The Visual Data Web

Curran Kelleher

Abstract

We investigate possible options in designing the ideal interactive data visualization platform from the ground up. An attempt is made to formulate precise conceptual models for every system component introduced, so as to facilitate implementation at a later date.

There is a common ideal running through the design of three systems built in the past by the IVPR group: The Universal Visualization Platform (UVP), JyVis, and Weave. Here I do not attempt implement *yet another* similar system, but to take a step back and try to identify what these systems are really trying to achieve. Once a specific overarching purpose for the system is identified, we can begin to identify concrete requirements of the system in a precise manner, then design the system. There have always been recurring themes in the development of these three systems whose realization was always slightly out of reach. In this document I seek to analyze and address as many of these goals as possible in a single conceptual system design. The ultimate aim of this work is to eventually bring the capability of interactive data visualization into the hands of the masses, so as to improve day to day life by empowering individuals and communities to make fact-based decisions.

1 The UVP, JyVis and Weave

The Universal Visualization Platform (UVP) resulted from the Ph. D. thesis of Alexander Gee, and was a research test bed for the IVPR group for many years. JyVis was an attempt by me to re-write the UVP code to be more elegant and amenable to easy development of new visualization tools. Weave is a project undertaken to bring the architecture embodied in the UVP and JyVis to the Web, developed in close collaboration with members of the

OpenIndicators Consortium; a group of universities and non-profit planning agencies from across the US.

The Universal Visualization Platform, JyVis and Weave have many elements of system design in common. One is that they were all designed to take as input a *data table*, which we'll define for our purposes as a table with named columns, cell values encoded as strings and no additional constraints. This is exactly what is expressible in a comma separated value or *CSV file* - a text file containing the column names and table contents as strings, with columns in a row separated by commas (or another delimiter) and rows separated by new lines. The CSV file is the basic input data format for all three systems (though Weave does use a database to store the data).

In all three systems, once a data table is loaded, the user is then able to create interactive visualizations of the data. The unit of code for interactive visualizations is called a *visualization tool*, which is (from the user's perspective) an application-internal window which pops up in response to selecting an entry (such as "Scatter Plot", "Bar Chart", etc.) from a menu. A visualization tool typically has an associated user interface for configuration of options such as "which column to use for the X axis?" or "what color map should be used?". These configuration interfaces are coded per tool, and are not automatically generated.

All visualization tools accessing the same underlying data table are all linked together by two things common to all three systems: Brushed selection and probing. *Brushed selection* is the notion of dynamically selecting a subset of records (rows of a data table) to be highlighted synchronously across tools. Brushed selection (or just "selection") is typically made using a mouse drag operation such as defining a selection rectangle or lassoo on the visualization tool. There are two modes to brushed selection: that in which highlighted records change with every new mouse event, and that in which highlighted records change only after the mouse drag is completed. *Probing* is the notion of selecting a small set of records in a temporary manner such that they are highlighted in all visualizations, and detailed information about them is shown in a tool tip or designated area.

Additional dynamic features applied to loaded data tables include the notion of a subset. In Weave and JyVis, the user can make a selection of records, then tell the system to exclude or include only selected records. Also, in Weave, the same color map must be used by all tools which access a given data table, thus in Weave the color map is associated with the open data tablerather than with individual tools as is the case in the UVP and JyVis.

Because the input to these systems are CSV files, the internal data models used to represent the data in the system reflects the notion of a simple table. None of the three systems supports the notion of a foreign key, therefore interactions such as selection and probing are limited to a single table at a time. Also, the idea of type is not highly developed in the data models, all three systems effectively only make a distinction between numeric and string types. In all three systems, string columns are transformed into numeric columns by assigning integers to unique strings in alphabetical order. Since no type information is provided in a CSV file, all three systems try to guess column type based on a simple procedure: attempt to parse all values as numbers; if none fail, the column is numeric, otherwise the column is a string type.

The notion of a visualization tool encapsulates the computation which transforms data tables and visualization parameters into an interactive visual representation. In the UVP, the encapsulation stops there; every visualization tool must implement drawing and mouse interaction code which interacts with the underlying data table, selection and probing. In JyVis, an attempt was made to encapsulate selectable graphical objects linked with data records. Every visualization tool in JyVis is implemented using an API for creating and manipulating "visualization primitives". This visualization primitive API handles all mouse interaction code including selection and probing, leaving the visualization developer to implement only the remaining parts: transforming data into visual representations, and the user interface for manipulating visualization parameters.

Integration with the R statistical package has been a long term goal for all three systems. R scripts can be run in the UVP, but they are restricted to those which create new data tables or new columns meant to be appended to existing data tables. This allows operations like clustering to be performed, but makes a large class of operations infeasible, such as statistical summarization. For example, if a linear regression is computed for a given set of columns, the result is a set of coefficients rather than a data table, which cannot be used effectively by the UVP without having to write special code. JyVis also attempts to integrate with R, but all instances were hard coded. Weave developers created a unified framework for working with R scripts, but still not all results are viewable visually within Weave without extra coding.

In summary The UVP, JyVis and Weave are built using the same high level feature set in mind, namely that of data tables, visualization tools, brushed selection and probing. The purpose of all these tools is to allow the user to visually analyze data. Such an activity invloves many aspects and could be a part of many various tasks in many contexts. For example, the user may be a city analyst observing trends in poverty, a student plotting data for her physics homework, a prospective homeowner analyzing regional demographics, or a news agency plotting recent election results. To design and implement such a general purpose system is a challenging undertaking. To develop such a system to the extent that it becomes widely used and available to average people is even harder, but I argue this is the ultimate goal in developing these systems.

In creating the UVP, JyVis, and Weave, a great number of simplifying assumptions had to be made in order to keep the scope of the projects feasible. Such simplifying assumptions, however, typically lead to comprimises in functionality. My aim in writing this document is to *design* a next generation platform for interactive data visualization, not to implement one. This fact affords comprehensive investigation of each design decision made in the development of the UVP, JyVis and Weave. By systematically tackling each design challenge, from the data model to the interaction architecture, I hope to construct a cohesive set of clear specifications for an interactive data visualization platform that can achieve wide adoption and positively affect the lives of millions.

2 The Purpose and Context of Interactive Data Visualization

Civilization has flourished with the great help of devices for externalization and communication of knowledge, namely written documents. I believe this interlinkage between written documents and civilization has to do with the notion of *memes*, recurring pieces of knowledge, and the *meme pool*, the collective knowledge of humanity. I hypothesize that the more highly developed the meme pool of a given civilization is, the better the civilization is generally able to function. This is because memes are not dead knowledge, but knowledge about how to perform certain tasks to achieve certain results, which transforms into *human capability.* In this way, knowledge is power.

Throughout history, document systems evolved to include numerical figures in the form of tables and plots, but have always been primarily a means of externalizing and communicating knowledge. Once data collections became sufficiently large, visualization is used not only to externalize and communicate knowledge, but also to explore the raw data. Therefore the appropriate realms to conceptually place data visualization platforms in are twofold: as tools for externalization and communication of knowledge, and as tools for data exploration.

Ultimately, through the combined long-term effects of data exploration, knowledge externalization and communication, interactive data visualization is a tool for humanity to extend its *collective* cognitive capabilities to the realm of data through visualization. Therefore, I argue, *collaboration between people* is really at the core of what data visualization is about, and should be at the heart of the design of any visualization system.

The Internet and World Wide Web have been cataclysmic breakthroughs which have effectively allowed humanity to liberate it's meme pool in the form of Web documents. As time passes, more and more functionality which used to be exclusively avaible to desktop applications is being ported to the Web. Also as time passes, more and more aspects of people's daily lives are integrating with the Web. Just in the past few years, in-browser graphics performance has skyrocketed, giving hope for in-browser interactive visualization. In designing any future data visualization platform, one must carefully consider the decision to create a desktop application or a Web application. Also, given that the natural place for visualizations to be situated is within documents, and the Web seems to be the best place for documents to be, the Web seems a natural habitat for a collaborative data visualization platform.

Data itself has been investigated thoroughly. Database technology has matured to the point where anyone with adequate training can construct a relational database. XML has evolved as a data interchange framework. Recently, the Semantic Web has emerged as a unifying framework for the world's data , just as the Web emerged as a unifying framework for the world's documents. Databases are intentionally limited in scope to locally defined data, and do not have an international focus as the Semantic Web does. XML documents require parsing with knowledge of their schema. Only the Semantic Web is intended to deliver a general-purpose data modeling and interchange framework. As such it has in recent years received tremendous attention, and numerous public databases have been made accessible through Semantic Web technologies.

In most data visualization systems, the responsibility of acquiring data in the proper form has been left entirely to the user of the system. Some say an average of 80 percent of the time on a visualization project is spent aquiring, curating and preprocessing data. The effort involved in data preparation is typically beyond what average people are willing to put in. This is a major factor limiting the adoption of data visualization by the masses, and must be addressed in the design of future systems.

Once the prepared data is visualized, instantly a perennial set of problems arises. How does the user of the visualization system know what the data means? Where did the data come from? Who published it? What does a given column represent? What unit is a given column using? Exactly which object does a given row represent? Which objects are parents and children in hierarchical relationships? Precise documentation, or *metadata*, describing the meaning of the data is needed, yet in practice is rarely represented explicitly. This acute need must be addressed in order to achieve a visualization system in which it is *always clear* what data is being visualized.

Metadata challenges also force one to consider more deeply what the *structure* of the data itself really is. Data is much richer than CSV files, and this richness must be expressed in order to navigate the data. For exanple, hierarchical relationships are present in all data sets attributing numbers to regions of time or space. Classification taxonomies are often the primary structure around which data is built. Furthermore, data may represent the *intersection* of many taxonomies simultaneously (aggregated spatiotemporal data for example).

The cases listed above beg the question "what *is* the structure of data?". I believe the answer can be found in two data models: the entity-relationship model (ER) and the multidimensional () model, also called the OLAP (On-Line Analytical Processing) *data cube* model. In the ER model, classes of individual things, termed *entity sets*, are defined along with *relationships* between them. In the data cube model, data is understood as numeric values for *measures* (e.g. population or average income) associated with region hierarchies (of time, space or taxonomy for example) called *dimensions*. I believe that the contents of most relevant CSV files can be precisely modeled using wither the ER or data cube models.

Though the ER and data cube models can adequately model data within a single computer or data warehouse, they do not provide metadata support. What is needed to accomplish the usability goals for a next generation visualization platform is an internationally oriented framework for publishing ER data sets and data cubesas well as structural and descriptive metadata for each. Such metadata will enable richer interactions as well, such as navigating multidimensional hierarchies and traversing between tables in relational databases.

3 Requirements

References