

# A Universal Visualization Platform

**Georges G. Grinstein**

**Alexander G. Gee**

University of Massachusetts Lowell  
Institute for Visualization and Perception Research  
{grinstein,agee}@cs.uml.edu

## Part I

*What functionality should a general InfoVis infrastructure provide?*

A general information visualization infrastructure needs to support both data analysts and research scientists. The infrastructure needs to support the development of custom applications based on a sound common infrastructure, and customized for specific tasks. The initial environment needs to provide a predefined set of tools from which analysts can use to explore data. The infrastructure should also enable developing, testing and sharing customized and novel tools, both visual and analytical. Furthermore, this infrastructure should also include the capabilities for tuning, customizing or replacing fundamental subcomponents necessary for specialized tasks. Finally, the development of techniques needs to be defined on multiple levels, from novices to experts, and the addition of tools should be simple.

*What do you see as the main technical challenges for creating a central but flexible and universally useful (information) visualization software infrastructure (as opposed to 100 different ones)?*

The idea of defining, developing and distributing a single flexible and universally useful visualization software infrastructure comes down to providing a system that can be used without modifications, but which can be easily modified to include custom tools. The infrastructure needs to be flexible enough to support a variety of applications spanning numerous fields and domains, and able to work on data sets of varying sizes from small statistical samples to extremely large real-time generated databases, and everything else in between.

## Part II

Please describe the (information) visualization software infrastructure you are working on.

### *Project Name and Web Address*

Universal Visualization Platform (UVP)

(No web address is currently available as the licensing terms are still being defined.)

*Core Team Members (Please list in order, Role of Project Member, Full Name, E-mail. eg: Developer, John Doe, jdoe@univ.edu)*

Director, Dr. Georges G. Grinstein, grinstein@cs.uml.edu

Principal Architect, Dr. Alexander G. Gee, agee@cs.uml.edu

Developer, Hongli Li, hli@cs.uml.edu

Developer, Min Yu, my@cs.uml.edu

Developer, Howie Goodell, hgoodell@cs.uml.edu

Developer, Chih-Hung Chiang, cchiang@cs.uml.edu

Developer, Vivek Gupta, vgupta@cs.uml.edu

Developer, Mary Beth Smrtic, msmrtic@cs.uml.edu

Developer, Christine Lawrence, clawrenc@cs.uml.edu

Developer, Jianping Zhou, jzhou@cs.uml.edu

Developer, Liwu Hao, lhao@cs.uml.edu

### *Project Start Date*

September 2000

### *Targeted User Group*

Students and research scientists

### *Supported User Tasks*

The primary task of the UVP is the development and sharing of visualization tools.

### *Major Features of the System Architecture*

The UVP is the implementation of a common framework from which various data visualization and analysis applications can be designed. This framework provides the functionality supporting a common data model, linked collections and user sessions. Also, the framework includes the mechanism by which tools are plugged into the framework; tool code is automatically identified and dynamically loaded. This framework consists of four main components wrapped by a common API.

The UVP provides a common data structure able to handle multiple large data sets. This data structure supports dynamic data arrays via the generation of data modification events. Additionally, these data sets can be linked together during analysis sessions using standard database joins and merge operators providing access to associated data sets and information. Furthermore, trees and graphs are encoded using standard data arrays.

Tools within the platform can be dynamically linked together based on linked collections. Changes and events performed in one tool will update other linked tools; for example, brushing selected records on a scatter plot will be observed in a linked table or radviz plot. Furthermore, the UVP maintains multiple linked collections useful for analyzing independent data arrays within the same analysis session or for comparing similar data sets independently.

The UVP provides for the monitoring of user and system events supporting standard undo / redo actions plus advanced interactions within the user's history graph. User sessions can also be saved either as a snapshot of the current system state or as the complete session history, both supporting restoration and continued analyses. Fully recorded user sessions can be replayed for demonstration, education, and validation, or analyzed and visualized to study analysis procedures and associated steps.

The UVP provides a mechanism by which tools are easily shared and integrated. Tools are dynamically discovered and made available from multiple local directories and remote sites. Custom tools can be easily developed and included with existing tools enhancing proprietary in-house data analysis procedures.

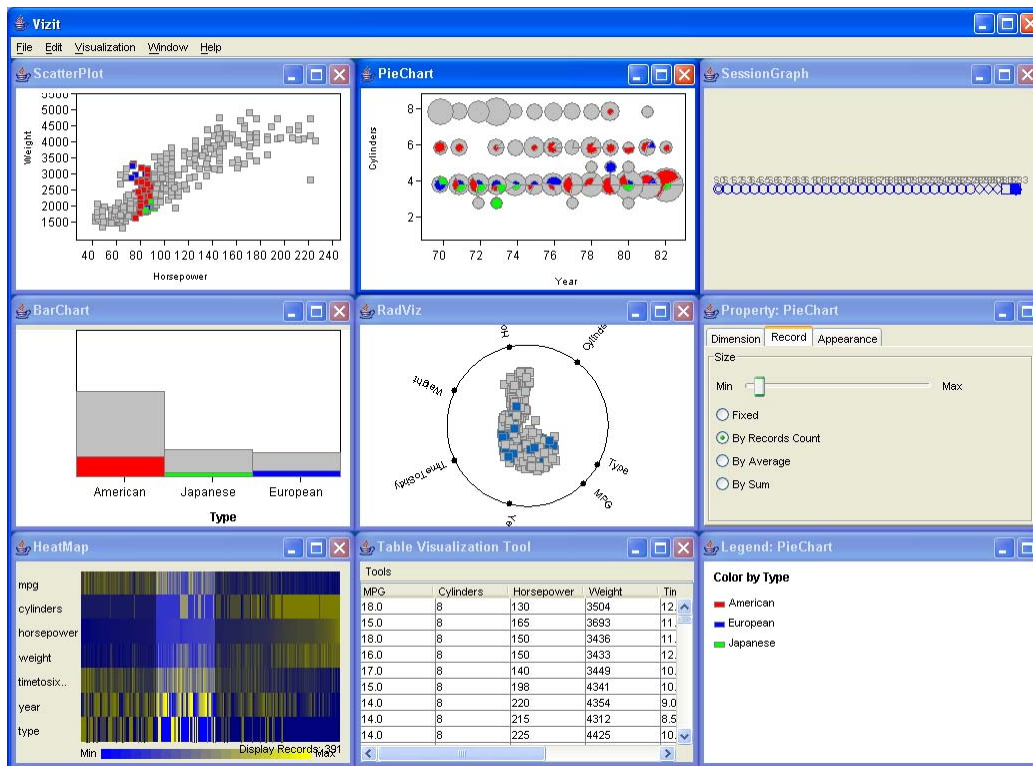
The aforementioned components are further supported by the UVP API from which custom applications can be produced.

*Algorithms Provided*

Visualization techniques: table, bar chart, scatterplot, pie chart, parallel coordinates, and radviz

### *Snapshot of the Interface*

Screen capture of one example application, *Vizit*, used for demonstration purposes. To date, we have four different applications all based on the UVP.



### *Development Platform*

Java

### *Supported Operating Systems*

Windows, Mac OS and Linux

### *Software Dependencies/Required Libraries*

Java 1.4 and higher

### *Current License*

Licensable through the University of Massachusetts Lowell, Office of Research Administration, Office of Research Administration, 600 Suffolk Street, Second Floor South, Lowell, MA 01854.

### *Number of Users/Downloads*

Roughly twenty students and ten clients

### *Pros and Cons*

The focus of the UVP is to provide an environment for exploring data while enabling the addition of custom tools. As a research system for students, the UVP and accompanying applications provide a basis upon which to implement and test research ideas, reducing the time required to provide that first demonstration. Furthermore, using an existing system enables students the potential to compare their ideas with existing techniques, via the supported session recording mechanisms.

The ideas of the UVP are sound and represent a step forward in providing an infrastructure for future work; however, the system is still young and includes a minimal set of tools. In the absence of external influences, the UVP will continue to grow based on client work and student research.

### *Planned Work*

There are a number of aspects still to be considered, designed, implemented, tested, and integrated. First, the initial set of available tools needs to be expanded. Second, the performance of the system needs to be continuously improved. Third, the development and customization of tools needs to be simplified.

The current platform provides a standard set of visualization tools including the most common statistical graphics, scatterplots, parallel coordinates, radviz displays, and heatmaps; and the system also includes a complement of analysis tools. The next step includes increasing the collection of tools with additional visualizations, analysis procedures, mining algorithms, a variety of data import and export tools, and connection to other commonly used third-party software packages. Furthermore, the common components of each tool type needs to be identified and provided as widgets and standard shared components.

Performance tuning to meet the demands of researchers working on large data sets will continue. OpenGL and Agile2D will be considered for inclusion, which has been demonstrated by Fekete to improve rendering performance. Additionally, further work on optimizing the platform's underlying data structure will take place; and the direct integration with third party databases will be analyzed.

The handling of very large data sets can be accomplished using a variety of mechanisms, including caching to disk, in-memory compression techniques and streaming sources. First, pages of records are defined and cached to disk as required to support data spaces too large to fit within system memory. Next, run-length encoding of individual dimensions could boost the number of records loaded in system memory. Last, extremely large data sources could be streamed into the system and displayed appropriately on systems with limited memory and disk space.

The reloading of class code during the run-time operation of the UVP is designed to assist tool developers by providing a mechanism for reloading updated code. In order to maintain system state, a mechanism is required to interactively define tool versions that change during a developmental exploration session.

Finally, the aim is to provide a range of options for developing tools for the platform. Currently tools are created with minimal sharing of common components; and for some developers of tools this is exactly how they like it, working from the ground up with access to all the internal workings of the tool. However, the goal of this platform is to provide support for researchers at every skill level to be able to define and share tools. Therefore, a collection of sharable visualization widgets and tool components will be provided simplifying the development of custom tools. Later, a scripting mechanism for defining tools will be defined, similar to that used by *Processing* by Fry, or a high-level application specification language like UID (User Interface Language) for Motif & X-Windows.

### **Part III**

*Please describe your main interest in participating in the workshop*

Our main interest in participating in this workshop is to exchange experiences developing a visualization and analysis environment. From a research and academic point of view systems need to be designed that support the inclusion of custom algorithms and visual components.

Furthermore, the underlying mechanics of these systems needs to be customizable to work in different domains with their own unique requirements. On the other hand, to be of value these systems need to support licensable technologies and commercializable property. If a single, common infrastructure could be defined to support both research and production then we gain the best of both worlds.

*Determining the feasibility of combining efforts to create one common, shared IV infrastructure as opposed to 100s of underfunded or proprietary toolkits, platforms and frameworks. Scouring for ideas for a common data protocol for communication between plugins. Eliciting feedback about the IVC software architecture with regard to extensibility and ensuring that it is future-proof.*

One major hurdle when attempting to tackle the creation of a single common, shared infrastructure is the underlying programming language. The introduction by Microsoft of the .NET Framework is one possible solution for bypassing the language problem but limiting the software to Windows machines. The UVP was written in Java as a way to be cross-platform compatible. The question then comes down to, can we all agree on one platform or on one language?

Another hurdle to overcome is the underlying infrastructure itself. Can we define an all encompassing environment to support all our needs? Individual systems are generally designed to specific tasks or to provide specialized functionality. Is it possible to define an architecture that is both robust and flexible? Is there a way that a single core can be defined using modules that can be quickly and effortlessly interchanged to provide customization?