Medical Data Visualization for use in Clinical Research and Practice

Brian Leung

A Thesis

in

The Department

of

Computer Science and Software Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of Master of Computer Science at

Concordia University

Montreal, Quebec, Canada

February 2008

Your file Votre référence
ISBN: 978-0-494-40944-2
Our file Notre référence
ISBN: 978-0-494-40944-2

ABSTRACT

Medical Data Visualization for use in Clinical Research and Practice

Brian Leung

The need for medical data visualization within the health care domain to support both clinical research and clinical practice is an important area of research. The need to support clinical research is great as new breakthroughs lead to more efficient and effective treatment for patients that can reduce hospitalization and improve health. With this in mind, this thesis aims to create a real-life software system for the purpose of supporting clinical research and clinical practice in a tertiary care hospital. In this undertaking we employ a combination of the known software engineering methods, namely eXtreme Programming (XP) and the Personal Software Process (PSP). As part of this thesis, we analysed the software requirements in support of clinical research and clinical practice based on which a software prototype is developed. The resulting software is a combination of two software modules: Electronic Flow Sheet (EFS) based on a metaphor used in the real world application and Query Module Processor (QMP) based on the clinical research practices of medical doctors. The software prototype was tested with a live database with real end-users and evaluated for its usability. The results of our evaluation show that our prototype does indeed satisfy the needs of clinical researchers and practitioners. According to our evaluation, our EFS performed better than the current methods used for many of the tasks the doctors did everyday and also scored well on the quality parameters of *Efficiency, Affect, Helpfulness, Control* and *Learnability*. We believe that the user centered approach practiced in this development allowed us to create a software adaptable for both clinical practice and research.

Table of Contents

List of Figures

1. Introduction

1.1 Healthcare and Computers

In the past decades we have been seeing the integration of computers into the everyday practices of many sectors of society. Computers are now an integral part of how companies and organizations function, as computers are efficient for storing as well as processing information. However, the adoption of computers into the healthcare sector has been relatively poor. Currently the integration of computers into healthcare has predominantly been centered on Administration Information Systems (AIS). The use of AIS in healthcare helps ensure the smooth functioning of everyday hospital activities such as scheduling, personnel, finances, etc. Only recently have hospitals started experimenting with, and integrating Clinical Information Systems (CIS) into their everyday ward activities. The reasons for slow integration of CIS in hospitals have been well documented in the literature [1]. One reason for this slow integration is the fear of change from practices that have been in place for decades in the health industry, and addressing this issue carefully has shown to favour integration of computers in nurse's work [2]. Another cause is concerns over the ease of use, confidentiality and security of such systems.

The use of CIS in daily ward activities has almost immediate benefits once nurses and doctors become accustomed to its usage, such as improved quality of care, improved effectiveness of care, and decrease in cost of services. Another known benefit of CIS is that it reduces the amount of errors that at times occurs when using paper-based forms

and methods. Miswritten notes, miscalculated medication doses and misunderstandings from paper-based forms can lead to complications and can even lead to death if severe enough. The CIS helps to remove the misunderstandings that can occur from bad hand-writing and inconsistent data among the stakeholders. It can also be used for decision support as well as medication checking. Another benefit of CIS is that it improves communication between different parts of the hospital. Labs, exams and consults ordered by physicians can be immediately relayed to the proper people. This fast and efficient form of communication reduces the length of time patients need to stay in the hospital. CIS communication also allows for patients to move from hospital to hospital while always having their information available for treatment.

As research into different forms of CIS continue most of its focus has been on improving clinical practice, which is concerned with the treatment of patients. Only recently has research started on ways to integrate CIS and other systems into supporting clinical research. Though clinical research and clinical practice are closely related their needs in terms of computer support differ in many respects. This thesis addresses both concerns.

1.2 Medical Data, Doctors and Stakeholders

One of the major challenges for CIS developers is understanding the nature of Clinical Data and how to represent, store, and manage it. The characteristics of clinical data are analyzed and enumerated below and we comment on how these characteristics can create challenges for CIS developers.

1. Time-Oriented data refers to the property of the data entries that happen over a period of time. Time periods where data arrives or entered by humans vary and the amount of time between entries varies as well. During hospitalization entries may be added daily or even hourly, conversely, once a patient leaves the hospital they may come back periodically for checkups and it may be months or even years between entries. This wide variability between entries creates one type of, though manageable, challenge for developers of CIS.

2. Clinical Data is very dense and multivariate, the amount of data being added to these hospital and research databases grows daily. As patients come in and as new tests become available hospital databases grow both vertically in terms of entries as well as horizontally in terms of new variables that are added. Data is not only limited to one particular hospital; technology has allowed many hospitals to be linked together and to share data between them. The amount of data available to clinical researchers is astounding; however, being able to sort and manage this vast amount of distributed data and deriving information becomes the real challenge [3]. Another challenge arises from being able to accurately filter and query these large datasets for the information the researcher wants to explore. As databases grow in size and the amount of data resources from outside sources increases the amount of time to query and find information grows as well.

3. Clinical Data comes in many forms. In cases such as lab results the values that are stored are in numeric form; however, for some exams there are textual components that are coupled with 2D or 3D images or numeric values. The textual components complement the numeric data by giving an expert's interpretation of the value and its

meaning. Numeric data is easier to utilize as it can be ordered, compared and clustered for different purposes; conversely, textual data provides a challenge in these areas. In the case of textual data, text mining techniques are required to find key words and phrases that can then be associated with the values. Clinical research relies heavily on numeric data because of its properties of being orderable and easily understandable. Numeric data is also important because it is easily communicated to others and has a clear meaning. Once they have the numeric data of interest they may choose to go back and look at the textual data for more information on the context. Image and signal based data are usually interpreted by radiologists and represented as textual reports. A wealth of information available in textual form is currently under utilized and unexplored. Image data are interpreted and converted to texts and numbers.

4. Clinical Data is highly correlated. Because the body acts as one system, variables and their values are dependant on each other such that they will move together in a certain way. Correlated Data is important in clinical research because it helps researchers know how the body as a whole will react to certain treatments.

1.2.2 Data Generators and Users

In the treatment of disease, there are four groups who generate or use data. They are medical doctors, nurses, technicians and patients themselves. These 4 groups of data users will be discussed in the section below.

1. *Medical doctors* working in hospitals usually engage in clinical practice as well as in clinical research. In clinical practice they are concerned with the access and visualization of individual patient's data; whereas in clinical research they view a cluster

of data pertinent to a set of patients selected based on certain criteria that are varied dynamically and interactively. In the treatment of chronic diseases or complex cases like organ transplants or cancer, doctors follow predetermined "protocols". Establishment of such protocols and their adaptations to cases are not uncommon. Moreover, in complex cases, multiple doctors work as a team to treat these patients. The visualization requirements for clinical research and clinical practice are based on the doctor's goals and the team requirements. While clinical research works towards gaining new knowledge about treating patients; clinical practice works towards applying this knowledge towards treating individual patients. In clinical research, knowledge is gained through understanding past experiences and adherence to protocols and formulating hypothesis to describe what has been seen. If a new protocol offers a better outcome than a current protocol then clinical practice can be changed to what has been discovered through clinical research.

In reference [4] medical doctors are characterized as follows:

1) Intelligent

2) Computer naive

3) Little time to learn

4) Performs retrievals occasionally

5) Antipathy towards computer

6) Has no mental modal of database;

7) Has good knowledge of information in database; and

8) Has poor knowledge of database names.

The characteristics listed above must all be addressed when creating CIS to assist doctors when they are performing clinical research and clinical practice; that is our goal in this thesis. Clinical practice and clinical research are very closely knit, as clinical practice provides the experience as well as the data needed for clinical research such that clinical research can improve clinical practice.

2. *Nurses* are another category of stakeholders in the research and development of CIS. Nurses are one of the primary care providers to patients and also one of the primary users and data creators of the CIS. It is well known that no CIS will ever be successful without being able to satisfy and suit the work practices of the nurses. Nurses play a primary support role to the doctors on the ward. As doctors diagnose and prescribe treatments for patients, it is the job of the nurse to carry out those orders. Nurses must also have enough medical knowledge to deal with emergencies as they occur, and to notify doctors of changes appropriately and swiftly. The main goal of nurses is the caring for and treatment of patients.

3. *Technician* is a broad category of healthcare workers who help in gathering data for the CIS. They are the points of contacts with patients and use a variety of instruments, tools, and processes to capture data directly or indirectly. Blood technicians, x-ray technicians and technicians operating the instruments like scanners and analyzers all belong to this category. The data generated by these technicians are normally used for diagnosis and are normally ordered by doctors when more information is needed about a patient's condition. Technicians are not users of the data, in general.

4. The last category of stakeholders who generate data are the patients themselves. The data they generate as input we categorize into 4 types (a) symptoms and signs, (b)

qualitative and quantitative information, (c) system-based data such as cardiac and pulmonary data, and (d) historical information. In the modern times, patient awareness regarding the disease, the treatment process and the medications is considered beneficial in many ways. In this situation, the CIS can help to give individualized patient education as opposed to the mass education in practice today mainly through printed pamphlets or word of mouth.

1.3 Problem Statement

Medical Data Visualization is a growing field of research as people are realizing that Clinical Information Systems (CIS) have major benefits, such as improved patient care and information tracking [5,6,7], within clinical practice and clinical research. As adoption of CIS grows, the need for efficient and effective visualization of an array of different formats of data, as well as the massive amounts of data, is necessary. Though Data Visualization is not a new field of study its application and focus in medicine has only taken shape recently. Because of the improvements in technology that allows for the storage of massive amounts of patient data in multimedia form and their ready access available at the point of care, the need to view this information in a useful manner has been steadily increasing.

Medical Data Visualization refers to the different methods of viewing, presenting and exploring data such that it improves the understanding of the massive data by different stakeholders. Without a means to analyze and explore this massive amount of data, clinical researchers cannot find new methods and protocols of treatment that would improve clinical practice. Each of these stakeholders holds different goals, have different

needs and preferences when visualizing patient data. Currently, there does not exist a complete knowledge of differing visualization needs of both clinical practitioners and clinical researchers.

Following established software engineering practices and the principles of interactive information system design in this thesis our goal is to iteratively develop a prototype to understand and address the needs of clinical research and practice. The domain of applications is CIS with a specific focus on clinical practice and clinical research. This being a very large domain in itself we restrict ourselves to the activities in a transplant clinic in a university teaching hospital in Montreal. Thus, we will deal with a real and live database, real stakeholders, and face the real-world problems. Our first aim is to analyze the current practices in the said clinic and develop clear requirements for a software prototype. Then apply a systematic methodology while keeping in mind the constraints of the target users. When completed, the prototype would be field tested by the real-world users to draw conclusions. In the selected hospital for our study a transplant database exists that is constantly updated. It deals with more than 1000 patients of different organ transplant cases such as kidney, liver, pancreas, etc; over the past 10+ years. We are given a privileged access to this database to demonstrate the usefulness of our prototype and our approach.

1.4 Thesis Organization

This thesis is organized as follows: Chapter 2 discusses various data visualization techniques and how they are applied to medical data. We will also discuss various projects that are currently trying to support medical data visualization for clinical

research and practice. Chapter 3 will discuss the requirements for our proposed solution as well as the needs of both clinical research and clinical practice. Chapter 4 presents an overview of our design and methodology as well as some of the difficulties we faced in our development. Chapter 5 describes our evaluation and prototyping with users and with a live database. The results and our analyses of our results are also presented in chapter 5. Chapter 6 provides a summary of this thesis and recommendations for future work.

2. Related Literature

As far as the related literature to this thesis is concerned, there are four aspects to this thesis: (1) following an established software process to arrive at a software requirement for an interactive system intended for medical data visualization, (2) software prototyping, (3) real world application of this prototype and (4) system evaluation. Considering reviews and books that have been written on some of these topics, in this chapter we restrict our coverage of the related literature specifically to medical data visualization because that is the focus of this thesis. In this context, we are interested in both clinical practice and clinical research. The literature related to this is relatively limited and we try to establish the state of the art in this chapter.

2.1 Interactive Use of Medical Data

As stated by previous researchers, information visualization is "the use of computer-supported, interactive, visual representations of abstract data to amplify cognition" [8]. Visualization of medical data provides a useful abstraction of large amounts of data that otherwise overwhelm clinical researchers and practitioners. Different visualization methods allow for enhanced views of the data while emphasizing different characteristics. Along with visualization, interacting with the data allows researchers to explore and to manipulate the data in ways that are more meaningful to their individual needs and the problem needs. Interaction methods also allow medical doctors to use their intuition, experience and human intelligence to find and discover new areas of interest by exploring the voluminous clinically collected data. Visualization of

medical data can be complex because of the characteristics of clinical data which is time-oriented, dense, multivariate (containing both textual and numeric data), and having several correlated variables. These characteristics make viewing and interacting with this data hard to handle effectively and efficiently when performing clinical practice and clinical research. Human experience and intuitions also play an indispensable role along with measured quantitative data.

There are several methods to handle, interact and visualize clinical data. Traditionally, visualization and interaction methods such as, graphing, time-lines, and tables are used to address one or several of the characteristics of clinical data. Though it is hard to address all these issues, ongoing research is being done to improve the tools for researchers in this area. Examples of these methods include using 2 Dimensional and 3 Dimensional graph spaces to accommodate the dense and multivariate data. Another method of visualization and interaction is the Timeline which presents data in a linear, temporal sequence. To handle the correlations in clinical data they can be displayed as stacked line graphs so that the variables that are dependent on each other can clearly be seen moving together. Stacked line graphs can be used as a visualization method and as a pattern discovery method as well. In the sections to follow we will give several methods of visualization and interaction and how they apply to clinical data and clinical research. In most cases a combination of methods such as tabular, graphical, and timeline views becomes a better solution; but the challenge in merging methods is to handle large amounts of data in a timely fashion and to avoid cluttering.

All interactive visualizations involve a human in the loop as shown in Figure 2.1. The human performs three different operations: (a) Selecting or specifying his/her

interest on what needs to be viewed (b) optionally exercising control on the choice of visualization parameters, and finally (c) interpreting the displayed data in light of the human's expert knowledge and possibly relating it to what was observed in the previous iterations. The general paradigm is true for all interactive applications. What makes this different in medical data visualization is (i) the doctor might do this either for clinical practice when the patient is sitting in front of them physically or remotely, or for (ii) doing clinical research viewing data pertinent to hundreds of thousands of patients; but not (i) and (ii) at the same time. Quick response time and convenient access are highly critical for the success of a system in case (i) but not by the same degree in case (ii). In certain medical research areas, like Paediatric Lung Transplant, the volume of data available for research could be insufficient at any one hospital, which results in a need to pool data from multiple centers that may be widely distributed nationally and internationally.

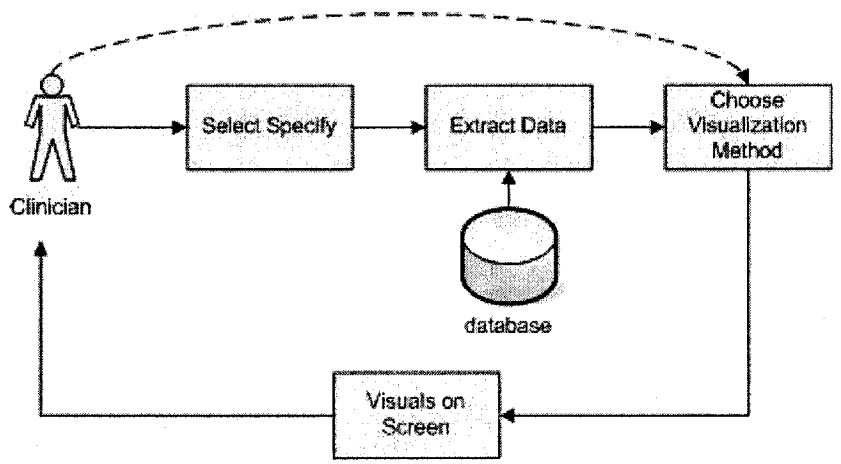


Figure 2.1- Interactive Visualization

2.2 Graphical Visualization

2 Dimensional and 3 Dimensional graphs have been around for many years and have a wide range of applications in various fields. These types of graphs have many uses in Clinical Research when needed to display data that is dense and that contains multiple variables. This abstraction allows for researchers to view trends and correlations easily instead of having to deal directly with large amounts of numeric data. Interactions with 2D and 3D graphs include narrowing, expanding the dimensions of the graph, and slicing along any of the multi-dimensional axes in order to reduce to 3D space or to a 2D space of interest. When using 3D visualization we can usually label the 3 axes with the following: patients, time, and medical data variables. On occasion, multiple variables can be drawn into the 3 dimensional spaces. This division of axes allows researchers to view large amounts of data at the same time and makes viewing similarities between multiple patients easier to see. 3 Dimensional spaces can be reduced to 2 Dimensions by slicing along the various axes as seen in Figure 2.2.

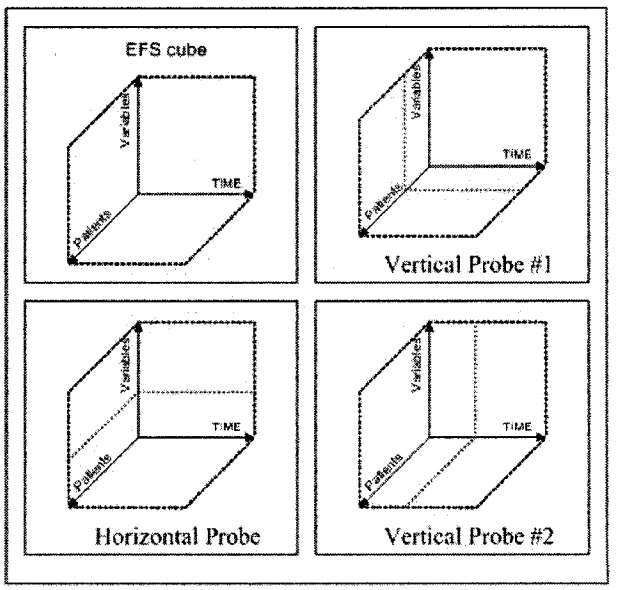


Figure 2.2-Slicing along Axes to Reduce Visualization Space in 3 Dimensions

An example of a use of 3 Dimensional Visualization in Clinical research was an experimental application created to help in the haemodialysis ward of a hospital in Italy. The researchers named their application Interactive Parallel Bar Charts (IPBC), Figure 2.3, in their context was intended to aid in Visual Data Mining (VDM) [9]. Haemodialysis is a method of removing unwanted waste from the blood of patients who have kidneys with reduced or no function. Patients must sit for long periods of time as blood is passed from their bodies through machines that perform this operation. As the blood is cleaned, readings are taken to ensure that the patient is stable. By connecting these haemodialysis in real-time, and displaying data in an easy to understand manner, the researchers manage to reduce the visual space from 10 different views and monitors to a single one.

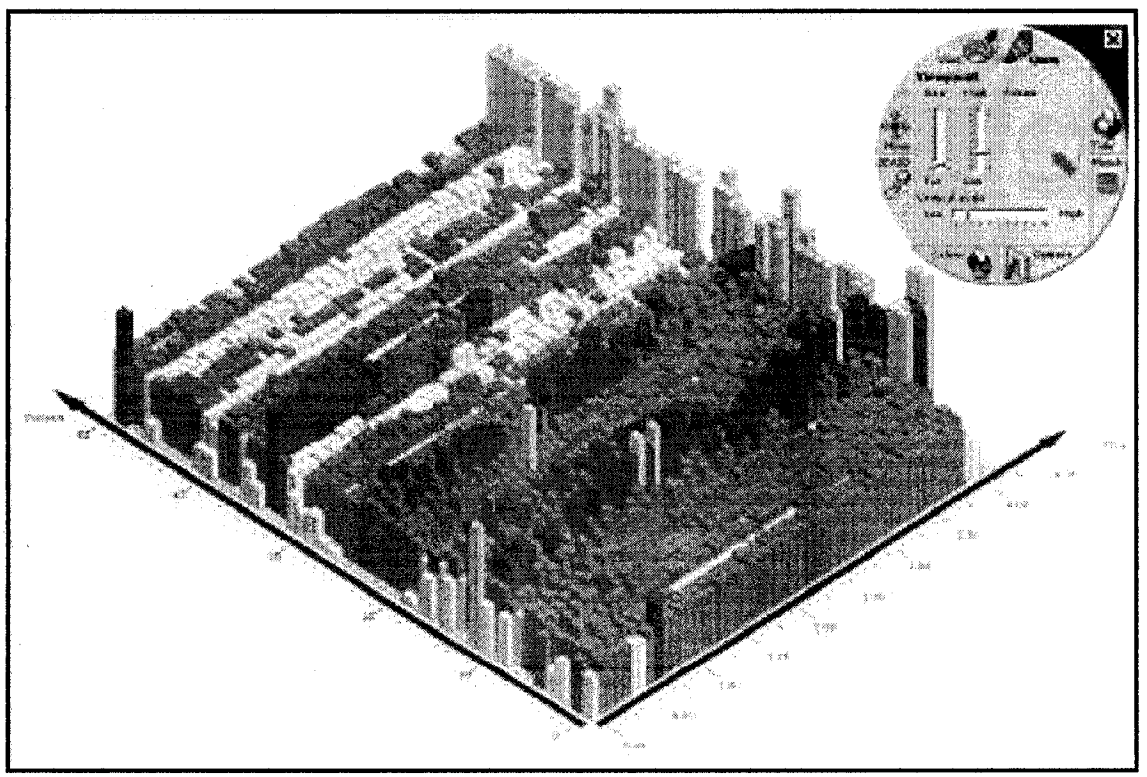
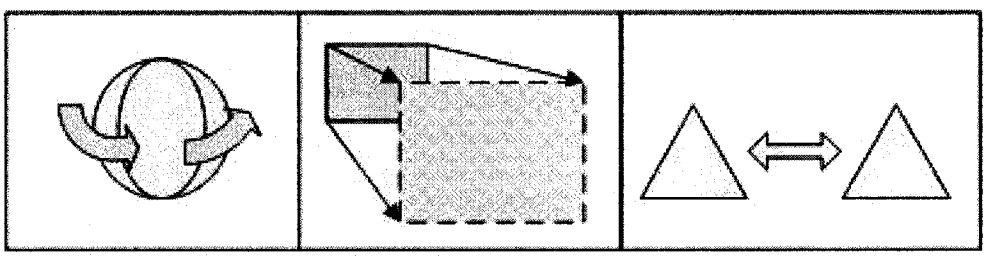


Figure 2.3-Interactive Parallel Bar Charts (IPBC)

The visualization method that these researchers provided was a 3D coloured space where the x-axis contained time, y-axis contained a range of values and the z-axis contained "bar charts" in an orthogonal view. Obtained bar charts could be either multiple bar charts of different patients or a single patient's previous haemodialysis session. Showing readings for previous haemodialysis visit allowed the doctors to adjust treatments for better results. Given only a 3D visualization for patient data is adequate for clinical practice; however, it is insufficient for research and VDM. "Operators and interactive tools" must also be provided to medical researchers so that they can navigate the visual space and narrow down areas of interest. These operators also help in overcoming certain problems that occur with 3D visualizations of large amounts of data and allows researchers to interact and visually mine it for interesting patterns. One problem that occurs with 3D visualization is occlusions, which refers to large forward objects blocking the view of smaller objects behind it. Another problem is, as the amount of information increases, patterns and correlations become less visible and at times may not fit within the physical visual space of the screen. The 3 operations that were created to deal with these problems are:



1. Rotation 2. Zooming 3. Scrolling

These operators allow the researcher to see the whole picture from different angles. The rotation operation would allow researchers to view around occlusions in the data. The zooming operation would help researchers view interesting bits of information that at

times may be hard to see with large datasets. The scrolling would ensure that all the information is potentially viewable by scrolling to sections that are not currently on the screen.

Use of 3D visualization for VDM simplifies this process greatly and makes it effective for medical research. Researchers no longer need to analyze large datasets from tables of numbers and can now look at general trends and movements of variables. Visual data exploration must offer powerful analysis tools to researchers while not requiring them to learn complex operations and functions in order to use them. Visual data exploration differs from automated methods because it allows the researcher to be directly involved and augment with human intelligence the data mining process, to discover for themselves new ideas and to explore the datasets on their own. Given the proper tools, VDM within a 3 dimensional space is very beneficial. Researchers can color certain ranges of values, color certain points which are above or below a single fixed or variable threshold value, as well as view general declines and inclines in values. All these help towards understanding the large amounts of data presented and coming up with new hypotheses. This 3D visualization also gives a usable abstraction from the actual data and helps to determine where that data may be heading. One of the benefits of this type of visualization and interaction methods, experienced by the creators of this software for haemodialysis research, was that their software was intuitive and easy to learn after only a few uses. They also found that users experienced difficulty when using more than 3 views at a time as the visual area got too cluttered.

A novel approach to using graphs for VDM was an application that was created in a different field for mining financial data reported in [10]. It has applications within the

medical domain as well. As financial data carries some of the same characteristics of clinical data such as its time oriented nature, this application's method of mining could be used for clinical researchers as well. The query interface for this application is a blank panel with a simple x-axis denoting time and a y-axis denoting values. In order to interact and query the database, the user must select a variable and then draw the graph of values that they want to see. Once the graph is complete, the application automatically queries the database and finds all instances of the variable which follows the particular curve that has been drawn. In other instances, a graph can be drawn in which case the application will find all variables that satisfy the drawn curve. By finding all variables that satisfy a curve we can find correlations easily in the data that may exist. We can see an example of this application in Figure 2.4 where a user has drawn a curve and the results are seen at the bottom. The results can be selected to bring the user to additional information about the particular matching variable. This is a very promising interaction method for VDM and querying as it does not require users to learn querying languages or unfamiliar interfaces as most doctors are often too busy to learn new techniques. This method also allows for the user to be directly involved in the data exploration process.

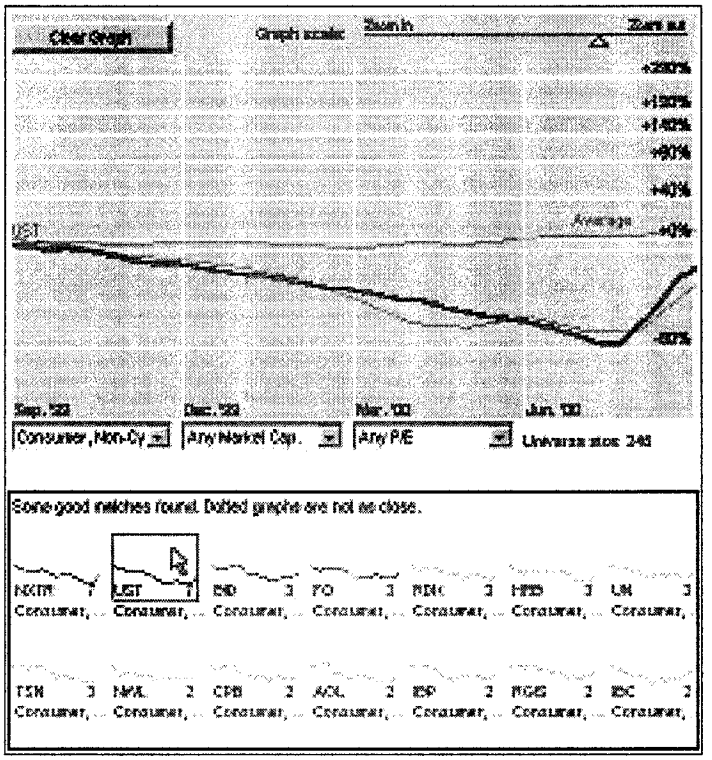


Figure 2.4-Interface showing Results of a Query

2.3 Timelines and Data

Any visualization that does not take into account the time oriented nature of clinical data will be lacking in a key aspect and characteristic of clinical information. Although one of the dimensions of 2D or 3D visualization could be time, multi-dimensional views focus on dealing with large amounts of data and how to display them. On the other hand, timelines do not focus on large amounts of data but almost focus exclusively on the time-oriented nature of the data and how to reason among them and order them meaningfully. In doing this, timelines abstract the data as objects and place them on a linear time-line. One important benefit of using timelines is that they handle textual data easily unlike 2D/3D graphs. Because textual data expresses opinions, descriptive observations and notes, they cannot be expressed in numeric form and placed

on graphs or bar charts, however, they can be placed on a time-line as text objects depending on when in time this textual data was created.

When studying time-oriented data and creating a visualization, 3 choices exist to represent time. One choice is to display time as *instants of events* or as *intervals of time.* Displaying the instants of events has the benefit of showing non-related incidents together that occur over a wide range of time. Displaying intervals of time allows for the grouping of related events. An example of displaying time intervals would be an interval called *renal failure* and then having the events and test displayed inside that interval.

Another choice when displaying time oriented information is whether to display *linear*, *branching*, or *circular times*. The first, linear, in Figure 2.5a, is the most common as that is the normal path that time takes. When viewing 2 dimensional and 3 dimensional data it is usually in a linear fashion as time is straight-forward, Branching timelines, Figure 2.5b, usually refers to differing paths of treatment where one form of treatment will bring a differing timeline than another form of treatment. Branching is usually useful for decision making and finding the best course of treatment for a certain disease. Branching is also good for clinical research as it helps researchers come up with new ideas and new hypotheses that may have better branching results. Circular timelines, Figure 2.5c, can be used for chronic and re-occurring events with similar treatments and diagnosis patterns.

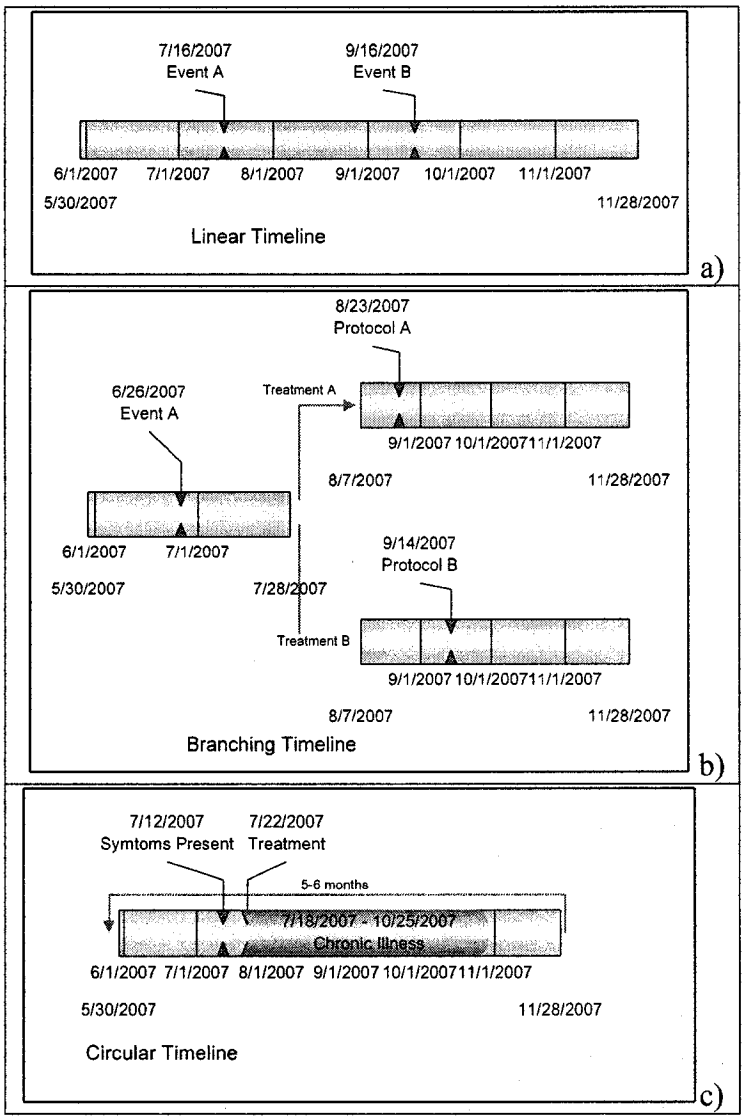


Figure 2.5-a) Linear Time b) Branching Time c) Circular Time

Another decision when using timelines in visualization is whether to use *absolute* or *relative time*. Absolute time is usually associated with events such as dates in a clinical database. Relative time is usually used when referring to a general occurrence or occurrences relative to another point in time that is not absolute such as the "Two months after Transplant".

Interacting with Timelines can be done in several ways. Allowing users to query and group temporal abstractions are one way to allow researchers to explore and navigate the data. Other tools could allow researchers to explore and provide visualization for

these temporal abstractions. However, researchers working with timelines [11] find that by providing useful visualization interface, one can remove the need for many queries that researchers may have. These researchers are also working on ways of using domain specific knowledge to augment the amount of "intelligent" support the interactive systems could provide, as an example, using the domain specific information to determine the level of information abstraction as well as the level of temporal granularity to be provided to the user at a terminal.

2.3.1 GANTT Charts

A popular use of Timelines in visualization is the use of GANTT Charts as seen in Figure 2.6. GANTT charts are normally used for managing projects and handling hierarchy of tasks with respect to time. GANTT charts normally consist of a list of tasks on the right hand side ordered hieratically with lines representation the length of time each tasks takes. Within clinical practice, GANTT charts could be used for planning of patient therapy treatments as well as planning overall clinical tasks. The main drawback for these types of charts is that as the number of tasks increase, the amount of space required increases dramatically [12].

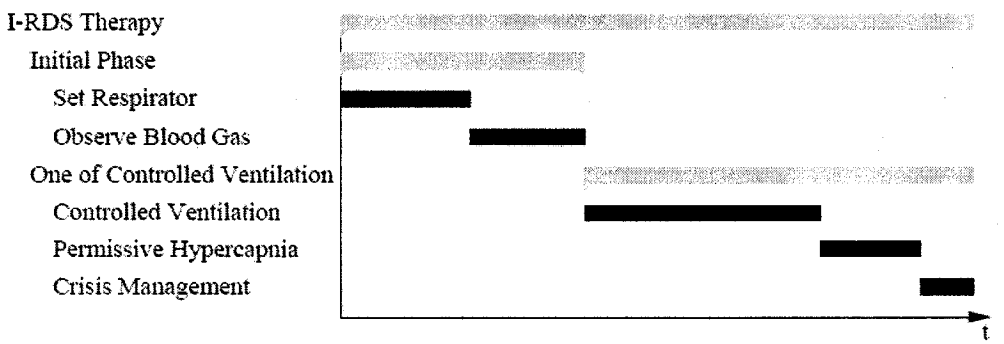


Figure 2.6-GANTT Chart example [12]

2.3.2 Paint Strips

Another extension of timelines explored for its use with medical data is called Paint Strips [13]. This approach uses the paint-roller metaphor as a way to represent lengths of time between tasks or intervals of time with respect to one another. As an added feature to this technique, we can add relationships between paint strips as well as combine multiple paint strips. Paint strips that are related to each other can be shown by connecting a 'rope' to each related paint roller and connecting them to a weight as shown in Figure 2.7. Due to the simplicity of this metaphor, the researchers associated found paint strips easy to use.

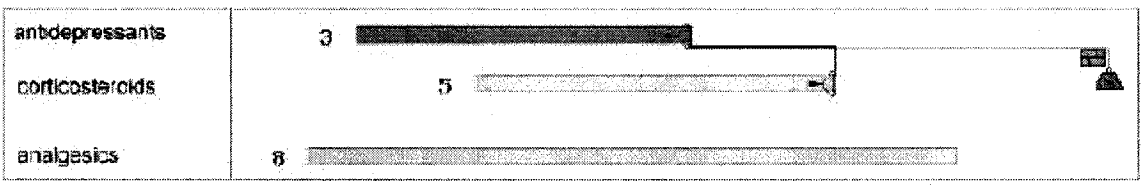


Figure 2.7-Paint Strips

2.3.3 Lifelines

Based on the time series, Plaisant's research team introduced the concept of "Lifelines" for visualizing the information contained in electronic patient records [14]. Lifelines, shown in Figure 2.8, provides a general visualization environment for viewing patient's histories. The 3 main goals of the Lifelines application are:

1) Present a patient's history on a single screen,

2) Provide easy and quick access to all additional information that a doctor might need, and

3) Allow for software generated alerts and viewing critical information at the overview level.

They provide several functionalities that help in the exploration and analysis of patients' data. Information about a patient's diagnosis, medications and treatments can be viewed by simply moving the cursor over the item in question. Another functionality provided in Lifelines is zooming. Zooming allows doctors the ability to see details in a patient's timeline at a detailed level. In addition to zooming, lifelines also allow doctors to perform text searches based on keywords from a patient's history. These keyword searches also highlight all related events, medications, and processes.

Researchers [14] found that doctors performed their tasks 50% faster using Lifelines than with normal patient records. Response times were particularly faster for questions involving intervals of events and categories of events. The retention of this data, the amount of information the clinical staff remembered after seeing it, also increased.

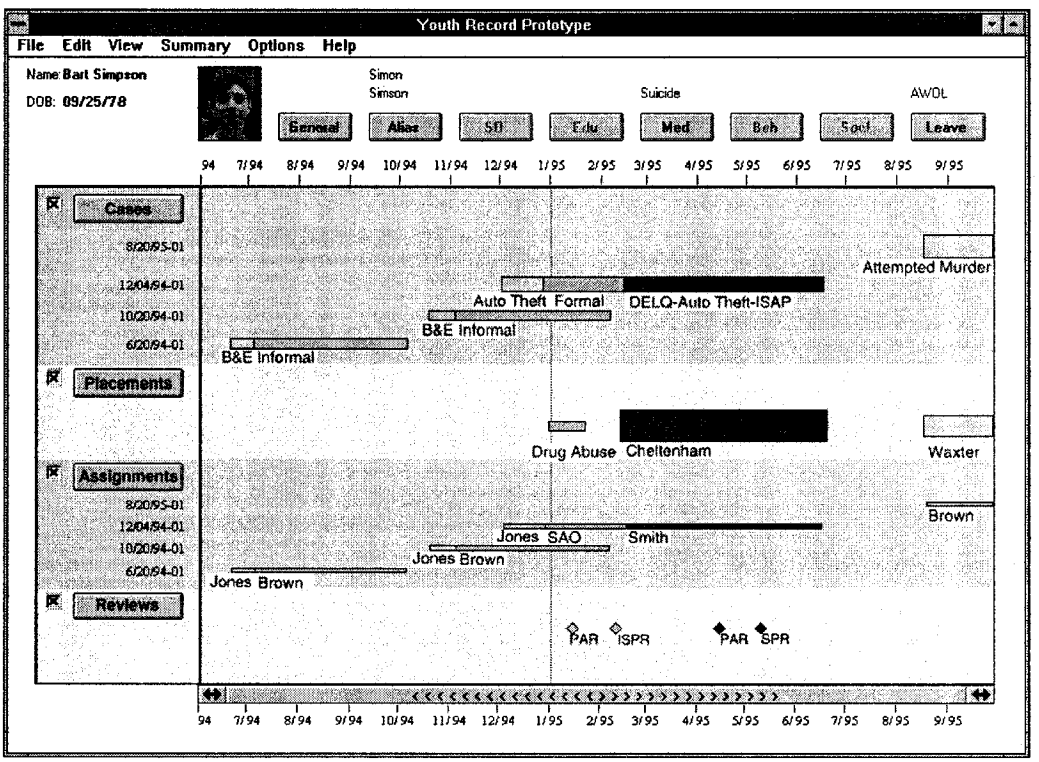


Figure 2.8-Lifelines User Interface

2.3.4 Asgaard/Asbru

Another notable research project related to medical data visualization is from Germany [15] and it is titled "Asgaard". As a part of this project they have developed a specialized language called 'Asbru' to describe the clinical protocols in the medical domain. The Asbru syntax resembles LISP making it difficult for physicians to learn and to use effectively and the developers have created a user friendly interface that makes working with Asbru easier. Along with a LISP-like syntax, there are many domain specific keywords as well as conditions and parameters that can be added to these treatment plans. Asbru also allows specification of "functions" where tasks and sub-tasks can be linked together to create bigger "functions". Defined tasks can be reused in different situations and for different protocols. Time annotations are also a key aspect of Asbru, Asbru uses the following Time Annotations when specifying tasks. Figure 2.9 shows an example of these annotations.

1. Reference Points: a reference point for when the following task should be performed. May be abstract or finite.
2. Earliest Start Shift (ESS)
3. Latest Start Shift (LSS)
4. Earliest Finish Shift (EFS)
5. Latest Finish Shift (LFS)
6. Minimum Duration (MinDu)
7. Maximum Duration (MaxDu)

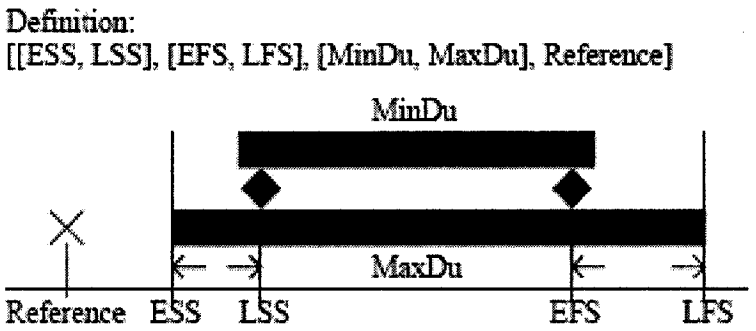


Figure 2.9-Asbru Time Annotations

Asbru is a complex but powerful language for specifying treatment plans and protocols. With more research going into useable tools for manipulating the content specified in this language we will know in the future how physicians will find this language useful. Another benefit of having a domain specific language is that it is easy to communicate to other doctors and physicians using the same tools.

2.3 Knave & Database Querying and Mining

Within the past few years researchers, at Stanford University, have developed the KNAVE software framework [16]. KNAVE stands for Knowledge-based Navigation of Abstractions for Visualization and Explanation. KNAVE is a collection of tools that work together to provide interactive summarization, visualization and exploration of time-oriented clinical data by providing multiple levels of temporal abstractions. By using temporal abstractions, they can summarize events and reduce the amount of information that can otherwise overwhelm users. The architecture of KNAVE is shown in Figure 2.10. The KNAVE architecture can be divided into 3 parts, database and abstractions, ontology and knowledge-bases, and the user-interface.

The database and abstraction part of the KNAVE architecture is responsible for creating useful temporal-abstractions from the database. This portion will access multiple databases to provided answers to queries provided by the user. The ontology server provides domain specific knowledge to other parts of the architecture. This could be used to get relationship information about objects in the database that can help with interpreting and understanding the data. The visualization module contains the interface as well as exploration tools needed to navigate the vast amounts of data being retrieved by the databases.

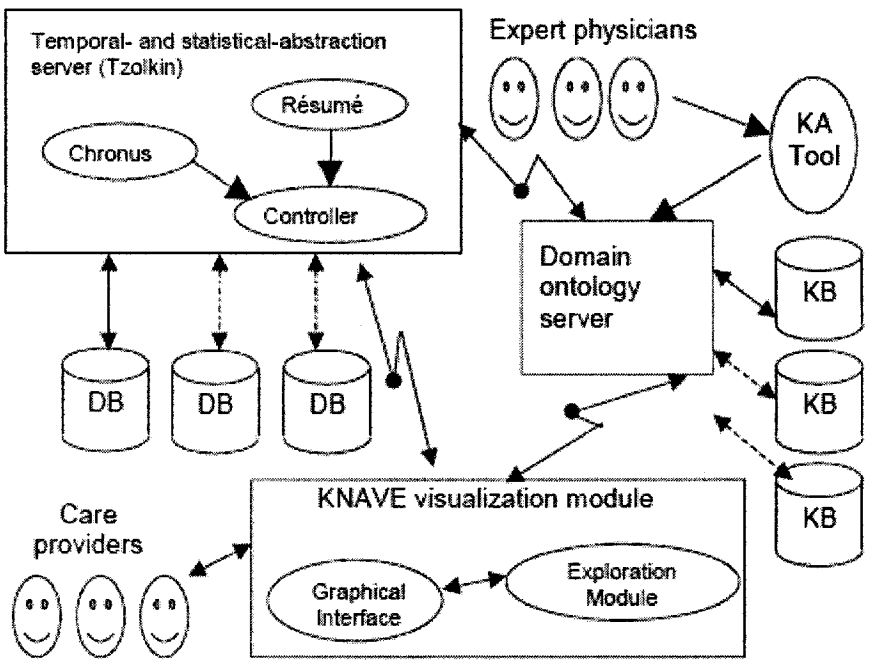


Figure 2.10-KNAVE Architecture

The researchers involved with this project breakdown the relevant functionality into the solution of 3 main subtasks, *temporal abstractions, knowledge acquisition, and information visualization.* The main interaction module of the KNAVE system is the temporal-abstraction visualization interface. Using this module, the user can query

domain-specific information in a temporal manner. The visualization tool then allows the user to explore these temporal abstractions as they see fit. This is where KNAVE differs from almost all other visualization techniques available. Whereas most visualization is focused on the display and interactions with raw data, KNAVE works on the visualization of domain specific temporal abstractions. Researchers of this project have noted the benefits of their framework with a variety of medical tasks such as therapy planning and patient monitoring [16, 17].

Querying of medical databases is a complex process because of the size of medical databases. This complexity of system design and dynamic user needs make it nearly impossible for doctors and nurses, who are not database proficient, to retrieve data that is of interest to them. In order for users to query medical databases, they would need to know the relationships between tables as well as the field and table names. They would also need to know when field names refer to specific lists and where these lists are located. Even with query building software, which would reduce the need for users to learn the SQL query language, the time needed to learn how to use these softwares is still unrealistic for doctors who are already very busy. The KNAVE project does allow some form of querying, though almost completely focused on temporal data and their abstractions. By providing this domain specific querying with a simple interface, users do not need to learn complex query languages in order to retrieve information. The KNAVE-II visualization and exploration module is divided into 2 portions, the user-interface module and the computational module that formats all data going into the UI. The KNAVE-II query model involves 4 specifications that must be defined, the parameter or variable, the value of the parameter, the context of the parameter and finally the time-

span or temporal parameter. Using these 4 specifications, data can be retrieved form the database and be displayed. The KNAVE-II UI is also divided into 2 portions where the left side is used for specification of queries and the right for displaying and exploring the data. The evaluation of the KNAVE-II user interface followed the 5 usability dimensions defined by Nielsen [18] which are learnability, efficiency, memorability, error-tolerance, and satisfaction. Using 8 subjects ranging from doctors, medical residents and fellows, the researchers were able to determine that their UI was successful in all 5 dimensions of usability [17].

2.4 Summary

In summary, we found the published literature is quite sparse in the area of medical data visualization. In our case, we did not consider the aspects of visualization of images like CT, MRI or PET scans which are special cases of visualization. The KNAVE project is one of the best efforts in this field. Much research needs to be done especially in using or abstracting the domain knowledge of the experts in an effective manner for interactive visualization. We also find the requirements will have similarities and differences when considered for clinical research support versus the support for clinical practice. When looking at most of the commercially offered medical software we find that most of them rely heavily on spreadsheet style visualization for numerical and textual data. Some software also provides timeline visualization and GANTT charts to provide more information about the temporal data and tasks. The characteristics of clinical data means that software created to visualize such data must be able to handle differing frequencies of data as well as higher dimensional and relational data.

3 Requirements Analysis and System Requirements

3.1 Needs at Large

The needs of clinical researchers and clinical practitioners are many. Among the implemented commercial systems great strides have been made in the area of Clinical Information Systems (CIS) to better satisfy the needs for clinical practitioners [20]. A typical CIS satisfies the needs of efficiency, accuracy and reliability of the data accessed by clinical practitioners while treating patients. A major goal is to provide timely access to the electronic health record that is up to date, consistent and error-free. Data sharing research and development require that electronic health records be truly portable across organizational boundaries. This would be useful if patients visit multiple hospitals or clinics for treatment and would give clinicians access to a complete patient record. The ability to share patient records across organizational boundaries and standards still requires a lot of research in the areas of distributed databases, ontologies and privacy policies of patient information.

Though CIS satisfies the needs of clinical practitioners in a relatively better way, it does not provide the functionality required for clinical researchers to do their work. Whereas clinical practitioners are focused on individual patients and their status at the current moment of patient's visit-time, clinical researchers focus on groups of patients and the data-relationships to each other at different windows in the past and try to extrapolate better methods of treatment and protocols. Because of this focus, clinical researchers need a way to view multiple patients' data together in a correlative manner

and to navigate through the data easily on demand basis. Clinical researchers also need different ways to view and analyze the data since viewing strictly numerical data does not always give a complete understanding of the trends and correlations in the context which might be present in other forms. Different views can also remove a lot of the mental overload on the part of the human explorers that occurs from viewing the dense and multivariate nature of clinical data. Along with different methods of viewing medical data, researchers also need ways of exploring, analyzing, and interacting with this data. In a "human centered manner" the navigation and visualization methods must be intuitive, easy to use and to learn, and adapt to the dynamic needs during interactions. Another important aspect of clinical research is that it can be a collaborative activity between many researchers who may be located in other organizations. As doctors may be located in different areas, researchers need an efficient and effective way to relay their thoughts, hypotheses, proposals and findings to others, and also a way to request data from other places. Both PUSH and PULL techniques are useful in this context. The area of Computer-Supported Collaborative Work (CSCW) is focused on such collaborative working. Much research still needs to be done into the area of CSCW for clinical research. [21]

First, in order to aid the collaboration between researchers using computers we must know what type of information needs to be shared. Through advances in communication technology, images, textual and numeric data can be easily transmitted, however, sharing the understandings and knowledge about the data is a different matter. Also, an automated system for sharing databases of information still needs to be researched as knowledge of the variables and tables in remote databases is unknown.

Secondly, in order to support doctors with CSCW we must know the "context" in which the researchers are working and allow them to access information in a context sensitive manner. CSCW systems should also allow doctors to delegate work to members working together in order to speed up the completion of the work that needs to be done. In this context it should be noted that the medical domain of doctors and doctors in training have a strictly practical hierarchy. The responsibilities and protocols differ from organization to organization. Our present research does not address these distributed data sharing and collaborative working issues.

3.2 Scope of the Project

Because of our motivation for this research and background we decided to work with a real world problem, real people, and a live database. The location we chose to perform our research was the transplant clinic of a local hospital. The transplant clinic at this hospital is in charge of patients who have had transplants in the past and are now followed as out-patients. These transplant patients routinely come back for check-ups to ensure that their transplanted organs are functioning properly and also to diagnose and treat problems at an early stage. Transplant patients are assigned to designated nurse coordinators who are in charge of monitoring the status of these patients. These coordinators browse over the lab results and exams and mark problems that may require the attention of the doctor. Each coordinator deals with 100 to 150 patients in a typical center.

This transplant clinic relies on 2 databases to function. The first database is the hospital's general database that holds the patient's general data as well as where patient

labs (blood tests, etc) are sent to. The second database is the Transplant Database (TDB) which was specifically created for this clinic and provides the functionality to store and retrieve all the information gathered from the clinic. Even though the TDB runs separately from the hospital's database, some information from the hospital databases can be found within the TDB. The TDB is maintained by a small group of individuals, containing a single database expert and a few data entry personnel, however, data is entered by nurses such as clinic coordinators as well. The author of this thesis has been working as a part-time member with this group for 2 years. The scope of our project will be the support of the doctors in this clinic who treat the various transplant organs such as liver, kidney and pancreas. The doctors in this clinic routinely perform both clinical practice and clinical research in their daily activities though to a varying degree. Our goal is to examine the activities within this Transplant clinic, with respect to the practice and research, and to find where data visualization can support them effectively and to understand what type of support is needed.

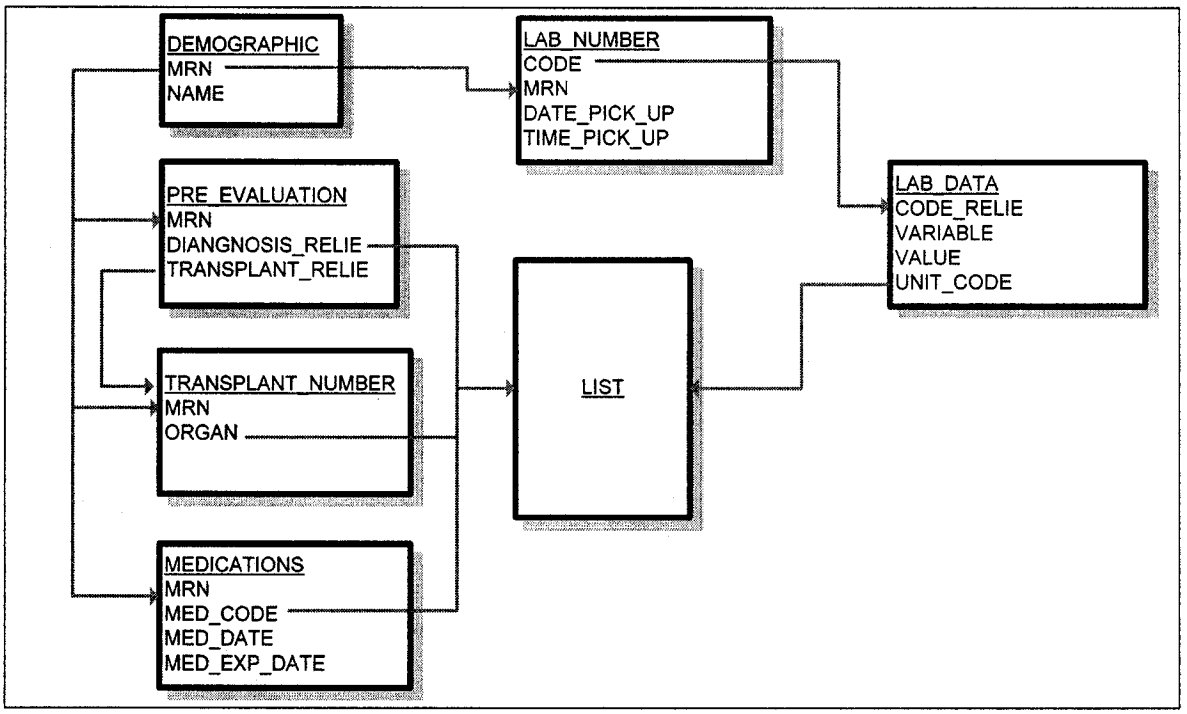


Figure 3.1-Abstracted Database Structure

3.3 Usage of the Transplant Database (TDB)

The Transplant Database (TDB) has evolved over time and the present schema is simply abstracted in Figure 3.1. Figure 3.1 shows only the main tables within the TDB and how they relate and interact with each other. Though the doctors know of the existence of the TDB and some of the information that it holds, their knowledge of how it functions or how to retrieve data and make the best use of it is quite limited. The TDB was not designed by an information system designer but by a domain expert who was knowledgeable on databases. The contents of the TDB can be broadly divided into:

(a) General Information on Patients with Unique Medical Record Numbers

(b) Transplant Details – Dates, types, and other specialized information

(c) Numeric Data – Blood work that has been done as well as vitals

(d) Current list of medications taken by the patient that is updated

(e) Text Data – Describing reports and consults

The numeric data is updated (mainly new data from the recent blood work) frequently within hours after the blood is drawn. This is done as a batch update a certain number of times in a day. The textual data update is done through manual data entry and thus the delays in data entry are much longer. Currently, the doctors and other stakeholders use the TDB mainly to view the details of the blood work as well as to keep track of and update the medications that have been prescribed to the patient. To access the TDB, a specialized software is used which is run on authorized computers found within the hospital. Password based control further ensures that privacy regulations are not violated.

As far as clinical research is concerned, when researchers require data, they visit the "nurse-expert" who knows this database, to discuss what variables are of interest to their research. This person is different from the transplant nurse coordinator. Once the nurse-expert gathers this data, researchers get the reports they need for their research. The turn-around delay in this process varies considerably depending on the workload of this nurse-expert and the time available. At times, many cycles are necessary as hypotheses are created or refined and more data needed. There also exist times when the data is not available within this TDB in which case the nurse-expert will create such fields to start collecting the needed data, thus the database design has evolved over time.

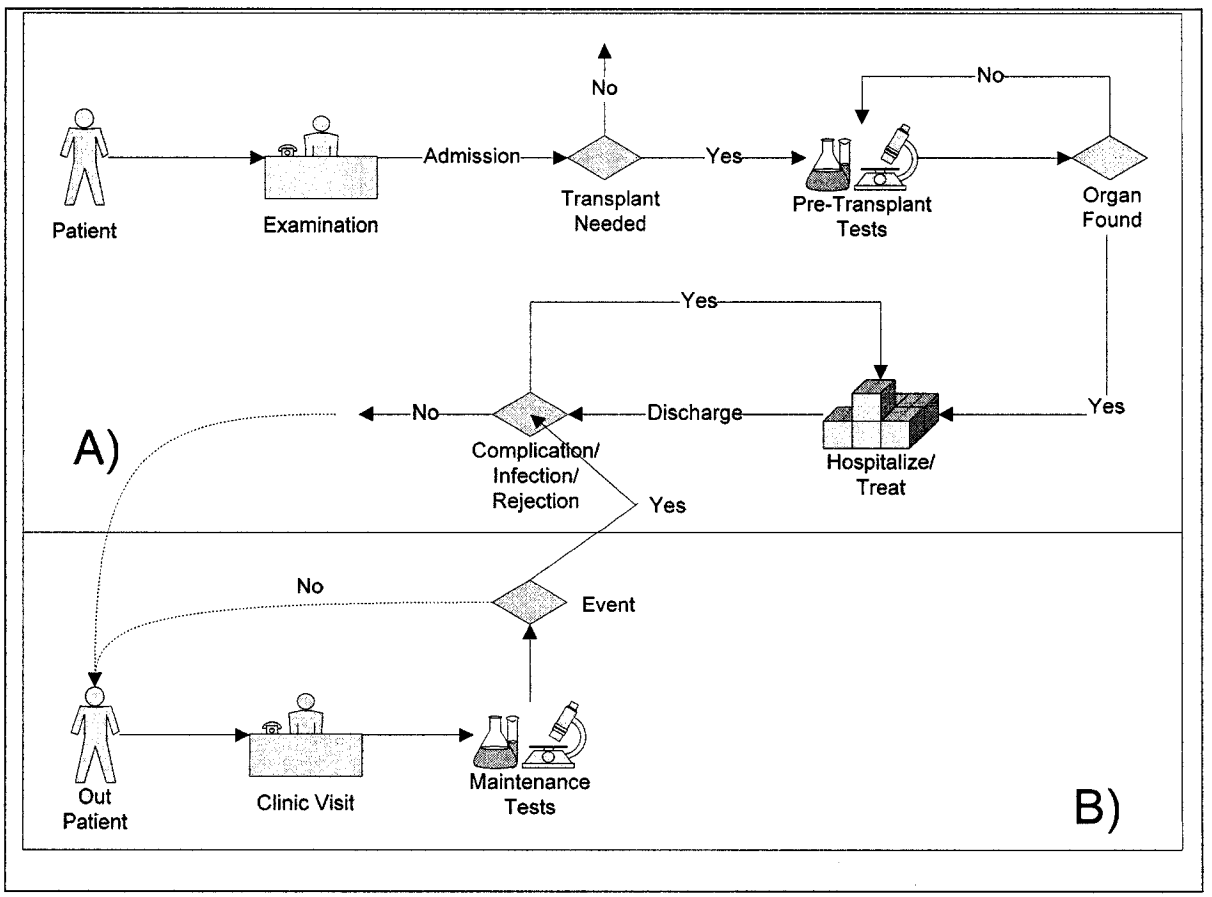


Figure 3.2-Transplant Patient Process

The TDB is also used for creating periodic reports. Most of these reports are pre-programmed and are used for discussions in various meetings among doctors for planning. Reports are also generated for patients to remind them of their medication and dosages. The treatment processes of transplant patients are shown in Figure 3.2; A) shows the in-patient process and B) shows the out-patient process.

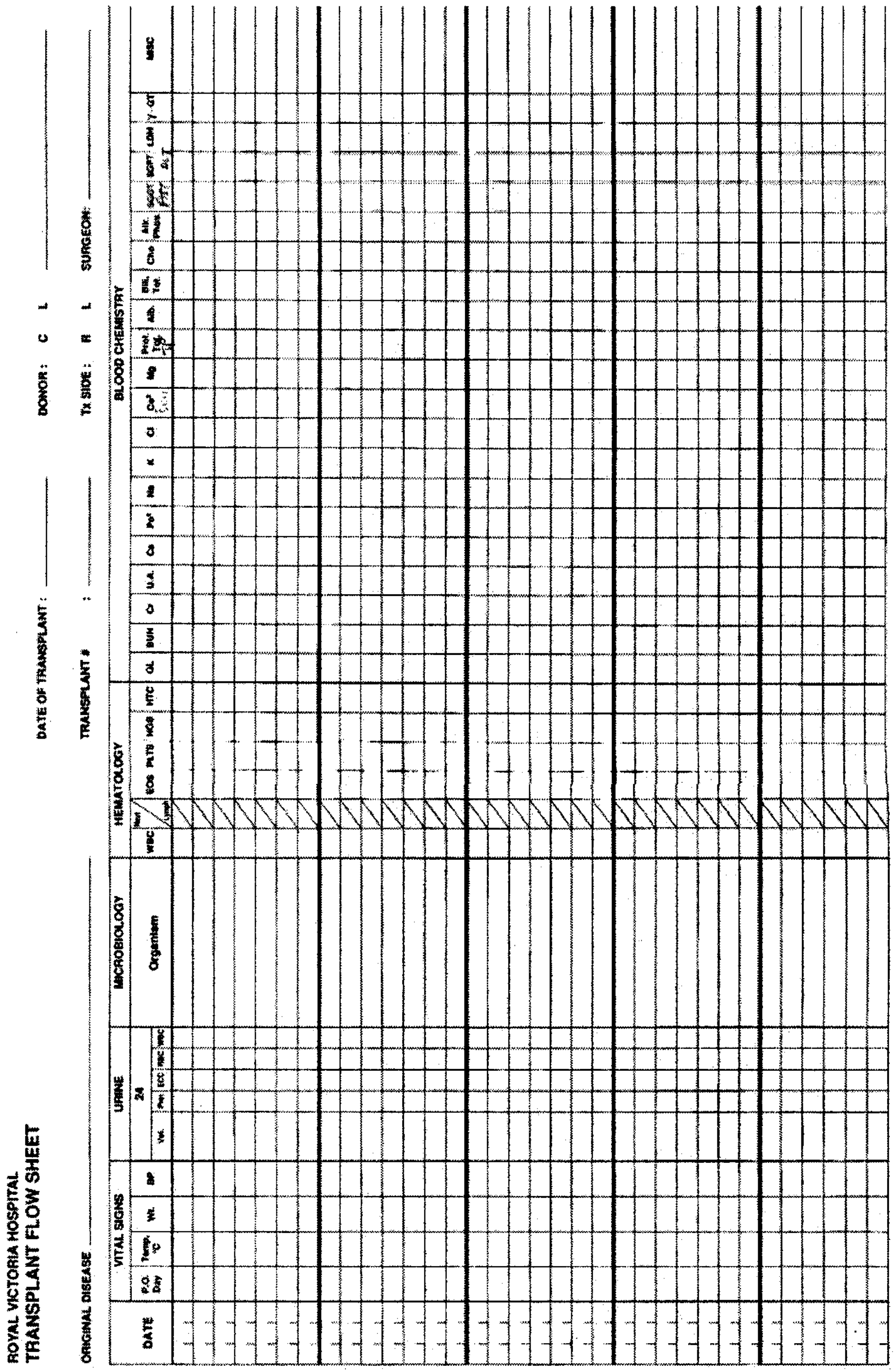
Patients are treated as in-patients in the hospital or as out-patients in the Transplant Clinic. As in-patients, they are treated in the ER or in the ward. The TDB is used in all these three points of care. In addition to the TDB, three other artifacts are used in the treatment process of transplant patients.

1) Hospital Database

2) Patient File (paper based)

3) Manual Flow Sheet (MFS)

The hospital database is more general than the TDB and it caters to the entire hospital. It contains patient registration data. Patient file is a paper-based file system that contains clinical notes written by doctors during patient visits, medication changes, test orders, reports, consults, and all other medical reports regarding the patient. It is also kept for legal purposes. Information (data) is thus scattered in different places. In order to make all the relevant data for treatment available at a glance, they have developed the MFS. Figure 3.3 shows a sample copy of an empty MFS. Each row in the MFS corresponds to one clinical visit for out-patient treatment, however, in-patients who have multiple blood-tests in a day will have a row for each test taken.



ROYAL VICTORIA HOSPITAL
TRANSPLANT FLOW SHEET

ORIGINAL DISEASE ______ DATE OF TRANSPLANT: ______ DONOR: C L ______

TRANSPLANT # ______ Tx SIDE: R L SURGEON: ______

DATE	VITAL SIGNS				URINE 24					MICROBIOLOGY	HEMATOLOGY						BLOOD CHEMISTRY																				
	P.O. Day	Temp. °C	Wt.	BP	Vol.	Prot	ECC	RBC	WBC	Organism	WBC	Neut / Lymph	EOS	PLTS	HGB	HTC	Gl.	BUN	Cr	U.A.	Ca	Po4	Na	K	Cl	Ca2+	Mg	Prot. Tot.	Alb.	Bil. Tot.	Cho	Alk. Phos.	SGOT	SGPT	LDH	γ-GT	MISC

Figure 3.3-Manual Flow Sheet (A)

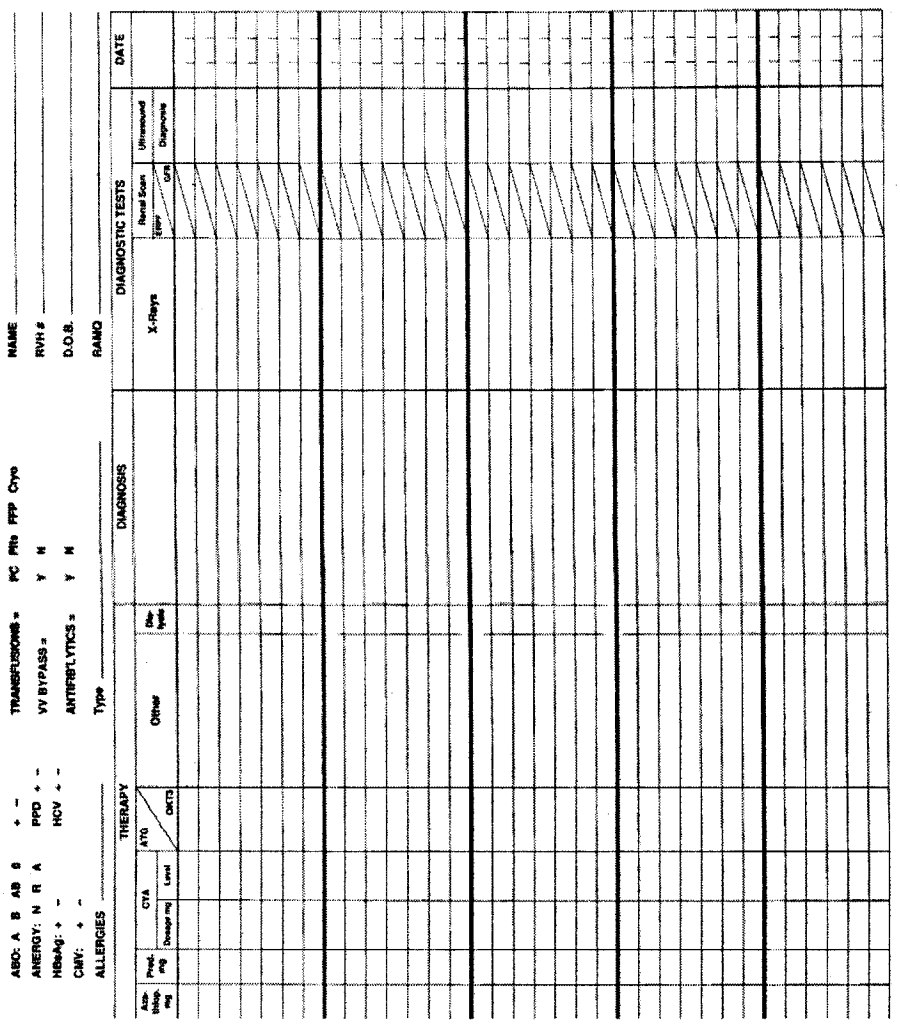


Figure 3.4-Manual Flow Sheet (B)

The TDB and the hospital database are also used for updating the Manual Flow Sheet (MFS). This flow sheet contains vital signs, urine analysis, haematology, microbiology, blood analysis, and other such variables about the status of the patient. Events such as "Stent put in" and text data are also recorded on the MFS. This flow sheet is updated each time the patient comes in for a check-up in the clinic and daily during in-patient treatment whenever results are available from the various labs in the hospital. Currently, the MFS is constantly and manually updated by the nurse coordinators in the transplant clinic or the unit coordinator in the ward by copying down the results from multiple sources. These MFS sheets are then given to the doctors who are in charge of the patient's treatments. From analyzing the data contained in the MFS the doctors and coordinating healthcare team follow treatment protocols or take proactive steps. The MFS, as a paper-based form, allows doctors to make additional notes in the margins to communicate and to record decisions and events that may be important for the treatment of that patient. The MFS is a "proven solution" as it has been used for many years and is practically stable. However, as a paper-based form it has a few downfalls, such as the copying of results from computers can introduce errors, inconsistencies, and delays. Another problem with paper-based forms is that it is only accessible from one location at a time unlike computer files accessible at multiple locations over a network that can have many access points.

3.4 Clinical Practice vs. Clinical Research Needs

The visualization requirements for clinical research and clinical practice are based on the doctor's goals. While clinical research aims towards gaining new knowledge about

treating patients or diseases, clinical practice aims towards applying the accumulated knowledge towards treating individual patients. In clinical research, knowledge is gained through understanding past experiences, existing protocols, time-series data of multiple groups of patients and formulating hypothesis to describe what has been observed or intuitively felt and then scientifically testing the hypothesis. If a new protocol offers a better outcome than a current protocol then clinical practice can be changed to what has been discovered through clinical research.

Functional Needs for Clinical Research

Clinical research could be an interactive as well as collaborative activity between researchers. Clinical researchers need to be able to intuitively and actively explore the data in order to find points of interest and patterns of significance in order to propose newer hypotheses; and they might like to collaborate with other researchers in validating their proposed new hypotheses and also to share this knowledge with other researchers. Clinical research focuses mainly on events of the past, or cycles of time in chronic diseases. Clinical research also focuses on a much larger set of patients grouped together on certain criteria. Because of this focus, functionality in the tool to support is to easily move through temporal data, dynamic clustering of large data and to abstract it into something meaningful. In order to perform these tasks we believe that researchers need some key functionality.

Visualization Need	Reason
1. A manner of visualizing raw data	This visualization need satisfies the need of research doctors to be able to get a general view of large amounts of numeric or textual data.
2. A manner of selectively viewing raw data that is of interest	Because of the large amounts of data available to clinical researchers, they need a way to remove visual clutter and focus on parts of the raw data that are of interest.
3. A manner to switch between different views of the data	As raw data in its numeric form is hard to analyze and explore, at times viewing it in a different manner such as graphs, charts, or timelines may provide additional insight. Also, switching between methods of visualization should be seamless and should not add additional burden to researchers.
4. A manner to view the data of multiple patients at the same time	Because clinical research deals with large sets of patients that satisfy certain criteria's they need a way to view large sets of data from multiple patients in order to see correlations between them.
5. A manner to supplement their findings with additional data	In clinical researcher more information is always beneficial. Labs can be supplemented with consults, exams, and medication lists to give a better view of a patient's condition.
6. A manner for building and executing queries that is intuitive and easy to understand	Clinical researchers need to be able to retrieve sets of patients that satisfy certain parameters that they have of interest. Because clinical researchers have a limited knowledge of querying languages and databases they need a way to build queries in an intuitive and easy fashion.
7. A manner to communicate their findings to others	As clinical research is a collaborative activity between many researchers and facilities, a manner to communicate results and findings is very beneficial.
8. A manner for drawing statistical inferences from the data	Researchers need to distinguish statistically significant trends from random variations in data.

Figure 3.5-Visualization Needs of Clinical Research

Functional Needs of Clinical Practice

Clinical practice in a hospital ward is also an interactive activity between doctors, nurses, and others. The visualization needs for clinical practitioners are more focused on the current status, decision, analysis and treatment of patients while the patients are in-front of the doctors. Thus the needs of clinical practice differ from that of clinical researchers particularly in timely response and contextual ease of access. This focus requires a more general look at the patient and their current state including all past relevant events that may help treat this patient's current illness. From analysis, we found that the functionality required of a software support system to be as shown in the Figure 3.5.

Visualization Needs	Reason
1. A manner of visualizing raw data	This visualization need satisfies the need of clinical doctors to be able to get a general view of large amounts of numeric or textual data.
2. A manner of selectively viewing data that is of interest	Depending on the specific condition of a patient there may be only certain variables that are of importance for determining treatment. By reducing the amount of unwanted visual clutter it makes it easier to analyze.
3. A manner of switching between different views of the data	Because of the dense nature of clinical data, it becomes important to have different views of the same data.
4. A manner to easy find supplemental information on the status of their patient (Exams, Consults, Labs)	During hospitalization patients go through a variety of tests, consults and labs that are necessary for treating that patient. Having access to supplemental information helps determine the best course of treatment.
5. A manner to analyze the current patients state	Tools must be given to analyze the data available.
6. Communication between the stakeholders involved in the treatment protocol (nurse coordinator, pharmacist, dietician)	

Figure 3.6-Visualization Needs of Clinical Practice

3.5 User Characteristics

Recall that people have characterized a stereo-typical medical Doctor as [5]: 1) intelligent; 2) computer naive; 3) little time to learn; 4) performs retrievals of data or information as needed; 5) antipathy towards computer; 6) no mental model of database; 7) has good knowledge of information contained in database; and 8) poor knowledge of database names.

These characteristics relate directly to how clinicians feel towards information systems designed to support them in their tasks and the attempts to change their current practices to use a new technology. System designers must take these facts into account if they want to ensure that their software is usable and acceptable to the end-users. These characteristics will be further elaborated below.

Clinicians are intelligent. They have a wealth of knowledge that is centered on medicine and can easily understand and grasp different concepts related to computers if presented properly to minimize the learning time.

Clinicians are computer naive. This characteristic refers to the unfamiliarity between doctors and computer terminology and computer functionality. Many times, doctors are not aware of the capabilities or limitations of the computer systems (software systems) at their site which frustrates and de-motivates them. Many doctors are also not aware of what benefits computers can offer them with a little investment of their already scarcely available time.

Clinicians are very busy, whether they are engaged in treating patients, performing procedures or conducting research. Therefore, the time available for them to learn new skills, especially technical skills needed with their computer software and

hardware, are limited. It has also been shown that instruction manuals and operation guides which use technical jargon increase their learning time; face-to-face training might be more suited in certain cases.

Clinicians perform retrievals of data and information as needed. This is true for clinical practitioners as well as researchers. Some clinicians rely purely on nurse coordinators to bring them the data of interest; therefore, do not need to perform the retrieval themselves. Another possible reason when the low number of data retrievals occurs is that the clinicians do not have the expertise or database knowledge to create queries and must then rely on the limited number of database experts at their disposal for this task.

Also, it has been seen that some older clinicians have some animosity towards computers and are hesitant to change to newer methods of data collection and support after decades of their practice. Unreliability or failures due to software bugs or hardware faults can discourage the doctors and result in unacceptability. The system design should not only minimize such faults but also have a fail-safe procedure for doctors to continue their clinical practice and recover from faults when the cause of failure is resolved. Because of these characteristic any computer support for both clinical researchers and clinical practitioners must be both easy to learn, intuitive to use, reliable and recoverable without data loss. The software created should match closely to the mental model that clinicians have of how the software should work to support their tasks that they are routinely doing. This will allow clinicians to feel more comfortable with the software and more willing to use it. This we have practically observed.

Clinicians also have no model of the database. They may know what the database holds but do not have any idea of how the database tables or schemas and variables are named and related. Because the TDB is tailored for the use of the transplant clinic and its doctors, in our context, the doctors have prior experience in the use of the software associated with the TDB for data access but not the TDB itself.

3.6 Requirements Document

In the work of this thesis we have proposed two components; (a) EFS or Electronic Flow Sheet which uses the metaphor of the MFS with added functionalities and structural additions; (b) QMP or "Query Module Processor" to support interactive and flexible querying of the targeted database to support clinical research.

In this section we will outline the requirements for both the EFS and the QMP. These requirements were derived through meetings with selected doctors and observations of the tasks performed by clinical practitioners and clinical researchers. These requirements were adjusted, though only slightly, throughout the design process as new ideas arose through meetings and as demonstrations of our prototype.

3.6.1 Constraints

Because the EFS and the QMP must work on pre-existing computers we are constrained by the current environment of the transplant clinic as well as the databases supporting this clinic. One of the constraints placed on our prototypes was that, though we could perform any number of retrievals from the databases in place we could not enter or modify this data. This was to protect the integrity of the database and to ensure that

there were no errors caused by our prototypes. If the EFS or QMP needed to store additional information we would be required to create our own database for use.

3.6.2 Objectives

Electronic Flow Sheet

The main objective of the EFS is to support the activities of both clinical practitioners and clinical researchers during their everyday tasks. This includes checking up on current patients' labs as well as viewing lab information for potential research subjects. At minimum, the EFS should perform at least as well as the current MFS by providing a viewing method of laboratory results of blood-work, hematology and other labs. Another objective of the EFS is to reduce the amount of work currently on the transplant coordinators who are charged with updating the MFS by manually copying the lab results from the database. Beyond supporting these tasks, we also wish to provide the benefits of computerization such as efficient searching, different visualization methods and information storage using databases.

Query Module Prototype

The objective of the QMP is to provide an alternate manner for clinical researchers to retrieve and explore information from the databases easily without relying solely on the database expert. The QMP could also be used by researchers to perform preliminary retrievals to see if new ideas arise before going to the database expert for more complicated queries. Figure 3.6 A) shows the current method used by researchers to obtain information and B) shows the preferred method for the same retrieval. Currently, researchers are limited by the amount of time available from the database expert. As

more researchers require information from this person the amount of time available for each decreases and the turnaround delay increases.

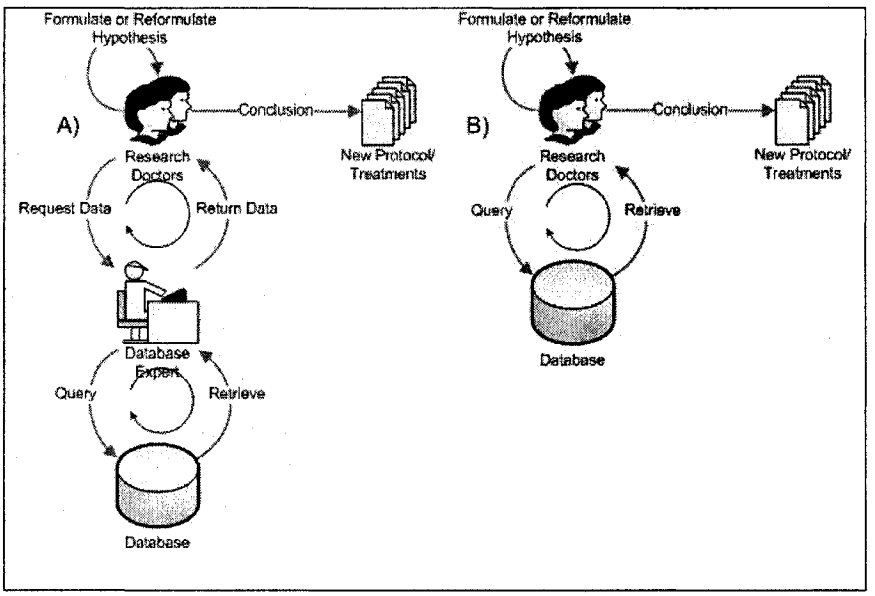


Figure 3.7- Iterative Exploration in Clinical research

3.6.3 Functional Requirements

Electronic Flow Sheet

1. The EFS should be able to only display items that are needed by the user. The EFS should allow for removal of columns from the lab tables in order to remove items that are of no interest to the current user. This will help reduce the amount of visual clutter and allows for the user to focus on only what is necessary for them to perform their task.
2. Searching - The EFS should allow users to retrieve data from dates that are not currently visible as well as search for Patients using their unique identifiers.

3. Notes – The EFS should also provide the user with a manner to write down their observations so that it is visible and available for other users. These notes should be storable.
4. Additional information – The EFS should make external information available that can aid the user in decision making or help in understanding the current status of the patient.

Query Module Prototype

1. Retrieval – The QMP should be able to run user created queries on the database and be able to display them.
2. Swapping – The QMP should be ale to swap to the EFS and back seamlessly and easily.

3.6.4 Non-Functional Requirements

1. Time to Learn – The time to learn the EFS should be very minimal and intuitive to use. As the EFS resemble the MFS we believe the learning time should be reduced. However, the learning time for the QMP could be relatively more as it may be unfamiliar.
3. Easy to Use – Any new features in the EFS should be a natural extension so that retention of how to use them is not difficult. Simple things should be simple to do.
4. Initial Start – Since there is inertia when using a new tool; a brief (10 minute) demonstration and introduction to the EFS/QMP must be devised. This time

can also be used to motivate them to explore the possibilities and features available to the users.

5. GUI Based – The UI should be graphical with a pleasant appearance.
6. Quality Parameters – Data privacy, security and flexibility of the QMP are to be assured. Use of real data will motivate the end-users even during the prototype testing stages. Response time in clinical practice applications is very crucial as well.

4 System Design and Prototyping

In this chapter we will discuss the design and development of our proposed solution to the problem of interactive data visualization within the field of clinical practice and clinical research. The software solution, composed of two components namely the Electronic Flow Sheet (EFS) and Query Module Processor (QMP), is meant to work as a single unit to provide an interactive visualization environment for doctors performing both clinical research and clinical practice. In order to support interactive data visualization within this domain we first needed to understand the current practice of these two activities and find where visualization could provide the most benefit and what kind of interactive visualization requirements are present. Along with understanding clinical practice and clinical research tasks we needed to explore and comprehend the end-user characteristics and limitations as well as the data needs to fulfill those needs such as databases, tables and querying facilities. Our analysis of clinical practice, clinical research and data were summarized in Chapter 1 and our analysis of requirements has been presented in Chapter 3. The details of our design and development are discussed within this chapter.

4.1 Overview

The design and development of the EFS and QMP spanned 12-15 months and involved several interactions with stakeholders and end-users. A closer interaction with the end-users (medical doctors) was made possible as I was employed at the hospital to work three days a week to help merge transplant databases and then eventually maintain

this database. The development of our prototype also involved many months of working at the Transplant Clinic. The experience gained from this work greatly added to our understanding of the clinical research and clinical practice tasks as well as the technology and data that supported these activities. In turn, the knowledge gained led to many modifications and additional features that are incorporated into EFS and QMP that we believe would benefit the activities we wished to support.

4.2 Process employed

The software methodology we used in the creation of our prototype was a mixture of the agile software development method and the Personal Software Process (PSP). Agile software development, particularly eXtreme Programming (XP), is a rapid development framework that focuses on two key aspects, namely, quick iterations, and many face-to-face meetings between the developer and the end users with useable prototypes at the end of each iteration. The Personal Software Process, on the other hand, is more concerned with the improvement of the quality and productivity of software developers by improving planning and analysis when it is developed by an individual. Each of these software processes will be discussed briefly in the section bellow, extensive details can be found in the open literature [22,23]

4.2.1 Agile Software

Agile software development [22] is a rather new development framework which was developed in the 1990's to deal with rapidly changing requirements in certain projects that made the traditional waterfall model hard to use. In 2001, The Agile

Manifesto was published outlining the principles of the Agile Method. These principles are listed below in Table 4.1.

Table 4.1-Agile Software Principles

1. Customer satisfaction by rapid, continuous delivery of useful software
2. Working software is delivered frequently (weeks rather than months)
3. Working software is the principal measure of progress
4. Even late changes in requirements are welcomed
5. Close, daily, cooperation between business people and developers
6. Face-to-face conversation is the best form of communication
7. Projects are built around motivated individuals, who should be trusted
8. Continuous attention to technical excellence and good design
9. Simplicity
10. Self-organizing teams
11. Regular adaptation to changing circumstances

Extreme Programming (XP) falls under the category of Agile Software Development Methods [19]. XP focuses on 5 values during the creation of software namely: communication, simplicity, feedback, courage, and respect. Communication is important, both between team members as well as customers. Software requirements need to be communicated clearly from the customers to the team and development knowledge must be shared between team members as well. Because of the importance of

communication, simple designs, shared metaphors and collaboration between users and programmers is highly valued.

Simplicity is also important in XP as a functional prototype early on is needed. Additional functionality and other features can be added later on in the development cycle. Simple design and implementation means that prototyping is fast and efficient and little time is wasted on uncertain features and unneeded code.

Feedback between team members as well as customers is another requirement for XP. Feedback, communication and simplicity are all tied together within the development cycle. Feedback refers to customers testing developed prototypes and communicating changes and new requirements back to the team. Once the team obtains these changes they must give feedback to the customer about projected costs and time of the new requirements.

Courage means that programmers must be willing to refactor the code and make adjustments to current code when needed. Courage also means that programmers are willing to remove and replace code which is no longer needed in order to improve efficiency and make additional functionalities possible. Courage also refers to the persistence of programmers to solve a problem and to continue working on solutions.

Finally, programmers must respect the code of others in the team and should not hamper the testing or work of their peers.

We chose to follow an agile method of software development because of our close working relationship with our end-users and stakeholders that gave us a unique understanding of the needs and work of clinical researchers and clinical practitioners. This relationship also allowed for frequent meetings and quick problem solving and

prototype testing as different iterations were completed. Another reason for choosing this methodology is based on the characteristics of the doctors as end-users of the intended software product and our desire to make the product useful and usable.

4.2.1.1 Iteration Design Framework

The iteration design framework outlines the general process that each iteration took shown in Figure 4.1. The early iterations were used to develop a useful UI that was acceptable and useable by the end-users while the later iterations were used to develop the functionalities of both the EFS and the QMP. The iterations of the EFS and the QMP overlap in a way that the later iterations of the EFS were also used to begin the development of the QMP user interface.

Step 1: Information Gathering

This step includes finding the materials and forms the clinical researchers and clinical practitioners use to perform their daily tasks, and face-to-face meetings to discuss how these tasks are done. In the early iterations of information gathering the traditional forms and artifacts were used to prototype the UI while in the later stages it is used to discover what missing functionalities the prototypes should support in order to support their daily tasks better. This stage brought to our attention the missing elements in the current Manual Flow Sheet as well as the Transplant Database and Hospital Database that was in use. This stage also introduced us to more researchers and practitioners who work within the transplant clinic and thereby expanding our input services for requirements gathering.

Step 2: Build Prototype

Once information is gathered, development of a prototype began. This stage involves the design of the UI in the early iterations as well as the design of the back-end functionality in later stages. The early prototypes that were developed in this stage contained only the User Interface such that once we had the appropriate interface we could start developing the functionalities as well as database communications. This stage was facilitated by the MFS when developing the EFS. The early prototype was presented to a team of doctors and medical researchers in 2006 December and feedback were obtained.

Step 3: Prototype Evaluation

The Prototype is demonstrated to a few select end-users with the goal of seeing if it is acceptable as a tool to support their tasks. Changes are explained as well as the rational behind those changes. During these evaluations many improvements, suggestions as well as new and novel ideas were made for the succeeding iterations. Some of these new features include:

1. The ability to add free-text notes to the EFS so that doctors can communicate observations and diagnoses.
2. Flagging of notes that would eventually allow doctors to notify each other of important updates of a patient's condition.
3. Batch loading of doctors patients so that searches and retrievals would be faster.
4. Addition of a column for Transplant Medication Dosages as well as the dosage of the medication within the patient's blood.

Through these evaluations we could tailor the prototype and arrive at the final prototype for testing and evaluation.

The iterations brought us closer to the final versions of the EFS and the QMP. As each iteration was completed the prototype was demonstrated to the available end-users for constant feed back.

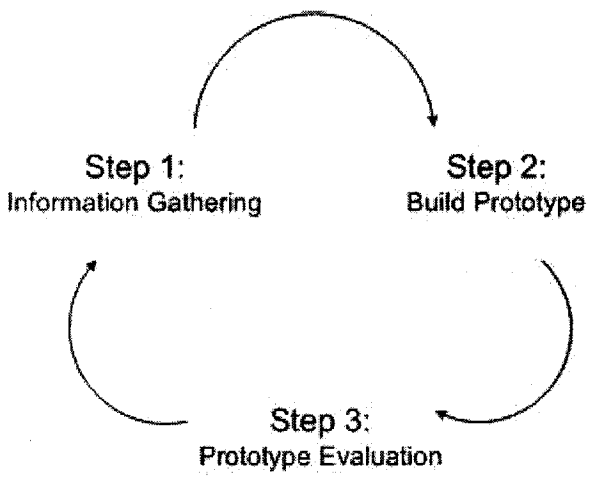


Figure 4.1-Iteration Design Cycle

4.2.2 Personal Software Process (PSP)

The Personal Software Process is largely based on thorough planning, review and measurement of each phase of development with the goal of improving software engineer's personal performance [24, 25]. The PSP process shows software engineers how to:

- manage the quality of their projects
- make commitments they can meet
- improve estimating and planning
- reduce defects in their products

By doing thorough analysis of projects, software engineers must first estimate how much total time is needed for the given project, and then decide how to allocate their time accordingly. Using logs to track project plans also help software engineers to stay on course and to cut down on unnecessary wastes of time. Tracking size and estimating size is also important when planning projects. Other logs are used to track defects in different phases of a project and how to fix them. The term Defects refers to portions of the software that do not function as they are supposed to and can occur during any phase of development such as planning, coding, testing and even the requirements stage. Finding defects early reduce the cost of fixing them later on, this is the reason code and design reviews are needed rather then waiting for end-users to discover defects. Defects have a direct impact on the overall quality of the software as some defects will make software completely unusable or unacceptable to end users.

Table 4.2- Defect Type Standard

Type Number	Type Name	Description
10	Documentation	Comments, messages
20	Syntax	Spelling, punctuation, typos, instruction formats
30	Build, Package	Change management, library, version control
40	Assignment	Declaration, duplicate names, scope, limits
50	Interface	Procedure calls and references, I/O, user formats
60	Checking	Error messages, inadequate checks
70	Data	Structure, content
80	Function	Logic, pointers, loops, recursion, computation, function defects
90	System	Configuration, timing, memory
100	Environment	Design, compile, test, or other support system problems

Within this thesis aspects of the PSP method were incorporated during the planning and coding portions of development of our prototype. By collecting requirements from our users we created a detailed model of how our software should work and what objects, modules, and functionality would be needed. After the project was completed, we went back and estimated how much time we spent on each portion of

our development to see where time was spent unnecessarily and where improvements could be made in future iterations. Our documentation, time logs and defect logs can be found in Appendix E. We have divided our documentation and development into 4 parts, namely GUI, Database's and related functions, Objects and related manipulators and constructors, and finally functionality.

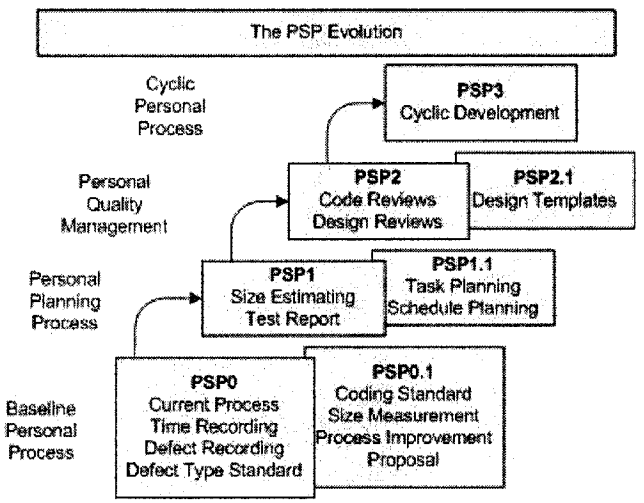


Figure 4.2-PSP Model [21]

We can see from our *Time Log* that there are large disparities in time taken between the EFS and the QMP. This can be explained by the differing characteristics of each module. The EFS used a form as a metaphor, therefore, most of the planning and design for the EFS GUI was very short, whereas the design of the QMP needed to be planned and designed on its own. Most of these differences are discussed within the Appendix.

When analyzing our defects, we must first acknowledge the use of the Borland JBuilder development environment which had an impact on the number of defects in our syntax, building, and assignments. This is due to the nature of the development environment which notifies its users of syntax errors dynamically as they are made. Some

defects in design planning that we recognize now are requirements of doctors that they be able to search patients by name, however, during development we only allowed them to search by patients unique identifiers.

The PSP requires time to learn and use as well as commitment on the part of the developer to continue using the process. However, these costs are offset by the potential benefits that result from better coding, planning, and maintenance of code.

4.3 System Design

The EFS and QMP were both developed in Java and were designed to work with an Oracle or mySQL database. Java was chosen because as an object-oriented language it was easier to create modular code which would be easier to maintain and reuse. The EFS and QMP were each developed separately but were then later merged through a controller module that would later be used to pass information between the modules. Both the EFS and QMP rely on the same database of information; however, one is used to display this information whereas the other is used to retrieve such data. The transplant database is already in existence and our constraint is that it can not be modified; our systems would have to be created around the current tables and database relationships. In order to overcome the obstacles of working with large amounts of data, we create objects that we can use to group meaningful pieces of data together. These objects allow us to create abstractions that are easy to use. Some useful abstractions that we created were Labs, Queries and Medications. Another abstraction we created was the Patient, the EFS and QMP makes use of a Patient object, which is a class used to represent the variables and attributes of a single transplant patient. Once filled with data, this Patient object contains

all assessors and data pertinent to a single patient. These objects remain in memory so that they can be recalled easily for display or processing purposes and this approach reduces the number of database accesses needed. Labs belonging to patients are also stored inside these patient objects. Challenges related to the development of our prototype are listed below.

4.3.1 Design Challenges

4.3.1.1 Efficiency

One of the design challenges we faced while creating the EFS was "efficiency" when loading the large number of lists that accompany the database. Minimizing this delay is very essential for providing timely response to everyday end-user commands or queries. We refer to "efficiency" as the time it takes for the EFS to retrieve and load these lists or to retrieve all the different lab results for a patient. Initially, the EFS would pull up lists sequentially which proved to be slow. In order to reduce the amount of typing entered by the database users, list tables were added to the database so that entries could be selected from lists. By providing lists for variables such as procedures, medications, diseases, etc... we standardized what can be added into certain columns and reduce spelling errors and variations. This standardization makes searching and data mining easier for researchers. Because these lists are massive and can contain thousands of entries, retrieving them sequentially takes a large amount of time. In order to overcome this, we implemented simple threads that would independently query the database for all the lists. Each thread would be given the appropriate SQL statement to execute and then load the lists into memory. The same sort of threading was added to the process of

retrieving labs from the database seen as pseudo-code in Table 1. Each thread would be responsible for retrieving a single category of labs such as vital signs, blood, hematology, etc... after retrieving these labs; all the threads must be synchronized together in order to decompose them and order them for display. By adding threads, we found we saved on average 2 seconds for each patient depending on how many lab entries a patient has. Timing calculations were done informally by running queries on a single patient 10 times with threading and then 10 times without and taking an average of the difference.

Table 4.3-Sample Pseudo Code for Lab Retrieval

Sequential Lab Retrievals	Concurrent Lab Retrievals

...
Vector Hemodialysis
Vector Blood
Vector Urine
Vector Vitals
 	
Hemodialysis=get_Labs(SQL_String("Hemo"));
Blood=get_Labs(SQL_String("Blood"));
Urine=get_Labs(SQL_String("Urine"));
Vitals=get_Labs(SQL_String("Vitals"));
...
CreateTable(Hemodialysis);
CreateTable(Blood);
CreateTable(Urine);
CreateTable(Vitals);
	...
	Vector Hemodialysis
	Vector Blood
	Vector Urine
	Vector Vitals
	
	thread hemo = new thread(SQL_String("Hemo"));
	hemo.run();
	thread blood = new thread(SQL_String("Blood"));
	hemo.run();
	thread urine = new thread(SQL_String("Urine"));
	hemo.run();
	thread vitals = new thread(SQL_String("Vitals"));
	hemo.run();
	
	While (threads not complete){
	wait(5);
	}
	
	CreateTable(Hemodialysis);
	CreateTable(Blood);
	CreateTable(Urine);
	CreateTable(Vitals);

Another "efficiency" issue was how to make the EFS efficient when switching through many patients. Switching patients requires the EFS to constantly go and pull up patient records from the database in order to display them, which takes up a lot of time. One of the characteristics of clinical practice is that practitioners need to switch from one patient to another quickly and efficiently when they are reviewing labs and results. In a

real life situation, doctors will have stacks of patient records and manually flip through each one as new labs and data are added about that patient. A functionality that arose from these needs was a method of queuing records into the EFS. As the patients arrive at the doctor's office for visits, their relevant data are pre-loaded into the EFS's memory so that they are available by simply selecting their name from the queue.

4.3.1.2 Query Building

The biggest challenge from the development of the QMP was how to model the mental process of the medical researcher in their incremental query formulation and build a relevant query dynamically to match it. The goal of the QMP is to allow doctors to retrieve information by helping to build SQL queries using a graphical user interface. In order to build these queries we had to find a way to capture the variables involved and the relevant constraints which the user wishes to apply on the database. This can be done in several ways, one way is to allow the user to use the graphical interface to select constraints and variables and then parse this data and transform it into an SQL string automatically. However, we find that this would limit the flexibility needed when the user adds or removes constraints in an incremental manner. We then chose to break up the portions of the SQL queries into different parts representing the variables that are requested, the tables these variables come from and finally the constraints that the researcher wishes to place on these variables. We decided that the easiest way to approach this problem was by using objects to help model the variables. By using objects to model the variables, we could dynamically add and remove them as a list, as the requirements of and specification by the user changes. We chose to model constraints also as objects. As constraints can be divided into several categories such as numeric

constraints, textual constraints and date constraints with a common parent, we felt that using objects to represent these would make them easier to handle and to assign to variable objects as they need to be constrained. This way of modeling the constraints provides flexibility to the users to remove or to add constraints as they please. This method also makes it easier to break down the list of constraints later on when creating the SQL statement as needed. Once the user has selected all the variables and constraints they are ready to query the database. The QMP takes the list of variable objects and extracts the variables and constraints and parses them into the SQL query. Figure 4.3 shows a simplified view of this process. The resulting SQL query is then run on the database and the results are parsed into tables for interactive viewing.

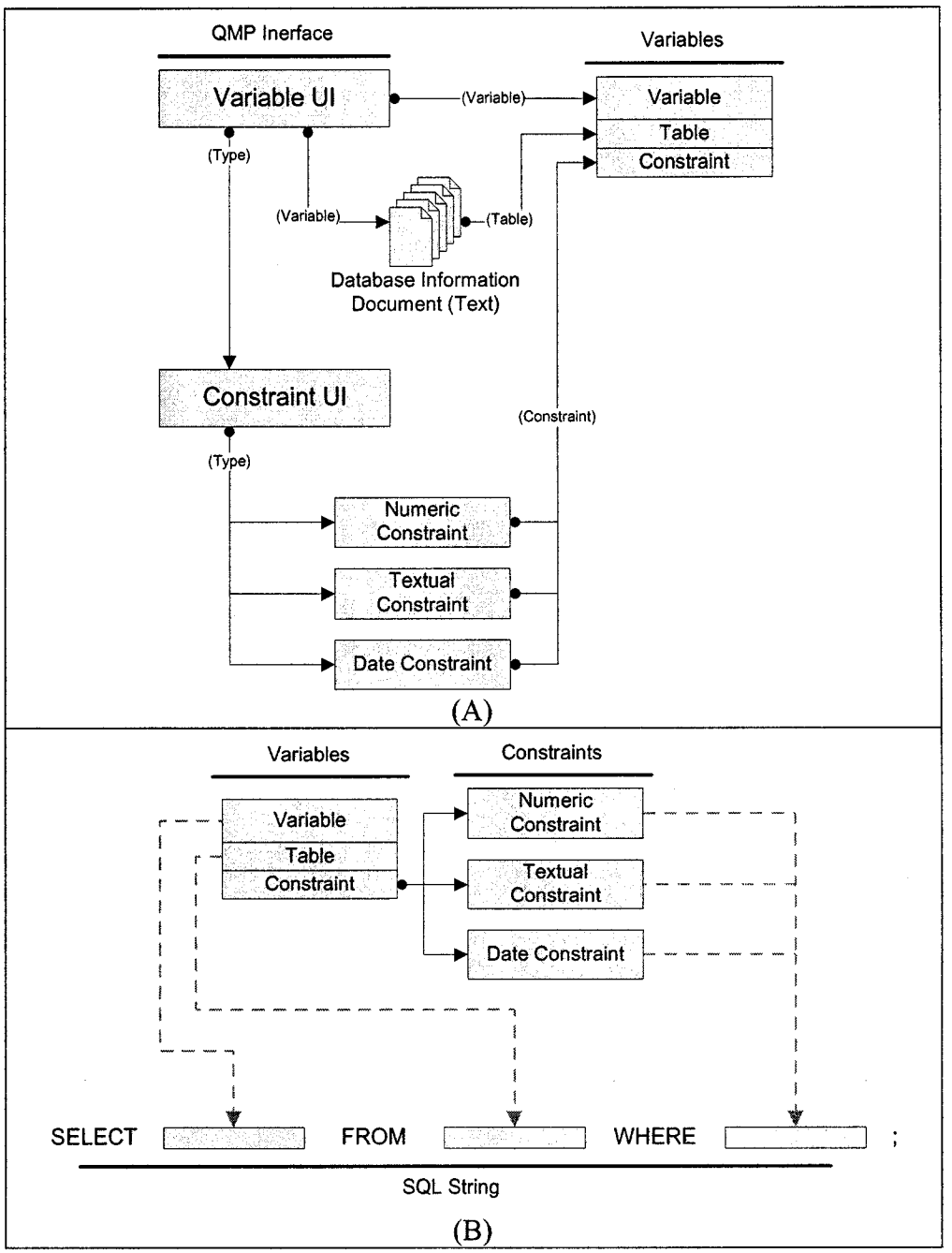


Figure 4.3-a) Creating QMP Variable Objects b) Converting QMP Variable Objects into SQL Strings

4.3.1.3 Constraint Grouping

Another challenge to the creation of the QMP was that the large number of possible variables made the creation of separate sub-panels to display each variable infeasible. In order to reduce the amount of redundancy we decided to create generic sub-

panels that could be easily configured to whatever variable and constraint are selected. By specifying the variable and the type of constraint that can be logically placed on the variable the sub-panels were created.

4.3.1.4 Database Characteristics

Because we were working with a live database that was in constant use in the hospital our ability to modify and add tables and information was restricted. The danger of accidentally changing important data existed so we decided to use only retrieve SQL operations on the live database. However, in order to apply the functionality of note taking and diagnosis recording we needed a method of storing this information. For this purpose, we created a temporary local database in Microsoft Access to record this information. This maintained the integrity of the live database but required extra code for implementation.

4.3.2 User Interface Design Challenges

4.3.2.1 Electronic Flow Sheet

The challenge with the EFS was to create an interface that would be easy to navigate and to use. Our solution to this was to use the current MFS as a metaphor for our EFS so that clinical practitioners and researchers who are familiar with the MFS would feel familiar with the EFS. Though the EFS looks and feels a lot like the MFS there have been modifications that reflect what we believe are more beneficial and acceptable. Our observations and meetings led to the use of the MFS as a metaphor for the creation of our Electronic Flow Sheet (EFS). Our use of the MFS as a metaphor for our prototype allowed our users to see the EFS as familiar and reduced anxiety over the unknown. The

translation from the MFS to the EFS required us to make certain modifications. The MFS form was printed on longer than normal paper, to accommodate the large amount of lab data, which would not fit on a computer monitor. To overcome this we added scrollbars within the prototype so that users would be able to view the entire EFS. The creation of the EFS also allowed us to remove the parts of the MFS that were un-necessary and which caused visual clutter for users such as certain tables on the MFS. Along with removing features of the MFS from the EFS we also added some key features that arose through our meetings and observations. Once these additions were made, meetings were held so that we could decide whether these modifications were better than the existing MFS. In general, the overall look and feel of the MFS remains in the EFS. The top and bottom of the MFS sheet remain unchanged in the EFS as it provides general information on the patient such as disease, transplant organ, etc. The addition UI options we added to the EFS are listed below; some were incorporated into the EFS lab tables while others were added as menu options.

4.3.2.2 Filtering

Another interface challenge was how to reduce the amount of visual clutter that was distracting to physicians. Because we felt that the MFS, though complete in its display of lab data, provided too much visual clutter we added the functionality of filtering. From our meetings with clinical researchers and clinical practitioners, we found that researchers and practitioners focusing on different organs looked at different variables on the MFS sheet. As the MFS was meant for general use by all clinicians and researchers, it held all variables available from the labs; however, specific clinical practitioners dealing with specific organs need to focus on only certain variables. This

narrowed focus meant that all unneeded variables became visual clutter for clinicians and thus led to more mental stress. The filtering functionality we added in the EFS meant that by selecting specific variables to filter from the lab tables, all other unneeded variables would be hidden. Within the EFS this can be done actively for each patient, or pro-actively by the EFS from a pre-stored preference. We believe that by introducing this type of filtering can speed up the analysis process of both clinical practitioners and clinical researchers.

4.3.2.3 Graphing

As we wanted to provide visualization support for clinical practice and clinical research we needed to find an appropriate way to add it to the EFS. Because the MFS is used to display mostly numeric and textual data we felt that adding a graphical visualization method would be beneficial. This graphical visualization method would allow researchers as well as practitioners a new way to analyze and view the data on the EFS. There exists situations where knowing the direction in which the variables are trending is as important as knowing the actual values of the variables. Though it is possible to see trending of variables by looking at the numbers, graphical representation is usually faster and can give a larger range of values. Many variables can be graphed together using this functionality. Within the EFS, when graphing, the first graph will show all the selected variables together. This view helps researchers and practitioners see correlations within the data that may be important in determining the course of the disease or treatment. Along with seeing all the variables graphed together, the EFS will also display individual graphs so that they can be analyzed without the clutter of other variables.

4.3.2.4 Supplemental Data

With the MFS, when clinicians need additional data such as consults, exams and medications they must either go through the patients file or consult the hospitals database. This can cause problems at times because this would require doctors to access 2 or 3 different data sources (files, papers, databases) that may not be located within the same area. Within the EFS we have provided the functionality to check a patient's current medication as well as all the patients past exams and consults. We believe that by providing this functionality we can provide clinicians with the tools necessary for them to do their work efficiently and effectively. This supplemental data is available to users from a menu at the top of the EFS. This facility was not possible with MFS.

Other supplemental data that we have made available through the user's interface with the lab tables is 'notes'. Doctors write these notes for several reasons, such as consults, observations or explanations. These notes help to communicate to other doctors and caregivers the status of the patient and helps keep an accurate medical record for the patient.

4.3.2.5 Query Module Prototype

Because the hospital did not have a current system that allowed for doctors to query the databases for data we had to develop our own user interface and see if it was acceptable. Using other querying software as a guide for general UI layout, we proceeded to select variables that we felt would be the most useful to doctors wishing to query for information. These variables were taken from sample queries that were supplied to us by the database experts and represented real queries needed by doctors. Once we selected the variables we divided them into 4 categories, Unique Identifiers, Demographic Filters,

Specific Filters, and Retrieval Filters. We allow users to put constraints on these variables by selecting them and filling in the appropriate information. In most cases, constraints will fall under Numeric Constraints, Textual Constraints and Date Constraints. These constraints are then collected into a table so that they are visible to the user and they can make changes as needed. Once all the constraints are done, the results are displayed in the same manner as the EFS.

4.4 Summary

We have experienced in the work of this thesis as well as in two previous Master's theses [6, 26] in our research team that a combination of careful requirements engineering, practicing a user-centered design and applying a suitable software method for iterative design and shorter iteration cycle, the probability of success is much higher for developing a health care application that can be "acceptable" for the healthcare community. In this thesis, we have used an agile method of development, along with a personal software process and a close working relationship with our end-user to easily employ an iterative design process. The close working relationship that we have established by participation in their environment also allowed us to quickly prototype, discover and add new functionalities as they were needed. Through iterative prototyping, we allowed the end-user to take part in the design process which also leads to higher acceptability and software that is tailored to their needs. The design challenges we faced are directly related to the characteristics of our end-users, their tasks and environmental characteristics. Because medical doctors are constantly busy, a UI that is intuitive, requiring little time to learn, easily recognizable and efficient is necessary. A UI must

conform to their daily needs and activities and must provide functionality that supports them. Groupings of medical categories must also fit the mental model of doctors of how they perceive the data.

5 Evaluation and Testing

The reason for Evaluating and Testing any prototypes is to prove that it does, indeed, satisfy the tasks and needs that the prototype was meant to support. With respect to the EFS and the QMP our goals were to see how Medical Data Visualization can be incorporated into the daily activities of clinical researchers and clinical practitioners such that it improves the way they do their work. In this section, we will discuss how we performed our evaluation, our results, analysis of our results as well as a brief discussion on some of the difficulties we faced while trying to evaluate and test our prototype within a hospital setting

5.1 Plans for Evaluation

The pilot and evaluation of the EFS and QMP was conducted over the course of several weeks at the Transplant Clinic of a large university teaching hospital, during which we invited 9 doctors to evaluate our prototype who were all familiar with the MFS. The computers on which we ran our prototypes were the computers that were already being used within this large university teaching hospital. Locations of these evaluation sessions ranged from doctors' offices, transplant clinic computers as well as ward computers. This was to show that our prototype would work on any computer in the hospital that was currently connected to the hospital's databases. Though the EFS and QMP prototype could fully access all the patients in the hospital database, we chose to limit the demonstration to a set of 10 transplant patients and their related data. For each of these patients we have manually entered certain data pertaining to their transplant that

are currently kept in manual files. Doctors were also encouraged to browse their own patients so that it could mimic their daily work. Along with the doctors we also enlisted the help a few transplant nurse coordinators working at the transplant clinic to help in our refinement of the EFS and QMP. The refinement of the EFS and QMP took an additional 2 weeks to complete and led to some added functionalities such as exporting to Microsoft Excel as well as batch MRN searching that facilitated switching between patients. During the 2 weeks of refinement we created a simple User Manual, shown in the Appendix, that gives users a quick overview of the functionalities of our prototype. This user manual, coupled with incremental demonstrations at each stage of development increased the ease of learning for doctors.

5.2 Conducting the Evaluation

Because of the differing schedules of the doctors we worked around their varying schedules, which is the reason why our evaluation period took months. Every one was given a copy of the user manual before hand, but doctors hardly anyone had time to go through it prior to the evaluation session. During each meeting with the evaluating doctor for the first tem minutes, we demonstrated the EFS and QMP before explaining the questionnaire and evaluation form and its purpose. The user manual, prototype and the evaluation were all placed on the shared network drive for their ready access. This allowed for the doctors to go back to their routines and try the EFS and QMP whenever they found they had free time within their busy work schedule. Once they felt they had adequately tested the prototype they were asked to fill out the evaluation form and save it onto the network drive to be picked up or print it out so that we could obtain the

evaluation data. We found this offered the doctors the flexibility they needed to perform their daily tasks as well as assist us in our study. During these evaluation meetings many suggestions were made and concerns were brought up. Most of the suggestions were simple and focused on the naming and positioning of the columns of the EFS because most of the doctors were dissatisfied with the present layout of the MFS; EFS was an improvement and they wished more changes. One of the bigger concerns was related to 'Security' and 'Access rights' for users of the EFS. We had not addressed those issues within the EFS because it did not fit into the scope of out project, however, due to the sensitivity and the privacy related to medical information, security is a major concern for administrators. With future iterations we can enforce the EFS to the same security level as the current hospital databases.

5.3 Results

In this section we will present the results of our evaluation of the EFS and QMP along with an analysis and its match with our expectations.

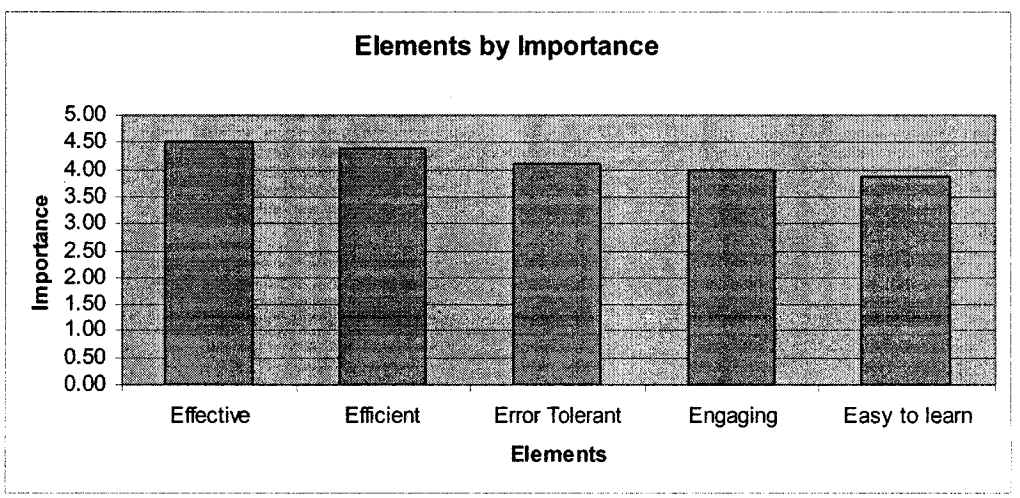


Figure 5.1-Median Values of Elements by Importance

The Questionnaire, shown in Appendix B, was split into four differing sections. The first section pertained to the personal preferences of different doctors or characteristics of software that were of particular importance to them. Characteristics of software such as *effectiveness*, *efficiency*, *engaging*, *error tolerance*, *learn ability* were rated by them from 1 to 5 where 1 was considered less important and 5 being more important. Within our evaluation *efficiency* refers to the property that the software allows its users to perform tasks faster, also known as *productivity*. We gathered this input because we felt that by determining the preferences of doctors, one could better understand their responses in light of what is important to them. The results of this section can be seen in Figure 5.1 and has been ordered by importance of the elements. We find that the two most important elements to doctors are the effectiveness and efficiency of software whereas the lowest in importance is ease of learning. We refer to effectiveness as the characteristic of software to help the doctors perform their task easily and efficiently. From the bar chart above we find that this set of doctors, whom we approached, are willing to invest time in learning complex software if the software is perceived to be effective and efficient for them to do their job. This may be their understanding that a small amount of time invested in learning to use the software will save them more time later on if the software is efficient and effective in helping them perform their daily tasks

The next section contained 9 statements taken from the original industry Standard SUMI questionnaire [Appendix C] commonly used for usability studies. **SUMI**, Software Usability Measurement Inventory, is a *de facto* industry standard of proving software

quality from a user's point of view. Given our list of 9 statements the doctors were asked to what degree they agreed with the statements provided. Using the answers from the section specialized software was used to analyze and to break down the results into the six software indicators: (1) global software quality, (2) efficiency, (3) effect of the software on work, (4) helpfulness of the help material, (5) control of the user over the software and (6) learnability. The results of our evaluation of these indicators are shown in Figure 5.2.

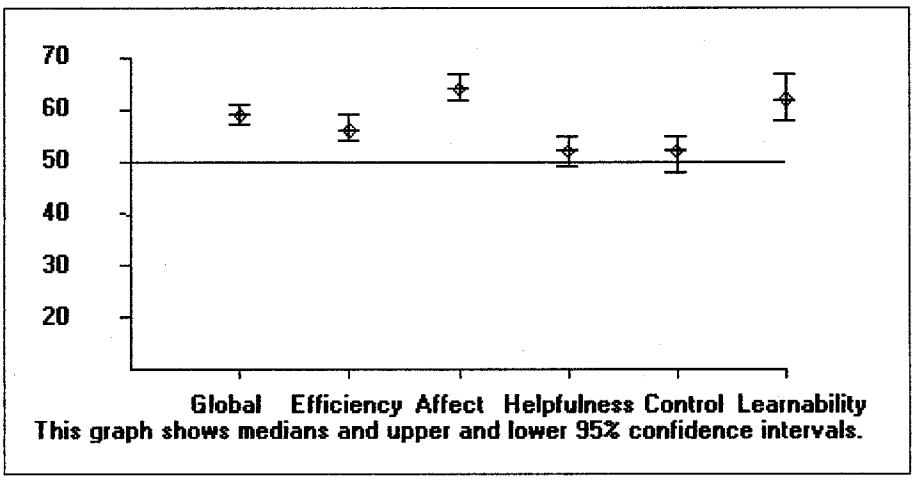


Figure 5.2- SUMI Results

Figure 5.2 shows the results of our SUMI questions showing the mean as well as the upper and lower 95% confidence intervals. We see that the 95% confidence intervals are small showing that we have a rather homogeneous group of testers. We can see from the figure that the 2 highest rated are Affect and Learnability. Affect can be explained through our close working relationship with our testers that helped us tailor the software to their needs. Because the software was specifically created to help them in their daily tasks we can see that the affect on their work has been greatly affected in a positive way. The high score in learnability can be explained because demonstrations and evaluations were done in a one to one fashion allowing doctors an easier learning time. Another

factor that contributed to high learnability is the use of the MFS as a metaphor for the EFS meaning that doctors were already familiar with its layout or look and feel. The score for Helpfulness was low as in most cases the user manual was not needed in the evaluation but was made available.

The third section asked the doctors to consider 5 tasks and compare the ease of performing these tasks in the EFS to that of the current MFS. With respect to these tasks, the doctors were also asked the frequency with which they performed such tasks in a week. The goal of this section was to see if our EFS was an effective solution over their current method of operation while taking into account the frequency of such tasks. We also left 2 open spaces so that the doctors could create their own tasks to rate our EFS against the MFS.

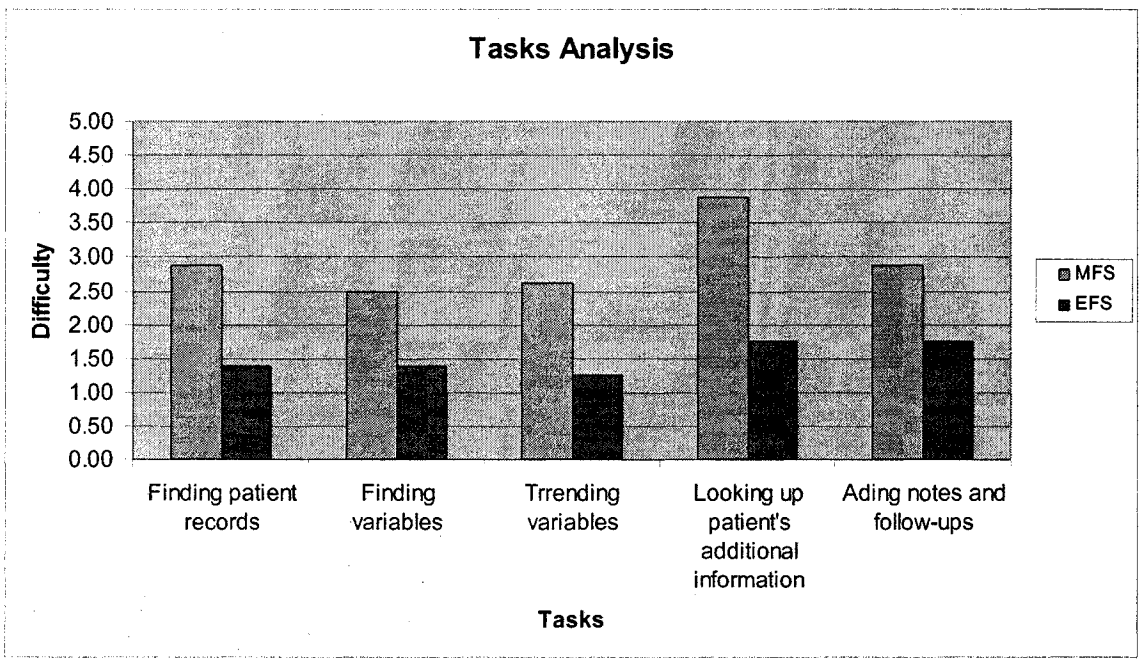


Figure 5.3-Median Values of Task Comparisons between the MFS and EFS

For the tasks listed we find that the EFS performs better than the MFS. The first two tasks, "finding patients" and "finding variables", were tasks that we chose because of

their contribution to efficiency. In real life, doctors waste time when they search for patient variables on the MFS, patient's charts or even for additional information. This varies, however, as doctors in the transplant clinic have their patient's charts maintained by nurse coordinators whereas ward doctors spend more time searching for charts. Trending variables and the ability to add follow-up notes with the EFS for other doctors are tasks that allowed doctors to perform their job more effectively. Clinical practice and clinical research are collaborative activities between caregivers so an efficient communication method is necessary. Also, trending variables in the EFS is easier and it is done graphically unlike the MFS which has many other uses such as teaching.

The last section referred to the usage of the QMP. The doctors were asked to perform several queries using the QMP and then we asked them to rate the ease with which they could construct the needed constraints of the query. Due to the complexity of the database design, only the first and second queries, which were simple, would yield results. The other queries worked as a proof of concept that the QMP is useable and easy to learn for doctors wishing to create queries to retrieve information. There were some difficulties that arose during this part of the evaluation because only the first two queries generated results from the database. Because of the complexity of the database, generating the specific SQL string to satisfy the last two queries were complicated and were left out. Another reason these queries were not implemented was because the Transplant Database would be undergoing several changes in order to be integrated with a hospital project. The changes would disrupt the functioning of the QMP so we did not find it effective to integrate the harder queries into the QMP. We informed the doctors that the implementation was not complete and the UI evaluation was on the proof of

concept. Eventually all aspects of the QMP functionalities can be implemented. The bar chart below shows the ease with which the queries were entered by the doctors. The figures show that the queries got more difficult for the doctors as more criteria were requested in the evaluation form. This is to be expected as more criteria would require the doctors to use additional parts of the UI, however, none found them impossible to do.

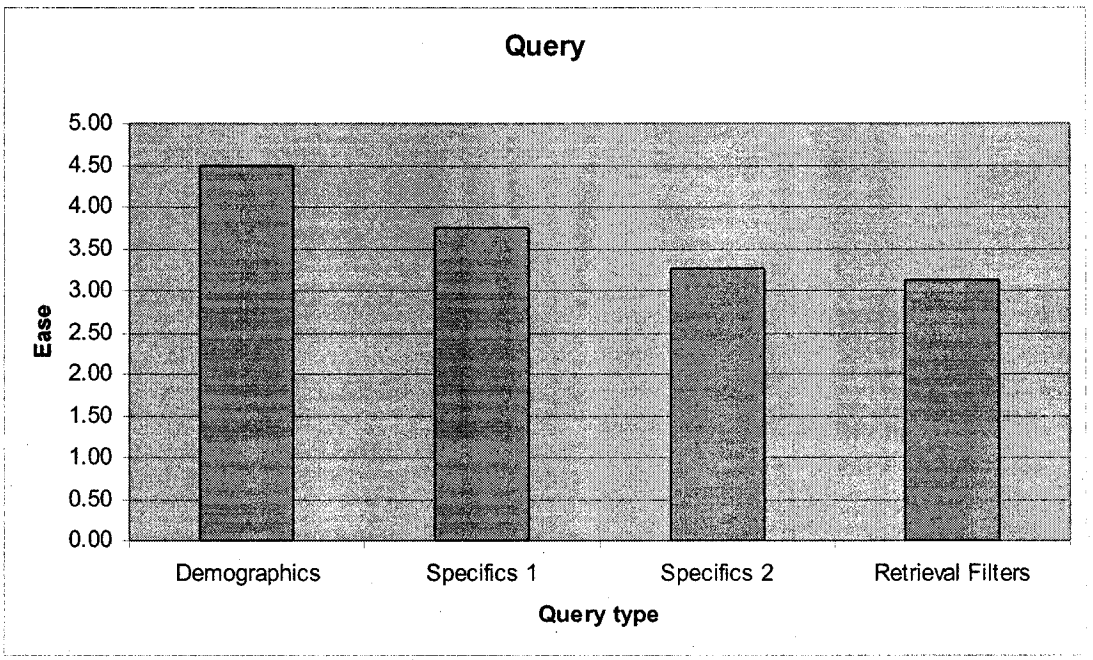


Figure 5.4-Median Values for Ease of Building Queries

5.4 Difficulties in Evaluation

Several difficulties arose during our evaluation of our prototype. Firstly, as doctors are almost always busy either being on call, in clinic or in surgery scheduling appointments with certain doctors were difficult. At times meetings had to be interrupted and we had to rush the demonstrations.

Another difficulty arose after our demonstrations when the doctors were asked to try on their own and then complete the evaluation. We found that because most doctors

are very busy they tend to forget and needed to be reminded weekly to try to finish the evaluations. Some doctors felt comfortable completing the evaluation form few hours of use after our meeting while others took 3-4 weeks to return their evaluation sheets.

The wide range in computer knowledge between doctors also proved a challenge during our evaluation. Some understood our interaction methods quickly while others took longer to get use to them. We do not find that this had a significant impact on the evaluation of our prototype as the doctors still found our prototype easy to learn and to use.

5.4 Summary

We found that in all cases the EFS outperformed the MFS in all tasks that we evaluated. However, the range of ease for which doctors found these tasks varied with the MFS due to their location in the hospital. Doctors in the transplant clinic found it easier to find the MFS as in most cases the nurse coordinator kept track of patient charts compared to the ward. Through our evaluation we find that a close working relationship along with frequent meetings and many development iterations helped build software that was easy to learn, easy to use and an effective support in the daily work of the busy healthcare professionals.

6 Conclusion and Future Work

6.1 Summary and Conclusion

The benefits of computerization are many and are increasing as new technologies in hardware and software arise. However, the benefits of computerization have had little impact on the way hospitals and other health organizations function due to the fact that integration of these new technologies have been slow and limited in this domain. Hospital functions, in terms of patient care, have been autonomous as most hospitals have not had any major need to interact with each other. We believe that clinical practice and clinical research can be supported by suitable software that provides interactive human centered exploration and medical data visualization. This thesis addresses the problem which currently exists within the healthcare field in the development of support both for clinical practice and clinical research.

By using the agile method of software development coupled with the personal software process (PSP) we have managed to create software that has proven to be useable and beneficial to researchers. We applied, within the scope of a graduate thesis, the UCD methodology during the creation and refinement of our UI for both the EFS and the QMP. With the help of quick prototyping along with meetings with doctors and researchers we have managed to document their needs and incorporate the required functionalities into our prototype.

Also, by using the currently existing Manual Flow Sheet as a metaphor for our prototype we reduced the learning time for our Electronic Flow Sheet (EFS). During many demonstrations of our prototype we found that clinicians could easily identify and find the variables of interest to them. They could also point out areas within the MFS that were improved in the EFS as well as areas that still needed improvements. From this experience, we believe that application software developers should look to discover suitable metaphors by studying the existing workflow practices. We also improved learning of the QMP by creating logical categories for the constraint filters of the queries. By grouping these constraints to match the mental model the doctors had of the data we made it easier to use.

Our evaluation of the EFS and QMP led us to the following conclusions.

- Our Software is Useable: In our evaluation we found that our EFS outperforms the MFS in all tasks in terms of functionality and accessibility. This conclusion was drawn from the results of Section 3 of our questionnaire.
- Our Software Usability is good: Based on the SUMI portion of our questionnaire we found that the usability of our software is good. Our prototype scored above average on all the SUMI software indicators which are global quality, efficiency, affect, helpfulness, control and learnability.
- Through our evaluation we also found what properties of software are most important to doctors. The two most important properties were effectiveness and efficiency which were met in our EFS and QMP.

- The QMP allows the users to perform research and has the potential to support clinical research.

6.2 Future Work

There are still many areas of research associated with Medical Data Visualization. Most of these areas are associated with querying as well as computer supported collaborative work. Some future work associated with this thesis is listed bellow.

1. Repeatability and applicability of the EFS and QMP design methodology using workflow analysis and user recognizable metaphors at different hospitals with differing users.
2. The incorporation of data visualization for scans and images and not just textual and numeric data needs to be looked at. Visualization and interaction methods with scans and images need to also be addressed to see where it can benefit clinical research and practice.
3. The adaptability of the UCD by industrial software developers where one is constantly challenged with diverse user types contrasted with the need to use metaphors familiar to users.
4. One major concern that was brought up in relation to this thesis was privacy and security concerns. Users accessing the EFS and QMP would need to be restricted and as many more databases are connected from external sources security would have to be addressed in depth.

7. References

[1] L. T. Kohn, J. M. Corrigan, M. S. Donaldson, "To Err is Human: Building a Safer Health System." Washington, D.C, *Institute of Medicine*, National Academy Press, pp. 312, 1999

[2] A. Narasimhadevara, "Iterative Design and Evaluation of a Software System for Nursing Applications in Healthcare" Masters Thesis, Concordia University, Montreal, Jan 2006

[3] J. C. Prather, D. F. Lobach, L. K. Goodwin, J. W. Hales, M. L. Hage, W. E. Hammond, "Medical Data Mining: Knowledge Discovery in a Clinical Data Warehouse", *Proceedings of Annual Conference of AMIA*, pp. 101–105, 1997

[4] C. G. Burgess, "A graphical, database-querying interface for casual, naïve computer users", *Int. J. Man-Machine Studies*, vol. 34, pp. 23-47, 1991

[5] D.W. Bates, M. Cohen, L. L. Leape, J. M. Overhage, M. M. Shabot, et al., "Reducing the Frequency of Errors in Medicine Using Information Technology", *J Am Med Inform Assoc*, vol. 8, pp. 299-308, 2001

[6] B. Raymond, C. Dold, "Clinical Information Systems: Achieving the Vision", Kaiser Permanente Institute for Health Policy, Oakland, Feb 2002

[7] J. G. Anderson, "Clearing the way for Physicians' use of clinical information systems", *J Commun., ACM,* vol. 40, pp. 83-90, 1997

[8] S. Card, J. Mackinlay, B, Shneiderman, "Readings in Information Visualization: Using Vision to Think", Morgan Kaufmann Publishers, 1999

[9] L. Chittaro, C. Combi, G. Trapasso, "Visual Data Mining of Clinical Databases: an Application to the Hemodialytic Treatment based on 3D Interactive Bar Charts". *Proceedings of VDM: 2nd International Workshop on Visual Data Mining*, Helsinki, Finland. 2002

[10] M. Wattenberg, "Sketching a Graph to Query a Time-Series Database", *Conference on Human Factors in Computing Systems* (CHI'01), pp. 381-382, 2001

[11] Y. Shahar, C. Combi, "Timing is Everything: Time-Oriented Clinical Information Systems", *West J Med* vol. 168(2):pp. 105-113, 1998

[12] J.M. Wilson, "Gantt Charts: A centenary appreciation", *European Journal of Operational Research*, vol. 149 (2), pp. 430-437, 2003

[13] L. Chittaro, C. Combi, "Representation of Temporal Intervals and Relations: Information Visualization Aspects and their Evaluation", 8th *International Symposium on Temporal Representation and Reasoning* (TIME-01), pp. 13-20, 2001

[14] C. Plaisant, R. Mushlin, A. Snyder, J. Li, D. Heller, B. Schneiderman, "Lifelines: Using Visualization to Enhance Navigation and Analysis of Patient Records", *Revised version in American Medical Informatics Association Annual Fall Symposium* AMIA, pp. 76--80, 1998.

[15] R. Kosara, S. Miksch, "A Visualization of Medical Therapy Plans compared to Gantt and PERT Charts", *Proceedings of the Seventh International Workshop on Temporal Representation and Reasoning* (TIME-00), pp. 153-181, 2000

[16] S. Martins, Y. Shahar, M. Galperin, "Evaluation of KNAVE-II: a Tool for Intelligent Query and Exploration of Patient Data", *Medinfo*. vol. 11(1), pp. 648-52, 2004

[17] Y. Shahar, C. Cheng, "Intelligent Visualization and Exploration of Time-Oriented Clinical Data", *HICSS-32*, 1999

[18] J. Nielsen, R. Mack, "Usability Inspection Methods", *John Wiley & Sons*, New York, pp. 25-62, 1994

[19] W. Aigner, "Interactive Visualization of Time-Oriented Treatment Plans and Patient Data", Masters Thesis, *Vienna University of Technology*, Vienna, Austria, 2003

[20] Oacis EMR, http://www.emergis.com/solutions/health/clinical_care/default.aspx

[21] E.H. Shortliffe, V.L. Patel, J.J. Cimino, G.O. Barnett, R.A. Greenes, "A study of collaboration among medical informatics research laboratories", *Artificial Intelligence in Medicine* vol.12(2), pp. 97-123, Feb 1998

[22] A. Cockburn, "Agile Software Development", 1st Edition ed., *Addison-Wesley*, 2001

[23] W.S. Humphrey, "A Discipline for Software Engineering", *Addison-Wesley Professional*, Jan 1995

[24] R.E. Jeffries, XProgramming.com: An Agile Software Development Resource, *http://www.XProgramming.com*, Jan 2008.

[25] W. S. Humphrey, "Introduction to the Personal Software Process", *Addison-Wesley*, Nov 2005

[26] Carnegie Mellon, Personal Software Process (PSP), http://www.sei.cmu.edu/tsp/psp.html, Jan 2008.

[27] R. Radhakrishnan, "Model Driven Educational Assistance for Patients" Masters Thesis, *Concordia University*, Montreal, Spring 2007

8. Appendix

A. EFS and QMP User Manual

Electronic Flow Sheet

INTRODUCTION:

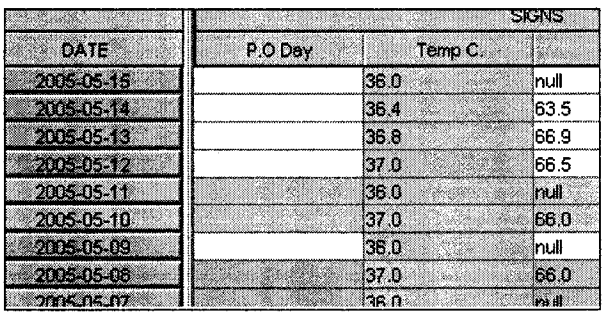
This Software Research & Development work is done as a part of the Masters in Software Engineering Thesis of Brian Leung working for his degree at Concordia University. A working prototype is demonstrated in this study.

Based on the notion (metaphor) of the "flow sheet" used in many chronic care hospitals or transplant wards, we have created the notion of EFS or electronic flow sheet. The intended use of EFS is to support in the improvement clinical practice and clinical research. The EFS is a software system with appropriate user interface to a back end database. In our present case the back end database is the Transplant Database used at the RVH. Technically it is feasible, in future, to make the back end as a "Distributed Database" that can be the union of transplant databases from different hospitals and centers connected by means of a "highly secured, high speed, wide area network connection"

FUNCTIONALITY:

Squeezing – This is a name given to "selecting certain number of rows or columns and displaying them together without showing the unselected rows and columns". The EFS allows permanent as well as temporary squeezing of columns. Permanently squeezing a column means that for all patients, this effect will occur, however, for temporary squeezing, this effect will be removed when the researcher switches to another patient.

Squeezing can be done to both Columns and Rows. Simple Right-Click on the Columns or Date's of interest (1) and selects the "Compile" button on the Bottom Left (2).



	SIGNS		
DATE	P.O Day	Temp C.	
2005-05-15		36.0	null
2005-05-14		36.4	63.5
2005-05-13		36.8	66.9
2005-05-12		37.0	66.5
2005-05-11		36.0	null
2005-05-10		37.0	66.0
2005-05-09		36.0	null
2005-05-08		37.0	66.0
2005-05-07		36.0	null

Figure 8.1-- Selecting Columns and Rows (1)

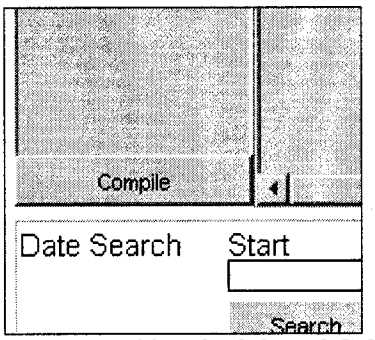


Figure 8.2 - Compiling the Selected Columns and Rows Together (2)

Date Searching – This functionality allows for the researcher to search for a date range. Currently the EFS will only display 21 data entries due to programming limitations.

To Search for a Date Range, go to the Date Search area at the bottom left and select a Start and an End date (2). Once this has been done, select the "Search" button (1).

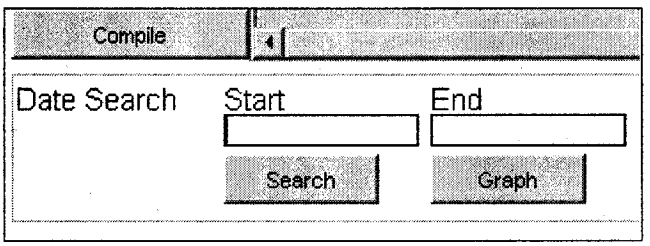


Figure 8.3 - Date Search Area

Figure 8.4 - Date Selection Pop-up

Graphing – This functionality allows the researcher to view the data of interest in graphical form. The first graph shows the combination of all the variables super-imposed on each other. The subsequent graphs underneath show each variable individually.

Graphing is done by right-clicking to highlight the columns of interest (seen in Figure 1) and then selecting the "Graph" button on the bottom left (seen in Figure 3).

Notes – Allows for Researchers to add Notes associated with certain dates

To add Notes, Simply Double-Click on the Notes column. Text can also be added to the Diagnosis Column as well. To Save and Exit push F10, to Exit without saving push F9 or the 'x' at the top right corner of the pop-up.

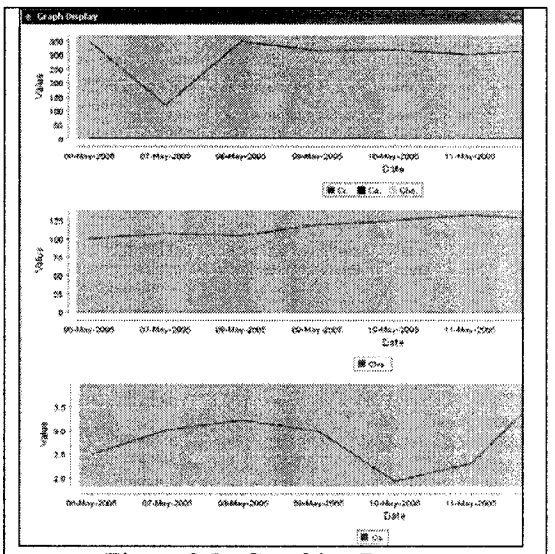


Figure 8.5 - Graphing Pop-up

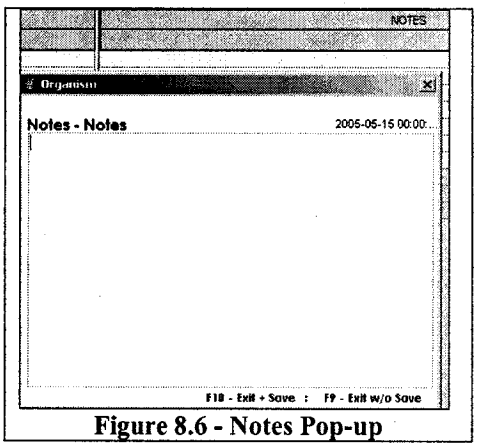


Figure 8.6 - Notes Pop-up

Exams, Consults, Medications – To access this section, simply select it from the Menu Bar on top. This functionality allows the user to see the Exams done, Consults, and current Medications list as seen in the Transplant Database.

Batch MRN Searching – To open the Batch MRN search window Double-Click on the MRN. Once this window is open, enter the MRN's you wish to retrieve in the box while pushing "Enter" after each one. While the EFS is searching the "pending..." tag will be shown next to MRN's that are being retrieved. The "Complete" tag means that the MRN has been retrieved and can be shown by clicking on the desired MRN. The "Not Found" tag will appear if the MRN does not exist in the Database.

Query Module (QM)

Query Module is a facility using which you can select patient records based on a "composite criteria" and put all the selected records in the format of a EFS. On that constructed data set you can do operations like what you did on the EFS, that is squeezing, sub-setting, graphing etc.

Use of QM requires the following 3 steps.

Step 1: Switch from EFS mode to QMP mode by clicking on the button on the top-left corner. This works as a toggle button. You can switch back from QMP to EFS mode again, if you wish.

Step 2: Construct the "composite criteria" which is used as a filter to select records from the Transplant Database. The criteria can be very complex. It is constructed in a step by step manner as a combination of many "conditions". As you interactively specify a condition it is entered in the right side "panel" which we call it as the "Criteria Panel".

There are 4 basic groups to specify the conditions for selection. They are: Unique Identifier of the Patient Records; Demographic filters that deal with age, sex etc; specific filter which deals with medications etc; and Retrieval filters. In order to specify a condition, when you click on that parameter you might see a dialog box pops up to ask you for further inputs needed in specifying the condition based on that parameter. After specifying the condition you ADD it to the "Criteria Panel". As you specify the various conditions they appear on the criteria panel.

Step 3: The elements on the criteria panel can be deleted if you wish to revise the "composite criteria" to be used. Finally press the ACCEPT button on this panel.

Then, the query module software will use these criteria and select the relevant records from the Transplant Database and display them on the screen below this panel.

Note that you can view the retrieved records by manipulating them as you wish. By combining what you observe with your own "intuition" or "mental reasoning" you could go back and modify the "composite criteria" and re-submit for the query module to process again.

Functionality

There are 5 Sections to the Query Module. Four of them are color coded condition specifying sub-panels and the fifth one is the "Criteria Panel". You can select any item from the sub panels by double clicking on it and then specify that condition further. For some of them like MRN what you need is only double clicking. It will immediately appear on the composite criteria panel.

Unique Identifiers – These are the identifiers used to identify patients. Double-clicking on the variables in this section will simply add the variable to the Filter Summary table since no meaningful constraints can be added to them.

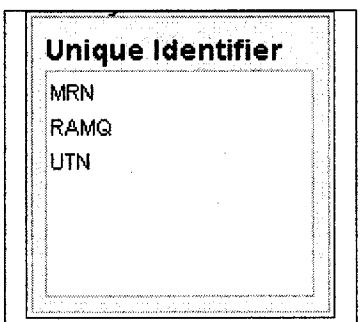


Figure 8.7 - Unique Identifiers

Demographic Filter – Demographic Filters contains variables that concern demographics without taking into consideration medical and transplant information. The variables in this section can be further broken down into other categories based on the constraints they have associated with them.

Numeric Constraint – Age, Weight Height
Date Constraint – Date of TX
String Constraint – Sex, Organ, Patient Type, ABO RH

Specific Filter – Specific Filters contains variables that concern pre-evaluation and transplant variables. In this section, all variables are categorized as String Constraints, however, on-top of String Constraints, some variables have an added Numeric and Date Constraint such as Diagnosis, Medications, etc…

Lab Filters – Lab Filters contains variables from Labs and other pre-calculated Scores/Indicators.

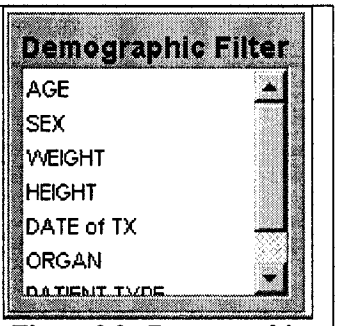


Figure 8.8 - Demographic Filters

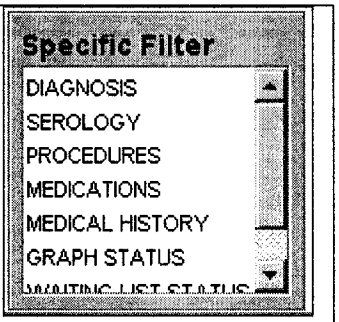


Figure 8.9 - Specific Filters

Criteria Panel – This is the panel which holds the added constraints and will pop-up each time a variable is double-clicked from the above sections. The List on the left hand side will show all the possible values for the current variable. By clicking the button to its right will open up further constraints available for that value.

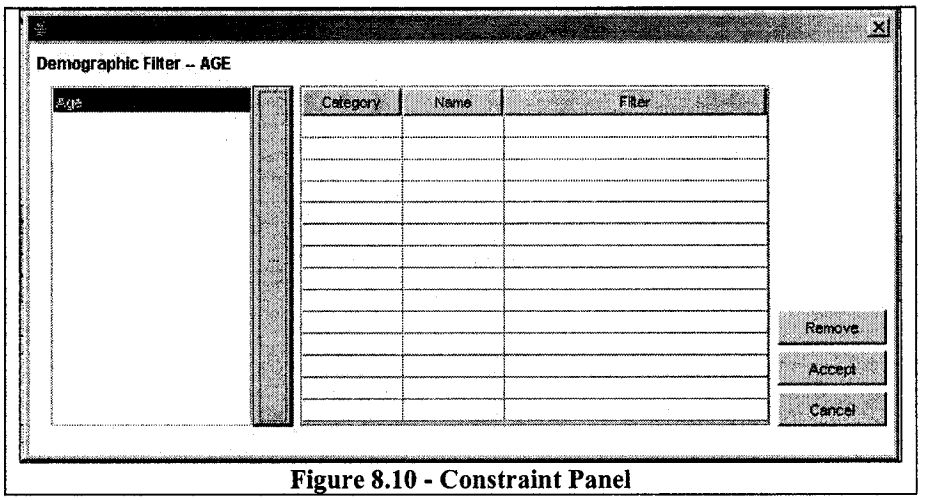


Figure 8.10 - Constraint Panel

Numeric Constraints – If the selected Item can be constrained numerically a Numeric Constraint Pop-up will be visible when you click the adding button.

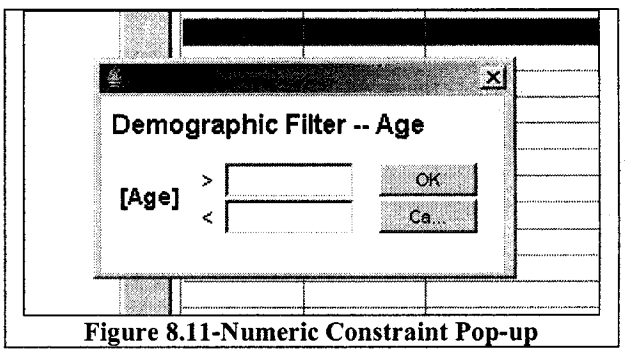


Figure 8.11-Numeric Constraint Pop-up

String Constraints – In most cases all the possible String Constraints that can be added will appear on the box in the left hand side. In some cases, such as medications, where a Date Constraint can be added, once the Medication has been added to the Filter Table on the right of the Constraint Panel, you can double-click the corresponding medication to add the Date Constraint.

Date Constraints – Date constraints work exactly like the Numeric Constraints panel.

B. EFS and QMP Questionnaire

Electronic Flow Sheet/ Query Module Test Study

PLEASE RATE THE FOLLOWING ELEMENTS

Using the scale: <Lowest, Low, Medium, High, Highest>

Category	Definition	Rating of Importance
1. Effective	The completeness and accuracy with which you can achieve what you set to do.	
2. Efficient	The speed (that is avoiding delays) with which you can complete the task you set to do.	
3. Engaging	The degree to which the tone and style of the tool makes it pleasant and satisfying for you to use.	
4. Error tolerant	How well the interface design prevents errors or helps you to recover from errors if occur.	
5. Easy to learn	How well the tool supports both initial orientation and deepening understanding of its capabilities.	

SUMI Questions – Standard Questions Used in the Evaluation of User Interface to Software Systems

	Agree	Somewhat Agree	Undecided	Somewhat Disagree	Disagree
1. It is easy to make the software do exactly what I do at my work 2. Tasks can be performed in a straightforward manner.					
3. This software responds too slowly to inputs					
4. Working with this software is satisfying. 5. Working with this software is stimulating.					
6. I thing this software is inconsistent. 7. Using this software is frustrating.					
8. I sometimes don't know what to do next with this software. 9. Learning to use new functions is difficult.					

Task Based Evaluation of the EFS (Electronic Flow Sheet)

Rate it on a scale of 1 to 5: 1 is Easy and 5 is Hard.

Tasks	Context	Manual Flow Sheet	Electronic Flow Sheet	Frequency
1. Finding Patient Records	In order to treat a patient, constantly referring to the patient's labs and flow sheets are required. As each patient has their own physical file, this file must be located for each patient that needs treatment. How easy is it to find a patients record, without asking for the help of coordinators, or when coordinators are not available?			
2. Finding Variables	When considering a patient's case, you might refer to two or more variables. Blood enzymes are examples of variables. These variables, however, may vary depending on what you are looking for. How easy is it to find two or more variables which are jointly of interest?			
3. Trending of Variables of Interest	The overall trending of variables, at times, tells a lot about the overall condition of the patient such that bad effects can be avoided or predicted. How easy is it to find trends in patient variables?			

3. Looking up Patients Medications, Exams and Consults (and related artifacts)	There are times when just the Lab results may not be sufficient for you when you are considering a patient; and additional information is needed. Examples: Medications, Exams and Consults can give a better picture of the current state, or the past state, of the patient in question. Suppose there exists a situation where variables are not enough, how easy is it to get access to the additional information?			
4. Adding notes and follow-ups for other Doctors	Being able to communicate to a transplant team about the treatment for a patient is essential in organ transplant care. One form of communication is writing a 'note'. There are many people that are involved in caring for patients. How effective is it to communicate these notes to other care-givers?			
5. <Your choice of Task not mentioned above>				
6. <Your choice of Task not mentioned above>				

Note: Frequency Column refers to how often these tasks are done within a fixed time frame and values should add up to 100.

Tasks & Evaluation of the Query Module.

Rate it on a scale of 1 to 5: 1 is Lowest and 5 is Highest.

Area	Query	Ease (1-5)
1. Demographics	Find all the Patients over 25 but below 50 that have had a Liver Transplant using their MRN as their Identifier.	
2. Specifics	Find all the Patients who have had Pancreas and Kidney Transplants within the past year with the Graph Status X.	
3. Specifics	Find all the Patients between 30 and 60 who have had Kidney Transplants with Diagnosis X and were given Medication Y, 10 weeks Post Op, and Medication Z, 20 weeks Post Op.	
4. Retrieval Filters	Find the Labs X, Y, Z for all Patients who have had Liver Transplants.	

C. SUMI Questionnaire

SOFTWARE USABILITY MEASUREMENT INVENTORY

(SUMI)

Your name . . . . . .

Name of software . . . . . .

Date . . . . . .

NB *the information you provide is kept completely confidential, and no information is stored on computer media that could identify you as a person.*

This inventory has fifty statements. Please answer every one of them. Against each statement there are three boxes.

You should mark the first box if you generally AGREE with the statement. Mark the central box if you are UNDECIDED, can't make up your mind, or if the statement has no relevance to your software or to your situation. Mark the right box if you generally DISAGREE with the statement.

In marking the left or right box you are not necessarily indicating *strong* agreement or disagreement but just your general feeling most of the time.

AGREE UNDECIDED DISAGREE

Put a ✓ mark in the box of your choice.

HFRG - SUMI - Q - 3.9UK

		Agree ↓	Undecided ↓	Disagree ↓
1	This software responds too slowly to inputs.	☐	☐	☐
2	I would recommend this software to my colleagues.	☐	☐	☐
3	The instructions and prompts are helpful.	☐	☐	☐
4	The software has at some time stopped unexpectedly.	☐	☐	☐
5	Learning to operate this software initially is full of problems.	☐	☐	☐
6	I sometimes don't know what to do next with this software.	☐	☐	☐
7	I enjoy my sessions with this software.	☐	☐	☐
8	I find that the help information given by this software is not very useful.	☐	☐	☐
9	If this software stops, it is not easy to restart it.	☐	☐	☐
10	It takes too long to learn the software commands.	☐	☐	☐
11	I sometimes wonder if I'm using the right command.	☐	☐	☐
12	Working with this software is satisfying.	☐	☐	☐
13	The way that system information is presented is clear and understandable.	☐	☐	☐
14	I feel safer if I use only a few familiar commands or operations.	☐	☐	☐
15	The software documentation is very informative.	☐	☐	☐
16	This software seems to disrupt the way I normally like to arrange my work.	☐	☐	☐
17	Working with this software is mentally stimulating.	☐	☐	☐
18	There is never enough information on the screen when it's needed.	☐	☐	☐
19	I feel in command of this software when I am using it.	☐	☐	☐
20	I prefer to stick to the facilities that I know best.	☐	☐	☐

		Agree	Undecided	Disagree
21	I think this software is inconsistent.	☐	☐	☐
22	I would not like to use this software every day.	☐	☐	☐
23	I can understand and act on the information provided by this software.	☐	☐	☐
24	This software is awkward when I want to do something which is not standard.	☐	☐	☐
25	There is too much to read before you can use the software.	☐	☐	☐
26	Tasks can be performed in a straightforward manner using this software.	☐	☐	☐
27	Using this software is frustrating.	☐	☐	☐
28	The software has helped me overcome any problems I have had in using it.	☐	☐	☐
29	The speed of this software is fast enough.	☐	☐	☐
30	I keep having to go back to look at the guides.	☐	☐	☐
31	It is obvious that user needs have been fully taken into consideration.	☐	☐	☐
32	There have been times in using this software when I have felt quite tense.	☐	☐	☐
33	The organisation of the menus or information lists seems quite logical.	☐	☐	☐
34	The software allows the user to be economic of keystrokes.	☐	☐	☐
35	Learning how to use new functions is difficult.	☐	☐	☐
36	There are too many steps required to get something to work.	☐	☐	☐
37	I think this software has made me have a headache on occasion.	☐	☐	☐
38	Error prevention messages are not adequate.	☐	☐	☐
39	It is easy to make the software do exactly what you want.	☐	☐	☐
40	I will never learn to use all that is offered in this software.	☐	☐	☐

Please continue overleaf

		Agree	Undecided	Disagree
41	The software hasn't always done what I was expecting.	☐	☐	☐
42	The software has a very attractive presentation.	☐	☐	☐
43	Either the amount or quality of the help information varies across the system.	☐	☐	☐
44	It is relatively easy to move from one part of a task to another.	☐	☐	☐
45	It is easy to forget how to do things with this software.	☐	☐	☐
46	This software occasionally behaves in a way which can't be understood.	☐	☐	☐
47	This software is really very awkward.	☐	☐	☐
48	It is easy to see at a glance what the options are at each stage.	☐	☐	☐
49	Getting data files in and out of the system is not easy.	☐	☐	☐
50	I have to look for assistance most times when I use this software	☐	☐	☐

Please check you have ticked each item.
Thank you.

D. Class Diagrams

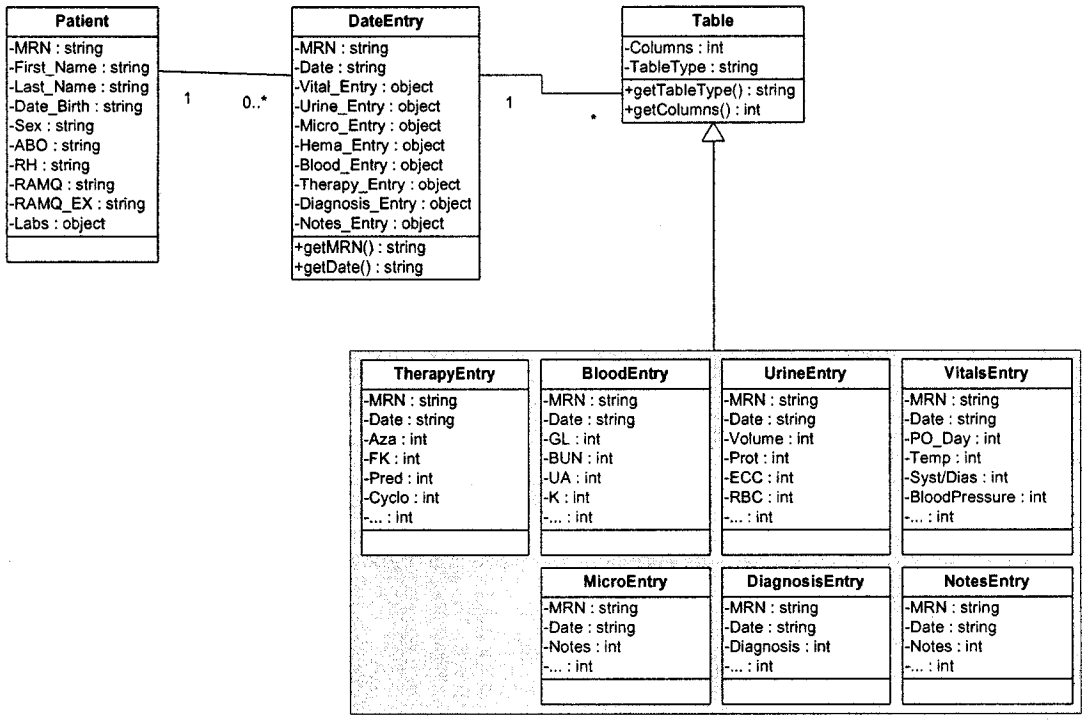


Figure 8.12-Patient Class and Related Classes

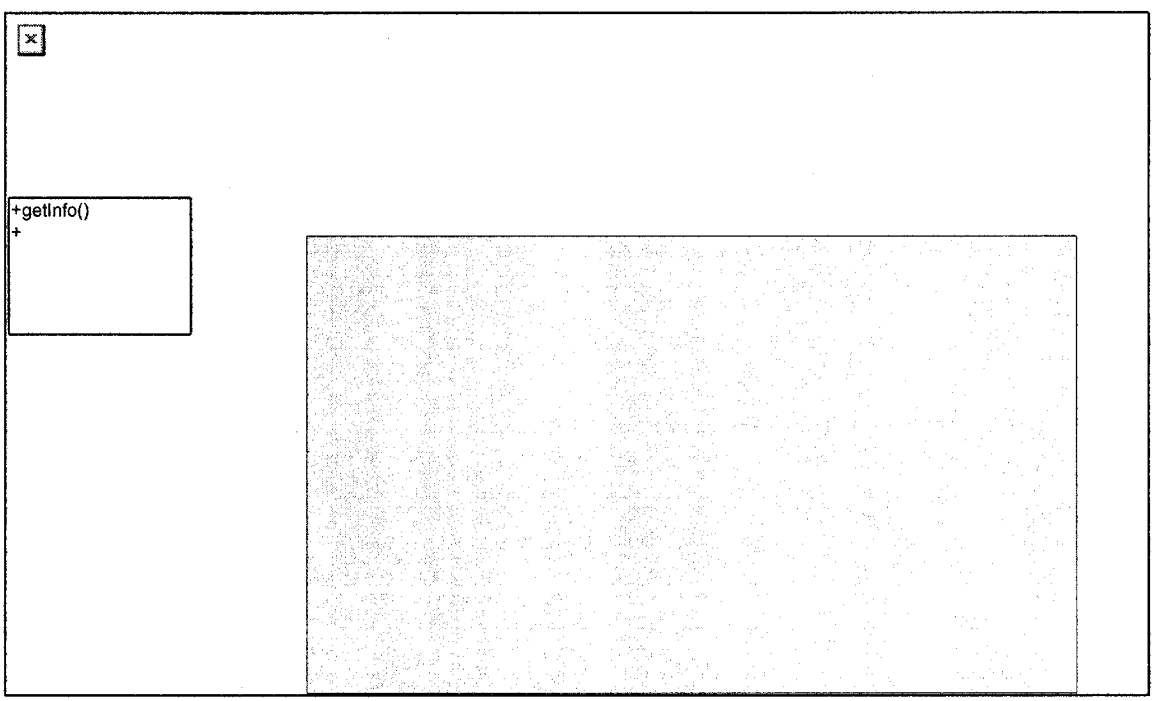


Figure 8.13-QMP Filter and Related Classes

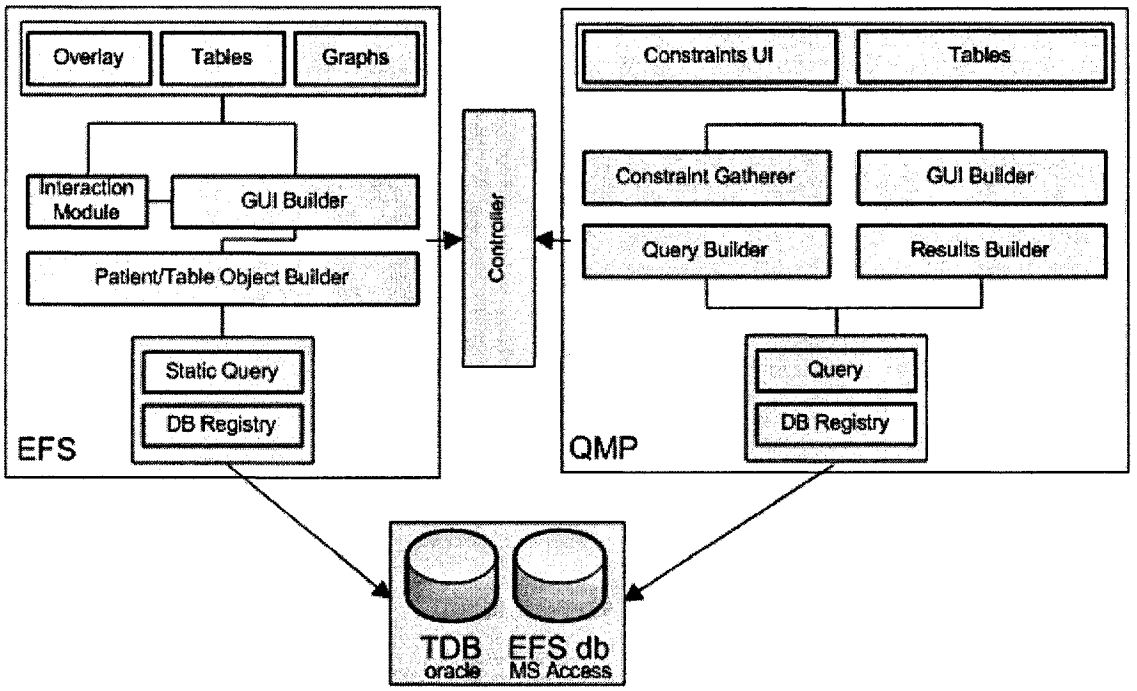


Figure 8.14-Software Architecture

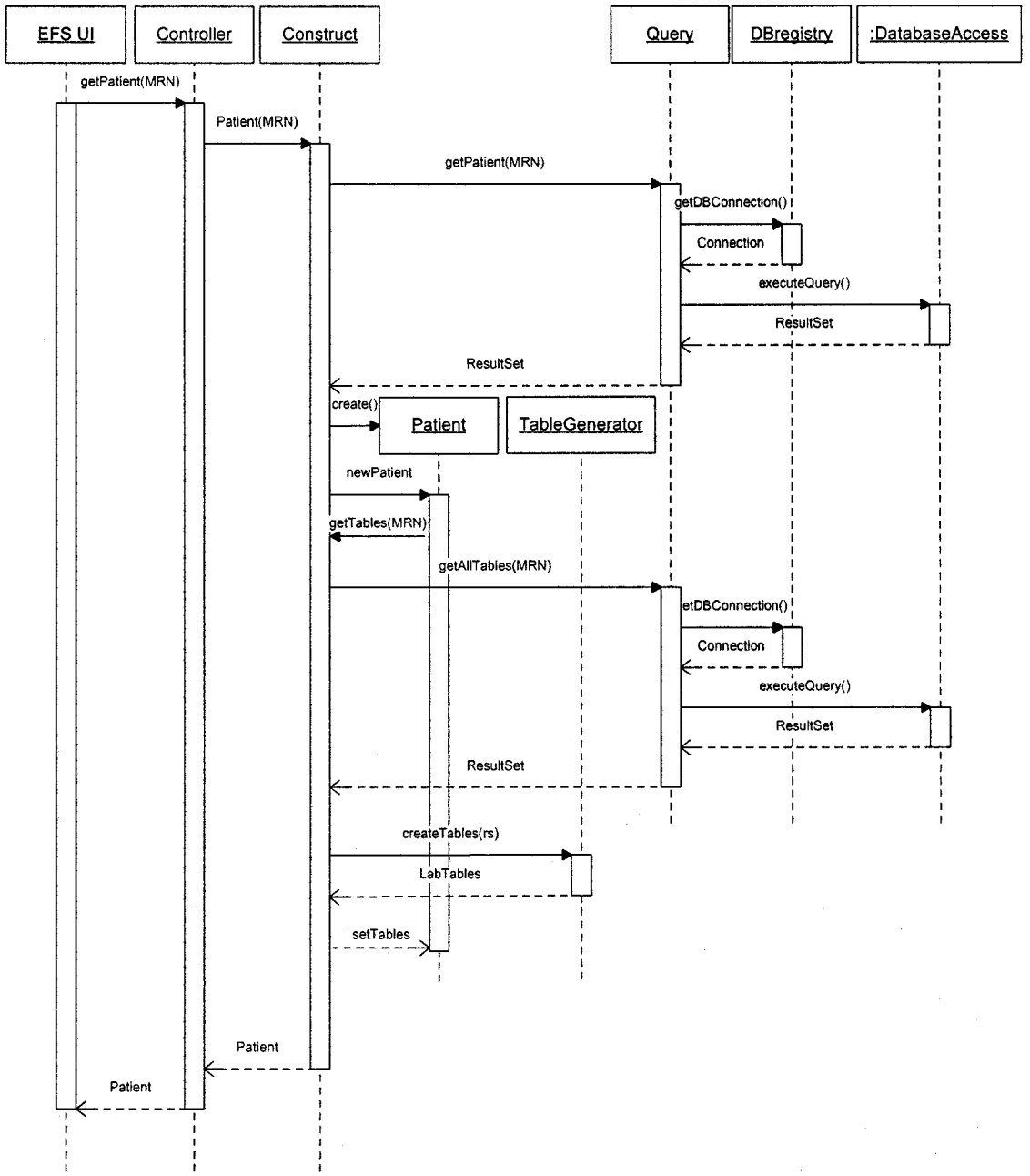


Figure 8.15-Patient Creation Sequence Diagram

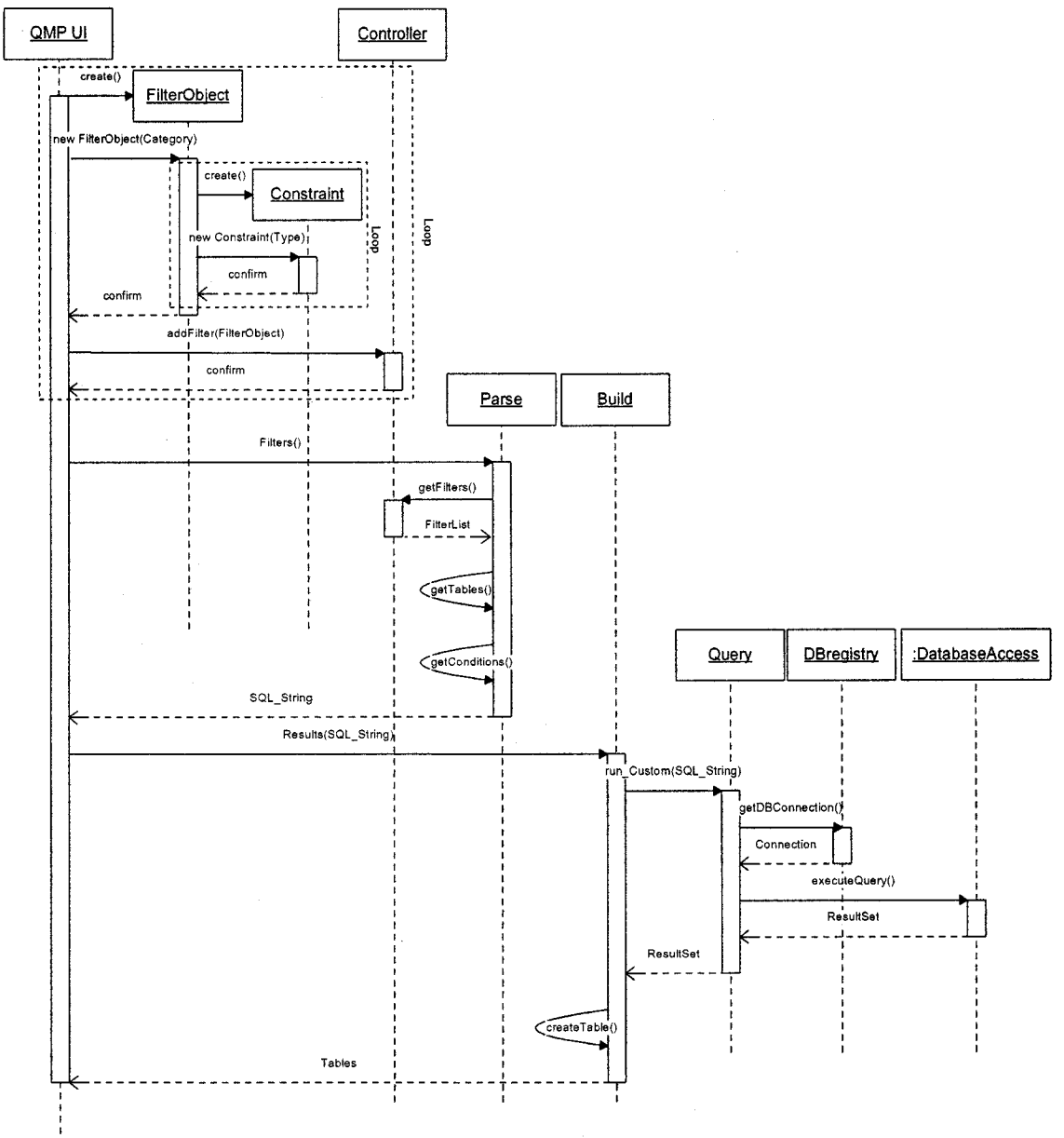


Figure 8.16-Constraint Creation Sequence Diagram

E. PSP Documentation

PSP Report
Brian Leung
PSP0 the base measures effort & defects by type are collected

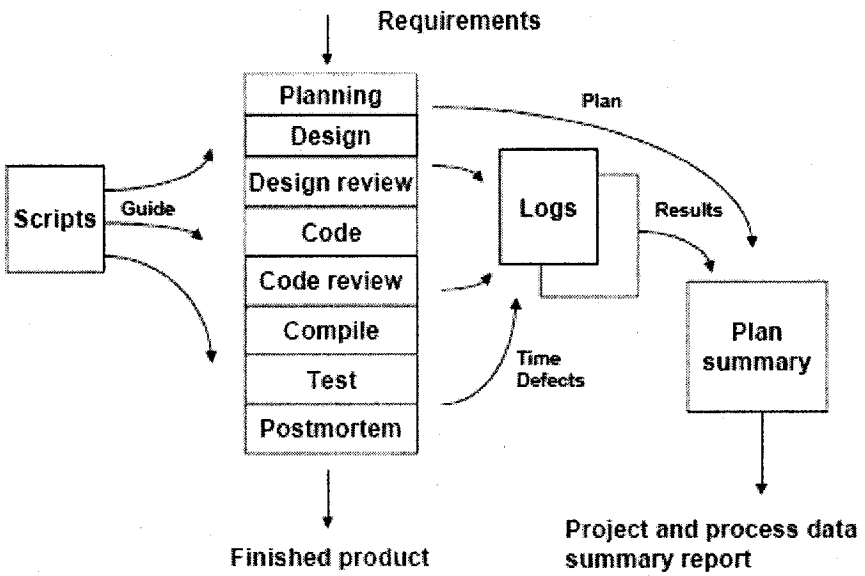


Figure 1: PSP Process Flow

1. **PSP0 Planning Phase Script**:
   Requirements:
   a. EFS: Create a usable User Interface using their current MFS as a metaphor adding functionality that will help the be more efficient in their daily tasks than the current method. Functionalities include table manipulations, access to additional information, graphing capabilities, etc.
   b. QMP: Create a usable User Interface which is intuitive such that users can create and perform queries to retrieve information from a Transplant Database.

2. **PSP0 Development Phase Script**:
   We will analyze the EFS and QMP in portions, documenting the planning, design and coding of each step. The EFS and QMP were planned, designed and coded in the following order:

   i. GUI
   ii. Database, Queries, Connections
   iii. Objects, Object Manipulation Functions, Object Constructors
   iv. Functionalities

Time Log (estimations)

1. EFS (GUI Phase)			QMP (GUI Phase)		
Time in Phase (mins)	**Plan**	**Actual (mins)**	**Time in Phase (hrs)**	**Plan**	**Actual (hrs)**
Planning		30	Planning		3
Design		90	Design		6
Design Review		30	Design Review		4
Code		45	Code		15
Code Review		30	Code Review		10
Compile		10	Compile		5
Test		30	Test		10
Documentation		15	Documentation		2
Total Milestone 2 development time	3hr	4hr 40mins	Total Milestone 2 development time	35	50
2. EFS (Database, Queries, Connections Phase)			QMP (Database, Queries, Connections Phase)		
Time in Phase hrs)	**Plan**	**Actual (hrs)**	**Time in Phase (hrs)**	**Plan**	**Actual (hrs)**
Planning		2	Planning		2
Design		4	Design		10
Design Review		2	Design Review		5
Code		1.5	Code		10
Code Review		3.5	Code Review		7.5
Compile		1	Compile		5
Test		3	Test		15
Documentation		1	Documentation		3
Total Milestone 2 development time	15	18	Total Milestone 2 development time	40	57.5
3. EFS (Objects, Object Manipulation Functions, Object Constructors)			QMP (Objects, Object Manipulation Functions, Object Constructor)		
Time in Phase hrs)	**Plan**	**Actual (hrs)**	**Time in Phase (hrs)**	**Plan**	**Actual (hrs)**
Planning		5	Planning		7
Design		15	Design		17
Design Review		10	Design Review		10
Code		18	Code		15
Code Review		10	Code Review		10
Compile		5	Compile		5
Test		10	Test		15

Documentation		5	Documentation		5
Total Milestone 2 development time	65	78	Total Milestone 2 development time	60	69
4. EFS (Functionalities)			QMP (Functionalities)		
Time in Phase hrs)	**Plan**	**Actual (hrs)**	**Time in Phase (hrs)**	**Plan**	**Actual (hrs)**
Planning		2	Planning		1
Design		12	Design		3
Design Review		5	Design Review		1
Code		10	Code		4
Code Review		8	Code Review		2
Compile		3	Compile		1
Test		10	Test		2
Documentation		3	Documentation		1
Total Milestone 2 development time	35	53	Total Milestone 2 development time	6	15

1. The large differences in the time it took to create the GUI for the EFS and QMP are due to the use of the MFS as a metaphor for the EFS which sped up much of the planning of the GUI. The QMP took a much longer time to design and to create the GUI as different GUI's were looked at and planned for.

2. The large difference for section 2, Database, Queries and Connections can be explained by the fact that Queries for the EFS are static and do not change whereas queries for the QMP need to be created dynamically The QMP took a lot more time to code and to test as different constraints and categories of constraints were considered.

3. For section 4 both the EFS and the QMP took similar amounts of time. The EFS took slightly longer as objects needed more repetition as more table objects are created.

4. Functionalities for the EFS took much longer than the QMP. This is due to the fact that aside from retrieving queries from the database the QMP does not manipulate this data in any way. When manipulation is needed it is done through the EFS interface.

Defects Log (estimations)

1. EFS (GUI Phase)

Defects Injected	Actual
Design	7
Code	21
Compile	5
Test	7
Total # of Injected Defects	40

QMP (GUI Phase)

Defects Injected	Actual
Design	16
Code	21
Compile	2
Test	0
Total # of Injected Defects	39

2. EFS (Database, Queries, Connections Phase)

Defects Injected	Actual
Design	3
Code	14
Compile	6
Test	3
Total # of Injected Defects	31

QMP (Database, Queries, Connections Phase)

Defects Injected	Actual
Design	15
Code	24
Compile	2
Test	0
Total # of Injected Defects	41

3. EFS (Objects, Object Manipulation Functions, Object Constructors)

Defects Injected	Actual
Design	20
Code	49
Compile	5
Test	3
Total # of Injected Defects	77

QMP (Objects, Object Manipulation Functions, Object Constructor)

Defects Injected	Actual
Design	7
Code	20
Compile	7
Test	0
Total # of Injected Defects	34

4. EFS (Functionalities)

Defects Injected	Actual
Design	15
Code	24
Compile	3
Test	3
Total # of Injected Defects	45

QMP (Functionalities)

Defects Injected	Actual
Design	0
Code	0
Compile	0
Test	0
Total # of Injected Defects	0

Defects Removed Log (estimations)

1. EFS (GUI Phase)

Defects Removed	Actual
Design	8
Code	0
Compile	12
Test	14
Total # of Removed Defects	34

QMP (GUI Phase)

Defects Removed	Actual
Design	3
Code	0
Compile	8
Test	23
Total # of Removed Defects	34

2. EFS (Database, Queries, Connections Phase)

Defects Removed	Actual
Design	0
Code	0
Compile	7
Test	22
Total # of Removed Defects	29

QMP (Database, Queries, Connections Phase)

Defects Removed	Actual
Design	0
Code	2
Compile	4
Test	27
Total # of Removed Defects	33

3. EFS (Objects, Object Manipulation Functions, Object Constructors)

Defects Removed	Actual
Design	15
Code	0
Compile	20
Test	34
Total # of Removed Defects	69

QMP (Objects, Object Manipulation Functions, Object Constructor)

Defects Removed	Actual
Design	0
Code	0
Compile	7
Test	18
Total # of Removed Defects	25

4. EFS (Functionalities)

Defects Removed	Actual
Design	0
Code	0
Compile	9
Test	17
Total # of Removed Defects	26

QMP (Functionalities)

Defects Removed	Actual
Design	0
Code	0
Compile	0
Test	0
Total # of Removed Defects	0

Defects infected pertain to something that is wrong with the software. In the case of the EFS and QMP our development environment, Borland JBuilder, reduced the amount of syntax errors significantly as mistakes were identified immediately. Still, though syntax errors were minimized, logical errors still existed through the system and requires us to test and review. Design defects refer to problems of misunderstandings between us and the users we interviewed to get requirements. Most of these defects were resolved quickly through subsequent meetings and discussions. Examples of such defects are Doctors needing to search by patient name whereas we assumed they would be searching by the Patients Identification Number. Another example is the configuration of the Diagnosis Table where the doctors needed a drop-down waterfall selection by category whereas we assumed an entire table would be sufficient.

3. PSP0.1 Size Measurement

PSP Size Measurement	
1. EFS (GUI Phase)	QMP (GUI Phase)
Actual Size: 143, 133, 565, 164, 1242, 293, 334, Total: 2874	**Actual Size:** 137, 581, 312, 866, 505, 369, 179, Total: 2949
2. EFS (Database, Queries, Connections Phase)	QMP (Database, Queries, Connections Phase)
Actual Size: 63, 220, 37, 394, 103, 50, 37, Total: 904	**Actual Size:** 41, 344, 118, Total: 503
3. EFS (Objects, Object Manipulation Functions, Object Constructors)	QMP (Objects, Object Manipulation Functions, Object Constructor)
Actual Size: 800, 870, 185, 1179, 497, 303, Total: 3834	**Actual Size:** 237, 127, 133, 118, 272, 38, Total: 925
4. EFS (Functionalities)	QMP (Functionalities)
Actual Size: 239, 266, 100, 44, 221, Total: 830	**Actual Size:** 0, Total: 0
Total Module Size: **8442 LOC**	Total Module Size: **4377 LOC**

4. Calculations and Discussion

Defect Density: calculates the number of Defects per thousand Lines of Code

EFS: 1000(193/8442) = 22.86

QMP: 1000(114/4377) = 26.04

The defect densities for both the EFS and QMP are low in comparison with other programming projects. This can be explained in several ways.

- The LOC calculated in both the EFS and QMP contain repeating code that is only slightly modified. This can be found in the Table Objects as Lab Tables share a lot of similar and slightly modified code. The Queries within the EFS are static and are large where queries are similar. These portions of similar code increase the number of LOC while having little affect on the number of Defects.
- The LOC calculations also include imported code from outside sources such as Database connectivity classes and Graphing classes which have been modified slightly to fit our needs.
- The numbers of Defects recorded in our logs are estimates and are most likely lower than the actual number of defects found in our prototypes.

Though the defect Density is lower than should be expected, we still believe that it is useable for predicting the number of defects in future similar projects.

Productivity: the measure of output of the amount of Lines of Code per length of time.

EFS: 8442/153.75 = 54.90 LOC/hr

QMP: 4355/191.3 = 22.24 LOC/hr

This is a measure of the productivity for each module we created. We see that the Productivity of the QMP is substantially lower than that of the EFS as a lot more planning was needed than coding.