A Multi-Tier Framework for Dynamic Data Collection, Analysis, and Visualization

by

Xian Ke

Submitted to the Department of Electrical Engineering and Computer Science

in Partial Fulfillment of the Requirements for the Degree of

Master of Engineering in Electrical Engineering and Computer Science

at the Massachusetts Institute of Technology

May 20, 2004

Author _____

Department of Electrical Engineering and Computer Science
May 20, 2004

Certified by _____

Andrew W. Lo
Harris & Harris Group Professor
Thesis Supervisor

Certified by _____

Dmitry Repin
Postdoctoral Associate
Thesis Supervisor

Accepted by _____

Arthur C. Smith
Chairman, Department Committee on Graduate Theses

A Multi-Tier Framework for Dynamic Data Collection, Analysis, and Visualization

by

Xian Ke

Submitted to the
Department of Electrical Engineering and Computer Science

May 20, 2004

in Partial Fulfillment of the Requirements for the Degree of
Master of Engineering in Electrical Engineering and Computer Science
at the Massachusetts Institute of Technology

ABSTRACT

This thesis describes a framework for collecting, analyzing, and visualizing dynamic data, particularly data gathered through Web questionnaires. The framework addresses challenges such as promoting user participation, handling missing or invalid data, and streamlining the data interpretation process. Tools in the framework provide an intuitive way to build robust questionnaires on the Web and perform on-the-fly analysis and visualization of results. A novel 2.5-dimensional dynamic response-distribution visualization allows subjects to compare their results against others immediately after they have submitted their response, thereby encouraging active participation in ongoing research studies. Other modules offer the capability to quickly gain insight and discover patterns in user data. The framework has been implemented in a multi-tier architecture within an open-source, Java-based platform. It is incorporated into Risk Psychology Network, a research and educational project at MIT's Laboratory for Financial Engineering.

Thesis Supervisor: Andrew W. Lo
Title: Harris & Harris Group Professor

Thesis Supervisor: Dmitry Repin
Title: Postdoctoral Associate

Acknowledgements

For the supervision of this project, I am greatly indebted to the following individuals at the Laboratory for Financial Engineering:

- Prof. Andrew Lo – for his vision and dedication
- Dr. Dmitry Repin – for his considerable guidance and patience
- Dr. Bradley Smith – for his experience and technical discussions
- Svetlana Sussman – for her dedication and support

This thesis marks the end of my five-year journey here at MIT. I chose the following quote upon completion of my bachelor's studies last year, and it holds ever true in this past year:

"Education is a progressive discovery of our ignorance".
- Will Durant, 1885-1981

It is not often that simply by being in the company of someone make one strive to be a better person. From my time in LFE and at MIT, I am grateful to know individuals for which this is true, and who kindly share their perspectives on work, life, and beyond.

Last but not least, my parents deserve thanks for bearing with me the longest out of anyone. Their perseverance and dedication placed a world of opportunity within my reach, and provided me with inspiration to pursue those opportunities.

Table of Contents

List of Figures and Tables

1 Introduction

The ability to gather, analyze, and present data is integral across many disciplines. Companies rely upon market research in their continual pursuit of products and services that meet customer needs. No scientific research paper is complete without quantitative measures of outcome and progress.

The development of computer technologies for storage and analysis has dramatically reduced the time and effort required to derive results from data. At the same time, the rise in popularity of the Internet introduces a completely "paper-less" data collection alternative, especially in the domain of self-administered questionnaires. Recent studies indicate that more than half of households in the United States are connected to the Internet [2]. Virtually every college student and white-collar worker has regular access to email and the Web, making data collection via the Internet highly efficient in those domains.

Once data has been gathered, a variety of software exists at a researcher's disposal to gain insight from the mass of information. These range from database systems to spreadsheet programs to complex analysis packages. Sheer technological capability is not the limitation of data analysis today. However, it is often difficult to find the right combination of tools and resources that can interpret data quickly and efficiently, especially if the data needs to first go through significant processing and transformations. Furthermore, analysis and visualization capabilities are generally not available to participants of the research studies, since they do not have immediate access to the stored data nor the tools to interpret it.

The purpose of this thesis project is to create a Web-based framework for collecting, analyzing, and visualizing data. Specifically, the focus will be on data from self-administered questionnaires, though the analysis and visualization tools can be used on any data stored in appropriate format. The implementation of the framework within a multi-tier architecture allows flexibility and robustness. Users will have a valuable tool to build questionnaires and derive preliminary understandings of collected data. Research subjects will have the ability to compare themselves against others in the study, gaining immediate feedback that will encourage continued active participation.

This thesis is divided as follows. Chapter 2 explains how the framework emerged from the context of a data-centric research group at MIT. Chapter 3 provides background information on data collection via the Web and some existing statistical analysis and visualization interfaces at researchers' disposal. Chapter 4 explains the design principles that guide the framework architecture. Chapter 5 provides an overview of the framework architecture, while Chapters 6-9 describe the functionality of major components in the framework. Specifically, Sections 9.2 and 9.3 include in-depth looks at two novel advanced visualization tools and how they aid interpretation of data. We wrap up with a case study on the incorporation of the framework into the Risk Psychology Network, a discussion of future work, and conclusion.

2 Motivation

The motivation for this project arises from the needs of MIT's Laboratory for Financial Engineering (LFE). This chapter discusses the data collection and interpretation requirements that underlie the various activities of the research group, illustrating the context within which the idea for this thesis emerged.

2.1 Data and Financial Engineering

Financial engineering is a discipline founded upon mathematical, statistical, computational, and visual methodologies. Quantitative analysis in financial engineering has historically relied upon spreadsheets and tables. Quantitative analysts employ algorithms to derive probabilistic and statistical models of data, utilizing graphical methods to supplement and present numerical methods of analysis. In comparison, technical analysis is a more controversial approach with foundations that lie in behavioral finance and psychology. A technical analyst, often a trader or investor, believes that market price reflects all known information about an individual security and that these prices move in trends. Geometry and pattern recognition are used to extract potentially useful information about the future, without any consideration of fundamental financial measures. Despite differences in focus, both graphical and numerical approaches benefit from the ability to collect and interpret data quickly and efficiently.

A wide variety of data is utilized for ongoing projects within LFE. While some of the data can be obtained wholesale from existing sources (e.g. historical financial data), there is often a need to gather custom data from subjects. Collection of custom data occurs through both automated and user-driven contexts. Automated data collection is characterized by the use of technology to gather information, often yielding large quantities of data in a short period of time. User-driven data collection relies upon human subjects to provide active input. Questionnaires are a familiar example of this type of data collection.

Data collected from human subjects can further be divided into traits and states. *Traits* are characteristics that are more or less stable over time for a given individual. For example, personality characteristics such as risk propensity change very little over a certain time period, especially once the characteristics are fully developed. In contrast to traits, *states* may vary from a

day to day or even second by second basis. Examples include performance, emotions, and physiological data.

Once data is collected into a repository, the science of extracting meaningful information from it begins. Matlab is the primary data analysis tool used at LFE. Collected data must first be imported into the format supported by Matlab. As such, data is often stored in multiple formats: raw data collected initially, transformed data that illustrates the relevant variables in the study, and the imported data used for analysis. The multiple stages involved in manipulating the data before relationships can be analyzed are typical of most research experiments.

2.2 Current Research Efforts

Several projects in the lab illustrate how various types of data are used to answer research questions. One project seeks to investigate the link between live trading performance and emotional states [12]. Sensors attached to financial securities traders record their physiological signals in real time as they trade, providing an indication of their emotional state. Market data is recorded from data feed services and then transformed to derive the financial effects of each trade. Another project aims to gain an understanding of how emotions from the news affect the stock market [24]. Word frequencies are analyzed from news sources and then categorized according to their semantics. These categories are compared to stock market data for the corresponding time period.

Another set of experiments attempts to determine a relationship between personality, risk propensity, and financial performance. Participants take the IPIP NEO Personality test to determine aspects of their personality, such as levels of neuroticism, extraversion, agreeableness, conscientiousness, and openness. Their degree of risk propensity can be gauged by the Sensation Seeking Scale test, or evaluated through other means such as the types of investments they undertake or their activity during an interactive trading simulation. In addition, demographics information such as age, gender, education, experience level, and investment size, and profitability are collected so that possible correlations may be examined.

2.3 Risk Psychology Network

The vehicle for collecting states and traits that require active input from human participants is Risk Psychology Network (RPN), a database-based Web site [13]. Through this Web presence, LFE intends to increase awareness of the influence of psychology in decision making while at the same time facilitate the lab's experiments in this area. LFE made the decision early on to perform data collection through the Web for its accessibility and flexibility (see Section 3.1). As of May 2004, RPN offers participants the option to take three tests, all of which collect trait information: the IPC Locus of Control Test, the IPIP NEO Personality Test, and the Sensation Seeking Scale Test. Additional trait-related questionnaires are in development.

When visitors take a test on RPN, they benefit from an immediate tabulation of their scores. Scores are categorized according to the number of scales and subscales for the particular test. In many of the tests, the results also include narratives explaining the significance of the scores. From these narratives, users can determine whether each score on a certain characteristic puts them at, above, or below average for the general population.

While the site is fully accessible to anyone, users are often introduced to the questionnaires by participating in an interview-based experiment that required the relevant information. In order to target RPN as a premiere Web site for anyone with an interest in "Bridging Finance and Psychology in the Risk Domain", the site seeks to provide additional incentives for visitors to return. This desire to increase "user stickiness" is a common concern on the Internet. E-commerce retailers' ability to improve this factor directly impacts their sales revenue. For a research site such as RPN, the tangible benefit is in increased data for future questionnaires and studies.

The quality of the data collected through the self-administered tests is another concern. There is no guarantee that the user is providing truthful information, especially since the tests are taken anonymously. Out of concern for user privacy, no identification information is requested at any point, and demographic information such as age and gender are optional. While most visitors who have arrived at the site by choice probably will not subvert the tests on purpose, it is possible that some visitors will be wary of revealing their true selves. RPN seeks to create incentives to encourage users to provide accurate demographic data and truthful answers to the questionnaires.

3 Background

3.1 Web-Based Data Collection

Traditionally, the data collection process has been one of the most time-consuming aspects of any research study. In many cases, the nature of the information sought necessitates interview-based approaches in which a research staff member gathers responses directly from the participant. For example, a study that requires personal or sensitive data may encounter resistance if an impersonal, self-administered approach to data collection is used.

Prior to the advent of the Internet, mail and telephone questionnaires were the predominant modes of gathering social science data. Substantial research exists on methodologies that increase response rates. Dillman's seminal work from 1978 [4] articulates how a multi-modal approach with up to five contacts with the recipient, including pre-notifications and reminders, substantially improved response rates. Even by following these recommendations, response rates above 80% are quite difficult to attain. A response rate of around 60% is more typically the norm [5].

The first Internet-based studies were conducted by electronic mail, with the results to the first e-mail survey published in 1986 [10]. These first paperless studies reduced the costs of data collection by an estimated 5-20% [23]. Of course, economies of scale decrease costs even more as the sample sizes increase. According to one study of published email surveys from 1986 to 2000, response rates are highly dependent on the salience or relevance of the questions posed to the sample audience, ranging from approximately 20% to 72% [16].

E-mail surveys eliminated the need to transcribe responses and promoted faster responses. However, the limited interface capabilities of plaintext messages renders data collection through email nearly obsolete today. The reason is that the World Wide Web provides a richer interface for response input. Presentation can be tuned to promote understanding of the questionnaire. On-the-fly error checking and corrections increase the accuracy of the collected information. Integrated storage of responses results in more analysis power and faster interpretation of the results. In addition, the Web precludes any need to collect identifying characteristics (email addresses, phone numbers, mailing addresses), allowing for anonymous responses that often yield greater information.

Of course, Web-based data collection also presents challenges that make mass adoption of the technology difficult. One of the primary obstacles is the significant sample bias in Web responses. Roughly half of households in the United States are unable to access the Internet, and even those people with Internet access may choose to not respond to Web-based surveys. Those who prefer responding electronically tend to have a higher economic status and education level, among other characteristics. Because of this sample bias, major poll organizations such as Gallup still rely on random telephone surveys to gather scientifically-accurate results [14].[1]

In a context where Internet access within the study population is the norm (e.g. university or workplace), issues of sample bias become much less relevant and the marginal cost of a Web-based data collection methodology clearly rivals that of alternatives. The main obstacle then becomes the relatively large technical barrier to creating a robust Web-based survey. Researchers with domain expertise in their area of investigation generally are not well-versed in the relevant technologies, and technologists knowledgeable in designing Web-based systems are not familiar with research methodologies and best practices.

Relatively little research been performed on Web-based survey-collection methodologies and interfaces. However, the research that does exist provides support for the intuition that the visual layout and design of Web input elements significantly influence questionnaire responses. Questionnaire designers must therefore pay attention to non-verbal cues in addition to question wording and other verbal cues [3]. Interestingly enough, a fancy survey design using the latest gee-whiz interfaces was found to reduce response rates compared to a plain design [6]. This result is probably due in part to longer download times and user confusion toward the fancier interface.

Web-based data collection is best viewed as another instrument in the toolbox of techniques for gathering user input. Investigators can choose to integrate Web-based data collection with other traditional methods to optimize quality of response. For example, the seemingly impersonal task of self-administered input to a personal computer need not alienate respondents if combined with a face-to-face meeting. A two-year epidemiologic study that required gathering highly personal diaries on a biweekly basis yielded a coverage rate of 96% by combining a Web-based interface with a one-hour introductory meeting, financial incentives, and appropriate reminders [1].

---

[1] Concerns about sample bias even in telephone-based surveys were not alleviated until 1986, when Gallup decided that the telephone had reached enough household to merit the phasing out of in-person interviews.

3.2 Data Interpretation by Numerical Methods

The aim of the analysis stage is to derive knowledge and observations from available data. When only a sample of the total population can be measured, researchers must rely on mathematical constructs to make inferences about the entire population. An assumption of normality (Gaussian distribution) is most commonly made before applying any inference analysis, and can be checked with a histogram or through quantitative tests.

Correlation is an inference that measures the relationship between two variables in a study, and can in fact be considered the fundamental goal of every research or scientific analysis [19]. Statistical methods help evaluate this relationship in various levels of detail and focus. Metrics include the Pearson's coefficient, a least squares distance measure that assumes a linear relationship between the variables and a normal distribution for each variable.

Timing patterns can be observed by calculating correlations after lagging or shifting one set of time-series data by some number of steps. In autocorrelations, the same time series is compared with a lagged version of itself. A sharply declining correlation coefficient for each additional lag indicates a trend in the time series data. An increase in the correlation coefficient at regular intervals implies a cyclic pattern in the data. When comparing two different sets of time series data, a stronger correlation when one set of time series is lagged hints at a possible causality relationship between the two variables.

Significance tests allow us to measure the likelihood that the results found in the data sample can in fact apply more broadly. The concept of statistical significance can be measured by p-values, defined as the probability of error that is involved in accepting the observed result as valid. That is, given that there is no correlation between the variables studied, p-value indicates the chances that any sampled data would yield a relationship equal to or stronger than the collected dataset. P-values less than 0.01 are commonly considered statistically significant.

Appendix A shows common statistical analysis methods, along with how the methods can be applied on data of various distributions.

3.3 Data Interpretation by Graphical Methods

Graphical representation of the dataset often allows patterns to be identified more efficiently than numerical methods. It is also quite commonly used as a first-pass, exploratory procedure to guide the investigator in choosing additional analysis methods. In some cases, graphical representations reveal trends that cannot be easily quantified. For instance, histograms present variable distributions in a much more discernible, memorable manner than their frequency table analog.

A common technique in handling data is to divide them into categories and then plot each subset. This technique reveals differences between categories that may be further investigated by statistical methods. For example, means, distributions, correlations, etc. can be plotted and compared across different groups in the dataset.

Appendix B shows a summary of possible ways to graphically represent collected data.

4 Design Principles

This chapter identifies and describes the design principles that were followed in developing the framework architecture. These principles help to improve the flexibility and versatility of the system.

4.1 Multiple Tiers

A multi-tier architecture divides computer hardware and software resources into multiple logical layers, where the number of layers varies according to the requirements of the system. Because each layer can be dealt with independently of other layers, the result is increased flexibility in developing, managing, and deploying a system. Developers only have to modify or add specific layers rather than disturb the entire system if technologies change, use scales up, or problems occur.

Abstraction, modularity, and separation of control are key concepts in computer science. As such, considering a system in terms of tiers is really nothing new. However, the term multi-tier architecture implies a greater degree of separation than typically happens within an application. It is not uncommon for two tiers to run on different platform environments, different machines, and across geographical boundaries. Furthermore, each tier that makes up a distributed application can be a cluster of physical machines, with each machine doing a share of the tier's processing work. Replication within the cluster can provide fault tolerance due to failure of individual machines. A switch or load balancer handles sending requests to the right machines in the tier.

A central rule in a multi-tier system is that only resources on adjacent tiers can communicate with each other. Such a restriction simplifies reasoning of the application by minimizing the dependencies needed to achieve functionality. One always has an understanding of how the state of a tier can be changed, because only a maximum of two other tiers can change it. Again, this concept of reducing dependency by decoupling modules is nothing new. It is widely regarded and taught as a best practice in software engineering. However, while it is often tempting to closely couple modules in a local application for code efficiency, doing so in a physically distributed application is much more expensive. Each dependency requires an additional channel of communication, not to mention the complexities and overhead of maintaining these connections in a distributed environment.

4.2 Tiers in a Web System

Applications on the World Wide Web must be able to handle surges in requests and failures of servers with minimal interruptions in service. One important constraint for a Web-based application is that the client, the Web browser, is specified to be essentially stateless. The only exception is "cookies" that the browser user can clear at any time. The HTTP protocol that the browser uses to communicate with the Web server is also stateless, making it impossible for any logic to be performed on the client side. Without layering, it is clear that a monolithic server application suffers from a severe lack in scalability, flexibility, and maintainability.

Name	Description
Client Layer	Displays the content sent from the server, storing and retrieving any cookie state in the process
Presentation Layer	Processes and delivers display content to browser (e.g. HTML), along with any cookie state
Logic Layer	Specifies the business objects and rules of the application, and handles interfacing between the presented information and the stored data.
Database Layer	Stores the volatile state of the application, and exposes ways to access this state.

Table 1: Standard tiers in a Web system

4.3 Model View Controller Design Pattern

The Model View Controller (MVC) design pattern and related variations have become widely implemented as a way to separate monolithic server-side applications into roughly the tiers outlined in the previous section. MVC concretely describes how a server application should handle an HTTP request from a browser client. The original design pattern emerged in the 1970's in the context of user interfaces and was first implemented in the Smalltalk-80 programming language.

To understand the need for MVC in the context of the Web, it is useful to discuss the evolution of Web applications. First, consider the manner in which static Web pages are requested and presented to the browser client. These HTML pages are directly retrieved from disk or memory, and require no additional parsing before being sent to the user.

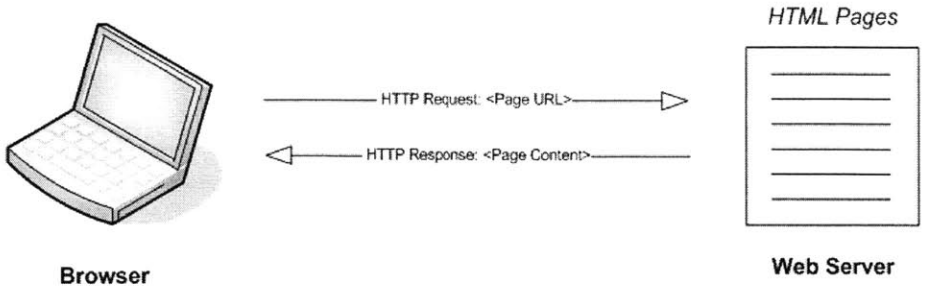


Figure 1: Retrieving static content from a Web server

The desire to gear information dynamically to each individual visitor necessitated server applications that accepted user state. The development of the Common Gateway Interface (CGI) in 1993 heralded a milestone in the history of useful Web applications by providing the standard for data to be passed from the Web browser to a server application. The response output could then be tailored according to the request input, providing users with an interactive experience. Server applications at the time were implemented using C, C++ or Perl. Since the idea for multiple tiers had not yet been fully developed, data access, logic, and presentation mechanisms were generally grouped into a single monolithic program.

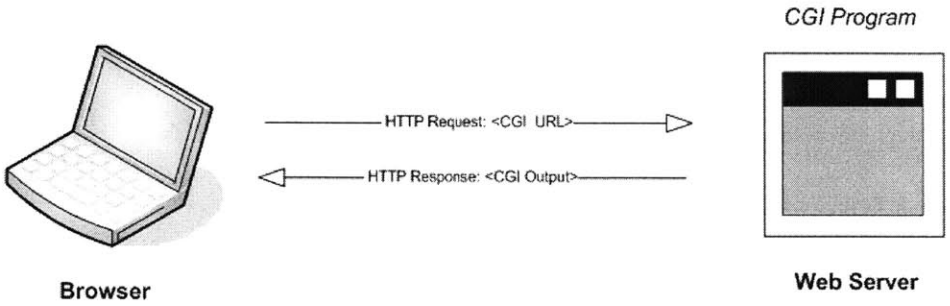


Figure 2: Retrieving dynamic content from a Web server

Several years later, architects at Microsoft and Sun Microsystems alleviated the difficulty of outputting HTML tags from server-side code by allowing developers to more elegantly embed logic and data access code into an otherwise typical HTML page. The advent of Active Server Pages (ASP) and Java Server Pages (JSP) simplified information presentation, but by themselves made little headway in easing the tide of monolithic applications.

In 1998, Sun Microsystems described the "Model 1" design pattern as part of version 0.92 of their JSP specification. This design pattern attempts to articulate a paradigm for developing flexible Web applications. In Model 1, each JSP page acts as both a server application to process inputs, and as the mechanism for presenting the next view to display. A "Model" component is clearly delineated in the form of Java beans. Instead of directly accessing a stored data source, JSP pages use beans to encapsulate data from a stored source.

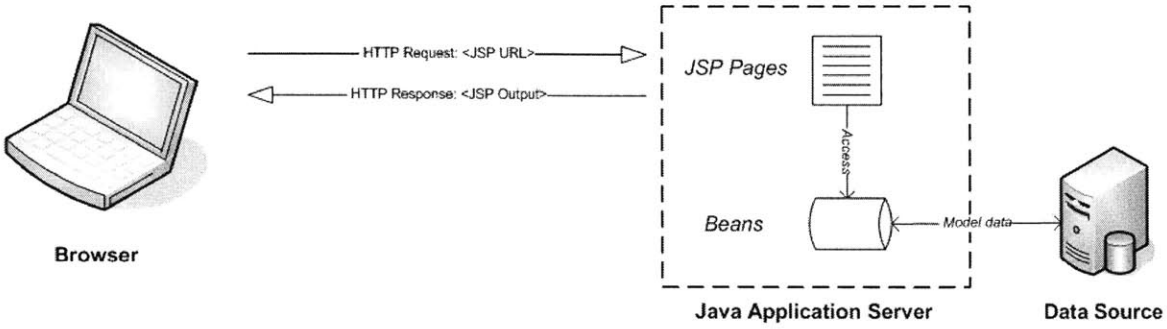


Figure 3: "Model 1" architecture

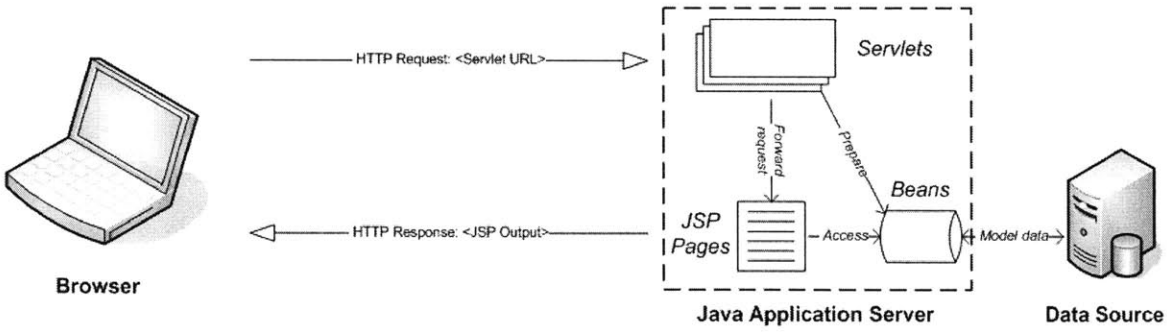


Figure 4: "Model 2" architecture

"Model 1" is sufficient for many lightweight applications, and in fact many current Web applications still follow this scheme. However, as page flows and display content increase in complexity, another level of indirection is necessary to separate control logic from the presentation. Sun's so-called "Model 2" revision specifies Java servlets as the "Controller" mechanism, while relying upon the strength of JSP is as the "View" or presentation mechanism. The transition to MVC means that Java code is reduced in the JSP pages, greatly facilitating a development environment where designers and programmers work in parallel. MVC can be implemented from scratch, or facilitated with programming frameworks such as Struts that is layered above a standard Java application server.

Model	Represents the data objects. The Model is what is being manipulated and presented to the user.
View	Serves as the screen representation of the Model. It is the object that presents the current state of the data objects.
Controller	Defines the way the user interface reacts to the user's input. The Controller component is the object that manipulates the Model, or data object.

Table 2: Model View Controller

5 Framework Overview

This chapter discusses the various layers in the data collection, analysis, and visualization framework. A block diagram of the major components in each layer is shown below. The implementation of each layer uses open-source tools in a Java application server environment.[2]

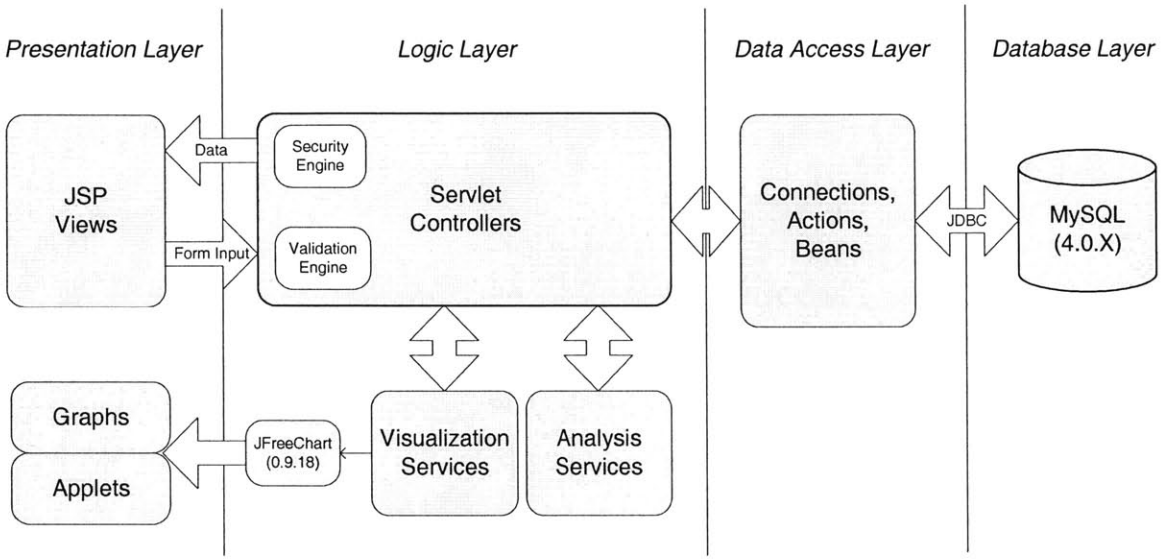


Figure 5: Block diagram of major framework components

5.1 Data Model

When designing any software system, the first step is to specify the states that are maintained and accessed. Figure 6 shows a high-level view of the data model for our framework. This data model is physically stored in a relational database management system. The framework uses the MySQL database since it provides enterprise features at a minimal cost of ownership (free). A data access layer reads and updates the data model, making use of Java beans to encapsulate object contents. We take advantage of connection pooling to allow multiple clients the ability to share connections, minimizing the performance costs of database accesses.

[2] The prototype environment utilizes the Resin application server without additional software frameworks such as Struts.

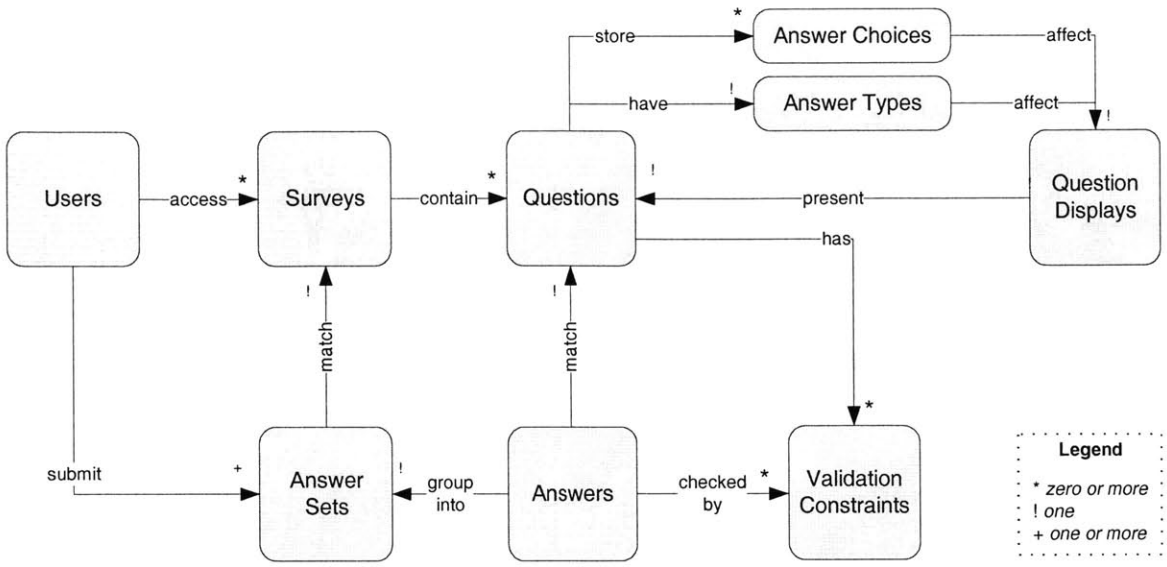


Figure 6: Data model of framework states

Users are the ultimate clients to the system and include those that are setting up a data collection instance and those that are providing data for an existing instance. For the sake of clarity, data collection instances are named "surveys" in the system, but may just as well be self-administered questionnaires accessible only by the user who created it. The system stores a state for each user that includes at least user authentication information so that access control can take place.

The states related to surveys are provided by the survey creator, and may be modified to some extent even after data collection has begun. Each question in the survey stores the data type of the answer submission. This data type cannot be modified after survey creation. Answer choices must be provided for those questions whose responses are constrained to a set of possible outcomes. All states pertaining to a question are gathered and processed to create a question display. Question displays improve the overall efficiency of survey presentation by caching the HTML tags related to a question, and can be regenerated as necessary from question information and the system's available input widgets (see Section 7.1). Each question also maintains zero or more validation constraints that are used by the validation engine to check that the user input conforms to a specified format (see Section 7.2).

In the data collection phase, each successful submission to a survey generates an answer set record. Answer sets include metadata such as the user who submitted the responses and the time

of submission. Individual answers are maintained as separate objects with references to their corresponding answer sets (see Section 7.3).

5.2 Logic Layer

In accordance with the Model View Controller design pattern, every input request gets received by a servlet, which may decide to forward or redirect the request to other aspects of the logic layer to provide the appropriate response. Together, the servlets and the various services achieve system functionality and direct page flow.

Access control and validation are two vital services every servlet takes advantage of. Access control ensures that only those with the appropriate security authorizations can access and affect the data model. Restrictions occur at the level of surveys. The module checks that only users who are given explicit permission to view and submit information to questions in the survey or modify the survey are allowed to do so. Security relies on a user identifier, which exists in a Web-browsing session only after the user has successfully authenticated into the system. The user identifier is compared with the survey identifier to determine if the servlet should continue processing the request or deny it. Validation ensures that user input into the system do not corrupt the data model. Details about validation can be found in Section 7.2.

Many of the servlets perform modifications to the data model. These include functionality such as adding a survey, adding or editing a question, answering a survey, and more. Implementations of these features are conceptually straightforward. The code takes user input, checks the input against the access control and validation engines, and then calls upon appropriate interfaces in the data access layer to update the database.

The analysis and visualization services do not modify the data model, but rather process the information to display useful results. To minimize network load, we perform processing directly within the database layer whenever possible. For instance, we utilize existing database functions to calculate descriptive statistics such as the mean and standard deviation in order to avoid having to transfer all stored data across layers.

Visualization services rely upon the JFreeChart library for drawing charts and graphs. The system uses JFreeChart in two ways: first as a server-side module to generate graph images, and second

as a client-side library to support applets. In the first case, the visualization services access the data and process it to a format understood by the JFreeChart modules. JFreeChart outputs a graphical image, which is sent directly to the client. In the applet use case, the necessary library files are transported across the network to the client browser, and any associated data is sent to the presentation page that hosts the applet. In no circumstances does the applet ever make a connection to the database layer, thus preventing database connection parameters from being exposed.

5.3 Presentation Layer

With the exception of graphs, servlet controllers respond to the client with HTML text. The presentation layer consists of the JSP pages that the servlets utilize to format the output. The benefit of using JSP is to avoid directly coding HTML within the servlet. Figure 7 illustrates the main pages in the system, along with the page flow dictated by the servlet controllers.
Figure 8 shows the frame-based administrative interface for creating questionnaires. The left panel displays the questionnaire in the process of being constructed, while the right panel gathers information for the data model.

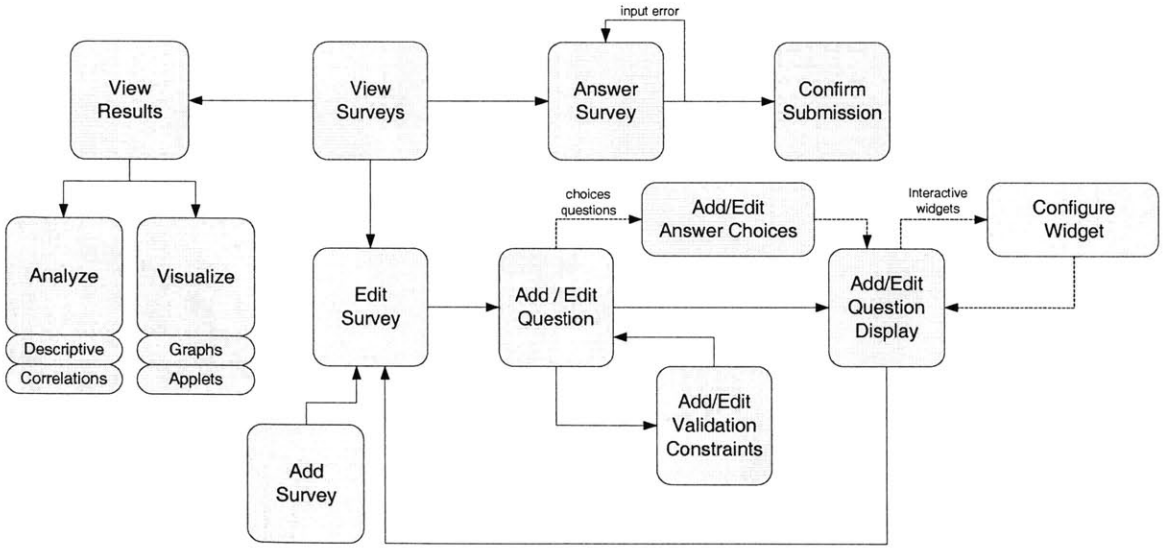


Figure 7: Notable presentation pages and corresponding page flows

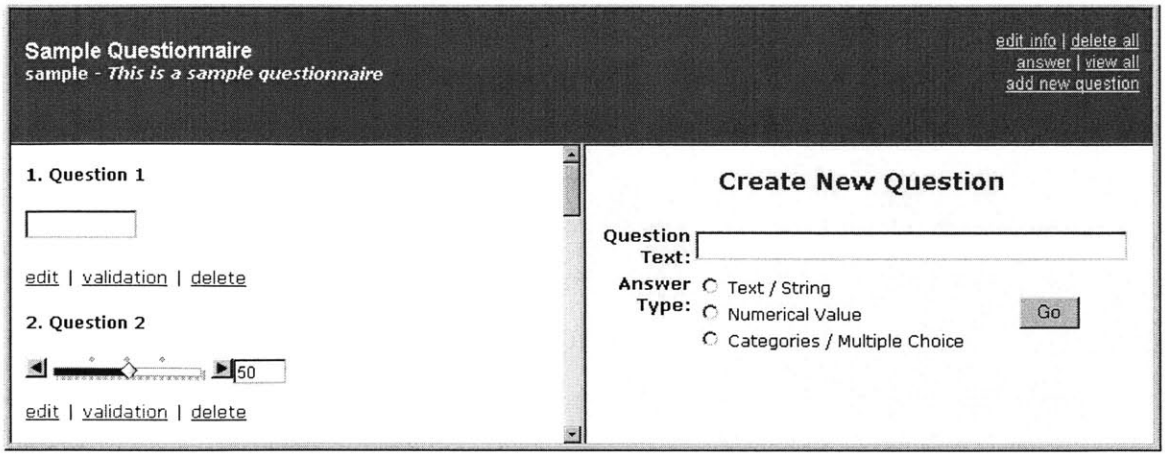


Figure 8: Prototype administrative interface for constructing and editing questionnaires

6 Data Specification

Before data can be collected, it is necessary to first understand the variable types that the system should expect as input. This information is necessary in order to perform validation during input time, and to be able to quickly decipher how different variables can be interpreted. In addition, specification of data types directly impacts the Web interface options for collecting the information (see Chapter 7).

6.1 Classic Researcher's Approach

A classic categorization developed by Harvard psychologist S.S. Stevens in the 1940's divides variables in terms of four measurement scales: nominal, ordinal, interval and ratio (see Table 3).

Data Type	Description	Examples
Nominal	Unordered, qualitative classification with arbitrary values	gender, race, color, city
Ordinal	Ordered measurement that cannot be compared with each other to indicate "how much more"	socioeconomic status, Likert scales (e.g. 1..5)
Interval	Ordered measurement that can be ranked and compared, but without a natural zero	temperature, dates
Ratio	Similar to interval but with a natural zero that allows comparison by division	height, weight, age, length

Table 3: Classic researcher's approach to specifying data

Since their development, these scales have been widely cited and used for selecting and recommending statistical analysis methods. However, a substantial amount of criticism exists regarding the strictness of the classifications. Statisticians have remarked that the scales unnecessarily limit the types of analysis that can be performed and add to the confusion of selecting a technique. Opponents of the scales argue that interpretation of data depends upon the question asked and should not be restricted by classifications that are often meaningless in the context of the analysis. A flippant example aims to answer whether the freshman members of a football team had been given low jersey numbers. Football numbers are clearly nominal values. Strictly speaking, computations such as adding and averaging to help measure whether the numbers were in fact too low could not be performed [22].

6.2 Technologist's Approach

Another approach to specifying data is by the raw format in which the information is stored. Since databases are intended to be an efficient means of storing data, we can look to their column types as potential data type specification options (see Table 4).

Column Type	Storage Required
TINYINT	1 byte
SMALLINT	2 bytes
MEDIUMINT	3 bytes
INT, INTEGER	4 bytes
BIGINT	8 bytes
FLOAT(p)	4 bytes if 0 <= p <= 24, 8 bytes if 25 <= p <= 53
FLOAT	4 bytes
DOUBLE [PRECISION], item REAL	8 bytes
DECIMAL(M,D), NUMERIC(M,D)	M+2 bytes if D > 0 M+1 bytes if D = 0 (D+2, if M < D)
DATE	3 bytes
DATETIME	8 bytes
TIMESTAMP	4 bytes
TIME	3 bytes
YEAR	1 byte
CHAR(M)	M bytes, 0 <= M <= 255
VARCHAR(M)	L+1 bytes, where L <= M and 0 <= M <= 255
TINYBLOB, TINYTEXT	L+1 bytes, where L < 2^8
BLOB, TEXT	L+2 bytes, where L < 2^16
MEDIUMBLOB, MEDIUMTEXT	L+3 bytes, where L < 2^24
LONGBLOB, LONGTEXT	L+4 bytes, where L < 2^32
ENUM('value1','value2',...)	1 or 2 bytes, depending on the number of enumeration values (65,535 values maximum)
SET('value1','value2',...)	1, 2, 3, 4, or 8 bytes, depending on the number of set members (64 members maximum)

Table 4: MySQL approach to specifying data

As the table above illustrates, databases often offer variants of the same data type with different storage requirements. While exposing these raw data types to users maximizes storage efficiency, there is a heavy price to pay in terms of understanding and flexibility. Even those knowledgeable

in database design are often at a loss as to which variant to choose and have to alter the data type at some later time.

6.3 Framework Approach

Both the classic research scales and the raw data formats are rather obscure to the general public, and it is not at all evident that the categorizations are sufficiently useful. Our approach to specifying data attempts to balance simplicity, understanding, and flexibility. The categories are shown in Table 5.

Data Type	Description
Number	Quantitative input
Text	Qualitative free-form input
Choices	Limited qualitative input

Table 5: Our approach to specifying data

These three simple types allow for specification of data at a much more intuitive level than the two alternative approaches previously described. Variants of data types are ignored, sacrificing storage efficiency for clarity and comprehension. Ordinal, interval, and ratio scales are condensed into a single "number" type. However, note that these three data types are simply those exposed by the Web interface, and no constraints on types are actually imposed in the data model. In other words, extensions of additional data types are supported by the framework.

Even with these three types, the user has some room for adjustment. The types "Number" and "Choices" can be specified as "Text" without limiting data collection. However, doing so would unnecessarily limit the dynamic analysis and visualization features of the system for that variable. An advantage of the "Choices" type over the "Text" type is more efficient storage of responses, since only a reference to the choice and not the text itself is stored. Further details about input storage can be found in Section 7.3.

The Web interface for providing the data specification can be seen in Figure 8.

7 Data Input

Once the framework has the necessary metadata specifying the variable types, it is ready to accept data input. While it is possible to supply input from an automated source via direct access to the database layer, the expected means by which input is to be collected in the framework is through the Web.

7.1 Input Widgets

The Web offers rich possibilities for collecting data interactively. However, the need to support a wide range of browsers and machines also limit the type of input interfaces that can be presented to the user. The first HTML specifications provide a set of form inputs that continue to be widely used today. Scripting languages and other enhancements allows for more interactive input methods, but may not be supported by all browsers.

We allow questionnaire creators to select the input interface for each question. The possible input widgets vary according to the answer type to the question (see Figure 9 to Figure 11), and include the basic HTML form interfaces that should be understood by even novice Web surfers as well as more complex interfaces. Since each input widget is stored as a JSP page in the framework, interface design experts can modify the look and feel of existing widgets and provide new widgets without dealing with Java code.

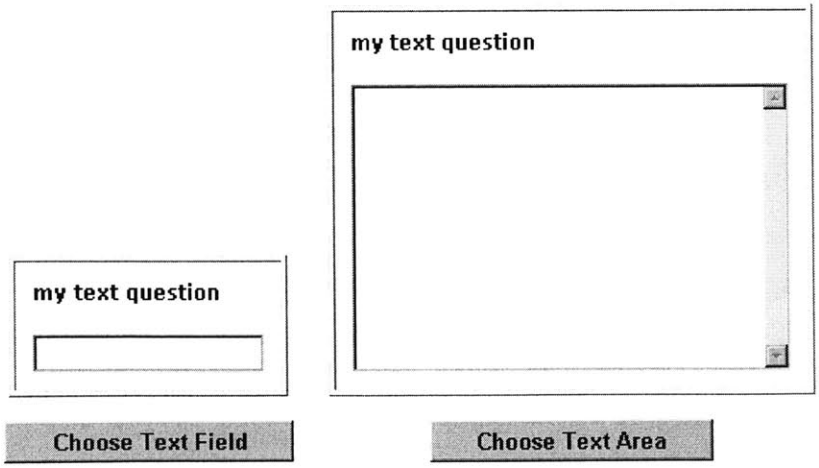


Figure 9: Sample input widgets for text data

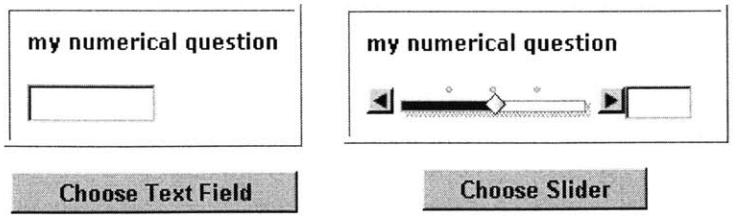


Figure 10: Sample input widgets for numerical data

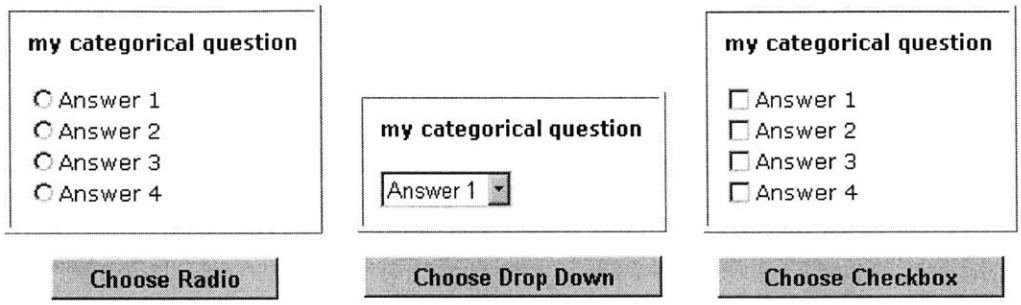


Figure 11: Sample input widgets for categorical data

While most of the provided input widgets are static HTML form elements that can be replicated on paper, the slider widget for numerical input provides interactivity that only the Web can offer. Implementation is through JavaScript, and the JavaScript code is integrated into the data collection interface so that the input is treated as if it were provided by any static HTML form element. The slider widget offers the capability of being configurable. The configuration screen appears after the slider is selected, and serves to constrain and discretize the input (see Figure 12).

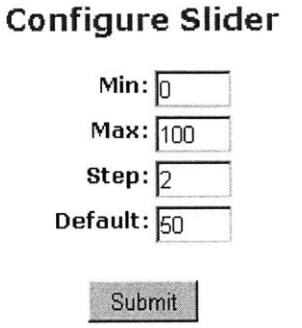


Figure 12: Configuration of the slider widget

7.2 Input Validation

As mentioned in the background section, one of the benefits of collecting user input through the Web is the ability to validate that the input is in the expected format before any information is stored and used. In fact, the mechanism that the framework uses to validate input to questionnaires is the same one used to validate any input into the system during the questionnaire creation process (e.g. to make sure that the configuration information to the slider are filled in with integer values).

Each input may be associated with zero or more validation constraints (see Table 6), which are interpreted by the framework's validation engine on the server side. The engine checks the constraints against the input and returns any associated error messages in case of validation failure. Some of the constraint-checking functionality utilizes the common validation methods provided by the Jakarta Project [9].

Name	Description	Parameters (if any)
required	A non-blank input must be provided	
requiredconditional	A non-blank input must be provided given that another input matches a certain value	Input name to check, Input matching value
multiplechoice	The input must be one of the given matching values	Possible matching values
integer	The input must be an integer	
decimal	The input must be a decimal number	
minrange	The numerical input must be greater than or equal to a given value	Number
maxrange	The numerical input must be less than or equal to a given value	Number
regexp	The input must match the given regular expression	Regular expression
minlength	The input must be at least a given number of characters in length	Length
maxlength	The input must be at most a given number of characters in length	Length
email	The input must match an email address	
date	The input must match a date	
url	The input must match a URL	
creditcard	The input must match a credit card number	

Table 6: Sample validation constraints

The questionnaire-building interface allows users to configure a subset of the validation constraints shown in Figure 13. Each validation constraint is stored in the data model and is takes effect when questionnaires are submitted. By default, each question is marked as required, numerical values are marked as integers, and categorical values are limited to the choices specified by questionnaire creator. Note that the form input used to configure question validation is examined by the very same validation engine, but with constraints specified programmatically within servlet code.

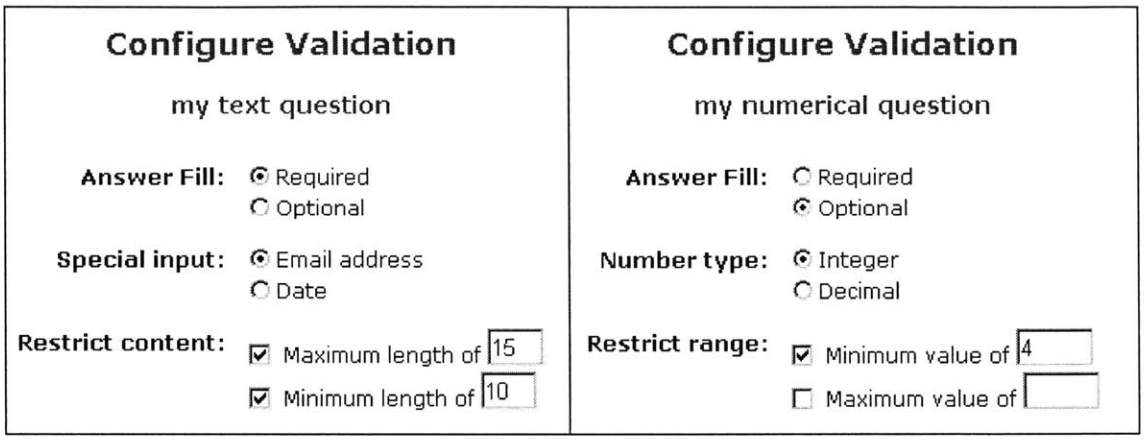


Figure 13: Specifying validation constraints for text and numerical questions

To minimize network connections and server load when input is invalid, validation can also be provided on the client side through JavaScript. The same constraints can apply, except that these constraints are checked before any input is sent to the server. An alert pops up with the appropriate error messages if any of the constraints fail. However, performing client-side validation does not replace the need for a validation engine on the server since a malicious user can easily circumvent the JavaScript error checking code.

7.3 Input Storage

By design, the data types understood by the framework do not correspond to those types provided by the database. While Chapter 6 showed that there are significant user benefits to creating our own data types, eventually the input will have to be stored in some format into the database. We choose to store the response in one of two ways: a text field and a numerical field. These correspond to the MySQL column types TEXT and DOUBLE, respectively.

Our scheme for input storage maximizes efficiency given the limited information about data format provided by the specification. Once validation verifies the integrity of the response type, we can successfully store data into their corresponding database fields. Categorical responses are stored numerically according to their integer-type references; the actual answer choice text has already been stored elsewhere during the questionnaire-creation phase. The size of text input is limited by the 64 kilobyte maximum length of the TEXT field, a reasonable and realistic constraint.[3]

[3] Longer text of up to 4 GB is possible by changing the column type to LONGTEXT, but doing so opens up the system to flooding attacks.

8 Data Analysis

As discussed in Chapter 3, a variety of statistical measures and tests exist to extract patterns in the data. It is not the goal to implement most or even a sizable fraction of the statistical analysis tests that exist, for many commercial systems already fulfill the goal. Rather, we demonstrate how analysis components can be plugged into the framework and utilized on existing datasets.

The analysis tools we choose to implement directly follow from the needs of the Risk Psychology Network (see Chapter 10). They calculate descriptive statistics on the data, such as the mean, minimum, maximum, and standard deviation. A correlations module runs on pairs of variables to quantify the linkage relationship between the variables. An autocorrelations and lagged correlations module delays one set of data to check for any influences or trends over time.

The existing modules allow users unfamiliar with statistical methods to interpret their collected data, as the underlying concepts are either fairly well-known or can be explained intuitively. Section 3.2 provides further descriptions of the numerical analysis methods.

A significant benefit of implementing the analysis modules for the Web is to streamline the interpretation process. Direct connection to the dynamic data source allows tracking of information on a continual basis, even while the information is being collected. Section 10.2 illustrates real-world applications for which these tools can help gain immediate insights.

Figure 14 to Figure 16 show samples of the presented statistical information. While the bulk of the presentation is in a tabular format, visualizations such as bar graphs and scatter plots (see Section 9.1) can enhance understanding of relationships and trends. Users who desire more sophisticated analysis or prediction capabilities can separately export their data and import it into one of the many available commercial programs.

Descriptive Statistics

Question	Number of Samples	Minimum	Maximum	Mean	Standard Deviation
Question 1	42	1.0	55.0	14.548	7.977
Question 2	42	2.0	36.0	14.429	9.027
Question 3	42	2.0	44.0	17.405	10.413
Question 4	42	2.0	212.0	40.333	63.246

Figure 14: Sample descriptive statistics

Correlations

	Question 1	Question 2	Question 3	Question 4
Question 1	1.0	0.74	0.573	0.36
Question 2	0.74	1.0	0.965	0.89
Question 3	0.573	0.965	1.0	0.955
Question 4	0.36	0.89	0.955	1.0

Top Correlations

1	Question 2	Question 3	0.965
2	Question 3	Question 4	0.955
3	Question 2	Question 4	0.89
4	Question 1	Question 2	0.74
5	Question 1	Question 3	0.573
6	Question 1	Question 4	0.36

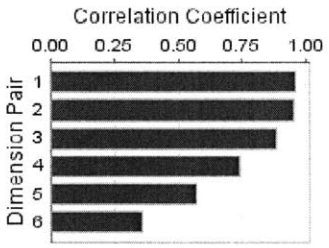


Figure 15: Sample correlation statistics

Autocorrelation:

Question 1

Lag	Correlation Coefficient
0	1.000
1	0.062
2	-0.050
3	-0.151
4	-0.250
5	-0.087

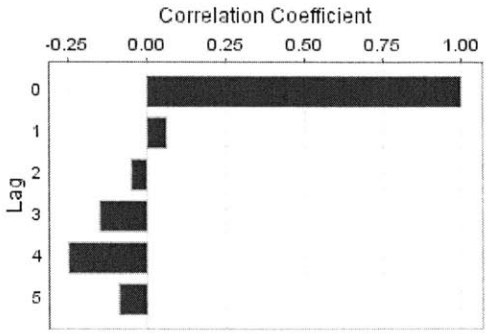


Figure 16: Sample autocorrelation statistics

Lagged Correlations:

"Question 2" vs. "Question 3"

Lag	Correlation Coefficient
-5	-0.151
-4	0.089
-3	0.333
-2	0.503
-1	0.658
0	0.965
1	0.699
2	0.539
3	0.365
4	0.136
5	-0.129

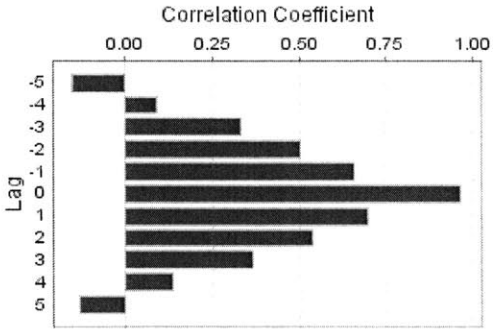


Figure 17: Sample lagged correlation statistics

Unlike with the visualization tools (see Chapter 9), it is not necessary for the analysis modules to extract entire datasets from the database before processing the result. Most of the necessary calculations are performed directly within the database using basic functions that the database query language provides. These functions include average, standard deviation, minimum, and maximum. For the autocorrelations and lagged correlations, we make use of helper table structures and the creation of temporary tables to perform multiple correlations in a single database access. By taking advantage of these mechanisms, we minimize the data transfer and database connection costs.

9 Data Visualization

This chapter describes the visualization options available to users, and how they are implemented. While a number of graphing packages for Java and the Web exist, JFreeChart was chosen for its relative maturity and availability of documentation and source code. Its capabilities are said to surpass even commercial Java-based packages, though the package is certainly not without bugs and instabilities.

9.1 Basic Plots for Dynamic Datasets

Since many experiments collect time series data, three basic plots provide the ability to visualize data with time on the domain axis. A line chart plots data points and connects adjacent ones. A bar chart and stacked area chart provide roughly similar views of the data, but differ from the line chart in that the horizontal space between each data point is the same. For all three, multiple datasets can reside in series on the same plot. For comparison between two variables paired by input time, a scatter plot can be presented.

Implementation of the basic plots relies upon JFreeChart's integration into a Java application server environment. This functionality allows all the processing for drawing to take place entirely on the server. The work flow is as follows. A servlet accesses data from the database layer and manipulates the returned database records into the Java structures that JFreeChart requires as input. JFreeChart processes the data and generates a graph as a static image file. The servlet sends the PNG-formatted file directly to the client browser.

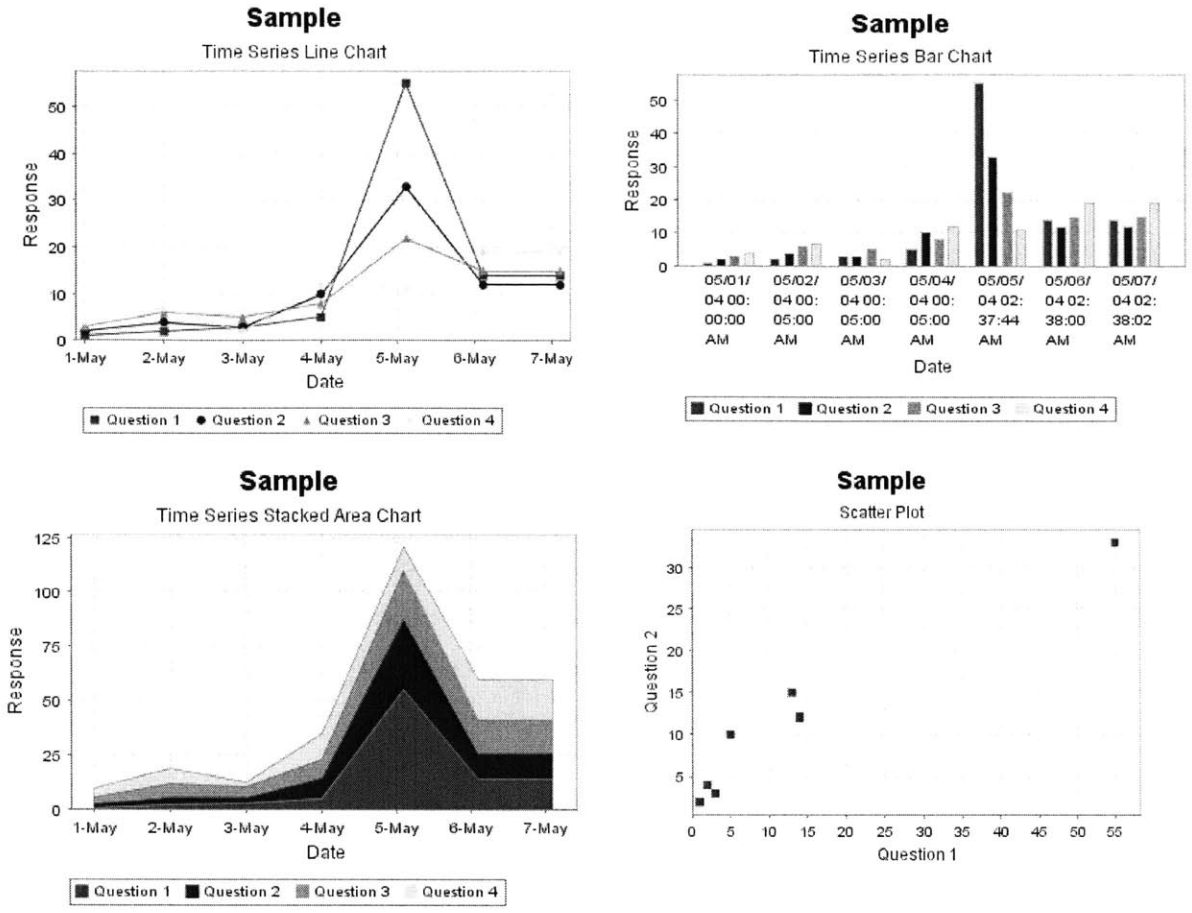


Figure 18: Basic plots for dynamic datasets

9.2 Advanced Stacked Area Percentage Plot

The basic charts suffice for many needs, but are limited in the information that can be displayed on a single chart. For instance, labeling the static plots with the actual values would clutter rather than enhance interpretation. To alleviate such difficulties, we created an advanced interactive plot that allows the user to navigate through an area chart and gain time slice views of responses in the form of a pie chart. As far as we know, this dynamic visualization is not available anywhere else on the Web.

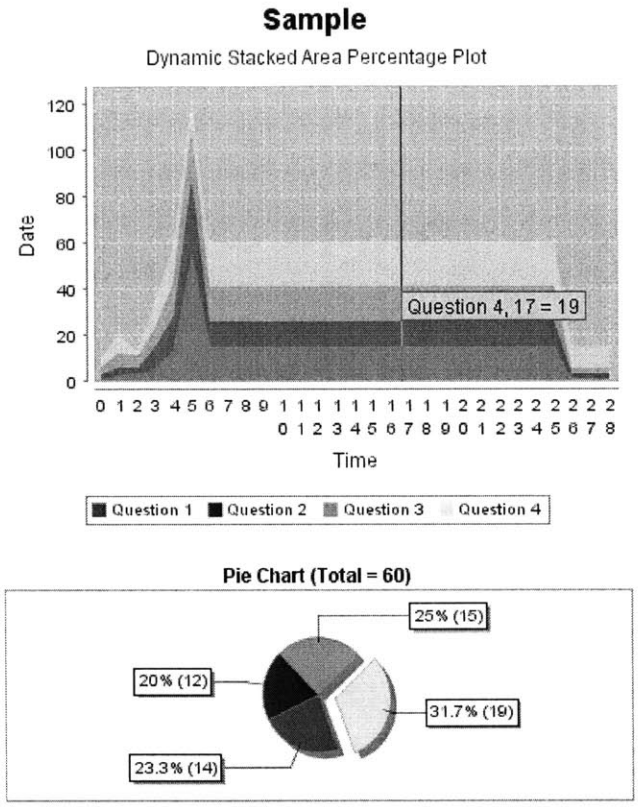


Figure 19: Dynamic stacked area percentage plot

Unlike the basic plots that consist of image files, this visualization is a Java applet that runs on the user's local machine. Charting functionality is provided by modified JFreeChart libraries, and specialized code connects the two charts and adds interactivity. The disadvantage of using an applet is longer download time and greater processing on the client browser. We attempted to reduce the download time of libraries by stripping them of unused JFreeChart functionality[4], but users on low-end browsers and slow connections likely will not be able to smoothly support this visualization.

An important architectural consideration in the design of this advanced applet is the separation of the presentation from the data access. Due to security and performance, it is highly undesirable to have the applet directly access the data layer. We were able to preserve the multi-tier scheme by having a servlet collect the relevant data and format it as input into the applet. A tangible benefit

[4] The total download size for the visualization is about 550kb.

is that the applet can be tested and used independently of the rest of the framework, simply by altering the parameters.

9.3 Advanced 2.5D Dynamic Response Distribution Plot

Another advanced visualization uses the same architectural concepts as the stacked area percentage plot in order to bring interactivity to user experience. The result is an innovative way to interpret frequency distribution information.

9.3.1 Visualizing Response Distribution

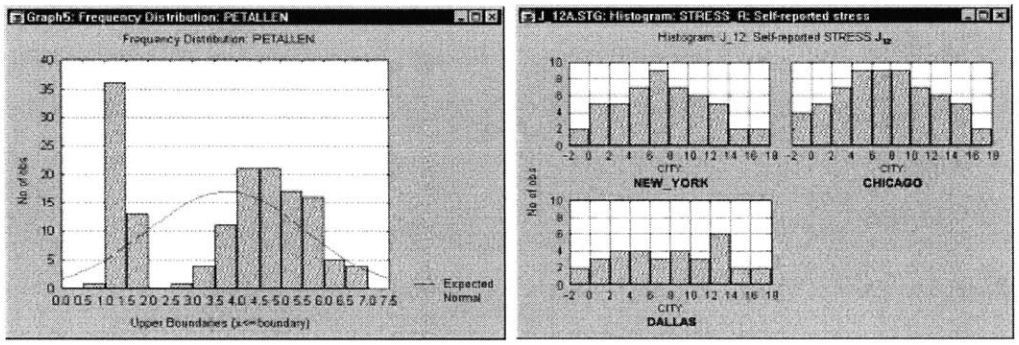


Figure 20: Basic frequency plots [19]

Frequency plots or histograms are a natural way to understand the distribution of any data variable. The horizontal axis contains the full value range of the variable being analyzed, and the vertical axis indicates the number of occurrences for each category in the range. For nominal data, the number of categories is fixed to the possible choices of value. For numerical data, the number of categories is based on an arbitrary division of the full value range. To consider the frequency distribution across multiple dimensions, one must fix the categories for all but one dimension, and plot a histogram for each permutation of the categories. Figure 20 shows histograms for a single dimension and for two dimensions.

As the number of dimensions and categories increases, it becomes more difficult to present distribution information without suffering chart overload. One way to enhance clarity is by plotting data points on a scatter plot, and then showing a separate histogram for each cell on the scatter plot (see Figure 21).

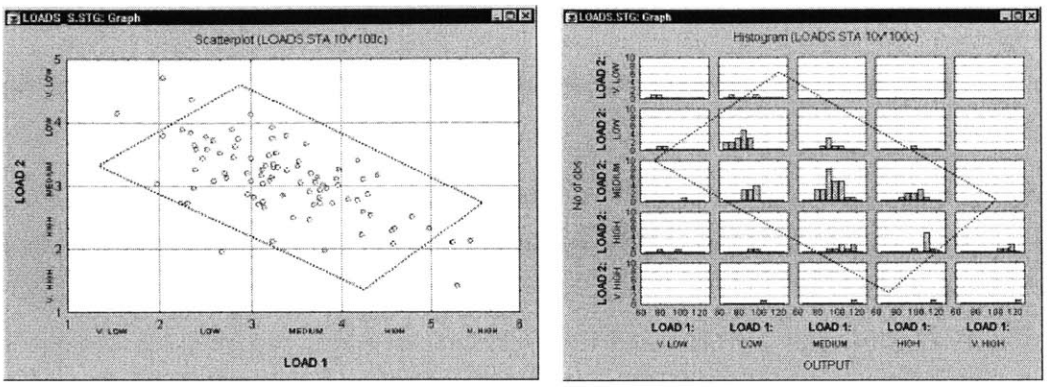


Figure 21: Showing distribution with scatter plot [19]

Our advanced 2.5-dimensional dynamic response distribution plot builds upon the aforementioned concepts in presenting frequency information. By taking advantage of the Web's capabilities for interactivity, we can present information in an even more condensed fashion (see Figure 22). The so-called "2.5 dimensional" visualization supports providing distributions for two data variables by combining a contour projection with histograms. The extra half dimension in the name refers to the use of color to provide at-a-glance frequency information in the contour projection. The visualization is "dynamic" in that mouse movement across the contour projection updates the two connected histograms. Each histogram shows the distribution of one variable as the other variable is fixed to the current location of the mouse. Information about the relevant ranges of the histogram is displayed via a popup tooltip. Since each variable is divided into twenty same-sized categories, a total of 400 unique sets of histograms can be shown.

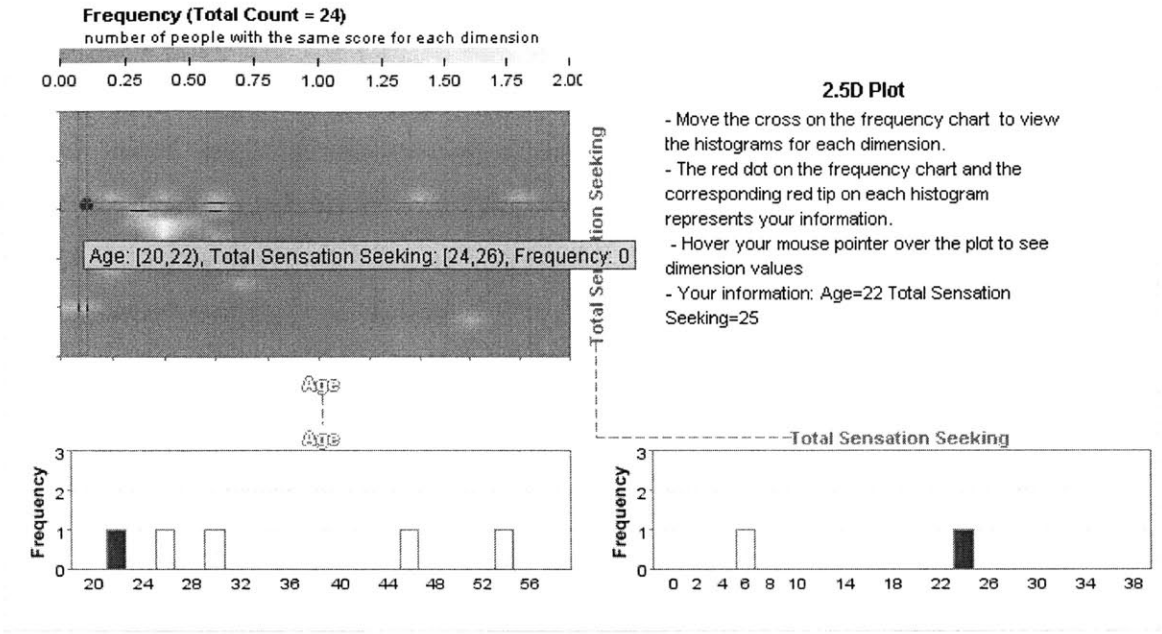


Figure 22: Advanced 2.5D dynamic response distribution plot

The rest of this chapter provides descriptions of the key features of the visualization and how they influence user interpretation of the data.

9.3.2 Color and Information

Color is a fundamental design tool that provides an efficient means of communicating information. The human eye intuitively perceives patterns through color before even being cognizant of underlying details. Color can be used to represent quantity, distinguish or deemphasize features, or simply add visual appeal to the presentation – all without imposing constraints on available space.

The default colors used in the contour projection of the 2.5-dimensional dynamic response plot were chosen in order to clarify the distribution without distraction or confusion. It follows Edward Tufte's recommendation that "a good strategy for choosing colors to represent and illuminate information is to use colors found in nature, especially those on the lighter side." The light blue used at the lower range and the yellow for the upper range hints at the topological differences between sea and peaks. Users can choose to select alternative colors through the provided configuration options (see Figure 23). These options allow for experimentation,

accommodation of colorblind individuals, and adjustment for presentation differences across monitor types.

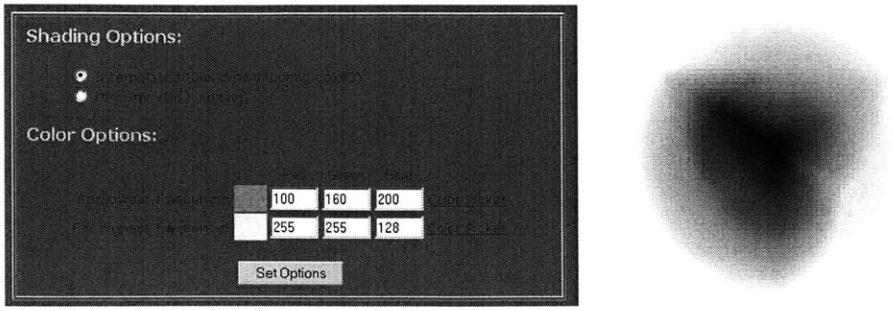


Figure 23: Color configuration options and the color picker

9.3.3 Color Smoothing for Trend Detection

Another configuration option allows users to turn on or off color smoothing for the contour projection. Figure 24 shows the effect of interpolating the discrete colors of a 5x5 matrix. Even in this contrived example, the effect of smoothing on trend recognition can be seen. The algorithm we devised achieves the effect by first considering the center of each cell as discrete points of color. Then, vertical gradient lines are drawn between adjacent pairs of these discrete points. Finally, extrapolating the colors between the pairs of vertical gradient lines smoothly fills in the plot.

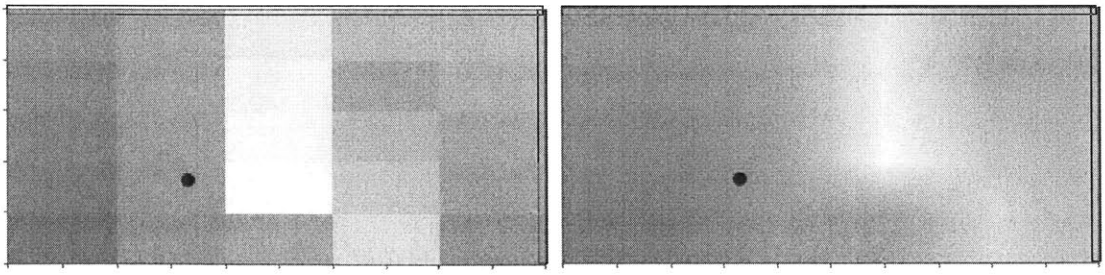


Figure 24: Contour projection before and after applying color smoothing

9.3.4 User Identification

The visualization supports the addition of an annotation to indicate a point of interest. This feature can be used to highlight where a single user's results fall in the distribution. The color used for the highlighting is a bright red, in order to distinguish the identifying marker from the rest of the plot. Corresponding bars on the histograms are also highlighted whenever the mouse hovers over relevant ranges (see Figure 22). The highlighting is done in such a way that the height of each bar still accurately reflects the total frequency for that range.

9.3.5 Deployment

A fair degree of custom coding (approximately 5000 lines) and the use of an older version of the JFreeChart library makes the visualization under 300 kB to load despite being more complex than the stacked area percentage visualization (see Section 9.2). Again, all information necessary for the visualization is taken from parameter inputs. A servlet processes data points and generates frequencies according to the number of categories for each dimension.

It is processing-intensive to recalculate the frequencies at every invocation of the applet. For the most part, the dataset should change at a slower rate relative to the number of invocations of the applet. We can minimize recalculations by caching the frequency information used as input into the visualization. The servlet accesses the cache only, while a separate server-side mechanism processes the raw data into frequency information at some set time interval.

10 Case Study

The motivation for this project arose from a desire to improve the Risk Psychology Network, a research project in MIT's Laboratory for Financial Engineering. So far, we have described the framework in a generic context. In this chapter, we will discuss more specifically how the incorporation of framework components into the Risk Psychology Network can improve user experience, prompt visitors to provide more accurate and truthful information, and offer incentives for them to return to the site.

10.1 Result Comparison through Visualization

The multi-tier architecture of the framework allows visualization components to be used independently of the underlying data model. While the data model on RPN holds many similarities to that of the framework (see Appendix), the models substantially differ in key respects. Specifically, the questionnaires supported by the RPN data model expect all responses to be on a Likert scale. RPN also includes a scoring mechanism for questionnaires, with the resulting scores saved into the data model. In addition, profile information is stored separately for users that provide the data.

We want to take advantage of the 2.5D dynamic response distribution plot (see Section 9.3) to allow visitors to graphically compare their questionnaire scores against others who have taken the tests. We can do so by extracting the dimensions we want to compare from the RPN data model and then computing frequency values across the dimensions. These values get submitted to the applet for visualization. While the data extraction and frequency computations rely on the data model, the visualization itself does not.

Category	Dimensions
Demographics	Account Size, Age, Profitability, Years Trading
Locus of Control Test	Chance, Internality, Powerful Others
Personality Test	Agreeableness, Conscientiousness, Extraversion, Neuroticism, Openness
Sensation Seeking Test	Boredom Susceptibility, Disinhibition, Experience Seeking, Thrill and Adventure Seeking, Total Sensation Seeking

Table 7: Dimensions available for comparison in the Risk Psychology Network

The two natural dimensions of comparison are demographic information and scores. Table 7 shows the available demographic dimensions as well as the psychological dimensions computed from the scores from the site's three tests.

Visitors to the site who want to try out the visualization must first answer at least one questionnaire. Since one of the dimensions of the visualization is demographic information, visitors are encouraged to provide the site with that information if they wish to understand how they compare in relation to their peers. They are encouraged to provide truthful answers since both the scores and the visualization will allow them to gain more knowledge about themselves. As described in Section 9.3.4, the user's data point is indicated with a red annotation on the contour projection and corresponding annotations on the associated histograms.

The visualization becomes increasingly useful as more and more participants fill out their demographics information and accurately respond to the questionnaires. Therefore, users who have previously responded to all questionnaires on the site and viewed their results can still expect to gain new information on future visits.

10.2 Behavioral Logs

Users who have completed all questionnaires on the site may not be tempted to make repeat visits despite the additional insights they can gain from the questionnaire results of others. It is commonly observed that the more information users leave behind about themselves on a Web site, the more they tend to return to that site. One way we can encourage users to provide more information about themselves is through the addition of questionnaires. This approach has several drawbacks. First, adding a new questionnaire requires the intervention of site administrators and some time to perform regression testing of functionality. Since the site does not collect the contact information of visitors, there is no way to alert past visitors to the addition of a questionnaire they might be interested in. Finally, there is no way of knowing how relevant a questionnaire might be to users.

Here is an area where the capabilities of the framework can substantially improve the site functionality and user experience. Using the framework's data collection and input capabilities, we can allow visitors to construct their own questionnaires. Unlike existing RPN questionnaires that collect trait information, the purpose of the user-constructed questionnaires, which we call

"behavioral logs," is to provide visitors with the means to track state information. While RPN may provide examples of states that the user might want to collect in order to encourage initial use of the system, the user is free to track any information they desire. Once collected, the visitor can take advantage of analysis and visualization components to display and interpret their data.

The rest of this section provides some sample applications of the behavioral log feature that are relevant in the finance and risk domain.

10.2.1 Trader Log

Traders are generally a rather superstitious group. They sometimes attribute their trading performance or lack thereof to environmental factors and personal behaviors, and seek to maximize those factors and behaviors that result in better performance. A trader who visits RPN might desire to record the factors that they suspect have an impact on the day's profit and loss numbers. Figure 25 shows the questionnaire that they can construct and the interface to track their results.

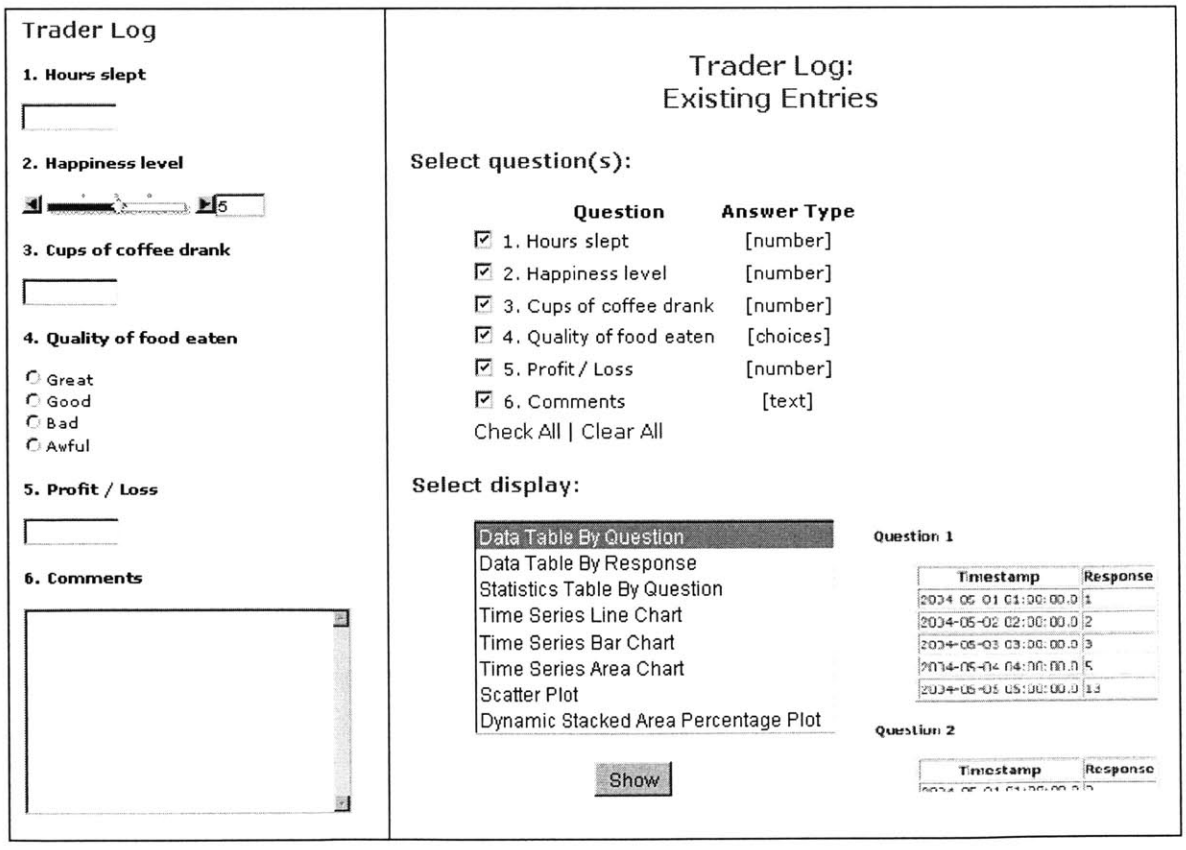


Figure 25: Sample trader log and results page

Questions that traders may seek to answer through the log include how their emotional state and performance are related to their sleep level, caffeine intake, or food quality. They can calculate descriptive statistics such as the mean and standard deviation for each variable to get a sense of their behavior. They can plot time series graphs for each question, visualizing the trend of any variable over time. Correlations quickly provide a list of variables that potentially have the strongest relationship to performance. Lagged correlations hint at how much their behavior and performance in the past can influence future states. Using the information gained from the framework's data interpretation tools, traders can choose to modify their behavior to optimize for certain states. Or more simply, traders can benefit from reflecting upon their journal of thoughts as provided in each day's comments field.

10.2.2 Investment Tracker

Another application for the behavioral log is to track financial allocations through time. The information can be used for the purposes of budgeting or merely historical reference. Visualizations such as the dynamic stacked area percentage plot are especially useful for comparing allocations. Figure 26 shows a sample log for a user's investment portfolio, and a related visualization.

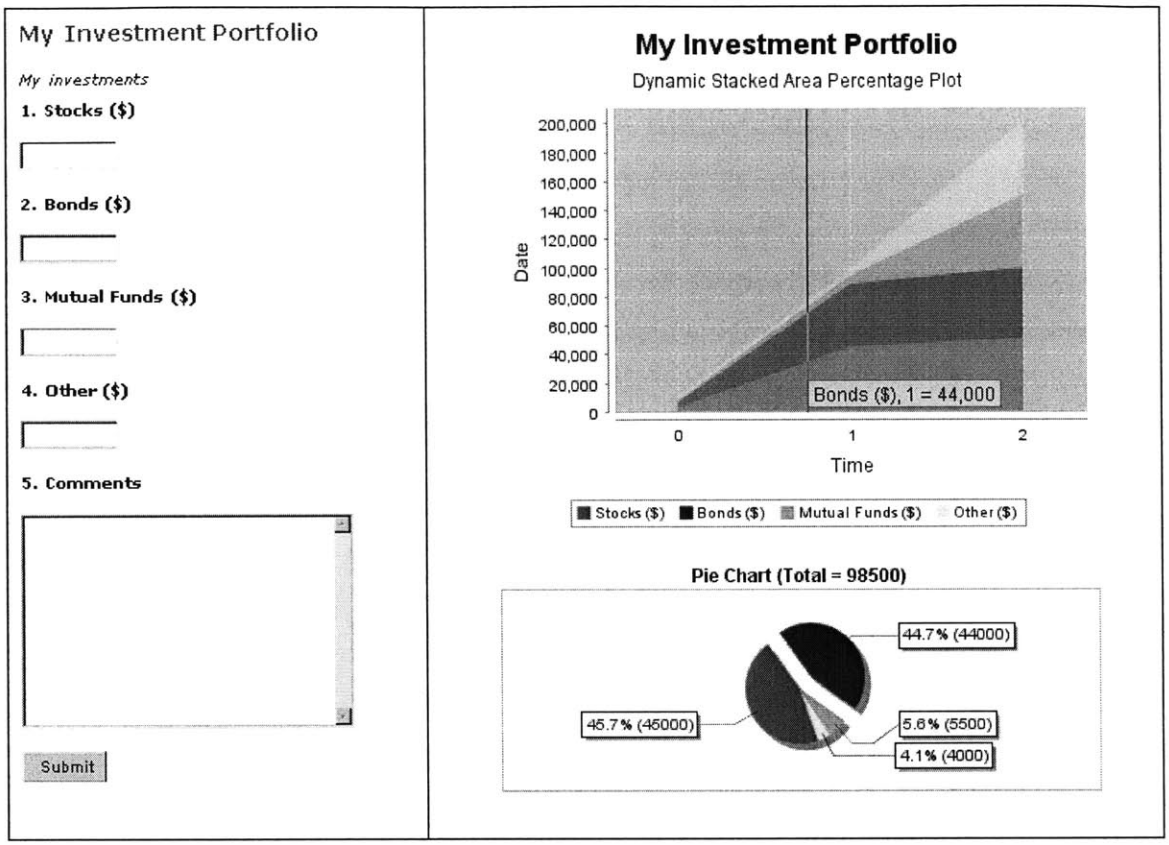


Figure 26: Sample investment tracker and plot

10.3 Integration and Deployment

The integration of the framework into Risk Psychology Network requires reconciling some aspects of the architecture and the data model. The chief benefit of doing so is maintainability; a common foundation architecture reduces maintenance costs. The RPN architecture relies on the Struts application framework, an implementation of the Model View Controller in which form inputs are stored into so-called form beans. A Struts configuration file specifies the page flow. Our integrated framework resides on top of the Struts architecture, using Struts facilities as needed to support our custom validation and security engines.

The common point of integration with regards to the data model is the user information. No user state will need to be maintained in the framework's data model, since they already exist within RPN. We establish a relationship between the users in RPN and the questionnaires that they create, and modify the security engine to check against RPN users.

11 Conclusion

This thesis described a multi-tier framework for collecting, analyzing, and visualizing dynamic datasets. The motivation for this project arose from research efforts within the Laboratory for Financial Engineering that required collection of custom data from human subjects. A database-backed Web site, Risk Psychology Network, collected trait information through questionnaires.

The creators of Risk Psychology Network desired to improve user experience, prompt visitors to provide more accurate and truthful information, and provide incentives for them to return to the site. Components of the framework were developed with an eye toward these objectives. The 2.5D dynamic response distribution plot enhances the site's existing questionnaires by providing a novel way for research participants to understand and compare their results in relation to the results of other participants. The custom data collection component of the framework allows users to build their own questionnaires to track any state information they desire. Users can then use the analysis and visualization components to interpret and investigate trends in their data directly on the Web.

Whether we have succeeded in building a system that improves the ability of LFE researchers to attract and maintain visitors has yet to be measured empirically. There are certainly numerous areas of improvement, some of which we describe in the next section. However, it is hoped that the architecture and tools developed as part of this thesis project will be specific enough to begin meeting the challenges of Risk Psychology Network, yet flexible and extensible enough to handle projects that extend beyond that realm.

12 Future Work

The framework we have described establishes the foundations of a flexible data collection and interpretation system on the Web. However, the implementation is merely a first stab at meeting a few of the many data-centric scenarios. In this case, the needs of Risk Psychology Network guide our data interpretation and collection offerings, with a focus toward providing individual logging capabilities for site visitors. The current implementation does not fully support the publication and management of questionnaires intended to be taken by others. Providing such a hosting service could be highly valuable to researchers and participants alike. A sample scenario is a research study that requires participants to periodically enter state information. Current framework functionality allows participants to track their own responses, but does not support administrative functionality such as email reminders or the interpretation of results from a group of users for the same questionnaire.

Significant potential exists to develop advanced input, analysis, interpretation tools that would showcase the benefits of performing data collection over the Web. The two novel visualization applets we have created provide interesting approaches to interpreting data, but at the writing of this document only one of them is integrated to visualize custom state information. The framework can provide a greater degree of database abstraction so that it is more straightforward to develop analysis and visualization modules without having to handle some details of the data storage scheme. Furthermore, conducting user interface and usage studies would provide a better understanding of whether each visualization or analysis component does in fact meet the needs of site visitors.

Another significant area of further investigation is in the performance of the framework as usage and data flow increase. Any visible slowness to users may very well deter them from continued participation. Potential bottlenecks include JFreeChart, which was not designed to handle graphing a large set of data. In fact, limitations of JFreeChart can already be seen when a large quantity of time series data is plotted; axis labels on the domain axis are compressed to such an extent that the information is indecipherable. To avoid this problem, we condensed the axis labels by sacrificing detailed timing information. Other bottlenecks include the number of servlet calls and database accesses required for many operations.

13 References

[1] Baer, Atar, et. al. "Obtaining Sensitive Data Through the Web: An Example of Design and Methods." *Epidemiology*, Volume 13, Issue 6, pp: 640-645, November 2002.

[2] CNN, "Report: More than 50 percent of U.S. on Internet," February 2002. Available Online: http://www.cnn.com/2002/TECH/internet/02/06/internet.use/index.html

[3] Christian, Leah M. and Dillman, Don A.. "The Influence of Symbolic and Graphical Language Manipulations on Answers to Self-Administered Questionnaires: Results from 14 Experimental Comparisons." August 2003.

[4] Dillman, D. A. *Mail and telephone surveys: The total design method.* New York: John Wiley, 1978.

[5] Dillman, D.A. *Mail and Internet Surveys: the tailored design method.* New York: Wiley & Sons, 2000.

[6] Dillman, Don A., Tortora, Robert D., Conradt, John, and Bowker, Dennis, "Influence of Plain vs. Fancy Design on Response Rates for Web Surveys," presented at Joint Statistical Meetings, Dallas, Texas, 1998.

[7] Goodwill, James. *Mastering Jakarta Struts.* Wiley & Sons, 2002.

[8] Gunn, Holly. "Web-based Surveys: Changing the Survey Process", *First Monday,* December 2002. Available Online: http://www.firstmonday.dk/issues/issue7_12/gunn/#g1

[9] Jakarta Project. "Commons Validator." http://jakarta.apache.org/commons/validator/

[10] Kiesler, S., & Sproull, L. S. (1986). "Response effects in the electronic survey." *Public Opinion Quarterly*, 50, 402-413.

[11] Lo, Andrew W., Mamaysky, H., and J. Wang, 2000 "Foundations of Technical Analysis: Computational Algorithms, Statistical Inference, and Empirical Implementation," *Journal of Finance* 55, 1705-1765.

[12] Lo, Andrew W. and Repin, Dmitry V., "The Psychophysiology of Real-Time Financial Risk Processing," *Journal of Cognitive Neuroscience* 14, pp.323-339, 2002.

[13] MIT Laboratory for Financial Engineering. "Risk Psychology Network". http://www.riskpsychology.net

[14] Newport, Frank, Saad, Lydia, Moore, David "How polls are conducted." *From Where America Stands*, John Wiley & Sons, Inc., 1997.

[15] Object Refinery Limited. "JFreeChart." http://www.jfree.org/jfreechart/

[16] Sheehan, Kim. "E-mail Survey Response Rates: A Review." *Journal of Computer-Mediated Communication*, January 2001.

[17] Solomon, David J. "Conducting web-based surveys." *Practical Assessment, Research & Evaluation*, 7(19), 2001.

[18] Spetic, Ales and Gennick, Jonathan. *Transact-SQL Cookbook*. O'Reilly, March 2002.

[19] StatSoft, "The Electronic Statistics Textbook," Available Online: http://www.statsoft.com/textbook/stathome.html

[20] Tufte, Edward. *Envisioning Information*. Graphics Press, 2001.

[21] Turner, James and Bedell, Kevin. *Struts - Kick Start.* SAMS, 2003.

[22] Veleman, Paul F., and Wilkinson, Leland. "Nominal, Ordinal, Interval, and Ratio Typologies Are Misleading," *The American Statistician*, Vol. 47, No. 1, February 1993.

[23] Watt, J. H. "Internet systems for evaluation research." *Information technologies in evaluation: social, moral, epistemological and practical implications* (pp. 23-44). San Francisco: Josey-Bass, 1999.

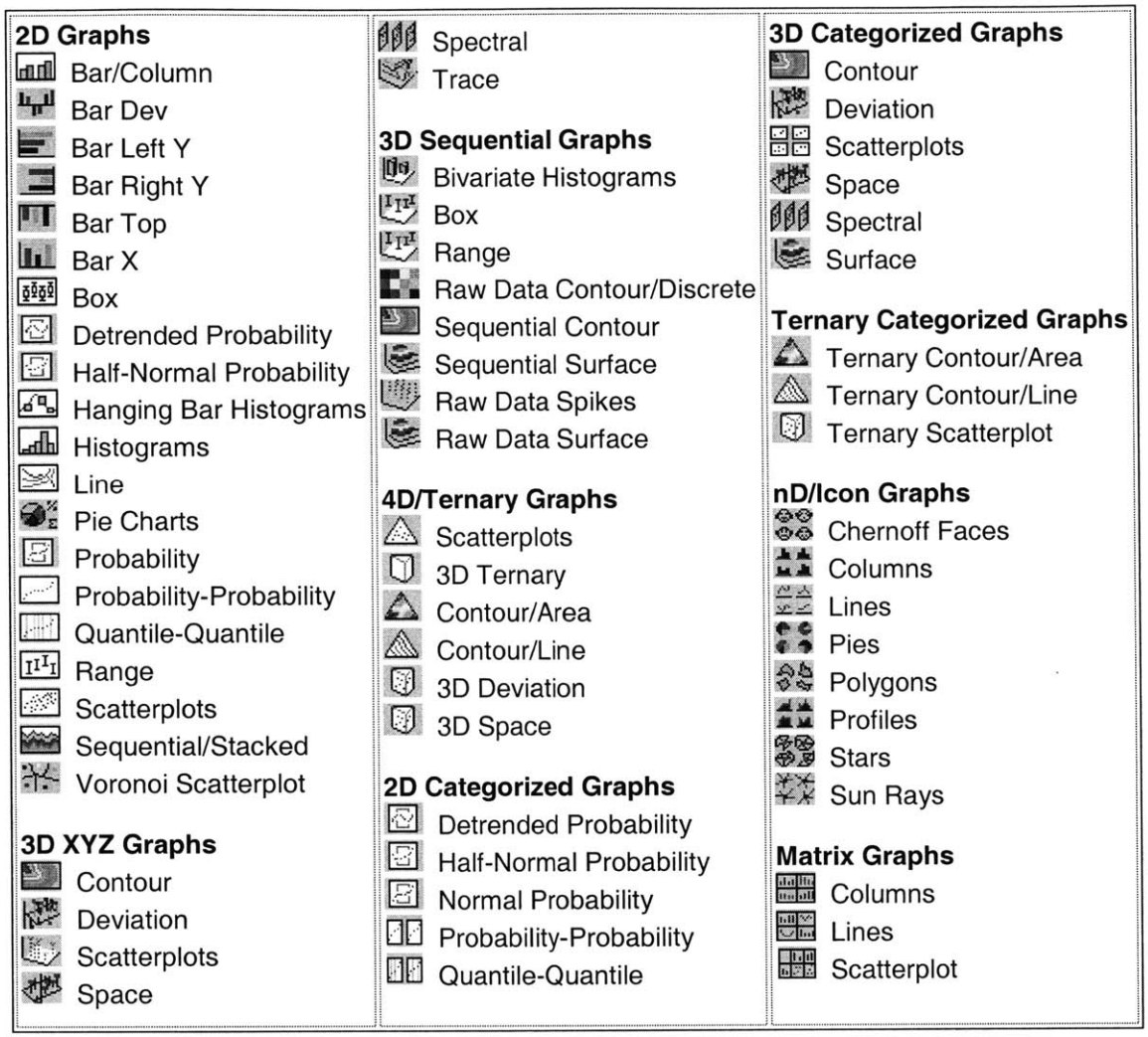
[24] Zhu, Wan Li. "Emotional News: How Emotional Content of News and Financial Markets are Related," Master's Thesis, Massachusetts Institute of Technology, EECS, 2004.

Appendix A: Statistical Analysis Methods[5]

	DATA TYPE			
GOAL	**Measurement (from Gaussian Population)**	**Rank, Score, or Measurement (from Non- Gaussian Population)**	**Binomial (Two Possible Outcomes)**	**Survival Time**
Describe one group	Mean, SD	Median, interquartile range	Proportion	Kaplan Meier survival curve
Compare one group to a hypothetical value	One-sample *t* test	Wilcoxon test	Chi-square or Binomial test	
Compare two unpaired groups	Unpaired *t* test	Mann-Whitney test	Fisher's test (chi-square for large samples)	Log-rank test or Mantel-Haenszel
Compare two paired groups	Paired *t* test	Wilcoxon test	McNemar's test	Conditional proportional hazards regression*
Compare three or more unmatched groups	One-way ANOVA	Kruskal-Wallis test	Chi-square test	Cox proportional hazard regression
Compare three or more matched groups	Repeated-measures ANOVA	Friedman test	Cochrane Q	Conditional proportional hazards regression
Quantify association between two variables	Pearson correlation	Spearman correlation	Contingency coefficients	
Predict value from another measured variable	Simple linear regression or Nonlinear regression	Nonparametric regression	Simple logistic regression	Cox proportional hazard regression
Predict value from several measured or binomial variables	Multiple linear regression or Multiple nonlinear regression		Multiple logistic regression	Cox proportional hazard regression

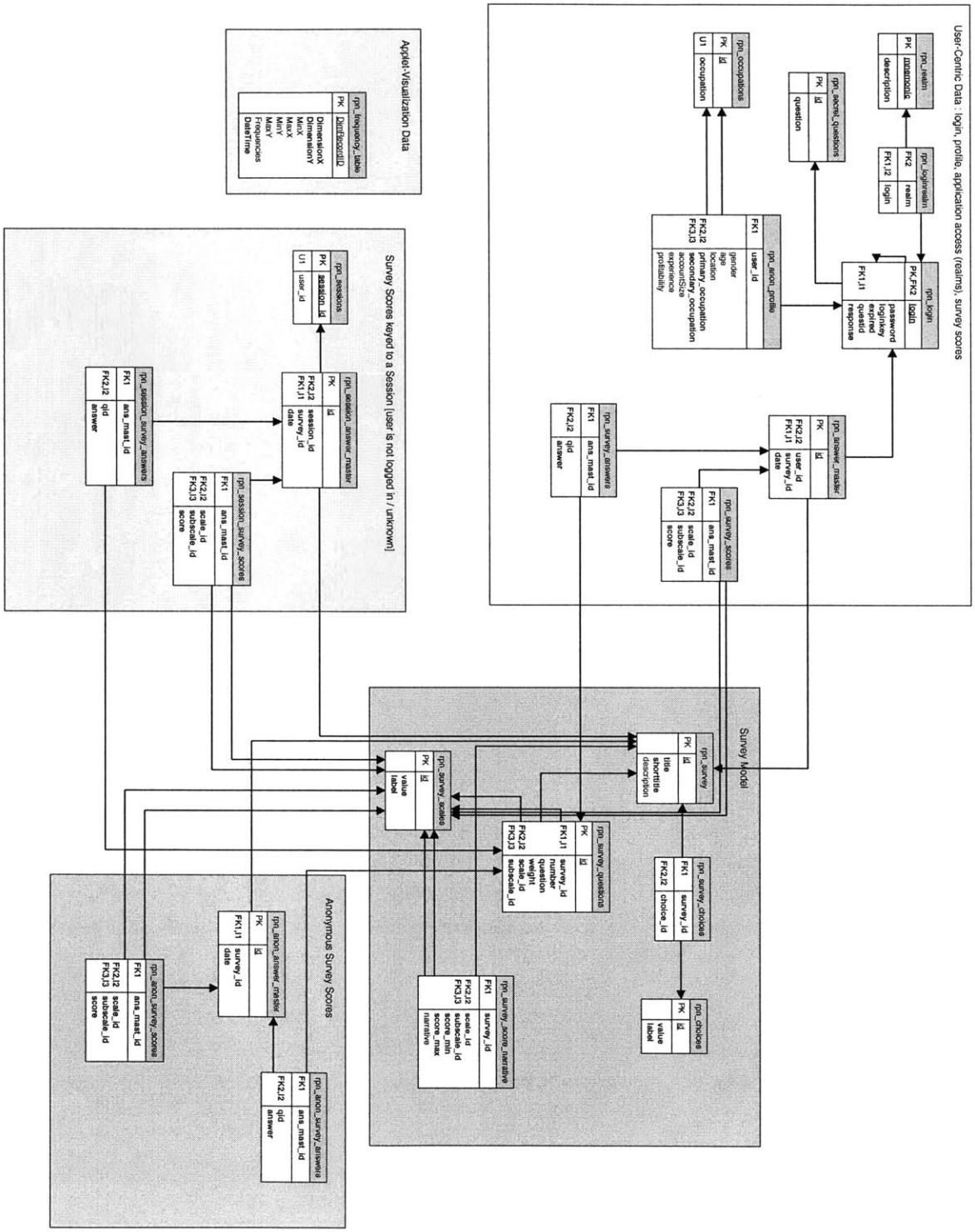
[5] Motulsky, Harvey. *Intuitive Biostatistics*. Oxford University Press Inc., 1995.

Appendix B: Visualizations Index [6]



[6] StatSoft, "The Electronic Statistics Textbook."

Appendix C: Risk Psychology Network Data Model



Courtesy Bradley Smith