A functional framework for evaluating visualization applications, with a focus on financial analysis problems

by

Luan Quang Dang

B.Sc., The University of British Columbia, 2007

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

in

THE FACULTY OF GRADUATE STUDIES

(Computer Science)

THE UNIVERSITY OF BRITISH COLUMBIA

(Vancouver)

May 2012

Abstract

The aim of this project is to bridge the gap between Visual Analytics (VA) research and application development and deployment in imperfect conditions, and to address some of the multi-layered and often vaguely-defined problems in industry. The first goal of this project is to create a functional framework to evaluate the suitability of VA applications for various investment analysis problems. This framework can guide the development process of such applications and address several secondary challenges to deploying VA solutions in financial organizations, such as problems in file formats and information management. These secondary challenges are relevant not only to VA, but also to the deployment of other computer-aided analytic processes. The construction of the framework began with an extensive literature survey on common investment analysis problems, VA techniques, sense making and intelligence analysis theories, and previous attempts by other VA researchers in creating classifications of VA techniques. It was then developed to evaluate the suitability of several mature VA products with regard to common investment analysis problems.

This evaluation framework was next applied in a case study in a Fixed-income investment company, with the supervision of Dr. Victoria Lemieux and domain-expertise advise from two Masters of Finance students. This study involved systematically investigating the analytic process of the firm and isolating areas that could be improved with VA. These detailed problem definitions were then mapped to VA techniques to find the optimal visualizations and interactions for the problem. The case study also involved implementing a solution for the company by building upon an open-source toolkit that was evaluated with the framework. Preliminary results suggested that the prototype “significantly improves the speed with which we (analysts) see the impact of movements of the yield curve on client’s surpluses/deficits…” The field-testing analyst also did not find any aspect of the interface to be difficult to learn or unpleasant to use.

Preface

An abridged form of this thesis, excluding most of Chapter 5 (the design case study), was accepted for publication as an article for the 2011 Records and Information Management for Financial Analysis and Risk Management Workshop in Vancouver.

A summary of Chapter 5 was included in a chapter of the Handbook of Financial Risk Data (Lemieux, Fisher, and Dang 2012).

Chapter 5 of this thesis is based on work conducted at a Vancouver-based asset management firm, under the supervision of Dr. Lemieux, and with the assistance of Yao Shen and QianQian Yu, two University of British Columbia Masters of Finance students. I was responsible for co-leading the team with Dr. Lemieux, being in charge of the technological design and implementation, and the writing of reports and research articles related to this project. Yao Shen and QianQian Yu acted as my advisors in the domain of finance, especially fixed-income analysis.

The project conducted at the asset management firm involved the interview of human subjects (financial analysts), and thus required ethics approval. The ethics approval for Dr. Victoria Lemieux was granted by the Behavioural Research Ethics Board, UBC BREB NUMBER: H11-00172.

Table of Contents

List of Tables

List of Figures

Acknowledgements

I offer my enduring gratitude to the faculty, staff and my fellow students at UBC, and especially the Department of Computer Science and the School of Library, Archival, and Information Studies, who have inspired me to continue my research.

I owe particular thanks to Dr. Ronald Rensink, my supervisor, Dr. Sidney Fels, Dr. Joanna McGrenere, and Dr. Karon Maclean, whose penetrating questions taught me to be more critical and thorough in considering research problems. These excellent educators encouraged me to go deeper.

Most importantly, I would like to thank Dr. Victoria Lemieux, my co-supervisor, for her on-going guidance, and for enlarging the scope of my visions. Dr. Lemieux encouraged me to see the bigger picture.

Special thanks are owed to my parents, whose infinite patience has supported me throughout my years of education.

Parts of this research project were made possible by funding from The Boeing Corporation, and from MITACS Inc. via a MITACS Accelerate grant.

Dedication

For old but never-abandoned dreams.

1 Introduction

1.1 Background

The use of visualizations in Finance is not new. In fact, the candlestick visualization, one of the most widely used visualizations in Finance was invented by Japanese rice traders in the 16th century (Nison S. 2001). A candle, red to denote a day of lower close and white to denote a higher close, is shaved at the top and bottom to reveal two wicks. The length of the top wick is the highest price of the day, and the bottom wick is the lowest price of the day. This demonstrated the benefit of leveraging human visual perception in financial analysis. Thus, even before the age of computer graphics, it was known that it could be easier to understand a candle chart at a glance, than to read and digest the five dimensions of trading data encoded by this visualization.

Visual analysis has advantages over many other analytic techniques because humans have great visual and spatial skills. These include the ability to detect edges and discontinuities, variations in color, shape and motion; to recognize both patterns and outliers; and to retrieve information using visual cues (Kosslyn, 1980, 1994; Lurie and Mason, 2007). Some of these features, such as edges and color are the result of “pre-attentive processing”, thus perceived prior to conscious attention and understood much more rapidly than words. (Roberts, 2003, Ward et al., 2010). The ability to detect edges and estimate the sizes of a candle allows an analyst to compare rapidly the opening and closing price, and the high and low, between any two days at a glance, without having to calculate additions and subtractions. Likewise, a gap up or gap down from one day to the next jumps out in a candlestick or line chart due to the visual discontinuity it creates. In contrast, to determine such a phenomenon via a table of numbers requires considerable work: two subtractions involving three numbers for each day, over a potentially large number of days, requiring a linear visual search and calculation sequence.

The advent of automatically generated computer graphics has increased the value of visualizations even further. A candlestick chart can now be generated on demand from tabulated price data over arbitrary periods of time. A candlestick chart on 1-day term, over 2 years, represents a staggering amount of information: a table of 520 rows and 5 columns. The

visualization thus acts as a repository of data, which allows people to offload cognition to the perceptual system (Munzner, 2009). This can enlarge problem-solving capabilities by enabling the processing of more data without overloading the decision maker (Tegarden, 1999). Because of such benefits, simple visual analysis has been utilized extensively in many aspects of finance for the last twenty years, and more complex forms have lately enjoyed increasing attention (Lemieux, Fisher, and Dang, 2011). Many visualization products were developed with finance and business intelligence in mind, such as Xcelsius, Quantum4D, and Panopticon.

Curiously, however, scholarly research into the application of visualizations in finance has been relatively sparse. One of the major reasons for this has been the difficulty in obtaining real, internal data and detailed problem descriptions from financial organizations, and conducting research on the inner workings of the firms. The field-defining document for Visual Analytics (VA), “Illuminating the Path”, pointed out that visualizations cannot exist in a vacuum, and that the development of new visualization techniques should be guided by the needs of “customers”, their analytics problems, workflow, organization, and data (Thomas et al., 2005). But to date this has apparently not been taken seriously in the area of finance.

To encourage the use of VA in the financial industry and other related organizations, the VA researcher needs to rigorously prove its benefits. In particular, the VA researcher needs a model to put a value on the solution of domain problems with VA techniques. The case for VA research in finance becomes more powerful if the value of the solutions, less the cost of developing and deploying VA systems, can be quantified and compared with the value of more proven paradigms of analysis.

1.2 Justification of Project Topic

The first part of this thesis develops an evaluation framework of applications of VA in financial analysis using a literature and product survey. It represents a valid contribution to the domain of Computer Science by:

- Describing several recent commercial visualization products, which may not have appeared in scholarly literature, but are still relevant to the VA field.

- Providing a roadmap, in the form of the evaluation framework of applications, for researchers in VA to identify current unmet needs in the domain of Finance analysis.

- Identifying interesting problems in VA that are unsolved or not easy to identify, but clearly relevant in the context of applying VA to Finance.

The second part of this thesis is a detailed case study of the development of a Fixed-Income Asset Management Visual Dashboard. This also represents a valid contribution to the field of Computer Science, by:

- Identifying several types of analytic problems that Financial Professionals (in this case Fixed-Income Asset Managers) seek to solve, and assisting to determine their understanding and acceptance of VA systems as analytic tools.

- Critically analyzing existing VA technologies and their effectiveness in solving real world problems. This includes identifying the root causes of failures, including those due to the limitations of current technology, inherent problems in the visual analysis paradigm, and the process of integrating and applying the technology in a real setting.

- Raising awareness and interest within both the Finance and VA communities about the potential synergies between the two fields in terms of further research and investment interests.

The scope of this project is adequate as a thesis for the partial fulfillment of the requirements for the degree of Master of Science in Computer Science, because:

- Building an application evaluation framework of VA in Financial Analysis via a literature/product survey requires an understanding and expertise in both domains, and requires a larger effort than the literature surveys done in a typical thesis because:

    - The domain of VA by itself is a combination of multiple disciplines, such as Computer Science, Cognitive Science, and Information Science.

- Framework building involves the categorization and synthesis of knowledge and theories gleaned from the literature survey, and also involves framing a large body of knowledge into a coherent structure that would be useful for design decision-making. This takes both expertise and time.

- The detailed case studies of building a Visual Analytic Dashboard for a Fixed-Income firm demonstrated several skills typical of a graduate thesis:

  - Relative mastery of many topics in VA, as well as working knowledge of the process of applying these technologies into real world problems. This commands a similar amount of due diligence as taking a course in VA to acquire the same knowledge.

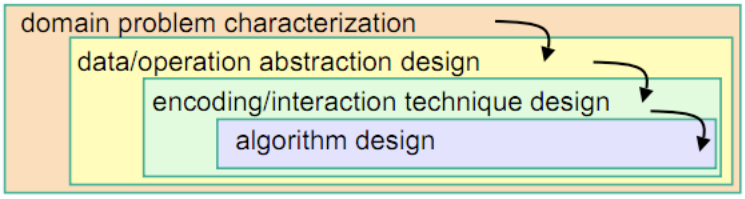
  - Ability in Human Computer Interaction by co-leading an interdisciplinary project team, working with a real organization and real data, in a participatory design project.

  - Technical ability, by implementing the software for the Fixed-Income Visual Dashboard.

1.3 Related Works

This section discusses the progress and challenges of VA research, especially in financial analysis, with regards to a recent model (Munzner 2009) for the creation of visualizations (Figure 1).

Figure 1 Four-layer Model for Visualization Creation



Source: Munzner (2009)

Before the formal creation of VA as a distinct enterprise, the greatest strides in related research were in the development of graphical presentations, interaction techniques, and visual metaphors (as a whole, the "visualization") of information. The field of VA inherited this foundation. Many of these visualizations have been applied to financial data: variations of candlestick and line charts, stacked and bullet graphs, scatterplots, parallel coordinate plots, bead cluster diagrams, treemaps and heatmaps, node-link diagrams, etc.

Commercial financial visualization products have attempted to support the many proprietary data formats specific to large data vendors such as Bloomberg, Thompson Reuters (Datastream), Nasdaq, etc. There has also been a push to standardize the data formats for finance and business intelligence, with the development and adoption of OLAP "cube" (Codd et al., 1993; Chaudhuri & Dayal, 1997; Gray et al., 1997) and XBRL formats (Gunn, 2007; SEC, 2008). These standards ease the process of applying visualizations into finance, by shortening the deployment pipeline, bypassing moats of proprietary data storage and transmission standards.

At the current state of the art, one could make a reasonable hypothesis that there is a wealth of research on visual encoding/interaction and algorithm design in the last two decades, which could potentially help optimize and enrich financial analytics, but is currently under-utilized. This hypothesis cannot be easily verified, and the mapping of financial domain

problems to existing taxonomies of analytic tasks in VA could benefit from further empirical validation. However, one may convince oneself of the under-utilization of VA in Finance through an observation of modern financial analytics suites, such as the Bloomberg Terminal (Leca D., 2010) (Figure 2), or computerized trading platforms, such as ThinkorSwim (2011). Apart from visualizations that are hundreds of years old, such as candlestick and line charts, tables of numbers and rows of text dominate these two interfaces.

Modern visualization techniques, even those developed during the past decade, are still not widely employed in financial analytics products. ThinkorSwim recently implemented heatmap and glyph visualizations in their trading software, but chose to hide them deep in the interface instead of making them the default view. In 2010, three design firms, HappyCorp, IDEO, and Ziba, were commissioned to present proposals for a revamped Bloomberg terminal. Only HappyCorp used visualizations other than line charts in its redesign. Why is the adoption of VA techniques and concepts in the financial industry lagging behind the progress of VA research?

Figure 2 The Bloomberg Terminal, as of 2010

Source: Leca D. (2010)

A user-experience (UX) designer in charge of a hypothetical design project for a financial analytic interface is clearly armed with a powerful arsenal of visualization and interaction design techniques. What is lacking, however, is an understanding of domain problem

characterization and data operation/abstraction design. As Tegarden (1997) noted, good information visualization is not only task dependent, but also domain dependent. Amar and Stasko (2004) further elaborated the phenomenon of “representation primacy”, the drive in visualization research to focus on the faithful visual representation of information instead of the facilitation of analytic processes. Given the high monetary stakes invested in the financial industry, it is crucial that a practitioner or researcher of VA be able to rigorously justify the choice of modern visualization and interaction design techniques to financial analysis problems. This involves proving that the visualizations and interaction design can indeed enhance the speed and accuracy of decision-making, produce new and valuable insights, and ultimately, drive the bottom line.

Information visualization has recently begun to place greater emphasis on the problem domain. For example, Savikhin and colleagues’ developed PortfolioCompare (2011), an interactive VA tool supporting the task of comparing and choosing among several portfolios consisting of different financial instruments. Likewise, WireVis (Chang et al., 2008) contained visualizations built around the task of detecting anomalies in wire transfer data.

Lei and Zhang’s 2010 work specifically focused, not on “developing visualizations”, but on a domain-specific system for financial time-series data. The task of technical analysis of securities was reduced to its comparison and pattern recognition aspects, and the semantics of the historical security price data used in these analysis tasks were described. The resulting visualizations could then accommodate these domain-specific concerns.

To understand the process of justifying financial VA systems in general, one can start by examining the previous examples of domain-focused VA research. Here, VA experts evaluate a system in terms of its ability to help analyze the high-level, domain-specific problems and the data abstractions, and choose the appropriate visual representations. In the financial domain, therefore, what is needed is a functional evaluation framework of visualizations and interaction techniques with regard to analytic problems in finance. There is likewise a need for a method to reduce domain and firm specific problems to better-defined cognitive and visual perception problems, for which visualizations and interaction design techniques could be evaluated.

As such, the goal of this thesis echoes Tegarden's call for taxonomies of domains, tasks, and visualizations (1997). A functional evaluation framework of VA applications in financial analysis is analogous to a taxonomy of financial analytic domains, tasks, and visualizations, which mirrors the phases of a VA application design project:

- Understanding the domain.
- Defining the analytic tasks.
- Choosing visualizations and interactive elements.

Finally, there is a need for the evaluation framework to extend to the implementation-phase concerns, such as the choice of whether to adapt an existing VA product to the task, or building a custom application with a visualization API. In real world projects, VA experts may be called upon to provide technical recommendations for the implementation of VA solutions. The distance between theoretical analysis and practical implementation would shorten if the VA experts could call upon existing products or API's that had implemented the visualizations and interactive elements needed, instead of having to reinvent the proverbial wheel after the problem definition and design phases.

2 Methodology

The task of evaluating visualizations and interactions for analytics tasks is complicated, because real life problems are complex, context-dependent, and very difficult to decompose into well-defined elements. Decomposing a domain problem in a systematic manner involves three processes:

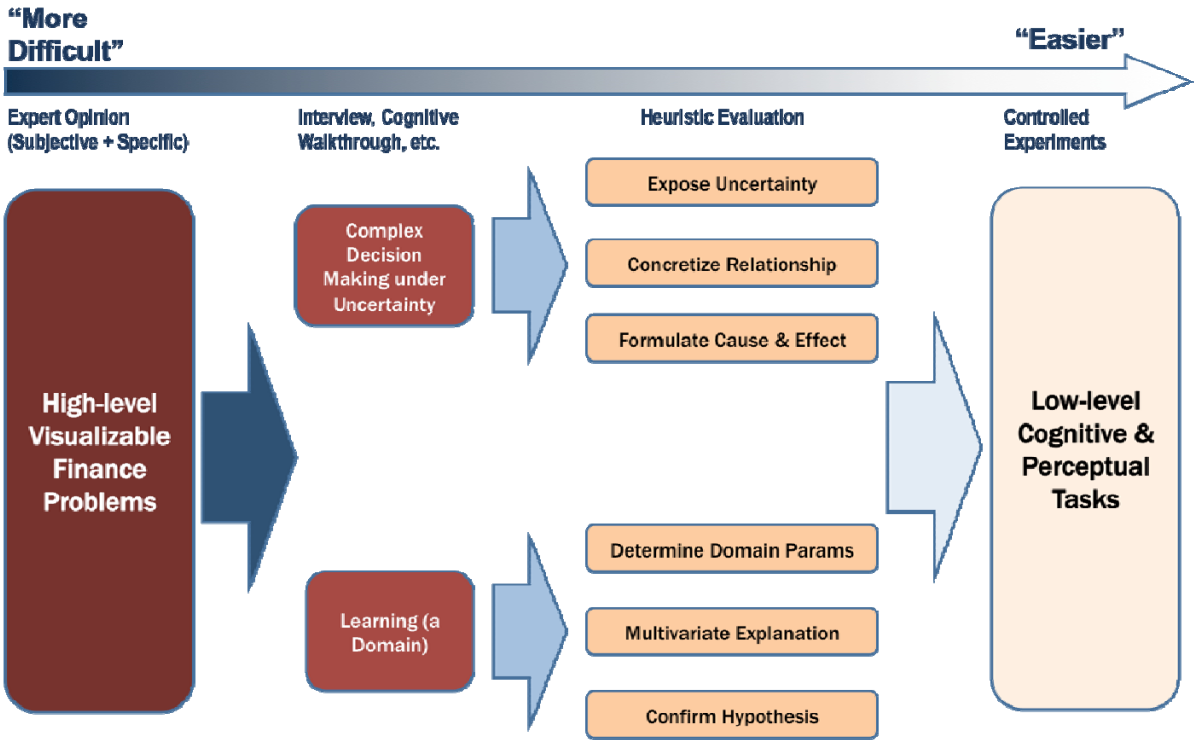
1. Narrowing down the potentially infinite analytic problem space in the real world (even within just a branch of finance) to a finite number of general classes of problems, which represent a large number of common analytic tasks.

2. Defining, in precise detail, a finite list of simple operations, such as information seeking, perceptual abilities, and cognitive tasks, that real world problems can be reduced to. This list must satisfy the requirement of being both general enough that it can be used to describe a large set of analytic problems and yet concrete enough that scientific studies can be designed to measure the suitability of visualizations and interaction techniques.

3. Making explicit all assumptions in the definition of analytic problem classes and the low-level elements, into which analytic problems could be decomposed. At the point of this work's publication, there are many competing taxonomies of financial terms and financial analysis tasks, and yet few existing works on systems for decomposing complex analytic tasks into elements. This situation is similar to the early days of chemistry, where a plethora of physical matters had to be described in terms of a much more limited periodic table of the elements, and, more importantly, a limited understanding of compounds and how the elements can bond together. To define a list of elements, and to settle upon a definition of, say, the "portfolio monitoring" task, would require assumption making for the sake of practicality.

Figure 3 illustrates the process of evaluating the suitability of a visualization for classes of analytic activities, from abstract, complex, and domain specific, to concrete, well-defined cognitive and perceptual tasks.

Figure 3 Difficulties in Evaluating Visual Analytics



The term "functional evaluation" in the title of this framework is defined as an evaluation of all the necessary aspects to be considered by a designer in a VA application design project. This broadens the scope of evaluation, encompassing not only an evaluation of whether certain visualization and interaction techniques are helpful in supporting the analytic tasks, but also whether these techniques are suitable for the data available, as well as an evaluation of existing products and toolkits that a designer can leverage given the implementation constraints (which can be entirely unrelated to the analytic tasks).

2.1 Building Upon Previous Models for Visualization Design and Evaluation

In order to evaluate visualization techniques, the author built upon Munzer's 2009 nested model for visualization design and validation. In this model, Munzner described the high-

level steps to designing visualizations, identified the threats to the validity of the design, and prescribed ways to evaluate these threats. Although the phrase “Visual Analytics” was not mentioned in that work, the nested model reflected a focus on the problem domain characterization and data abstraction aspects, which are crucial for analysis work. Munzner’s model is appropriate as a foundation for this evaluation framework, because this model “was also influenced by previous pipeline models to guide the creation of visualizations”, such as those of Card et al. (1997, 1999), Chi and Riedl’s (1998), and, especially, Shneiderman’s task-by-data-type taxonomy (1996). Munzner’s improvement upon the previous models was to couple the act of evaluation with the step-by-step process of a visualization design project.

This thesis extends Munzner’s previous work defining the evaluation dimensions within each phase of the design process in enough detail so that the framework can be readily applied to a real project setting, while acknowledging the limitations of current technology and the difficulties in defining and breaking down domain problems. This evaluation framework also branches from previous models by its focus on one specific domain (financial analysis) and on implementation/deployment issues.

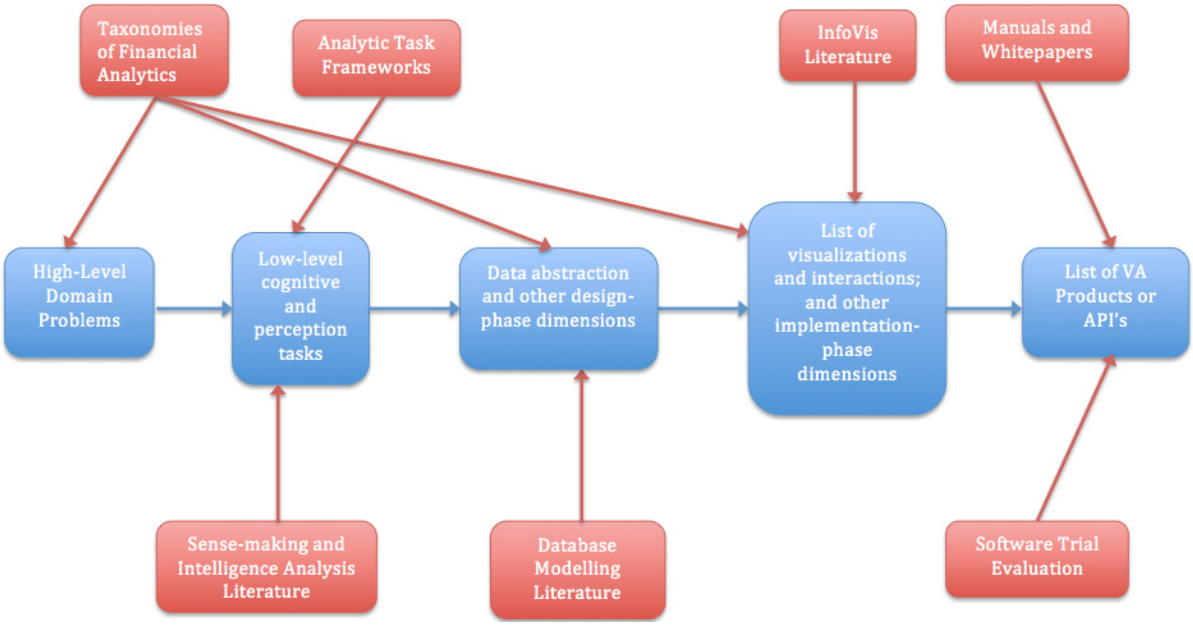
The following high-level dimensions are used in this framework to distinguish between various types of visualizations. The first three dimensions correspond directly to the first three steps in Munzner’s model:

1. High-level problems specific to the financial analysis domain
2. Common data abstractions and operations in financial analysis
3. Interaction design and visual encoding

For the first two dimensions, several strategies are recommended for the VA designer to identify a subset of these analytic tasks and data issues in an organization. Strategies are also proposed to deconstruct “problem characterizations” and “data abstraction / operation” further via organization-specific details. For the third dimension, a representative set of interaction techniques and visualization techniques will be evaluated for suitability with regards to the analytic problem and data dimensions.

Figure 4 shows an overview of the way that the functional evaluation framework was structured and the methodology on which this structure was developed. This approach was based on a generalized development process for a financial VA application. The blue text boxes represent the phases by which the dimensions will be grouped, and the red boxes represent the literature or product / API review work that grounded the choices and descriptions of the dimensions.

Figure 4 The Development Process and Conceptual Structure of the Functional Evaluation Framework



The framework also added a new dimension to the taxonomy:

4. Workflows and operational processes into which VA would be integrated (and that would influence the choices in implementation).

The other major difference between the framework presented in this thesis and Munzner's approach involves the focus on algorithm. This framework does not involve a detailed evaluation of algorithms, due to the difficulties in evaluating algorithms in visualization API's and products. The evaluation of algorithms involve several aspects, such as

correctness, which is evaluated by comparing expected behaviour and algorithm logic or code, and *performance*, evaluated with regard to execution time and memory consumption.

To begin with, obtaining the logic or code of key algorithms in visualization systems is difficult. Many vendors use their own trade-secret or classified algorithms that are not described even on a high-level, let alone documented in details. Even on open-source systems and toolkits, to discover and examine the algorithms would require navigating large and often under-documented code bases. A correctness evaluation of the nuances in a significant number of algorithms would be prohibitively time consuming for this work. That said, this issue does not cripple a VA application designer in practice, because during the consideration of visual encodings, data, and interactions, many elements of algorithm correctness are already considered. This is because visual encodings and interactions tend to be tightly-coupled with underlying algorithms (Munzner 2009, p.6). Algorithm correctness in visualization systems can then be evaluated qualitatively, in terms of the visual quality and perceived effectiveness of the visual encodings and interactions in solving the target problem (Munzner 2009). In practice, the application designer often can test algorithm correctness by presenting "still images created by the algorithm or video of its use" or a live demo, where the target audience of the application "can directly see that the algorithm correctness objectives have been met (Munzner 2009, p.5). This post-hoc method is adequate for the scope of this work, because the goal of this evaluation framework is not to guide the design of new visualization techniques, but that of visualization applications that are both domain-specific and organization-specific.

Performance aspects of algorithm, such as execution time and memory cost, are also relevant factors. While several alternative algorithms can yield the same correct visual encodings and interactions, time and memory performance can significantly affect the usability of the final product. A thorough evaluation of time and memory performance would involve either computational complexity analysis or building and benchmarking test applications. Computational complexity analysis is impossible for many algorithms, products, and toolkits due to a lack of documentation; benchmarking test applications for a significant number of algorithms would be very time consuming. In the current iteration of this framework,

algorithm performance is only evaluated via a few simple metrics of scalability. This is a limitation that could be addressed in future research.

The major goal of this functional evaluation framework is to develop methods of

i. Choosing among various existing visualizations and interactions, and

ii. Creating technological pipelines using existing API's and, when feasible, products, that would satisfy all the implementation constraints of a project

Other taxonomies of visualization techniques and/or products exist. For example, Keim and Kriegel (1996) classified visualization techniques according to five classes of graphical representation. Chuah and Roth (1996) classified visualization techniques in terms of the interactive features that could be implemented into them. Tory and Moller (2002) classified visualizations based on data models, with a focus on numeric data and leaving out multi-dimensional and relational models. However, these taxonomies tend to focus on one aspect of visualization design, not the whole process.

Several taxonomies of visualizations take into account more than one factor of VA application design. Examples include those of Buja et al. (1996) and Qin, Zhou, and Pei (2010). Buja et al.'s taxonomy considered several analytic tasks: finding gestalt features of data, posing query (on the details), and making comparisons. These tasks were then supported with three classes of interactive manipulation (i.e. focusing, linking, and arranging views) and three types of visual representation: scatter plot, functional transformation, and glyphs. This work was limited by considering only a few visualization and interaction techniques. Qin et al. evaluated a large number of techniques and products, and considered four aspects of VA application design: task, data, representation, and interaction. However, the dimensions of evaluation were not extensively and thoroughly-defined. For example, the interaction types considered were "manual", "mechanized", and "steerable", but these were defined only on a high conceptual level. No rationale was given for the choice of these specific classification criteria in each dimension.

One of the main goals of this work is to help VA application designers communicate and justify design decisions, and thus, the dimensions are identified and described to a much

higher level of detail, reflecting the nuances of practice. In addition, this functional evaluation framework takes into account many implementation-time and deployment-time dimensions that are not addressed by any previous evaluation framework or visualization taxonomy, to the best of the author's knowledge. The word "functional" in the name of the framework reflects the emphasis on assisting the VA application designer in practice.

All in all, this *functional framework for evaluating visualization application (2FEVA)* in financial analysis takes into account nearly one hundred criteria of evaluation, arranged into eighteen broad categories derived from the four high-level dimensions. The domain-agnostic name was chosen to reflect that many of the dimensions in the framework are relevant to VA application design beyond the financial domain. Nevertheless, the discussion in this work takes into account domain-specific factors of financial analysis. In the remaining sections, the four high-level dimensions in this framework will be described at increasingly lower levels of abstraction, and increasing levels of detail.

2.2 Describing the Dimensions from a High Level

The process of describing the dimensions of evaluation begins at a domain-specific conceptual level, with identifying classes of problems in finance that may benefit from interactive visualization applications. This involves a survey of previous works on classifying financial analysis tasks, beginning with an examination of a well-known and frequently-updated taxonomy of financial services, the Dow Jones Financial Services Taxonomy (2010). This led to the identification of a list of financial services personnel, the actors in most use-cases in finance that may be visualizable. This list turned out to be vast and difficult to analyze further, as each actor would potentially be initiating multiple use-cases, each of which is a high-level domain task. Although the eventual goal of creating a functional evaluation framework is to account for a majority of common visualizable problems in finance, the problem characterization dimension of the framework was narrowed in this thesis for the sake of practicality. This work only discusses the domain problems related to two very common, and related, actors in finance: financial analysts (buy and sell side) and asset managers.

The finance literature on the taxonomy of activities of these actors was then consulted. Ramnath, Rock, and Shane (2008) described a taxonomy of financial analysts' activities in terms of information input and analytic output, based on a review of nearly two hundred and fifty research papers in financial analysis. The role of portfolio manager overlaps in some duties with that of the financial analyst, with the addition of working with a list of client data to manage risks within a portfolio of many securities and, often, interfacing with clients.

From this model, the following high-level activities were identified:

- Conceptual, qualitative analysis of a firm's prospect
- Earning forecast and price forecast over various time frames (and methodology)
    - Fundamental analysis: a qualitative, mostly manual process
    - Semi-automated technical analysis: a hybrid qualitative and quantitative process that can be aided with some automated tools, but still depends on the judgment of a human analyst
    - Automated technical analysis
- Client management
    - Relationship management
    - Financial planning
- Portfolio monitoring (for performance, allocation, diversification)
- Social networking
- Risk management
    - Modeling risk
    - Scenario analysis

The following high-level classes of information input were identified:

- Earnings
- Price / Volume Data
    - On-manual-demand quotation, typically for over-the-counter securities
    - Streaming quotation, typically for exchange-traded securities
- Information from filings with security regulators (SEC, FINRA, etc.)
- Industry Information
- Macro-economic Information
- Management Communication
- Social Network
    - Relationship
    - Expertise and skills
    - Reputation

Several of the high-level tasks were not included in this mapping, namely financial planning and risk modeling. Both of these are related to the work of the financial analyst and the portfolio manager. However, they are broad enough to merit consideration separately outside of this work. Indeed, modeling risk and financial planning are also the domains of two separate actors: risk manager and financial planner.

The framework also takes into account the high-level but not finance-specific task of *documenting analytic provenance*. Analytic provenance is defined as "understanding a user's reasoning process through the study of their interactions with a visualization" (North, 2011). A similar concept, "keeping a history of actions to support undo, replay, and progressive refinement", was first mentioned as one of the component tasks of information exploration in

Shneidermann 1996's task framework, and was named *History*. Financial analysis activities can benefit from documenting analytic provenance, because it aids in convincing stakeholders, supports the auditing process, and increases transparency of internal processes and external reporting. However, it is a difficult task to define, because it contains both a high-level and a low-level component. In particular, this task involves the comparison between historical states of a visualization, a low-level task, and the documentation of analytic processes, which can be a very complex and abstract task. *Comparing historical states* can be conceptually decoupled from *documenting analytic process*. By comparing the states of a visualization over time, an analyst can reason about the analytic process to some extent, without needing the application to explicitly record the actions in that process. Capturing only the execution state of an application does not require any high-level conceptualization.

The low-level element *comparing historical states* is included in the problem characterization phase, as one of the component tasks into which financial analysis activities can be decomposed. This is because the choice of visualizations or interactive elements that accommodate showing more than one states simultaneously for the purpose of comparison is not implementation-dependent. However, to enable this comparison in practice, the implementation must still involve at least the capture of low-level application states.

The high-level element *documenting analytic process* is discussed in the implementation phase, because supporting this task involves more than a choice of visualizations or interaction techniques. Gotz and Zhou (2008) state that to implement a system that documents analytic processes, a taxonomy of analytic actions must be defined for the system to record. The analytic actions can be defined to different levels of abstraction. The more high-level the definitions, the more useful they are for the purpose of understanding analytic processes. However, for financial analysis activities, analytic actions may not only be domain-specific, but also be firm-specific, making taxonomy-development difficult and not generalizable. Because of this complexity, the capability to automatically record and replay analytic actions is a relatively new research direction in VA system development. Visual analysts still often document their reasoning process by annotating visualization states or journaling their analytic actions with pen and paper (Gotz and Zhou 2008). A hypothetical

VA application can support the documentation of analytic process by recording analytic actions to different levels of abstraction. At the lowest level of abstraction, the application will capture and time-stamp only the low-level execution events, which is equivalent to the minimum requirement to enable the comparison of historical states. Another method to record analytic process is to provide facilities for annotation and journaling.

Some of the first VA products that both keep track of analytic actions and support the comparison of visualization states were Harvest (Gotz and Zhou 2008) and CZSaw (Kadivar 2009; Chen et al. 2010). Harvest implemented a horizontal list of labels and icons to display analytic actions. Clicking on each of these actions reveals the corresponding visualization state. This interface is similar to the "history pallete" in the graphical editing program Adobe Photoshop, introduced in 1998. CZSaw implemented history view where analytic actions or visualization states are displayed as nodes on a directed network graph. The spatial layout of existing nodes and links in the diagram is kept consistent when a new state is added, so the user can follow the progression of the analytic steps.

Historical states can also be displayed side-by-side or super-imposed (layered) upon the current state on the user interface, using different visual encodings. This method has long been employed by non-VA applications, such as word processors (Microsoft Word, Open Office). Certain classes of visualization, such as scatterplots, stack graphs, candlestick charts, and line charts, can display multiple series of data, and hence also have the potential to display historical states super-imposed on the current state. Comparing history states side-by-side or via super-imposition requires two conditions:

- The visualization maintains a consistent spatial layout, such as a coordinate system, from state to state

- The visualization is not space filling, leaving room for displaying information other than the current state

The use of overview techniques to reduce the size of space-filling visualizations, such as treemaps, heatmaps, and geographic maps, can enable these visualizations to be compared side-by-side, such as in the tool Vistrails (Bavoil et al. 2005). Slideshow or more

sophisticated animations can be used to show the transition between states, such as in Tableau (Heer et al. 2008).

The level of support for documenting analytic provenance can also vary in terms of *persistence*. The terminology is borrowed from software design, which indicates whether data outlives one execution session of the software. If analytic actions and/or historical states are saved as persistent data, the analytic provenance can be traced from one session to the next. Historically, many software applications supported the automatic recording of states and/or actions for undo/redo purposes, but these recordings tend to be transient because of storage space constraints. Among VA applications, CZSaw supports automatic, persistent recording of analytic actions and procedural reconstruction of states, and Harvest supports manual saving of visualization states.

Finally, a practical concern in documenting analytic provenance is the coupling of raw data in the back-end of the application and the visualization state and analytic actions leading to that state. This is important because VA is not the only analytic method that may be used in a financial analysis problem. A real-world use-case may involve the chaining of multiple tools and analytic methods, with the VA application being only one part of the chain. That means the VA application must be able to export the raw data underlying a visualization state. This data export can also be either *persistent* or *transient* across session, and can be at different levels of abstraction.

Several other theories related to general analytics, such as structured analytics / intelligence analysis (Heuer, 1999, 2010) and sense making (Russell et al., 1993), inspired a few more dimensions of the functional evaluation framework: time frame of analysis, update frequency of data, and expected level of detail of value retrieval.

2.3 Designing an Interface for the Functional Evaluation Framework

For the functional evaluation framework to be useful, it must be represented in a format that is suitable for rapid and accurate referencing. In any use-case, VA experts must be able to rapidly remind themselves of the dimensions on which to evaluate the visualizable domain problem(s). After evaluating the domain problem(s) at the customer firm, they must be able to rapidly consult the framework to find out the visualizations and interaction design

elements suitable for the task. Then, they must be able to determine an implementation strategy, a technological pipeline that ends with either a visualization product, or a visualization API, that would accommodate the implementation constraints of their project.

The presentation format of the functional evaluation framework is summarized in Table 1. It consists of four tables representing the problem characterization dimensions, visualization design dimensions, interaction design dimensions, and implementation dimensions, respectively.

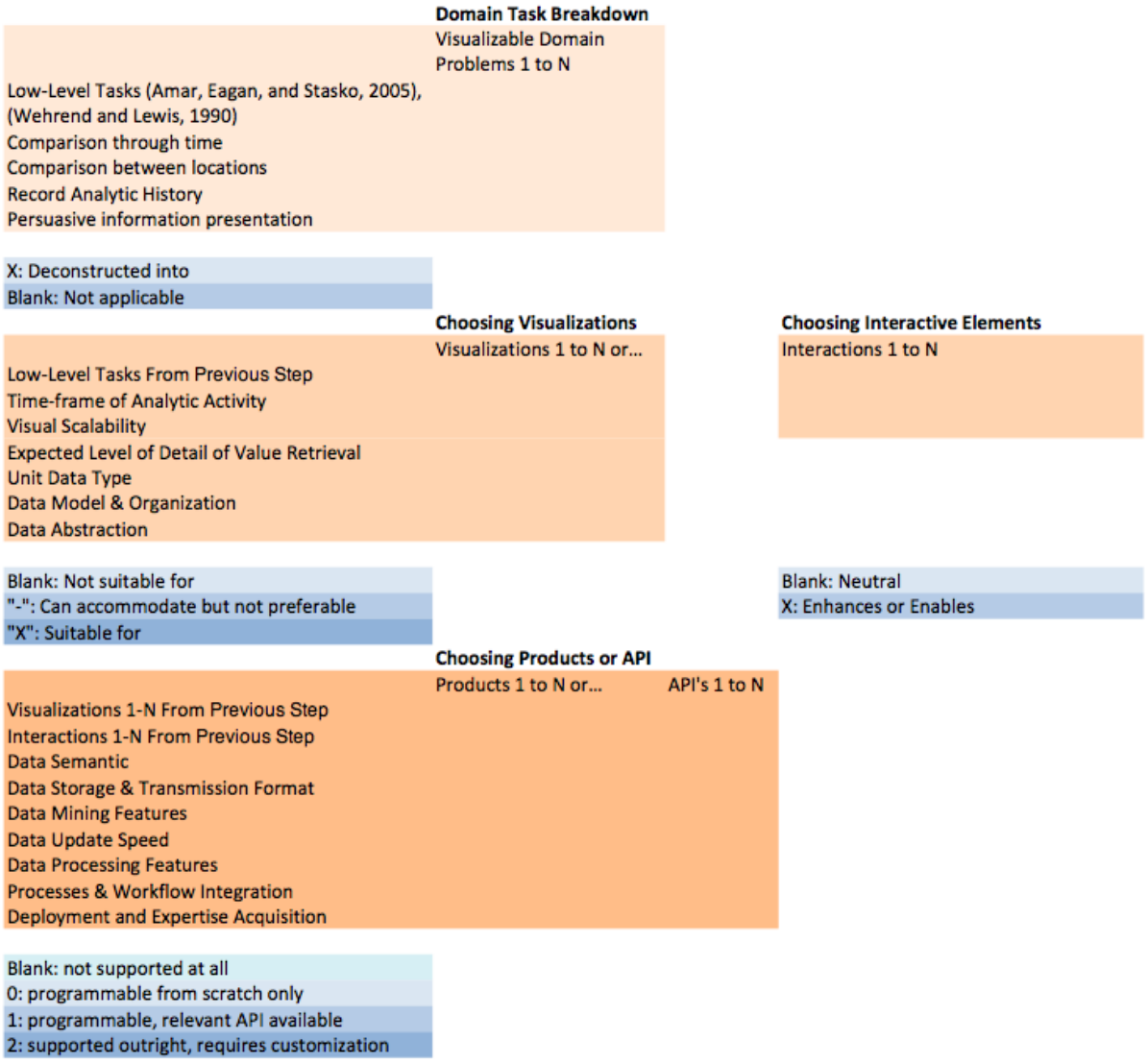
- The *Domain Task Breakdown* table maps visualizable domain problems in finance to low-level cognitive & perceptual tasks. These low-level tasks become input dimensions used in the design-phase tables.

- The *Choosing Visualizations* table maps the design-phase (implementation-independent) dimensions to many common types of visualizations.

- The *Choosing Interactive Elements* table maps implementation-independent (design-phase) dimensions to several common interaction features that can be incorporated into the VA application's user-interface or the visualizations themselves, in order to enhance the analytic potential of the visualizations or the usability/comfort of the interface.

- The *Choosing Products or API's* table maps the list of visualizations and interaction features, along with implementation-specific dimensions, such as data storage and transmission formats, and relative deployment time and costs (licensing, infrastructure, expertise acquisition and retention).

Table 1 High-level Representation of the Evaluation Framework



The design phase is represented with two tables to evaluate the visualizations and interactions separately. This is done because the semantics for evaluating visualizations and interactions are different, even if the dimensions are the same. A visualization can be “not suitable for”, “can accommodate but not preferable”, and “suitable for” each aspects of the design phase. An interaction, on the other hand, only either “enhances or enables” one of the aspects, or is “neutral”. Empty templates for all of these tables are in Appendix B.

2.4 Mapping Financial Analysis Tasks to Low-level Perceptual / Cognitive Tasks

The high-level financial analysis activities identified previously were then deconstructed, in order to separate the analytic tasks from the data. The data was further deconstructed to generalized aspects: semantic, data unit type, data model & organization (compound data type), data storage and transmission format.

The high-level tasks, detached from the data input, were then mapped to low-level perceptual / cognitive tasks. The mapping was done according to two major low-level "Perceptual & Cognitive" task frameworks:

- Amar, Eagan, and Stasko's (2005) model, with high-level tasks mapped to one-or-more of the low-level perceptual/cognitive tasks: *Retrieve Value, Filter, Compute Derived Value, Find Maxima and Minima, Sort, Determine Range, Characterize Distribution, Find Anomalies, Cluster, and Correlate.*

- Wehrend and Lewis's (1990) model—*identify, locate, distinguish, categorize, cluster, distribute, rank, compare within entities, compare between, relations, associate, correlate*.

These two task frameworks were reconciled by identifying tasks that have very similar definitions between the two works, and removing the near duplications. Tasks that are exclusive to each work were listed with their respective terminologies and definitions. The major source of differences between these two works is methodology. Wehrend and Lewis conducted a literature review, whereas Amar et al. conducted an experiment, having students generate data analysis questions given several data sets and a list of high-level analytic tasks.

These two works were used in conjunction because each work had its own weakness. The experiment of Amar et al. suffered from having a finite collection of analytic problems for the subjects to deconstruct. Furthermore, a limited set of analytic tools were available and familiar to the subject pool, so the problem deconstruction may have been biased toward analytic questions solvable by the given tools. The literature-research in Wehrend and Lewis' work produced a more extensive set of tasks, but there were also limitations, such as a lack

of consideration for relationships between data objects and the operations that can be done on compound data structures.

The “compute derived value” task was excluded from the model, because that is a calculation task that does not involve the visualization. It can be safely assumed that in most application design situations, such calculations would be in the back-end logic and not shown to the user (except on demand). “Sorting / Ranking” was also not listed in our model for the sake of simplicity, because it appears very rarely as a task unto itself, but usually appears only as an intermediate step to extrema finding (Amar, Eagan, Stasko, p. 114).

The “retrieve value” task was not listed as one of the low level tasks, but was turned to a design-phase criterion, defined in more details, and renamed as “Level-of-detail of value retrieval”. Value retrieval was defined as “finding attributes of a data case” in Amar et al.’s framework. However, with this general definition, this task seemed to appear as a component of many of the other tasks (Amar, Eagan, Stasko, p. 113) and most high-level problems would then involve value retrieval to some degree. Application designers need to know the necessary level of detail of value retrieval in an analytic task, because this can affect the choice of visualizations and interactive features.

The tasks in Wehrend and Lewis’s model were later refined in Zhou and Feiner’s 1998 Visual Task model, which discussed Wehrend and Lewis’s perceptual / cognitive tasks in terms of visual accomplishments, providing an even lower level of abstraction. This more detailed low-level task model would have been very useful in guiding the design process if high-level analytic tasks could readily be deconstructed into simple visual operations. In practice, deconstructing domain-specific activities into data and high-level analytic tasks, then into cognitive / perceptual tasks, then finally into individual visual operations that contribute to each cognitive / perceptual task would be very time consuming. The extra level of deconstruction is also difficult to justify in terms of usefulness to the design process, because analysis tasks can involve more than just visual operations. For example, the act of querying can be supported by a search box instead of a visualization that affords detail-on-demand.

Another alternative approach to breaking down analytic tasks to simpler component tasks is to translate the high-level domain-specific tasks to generalized high-levels, using models such as Amar and Stasko's (2004) or Neumann's (2007). However, it seemed more practical not to employ this intermediate step because of the current focus of information visualization (InfoVis) on information representation. InfoVis literature usually discusses how particular visualizations or interaction techniques leverage human perceptual and cognitive processes, not how effective they are in supporting generalized high-level analytic tasks. Furthermore, it is considerably more difficult conceptually to evaluate the suitability of a visualization for high-level, abstractly-defined tasks. Experiments involving high-level, abstractly-defined tasks are often difficult to design and very time-consuming to carry out in practice.

Two additional low-level tasks were added based on a review of existing financial analytics systems: comparison through time and comparison between locations. Extracting tasks from the features of existing systems is similar to the method employed by Shneiderman to construct a task taxonomy by data-type (1996).

Table 2 shows the final list of high-level activity in financial analysis and portfolio management and the final list of low-level perceptual and cognitive tasks. This is the *Domain Task Breakdown* table in the framework.

Table 2 Mapping of High-level Tasks to Low-Level Cognitive / Perception Tasks

X: Deconstructed into Blank: Not applicable	**Conceptual, qualitative analysis of prospect**	Earning and price forecast, fundamental	**Earning and price forecast, semi-automated, technical**	Earning and price forecast, automated, technical	**Relationship management (client & peers)**	Portfolio Monitoring	**Scenario Analysis**
Filter / Locate	X	X	X		X	X	X
Compute Derived Value		X	X			X	X
Find Extrema			X				
Sort / Rank			X		X	X	
Determine Range		X					
Characterize Distribution	X	X				X	
Find Anomalies	X		X			X	X
Cluster / Associate	X	X					
Correlate	X	X			X		
Compare through time	X	X	X		X	X	
Compare between locations	X				X	X	
Compare historical states	X	X	X	X			X
Identify							
Compare between entities	X	X			X	X	X
Categorize	X	X			X		

This mapping of several high-level activities was meant as a starting point for the VA application designer in requirement elicitation. The VA application designer will need to gather more detailed and organization specific information from the client. Furthermore, in practice, the precise definition of each high-level analytic task often contains specific variations between firms and even between analysts. An empty table of low-level cognitive and perceptual tasks (Table 3) can also be shown to the analyst at the target firm during design, so that the analyst can help specify the list of high-level analytic tasks and break down these tasks from a firm-specific point of view.

Table 3 Template for Deconstructing Domain-specific and Firm-specific Analytic Tasks into Atomic Cognitive and Perceptual Tasks

	Domain or Firm-specific Task 1	Domain or Firm-specific Task 2
Filter / Locate		
Compute Derived Value		
Find Extrema		
Sort / Rank		
Determine Range		
Characterize Distribution		
Find Anomalies		
Cluster / Associate		
Correlate		
Compare through time		
Compare between locations		
Compare Historical States		
Identify		
Compare between entities		
Categorize		

Legend:

X: Deconstructed into
Blank: Not applicable

Although the framework is represented as a list of tables, the VA expert does not necessarily have to use these tables exclusively, but can acquire the same project requirements via other methods, such as interviews, observations, surveys, etc. Appendix A.2 and A.3 show a sample questionnaire and some sample interview questions that were used in a VA application development case study. Those questions did not directly reference the tables in the framework, but still collected much of the information needed for the tables. Over the course of the semi-structured interviews in that case study, the application designer collected design requirements based on the dimensions defined in this framework.

3 Design-phase Dimensions in Detail

This section describes design issues that are relevant to visualization and interaction design, but are not dependent on any particular technology pipeline or product. These dimensions are then used to evaluate a selection of visualization and interaction techniques.

3.1 Time Frame of Analytic Activity / Data Update Speed

The time frame of analytic activities is important both in the design and implementation phases. This dimension is cross-listed in the implementation-phase table as *Data Update Speed*.

Data Update Speed in the implementation-phase table does not use the same scale as *Activity Time Frame*, but a more implementation-centric scale: *manual* (batch of data must be manually fed to the tool), *pull update* (automatic, but with a time interval), *push update* (automatic, real-time, limited only by network speed). Data update is important in implementation, because different products support different forms of data input, with their own limitations, including update time. A limitation on data update speed places a constraint on the design of VA applications, regardless of the visualizations chosen.

Time frame of activity is also important to the design phase, because some visualization transitions update a change in data faster and more faithfully than others. Most visualizations that map raw data or extractions (values derived from calculation) of raw data directly to visual elements can be transitioned very quickly as the data changes. Visualization that shows an abstraction of the data may not reflect the changes in the raw data as rapidly. The graphical design of the visualization also affects the transition speed. Reordering a cluster or network visualization will take time. Even more complicated charts will transition more slowly than a simple line chart. Furthermore, visual encoding and spatial layout affect whether a visualization can be compared between one state to the next in the under-one-minute term. This is because subtle changes are not salient on visualizations where

- Past states are not shown persistently along with the current state, and
- Spatial layout of visual elements are not guaranteed between state changes

In reality, the time frame can be either minute-term (minutes to seconds from the beginning of analysis to decision making), day-term, month-term, year-term, and long-term (multiple years to decades). Day-traders typically use technical analysis aided by human visual cognition, such as chart patterns, to make decisions in the day-term. Sometimes, this process is aided by quantitative models that would issue automatic action alerts or can be programmed to react to certain conditions, pushing the decision making time-frame to the seconds. Financial analysts, fund / portfolio managers, swing traders, and arbitrageurs usually operate in medium terms, from a month to 1, 2 years. Long-term investors sometimes operate on a multi-year to multi-decade timeframe. For the purpose of visualization design, only the capability of *real-time transition* and *rapid state comparison* is significant, because most visualizations can be generated and interpreted rapidly enough to accommodate all the longer time horizons. There is also a class of traders who use automated algorithms to trade entirely without human decision-making, making multiple trades per second. This class of High-Frequency-Trader, and the under-a-second timeframe, is beyond the scope of this analysis.

3.2 Visual Scalability

Screen real-estate usage is a design-phase issue that affects the choice of visualization and interaction techniques. Zooming and Panning, Drill-down (semantic zooming), Focus / Context, and Overview / Details are interaction techniques that can enhance any visualization's capability to display more data.

However, each visualization can only display a finite number of data points simultaneously before the data points becomes indistinguishable or cluttered on screen. Choosing a data point to display detail-on-demand also becomes very difficult with a cursor, when data points are very close together and only a few pixels in size. Screen real estate is a very finite resource, especially considering that any non-trivial VA applications will contain extra interface elements as well, forming a dashboard. The visual scalability of visualization techniques will be evaluated on the scale:

- (Capable of displaying) 1-100 data points

- Hundreds of data points
- Thousands of data points

This scale is applied before any interaction techniques, such as graphical zooming and panning, semantic zooming, focus/context, and overview/detail, is used, which could enable a visualization technique to accommodate many more data items than its basic form. The maximum range of “thousands of data points” was chosen after an analysis of the screen space usage of a hypothetical visualization technique trying to represent 10000 data points. This analysis assumed a screen size of 1440x900, a common resolution of wide-screen monitors, and a visualization size of 1024x768, reserving 1/3 of the screen space for other interface elements in an application. With 10000 data points, the visualization can reserve on average about 70 pixels, enough to barely contain one size-10 letter, to display a data point. The allowable screen space per data point becomes even smaller if the hypothetical visualization technique requires white space, connectors between data points, text labels, or using the size of the data point for visual encoding. While it is possible (with a heatmap for example) to display more than 10000 data points in a 1024x768 screen space, the resulting visualization would be very cluttered and difficult to navigate. Instead, it is more advisable to limit the number of data points displayed on a visualization at any one time, and to use one or more interaction techniques (especially overview/detail and focus/context) to enhance the capabilities of the visualization.

3.3 Expected Level of Detail of Value Retrieval

Financial decision-makers often face a trade-off of time versus precision. Reducing details in a representation can allow an analyst to process more information in a given time without getting overloaded. If a decision can be made without the need of a precise calculation, then the interface should present a low level of detail fire (a.k.a. an overview), allow for progressive filtering and zooming, and present the highest level of detail, usually a large collection of numbers or unstructured text, on demand (Shneiderman, 1996). The expected level of details can be classified as:

- Raw value shown on the interface: whether the exact number or text needs to be shown on the user interface or visualization to support the analytic activity
- Quantitative estimate: whether the visualization chosen must support the formulation of accurate, if not necessarily precise, estimates of underlying values
- Qualitative estimates: whether only estimates of the qualitative attributes of the underlying values are necessary to adequately solve the analytic task

3.4 Unit Data Type

This design-phase dimension represents the simplest units of data that can be used as building-blocks for more complex data types in real-world problems. The choice of unit types was proposed with inspirations from mathematics for the numeric types (Levy 1979), computer programming for the nominal / enumeration type (Kernighan and Ritchie 1988), and a survey of features in existing text analytics products (Inspire, Starlight, ManyEyes, etc.) for textual data.

- Cardinal (e.g. 1, 2, 3): representing order and distance, with equal distance between possible values.
- Ordinal (e.g. first, second; Agree, Mildly Agree, Disagree): representing order with no implication of distance. The order may also be of a qualitative and non-numeric nature.

- Interval (e.g. 1.0 to 2.5): representing order and distance, but with no implication of uniform precision for the distance. An interval is defined as a range of continuous values, the distance between two adjacent values are not guaranteed to be the same. Interval data has an implication of distance between values, while ordinal data is a ranking with no implication of distance.

- Nominal (e.g. RRSP, RESP, LIRA, SP-RSP, NR. etc.): discrete and finite entities in a set, without any implication of order or distance. Nominal data is often non-numeric, each entity having a conceptual, rather than numerical, relationship with others in the set. This is equivalent to the data type "enumeration" in software programming.

- Machine-Parsable "Structured" Text (e.g.: tags and keywords metadata): text data presented in a regular structure, which can be parsed by machines

- Unstructured Text (e.g.: news articles from multiple sources, emails, IM, web pages): text data without a regular structure, which is very difficult to parse by a machine, and very time consuming to analyze manually. The degree of structure in "unstructured text" may vary.

3.5 Data Model and Organization

This dimension represents the organization of complex data in the VA task, without considering the implementation approach. The choice of data models in this dimension was drawn from common database designs (Beynon-Davies 2004) except for bi-variate, unstructured collection, and OLAP, which were proposed as special cases.

Unstructured collection (file system) was included since it is the most basic form of data organization. Bi-variate is a simple special case of multi-variate, a table with one column. This was considered separately from multi-variate data, as it can be visualized with simpler but more familiar charting techniques. Financial data often appear in one of these organization types:

- Flat (Table-like, aka. Multi-variate)

- Hierarchical (Tree-like)

- Network
- Bi-variate (aka. Table of one column)
- Unstructured collection (file system)
- Relational
- Multi-dimensional Schema (e.g. OLAP)

Multi-dimensional schema and *multi-variate* data are often used interchangeably in popular communication, but they are distinguished in a subtle way by the formal definitions from scientific visualization (Dos Santos and Brodlie, 2004). Each dimension of multi-dimensional data can have different unit data type and value spaces, while different variates in multi-variate data tend to have the same value space.

The implication for visualization is subtle as well. While it is definitely possible to visualize multi-dimensional data with parallel coordinate charts, node-link diagram resembling parallel coordinates (if the value space is non-numeric), or a matrix of visualizations, all of these approaches do not encode the possible range of values for each dimension. Nevertheless, since the value space is often familiar to the domain experts who would be interpreting the visualizations, this small limitation typically does not hinder usage. However, multi-dimensional and multi-variate data must still be distinguished from each other, because the dimension in multi-dimensional data can also contain one or more hierarchical structures (Kimball and Ralph 2008), which will require a completely different visualization design than multi-variate data.

Relational and multi-dimensional data are special cases that are presented as implementation-phase concerns. Conceptually, both of these data models are not implementation-specific, and should be considered in the design phase along with the other types of data models. However, these data models can be broken-down into flat and / or hierarchical and / or network data.

In the design phase, the VA application designer can determine the sub-model of any relational and multi-dimensional datasets, and break them down to simpler data models for which it is easier to design visualizations. In the implementation phase, the VA application designer can then consider whether the technology pipeline of choice is capable of importing and manipulating relational or OLAP file or database formats. Row-oriented and column-oriented databases and OLAP cubes are represented as data formats in the implementation-phase table.

Research on the visualization of databases and data cubes also tends to take the direction of dimensional reduction: providing interactions for querying and visualizing subsets of the data model, using SQL for relational data base or MDX (Microsoft, 1997) and XMLA (Microsoft and Hyperion, 2001) for data cubes. The subsets that result from navigational operations, like *drill-down, roll-up, slice,* and *dice,* now take the forms of flat or hierarchical data, for which there are many visualizations.

The two field-defining works on visualization of OLAP since its introduction are Gebhardt, Jarke, and Jacobs (1997) and Microsoft Corp (1998), recent works are represented by Maniatis et al. (2003), Techapichetvanich and Datta (2005), Mansmann (2007), and Ieronutti and Pighin (2009).

Apart from Ieronutti and Pighin's system, none of these works attempted to create a visualization of the entire structure of an OLAP data cube, choosing instead to break the OLAP data via navigational operations down to the constituent flat, hierarchical, or network data. Ieronutti and Pighin use 3D cubes each representing three data dimensions and one numeric measure encoded by the location of the cube in 3D space, and multiple cubes can be combined into multi-dimensional OLAP structures. However, the effective exploration of this 3D representation requires interaction techniques such as point-and-click selection, panning, rotation and zooming, and details-on-demand.

3.6 Evaluating a Sample Set of Visualizations and Interaction Techniques

After the design-phase dimensions for evaluating visualizations and interaction techniques were defined, an evaluation of a sample of common visualizations and interaction techniques was undertaken. The evaluation began with comprehensive but high-level survey literature. A discussion of many newer visualization techniques and application design studies was found in (Heer et al. 2010) and Munzner (2009). The works of Card et al. (1999), and Cleveland et al. (1984) offered detailed analyses of many traditional graph visualizations and discussed the principles of perception and cognition underlying visual encodings and interactions.

Several finance-specific case studies on the design of VA applications were then consulted. These provide lessons on problem deconstruction in finance, as well as visualization and interaction choices. They include the work of Jungmeister and Truro (1992) and Wattenberg (1999) on applying treemaps to stockmarket data, and Wright's (1995) work on creating guidelines for applying 3D graphics and animations to market visualizations. In 2001, Girardin and Brodbeck presented a dashboard with linked views, geocoding, thematic clustering, and parallel coordinates on price and salary data. In 2002, Dwyer and Eades proposed to visualize the changes in size of nodes on a network graph by using the third dimension to represent time. Ziegler, Nieztchmann, and Keim (2008) described the use of a heat matrix to show the expected return of a security based on buying time and holding period, as well as the use of graphical filters to highlight regions of interest on the matrix. Roberts (2003) attempted to deconstruct the act of equity trading into simpler component tasks and to create a visualization of stock-tick data that used the third dimension to encode time and volume. Savikhin et al. (2008) conducted an experiment on economic scenario analysis task, where an interactive visualization using colour-coded line graphs and direct manipulation of a data field. Sawant (2009) described the process of formatting stock price data for importing to visualization tools and recommended visual encodings for the various dimensions in the price data. Alsakran, Zhao and Zhao (2010) demonstrated a variant of parallel-coordinate plots in which the plotting area is divided into rectangular tiles containing the density of plot lines, encoded with colour and opacity. By enlarging the tiles and effectively lowering the resolution of the plot, an overview can be created that emphasizes

only major trends. This reduces visual cluttering, and allows the parallel-coordinate plot to effectively display up to thousands of data items. Kubelec and Sa (2010) constructed a dataset of the financial relationships between countries, represented by the ownership of foreign assets and liabilities. This dataset was described structurally as a network, where nodes represent countries and links represent bilateral financial assets. This data was then visualized as a node-link diagram using the Pajek toolkit.

Finally, a number of single-technique papers, design case studies, and technique implementations in a number of existing products and toolkits were also consulted.

3.6.1 Visualizations Considered

- Graphs visualizations:
    - Parallel Coordinate (Wegman 1990, Inselberg 1990)
    - Scatter Plot, Dot Plot, Line Graph
    - Stack Graph, Bar Graph, Bullet Graph (Few 2006)
    - Candlestick and variants like Candletrend and Heikin Ashi (Nison 2001).
- Space-Filling visualizations:
    - Heatmap, Treemap (Shneiderman, 1991; Jungmeister and Truro 1992, Wattenberg, 1999)
    - Ringmap, Sunburst, Interring (Yang et al. 2003; Bostock 2011)
    - Horizon Graph (Few 2006), Pie Charts
- Network visualizations: Adjacency Diagrams (Holten 2006), Node-link Diagrams
- Temporal / Time Series (Aigner 2008, Zhao et al. 2011)
- Spatial / Geographic Coding (Fisher 2007, Wood et al. 2007, MacEachren 1998)

- Unstructured Text Clustering: In-spire galaxy & theme view, Starlight topic view, etc. (Risch et al. 2008, Pacific Northwest National Laboratory 2011, Future Point Systems 2011)

- Space-saving techniques:

    - Glyphs / Scented Widgets / Microcharts (Willet et al. 2007)

    - Matrix of visualization instances, commonly used to aggregate small overviews of multiple scatterplots and heatmaps (Tableau Software 2011, Roberts 2007, Weaver 2004)

- Presentation enhancement: 3D Views (Quantum4D 2011, Wong et al. 1997)

Finally, the special case of Tables of Values was evaluated along side the visual data representations. The comparison of the suitability of a table of raw data compared to different types of visualizations would be very helpful to a VA application designer in practice.

3.6.2 Interaction Techniques Considered

- Search / Filter (Ahlberg 1994)

- Cursor-based Picking and Manipulation (Chuah 1995)

- Semantic Zooming (Appert and Fekete 2006)

- Graphical Panning & Zooming, Overview / Detail, Focus / Context (Cockburn 2008, Rao and Card 1994)

- Linking / Brushing in Multiple Views (Plumlee and Ware 2006)

- Pop-up & Tooltip on visual elements (Microsoft 2011, Raskin 2000)

- Animation / Slideshow (Heer and Robertson 2007, Tversky et al. 2002)

- Direct manipulation of data

These two lists of techniques constitute a relatively broad representation of the literature and product space, but are not exhaustive. Similar to previous taxonomies of visualizations that used more than one high-level dimension, a limited selection of techniques were evaluated due to time constraint. The literature survey will always be a work-in-progress, because the body of knowledge is always growing. There are also many techniques demonstrating slight variations of interactions or visual encodings from the classes of techniques mentioned in this work. And finally, there are techniques implemented as experimental features in commercial products that have little documentation or appear in the scholarly literature, such as the “Monkeybar” scatterplot variant (ThinkorSwim 2011).

Derivative visualization types can usually be recreated by customizing the base visualizations and the mapping of data to visual elements. Using visualization API’s in the implementation phase, instead of pre-packaged visualization products, is the solution that would provide the most flexibility in customization. However, even closed-sourced toolkits and products often provide some degree of customization to the base types, so that subtle changes can be made. For example, the Candlestick chart was listed once in the design-phase tables, because Candletrend and Heikin Ashi are usually also available or easily recreated by modifying the Candlestick.

Table 4 and 5 show design-phase dimensions, applied to this representative list of visualization and interactions.

Table 4 Design-phase Dimensions – Choosing Visualizations

Blank: Not suitable for
"-": Can accommodate but not preferable
"X": Suitable for

		Parallel Coordinates	Scatterplot	Dot Plot	Bullet Graph	Line Graphs	Stack Graphs	Bar Graphs	Candlestick (a form of glyph)	Heatmap	Treemap	RSF (ringmap, sunburst)	Horizon Graph	Pie Charts	Adjacency Diagrams	Node-link	Time Series	Geocoding	Unstructured Text Clustering	Glyphs (whiskers, microchart)	Tables of values
Low-Level Tasks From Table 1	Filter / Locate	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-		-
	Find Extrema		X	X	X	X	X	X	X	-	-	-		X							-
	Determine Range		X	X	X	X	X	X	X				-			-	-				
	Characterize Distribution	X	X	X	X	X	X	X		-	-	-	X	X				X	X		
	Find Anomalies	X	X	X	X	X	X	X	X	X	-	-					X	X		X	
	Cluster / Associate	X	X	X						-	-	-			-	X			X		
	Correlate	X	X	X	X	X	X	X													
	Compare through time			-	-	-	-	-	-				X			-	X				-
	Compare between locations			-	-	-	-	-		-	-	-	X		X			X			-
	Compare historical states		-	-		-	-		-				-			-	-				
	Identify																				-
	Compare between entities	X	X	X	X	X	X	X	X	X	X	X	X	X	X		X	X		X	-
	Categorize	X		X		X	X	-			X	X			X						
Time-frame of Analytic Activity	Real-time transition		X	X		X		X	X	X			-				X	-			
	Rapid state comparison	X			X	X	X	X									X			X	
Visual Scalability	1-100 data point	X	-	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	-	-	-
	Hundreds of data points		X			X			X	X	-	-	X				X	X	X	X	-
	Thousands of data points		X															X	X	X	
Level of Detail of Value Retrieval	Raw values shown				-	-	-	-			-	-		X	-						X
	Quantitative estimates	X	X	X	X	X	X	X	X				-	X			X	X		-	
	Qualitative estimates				X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	
Unit Data Type	Cardinal	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X		X	X
	Ordinal	X	-	X	-	-	-	-		X											X
	Interval				X		X		X												X
	Nominal			X		-					X	X	-	X	X	X					X
	Structured Text										X	X			X	X			X		X
	Unstructured Text																		X		X
Data Model & Organization	Bi-variate	X	X	X	X	X	X	X	X	X	-		X	X			X	X	X	X	X
	Hierarchical (Tree-like)									-	X	X			X	X					-
	Network														X	X					
	Multi-variate (Flat, Table-like)	X							-				-							X	-
	Unstructured Collection (Files)																		X		-

Table 5 Design-phase Dimensions – Choosing Interactions

Blank: Neutral
X: Enhances or Enables

		Search and Highlight	Cursor Manipulation (Picking)	**Semantic Zooming**	Graphical Panning & Zooming	**Overview & Detail**	Focus & Context	**Linking & Brushing (Multiple Views)**	Pop-up & Tooltip	**Direct Manipulation of Data**	3D View (Rotate, Zoom, Fly-by)	**Matrix of visualizations**	Slideshow / Animation
Low-Level Tasks From Table 1	Filter / Locate	X	X	X	X	X	X		X				
	Find Extrema										X		
	Determine Range	X						X					
	Characterize Distribution	X						X			X		
	Find Anomalies			X	X	X	X						
	Cluster / Associate	X						X					
	Correlate							X		X		X	X
	Compare through time												X
	Compare between locations												X
	Compare historical states												X
	Identify	X	X						X				
	Compare between entities									X		X	X
	Categorize												
Time-frame of Analytic Activity	Real-time transition										X		
	Rapid state comparison					X	X						
Visual Scalability	1-100 entities	X	X						X	X	X		
	Hundreds of entities	X		X	X	X	X	X	X			X	
	Thousands of entities	X		X	X	X	X	X	X			X	

4 Implementation-phase Dimensions in Detail

This section describes issues in VA Application development that directly affect the choice of technology pipelines (visualization toolkits and supporting programming libraries) or existing visualization products. A sample selection of products and toolkits are then evaluated in terms of these dimensions at the end of the section.

4.1 Data Semantics

Data Semantics is represented in the implementation phase of the framework, affecting the final selection of visualization products or API. This dimension represents the degree of specialization of the product. Data semantic is reflected in the way the products are built, such as the incorporation of finance terminology into the UI or the visualization, or whether some visualizations have been customized for common classes of visualizable problems in finance. However, even a general product can still be useful in finance if it contains features that do not have equivalents in the designed-for-finance products. A product can be:

- Domain-agnostic
- Designed for finance
- Designed for a sub-domain in finance

4.2 Data Storage and Transmission Format

This implementation-phase dimension represents data-related features. These are concerns of whether a product or technology pipeline is ready to import and pre-process data formats commonly used in financial analysis, including:

- Built-in data (a special case indicating a useful, production-quality product that was purposely designed with its own data input, which also cannot be changed)
- VA-product-vendor specific
- Excel, Microsoft specific

- Delimited value spreadsheets (CSV, Tab-delimited, etc.)
- Word, Microsoft specific
- PDF, Adobe specific
- XML, user-customizable schema
- XML, product specific schema
- XBRL (extensible business reporting language)
- JSON
- Row-oriented Database (Oracle, MS SQL, etc.)
- Column-oriented Database (RPAS, Sybase IQ, etc.)
- Java Message Service (ActiveMQ, SonicMQ, etc.)
- Advanced Message Queuing Protocol (QPID, etc.)
- Customizable text format (is a customizable parser available? Is it trivial to write a parser in case of an API and technology pipeline instead of a product?)

4.3 Data Mining Features

- RSS feed
- Email server retrieval
- Web crawler retrieval
- Search engine retrieval

4.4 Analytic Process Documentation

- Recording Program Events

- Recording Analytic Actions
- Journaling and Annotation Features Available
- Analytic Process Information Persists Across Sessions

4.5 Data History and Export

- Data Underlying Historical States Persists Across Sessions
- Export preserving file format and schema of the input data. Ideally, this mode of export should be provided along one or more of the other modes, because preserving the input format and schema facilitates passing the output data of the VA application into the other tools in an analytic workflow.
- Export preserving only external (domain/firm-specific) semantics but using application-specific file format and schema
- Export containing only application-specific semantics (e.g. a memory dump that contains application variables but not domain/firm-specific terms)
- Export containing both external and VA application semantics, allowing the analyst to integrate the VA application into an existing analytic workflow.

4.6 Data Processing Features

- Clean & normalize heterogeneous or bad data
- Entity selection
- Entity extraction
- Sentiment Analysis
- Statistical Analysis
- Acceptable performance and stability (time and memory requirement) for at least 10000 records

The benchmark of 10000 records was chosen based on an examination of documentation for a number of visualization products and toolkits in both the commercial and open-source space. It is also based on the author's experience with using a number of visualization products and programming with several toolkits within the last two years. The criterion for choosing the benchmark is whether a visualization product is responsive enough to provide acceptable user experience. Some products claim a very high upper limit on data scalability, to the millions of records, but this may only be the limit at which the software can import the data without crashing. The tolerance for acceptable responsiveness tends to be much lower. This benchmark will need to be updated over time, as new products and improved versions of existing products begin to pass the benchmark comfortably.

4.7 Processes and Workflow Integration Constraints

This implementation-phase dimension includes issues regarding the integration of the new product or custom VA tool into the daily lives of the actors, such as specialization of user interface and data semantic, and interoperability with popular vendors. This includes creating a unified dashboard that contains at least all the visualizations and controls for interacting with them, embedding the VA interface in existing software UI.

- Dashboard building
- Excel UI integration
- Online sharing of visualizations
- Exporting & printing visualizations

Receiving data feeds from specialized data formats or vendors in finance is also an important dimension in the implementation phase. Some application can be made or broken by this interoperation with data sources that the financial institutions use.

Some of the most common financial data vendors are Bloomberg, Thompson Reuters, CarryQuote, Factset, Interactive Data, Markit, Moody's Analytics, Standard & Poor's, SIX Telekurs, SNL Financial, SunGard, etc. Some general-purpose data vendors can also be very useful to financial analysis, due to their unique data offerings, such as Google Analytics (web

browsing and email statistics, data mining of emails) or Facebook (social networking data, user metadata). Specialized data vendors were excluded from the scope of the first release of this functional evaluation framework, due to the expense and communication challenges in evaluating the technological aspects of the data services that these firms provide.

4.8 Deployment and Acquiring / Retaining Expertise

The implementation-phase table also includes practical issues such as:

- Whether the product is available. Several research products were evaluated for education and completeness purposes, but these products are not yet available for use
- Whether the product or API is free-to-use, free for educational use, or wholly commercialized
- Free technical support / documentation available
- Commercialized technical support / documentation available
- Relative helpfulness of documentation. This is the subjective opinion of product evaluators (the author and subsequent researchers), who ideally should be scholars and/or practitioners in VA.

4.9 Evaluating a Sample Set of Visual Analysis Products and API's

In a VA application design project, the VA designer will typically choose a small subset of visualizations, to observe the principle of minimalism, while incorporating a number of the typical interaction techniques listed above. Therefore, in choosing a final product or API from which to build the organization-specific VA application, the VA designer typically only needs support for a few visualizations that would adequately encode all the necessary information and afford the low-level tasks (Klein 2002), but would like to have enough alternative interaction features in order to provide more flexibility and richness in user experience design.

Apart from the visualizations and interaction techniques available, the sample set of VA products was also evaluated on the aforementioned implementation-phase dimensions. The

evaluation process involved a review of any product literature available, such as manuals and whitepapers, and a hands-on verification of the features in a trial evaluation whenever that option was available.

The list of products evaluated was compiled from a comprehensive search of the product space using Internet search engines, skipping over products and API's that appear to be at an early development stage, or had been abandoned. The products encountered during the InfoVis literature review were also featured in this evaluation.

Even though most of the academic projects found have not been commercialized or publicly released (so that they cannot actually be adapted for use in a firm yet), they represent the state-of-the-art in terms of methodology and technology, and so are relevant to this taxonomic construction.

Some of the products that were evaluated are visualization API's instead of complete toolkits. Most visualization API's found do not account for other important implementation-phase concerns like data format import. To evaluate visualization API's for practical usage, mature data-import/export libraries in the same language & platform as the visualization API being evaluated was also considered in order to verify whether the auxiliary features could be implemented separately with some programming.

Although only a small sample of products and API's were evaluated here for the purpose of illustrating the framework, a list of over thirty products and API's that deserve critical evaluation was compiled:

- Free-to-use: VTK (C++ API), IVTK, Prefuse, Improvise (Java API), D3JS, Protovis, JIT (Javascript API), Flare (Flash API), Orange, Ggobi, ParallCoord, Xmdvtool, Gapminder, Vis-stamp, Wirevis, Smart Money, Finviz, PortfolioCompare, Datagraph, Tulip

- Commercialized: Tableau, Panopticon, Xcelsius (SAP Crystal Dashboard Design), Quantum4D, In-spire, Starlight, Geotime, Palantir, Opentext (Nstein), SAP Text Analytics, Microsoft Business Intelligence Crescent, Centrifuge, and Excel

Several products that are neither VA-specialized nor finance-specialized were included in the analysis. Microsoft Excel was included due to its ubiquity in the financial industry. For a given financial analysis problem, a comparison between the analysis process using Excel and the analysis process aided by a visualization system—one or more visualization(s) augmented with one or more interaction(s)—is helpful for VA application designers. In a practical design situation, it is likely that the VA application designer would need to explain why, and in what condition, a visualization-based analytic process would be superior to browsing and manipulating the raw data in Excel. Table 6 below shows the implementation-phase dimensions and the sample evaluation of products.

Table 6 Implementation-phase Dimensions, as Applied to Several API's and Products

Blank: not supported at all
0: programmable from scratch only
1: programmable, relevant API available
2: supported outright, requires customization

		D3js / Protovis	Panopticon	In-Spire
Visualizations 1-N From Table 2A	Parallel Coordinates	1		
	Scatterplot	1	2	
	Dot Plot	1	2	
	Bullet Graph	2	2	
	Line Graphs	2	2	
	Stack Graphs	2	2	
	Bar Graphs	2	2	
	Candlestick	1		
	Heatmap	1	2	
	Treemap	2	2	
	RSF (ringmap, sunburst)	1		
	Horizon Graph	1	2	
	Pie Charts	2	2	
	Adjacency Diagrams	1		
	Node-link	2		
	Time Series	1	2	2
	Geocoding (programmable with GDAL)	1		
	Unstructured Text Clustering			2
	Glyphs	0	2	
	Tables of Values			
Interactions 1-N From Table 2B	Search and Highlight	1	2	2
	Cursor Highlight (Picking)	1	2	2
	Semantic Zooming	1	2	
	Graphical Panning & Zooming	1		2
	Overview & Detail	0		
	Focus & Context	0		
	Linking & Brushing	1	2	2
	Pop-up & Tooltip (on the visualizations)	1	2	
	Direct Manipulation of Data	1		2
	3D View (rotate, zoom, fly-by)			2
	Matrix of visualization instances	2	2	
Data Semantic	Domain-agnostic	2	2	2
	Designed for finance	1	2	
	Designed for a sub-domain in finance	1	1	
Data Storage & Transmission Format	Built-in, unmodifiable data	2		
	VA-product-vendor specific			
	Excel, Microsoft specific	0	2	
	Delimited value spreadsheets (CSV, etc.)	1	2	2
	Word, Microsoft specific			2
	PDF, Adobe specific			2
	XML, user-customizable schema	1		
	XML, product specific schema	1		2
	XBRL	1		
	Row-oriented Database (Oracle, etc.)	1	2	
	Column-oriented Database (Sybase IQ, etc.)	1	2	
	OLAP	1	2	
	JSON	2		
	Customizable text format	1		2
	Java Message Service (ActiveMQ, SonicMQ)		2	
	Advanced Message Queuing Protocol (QPID)		2	
Data Mining Features	RSS feed	1		2
	Email server retrieval	0		
	Web crawler retrieval	0		
	Search engine retrieval	0		2
Data Update Speed	Manual Input	2	2	2
	Pull Update (interval refresh)	1		
	Push Update (best effort "true" real-time)	1	2	
Data Processing Features	Clean & normalize heterogeneous or bad data	0		2
	Entity selection			
	Entity extraction			2
	Sentiment Analysis			
	Statistical Analysis			
	Scalable to 10000+ records		2	2
Data Export and History	Preserving format and schema of input data	0	2	
	Using application-specific format and schema	0	2	2
	Contain both external and app-specific semantics	0		
	Contain only application-specific semantics	0		2
	Historical Data Persists Across Sessions	1		
Analytic Process Documentation	Recording Program Events	0	2	
	Recording Analytic Actions	0		
	Journaling and Annotation Features Present	0		
	Analytic Process Info Persists Across Sessions	1		
Processes & Workflow Integration	Dashboard building	0	2	
	Excel UI integration			
	Online sharing of visualizations	2	2	
	Exporting & printing visualizations	2	2	2
Deployment & Expertise Acquisition	Availability (1: Yes, 0: No)	1	1	1
	Licensing (1: Free to use, 2: Commercialized, 3: Free for Educational Use)	1	2	2
	Tech support and documentation (0: None, 1: Free, 2: Paid, 3: Both)	1	2	3
	Documentation Helpful? (1: Not at all helpful - to - 5: Extremely Helpful)	1	3	4

5 Design Case Study of a Visual Analysis Application in a Boutique Asset Management Firm

To demonstrate a domain-specific approach to VA system design, and the application of the functional evaluation framework of visualizations and products described above, it will be useful to describe the entire development process of a visual analytics solution in a boutique *asset management firm*, or "AMF". This process will move from specifying the domain problems, to detailed requirement elicitations, to visualization and interaction design, to the choice of a technological pipeline for solution implementation.

The major benefit of the functional evaluation framework here was the simplification of decision making with regards to project management, design, and technological issues. The VA application designer entered design meetings armed with a guideline to ask questions about the analytic task and the data, to choose alternative visualizations and interaction techniques, and to consider technological constraints that affects the choice of tools and platform for implementation.

The result of this simplification was a reduction in requirement elicitation and design time. Had this been a commercial application development situation, the time saving would also translate to a reduction in cost. Having a framework to guide requirement elicitation and rationalizing design alternatives also helped to smooth the conversation with the client and maintain momentum in the design meetings.

5.1 Project Management Concerns

The most significant difficulty faced in the design process was accessing the analysts at the firm and sample client data. The analysts were often occupied and difficult to schedule, and were only available for meetings during the first month of the project. Access to the data and IT infrastructure at the target firm was extremely limited for confidentiality reasons. Thus, the design process had to be very flexible and iterative in nature, so that incremental progress could be made with incomplete data and a minimal cost of analyst's time.

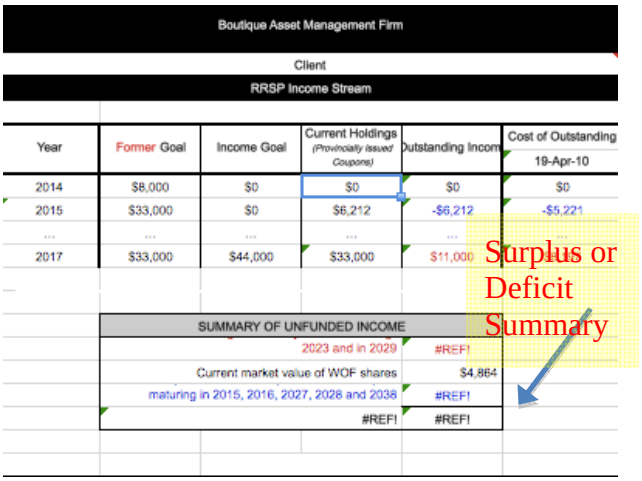
Three interviews were conducted with an analyst at the target firm over the course of one month. The interviews were moderated participatory design sessions; using user-centered design techniques, such as cognitive walkthrough, paper prototyping, and wire-frame mock-ups. To provide an incentive for the firm to collaborate, one of the project's goals was to produce a system that solved a real problem at the firm, and could be used by the firm. Therefore, the existing technology in use at the firm and the firm's operational processes were taken into account even from the design and mock-up stages. A list of questions asked in the first design interview, before the application designer gained a good understanding of the domain and the firm to structure design activities, could be found in Appendix A.3.

5.2 Data Format, Unit Type, and Semantics

The sample data consisted of a small series of Excel workbooks. Each of the Excel workbooks represented one anonymized client. Each client workbook had one or more accounts. The content of each account was not structured according to the instruments hold, but to the income stream generated by instruments maturing over the years. The workbooks contained both a display of data that would be explained to clients during quarterly meetings, as well as calculations used by the analysts. As a result, each of the workbooks contained many worksheets, some of which were very densely populated with numbers and formulae. In addition, some naming and terminology inconsistencies in the data also existed. These issues in information management represented both an interface design challenge and an opportunity for a custom interface and visualization to make a difference.

In terms of information organization, each workbook contained four types of worksheets: dashboard worksheets, account sheets, formulae sheets, and price data sheets. The account worksheet type was the core of each workbook. The structure of a typical account worksheet was similar to Figure 5.

Figure 5 A Typical Account Worksheet.



This was also the only type of worksheet with a mostly consistent semantic across all clients. Several notable fields in each account worksheet are:

1. Year (of retirement): The year column starts with the first relevant year for an account, which is the first year when the Income Goal and Current Holding field is non-zero. This is also typically the retirement years.

2. Income Goal (IG): The sum of the income goals in a corresponding year within all accounts that belong to a person is that person's expected income for that retirement year.

3. Current Holding (CH): In the target firm, the value of this field in a given year does not mean the market value of the strip coupons held in that year, but rather, the aggregated value at maturity of all the coupons being held that mature in that year. This reflects the risk-minimizing strategy of building up portfolios so that the value at maturity for all the holdings at a given year matches the income goal.

4. Outstanding Income (OI): This is defined at the target firm as the part of the income goal that had not yet been met: OI = IG – CH. A negative value in this field is a surplus and a positive value is a deficit (unmet goal). When there is a surplus in a certain year, then some of the coupons maturing in that year (that together yielding the amount of the surplus) could be sold at market value, so that deficit years may be shored up.

5. Cost of Outstanding Income, as of a given year (COI): This is defined as the actual dollar amount needed to balance a deficit year. COI = (IG – CH) * (Average) price of one or more coupons maturing in a given year / 100 (coupons per bundle).

6. Miscellaneous assets (MA) or liabilities (ML): There can also be assets (e.g. WOF shares or Cash) that do not fit naturally into the portfolio analysis processes involving strictly fixed-income instruments. MA/ML was included as semantically equivalent to a surplus in any one year.

7. Cost of Outstanding Income (Surplus or Deficit) Summary (COIS): This value, the final field in an account worksheet, is the sum of the cumulative deficit and the cumulative surplus, plus any Miscellaneous Assets or Liabilities. The COIS value is used by analysts to indicate the health of an account at-a-glance, and as an easy-to-understand value to show to clients. This summary, and the yearly Cost of Outstanding Income, is the performance measure that was represented in our dashboard to support the portfolio-monitoring task.

A 3% commission (1.5% per transaction) was applied to all the COI Summary values that were a surplus, representing the cost of two transactions that would be incurred if the holdings were rebalanced, by selling coupons maturing in surplus years, and buying more coupons for the deficit years. This commission was not shown as an explicit field in the data, but could be found in the formulae embedded in the Account worksheets. The 3% value was hard-coded in each formula that uses it. In the dashboard tool, this was extracted into a commission variable that was applied to all surplus COIS values, regardless of the client workbook that the value came from.

Even though the semantic of the account sheets was mostly consistent, the structure and positioning of the fields still had significant variation between different client workbooks and even within the same workbook. The information header sometimes contained one, two or four rows instead of three. The field containing the family member name in each account did not have a consistent position. The account holder’s name was also worked into the name of the account worksheet, but the naming of the worksheets was also not standardized. Therefore, the encoding of account holder’s name in the data was human (analyst) readable,

but could not be parsed reliably with software. For example: “Clark-RRSP” would be an account worksheet in the workbook “Kent-Household.xls”, but a semantically identical account in another client might be named “RRSP_Bruce_Wayne”. The “Former Goal / Holdings / Outstanding Income / Cost of Outstanding” columns might or might not be hidden in a worksheet, also causing parsing problems. The terminology used for the various important fields was also not standardized. Some account worksheets have the Outstanding Income and Cost of Outstanding fields substituted with Unfunded Income and Cost of Unfunded respectively, with apparently the same semantic.

To further complicate the design of a standard solution, the sample report documents that were sent to the clients usually addressed each person in the household separately and listed the accounts that each person holds. This was possible at the firm because the client report documents were personally written, not generated via software. To ensure the reliability and integrity of the data model, it was decided not to try parsing the non-standardized names to separate the household members from the accounts that they own. Instead, our solution would treat each client workbook as containing one or more accounts. It was decided, for the sake of feasibility at least in this iteration of the solution, to employ a treatment of the data that deviated from the real (but ad hoc) workflow at the firm.

The price data worksheet was populated manually, via copy & paste, from a separate price update Word document from one of the larger brokerage firms, such as RBC. Unlike the equity or option markets, the fixed income market has not been automated and computerized to the same degree. At the target firm, bids and asks were still over-the-counter, between one party and another. This manual update process was very time-consuming and error prone, especially when replicated over hundreds of clients, so price update was typically performed once every six months, or sometimes at the end of a quarter. Fortunately, while the potential source for errors and the labour-intensive nature of the price update was a serious problem, the timeframe of the update was currently adequate. The prices of fixed-income instruments are usually stable over time, moving only to significant macro-economic changes and monetary policy actions. The other source of price difference comes from the bid/ask spreads of different sellers. The price update process presented a great opportunity for workflow automation, but automating this process did not present a large improvement in portfolio

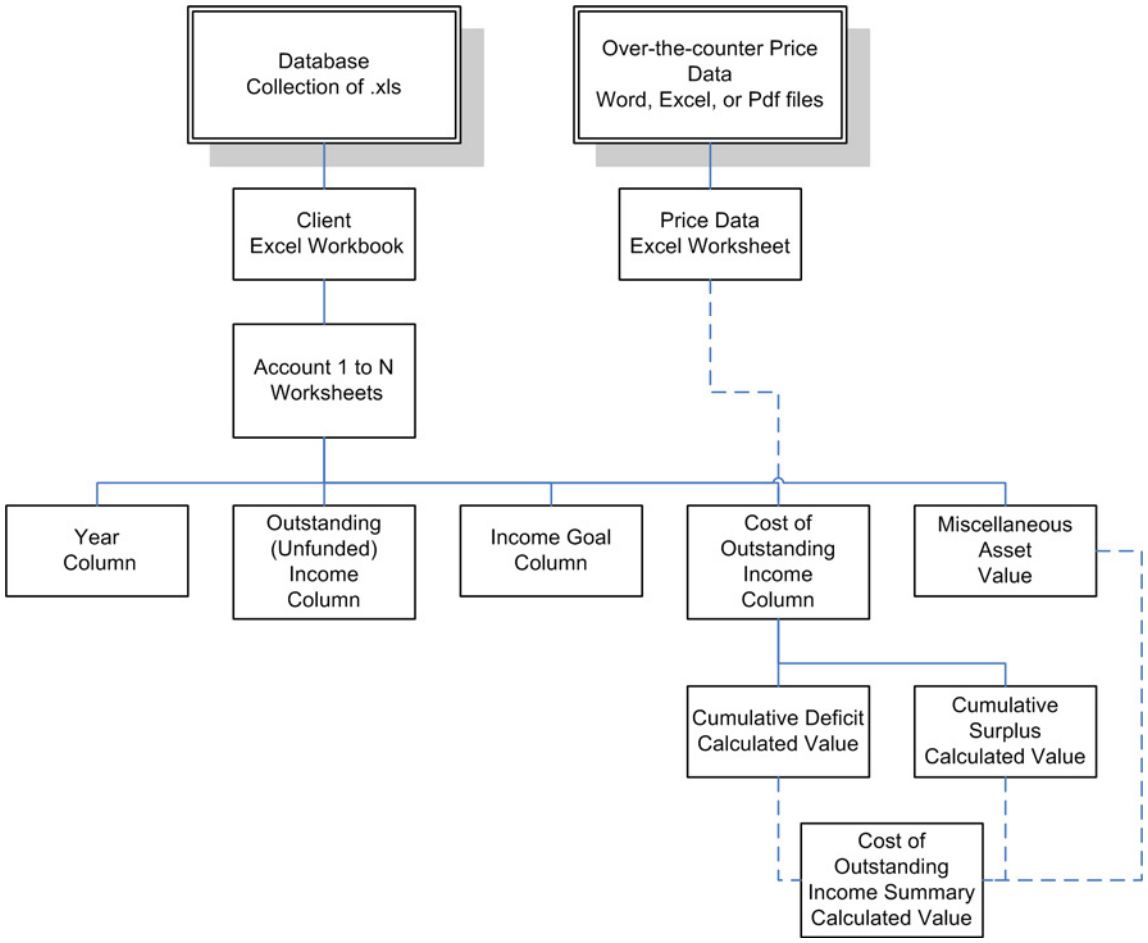
analysis nor an opportunity to demonstrate the use of visual analytics. Therefore, it was decided not to attempt to solve this manual price update issue in the solution.

5.3 Data Model

From examining the sample dataset, isolating common elements between the different client workbooks, including the elements that were semantically the same but were named differently, A conceptual model was created for the data that was used in the most common analysis tasks at the target firm. This was the data model around which the visualizations and graphical dashboard would be built.

The conceptual model was hierarchical, as shown in Figure 6. The storage format of each entity in the model was kept as currently found at the firm, but the model could be implemented using other file/database formats that would be more interoperable with the rest of the technology pipeline required for the solution implementation.

Figure 6 Model of Common Denominator Data Fields Between all Client Accounts.



Hypothetically, external data, such as price quotes and economic data can be automatically imported in the future via Bloomberg feeds. However, the schema necessary to integrate automatic data import with the calculations in the account worksheets were omitted from this report, because automatic fixed-income data import was not supported in the current iteration of the project (due to feasibility, considering the over-the-counter nature of many fixed-income instruments).

The price data quotes of fixed-income instruments, currently delivered to the target firm via a Microsoft Word document from the broker, contain these fields:

- Date of Maturity or Year to Maturity: The end of life of a fixed-income security, when the principle and all accrued interest must be completely repaid

- ID: A numerical, unique identifier of the instrument
- The issuer of the security
- Inventory: Due to the over-the-counter nature of fixed-income transaction, the inventory available for sale is often included in the quote
- Market Price, usually quoted per batch of 100 individual instruments
- Yield, which can be either the coupon yield (the interest rate fixed at issuance), current (the interest rate as a percentage of the current market price), and yield to maturity (an estimate of what an investor will receive if the instrument is held to its maturity date)

In the data model, there would only be a concern with the Date of Maturity and Market Price, because the other values could be derived from these values. The ID was also important for analysts' use to ease and clarify communication regarding a specific instrument.

5.4 Analytic Problems and Other Functional Requirements

The first element of the analytic problem was the high-level business processes. The firm serves a specific niche, and treats each client separately, with separate plans and investment strategies that take into account more than just the technical analysis considerations of fixed-income investment, but also the client's lifestyle, career plans, and risk tolerance. It is very difficult to identify and classify all of the possible analytic actions in a firm that specializes in personalized asset management. Thus, a decision was made to address two common-denominator analytic problems that often arise for all clients.

Portfolio monitoring: An analyst will often want to understand at a glance which client portfolio is performing to expectation, and which is lagging. After identifying clients that may be in trouble, the analyst will want to zoom in to only these clients and see the client portfolios at the account level, then the level of holdings within each account. This thought-process corresponds with Shneiderman's mantra (1996) of Visual Information Seeking: overview (of all clients), filtering (the clients in trouble) and zooming (to the constituent

accounts), and details on demand. The visual dashboard and visualizations could take advantage of pre-attentive processes in visual cognition to simplify the interactions required to monitor portfolios, matching the optimal information seeking mantra.

Scenario Analysis: an analyst will often hypothetically change certain values in an account, or certain market or economic conditions, and observe the results of these changes on the performance of the account. The major performance measures used in Portfolio Monitoring and Scenario Analysis at the time are COI values, either in a given year or aggregated over a retirement plan. In other words, the performance in a given year is defined as whether the income goal can be met exactly by all the fixed-income holdings that mature in that year. This is different from the common conception of portfolio performance: the profit & lost compared to the current market value of all the holdings.

In contrast to the optimal information seeking mantra, the existing processes of portfolio monitoring and scenario analysis at the target firm involved very high cognitive load. Since there was no dashboard interface that showed the consolidated information of all the client workbooks at once, the analysts had to open and examine each client workbook manually, even when only an overview was required. Each client's workbook contained both a display of data that would be explained to clients during quarterly meetings, as well as calculations that were used by the analysts. As a result, each of the workbooks contained many worksheets, some of which were very densely populated with numbers and formulae. This represented both an interface design problem and an opportunity for a custom interface and visualization to make a difference.

For each client, there was a consolidated dashboard worksheet, but they were numerical, and very dense. An analyst must read through these numeric fields and mentally process them in order to understand the performance of just one client. In this pre-VA process, the details were not provided on demand, but were always presented all at once. More importantly, the number of accounts for each client workbook was different, so the dashboard sheet in each account had a different number of columns. The number of relevant years to manage was also different from person to person in each client portfolio, and from one portfolio to the next, so the number of rows in each of the columns in the dashboard sheet was also non-

uniform. Analysts cannot count on their visual spatial memory regarding the positions of important fields in the data, and has to perform a linear visual search for every important field in each worksheet. In addition, the analysts need to read through multiple numeric fields and mentally process them in order to understand the performance of just one client. Reading and mentally processing the semantic meaning of a number took perhaps a second. Each linear visual search also only took seconds. Opening a workbook also took very little time. However, cumulatively, these activities were time-consuming and physically tiring, but, fortunately, avoidable via design.

A custom dashboard interface and visualizations of data also served as a form of virtual memory (Munzner 2009) to further reduce the cognitive load of portfolio monitoring. Investigating each client required the analyst to compare information from multiple fields in the dashboard spreadsheet. If there were a problem to investigate further, then the investigation would involve comparing fields from multiple account worksheets that must be switched back and forth. As the typical client portfolio had two family members, each having about two accounts, which contained two to three important summary fields, the seven-item limit of human short-term memory (Miller 1956) was often stressed just analyzing one client. Comparison between many client portfolios was currently not feasible at the firm. Even at the overview level, to find patterns or research strategies would currently require opening and comparing the dashboard worksheets in potentially hundreds of workbooks.

Another process intimately related to Scenario Analysis is Analytic Change Tracking (i.e. managing the history (progress) of analytic processes). By definition, Scenario Analysis involved performing hypothetical analytic actions, observing the results, and comparing the results of multiple scenarios. This required storing and retrieving analytic history, both within one working session and between different working sessions. This problem is analogous to Change / Version Control in Software Development. After all, changes to software represent the results of analytical processes, and these changes are tracked across working sessions by saving and comparing different versions of files.

Financial analytics actions can be classified and tracked as well. This change tracking was already performed manually. The analysts would manually save multiple versions of each

client workbook, and created a new working copy of the latest version before analyzing. Within each working session, the analysts relied on short-term human memory to remember analytic actions. Excel has a tracked-change feature, but the change tracking markers in Excel are generic, such as “cell J1 changes from 0 to 4500”, and do not provide semantic meaning. The current workflow for managing analytic history was very error-prone, labour-intensive, and cognitively taxing.

Analytic change tracking, including the side-by-side comparison of scenarios, was also a much-needed feature in our Financial Visual Analytics solution, and an interesting problem of VA research in its own right. This problem has only recently been explored in research prototypes such as CZSaw (Chen et al. 2011) and Harvest (Gotz and Zhou 2008), and has not been investigated in financial analytic solutions, to our knowledge. In fact, the analysts at the target firm singled out this problem as the one most important improvement to the analytic workflow for which they would like to have a VA solution.

The first step in designing a system and interface for analytic change tracking was to classify all, or at least most, possible domain-and-firm-specific analytic actions that the system would support (Gotz and Zhou 2008). However, doing this for a boutique firm with custom processes for every client would be prohibitively time consuming. Therefore, the decision was not to solve the Analytic Change Tracking problem in the prototype. Without supporting analytic change tracking, the analytic steps no longer had to be completely classified. This allowed for design effort to be concentrated on solving problems that are generalizable to other fixed-income firms:

I) The main performance measure for a portfolio at the target firm, Cost of Outstanding Income, was not generalizable, but for the purpose of visualization, it could be treated as any other cardinal numerical value, such as Profit/Loss. Surplus and Deficit would be analogous to Paper Profit and Paper Loss, respectively, when an analyst gauged the performance of clients and accounts.

II) At the holdings level, the firm organized holdings by year of maturity, not by instruments. However, COI values corresponded one-to-one to year of maturity, analogous to the one-to-one relationship of P/L to instrument in a typical listing of

holdings. Therefore, an effective visual encoding of year-to-maturity would be generalizable to encode a typical instrument-oriented list of holdings.

III) Inputting market prices, and price projections, for instruments was a generalizable action, and would be supported in the solution.

Finally, the use-case of presenting the visualizations to the firm's client was also taken into consideration. The aim was to simplify and clarify the client reports and save time and cost for the analysts to draft the reports. Not only did the design of the visualizations and user interface need to be conducive to the understanding of a lay-audience, the technology implementation had to afford the saving of the visualization as an image file to include in the client report. One of the research questions when evaluating the solution would be whether the solution had improved the ease-of-understanding of the client reports. The ease-of-understanding, from the client's point of view, or the ease-of-explanation from the analyst's point of view would be a direct measure of the usability of financial reports.

5.5 Visual Analysis Solution

5.5.1 Visualization and Interaction Design

The design involved a dashboard user interface that displayed at a glance: the list of clients, the list of accounts in the currently highlighted clients, the information in the worksheet of the currently highlighted account, and price data. Aware of the cognitive load involved in switching rapidly between multiple visualizations (Klein et al., 2002), the design approach was to use a minimal number of visualizations that would satisfy the data representation scheme: a bullet graph and a treemap.

The table of income goals/streams could be edited, as well as the table of price data. Linking / Brushing (Becker & Cleveland 1987) behaviours were built between the editable fields in the UI and the two visualizations, so that they could be used for scenario analysis. The analyst would edit the fields to make hypotheses about the factors, and the visualization would re-render instantly to reflect the results. Changing any of the data fields would instantly cause both of the visualizations to change. Selecting a client and account either in the list box of accounts would trigger the same selection on the treemap visualization, and vice versa. Linking / Brushing enhanced usability by providing multiple alternative actions for accomplishing a given task, as well as multiple redundant representations of the same data, reinforcing perception. The instant, synchronized updates also helped users learn the interface and semantic relationships between visual elements more quickly. Finally, the re-rendering of the visualizations was smoothly animated, which was both for aesthetics, and, more importantly, to draw the analyst's attention.

Figure 7 below shows the Visual Dashboard interface, with the three zooming states of the treemap visualization. The Visual Dashboard could be run from within the Firefox web browser, and the look and feel styles of the widgets were currently the default values for this browser. This default look-and-feel, though somewhat rustic by Web 2.0 user interface design standards for use in a production system, was adequate for a high-fidelity prototype.

Figure 7 Visual Dashboard with Treemap Showing Three States (All Accounts, All Accounts In One Client, And Holdings In One Particular Account)

The treemap (Shneiderman, 1991; Wattenberg, 1999) in our application was the major visualization for supporting the Portfolio Monitoring task. In the lowest level of detail, the

default state, the treemap visualization depicted a bird's eye view of all the clients, and their accounts. The underlying value of each node was the aggregated cost of outstanding income (COI), which was the sum of all COI values for all the retirement years within each account. The amount of information being displayed at any one time was minimized according to the principle of "overview, filter and zoom, details on demand", using semantic zooming (drill down) through different layers of the hierarchical data model. Zooming in once, the treemap would display one client, instead of all clients at once. This intermediate zoom level facilitated the demonstration of the visualizations to clients, by tactfully removing the other clients' view to preserve privacy. Prominent clickable icons above the treemap, in the now-common metaphor of magnifying loupes, provided the zooming interactions. The title label of the visualizations also indicated the semantic layer currently shown. Another benefit of semantic zooming was to enhance the (already formidable) scalability of the treemap visualization to larger datasets.

Zooming in twice from the lowest level of details would drill down to the highest level of details, the list of holdings in an account. The underlying data also represented COI, but for each year instead of cumulative. Here, the labels were more detailed, with the format "client name: account name: year: value of deficit or surplus." The color of the nodes encodes discrete states of account performance: whether the account is in surplus-green and deficit-red. The red was darker than the green, so that intensity of the cells provided redundant visual encoding to mitigate the effect of color-blindness or black-and-white printing. A gradient pointing to two opposite directions was also built into the red and green, to further distinguish the red and green cells. The relative size of the nodes represented the amount of the deficit / surplus, because relative sizes of rectangles could be compared very quickly and accurately via human visual perception.

The treemap was chosen to visualize the list of client, accounts, and holdings partly because of its efficient use of space. Compared to alternatives such as adjacency diagrams, treemaps do not have any whitespace, and do not need extra space for connectors between nodes. This treemap design could also be readily modified to use conventional P/L measures instead of Cost of Outstanding Income. One notable alternative technique that was considered was radial, space-filling (RSF) hierarchy visualization, such as the Sunburst or InterRing (Yang et

al. 2003), which could display more layers of data at once than a treemap while being semantically very similar. However, since the innermost layers near the center represent a higher level in the tree, there was actually less space in an InterRing for displaying the list of clients or accounts, which was the only indefinitely expanding dimension.

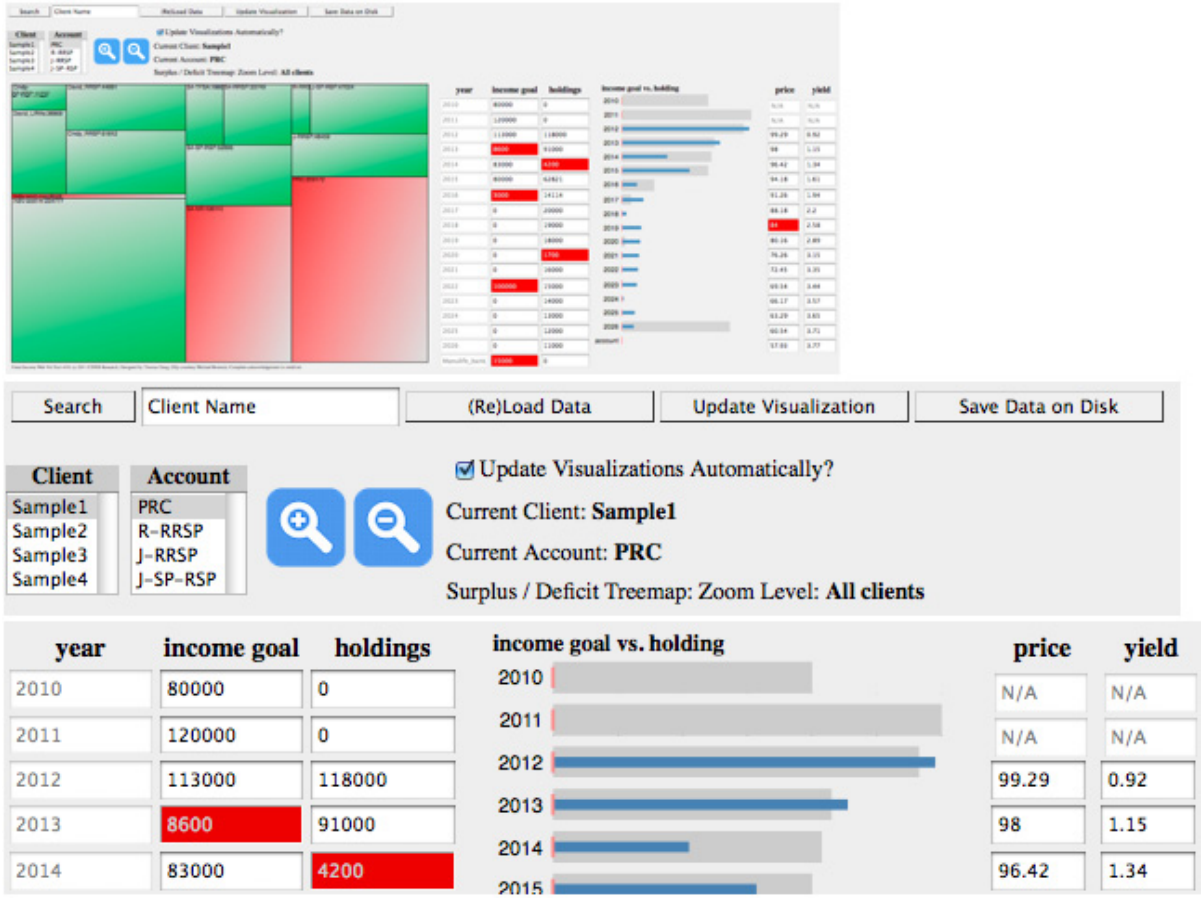
In addition to minimizing the number of visualizations used to represent all the necessary information to support the two target tasks, we also minimized the amount of information being displayed at any one time in the treemap according to the principle of "overview, filter and zoom, details on demand." The zooming metaphor was not a literal one, because clients' containing accounts, containing income projections (in the form of value at maturity of coupons), was an abstract relationship. The zooming is semantic in nature, which does not involve a scaling of the graphics, but rather a drill down into the hierarchical data model to show increasing level of "details on demand" (Shneiderman 1996). Another benefit of semantic zooming was to enhance the scalability of the treemap visualization, given the limited screen real estate, as only one level of detail was displayed at any one time. With an 800x600 area of screen space, it would take 1000 clients for the average space given to one client to be 20x24 pixels, which could contain four 10x12 "account" cells, which were still comfortably visible. There was no scalability problem for the "holdings" level of zoom, because the maximum number of retirement (relevant) years in an account could be safely assumed to be at most a hundred. Finally, semantic zooming limited the amount of information displayed on the visualization, which would make the visualization easier to explain and understood when included in client reports.

The table of account information (Year, Contribution Goal / Income Goal) could be edited, as well as the table of price data. The field denoting the interest rate projection at the currently highlighted year was also editable. These editable data fields were used for the scenario analysis use-case, when analysts could make hypotheses about the factors, and observe the result instantly.

Figure 8 below shows the layout of the dashboard, with interactive elements (widgets and editable data tables) shown in details. When a data field changes from the value saved in the

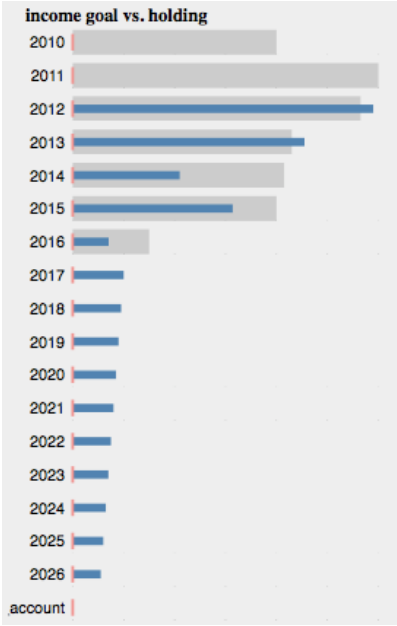
data file, the field will have a red background on the user interface, to provide a visual reminder for the analyst of the values changed in a scenario.

Figure 8 Visual Dashboard Layout and Interactive Elements



The other visualization was a bullet graph (Few, 2006) that displayed the income goals over time versus the cumulative value at maturity of current holdings over time (Figure 9). The three data types to be visually encoded included two numerical, cardinal values, which were very suitable for bar graphs, and discrete time series, which fit well into the X-axis of a graph. Income goals and current holdings were both denominated in the same currency, and were both in the same range, so they were ideal for bar graphs stacked side-by-side, or bullet graphs, where one bar graph is nested within the other to better utilize screen space. The grey bars outside, typically recommended for encoding a performance measure or benchmark, represented the income goal, while the slim blue bars represented current holdings.

Figure 9 Bullet Graph Showing "Goal vs. Holdings"



The Outstanding Income (surplus and deficit before being scaled by market prices of instruments) for each year could be derived visually from this visualization, as the gap between the two bars. A bullet graph often also contains a marker floating on top of the inner bar, which would be a red line, which could act as an optional third "bar" to visually encode another cardinal, numerical value. For our application, this value encoded by the red marker would be the "previous income goal" column in the data, which represented an ad hoc attempt by analysts to keep track of their own analytic history.

5.5.2 Choosing a Technology Pipeline

After finalizing the functional requirements with the analysts at AMF, as well as the list of features—the visualizations and dashboard designs—to be supported, it was necessary to consider alternative technological platforms on which to build the implementation.

The technology pipeline chosen to implement the solution consisted of the JavaScript language and the Data-Driven Documents (D3js) (Bostock 2011) visualization library. For rapid development of high fidelity prototypes of visualization systems without concerns for proprietary data formats, this pipeline provided the most advantages among the alternatives considered. As with all JavaScript programs, a D3js visualization project could be developed,

deployed and tested relatively quickly compared to using more comprehensive frameworks. Last but not least, implementing the high-fidelity prototype via JavaScript also eased the process of creating a printout of the visualizations to include in a client report by leveraging the mature printing facility built into web browsers.

Two full-featured finance-oriented commercial visualizations products were considered: Panopticon and Xcelsius (SAP Crystal Dashboard Design). Both of these products had relatively high licensing cost per user for a small firm, especially considering the experimental nature of the project. Panopticon (2010) is a very powerful software framework for building business dashboards that incorporates sophisticated visualizations. There is a treemap and a bullet graph builder in Panopticon that supports all the visual encoding in our design, and more. The data back-end can also be multiple formats, including OLAP cubes in JSON format for the implementation of real time data streams. A dashboard with multiple visualizations and some interactivity features could also be implemented relatively easily. However, Panopticon was not chosen as the platform for implementation, for these reasons:

- Despite being a commercial solution, with all the required features, Panopticon was not a turnkey solution, but only a technological scaffold for developing solutions. After licensing such a product, it would still be necessary to conceptualize the data, define the analytic problem, develop, deploy, and maintain the visual analytics solutions.

- Developing and modifying visualization systems in Panopticon is also non-trivial, so the presence of a technology expert is still necessary.

- Most of the time and financial cost of developing an analytic support solution is with the requirement elicitation and design, and a commercial development solution does not reduce this cost. Because of the myriad of possible variations in financial problems, there is no known turnkey solution for developing financial VA solutions at the time of this project.

Crystal Dashboard (Xcelsius) from SAP was also considered. Similar to Panopticon in power and feature-completeness, Xcelsius had the distinct advantage of integration with Excel.

Xcelsius dashboards could be built into an Excel workbook, integrating seamlessly with the familiar workflow at AMF and many other finance firms. However, Xcelsius had all the disadvantages of Panopticon, and other commercial information visualization frameworks. Despite the high overhead cost, Xcelsius was also not a turnkey solution, and would still require technological expertise to deploy and maintain. The construction of the dashboard was also non-trivial, and did not result in any significant saving either in development time or cost over developing and deploying custom software from scratch using a visualization library, such as D3js. Xcelsius, like Panopticon, was designed for a large organization with multiple problems, or proprietary data sources for which it would be difficult to develop a custom system.

These commercial visualization suites would be more useful when their cost per user could be distributed over many potential applications. The solution designed could be implemented with free visualization API that provided a more limited feature set, but was also simpler to learn, deploy, and maintain. Only for a very large firm may the use of a comprehensive and professionally supported technology platform have enough benefit to outweigh the costs, because there may be many different processes, for which the platform could be used again and again to develop solutions. Part of the cost of maintenance and training could be mitigated if there were an in-house IT workforce, because the cost of retaining these in-house technology experts would already be committed.

General-purpose Numerical Data Visualization software, such as Tableau and Orange, may be as powerful or more so than even the commercial products designed specifically for finance. However, while the commercial products of this class also tend to have high overhead costs, they cannot be adapted naturally to specific problems in finance. Because these tools are general purpose, it would also be difficult to integrate them seamlessly into an existing financial analysis workflow, because the data semantic and terminology familiar to the financial analysts would not (and currently cannot) be built into the software. The interfaces of these products also tend to be very sophisticated and not very customizable, in order to accommodate a generalized problem space. Deploying these general-purpose solutions would require an even higher training time and cost, or the retention of a visual analytics expert in order to use these tools effectively to solve specific problems.

After deciding on implementing the custom solution from scratch with a visualization library, we also explored multiple potential languages and libraries, trying to create the technology pipeline that would support all of the current requirements, as well as leaving the door open for future improvements. Some of the alternatives considered were:

- Visual C++ + DLL / VBA bridge in Excel + Visualization Toolkit (VTK): The major benefit of this approach is the sheer power of VTK. VTK is by far the most powerful visualization API reviewed for this research, while also being free and open-sourced. Integration with Excel was also simpler in Visual C++, a sibling technology in the Microsoft family. However, this approach had the same drawback as using commercial visualization toolkits: high overhead cost (but in terms of learning and development time). The design could be implemented with a much simpler technology pipeline, even at the cost of Excel integration.

- Java + JExcel + Information Visualization Toolkit (IVTK): Java is a very powerful and mature programming language and is also cross-platform. A Java application could be deployed across an organization, regardless of the operating system or hardware platform of the target machine. JExcel (Khan A., 2010) is a relatively mature library for reading/writing Excel workbooks in Java. IVTK is a mature visualization library that supports treemap and time series line chart, which could be used in place of the bullet graphs in our design. However, IVTK has not been actively supported since 2006, and its look and feel was dated by today's standards.

- Java + Prefuse: Prefuse is a mature and powerful visualization library that had support for building treemaps. Building bullet graphs was not natively supported, but could be implemented from the basic facilities in the library as well. The decision to use JavaScript + D3js instead of Prefuse is due to the comparative ease of setting up a bullet graph, and the easy leveraging of printing facilities in a browser-based technology.

- JavaScript + JSON + D3js: D3js is the successor of Protovis, a mature JavaScript visualization library that was actively supported up to June 2011, before D3js was first released.

- Protovis itself was also evaluated. Although Protovis is more mature and feature-rich, the decision was made to use the more modern, and actively supported D3js, because of the greater potential for future improvement. D3js has a completely different architecture from that of Protovis. While Protovis is a monolithic library of visualization types and interactions, D3js provides only the data binding and manipulation features. The visualization capabilities in D3js was built separately as small, compartmentalized, wrappers around underlying technologies such as CSS3, HTML5, and SVG. D3js is very fast and scalable to large datasets, because of its extensive use of web standards built into browsers.

5.5.3 Generalizability and Reusability Concerns

The visual encoding of performance measures are generalizable to conventional Profit/Loss analysis, but the relationship between a change in price data and OI values (in the treemap) and COI (in the bullet chart) is completely different from the relationship to conventional P/L. Specifically, a change in the price of an instrument does not affect OI at all, as this was a value derived from yield-at-maturity. The COI value is affected in a subtle, additive way; the appreciation of an instrument exaggerates both a surplus and a deficit value. In a conventional holding listing, rising price corresponds to more profit, and vice versa. The reverse relationship is also possible, when instruments are short sold, but the P/L of a position is never moved in the same direction by a change in price.

To retain the generalizability of the implementation, the observer pattern (Gamma 1994) was applied to link the visualization and input fields together, so that the actual effects of an action in one UI element are deferred to the other UI elements that are notified of the change. In addition, the content of each node on the treemap, and each bar on the bullet graph, was populated via callback functions called whenever the visualizations re-render due to a change in the underlying data. The callback function pattern (Gamma 1994) enabled the semantic behaviours of the visualizations to be modular, and enabled the possibility of implementation with new rules for different situations.

5.6 Evaluation

The finished prototype, which could be run instantly from its installation folder, on any machine with a web browser, was deployed at the target firm for a two-month field test. The analyst was furnished with a brief guide (Appendix A.1) on the features of the prototype, and known limitations, to compensate for the fact that this software was still a vertical prototype, and so could not handle other analytic tasks beyond the two tasks that it was designed for.

After the field test, an evaluation interview was conducted with the analyst who agreed to try using the tool on actual client data. Before the interview, the analyst was asked to fill out a questionnaire (Appendix A.2), in an attempt to alleviate the bias that could be introduced in a conversation, especially since the analyst herself was involved in the design process. The analyst's responses were positive in both the questionnaire and the interview. In terms of usefulness, the analyst indicated that "(the product) significantly improves the speed with which we see the impact of movements of the yield curve on client's surpluses/deficits…" and she liked that "… the effect of changes in a certain variable can be seen instantly in each account and in all accounts of one client or all clients at once." The analyst also did not find any aspect of the interface to be difficult to learn or unpleasant to use. The complete response for the evaluation questionnaire can be found in Appendix A.2.

However, the analyst noted that she was not aware that the visualization could update instantly whenever a parameter was changed. She only realized afterward that the default setting was to update the visualizations manually with a button press. She was surprised that we did not make automatic updating the default behaviour, which indicated that the ability to make many changes in the data and then observing how the visualization changes may not be important or natural in the analyst's mental model.

The analyst indicated that although both the portfolio monitoring and scenario analysis task (with regards to COI) were adequately solved with the current prototypes, it would be useful to have the ability to revert back to previous states in scenario analysis, or even to visualize the analytic history. This ability would be especially important between sessions of using the tool. Since changes in any particular client portfolio do not happen every day, it became difficult to remember the changes between analytic sessions and their impacts to COI. In the

existing process at the firm, Excel only provided the undo feature within a session. A version control or document management system could provide the ability to revert back to past states, but would require extra funding and time to deploy. Having the capacity to track analytic history integrated within the analytic tool itself would also enhance usability, as the user would not have to switch between multiple tools and interfaces, for a feature that was important but not used often.

The result of this evaluation is only suggestive when generalized to other fixed-income analysts, because the sample included only one analyst in a firm employing three analysts. However, this evaluation is adequate for this study, because the small subject pool was due to a deliberate limitation of scope. The goal of this study is to apply the 2FEVA framework as a guide in a financial VA application design project in a real environment. While single case studies have an acknowledged limitation in external validity, the benefit of gaining an in-depth understanding of contextual information (specific to a subdomain of finance and one firm) outweighs the limitation, since it is highly pertinent to the research goal (Yin 2009). Evaluation is also difficult to apply to subjects outside of the firm, because only a small set of anonymized client data was available to the designer. A field trial undertaken by an analyst at the firm was mandatory to ensure confidentiality.

Some previous design case studies of VA systems also faced the challenges of evaluating prototypes in real environments, and have made similar trade-offs for the sake of feasibility. Wongsuphasawat et al. (2011) engaged in the participatory design and evaluation of a patient flow visualization in a hospital setting with one medical doctor. Savikhin et al. (2011) defined analytic problems using one portfolio-selection theory, without considering other competing theories or the specific details of any firm, and evaluated the prototype on student volunteers, not domain experts. Chang et al. (2008) aimed to solve a firm-specific problem, but also evaluated their prototype on students of human-computer interaction and information visualization, who “were not the intended users.” A limited subject pool can also be found in other case studies in HCI unrelated to VA, such as Light (2011) study on an elderly lady’s adoption of computing technology. The difficulty in finding and retaining subject in Light’s study was due to its time commitment and in-depth involvement.

6 Conclusion

6.1 Discussion and Lessons Learned

6.1.1 Fixed-Income Visual Analysis Dashboard

There are many challenges in the design and implementation of VA systems for finance. Financial analysis tasks often involve many dimensions, are difficult to describe conceptually, and tend to have variations among different firms.

Therefore, access to the finance professionals and stakeholders in financial institutions is crucial to the accurate specification of the analytic task and the desired characteristics of the solution. However, financial professionals are often difficult to access, so the methodology for participatory problem specification and design needs to be optimized.

There was also value in spontaneous cognitive walkthroughs and paper prototyping during the design sessions with analysts. This process helped the technology expert understand more about the fixed-income asset management domain and to rapidly confirm the accuracy of this understanding to the firm's personnel. Spontaneous prototyping also helped the analysts at the firm understand more about VA and the design possibilities offered by newer software technology. This acted as an icebreaker and stimulated the conversation, as the analysts discussed problems and desired features that they did not know to be feasible. Last but not least, the cognitive walkthroughs and paper prototyping were also very helpful in building a working rapport between the firm's personnel and the designer.

Data organization and quality are very important in a VA system design project. The majority of the time spent in developing the solution actually involved cleaning up the slight differences in formatting, naming, and locations of important fields in the various spreadsheets. The spreadsheets were largely similar in formatting, but slight and sporadic differences created many outlier accounts where the result of the calculations became completely wrong. The differences in formatting, field-location, and naming were also introduced through both variations in analytic needs for different clients, and personal styles of different analysts. The small and infrequent nature of these inconsistencies made diagnosing bugs difficult. More significantly, sinister calculation errors might not create any

observable warning signs on the visualization at all. Nevertheless, it was noted that the data organization and quality issues also impacted the existing workflow without the VA dashboard tool. A subtle calculation error caused by a hidden field in a particular spreadsheet caused confusion to the analyst who created that spreadsheet. An error caused by a particular analyst's naming / formatting style would be very difficult to spot by another analyst.

Several recommendations were made to the analysts based on the sample data clean-up work to facilitate input into software analytic tools and promote consistency and reduce errors in the spreadsheets. The following guidelines for formatting tabular data (spreadsheets) were recommended:

- Data, including column headers, should start on consistent pre-defined row and column (preferably zero).

- A set number of rows can be reserved beforehand, firm-wide, for any decorations, so that the data can be relied on to always start in a set row.

- Summary, aggregate, and other "Final Results" fields in a spreadsheet are often placed at the end of the data table. This result in inconsistent locations. Marking the field with a "name" field beside it can also easily introduce error, since the name contained within the name field may sometimes contain typos. Final Value fields in a sheet should be placed a set number of columns from the final column of data, with the row starting at the row of the column header. This method allows the location of the Final Value fields to be calculated programmatically, regardless of the number of columns or rows in the data.

- The naming of worksheets and workbooks should follow a consistent system across the firm.

- Terminology should be standardized within the firm for the purpose of naming worksheets and data fields. For example, the phrase "RRSP" should be spelled one way only for the purpose of field / worksheet naming, as opposed to "RegisteredRSP", etc.

6.1.2 Visual Analysis Application Evaluation Framework

Through applying the 2FEVA framework in a design case study, several lessons were learned regarding its strength and weaknesses. The many dimensions for deconstructing domain problems were very useful in helping the designer acquire a comprehensive understanding of the problem and in helping the analysts understand technological possibilities. Several examples of the framework's benefit are given below:

- The analysts were not conscious of the hierarchical structure built into the data, and the design implications, because they have always thought about the data as a collection of spread sheet files. The hierarchical structure became apparent when suggested by the designer.

- Once aware that the client portfolio data is conceptually hierarchical, the analysts also realized that price and commission data do not belong in this hierarchy, and thus should be kept in a separate file instead of hardcoded in many clients' files.

- The analysts struggled to find the right language to explain the task of analytic history tracking, but they could immediately confirm what they required when presented with a description of a software recording and rendering past visualization states super-imposed on the current one.

- The analysts did not suggest that "search" would be a frequent sub-task of their analytic activities and a crucial feature in the prototype. The framework helped to suggest this task.

- After having a comprehensive understanding of the business processes (which the analytic tasks were part of), the designer could consult a ready set of visualizations and interactions given the problem parameters.

- The designer was able to confidently discuss the choice of technology pipeline for implementation, especially regarding the file format import / export. This confidence helped instil rapport between the analysts and the designer and smoothen the conversation.

The framework also had several limitations when applied to a real project:

- The tabular format of the framework was suitable as a reference, since a lot of information could be condensed into a small size, but the designer still had to formulate interview questions based on the dimensions of the framework beforehand, since the tabular format seemed to be too dense for referencing in the middle of a conversation.

- Real-life tasks have multiple layers of abstraction that may need to be considered at the same time. For example, the client portfolio data was hierarchical conceptually, and was visualized as a treemap, but the physical level was an unstructured collection of files containing flat data, which affected the choice of technology. Similarly, the tasks of scenario analysis and analytic history tracking were not easily separable from the analysts' point-of-view. These real world insights did not fit the clean structure of the framework, where each task can be considered separately. The concept of abstraction should be introduced into every step of the evaluation process. There should be guidelines for breaking down high-level problems into sub-problems before the sub-problems become simple enough to be evaluated with the concrete dimensions in the framework.

- In the current iteration of the framework, the dimensions could not be evaluated in terms of time and monetary costs. Therefore, the designer could not have justified a price for the VA application, had this been a commercial development project.

6.2 Limitations and Future Works

6.2.1 Fixed-Income Visual Analysis Dashboard

This work has several limitations, which the author intends to address in future projects. The evaluation was only carried out with one analyst field-testing the prototype. This was due to the limited time availability of analysts for testing the software. There is an inherent risk in applying an unfamiliar analytic method (VA), and new software, to real client data, so the analyst could only use the prototype on a testing basis, in addition to carrying out the same analytic tasks with her existing process in Excel. Therefore, the productivity enhancement

potential of the tool can only be assumed based on the evaluation of the analyst, but cannot actually be confirmed. The other limitation comes from the relative difficulty in importing data into the tool, since the original Excel data must be saved as CSV files for import. Therefore, switching back and forth between the "real work" environment in Excel and the VA tool for Cost of Outstanding Income analysis is not seamless. This could create a bias in the evaluation of the tool.

The future direction for this project involves applying lessons learned regarding design process in other projects in financial institutions, with real data and operational constraints, to further refine the methodology. It was also necessary to connect to and undertake projects within large firms, which may have very different operating and technological constraints than smaller ones. The aim of this project is to develop guidelines for design processes that would be adaptable to the practical conditions in financial institutions of all sizes.

The evaluation of the tool developed in this project would benefit from a longitudinal study, where the tool is seamlessly integrated into the workflow of a firm.

6.2.2 Visual Analysis Application Evaluation Framework

Firstly, there were several minor inconsistencies within the evaluation framework.

1. The concerns of data pre-processing were not comprehensively addressed in this framework, because stand-alone data mining and processing products were not evaluated due to time constraint, although these products can greatly enrich visualizations. This limitation could be addressed in future versions of the framework.

2. The "Deployment & Expertise Acquisition / Retention" dimension contained criteria that was categorical in nature instead of ordinal, so the "level of support" scale (0: not support; 1: supported but needs programming; 2: supported outright (but may require customization) common to other dimensions in the implementation-phase table cannot be applied. Each of these criteria in this special dimension came with its own nominal scale instead.

Secondly, this project is, and always will be, a work in progress, because the domains of VA and finance, bridged together by this functional evaluation framework, will always be changing. New visualizations, and variations of existing ones, will be devised, and the framework will need to be updated to keep up with the progress of the field. Interaction design, a domain related to VA and referred to in this framework, is also rapidly changing, and new interactions, interfaces, and widgets will also need to be added to the framework. Similarly, new data formats will also need to be added, including those offered by specialized commercial data vendors.

Finance is a vast and diverse field, and a single study of this nature cannot claim to have taken into account an exhaustive list of the types of domain problems. Even the subdomains of financial analysis and portfolio management may contain problems not accounted for by this framework. Since visualizable domain problems form the starting point of the evaluation process, the structure of the framework will evolve over time to take into account a more complete taxonomy of domain problems. Indeed, this work describes not only the current state of the functional evaluation framework, but also the systematic model unto which further problems, visualization, interactions, and data features can be incorporated.

The current version of the functional evaluation framework has a heavier slant toward the subdomains of security (financial) analysis and portfolio management as applied to the equity and option markets. The majority of the concepts in this taxonomy are extendable naturally to trading in other financial products, such as fixed-income and over-the-counter derivatives, because the data abstraction, type, and storage format is often similar across sub-domains in finance, even if the semantic maybe different. However, certain aspects of fixed-income and OTC derivative trading will not overlap with equities; and there are yet other sub-problems beyond the author's expertise that were not covered.

It is extremely difficult and time-consuming to create a functional evaluation framework of VA that will take into account every organization-specific modifier to the domain problem, or every possible workflow and operational process. The breakdown of these elements in a functional evaluation framework is meant to be a starting point for the VA application designer to be mindful of considering these aspects during requirement elicitation. The

methodology of compiling literature between the domain of finance and VA and Human-Computer Interaction, and drawing parallels between the two, breaks down for this aspect of evaluation. While the number of visualizable domain problems, and the number of data abstractions and operations may be very large and possibly changing, it is not infinite. It is possible to create a taxonomy that takes into account at least a significant majority of common domain-specific tasks and forms of data. However, organization-specific variations on workflow and operational concerns are potentially indefinite. Instead of trying to document all the possible variations, it is recommended that strategies be devised for teasing out these details and for incorporating them into the problem definition during the requirement elicitation phase of a VA solution design project.

Lastly, the list of products introduced in the framework and the case study is not, and perhaps can never be, complete. This limitation, fortunately, does not majorly affect the usefulness of the evaluation framework, as long as the currently featured products do provide overlapping coverage for every variation of the dimensions. Over time, products will need to be updated in terms of features. Obsolete products will eventually need to be marked and/or removed, and new products should be added from time to time. However, the taxonomy will only change significantly by an added or updated product that provides a significant breakthrough in terms of feature set, easy of use, ease of deployment, and cost, which would render some of the old dimensions of the evaluation process irrelevant.

One possible method to effectively update this framework is to build a web-based tool that crowd-sourced the addition and evaluation of techniques, products, and toolkits, similar in concept to Wikipedia (Wales and Sanger 2001). Such a tool may be helpful both as a convenient reference of VA techniques and as a decision-making aid for VA Application designers. It is conceivable that researchers of visualization and interaction design, as well as builders of visualization products and toolkits, will be motivated to provide up-to-date and complete evaluations of their works. This concept may be explored as a next step of this project.

Bibliography

Adobe Corporation. (1998). Photoshop (version 5.0) [computer software]. Mountain View, California, U.S.

Ahlberg C., and Shneiderman B. (1994). Visual information seeking: Tight coupling of dynamic query filters with starfield displays. *SIGCHI '94*, pages 313-317. Boston, MA.

Aigner W., Miksch S., Muller W., Schumann H., Tominski C. (2008). Visual Methods for Analyzing Time-Oriented Data. *IEEE TVCG*, 14(1): 47-60.

Alsakran, J., Zhao, Y., & Zhao, X. (2010). Tile-based parallel coordinates and its application in financial visualization '. *Visualization and Data Analysis 2010 (VDA 2010)*, San Jose, California.

Amar, R., Eagan, J., & Stasko, J. (2005). Low-level components of analytic activity in information visualization. *Information Visualization*, Minneapolis, Minnesota, USA. 111-117.

Amar, R., & Stasko, J. (2004). A knowledge task-based framework for design and evaluation of information visualizations. *IEEE Symposium on Information Visualization*, Austin, Texas, USA.

Appert C., and Fekete J. D. (2006). OrthoZoom Scroller: 1D Multi-Scale Navigation. *SIGCHI 06*, pp 21-30. Montreal, Quebec.

Arias-Hernandez, R., Kaastra, L. T., Green, T. M., & Fisher, B. (2011). Pair analytics: Capturing reasoning processes in collaborative visual analytics. *44th Annual Hawaii International Conference on System Sciences*, Koloa, Kauai, Hawaii.

Bavoil, L., Callahan, S. P., Crossno, P. J., Freire, J., & Vo, H. T. (2005). VisTrails: Enabling interactive multiple-view visualizations. *IEEE Visualization 2005*, 135-142.

Becker, R. A., Cleveland, W. S., & Wilks, A. R. (1987). Dynamic graphics for data analysis. *Statis. Sci., 2*, 355-383.

Beynon-Davies, P. (2004). Database Systems. 3rd Edition. Palgrave, Houndmills, Basingstoke.

Bostock, M. (2010). Protovis API [computer software]. Retrieved April 15, 2012, from http://mbostock.github.com/protovis/

Bostock, M. (2011). D3.js API [computer software]. Retrieved April 15, 2012, from http:// http://mbostock.github.com/d3/.

Brodbeck, D., Chalmers, M., Lunzer, A., & Cotture, P. (1997). Domesticating bead: Adapting an information visualization system to a financial institution. *3rd IEEE Symposium on Information Visualization 1997 (INFOVIS '97),* Phoenix, Arizona.

Buja, A., Cook, D. and Swayne, D.F. (1996). "Interactive high-dimensional data visualization", *Journal of Computational and Graphical Statistics*, 5, 78-99.

Card, S. K., & Mackinlay, J. (1997). The structure of the information visualization design space. *IEEE Symposium on Information Visualization,* Phoeniz, AZ. 92-99.

Card, S., Mackinlay, J., Shneiderman, B., & Kaufmann, M. (1999). Chapter 1: Using vision to think. In *Readings in information visualization*.

Chang, R., Lee, A., Ghoniem, M., Kosara, R., Ribarsky, W., Yang, J., and Sudjianto, A. Scalable and interactive visual analysis of financial wire transactions for fraud detection. *Information Visualization, 1*(7), 63-76.

Chaudhuri, S., & Dayal, U.An overview of data warehousing and OLAP technology. *ACM SIGMOD, 1*(26), 65-74.

Chen, V., Alimadadi, S., Guenther, J., Kadivar, N., Lee, E., Dill, J., Woodbury, R. (2011). CZSaw [computer software]. Retrieved April 15, 2012. Available from http://czsaw.iat.sfu.ca/

Chi, E. H., & Riedl, J. T. (1998). An operator interaction framework for visualization systems. *IEEE Symposium on Information Visualization (InfoVis)*, 63-70.

Chuah M. C., Steven F. Roth S. T., Mattis J., Kolojejchick J. (1995). SDM: Selective Dynamic Manipulation of Visualizations. *UIST '95*. Pittsburgh, PA.

Chuah, M. and Roth, S. (1996). "On the semantics of interactive visualization", *IEEE Visualization (Vis'96)*, 29-36.

Cleveland S. W., McGill R. (1984). Graphical Perception: Theory, Experimentation and the Application to the Development of Graphical Models. *Journal of the American Statistical Association*. 79:387, pp. 531-554.

Cockburn A., Karlson A., and Bederson B. B. (2008). A review of overview+detail, zooming, and focus+context interfaces. *ACM Computing Surveys* 41(1).

Codd, E. F., Codd, S. B., & and Salley, C. T. (1993). Providing OLAP to user-analysts: An IT mandate. Codd & Associates.

Dang, T., & Lemieux, V. (forthcoming). Developing a functional evaluation framework of visualization techniques and products, and a case study in designing a custom visual analytics solutions for a boutique asset management firm. *Records and Information Management for Financial Analysis and Risk Management*, Frankfurt: Springer.

Dos Santos, S., & Brodlie, K. (2004). Gaining understanding of multivariate and multidimensional data through visualization. *Computer & Graphics, 28*, 311-325.

Dow Jones. (2010). Dow jones financial services taxonomy (Factiva). Retrieved April 15, 2012. Available from http://taxonomywarehouse.com.

Dwyer, T., & Eades, P. (2002). Visualizing a fund manager flow graph with columns and worms. *6th International Conference on Information Visualization (IV '02)*, London, England.

Few, S. (2006). Information dashboard design: The effective visual communication of data (1st ed.). California, U.S.: O'Reilly.

Fisher D. (2007). Hotmap: Looking at Geographic Attention. *IEEE TVCG,* 13(6), 1184-1191.

Future Point Systems, Inc. (2011). Starlight [computer software]. Retrieved April 15, 2012. Available from http://www.futurepointsystems.com/.

Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). Design patterns: Elements of reusable object-oriented software (1st ed.) New Jersey, U.S.: Addison-Wesley Professional.

Gebhardt, M., Jarke, M., & Jacobs, S. (1997). A toolkit for negotiation support interfaces to multi-dimensional data. *ACM SIGMOD International Conference on Management of Data*, Tucson, Arizona, USA, 1997.

Girardin, L., & Brodbeck, D. (2001). Interactive visualization of prices and earnings around the globe. *The 7th IEEE Symposium on Information Visualization 2001 (INFOVIS '11),* San Diego, California.

Gotz, D. and Zhou, M. (2008). Characterizing users' visual analytic activity for insight provenance. *Visual Analytics Science and Technology 2008 (VAST 2008)*, 123–130.

Gray, J., Chaudhuri, S., Bosworth, A., Layman, A., Reichart, D., & Venkatrao, M. (1997). Data cube: A relational aggregation operator generalizing group-by, cross-tab, and sub-totals. *Data Mining and Knowledge Discovery, 1*(1), 29-53.

Gunn. J. (2007). XBRL: Opportunities and challenges in enhancing financial reporting and assurance processes. *Current Issues in Auditing*. A36-A43.

Heer J., and Robertson G. G. (2007). Animated Transitions in Statistical Data Graphics. *IEEE TVCG* 13(6), 1240-1247.

Heer, J., Mackinlay, J., Stolte, C., & Agrawala, M. (2008). Graphical histories for visualization: Supporting analysis, communication, and evaluation. *IEEE Transactions on Visualization and Computer Graphics*, 14(6), 1189-1196.

Heer, J., Bostock, M., Ogievetsky, V. (2010). A Tour through the Visualization Zoo. *Communications of the ACM*, 53(6), pp. 59-67.

Holten, D. (2006). Hierarchical Edge Bundles: Visualization of Adjacency Relations in Hierarchical Data. *IEEE InfoVis 06*, 12(5), p 741–748.

Inselberg A., and Dimsdale B. (1990). Parallel Coordinates: A Tool for Visualizing Multi-Dimensional Geometry. *IEEE Visualization '90*, 361-378. California, U.S.

Johnson, B., & and Shneiderman, B. (1991). Treemaps: A space-filling approach to the visualization of hierarchical information structures. *IEEE Information Visualization*, 275-282.

Jungmeister, W. A., & Truro, D. (1992). Adapting treemaps to stock portfolio visualization. Baltimore: Center for Automation Research, University of Maryland.

Kadivar, N., Chen, V., Dunsmuir, D., Lee, E., Qian, C., Dill, J., Shaw, C., Woodbury, R (2009). Capturing and supporting the analysis process. *IEEE Visual Analytics Science and Technology*, 131-138.

Keim, D. A. and Kriegel, H. P. (1996). "Visualization techniques for mining large databases: A comparison", *IEEE Transactions on Knowledge and Data Engineering*, 8(6), 923-936.

Kernighan, B. W., & Ritchie, D. M. (1988). *The C programming language* New Jersey, U.S.A : Prentice Hall.

Khan, A. (2009). *JExcel API* [computer software]. Retrieved April 15, 2012, from http://jexcelapi.sourceforge.net/

Kimball, Ralph and Ross, Margy. (2002). The Data Warehouse Toolkit Second Edition, John Wiley and Sons, Inc.

Klein, P., Muller, F., Reiterer, H., & Eibl, M. (2002). Visual information retrieval with SuperTable + scatterplot. *6th International Conference on Information Visualization (IV '02),* London, England.

Kosslyn, S. M. (1980). Image and mind (1st ed.). Cambridge, MA: Harvard University Press.

Kosslyn, S. M., & Jacobs, R. (1994). Visual mental images in the brain. In M. J. Farah, & G. Ratcliff (Eds.), *The neuropsychology of high-level vision: Collected tutorial essays*. Hillsdale, NJ: Lawrence Erlbaum.

Kubelec, C., & Sa, F. (2010). The geographical composition of national external balance sheets: 1980-2005: Bank of england working paper no. 384. Bank of England.

Lei, S. T., & Zhang, K. (2010). A visual analytics system for financial time-series data. *3rd International Symposium on Visual Information Communication,* New York, NY.

Lemieux, V., Fisher, B., & Dang, T. (forthcoming). The visual analysis of financial data. In e. a. Mark Flood (Ed.), *Handbook of financial risk data*. Cambridge, UK: Cambridge University Press.

Levy, A. (1979), *Basic Set Theory*, Berlin, New York: Springer-Verlag.

Light, A. (2011). Democratising technology: Making transformation using designing, performance and props. *Proceedings of the 2011 Annual Conference on Human Factors in Computing Systems,* Vancouver, BC, Canada. 2239-2242.

Ieronutti, L., & Pighin, M. (2009). A novel visualization and interaction technique for exploring multidimensional data. *3/4* (4), 350-374.

Lurie, N. H., & Mason, C. (2007). Visual representation: Implications for decision making. *Journal of Marketing, 71*.

MacEachren A. M., Boscoe A. M., Haug F. P., Pickle D. (1998). Geographic visualization: designing manipulable maps for exploring temporally varying georeferenced statistics. *Information Visualization*. 87-94.

Maniatis, A. S., Vassiliadis, P., Skiadopoulos, S., & Vassiliou, Y. (2003). Advanced visualization for OLAP. *The 6th ACM International Workshop on Data Warehousing and OLAP*, New Orlean, LA.

Mansmann, S., Mansmann, F., Scholl, M. H., & Keim, D. (2007). Hierarchy-driven visual exploration of multidimensional data cubes. *BTW*, 96-111.

Microsoft Corporation. (1997). Multidimensional expressions (MDX) [computer software]. Redmond, WA.

Microsoft Corporation. (1998). OLAP services, part of SQL server (version 7) [computer software]. Redmond, WA.

Microsoft Corporation, & Hyperion. (2001). XML for analysis (XMLA) [computer software]. Redmond. WA.

Microsoft Corporation. (2011). Menus. Retrieved April 15, 2012, from http://msdn.microsoft.com/en-us/library/aa511502.aspx

Miller, G. A. (1956). The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review*, 2(63), 81-97. doi:10.1037/h0043158.

Munzner, T. (2009). A nested model for visualization design and validation. Paper presented at the *Information Visualization*, Atlantic City, NJ. , *15*(6) Retrieved April 15, 2012, from http://www.cs.ubc.ca/labs/imager/tr/2009/NestedModel/NestedModel.pdf.

Munzner, T. (2009). Visualization. In P. Shirley, & et al. (Eds.), *Fundamentals of graphics, third edition* (pp. 675-707) AK Peters.

Neumann, P., Tang, A., & Carpendale, S. (2007). A framework for visual information analysis. Technical Report 2007-87123, University of Calgary, Calgary, AB, Canada.

Nison, S. (2001). Japanese candlestick charting techniques, second edition (2nd ed.) Prentice Hall Press.

North, C., Chang, R., Endert, A., Dou, W., May, R., Pike, B., & Fink, G. (2011). Analytic provenance: Process+interaction+insight. *The 2011 Annual Conference Extended Abstracts on Human Factors in Computing Systems*, Vancouver, BC, Canada. 33-36.

Pacific Northwest National Laboratory. (2011). In-Spire [computer software]. Retrieved April 15, 2012. Available from http://in-spire.pnnl.gov/.

Panopticon Software, Inc. (2010). Panopticon [computer software]. Retrieved April 15, 2012. Available from http://www.panopticon.com.

Plumlee M. D., and Ware C. (2006). Cognitive costs of zooming versus using multiple windows. *ACM Trans. Applied Perception (TAP)* 13(2), 1-31.

Quantum4D, Inc. (2011). Quantum4D [computer software]. Retrieved April 15, 2012. Available from http://www.quantum4d.com.

Qin, C., Zhou, C., & Pei, T. (2003). Taxonomy of visualization techniques and systems - concerns between users and developers are different. *Asia GIS Conference* 1-14.

Ramnath, S., Rock, S., & Shane, B. P. (2008). The financial analyst forecasting literature: A taxonomy with suggestions for further research. *International Journal of Forecasting, 24.*

Rao R., and Card S. K. (1994). The Table Lens: Merging Graphical and Symbolic Representations in an Interactive Focus + Context Visualization for Tabular Information. *SIGCHI '94*, pp. 318-322.

Raskin, J. (2000). The Humane Interface. Addison Wesley. Boston: MA.

Risch, J., Kao, A., Poteet, R. S., & and Wu, J. Y. (2008). Text visualization for visual text analytics. In J. S. Simoff, H. M. Bohlen & A. and Mazeika (Eds.), *Visual data mining* (pp. 154-171). Frankfurt: Springer.

Roberts, P. (2003). *Information visualization for stock market ticks: Toward a new trading interface*. (Master Dissertation). Retrieved April 15, 2012, from http://dspace.mit.edu/handle/1721.1/16668.

Roberts J. C. (2007). State of the Art: Coordinated & Multiple Views in Exploratory Visualization. *Conference on Coordinated & Multiple Views in Exploratory Visualization (CMV)*. Washington, DC.

Russell, D. M., Stefik, M. J., Pirolli, P., & Card, S. K. (1993). The cost structure of sensemaking. *INTERCHI '93 Conference on Human Factors,* Amsterdam.

SAP.Crystal dashboard design, formerly xcelsius [computer software]. Retrieved April 15, 2012. Available from http://www.sap.com/solutions/sap-crystal-solutions/dashboards-visualization/index.epx.

Savikhin, A., Lam, H. C., Fisher, B., & Ebert, D. S. (2011). An experimental study of financial portfolio selection with visual analytics for decision support. *International Conference on System Sciences,* Hawaii.

Savikhin, A., & Maciejewski, R. (2008). Applied visual analytics for economic decision-making. *Visual Analytics Science and Technology*, Braga, Portugal.

Sawant, A. (2009). StockViz: Analyzing the trend of stocks in major auto, oil, consumer, and technology companies. *The 2009 International Conference on Modeling, Simulation & Visualization Methods (MSV 2009),* Las Vegas, Nevada.

Securities and Exchange Commission (SEC). (2008). *SEC approves interactive data for financial reporting by public companies, mutual funds.* Retrieved April 15, 2012, from http://www.sec.gov/news/press/2008/2008-300.htm.

Shneiderman, B. (1996). The eyes have it: A task by data type taxonomy for information visualization. *IEEE Symposium on Visual Languages*, Washington, DC.

Shneiderman, B., & Plaisant, C. (2006). Strategies for evaluating information visualization tools: Multi-dimensional in-depth long-term case studies. Paper presented at the *AVI Workshop on Beyond Time and Errors: Novel Evaluation Methods for Information Visualization (BELIV)*, 38-43. Retrieved April 15, 2012, from http://hcil.cs.umd.edu/trs/2006-12/2006-12.pdf.

Tableau Software, Inc. (2011). Tableau [computer software]. Retrieved April 15, 2012. Available from http://www.tableausoftware.com/.

TD Waterhouse. (2011). Thinkorswim [computer software]. Retrieved April 15, 2012. Available from http://thinkorswim.com.

Techapichetvanich, K., & Datta, A. (2005). Interactive visualization for OLAP. *International Conference on Computational Science*, 206-214.

Tegarden, D. P. (1999). Business information visualization. *Communications of the Association for Information Systems*, 1(1es). Retrieved April 15, 2012, from http://dl.acm.org/citation.cfm?id=374109.374113

Thomas, J. J., & Cook, K. A. (2005). Illuminating the path: The research and development agenda for visual analytics. *IEEE CS*, Retrieved April 15, 2012, from http://nvac.pnl.gov/agenda.stm#book

Tory, M. and Möller. T. (2002). A Model-Based Visualization Taxonomy. Retrieved April 15, 2012, from http://citeseer.nj.nec.com/564142.html.

Tversky, B., Julie, Morrison J., Betrancourt, M. (2002). Animation: Can It Facilitate? *International Journal of Human Computer Studies* 57:4, pp 247-262.

Wales, J., Sanger, L. (2001) Wikipedia: About. Retrieved May 4, 2012, from http://en.wikipedia.org/wiki/Wikipedia:About

Ward, M., Grinstein, G., & Keim, D. (2010). Interactive data visualization: Foundations, techniques, and applications. A.K. Peters Ltd.

Wattenberg, M. (1999). Visualizing the stock market. *The ACM SIGCHI Conference on Human Factors in Computing Systems (CHI '99)*, Pittsburgh, Pennsylvania.

Weaver, C. (2004). Building Highly-Coordinated Visualizations In Improvise. *IEEE InfoVis*, Washington, DC.

Wegman, J. E. (1990). Hyperdimensional Data Analysis Using Parallel Coordinates. *Journal of the American Statistical Association*, Vol. 85, No. 411, pp. 664-675.

Wehrend, S., & Lewis, C. (1990). A problem-oriented classification of visualization techniques. *The 1st IEEE Conference on Visualization (Visualization '90)*, San Francisco, California.

Wong N., Sheelagh M., Carpendale T., Greenberg S. (2003). EdgeLens: An Interactive Method for Managing Edge Congestion in Graphs. *IEEE InfoVis*, pp 51-58, Seattle, WA.

Wongsuphasawat, K., Guerra Gomez, J. A., Plaisant, C., Wang, T. D., Taieb-Maimon, M., & Shneiderman, B. (2011). LifeFlow: Visualizing an overview of event sequences. *Proceedings of the 2011 Annual Conference on Human Factors in Computing Systems*, Vancouver, BC, Canada.

Wood J., Dykes J., Slingsby A., and Clarke K. (2007). Interactive Visual Exploration of a Large Spatio-temporal Dataset: Reflections on a Geovisualization Mashup. *IEEE TVCG*, 13(6), 1176-1183.

Wright, W. (1995). Information animation applications in the capital markets. *The 1st IEEE Symposium on Information Visualization 1995 (INFOVIS '95)*, Atlanta, Georgia.

Yang, J., Ward, M., Rundensteiner, E. A., & Patro, A. (2003). InterRing: A visual interface for navigating and manipulating hierarchies. *IEEE Information Visualization*, 16-30.

Yin, R. K. (2009). *Case study research: Design and methods* (4th ed.). CA, USA: Sage Publications.

Zhao J., Chevalier F., Pietriga E., and Balakrishnan R. (2011). Exploratory Analysis of Time-series with ChronoLenses. *IEEE TVCG*, 17(12), 2422-2431.

Zhou, M., & Feiner, S. (1998). Visual task characterization for automated visual discourse synthesis. *The ACM SIGCHI Conference on Human Factors in Computing Systems (CHI '98)*, Los Angeles, California.

Ziegler, H., Nietzschmann, T., & Keim, D. A. (2008). Visual analytics on the financial market: Pixel-based analysis and comparison of long-term investments. *The 12th International Conference on Information Visualization 2008 (IV '08)*, London.

Appendices

Appendix A Supplemental Materials for the Fixed Income VA Case Study

A.1 Fixed-Income Analyst's Quick Start Guide to Using the Prototype

FIVA version 0.9 is a research prototype of a system designed to support the work of an asset manager / financial analyst. This is not a general-purposed tool, as the back-end logic is firm-specific, nor is it full-featured, as this is an on-going research project. Version 0.9 of the prototype is designed to support two specific tasks:

1) **Portfolio Monitoring:** this task is supported by the user-interface to manage the list of clients and accounts, and the treemap visualization showing the performance measure of the currently selected client / account. The performance measure is the Cost of Outstanding Income value, including miscellaneous assets when necessary.

    - The search box searches the list of client files and highlights the result
    - The client list shows all the client portfolios, each of which corresponds to an Excel workbook
    - The account list shows all the accounts within the selected client.
    - The treemap can show either
        - All the clients
            - The size of the box in the map is the Cost of Outstanding Income (COI) Summary of each account within each client. This includes all the extra assets, equivalent to the summary values in the excel sheets
            - The color of the box indicates either surplus (green) or deficit (red).
            - While the excel sheet has negative as surplus and positive as deficit, the FIVA interface shows only the absolute value, the size of the surplus/deficit, and uses the color to indicate the state.

- One selected client
    - This view focuses on one client, while still showing the COI Summary
    - This was intended as a way to show the interface to the client while preserving privacy
- One selected account
    - This view shows the COI for all the years in one account sheet, as well as any miscellaneous assets

2) **Scenario Analysis:** this task can be done via editing the tables of **income goal and holdings** and **price and yield**, and observing the changes on the bullet chart visualization and the treemap.

- The price/yield table is common to all the accounts. However, only the price/yield in relevant years to the selected account is shown.
- If the price/yield table shows "N/A" for certain years, that indicates that the account data actually started before the first year in the price/yield table.
- The bullet chart maps the income goal and holdings values directly. The grey bar shows income goal, and the blue bar shows the holding.

Limitations / Bugs

- Whenever a new account or client is selected, the data is reloaded from disk, so any change made on the interface is temporary.
- The changes also cannot be saved back to disk currently.
- Printing the page currently does not reveal the colors on the treemap, unless the PDF printer is available, so the treemap cannot currently be attached to client reports. However, the running interface can still be shown during client meetings.

A.2 Fixed-Income Analyst's Prototype Evaluation Survey with Results

Portfolio Monitoring

1) Does the interface offer you (a financial analyst) an improvement in monitoring the list of clients, their performance, and holdings, compared to the existing Excel-based interface?

FIVA significantly improves the speed with which we see the impact of movements of the yield curve on client's surpluses/deficits.

2) For the task of monitoring clients, would you prefer using the existing interface or FIVA, and why?

FIVA can't replace the excel-based work we do for clients since there is a great amount of detail involved in monitoring each client's portfolio. However, FIVA can be useful in identifying quickly when there is a momentum for trades in the different accounts and this is very valuable.

Scenario Analysis

1) Does the interface offer you an improvement in analyzing the effect of hypothetical changes to the data on the performance of the clients?

Yes, as it is described in items one above and two above.

2) For the task of scenario analysis, would you prefer using the existing interface or FIVA, and why?

If the only variable in the scenario analysis is the yield curve, FIVA will be faster and will enable to see all accounts at once.

Overall Quality of user experience

1) Please elaborate what you like about the FIVA interface and interaction

Answer: The effect of changes in the variable can be seen instantly in each account and in all accounts of one client or all clients.

2) Please elaborate what you dislike about the FIVA interface and interaction

Answer: Nothing

3) Please elaborate what you feel is missing from FIVA that you would have wanted:

    a. In terms of the interactions and "level of polish" in the existing features?

Answer: It would be good to have all surpluses/deficits in one screen or easier to read the actual numeric result. I suggest having each account's name listed by alphabetical order followed by each account's surplus/deficit in green/red. The area to the right of the screen containing details of one account is useful when zooming in to see one single account; but it is not needed when viewing an entire client portfolio or all the accounts at once.

    b. In terms of new features?

Answer: It would be useful to create a "search" feature for a specific holding. Let's say we want to know what accounts hold a coupon maturing in 2030; or we want to know who owns a specific stock, etc. we can enter in the "search" box the holding instructing FIVA to provide a list of the accounts that hold that security. It would also be useful having some kind of system that tracks historic results per client.

Evaluate potential benefits to client presentations

Author's note: These questions were not answered by the analyst.

1) (Assuming that the FIVA prototype will be loaded with the latest data of several clients,) Do you feel that the client would benefit from being shown the interface and visualizations instead of the Excel data?

2) (After you have shown the FIVA interface and visualizations to the client,)

a. Did you observe if the client found the visualizations easier to understand than the Excel data?

b. Did you find the visualizations easy to explain to the clients?

A.3 Fixed-Income Analyst's Background Interview Questions

Time: 1 to 2 hours

Agenda

- Project proposal presentation: 30 min
- Questions for the Analyst: 30 min
- Observation of analysts on specific target processes: 1 hour or less

Questions for the Analyst

1. Confirm the overarching assumption that within the sample data workbooks, each "sampleX" workbook represents one *client account*.
2. Confirm the overarching assumption that within each client account are one-or-more *investment accounts*, which can be NR, RRSP, SP-RSP, and LIRA/ LRSP.
3. Did we miss any other possible investment account type?
4. Confirm the assumption that the holding listings with the RBC security logo on top and several charts at the end are summaries of the one-or-more investment accounts within each client account. We did not have these spreadsheets in digital format, but we have paper copies of several printouts that have the same names as found in Sample 1.
5. Apart from the *RBC holding sheets*, are there any other spreadsheets or other data files related to each client that we need? (If yes, could we have these spreadsheets for sample client 1, 2, 3?)

6. Supposed that we are the clients of sample 1 (PRC, R-RRSP, J-RRSP, J-SP-RSP), please guide us through the explanations in a client meeting.

7. We found that the three sample workbooks all contain sheets with different names. What are the common elements in these workbooks? (Common Sheets? Common Formulae? Note: we'll use the printout on-site as visual aid to find these commonalities)

8. Miscellaneous clarifications:

    a. PRC, R-RRSP, J-RRSP, J-SP-RSP, NR, LIRA.etc.

    b. Spousal RSP can coexist with the RRSP account of the spouse?

    c. Are the coupons in the holdings strip coupons?

    d. Does the Firm invest exclusively in strip coupons?

On this final set of questions, we are attempting to zero in on the elements of the two workflows—scenario analysis and client reporting—that would benefit the most from visualizations.

9. What is the calculation(s) that analysts do most often at the Firm when showing a client's account to the client?

10. What are the common calculations that are performed for all of the client workbooks when performing scenario analysis?

11. Among these common-denominator calculations, please identify those whose results are meant for human analysis (not an intermediate steps to other formula(e))? The results of these particular calculations would benefit from a Visual Analytics approach, as human can process visualizations quicker than numbers, and usually produce clearer and richer insights as well.

(Note: the calculation(s) identified in Q9 and Q11 may actually be one and the same)

Observation of analysts on specific target processes

1. Observe an analyst performing a sample full chain of calculations leading up to the insights and recommendations in the client report (Q9 above)

2. Observe an analyst performing the final calculation(s) in a scenario analysis task. (Q11 above)

3. Identify critical *sheets* and *fields* from these calculations.

    a. What are the factors used in scenario analysis?

    b. Which factors are uncertain in nature? E.g. possible actions taken by client, projections of interest rate changes (Does the Firm project future changes in market/economic conditions in its analysis?)

4. Go through the sample workbooks with an analyst; inquire about the roles of the common sheets and their relationships with the other sheets in each of the three samples.

    a. Even though these common sheets (especially the account income stream & goal sheets) have different names in the three samples, are the sheets supposed to be exactly the same in format?

    b. Are there any other special markers for identifying types of sheets in the sample workbooks?

    c. The sheets in the sample workbooks are colour-coded into groups, what do these groups mean?

    d. Which sheets in all of the sample workbooks can we safely ignore? (Intermediate data or calculations that will be reflected in other final *sheets* and *fields*)

5. Match up the value of the fields in the *RBC holding sheets* and the fields in the equivalent sheets in the sample1.xls workbook. (Note: different settlement date)

6. The provincial yield curve chart printout is not found in the files; where does it come from?

7. Clarify some observations on this yield curve chart with an analyst:

    a. The line representing a coupon seems to go on pass its maturity.

    b. There seem to be only 7 lines on the chart, but there are 10 entries in the legend.

A.4 Mental Model Diagrams and Mock-ups of Alternative Layouts and Visualizations

This section contains

- A list of diagrams that were drawn during and after the design interviews to show the application designer's understanding about the domain / firm-specific problem, and to verify this understanding with the analyst.

- A list of mock-ups, in computer-graphic, of the alternative design directions considered. The mock-ups were designed with Visio, with some basic interactivity provided by PowerPoint.

- The paper mock-ups created spontaneously during the interviews were also rendered as computer-graphic mock-ups in PowerPoint later. These cleaned-up PowerPoint mock-ups were shown to the analyst again, to verify the assumptions and observations that resulted from the paper prototyping.

Figure 10 and 11 illustrates the mental model of the portfolio monitoring and scenario analysis tasks. Figure 10 is the current process, simple but very ad hoc, and contains bottleneck sub-processes that take a lot of manual labour, cognitive-load, and are error-prone. Figure 10 is the desired process, where the bottleneck sub-processes are broken down

into atomic elements that can be supported by software features, which hopefully would result in improved efficiency and usability.

Figure 10 Analytic Process Mental Model - Current

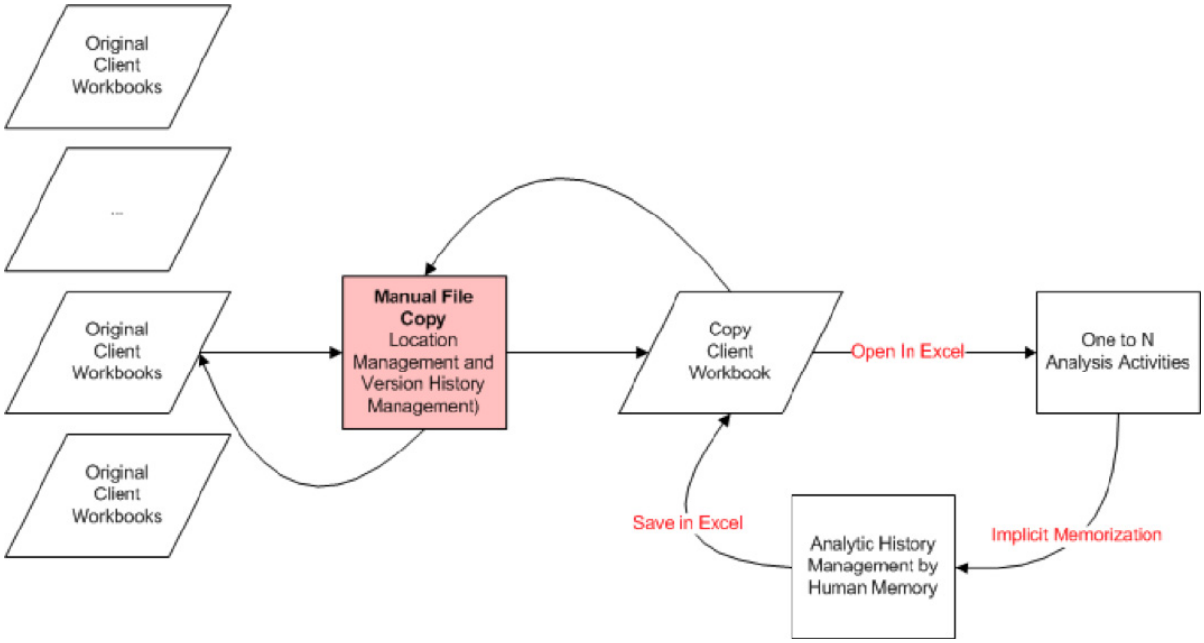


Figure 11 Analytic Process Mental Model - Desired

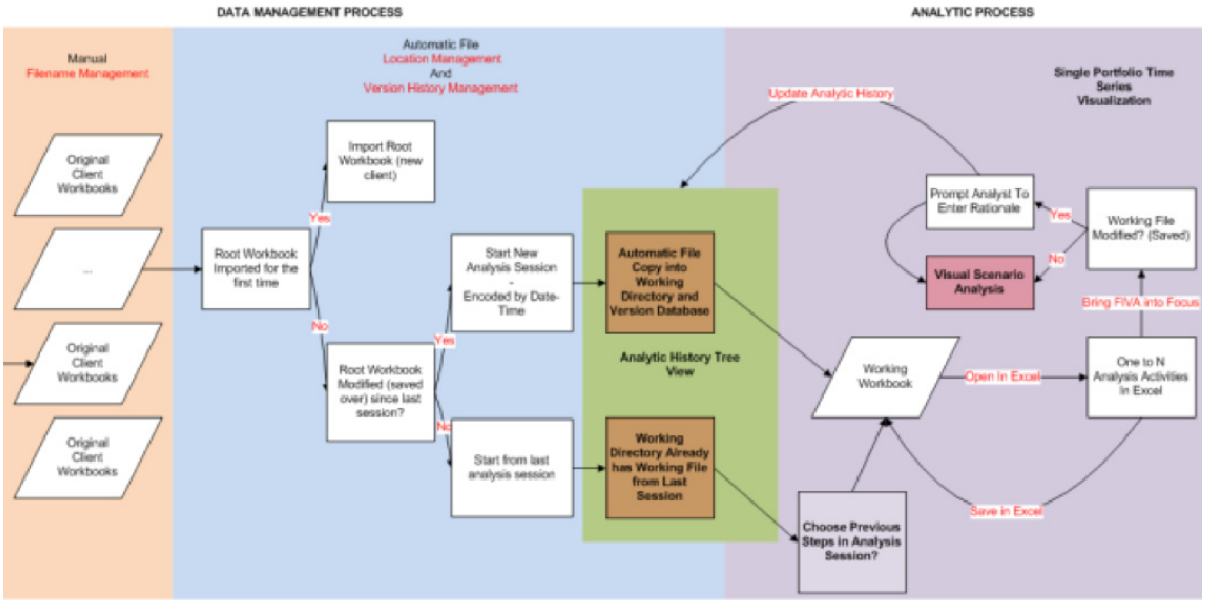


Figure 12, 13, 14, 15, 16 are alternative interface layout mockups, in five iterations.

Figure 12 Alternative Layout Mockup Iteration 1

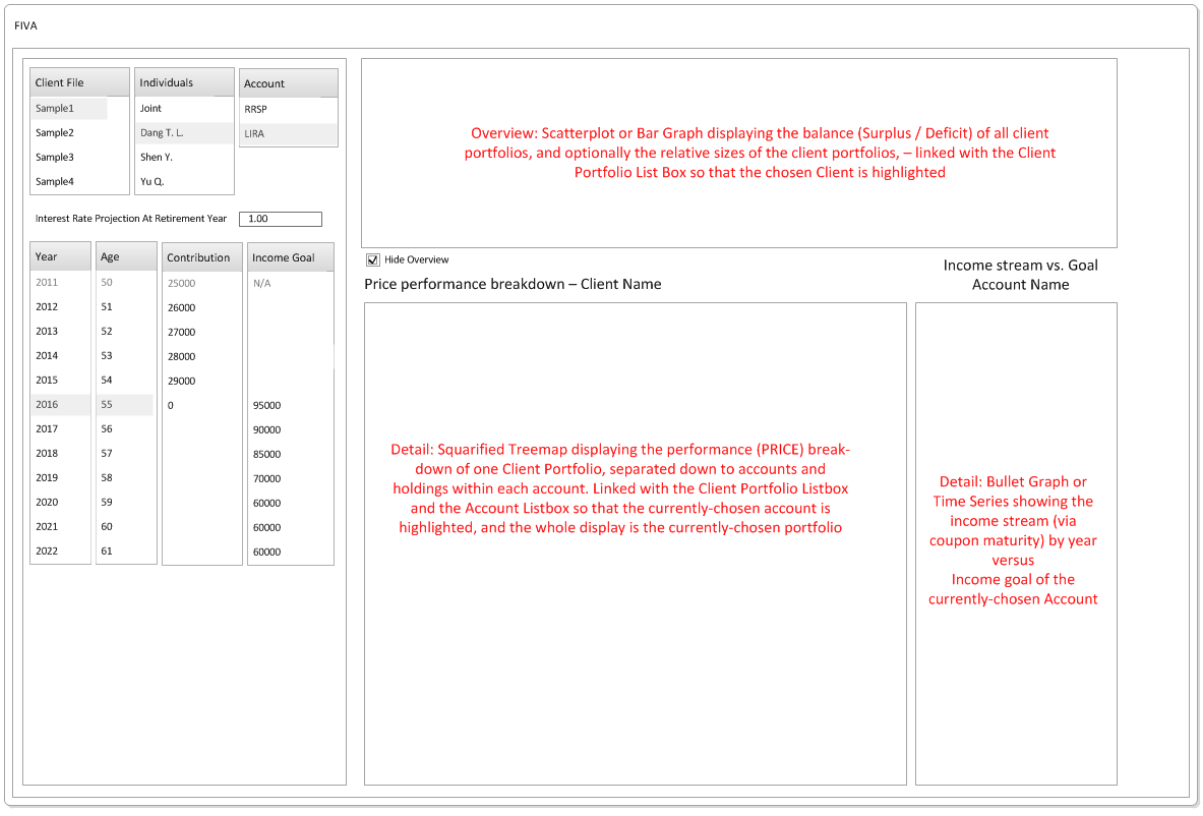


Figure 13 Alternative Layout Mockup Iteration 2

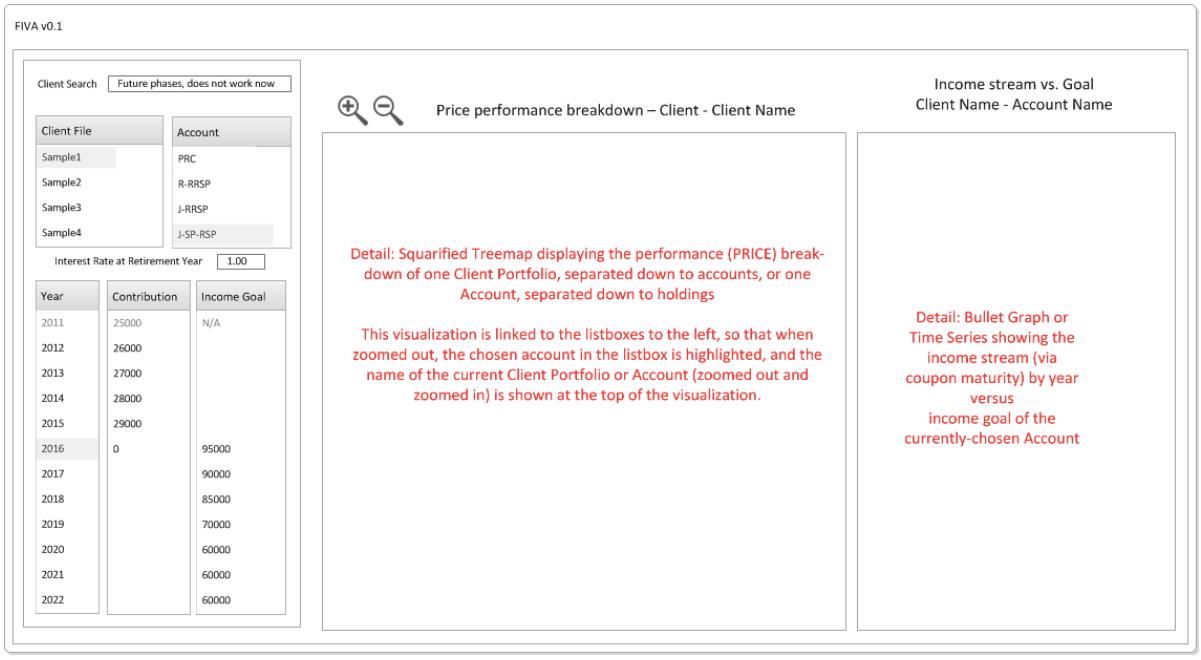


Figure 14 Alternative Layout Mockup Iteration 3

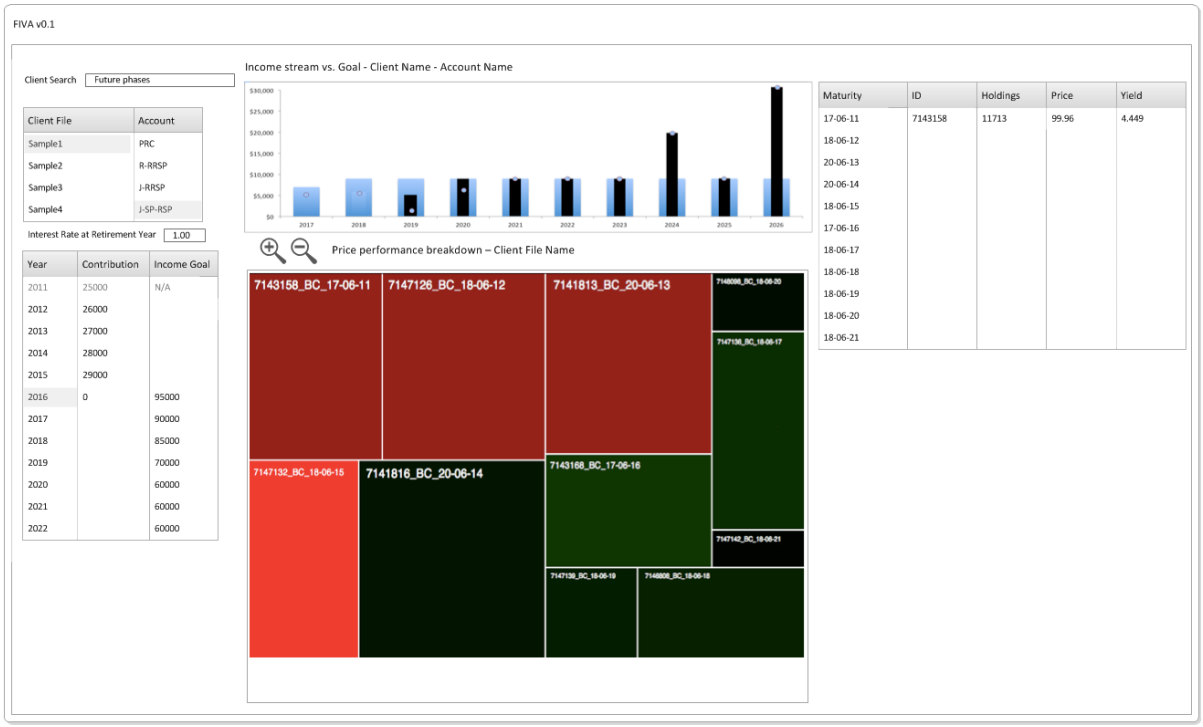
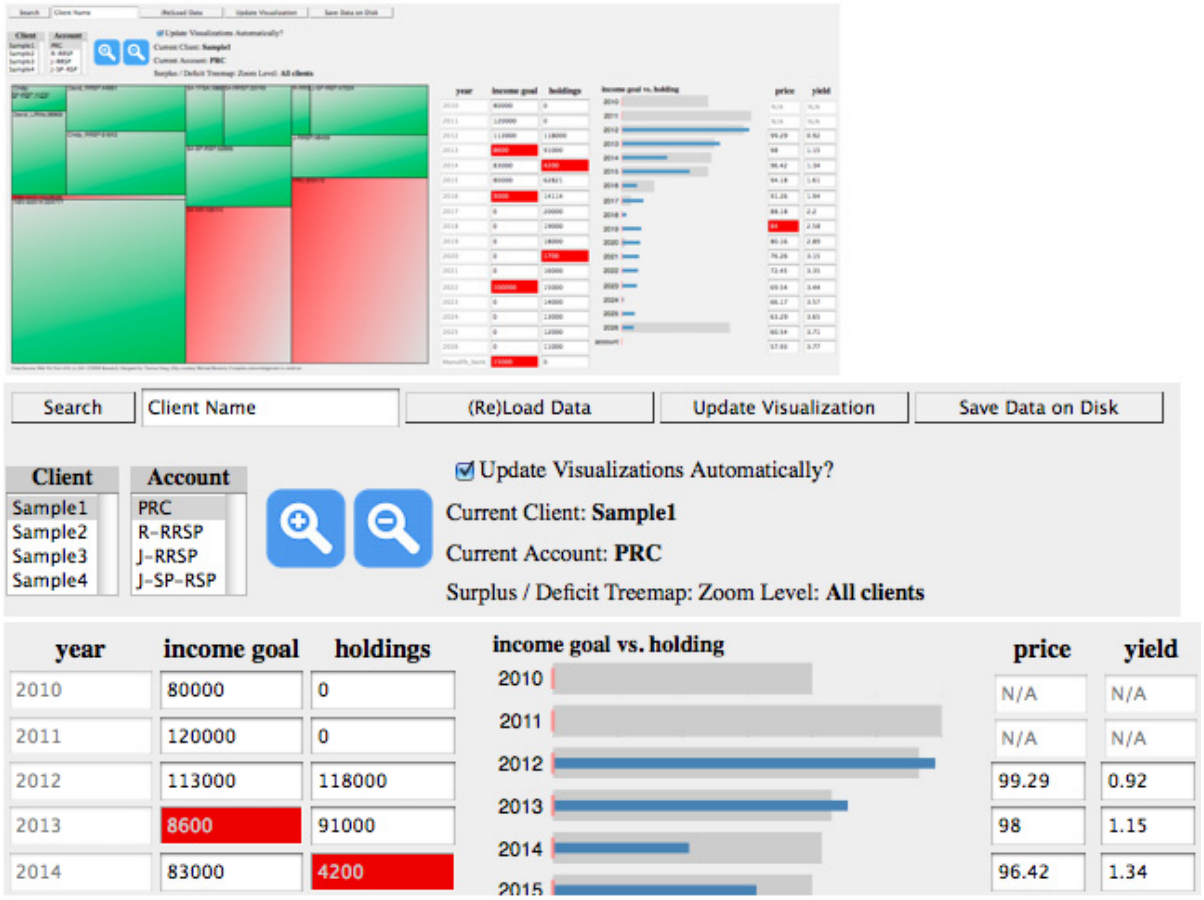


Figure 15 Alternative Layout Mockup Iteration 4

Figure 16 (duplicate of Figure 6) shows the final layout that was implemented as a high-fidelity prototype. The final layout is close to the iteration 4 mockup, but contains more redundant information on the display regarding the program's interactive state and the data being shown.

Figure 16 Fixed Income Visual Analysis Dashboard – Iteration 5 – Final Layout



Appendix B Supplemental Materials for the Evaluation Framework

B.1 Template for Deconstructing Domain-specific Analytic Tasks into Atomic Cognitive and Perception Tasks

	Domain Task 1	Domain Task 2
Filter / Locate		
Compute Derived Value		
Find Extrema		
Sort / Rank		
Determine Range		
Characterize Distribution		
Find Anomalies		
Cluster / Associate		
Correlate		
Compare through time		
Compare between locations		
Compare historical states		
Identify		
Compare between entities		
Categorize		

Legend:

X: Deconstructed into
Blank: Not applicable

B.2 Template for Evaluating Other Visualization Techniques in the Design Phase

		Visualization 1	Visualization 2
Low-Level Tasks From Table 1	Filter / Locate		
	Find Extrema		
	Determine Range		
	Characterize Distribution		
	Find Anomalies		
	Cluster / Associate		
	Correlate		
	Compare through time		
	Compare between locations		
	Compare Historical States		
	Identify		
	Compare between entities		
	Categorize		
Time-frame of Analytic Activity	Real-time transition		
	Rapid state comparison		
Visual Scalability	1-100 data point		
	Hundreds of data points		
	Thousands of data points		
Level of Detail of Value Retrieval	Raw values shown		
	Quantitative estimates		
	Qualitative estimates		

		Visualization 1	Visualization 2
Unit Data Type	Cardinal		
	Ordinal		
	Interval		
	Nominal		
	Structured Text		
	Unstructured Text		
Data Model & Organization	Bi-variate		
	Hierarchical (Tree-like)		
	Network		
	Multi-variate (Flat, Table-like)		
	Unstructured Collection (Files)		

Legend:

Blank: Not suitable for
"-": Can accommodate but not preferable
"X": Suitable for

B.3 Template for Evaluating Other Interaction Techniques in the Design Phase

		Interaction 1	Interaction 2
Low-Level Tasks From Table 1	Filter / Locate		
	Find Extrema		
	Determine Range		
	Characterize Distribution		
	Find Anomalies		
	Cluster / Associate		
	Correlate		
	Compare through time		
	Compare between locations		
	Record Historical States		
	Identify		
	Compare between entities		
	Categorize		
Time-frame of Analytic Activity	Real-time transition		
	Rapid state comparison		
Visual Scalability	1-100 entities		
	Hundreds of entities		
	Thousands of entities		

Legend:

Blank: Neutral
X: Enhances or Enables

B.4 Template for Evaluating Other Visualization / Visual Analytics Products and API's in the Implementation Phase

		Another Product	Another API
Visualizations 1-N From Table 2A	Parallel Coordinates		
	Scatterplot		
	Dot Plot		
	Bullet Graph		
	Line Graphs		
	Stack Graphs		
	Bar Graphs		
	Candlestick		
	Heatmap		
	Treemap		
	RSF (ringmap, sunburst)		
	Horizon Graph		
	Pie Charts		
	Adjacency Diagrams		
	Node-link		
	Time Series		
	Geocoding (programmable with GDAL)		
	Unstructured Text Clustering		
	Glyphs		
	Tables of Values		
Interactions 1-N From Table 2B	Search and Highlight		
	Cursor Highlight (Picking)		
	Semantic Zooming		
	Graphical Panning & Zooming		
	Overview & Detail		
	Focus & Context		
	Linking & Brushing		
	Pop-up & Tooltip (on the visualizations)		
	Direct Manipulation of Data		
	3D View (rotate, zoom, fly-by)		
	Matrix of visualization instances		

		Another Product	Another API
Data Semantic	Domain-agnostic		
	Designed for finance		
	Designed for a sub-domain in finance		
Data Storage & Transmission Format	Built-in, unmodifiable data		
	VA-product-vendor specific		
	Excel, Microsoft specific		
	Delimited value spreadsheets (CSV, etc.)		
	Word, Microsoft specific		
	PDF, Adobe specific		
	XML, user-customizable schema		
	XML, product specific schema		
	XBRL		
	Row-oriented Database (Oracle, etc.)		
	Column-oriented Database (Sybase IQ, etc.)		
	OLAP		
	JSON		
	Customizable text format		
	Java Message Service (ActiveMQ, SonicMQ)		
	Advanced Message Queuing Protocol (QPID)		
Data Mining Features	RSS feed		
	Email server retrieval		
	Web crawler retrieval		
	Search engine retrieval		
Data Update Speed	Manual Input		
	Pull Update (interval refresh)		
	Push Update (best effort "true" real-time)		
Data Processing Features	Clean & normalize heterogeneous or bad data		
	Entity selection		
	Entity extraction		
	Sentiment Analysis		
	Statistical Analysis		
	Scalable to 10000+ records		

		Another Product	Another API
Data Export and History	Preserving format and schema of input data		
	Using application-specific format and schema		
	Containing both external and application-specific semantics		
	Containing only application-specific semantics		
	Historical Data Persists Across Sessions		
Analytic Process Documentation	Recording Program Events		
	Recording Analytic Actions		
	Journaling and Annotation Features Present		
	Analytic Process Information Persists Across Sessions		
Processes & Workflow Integration	Dashboard building		
	Excel UI integration		
	Online sharing of visualizations		
	Exporting & printing visualizations		
Deployment & Expertise Acquisition	Availability (1: Yes, 0: No)		
	Licensing (1: Free to use, 2: Commercialized, 3: Free for Educational Use)		
	Tech support and documentation (0: None, 1: Free, 2: Paid, 3: Both)		
	Documentation Helpful? (1: Not at all helpful - to - 5: Extremely Helpful)		

Legend:

Blank: not supported at all
0: programmable from scratch only
1: programmable, relevant API available
2: supported outright, requires customization