Baumgartner, Jason, Börner, Katy, Deckard, Nathan J., Sheth, Nihar. (2003). An XML Toolkit for an Information Visualization Software Repository. Poster Compendium, IEEE Information Visualization Conference, pp. 72-73.

Interactive Poster: An XML Toolkit for an Information Visualization Software Repository

Jason Baumgartner*, Katy Börner, Nathan J. Deckard, Nihar Sheth
Indiana University, SLIS
Bloomington IN 47405, USA
* jlbaumga@indiana.edu

Introduction

In (Baumgartner & Börner, 2002) we motivated the need and introduced the beginnings of a general software repository supporting education and research in information visualization (Börner & Zhou, 2001). This poster describes the general architecture of the XML toolkit and reviews the currently available data analysis, layout and interaction algorithms as well as their interplay. Last but not least it describes how new code can be integrated.

XML Toolkit Architecture

The unified toolkit architecture aims to provide a flexible infrastructure in which multiple data analysis and information visualization (IV) algorithms can be incorporated and combined. This structure allows concurrent visualization and interaction with the same datasets accessed through standard model interfaces. The supported models include the TreeModel, TableModel, and ListModel which are part of the standard Java edition (J2SE) along with the MatrixModel and NetworkModel which are additional interfaces supported in this framework.

A persistence factory is utilized to enable a general and interchangeable layer for persisting and restoring those various data models. The persistence could be to an object database, a flat file, XML datastore, etc.

The implemented persistence layer is an XML-based interchange format that is used to unify data input, interchange, and output formats. The factory and interface classes allow all software packages to implement and to use a defined XML schema set that is hidden away in the persistence layer of the toolkit. This ensures that software packages can be easily interchanged, compared, and combined through the models that are generated instead of an algorithm-by-algorithm direct use of the XML structure. Also simple configurations of the XML input format suffice to use different algorithms in a wide variety of applications as they may produce different model types that are supported by different IV algorithms. Finally, all the Java-based IV algorithms can be run in stand-alone mode as an applet or application.

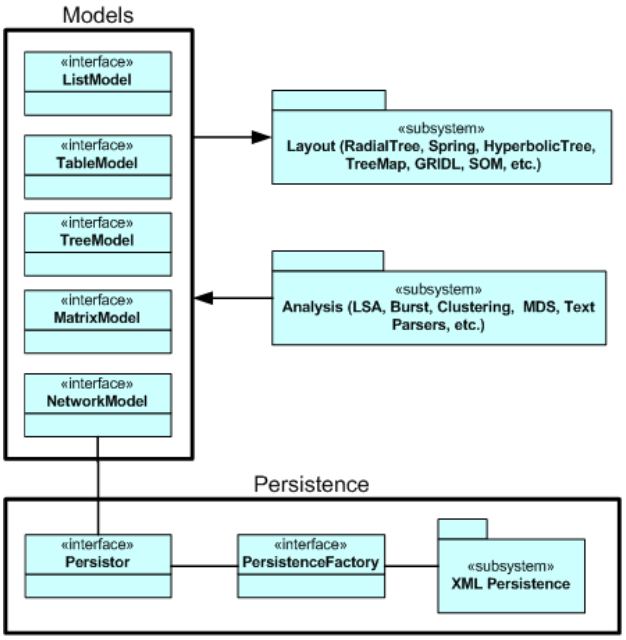

Figure 1: General Architecture of the XML Toolkit

The general structure of the IV repository XML toolkit, depicted in Figure 1, relies on the use of factory and interface classes to interact with various data analysis algorithms and to instantiate and populate the various visualization algorithms. Each algorithm class must implement at least one of the model interfaces for its internal data model in order to be registered with the toolkit. The XML data and the interfaced objects are managed through the persistence layer and the model interfaces that control access of the data and its population into the objects.

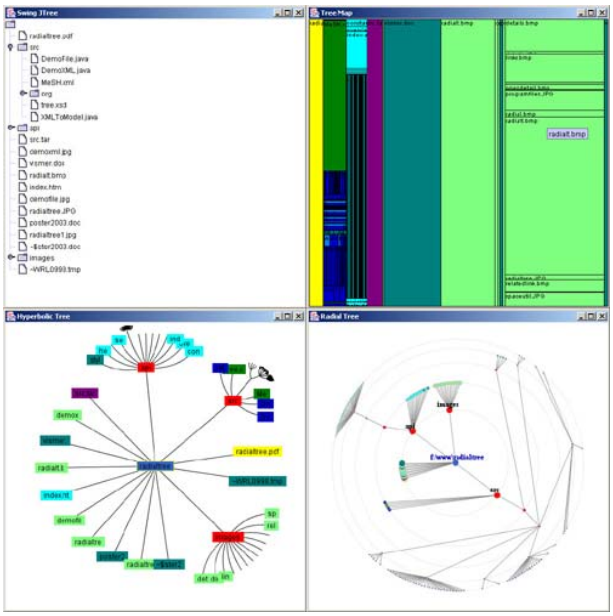

Figure 2: Different visualizations of TreeModel data: Jtree, TreeMap, Hyperbolic Tree, and Radial Tree

Figure 2 shows visualizations generated by algorithms that supporting a TreeModel for their data management.

Integrating New Code

The process to integrate code is to support either building a supported model type of ListModel, TableModel, TreeModel, MatrixModel, or NetworkModel; or to use one or more of these models for an algorithm's data representation. In order to integrate algorithms into the toolkit a code developer would need to either build their code with at least one of these model types or program a wrapper to interchange from their data structure to one of the interfaces.

There is also an interface for processing formatting options called IVFormat which generalizes some of the general node and edge format options (background, foreground, font, size, etc). The toolkit can work without the IVFormat interface as the defaults are all populated for different formatting options.

Outlook

The toolkit is available for non-commercial purposes. The following issues are all under current development or planned for continued development.

- Finalize the design and development of a simple graphical user interface for the application layer of the toolkit to more easily interact with data analysis and IV code pieces.
- Continue to incorporate other algorithms into the toolkit where licensing allows.
- Allow for a dynamic lookup of classes that can be integrated with the toolkit via Java reflection over a working directory and/or a set of jar files.
- Save interaction data such as manipulation changes in the data, the state of the visualization, etc. that could be advantageous to compare visualizations of different data sets among others.
- Metadata schemas, like the Dublin Core and the Resource Description Framework (RDF), will be employed to provide an interoperable way to represent meaning with data. The current schema for the persistence layer provides both a data description and a view description for the various node – link structures. The resource description in the schema defines a tag set as related to the Dublin Core tag set so it can be easily transformed to straight Dublin Core. Therefore the use of RDF and Dublin Core can be interchanged via XSLT transformations to the schema. This will allow the ability to generalize to a RDF / Dublin Core representation and back to the schema of the toolkit. The schema set will be registered with the Open Archives Initiative (OAI) protocol (Lagoze & Sompel, 2001) to allow the greatest interoperability of the data.
- Furthermore, the existing schemas will all be centered on exclusive document description, vector graphic markup, geographical layout, etc. The focus of the initial implementation will be on standard model structures and at this time will not include geographical layout representations found in most geographical information systems (GIS). The versioning of schemas will allow for extension of other schemas, i.e. scalable vector graphics (SVG), and future versions that could directly support items like GIS.
- Provide user documentation, JavaDoc, and a workshop on how to use the toolkit; including how to implement algorithms that work with general data model interfaces.

We hope that the Information Visualization community will adopt this toolkit to create a central data-code repository for IV research and education.

We believe the proposed architecture is flexible enough to facilitate easy sharing, comparison, and evaluation of existing and new IV algorithms. Its widely adoption will help to collectively understand the underlying issues of differing visualizations and to pool together existing and future IV efforts.

Acknowledgements

We are grateful to the students taking the IV class at Indiana University in Spring 2001, 2002, and 2003. They have provided invaluable input into the design and usage of the toolkit.

Todd Holloway, Ketan Mane, Sriram Raghuraman, Nihar Sanghvi, Sidharth Thakur, Yin Wu, Ning Yu, and Hui Zhang contributed to the integration of diverse software packages into the repository.

Ben Shneiderman, Matthew Chalmers, Michael Berry, Jon Kleinberg, Teuvo Kohonen and their respective research groups generously contributed source code to the repository.

References

Baumgartner, J., & Börner, K. (2002). *Towards an XML Toolkit for a Software Repository Supporting Information Visualization Education.* Paper presented at the IEEE Information Visualization Conference, Boston, MA,

Börner, K., & Zhou, Y. (2001, July 25-27). *A Software Repository for Education and Research in Information Visualization.* Paper presented at the Fifth International Conference on Information Visualisation, London, England: IEEE Press, pp. 257-262.

Lagoze, C., & Sompel, H. V. (2001). *The Open Archives Initiative: Building a low-barrier interoperability framework.* Paper presented at the First ACM+IEEE Joint Conference on Digital Libraries, Portland, Oregon, USA: ACM Press.