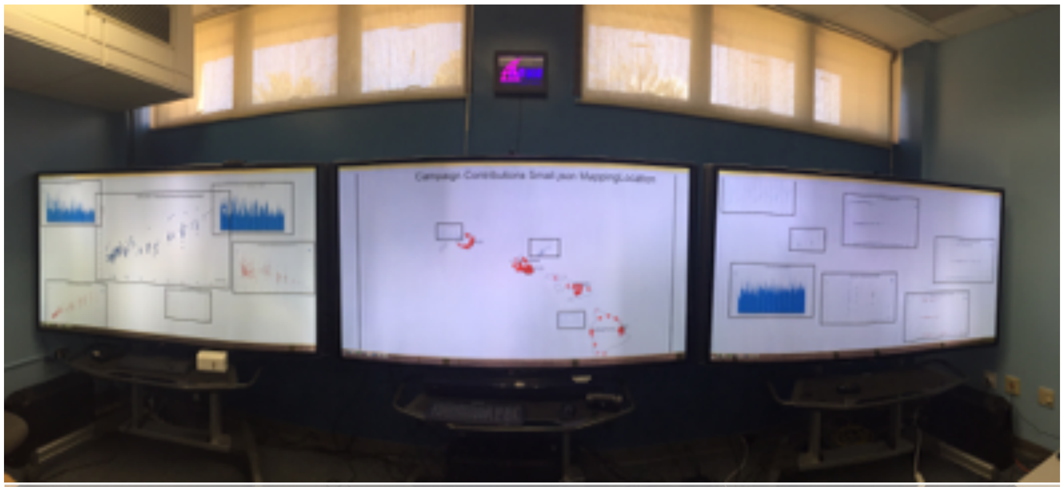
Developing an Interactive Visualization DashBoard Framework for Large-Multi-Display Environments



Alberto Gonzalez Martinez

Advisor: Jason Leigh

MS. I.C.S. at the University of Hawaiʻi at Mānoa

Spring 2015

1. Introduction

With the new advent of the “Big data Era” ·, companies and researchers are investing more effort than ever before in trying to gain knowledge from the ocean of data that is available to them. Machine learning and statistical techniques can be applied to a certain degree in order to extract the important features from data and to try to obtain some insight. Since humans are still better than machines at depicting visual patterns, visualization dashboards for data visualization tools are gaining popularity with new systems appearing every year.

The advanced state of the art of todays display technology with high-pixel densities available at reduced priced allows small teams and researchers to construct war rooms environments with large displays where ideas can be explored creatively and more data can be visualized simultaneously.

The majority of the dashboards for data visualization are not designed with this type of environment in mind. We advocate that the new dashboards for data visualization should be designed focusing on these type of environments, where multiple visualization could be rendered, arranged or compared at the same time by a team of people.

We define a large display environment as a system with one or multiple high definition displays whose sizes are large enough to allow working in parallel and visualizing more content than typically sized displays.

The framework that will be described in the following sections has been designed specifically focused on exploiting the space and high-pixel density available in large display environments. The framework is also designed to support what is termed “fluid interaction”[4] and does so, by providing tools to address the following challenges:

- How to make better use of the available large scale screen space.
- How to aid the task of creating new visualizations.
- How to keep track of the data in multiple visualizations.
- How to rearrange, resize and reposition the elements in the space.
- How to connect different databases.
- How to engage in collaborative work.
- How to reduce complexity of the operations.
- How to keep track of the visualization history.

2. Related Work

Design Interaction Patterns (table 1)

	Use of Space	Derivable Visualizatons	Group & Visualization linking	Visualization Comparison	Anotate Visualizations	Construct Complex Queries Within Visualizations	Interactive Histories	Conects to different Db	Emphasizes in Collaborative Analisis
Information Visualization on Large, High-resolution displays: Issues, Challenges and Opportunities. [1]	X								
Space to think: Large High Resolution displays for Sense-making. [2]	X		X				X		X
Panoramic Data: Data Analysis through pen and touch. [3]		X	X	X	X	X	X		
Fluid interaction for information Visualization. [4]									
GraphTrail: Analyzing Large Multivariate, Heterogeneous Networks While Supporting Exploration History. [5]		X		X					
Show Me: Automatic Presentation for Visual Analysis. [6]		X							
Interaction Support for Visual Comparison Inspired by Natural Behavior. [7]		X		X		X		X	
ExPlates: Spatializing Interactive Analysis to Scaffold Visual Exploration. [8]					X	X	X		
FromDaDy: Spreading Aircraft Trajectories Across Views to Support Iterative Queries. [9]		X				X	X		
Supporting the Cyber Analytic Process Using Visual History on Large Displays. [10]						X	X		X

	Use of Space	Derivable Visualizatons	Group & Visualization linking	Visualization Comparison	Anotate Visualizations	Construct Complex Queries Within Visualizations	Interactive Histories	Conects to different Db	Emphasizes in Collaborative Analisis
Rolling the Dice: Multidimensional visual exploration using scatterplot matrix navigation. [11]		x							
The FlowVizMenu and Parallel Scatterplot Matrix: Hybrid Multidimensional Visualizations for Network Exploration [12]		x							
T Visual comparison for information visualization. [13]				x				x	
DataMeadow: A Visual Canvas for Analysis of Large-Scale Multivariate Data. [14]							x		x
Exploiting History to Reduce Interaction Costs in Collaborative Analysis. [15]					x		x		x
An Evaluation of How Small User Interface Changes Can Improve Scientists' Analytic Strategies. [16]	x								
Graphical Histories for Visualization: Supporting Analysis, Communication, and Evaluation. [17]							x		
Our Data Visualization Framework	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**	**x**

There has been a lot of interest and research in the community to address the different aspects of developing interactive dashboards for data visualization. When coming up with the different interactions depicted in table1, we focused our efforts in understanding what are some of the more general design interaction patterns, that could benefit from the use of a large-multi-display environment, rather than looking into other more specific approaches that instead could only be applied to certain visualization types.

Above all, with the visualization framework that is going to be described in the following sections we aim to achieve what is defined as a “fluid interaction”, following their guidelines along with some others extracted from a review of the up-to-date literature. The rest of this section will focus on analyzing this interactions and explaining why, specifically we think these are suitable for developing visualization dashboards for a large-screen environment.

2.1 Use of Space

The resulting extra space and pixel capacity of these large-displays with higher-resolutions offer new possibilities and challenges, when used for data visualization. Todays applications do not scale well directly to this big displays sizes and do not make proper use of the high pixel density available.

[1] Advocates that large screen environments allow for a better proprioception and embodiment of the space and the data, aiding users in subconsciously using spatial memory, allowing, as a result, to perform rapid, natural, eye-head movement interactions.

Rather than overwhelming the users with to much information, the large available space instead assists users to organize concepts customized to their own preferences, while allowing to display more data simultaneously. Reducing the need for aggregation an the loss of information. In order to use the additional pixels to create more insightful visualizations, that where not possible in single-monitor environments, multiple views of the data should be allowed to show simultaneously in space, allowing users to work, at the same time, at different levels of abstraction, with different levels of detail and representations [1].

In [2] the author suggests that working in free unstructured environments, where users treat document windows as actual documents that they can rearrange using different organizational strategies, allows users to give a personal meaning to the Space. This unstructured space can be used as a *“flexible semantic layer”*, allowing to freely create different representations which add distinct meanings to the visualized data, by for example leading to the creation and formation of structures like clustering or timelines, constructed over an unstructured space.

There are some characteristics of this large screens that are challenging to address with respect to the big space available. Some interaction patterns such as brushing and linking are difficult to achieve with elements that are separated in the screen, but are needed in order to keep the context of the different exploratory paths. Different approaches have been applied successfully, in this direction, like applying some temporary flashing or motion to the selected data elements.

[16] mentions three interactions that would be suitable to have in a framework with these characteristics, although their paper isn't framed at large screens environments, we think they would perfectly suit the needs of multi-large display visualization framework

> Go-to: Centers the camera and zooms in into a selected view.

> Bring-in: Brings a selected view to the center of the display.

> Zoom-out: Centers the camera and zooms out to a position where all views are visible.

Even though this is our main design patter, multiple research questions are yet to be answered about the use of space in this large displays environments. How to understand the effectiveness of different scalable and not-scalable encodings when used with different sizes or at different distances from the screen? Does this mentioned physical navigation provide any advantages over current navigation techniques like zooming, panning or focus plus context? How these current navigations techniques should behave in these large displays environments? How these visualizations should be modified when the user is physically moving around the display? Should we show more elements when the user is close and less when the user is further away?[1]. Nevertheless we hope to help to answer some of this questions through the research and development of this framework in the following years.

2.2 Creating New Graphs

Creating new visualizations is probably the most common task of a dashboard visualization framework, in order to achieve a fluid interaction, a dashboard visualization framework should allow for rapid graph prototyping and creation.

In [3] [5] [7] [9] the idea of derivable visualizations is explored. Creating new visualizations this way allows users to explore other what-if scenarios by reusing or referencing some parts of existing visualizations. Derived visualizations usually are added without eliminating the original visualization keeping the original data intact, nevertheless in [9] a different approach is taken, whenever the user derives some part of a visualization, the derived elements get eliminated from the original view. This is a double operation, it derives a new visualization and at the same time applies a filter operation to the original visualization. Creating visualizations this way, deriving them from another visualization directly, could make creating new visualizations easier and faster to users, by reusing some parameters rather than creating all of them from scratch, with the added benefit of doing so without losing the global picture [3]. New derivable visualizations should maintain a visual consistency (e.g. same color or size) between data attributes that are shared between visualizations in order to keep track of the global context [3]. When new visualizations are derived, we should mark where the data for this new derived visualization is coming from in the original visualization to keep the global context, in [7] a border around the original data is shown when the new derived visualization is selected. New derived visualizations should change only one dimension at a time keeping the rest of the dimensions fixed in space, while the new added dimension is animated to let the users follow the new positions of the data points along the new dimension [9][11].

In order to aid users to select the proper graph type when creating new visualizations, using similar approaches as in [3] and [6], the type of new created visualizations should be, at least, modeled on simple heuristics, based on the type of dimensions of the data or the amount of dimensions to render. On [9], graph type selection is simplified. It is achieved by just connecting the different data dimensions with their visual variables, like color, size to the x or y Axis. On [3] they instead allow rapid graph creation by drag-and-dropping elements anywhere in the screen. Another useful strategy is to show users a preview of the graph before creating it to let them know how the future visualization will look like and aid them to select the proper visualization type [3] [9] [11] [12].

The visualization framework should, as well allow for creating copies of the visualization objects, some times this copies should be interactive and some other times, users will just want a static snap-shot of a visualization that does not update with the rest of the framework [3].

2.3 Linking and Grouping Graphs

Coordinating multiple views has to be a core design aspect of a framework emphasized on showing and working with different and multiple views of the data at the same time. Visualizations should be linkable to indicate somehow that a certain visualization is influenced by the actions performed on another. Visualizations should also be allowed to be grouped in order to apply certain actions to all of them at the same time or in order to assign them some structural meaning. Metadata values can be added to the created groups, this could be either inputed directly by the user or inferred by the system. The framework should allow free lasso selection whenever possible for fine grained selection control over the data and the visualizations[3]. The capacities of resizing and moving groups, keeping their internal structure, can be used as powerful visual cues for the user [2]. In [3] they also explore the use of groups to apply filtering and comparison operations to the whole group.

2.4 Visualization Comparison

We distinguish several interesting interactions from combining the taxonomy of comparisons presented in [13] which uses juxtaposition, superposition and explicit encodings to model relationships; with the interaction patterns to compare visual representations from [7].

Simple comparisons can be achieved by just using flexible layouts and setting objects side by side. Snapping methods can be used in this cases to help the user position the elements. When deeper comparisons are needed “shine-through”, “blink” or sometimes “folding” techniques can be used, to compare objects one on top of the other [3][7]. Working in an unbounded space provides a framework where visualizations can be directly juxtaposed for comparison, reasoning chains can be reviewed, while maintaining prior queries in the periphery provides temporal context [3][5].

2.5 Annotations

We are advocating for a framework which should presumably allow to represent more information at the same time than traditional frameworks, because is embedded in a large screen environment. With more information in the screen which the user has to remember, is a good idea to allow the user to specifically add annotations anywhere in the framework for himself or for others, if working on a collaborative way.

The annotations are done by pen in some systems [3][8]. On others you are allowed to make annotations to the history [15] while even in others you are able to sketch the creation of new visualizations by drawing an "L" form sketch [3].

2.6 Constructing Complicated Dynamic Queries

Because the framework is aimed at analyzing visual representations of numerical data we should allow users to construct operations and new queries within the visual representation themselves, instead of requiring them to work with the numerical representations of the data. Similarly to deriving new visualizations, this allows for exploring new possibilities faster than requiring to reselect the data from the numerical values with the added benefit of not loosing the global context. In [3] they make use of several Boolean operations (OR, AND) to affect the representation of the resulting visualization directly as filtering or brushing. Filtering and brushing are also used in [8]. Both of them also allow to apply different typical operators such as average, sum, count, max, min directly on the visual representation of the data. In [7] their system is able to calculate differences on demand. In [9] users are able to filter, remove, add and apply XOR and AND operations to trajectories in an iterated manner until they extract a relevant set of data constructed incrementally. In [14] users use a "DataRose" structure, able to show more than two dimensions at a time, that allows for multiple operations like selection, filtering, creation of sets and subsets of the data and to establish new data connections.

All of this different interaction models allow to build incrementally complex queries at different steps of the process from the visualization themselves, creating queries that would otherwise be difficult to accomplish if done directly over the database [3]. Another interesting aspect of this approach is, that this way the operations done to the data can be saved in different meaningful ways to the user, even customized ones, and then reused and applied to other groups of data or visualizations [3][10]. [3] and [14] allows the user to use pre-implemented gestures with the mouse for saved operations and interactions. Similarly we think that we should allow the user a way of specifying this gestures in a custom matter to allow them to create their own gestures and operations.

2.7 Interactive Histories

The large space available allows for keeping and retrieving larger amounts of information than traditional environments while is also capable of showing all of them simultaneously in the display. This helps users remember and retrieve more knowledge just by the inherited capacities of using this large displays [2].

Some applications allow to edit, annotate, search, filter and export the history from an external window from the visualization application [17]. The same paper, also mentions the idea of gaining insight from the users past track of history of visualizations. Some applications instead include the history in the same window as the visualizations [8][9][10]. Others explore different approaches such as changing the size of the history window proportionally to its most recent use [15], while allowing to create new branches manually to show that the user is exploring different hypothesis. [15] also mentions that it would be interesting to explore ways of automatizing the creation of new branches.

2.8 Connect Different Databases

Today, the amount of data and the rate at which is being stored is bigger than ever before and the estimations predict that the trend will continue to grow in the near future to serve the needs of technologies such as the “IoT”(Internet of Things) or the new “Big Data” approaches. The possibility of storing data from multiple sources in WebServers that are accessible to interested users, allows for retrieving very different types of data. Examples of this type of approaches are some governmental, Open Data and even industry initiatives.

Some related systems offer capabilities for connecting different databases and explore them simultaneously [8][14] but we feel there hasn't been done a real effort emphasizing on how to explore and visualize different databases simultaneously.

2.9 Collaborative analisys

We understand that this large display environments, when used by teams in simultaneous collaborative analysis or collaborative research, can serve as powerful tools, even more than when used by individuals working alone.

In [2] they mention several ways for document tracking; by saving them in different folders, for example tagged as: explored or not explored; or by assigning metadata to the different documents in the form of customized categories: “important”, “rejected”, “interesting”. In [15] the framework shows the user the most and less explored dimensions of the data, as well as which hypothesis was explored and which evidence was gathered from it, helping users engage in finding the less explored paths. [10] and [16] allow for parallel hypothesis exploration while also indicating which are the less explored dimensions.

3. Data Visualization DashBoard Framework for large-display environments

We have set up the requirements of our system based on the related work aforementioned and the different interactions that we distinguish from the literature.

We want to design a visualization framework focused on the proper use of the large space available. To achieve this, users should be able to reposition and resize all the elements at will allowing them to come up with their own customized structures over the unstructured environment.

As in [3] we are inspired by "a metaphor of narrative panoramas". We think that a visualization dashboard for large display environments should implement different visualization primitives and combine them in space to create complex and sophisticated visualizations. We understand that the large space available consistently adapts for applying a deriving visualization approach, where some of the graph type selection is automatically done, as well as aided, by a graph preview before actually rendering the selected graph. The framework should as well allow to create dynamic and static copies of the visualization elements at will.

We believe that users should be able to add annotations to the data, the visualizations, the groups of visualization and basically any represented element at will. Users should be able to do this easily and freely in the unstructured environment, so we advocate to implement structures in the framework that could support this type of annotations.

We think that a visualization framework should implement interactive histories that allow to review reasoning chains in order to validate results in a much more profound way that when done only based on snap-shots of the different visualizations.

We presume that bigger insights could be captured from the simultaneous exploration of different and disperse datasets. If we are able to find a normalization point between databases, such as date attributes that can be easily normalized, one could imagine connecting different databases and being able to visualize them together in the same visualization.

We understand that a dashboard visualization framework for this large-screen environments should benefit from a design aimed at enhancing collaborative work. The framework should not only allow to gain collective from individual knowledge, but should also support exploring in parallel the different hypothesis, implementing different interactions that could encourage users to work together in the same session.

Some commercial DashBoard Visualization Softwares allow some data operations within their software but this is not done directly from the visualization perspective itself, furthermore this systems almost always do not allow for multiple visualization interaction. The interactivity, when present, is usually confined to one visualization at a time. None of this commercial softwares are focused on working on large screens, so even though they can benefit in some measure when used on bigger displays, we advocate that a system mainly designed towards these big displays should be able to outperform them in numerous ways. There exists some specialized software for the interconnectivity of databases but none of these systems exploit the capacities of the described visualization framework. Some of the systems mentioned in the related work implement some of these operations, others are mainly focused to be deployed over big displays. But apart from the system that we are developing, conceptualized in the rest of this section, none of neither free or commercial systems mentioned are developed in JavaScript or are completely WebBased. The main benefit of this approach is that our system will have only requirement, the need of an up-to-date Google Chrome Browser, making the system deployable over any architecture or computer system, desktop or Laptop.

3.1 The Dashboard Framework

The system can be understood as a "black-box". Which as input takes different databases and whose output is a visualization framework where you can explore, rearrange and "play" with the visualized data directly. In order to customize and linearly update the future implemented system we have aimed to develop the system in a highly modular way. We aim to achieve a level of modularity, such that users could upload their own customized visualizations and work with them and the rest of the framework in a seemingly way.

As it can be depicted from figure 1, the framework consists of four different modules. A Data Manager Module, a Visualization Manager Module, an Operations Module and a Knowledge Manager Module. The Knowledge Manager Module monitors and updates the data as well as the visualization models inferred by the Visualization System. The data Manager Module aids in adding external data into the system. The Visualization manager renders the data into different visualization types inside a democratic framework that allows resizing and rearranging this visualizations in space among other characteristics. The operations Manager allows implementing data operations, like extracting new visualizations derived from the ones that are already in the framework, applying filters, selections or other types of operations, like calculating trends or averages on the data. All of this modules are going to be explained in detail through the following sections.

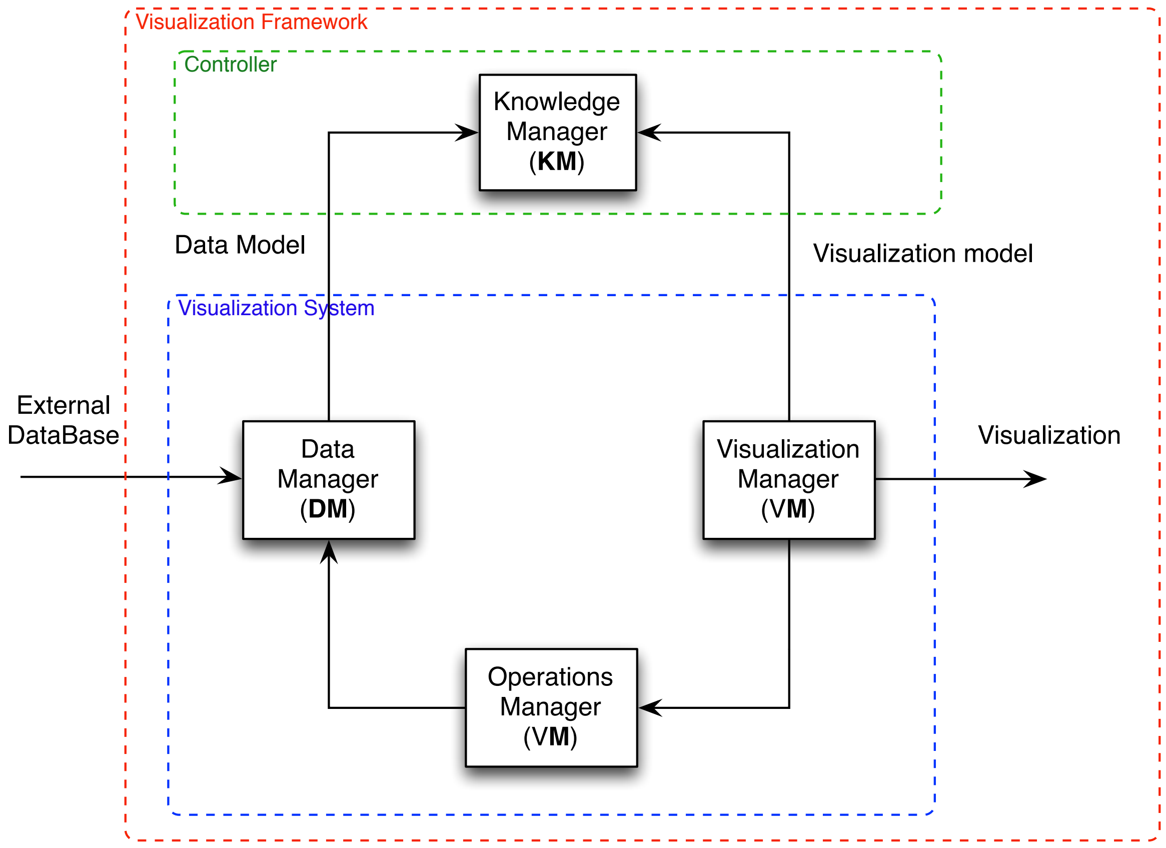


Figure 1

3.2 Data Manager Module (DM)

A first consideration when developing any data visualization dashboard application is how to feed your data into the application. Data visualization users typically possess different types of files holding their data. This files can be stored locally or distributed across computers, depending on the characteristic of the application and the size of the data. Ideally, a desirable visualization dashboard should posses the capabilities to blend all of this different data types, sizes or architectures together into the framework.

The DM module manages the data integration step in the framework. The ultimate goal of this module is to add different databases to the system integrating them in different models which can presumably encapsulate more knowledge that the simple raw data. In order to do this, the module proceeds in three different steps each of them defined as different submodules.

3.2.1 Data Manager Interface Module (DM-I)

In order to address the challenge of connecting to this abrupt data landscape, different independent interface modules are implemented in the system. The fact that the modules are independent supports our scalable developmental effort. This is so because new DM-I modules can be increasingly implemented, on demand, to fulfill different desires and needs of business or research allowing to connect to diverse systems simultaneously.

In general terms our ultimate goal should be to build a system that is:

1. Able to connect to multiple local files of different types : TXT,CSV, JSON,XML,TCV…
2. Able to connect to different Data APIS: SOCRATA, CKAN…
3. Able to connect to classical database frameworks: SQL,ORACLE, POSTGRE, SQL…
4. Able to connect to new available Big Data Systems: CLOUDERA, MONGODB, HADOOP…
5. Able to integrate the different types of data and work with the different databases simultaneously.

3.4.1 Data Manager Preprocessing Module (DM-P)

As it has already been stated, todays data landscape is diverse: Similar data is not structured along any normalization rules, available data appears to be disperse and suitable to contain errors. A framework that wants to make use and explore these multiple data sources needs ways of applying some preprocessing to the data this is the main function of this module.

Some databases already have some metadata attached to them, this metadata helps identifying the different columns, the type of data stored in each of them, the database origin, owner and other important information. This metadata, if present, is processed in this module. In reality most databases where not created along with a respective metadata file, so the information has to be directly inferred from the raw values themselves. A lot of work has been done in data preprocessing and it would be too long here to explain all of the useful preprocessing approaches that could be applied to improve the DM-P module in this project. The general idea should be that an in depth development of a DM-P module probably will enhance usability getting rid of some of the users workload. Even the best of preprocessing will generally require manual processing made by the user, designing the system with the capacity to help the user with this processing should be a good pattern to follow.

3.4.2 Data Manager Storage Module (DM-S)

The last step of the Data Manager module is to store the database along with any gathered preprocessed or inferred information for the framework to use it. The framework models and holds the databases in an internal objects database. Modeling the databases this way, allows the

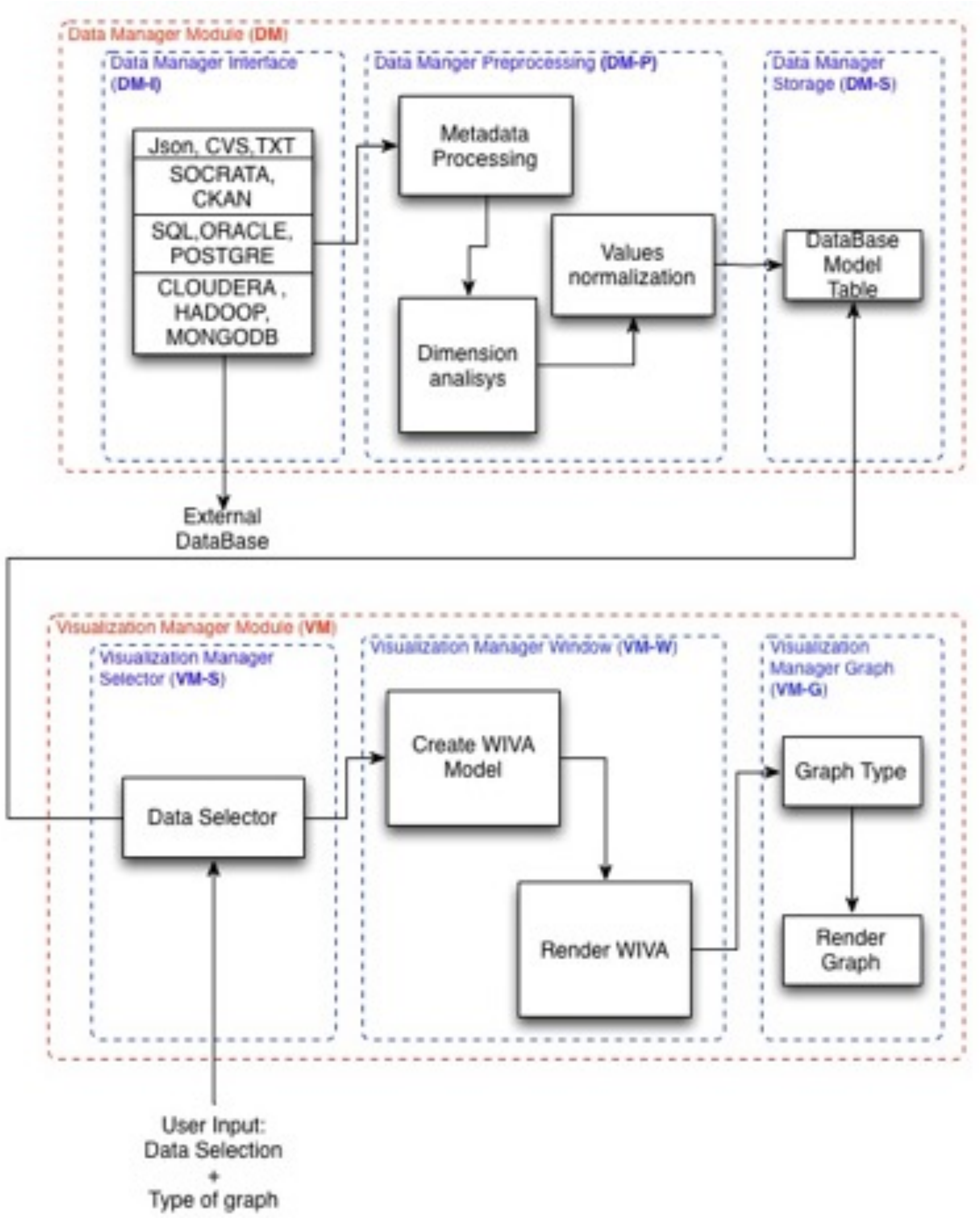

Data Manager Module (DM)
Data Manager Interface (DM-I)
Json, CVS,TXT
SOCRATA, CKAN
SQL,ORACLE, POSTGRE
CLOUDERA , HADOOP, MONGODB
Data Manger Preprocessing (DM-P)
Metadata Processing
Values normalization
Dimension analisys
Data Manager Storage (DM-S)
DataBase Model Table
External DataBase
Visualization Manager Module (VM)
Visualization Manager Selector (VM-S)
Data Selector
Visualization Manager Window (VM-W)
Create WIVA Model
Render WIVA
Visualization Manager Graph (VM-G)
Graph Type
Render Graph
User Input: Data Selection + Type of graph

Figure 2: Shows the framework structure for the DM and VM modules.

system to interact with different databases at the same time with the possibility of adding extra information to them. This information could be automated or come from the user as manual input. But other possibilities can be envisioned, like deriving information from the context or from the use of the application, such as statistical usage of databases or relationships generated when from connecting different databases together.

The main claim we use to support the implementation of this Data Module is that we believe that creating complete, accurate and descriptive models of the data, will eventually reflect in a better, complete and more complex knowledge that can then be exploited by this multi-visualization framework like the one that is being described.

3.3 Visualization Manager Module (VM)

Once the data is inside the application granting direct access to it. This data can be requested by the VM module. The module's goal is to provide the users with multiple ways of interacting and visualizing their data, making emphasis on the possibilities provided by the increased space capacity that the use of big displays support.

The VM module is launched whenever a user requests to create a new graph. As input it receives the type of graph and the data that is going to be visualized. As output it plots the requested data in the visualization form specified inside a **WIVA** (**Widow Interactive Visualization Area**) created by the Visualization Manager Window Module.

This module encapsulates most of the interaction ideas cited in the related work. Implementation of derivable visualizations. User capacity to select groups of data and visualizations while also offering possibilities to resize and reposition this groups in space at will. Users should be able to extract, filter, add, connect, compare and comment the data within the visualization, as well as the visualizations themselves, in order to reduce workload, keep the context and enhance effectiveness. We support the notion of interactive histories in the way that every new derived visualization keeps a child-parent relationship between both WIVA's, keeping track of the path that followed the discovery of an insight and allowing for complex linking between the data plotted in different related WIVA's. The system should allow for collaborative analysis supporting simultaneous users working in parallel and exploiting the tools that improve these collaborations.

3.4.2 Visualization Manager Data Selector Module (VM-DS)

The module selects the proper data values and dimensions needed from the databases for the different types of graphs. The module, based on simple heuristics such as the type or number of dimensions, aids to semi-automate the selection of parameters that are suitable for the different visualization types. The VM-DS should include some feedback in the form of a preview of the new desired visualization, before the user actually commits to render it.

3.4.1 Visualization Manager Window Module (VM-W)

The goal of this module is to create a **window interactive visualization area (WIVA)** along with a model based on the information from the data plotted, the type of graph and the WIVA information itself. An accurate model for a WIVA should include:

1. System Name
2. Title
3. xAxis Name (if needed)
4. yAxis Name (if needed)
5. Type of graph
6. Data plotted.
7. Database id of the data being plotted
8. Parent-Child Relationships

A WIVA should allow for any typical Window interaction, this means it will need to be repositioned, moved and resized by the users at will. Allowing users to create their own mental maps in the large available space.

This WIVA design supports the interactions described in previous sections, the fact that any WIVA can be repositioned in space allows, similar to real life, side-by-side, shine-through or folding comparisons.

3.4.1 Visualization Manager Graph Module (VM-G)

This module is the one in charge of plotting the specified graph on the WIVA. Different functions are needed in order to plot different graphs. Because new customized visualizations types are constantly being researched across different disciplines, it should be possible for a user to implement a customized visualization for their own domain independently from the framework. Then, the user should also be capable of easily embedding this visualization with the framework, in a way that it is consistent with the rest of the system.

The idea is to implement different, independent graph types functions whose output is plotted inside the WIVA. This means that the different graphs will share common functionalities implemented by the WIVA, but they could also possess other differentiating functionalities based on the capabilities defined in the graph type function that implements them.

3.4 Operations Manager Module (OM)

This module is in charge of managing and implementing operations. This operations can range from simple as calculating some average number for some parameter to complex as analyzing trends and correlations between parameters from different databases. Data operations can be specified at different levels of abstraction. This way we distinguish between data, visualization and groups of visualizations operations. We differentiate between operations applied to a data element from operations applied to all, or a group of, the data objects inside a certain, or multiple, visualization elements. In our framework data operations range from classic operations done typically directly to each of the individual elements present in the raw data, to visualization operations that are done within or between different visualizations grouped elements. Because WIVAS can also be grouped we could distinguish another level of operations applied to groups of WIVAS.

3.4.2 WIVA Operations

This module is in charge of managing operations related with the WIVAS. As mentioned previously WIVAS should allow for grouping operations. This way, groups of WIVAS could be rearranged or resized together in space at will, keeping their structure to act as accurate cues for the user even when made very small. This helps users to visualize and keep track of their mental maps. It also allows them to create abstract groups of visualizations that could interact together allowing for higher or lower levels of abstraction when applying operations like filter or linkage at different levels.

Visualizations should be allowed to be derived directly from the WIVAS, the WIVA operations module is also in charge of this types of operations. Deriving new visualizations within the visualizations themselves allows for constructing otherwise complicated queries if done directly over the numerical data itself.

We distinguish two main intentions when a user derives a new visualization from an existing one:

1. User wants to use the same visualization type to render different data values. This could go from just using a subsection of the data present in the original graph to, in other cases, a totally different set of data.
2. User wants to view the same data using a different visualization method.

Keeping track of the parent-child relationship of WIVAS allows the framework to keep a history of the visualizations. This can be used to understand and keep track of the different discoveries, control which are the less explored hypothesis, or can help gaining knowledge of what was the path that led to some insight or result. At the same time this parent-child relationship helps keeping track of the data elements present in the different WIVAS. This type of linking aids in understanding where subsections of the data are coming from within the bigger picture perspective.

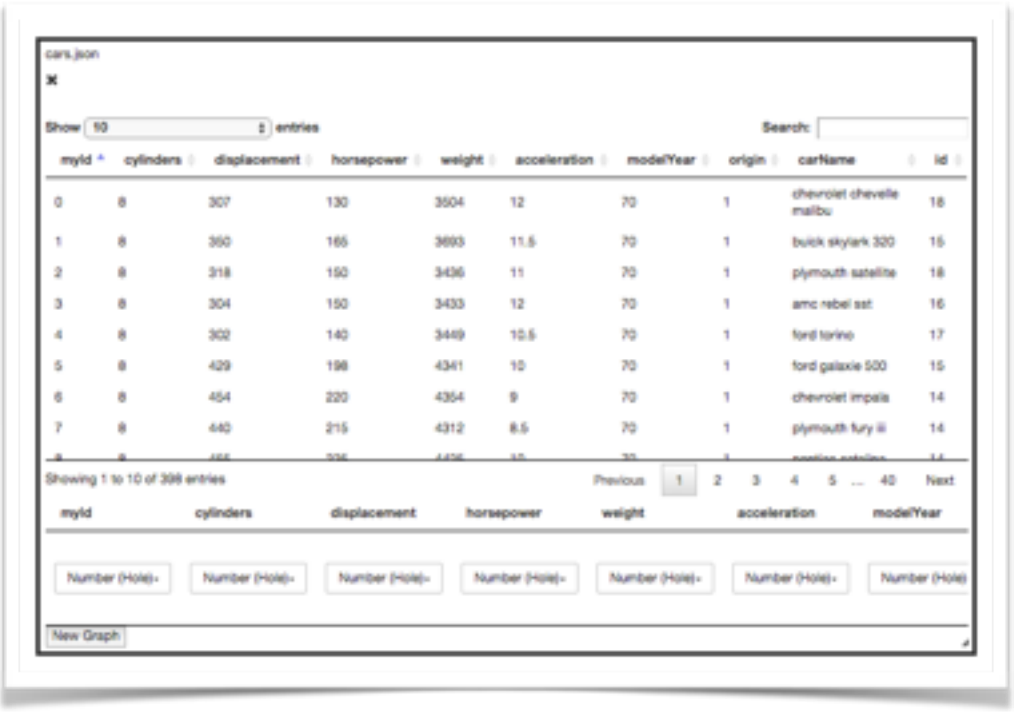
3.5 Knowledge Manager Module (KM)

This approach of modeling the databases and the WIVAS while keeping track of the operations and steps of the user interaction allows for a better understanding of the process of the “sense-making loop” and one could image multiple possibilities that could make use of this knowledge. One could think of applying Machine Learning techniques, for example, to analyze users activities, improve matching between certain dimensions and data type relationships or to gain general knowledge inferred from working with groups or connecting multiple different databases.

This Knowledge manager module could be used as a future system controller to automate part of the process and enhance the interaction between user and the system, learning about user preferences, for example, could a be one simple example of this.

4. Results

In the following section some images from the developed system will be shown, we try to emphasize in a typical exploration that can be done using this type of framework and in order to clarify the rest of the paper.



myId	cylinders	displacement	horsepower	weight	acceleration	modelYear	origin	carName	id
0	8	307	130	3504	12	70	1	chevrolet chevelle malibu	18
1	8	350	165	3693	11.5	70	1	buick skylark 320	15
2	8	318	150	3436	11	70	1	plymouth satellite	18
3	8	304	150	3433	12	70	1	amc rebel sst	16
4	8	302	140	3449	10.5	70	1	ford torino	17
5	8	429	198	4341	10	70	1	ford galaxie 500	15
6	8	454	220	4354	9	70	1	chevrolet impala	14
7	8	440	215	4312	8.5	70	1	plymouth fury iii	14

Figure 3

The system starts up by showing a white canvas, from here the user is able to select a database from a menu. Databases must be saved in a specific folder in the local computer so they can be

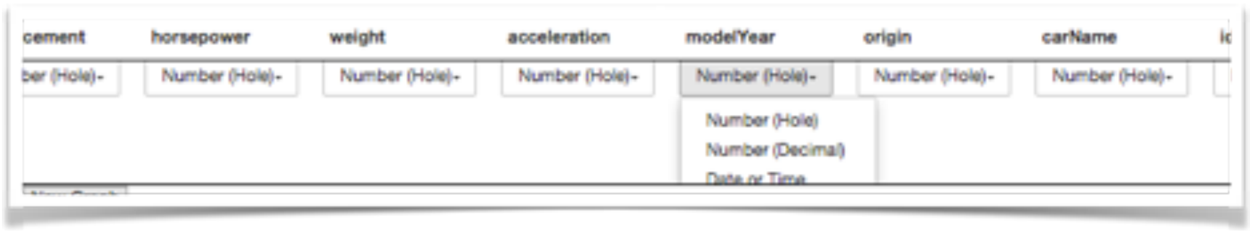


Figure 4

retrieved from the application. As for today the system only supports .cvs and .json files.

When the user selects a database, the data is shown as seen in figure 3, the data can be explored from here. This window is as resizable and movable in the empty canvas, so the user can eventually open several windows like this one and visualize the data simultaneously from all of them at the same time.

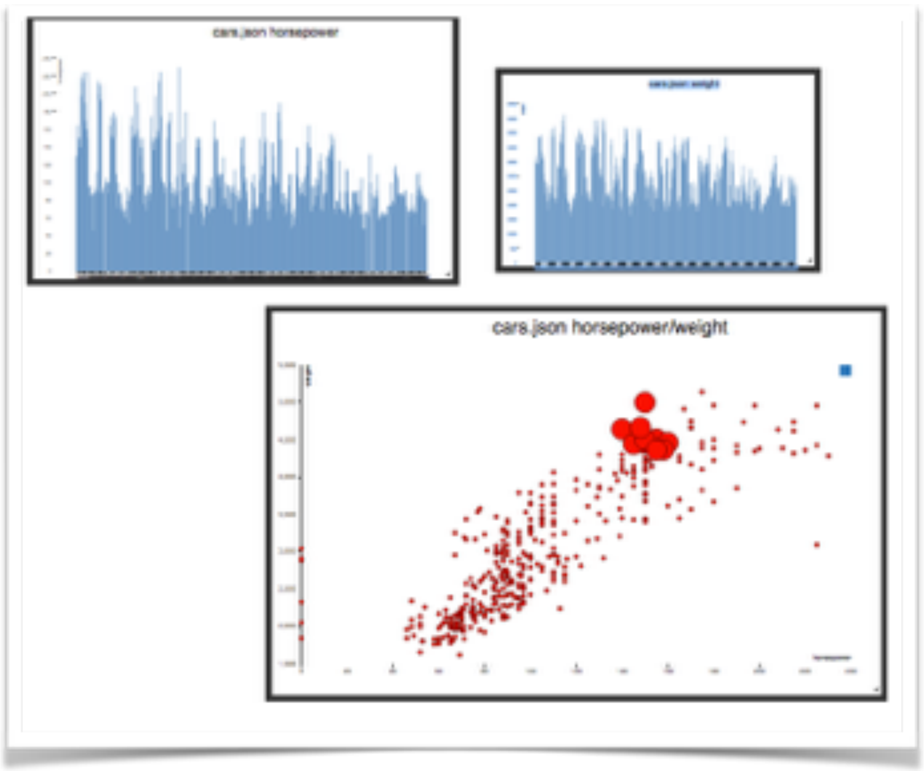


Figure 5

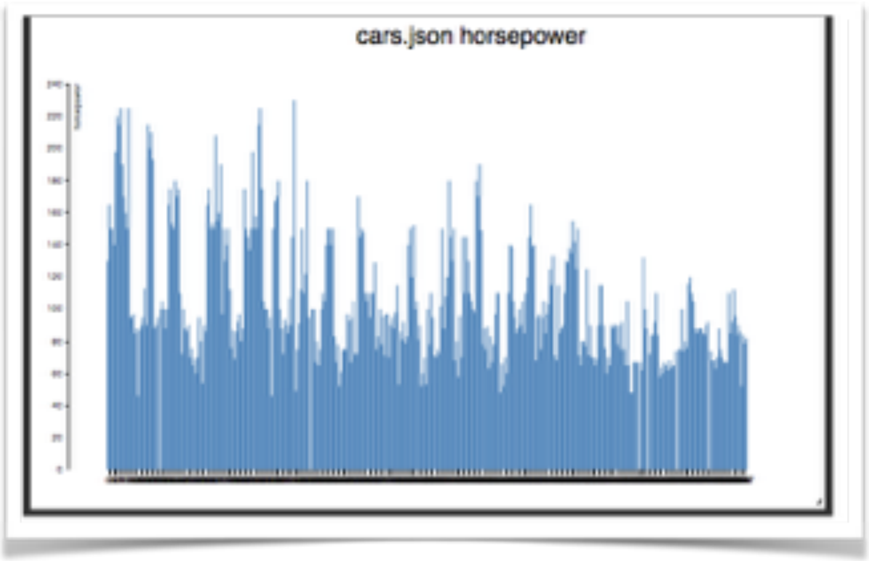


Figure 6

From this window users can specify and do some processing to the data like specifying the type of the different dimensions in case the values are not parsed correctly in the original database. For example the user can select the option Date or time from a drop-down menu in order to specify. the system that the values in that columns represent time values. Figure 4.

From this same window the user can create new graphs, being able to select from the different dimensions of the database which ones to render and which type of visualization should be used from a dropdown menu. Figure 7

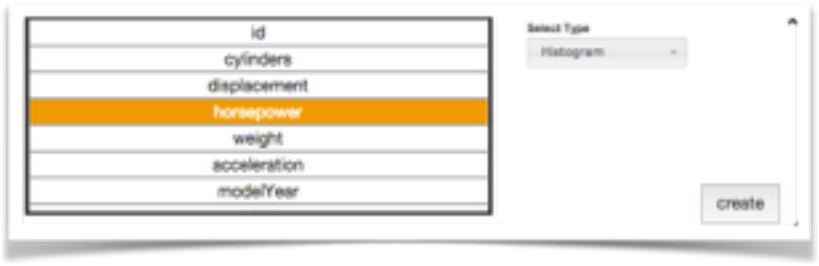


Figure 7

The user, for example can decide to visualize a histogram of the horsepower values in the dataset, a resizable movable WIVA with the desired graph inside will get rendered in the canvas.

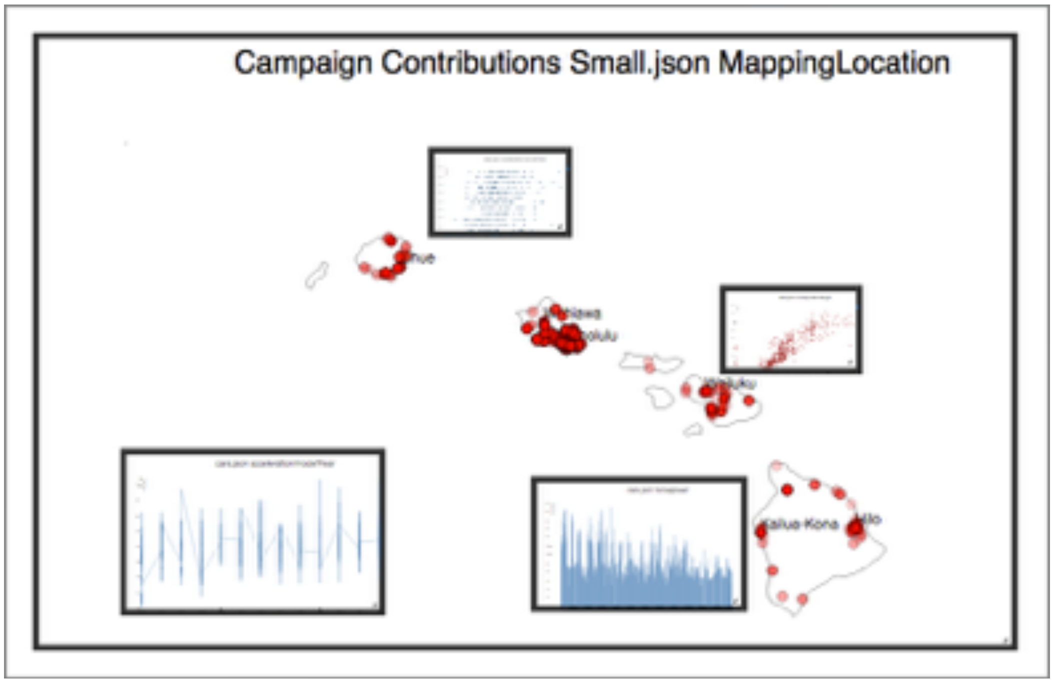


Figure 8

After looking at the data, the user might decide to proceed and do the same with the weight, in order to compare both graphs side by side. Then he decides to visualize both dimensions together in a scatterplot. He discovers that both of this dimensions seem to correlate linearly. He observes some values that are representative for what he is searching for and decides to mark them. Figure 5.

The user decides to derive another visualization containing only the selected values to focus the investigation only around the interested features. Figure 9.

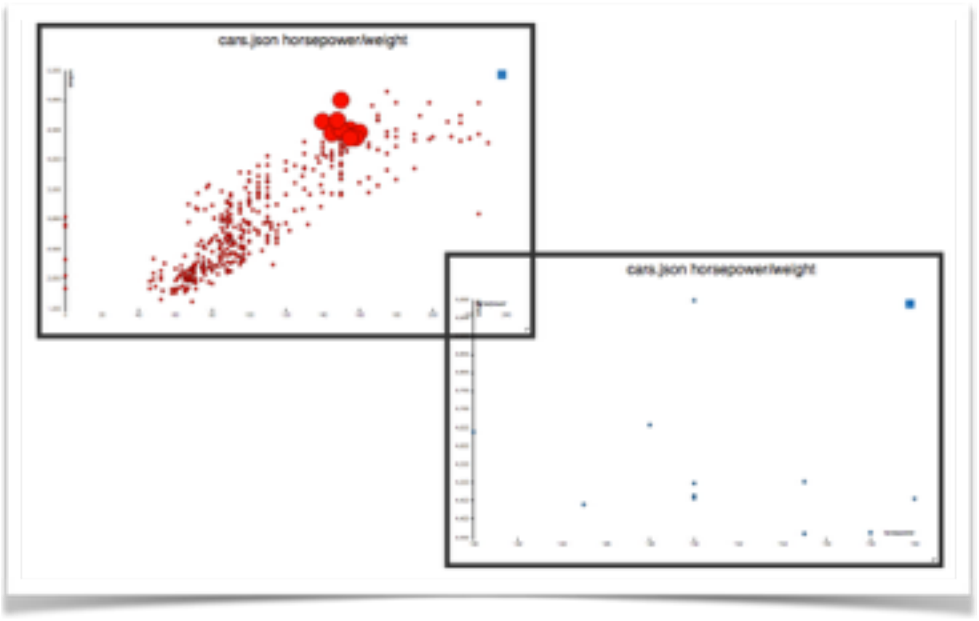


Figure 9

This free environment allows to construct much more complex representations of the data than it could be expected to, by just making use of the space semantics and the use of simple visualization types.

4. Conclusion

The framework described in this work is designed specifically to exploit the space and high-pixel density of large displays. The framework is also designed to support what is known as "fluid interaction" and does so, by providing solutions to the following challenges:

- **How to make a better use of the available space:** The designed framework provides the ability to place and organize visualizations arbitrarily on a large display canvas.
- **How to aid the task of creating new visualizations:** The designed framework provides the ability to derive visualizations from existing ones, allowing one to create new representations of the data faster. The designed framework provides ways for users to select subgroups of the data inside any visualization, or groups of visualizations, and create a new graph only plotting the elements of that subgroup. The designed framework provides ways to simplify the selection of the graph type based on simple heuristics and feedback to the user in the form of graph previews.
- **How to keep track of the data in multiple visualizations:** The designed framework tracks the origins and destiny of the data when deriving new graphs, enabling all child visualizations to refer to the same data represented in their parents, and allowing the tracking of the same data object along all visualizations.
- **How to rearrange, resize and reposition the elements in the space:** The designed framework provides tools that allow the reposition an resizing of all the elements in the space at will. Visualizations can be grouped together in order to reposition or resize the group at once. The free use of space combined with this tools allows the user to create its own customized descriptions of the space and give them their own meaning.
- **How to connect different databases:** The designed framework provides the ability to connect to multiple databases and explore them simultaneously on the large display canvas.
- **How to engage in collaborative work:** The designed framework provides the necessary tools to engage users to collaborate together making use of a democratic canvas where elements can be resized, relocated and explored at the same time simultaneously from various users.
- **How to reduce complexity of the operations:** The designed framework provides ways to reduce operation complexity and maintain global context by operating directly in the visualized data rather than over its numerical values.
- **How to keep track of the history:** The designed framework provides ways to keep track of the history of creating new visualizations on the large canvas, while keeping the old visualizations at the same time in the display. This way all the exploratory session can be visualized together keeping track of the history. The framework keeps track of the number of

the visualizations, their parents or/and children, their position and their ordering providing information that can be harvested and used to keep an informative up-to-date history of the exploration of the visualization.

The described framework is solely developed in JavaScript and currently is implemented in a very basic form as a WebApplication on the client side. The implemented work only encompasses a small part of the described capacities of the framework. In the future we expect to implement the system in a client-server environment to serve the needs for our SAGE Application. Following the architecture described, we believe it would be desirable to leave the Visualization Manager module on the client, implement the Knowledge Manager and the Data Manager module on the server, which could require more time and processing power and for which operations can be handled mostly asynchronously. Furthermore the Operation Manager module could be distributed across the client and server depending on the type of operation requested.

4. References

1. Information Visualization on Large, High-resolution displays: Issues, Challenges and Opportunities. C. Andrews. October 2011.
2. Space to think: Large High Resolution displays for Sense-making. C. Andrews. October 2010
3. Panoramic Data: Data Analysis through pen and touch. E. Zgraggen. December 2014
4. Fluid interaction for information Visualization. E. Elmqvist. October 2011.
5. GraphTrail: Analyzing Large Multivariate, Heterogeneous Networks While Supporting Exploration History. C. Dunne. 2012.
6. Show Me: Automatic Presentation for Visual Analysis. J. Mackinlay. November 2007.
7. Interaction Support for Visual Comparison Inspired by Natural Behavior. C. Tominsky December 2012
8. ExPlates: Spatializing Interactive Analysis to Scaffold Visual Exploration. W. Javed. 2013
9. FromDaDy: Spreading Aircraft Trajectories Across Views to Support Iterative Queries. C. S. Hurter C. 2009
10. Supporting the Cyber Analytic Process Using Visual History on Large Displays. A. Singh. 2011
11. Rolling the Dice: Multidimensional visual exploration using scatterplot matrix navigation. N. Elqmvist. November 2008.
12. The FlowVizMenu and Parallel Scatterplot Matrix: Hybrid Multidimensional Visualizations for Network Exploration. C. Viau. 2010.
13. Visual comparison for information visualization. M. Gleicher. October 2011.
14. DataMeadow: A Visual Canvas for Analysis of Large-Scale Multivariate Data. N. Elmqvist. 2007
15. Exploiting History to Reduce Interaction Costs in Collaborative Analysis. A Sarvghad. October 2014.
16. An Evaluation of How Small User Interface Changes Can Improve Scientists' Analytic Strategies. R. Jianu. 2012
17. Graphical Histories for Visualization: Supporting Analysis, Communication, and Evaluation. J. Heer. November 2008.