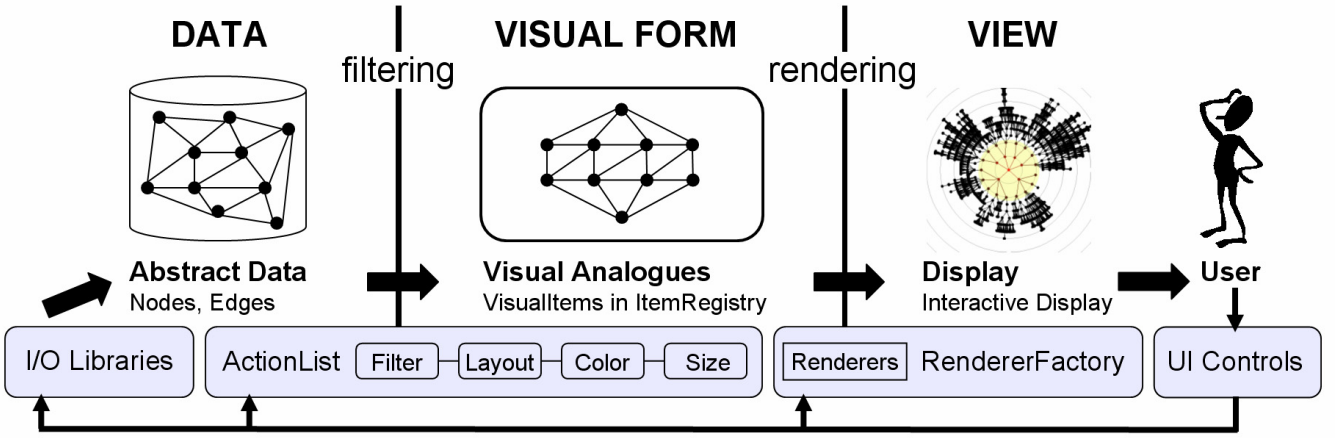

Figure 2. The prefuse visualization framework. Lists of composable actions filter abstract data into visualizable content and assign visual properties (position, color, size, font, *etc*). Renderer modules, provided on a per-item basis by a RendererFactory, draw the VisualItems to construct interactive Displays. User interaction can then trigger changes at any point in the framework.

Figure 2 illustrates the basic data flow of the toolkit. *Filtering* is the process of mapping abstract data to a representation suitable for visualization. First a set of abstract data elements are selected for visualization, such as a depth-limited view of a graph or a bounded range of values to show in a scatter plot. As a result, corresponding visual analogues (called VisualItems) are generated, which record visual properties such as location, color, and size. VisualItems are managed by an ItemRegistry, which oversees caching and garbage collection of items to ensure performance and scalability.

The basic components of application design in prefuse are Actions: composable processing modules that update the VisualItems in an ItemRegistry. While Actions can perform arbitrary processing tasks, most fall into one of three types: filtering data, assigning visual attributes (such as position, color, and size), and interpolating attributes for animation. prefuse includes a number of actions for various filtering schemes, common layout routines, and distortion techniques. These Actions are composed into ActionLists, processing pipelines that sequentially execute contained Actions. The execution of ActionLists is managed by a general activity scheduler, supporting animation and time-based processing.

VisualItems are drawn to the screen by Renderer components provided on a per-item basis by a RendererFactory. This layer of indirection allows for reuse of rendering modules, dynamic change of rendering schemes, and supports advanced features such as semantic zooming. The Renderers draw items onto a Display, a Java Swing user interface component that can be added to a typical Java application. The Display instance also handles user interaction, either by registering pre-built interactors (*e.g.*, for focus selection or dragging items) or by adding user-defined callbacks.

APPLICATIONS

We have used prefuse to build a host of applications, including both novel and traditional visualizations. Familiar visualizations we have implemented include animated radial graph layout, force-directed graph layout (Figure 1), the Data Mountain, fisheye graphs, fisheye menus, TreeMaps, scatterplots, and hyperbolic trees. In addition, the toolkit has supported the addition of new features to these visualizations. For example, panning and zooming can be added to any of these visualizations with 2 lines of code, force-based jitter to prevent occlusion was added to radial layout with 12 lines of code, and an overview display was added to the force-based layout (as in Figure 1) with 5 lines of code.

We have also implemented a number of novel visualizations, including a new version of Degree-of-Interest Trees [3], TimeTrees for viewing hierarchical data over time (*e.g.*, an evolving organization chart), and Vizster [4], a new social network visualization and exploration tool. In our experience, the granularity of reusable Actions and Renderers provided by prefuse has proven quite valuable for promoting code reuse and structured application design while still providing the flexibility needed to explore novel designs.

CONCLUSION

prefuse is part of a larger move to systematize information visualization research [3,6,7] and bring more interactivity into data analysis and exploration problems. In future work, we plan to continue to refine the API in response to user experiences and provide additional library components. Perhaps most importantly, we are continuing to use the toolkit to create novel visualizations. prefuse is open-source software. The toolkit software, as well as video and interactive demonstrations, is available on the web at `http://prefuse.sourceforge.net`.

REFERENCES

1. Card, S.K., Information Visualization, in *The Human-Computer Interaction Handbook*. Lawrence Erlbaum Associates, 2002.
2. Chi, E.H. A Taxonomy of Visualization Techniques Using the Data State Reference Model. *InfoVis '00*. pp. 69-75 2000.
3. Fekete, J.-D. The InfoVis Toolkit, Research Report RR-4818, INRIA Futurs, May 2003. http://www.inria.fr/rrrt/rr-4818.html
4. Heer, J. and S.K. Card. DOITrees Revisited: Scalable, Space-Constrained Visualization of Hierarchical Data. *Advanced Visual Interfaces*. Gallipoli, Italy, May 2004.
5. Heer, J. Vizster: Visualizing Online Social Networks. April 2004. http://www.cs.berkeley.edu/~jheer/infovis/final
6. ILOG Discovery. http://www2.ilog.com/preview/Discovery/
7. Tableau Software. http://www.tableausoftware.com/