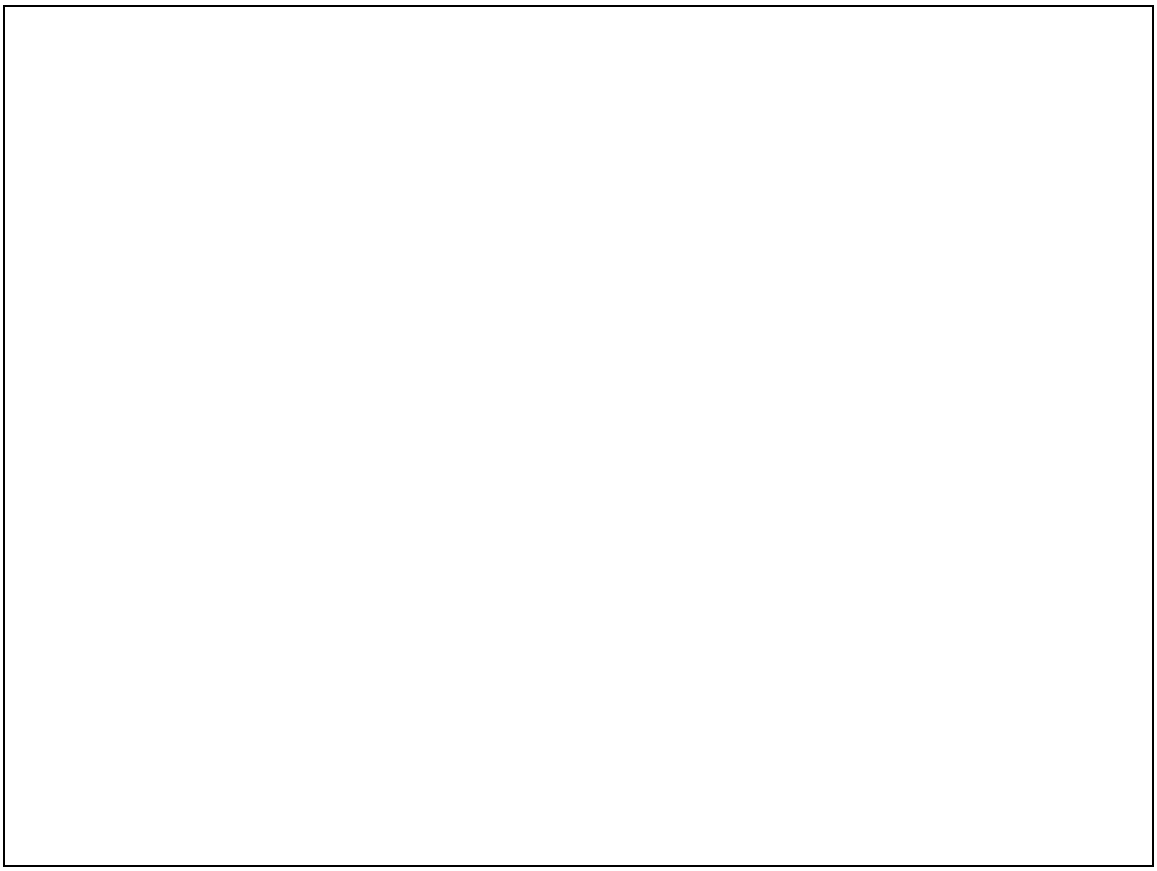
HEADHUNTING THE FUTURE…

Augmenting Information Technology Support Systems at Mandrake Management Consultants Inc.



Assignment 3: Detailed Design of an Information System
CMR Consulting [team 30]
Charissa Lai, Martin Ross, Ruslana Goncharenko
Course: CSC340s
Date: April 16, 2001
TA: Bowen Hui
Presented to: Professor John Mylopoulos

TABLE OF CONTENTS

1. Introduction

Mandrake Management Consultants is an executive search firm that relies heavily on the work of its consultants. In this day and age of modern and competitive executive search, consultants find dire need for proficient and stable infrastructure from IT resources within the company. Since IT is so critical to the consultants' and associates' jobs, it is imperative that an efficient and reliable IT support infrastructure exist. Without this, Mandrake will fall flat behind its more efficient competitors.

1.1. Current System

Reflecting Mandrake's rapid growth, the current IT support process and infrastructure is extremely informal. For example, if a consultant has a problem with his/her computer, he/she uses a variety of methods to inform Tom Metaxas of the problem, whether it is e-mail, ICQ, phone, a direct visit or other. There is no written policy in place or communicated to employees regarding prioritization of IT support. This structure is very time consuming for the IT director, since most of the time he is busy handling unorganized support requests when his time should be spent on network and system administration. Therefore, the current system has serious flaws in both administration and support area.

1.2. Recommended Solution: Web-based IT Support Form

The best alternative for this problem was the support form that will reside on the intranet. Users will fill out their IT requests on a standardized form that has details including the type of the problem (hardware, application related, etc.); difficulty and priority of the problem (can the user continue his/her job for the moment?); and other information that will assist in smooth-lining the IT department's job-resolving.

1.3. Requirements Analysis [1]

1.3.1. Functional Requirements

Functional Requirements involve having the system being able to:

- Authorize and verify the user's accessibility to the system.
- Allow users to reset the form, submit their requests and to receive confirmation of system reception or processing.
- Allow users to check the status of their requests, cancel or modify the submitted ones.
- Allow support staff to view requests, parse information submitted by users, add notes, list pending requests, list all the requests, check what requests are being currently processed, schedule the time to process the selected requests and choose the action with which to proceed.
- Provide detailed information regarding the unique ID assigned to the problem, priority, estimated complexity, problem category, completion date, and all the details about the user and his/her interpretation of the problem.
- Allow support staff to store the processed requests in the database and do relevant general processing.
- Generate monthly or weekly reports based on the data stored in the archived database.

1.3.2. Non-functional Requirements

1.3.2.1. Interface requirements

There will be three interfaces for this information system: one is for users who need technical support, one for technicians and the other for the IT Manager. They both should be user-friendly, well structured and tested with each set of users. Note: the interface for the users should be simpler and more straightforward.

[1] See Appendix (Requirements Details: Functional and Non-Functional Requirements)

1.3.2.2. Performance requirements

It is important to keep the following issues in mind when thinking of the performance requirements (for more details, see the Appendix):

a) *time/space bounds*
b) *reliability*
c) survivability
d) *efficiency:* The capacity of the system should be such that the company network system is able to handle 80-90 terminals (appropriate for the number of employees) with the ability to expand if necessary. The degradation of service may occur when load on the system exceeds its capacity. In terms of timing requirements,
   - *stimulus-response:* e.g. the system will submit a request within a half to five seconds after it was completed by the user
   - *response-response*: e.g. the system will deliver a submitted request to the appropriate category/account based on calculated priority level within one minute.
   - *stimulus-stimulus:* e.g. a user may check for the status of his/her request two minutes after it was submitted.
   - *response-stimulus:* e.g. a user must wait for the technical support after he/she receives confirmation that his/her request is being processed.

1.3.2.3. Operating requirements

Again, it is important to consider the following issues:

a) *physical constraints*
b) *portability*
c) *security*
d) *personnel availability*
e) *skill level considerations*
f) *accessibility for maintenance*
g) *environmental conditions*
h) restart requirements
i) *backup requirements*
j) *fallback requirements*

1.3.2.3.1. Lifecycle requirements

It is important to consider the following issues:

a) *quality of design*
b) *limits on development*

1.3.2.3.2. Platform requirements

It is important to consider the following issues:

a) *memory*
b) *disk space*
c) *operating system*
d) *CPU*
e) *Peripheral*
f) *Network*

2. Global Architecture

2.1. Software Architecture

Basically, we were faced with two major options for the software architecture. We were to either use traditional client-server architecture or to deploy a web based three-tier architecture. Both architectures would store the data on a traditional relational database server.

2.1.1. Client-Server Architecture

2.1.1.1. Server

A separate server program also needs to be written that would contain the business logic and a database access layer. RPC is the most viable method of serving the requests of the client. Since RPC carried over TCP and TCP is responsible for ensuring reliable delivery, it will be very easy to implement this type of connection. Furthermore, RPC is widely supported on Microsoft platforms. The volume of calls is not so much so as to require a dedicated machine, thus Mandrake could use one of the numerous existing servers.

2.1.1.2. Client

This option involves developing a standalone client program and installing it on every PC on the network. A complete user interface would have to be written for each type of client to be supported. Updates to the client would require manually deploying updates to the client. The chief asset for this client would be complete control of the user interface and very low bandwidth usage.

2.1.2. Three-Tier Web Architecture

To be more precise, ”web architecture” involves using various Internet or web technologies/protocols to implement the design. Often, an HTTP server is the primary method of communication between the client and the server. The availability of so many standardized technologies (essentially reusing other developers’ work) is an enormous benefit to any software project.

2.1.2.1. User Interface

Three-tiered web architecture essentially eliminates the need to develop a client. Rather, the standard PC web browser renders the information stream sent from the web server. A very attractive feature of this architecture is that all of Mandrake’s PC's already include a web browser (Internet Explorer 5.5). Finally, various technologies are readily and cheaply available to help quickly build the web site required to support this architecture. They are considered later in this document.

2.1.2.2. Business Logic

The business logic can be separated from the web server if necessary. However, due to the low usage requirements of this project, it is currently not necessary. Instead, some sort of component-based technology can be used to isolate the business logic from the user interface on the web server. A high degree of scalability and maintainability is ensured with this design.

2.1.2.3. Database

This option is not significantly different from the client-server design.

2.1.2.4. Conclusion

Table 1 Decision support table for Software Architecture (Ranked from left to right)

Model	Development Cost	Scalability	Upgrade Deployment Time	Bandwidth Usage
Client-Server	High	Medium	High	Low
Three-Tier Web	Low	High	Low	Medium

The Three-Tiered web architecture is the most appropriate based on the given criteria.

2.2. Computer Network

The backbone of the existing computer network architecture consists of 10/100 Mbit switched Category 5 Ethernet. Separate Windows NT 4.0 servers provide Domain, DNS, File, Database, and Print services for the internal network. A Windows NT Checkpoint Firewall isolates the external and internal networks from each other. It also is responsible for the Network Address Translation of internal private IP addresses to the available pool of external addresses. Another Windows NT box is responsible for external Web and DNS services for the Mandrake namespace domain. The connection to the public Internet connection is a dedicated T1 line connected to a CISCO 1200 router.

The existing network design is well engineered and more than adequate for the requirements of the purposes of this project. The bandwidth requirements of 100 Mbit Ethernet are more than adequate to handle the expected number of requests (<100 Per Day).

2.3. Hardware

Since we intend to use thin clients (web browsers) and hence a client-server architecture, the major hardware decisions revolve around choosing the servers. Essentially there are at most three server machines needed to fulfill the requirements: a business logic server, a database server, and a web server. The three considered options for a hardware platform are a Wintel machine (either a brand name or a clone), Macintosh G4, or a dedicated Unix machine from some proprietary vendor (Solaris, DEC Alpha, or SGI Indigo box). The ranked assessment criteria for all three functions were:

1. Ease of integration with the existing hardware infrastructure
2. Compatibility with our software requirements
3. High price/performance ratio
4. Reliability

We choose to go with a Wintel box for all three boxes since it was clearly superior to the other two options in criteria 1,2,3 and closely competitive within our reliability criteria (since our requirements specification only dictated a 99% reliability). Since we chose a Wintel box, it seemed logical to examine the possibility of using the existing infrastructure instead of spending money for purchasing new servers.

In all three cases, we determined that we could simply use the existing hardware resources since the memory and CPU requirements of this project are very modest compared to the available resources in the existing boxes. In particular, we rolled the functions of the business logic and web server into the existing web server box and simply allotted the database server role to the existing database server. Table 2 shows the estimated requirements versus the available resources on the existing boxes:

Table 2 Hardware-Role Requirements Analysis

Machine	Role	CPU	Memory	Disk	Required CPU	Required Memory	Required Disk
Web Server	Business Logic Web	Pentium III 500 MHz	128 Megs	10-Gig IDE	< 5%	< 15%	< 5%
Database Server	Database	Pentium III 450 MHz	512 Megs	Six 20-Gig Ultra-SCSI 2 RAID 5	< 2%	< 10%	< 1%

2.3.1.1. Conclusion

Based on our primary decision criterion of cost along with the recommendations given for the software architecture (See 2.4 Software) and network architecture it makes great sense to use the existing Intel hardware.

2.4. Software

There are three software components for which we need to choose an implementation:

1. Operating System
2. Web Server
3. Database

2.4.1. Operating System

Generally speaking, the choice is between some variant of Unix and Microsoft Windows for the server components. It should be noted that the choice of operating system affects the choice of web server and database and vice versa. Thus, this category is not really chosen in isolation. The recommendations for all three components will be suggested after presenting the alternatives. Linux is clearly the best variant of Unix to use with the chosen hardware (i.e. a standard PC) so Linux will be used for the comparison.

2.4.1.1. Linux

Linux is an operating system developed using an open source model. Although different distributions exist (RedHat, Slackware, etc.) the central core of the operating system is standardized and free. Linux is very stable, scales very well, leverages 20 years of Unix software developed, and is basically free. Unfortunately, Linux requires a highly level of training and experience compared to a Windows box despite recent claims to the contrary. In Mandrake's case there is a major problem since no trained Linux administrator is available. At the Mandrake premises there is no existing Linux machine, making the purchase of a new server necessary.

2.4.1.2. Microsoft Windows 2000 Server

Microsoft Windows 2000 Server is the newest edition of the Microsoft Windows NT kernel based software. It includes many different types of server components including Web, FTP, DNS, DHCP, File, and Print services. It includes graphical configuration tools for all of the various services. A considerable advantage from this is that the existing IT support staff at Mandrake would not require extensive retraining. Furthermore, Microsoft provides technical support (on a per incident basis) to resolve any problems. Although the license fees for a Windows 2000 server are expensive, Mandrake already purchased numerous licenses for its existing servers. The expected load is small enough to make it possible to simply utilize the existing machines.

2.4.2. Web Server

The two realistic choices are tied to their respective host platforms. The Unix Apache web server ships with Linux and runs the greatest number of web sites on the Internet. Microsoft's Internet Information Server 5.0 (IIS) ships with Microsoft Windows 2000 Server. Both servers offer the ability to serve simple HTML pages via HTTP and provide frameworks for component based software development.

2.4.2.1. Apache 1.3

Apache is a free software package developed over the past decade. It is very powerful since it has good scalability, reliability, and customizability. Moreover, since the source code is available, the security community continuously audits the source code for security leaks. Unfortunately, the configuration and administration of Apache can be very complex. Many different types of component technologies exist for Apache and Unix. However, the most common and feasible approach uses server side Enterprise Java Beans (EJB) glued together with server side JavaScript pages (.jsp files). Both these technologies are available to be licensed for free from Sun Microsystems. As a bonus, Java Integrated Development Environment (IDE) tools are readily available from numerous vendors.

2.4.2.2. IIS 5.0

IIS 5.0 also serves plain HTML web pages. Its scalability, reliability and customizability are considered inferior to Apache's. Furthermore, the lack of source code makes assessing its security very difficult. However, IIS 5.0 is very easy to configure and maintain compared to Apache. Microsoft software component technologies are grouped under

the ill defined label COM and is normally bound together on the web page side using Active Server Pages (ASP). Unlike, Enterprise Java Beans (EJB) COM is at least somewhat language independent. In practice, the actual components are written either in Visual Basic or C++, while the scripting is done with VBScript or JavaScript.[2] Microsoft sells the Visual Studio IDE to develop these components.

2.4.3. Database

Once again we examine the leading contenders between Windows and Unix in the field of databases. Although there are many alternatives under both platforms, we chose two commercial database products to evaluate: Microsoft SQL Server 2000 for Windows and Oracle 8i for Linux. Although there exist open source alternatives for both platforms, these have not gained the level of acceptance that these two databases have.

2.4.3.1. Oracle 8i Standard Edition

Oracle databases are considered the most reliable, scalable, and powerful databases for anything short of the largest computers. The standard edition includes the basic relational SQL[3] database access along with the appropriate EJB wrappers necessary for interaction with the software. As mentioned above, a separate machine will be required since no Linux machine exists at the premises. Approximate cost for the database is about $12 000[4] CDN. This is necessary since Mandrake does not have any Oracle licenses available. Furthermore, the existing Mandrake IT Support Staff will need to be retrained to operate and maintain the Oracle Database.

2.4.3.2. SQL Server 2000 Edition

Microsoft SQL Server is not as mature as Oracle. However, for the scope and purposes of this project it is well within the specified parameters. Furthermore, there are considerable advantages with this software. Mandrake already owns a hardware box and license for SQL Server. The IT Support Staff is thoroughly trained in its use and operation. Thus the cost is negligible for this solution. Microsoft includes components and wrappers with its Visual Studio products to access SQL Server. Specifically, the Advanced Data Objects (ADO) interface is used for the Active Service Pages and the COM components.

2.4.4. Conclusion

The decision criteria are ranked in Table 3 from the most important on the left to least important on the right. The clear winner is the Windows, IIS 5.0, and SQL server 2000 solution.

Table 3 Decision criteria for Software selection

Solution	Overall Cost	Deployment Time	Ease Of Use	Reliability	Scalability
Linux, Oracle, Apache	High	High	Poor	Excellent	Excellent
Windows, IIS, SQL Server	Low	Low	Good	Good	Good

2.5. Software Components

We choose to use UML to represent various sub-systems and modules in the system. Please see Figure 1 for a graphical view. Since we have chosen the three-tiered web architecture, the system was broken down into three major subsystems:

1) Business Objects (the security package is enclosed in this subsystem)
2) Database Package
3) User Interface Package

Please note that while the class diagrams for each of these subsystems are shown here, **please refer to the appendices for the state/activity and sequence diagrams embedded in the data dictionary.**

[2] Please note that this version of server side JavaScript is different from the Apache's .jsp files.
[3] Structured Query Language
[4] See Appendix 2 for a quick calculation from Oracle.com's online store.

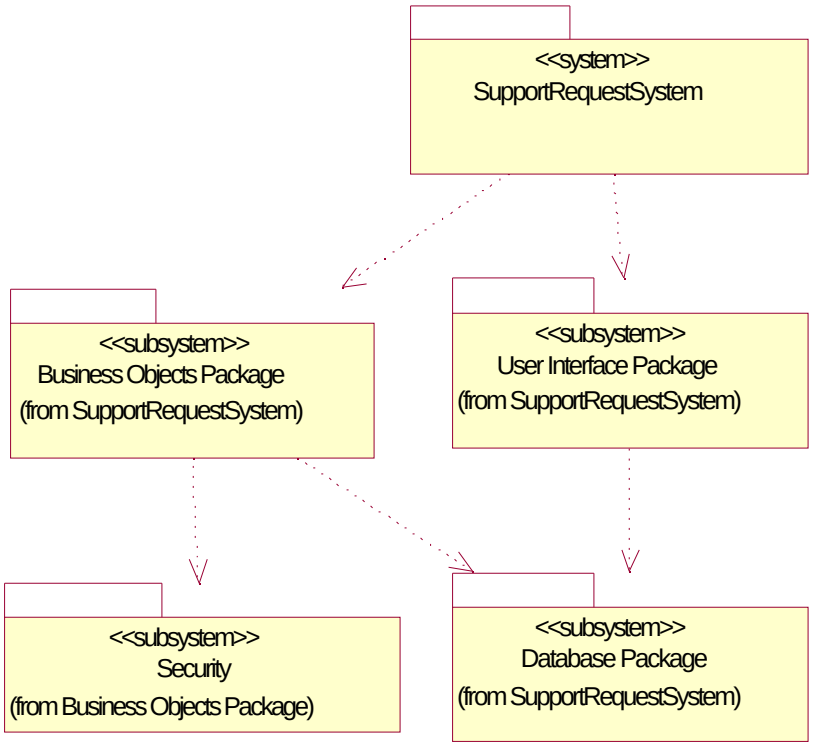


Figure 1 Subsystems UML Diagram

2.5.1. Business Objects

The business objects are the heart of the system. The most important class is the SupportRequestCore that separates the database layers from the rest of the system. The three types of users all call the operations in SupportRequestCore that deal with adding, deleting, and listing different types of requests. The three types of users are represented with three classes UserInterface, ITSupportPersonInterface, and ITManagerInterface. The operations for each of these classes correspond nearly one-to-one with the use cases - thus justifying that the program design meets its requirements. The chooseAction() method is called whenever a person submits a request and the system has to determine what action it needs to use. The mailer class gives mailing services so that the classes can send confirmation to users when appropriate. Finally, the SupportRequest, SupportRequestList and SupportRequestEntryLog model the support requests entered by users. SupportRequestEntryLog's are added by the IT Support people to describe any notes or progress on a particular issue.

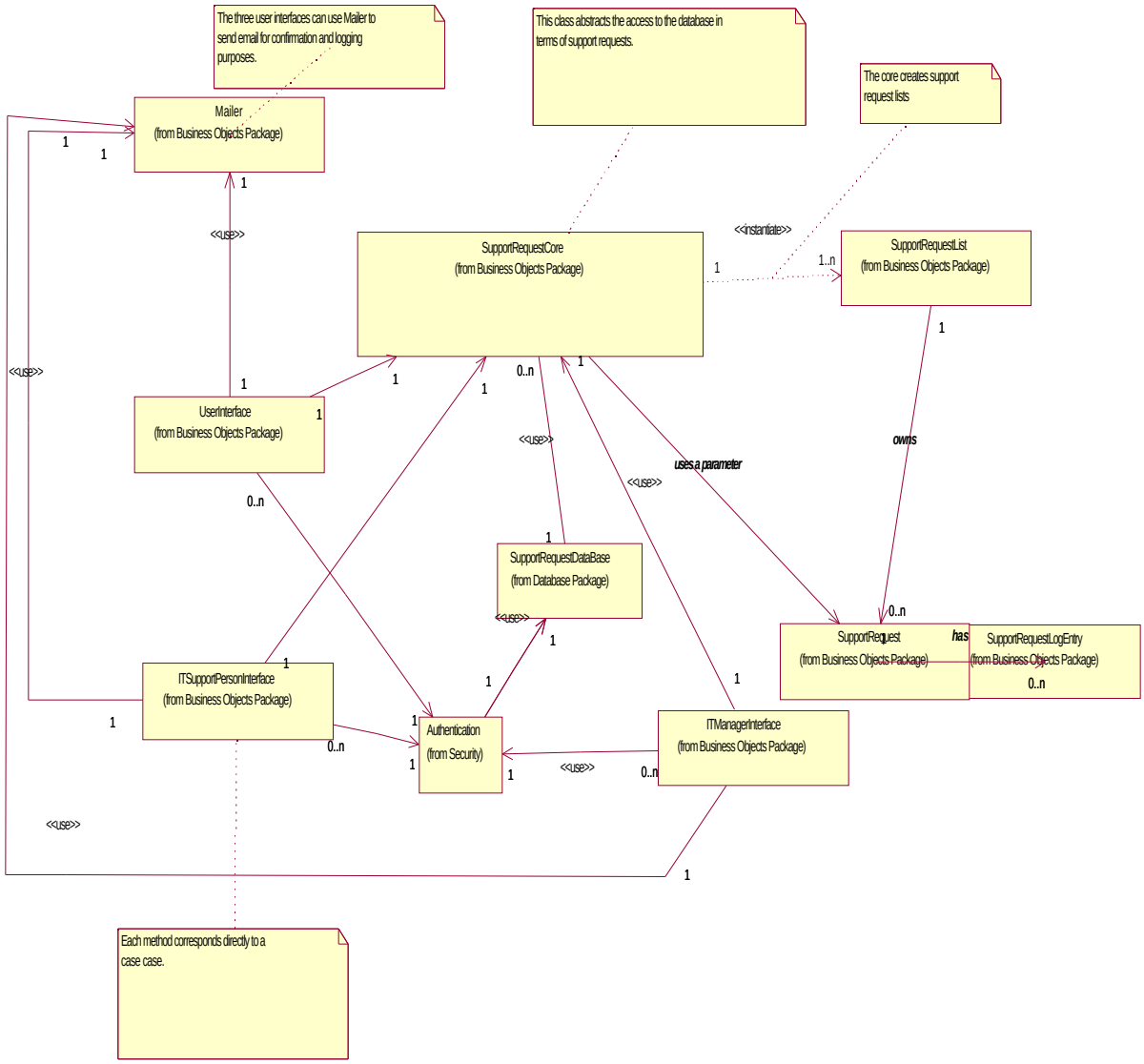


Figure 2 Business Logic Class Diagram

2.5.1.1. Security Subsystem

The security subsystem is responsible for authenticating the various classes of users. The three main classes (ITSupportPersonInterface, ITMangerInterface, and UserInterface) call the Authentication class using an instance ActiveUser that is created when the user session is established (by createInterface()). This authentication class calls the SupportRequestDatabase to perform a lookup based on a username and password to determine if access can be granted.

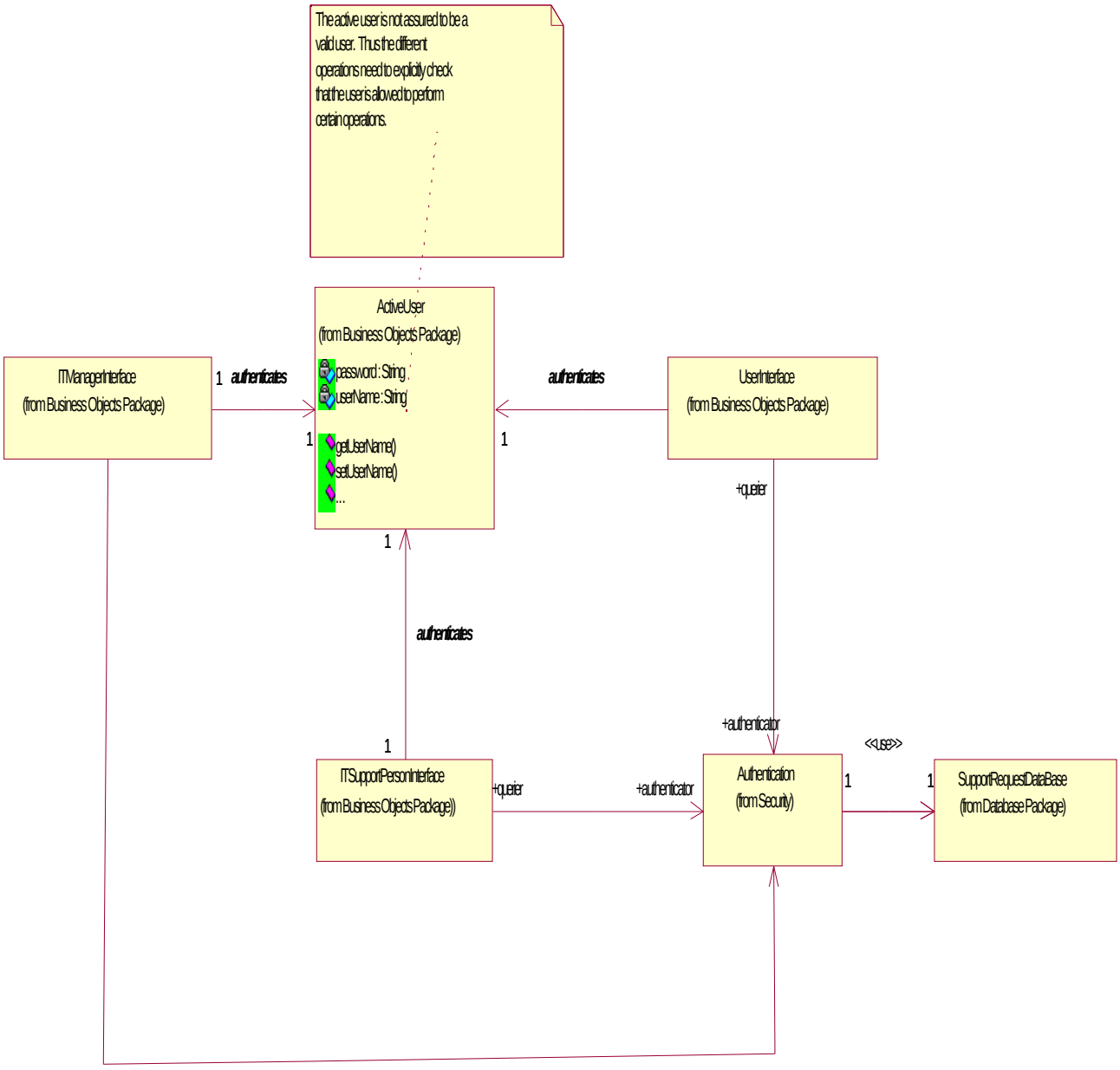


Figure 3 Security Class Diagram

2.5.2. Database Package

The database subsystem has the simplest UML representation of all the major systems. However, since the implementation uses Microsoft technologies (specifically ADO), there are many implementation details that need to be addressed. The most important is the issue of the ADORecordset class that is not defined by this document. ADORecordset is the standard Microsoft designation for a record set for either of the two most popular languages for

writing COM components C++ and VB. Interestingly, the calling classes of SupportRequestDatabase also must have an awareness of the ADORecordset.

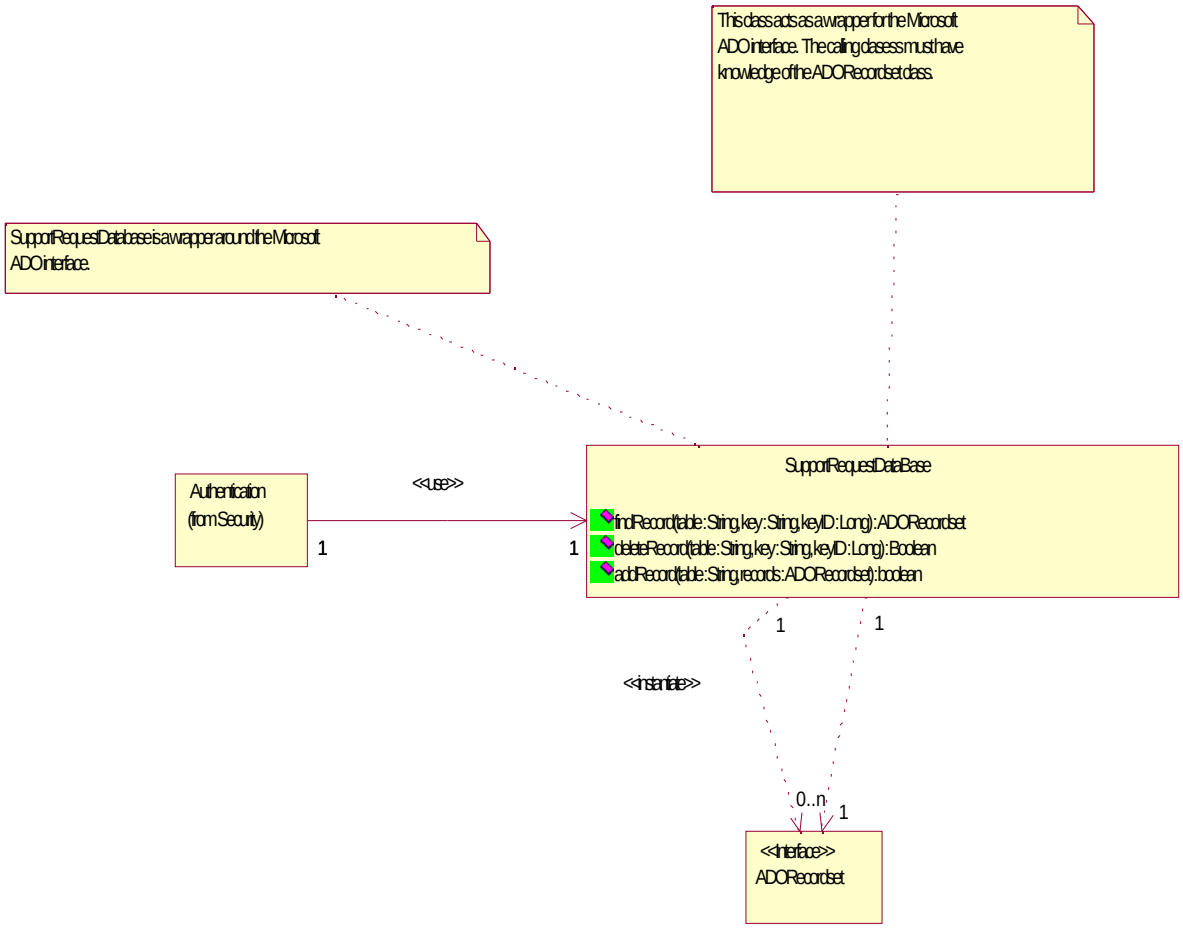


Figure 4 Database class diagram

2.5.3. User Interface Package

The UserInterface package is responsible for dealing with spawning connections and dealing with input and output. When the IIS server receives a connection for the first time is creates the COM component and the ASP script passes control to the component. The dispatcher object waits for connections and creates UserInterface derived (ITManagerInterface, UserInterface, ITSupportPersonInterface) classes to handle each of the requests based on the type of page accessed. IIS takes care of maintaining a virtual session since HTTP is actually a stateless protocol. For security purposes, authentication is performed at the start of every call. The parseForm() method is responsible for decoding the information from a web form into a more suitable format. Please note that the three main class' methods (ITManagerInterface, UserInterface, ITSupportPersonInterface) are the actual methods that return the input and output to the user. Although, this design does not separate business logic and interface as cleanly as possible, in practice, web based development makes such a separation infeasible.

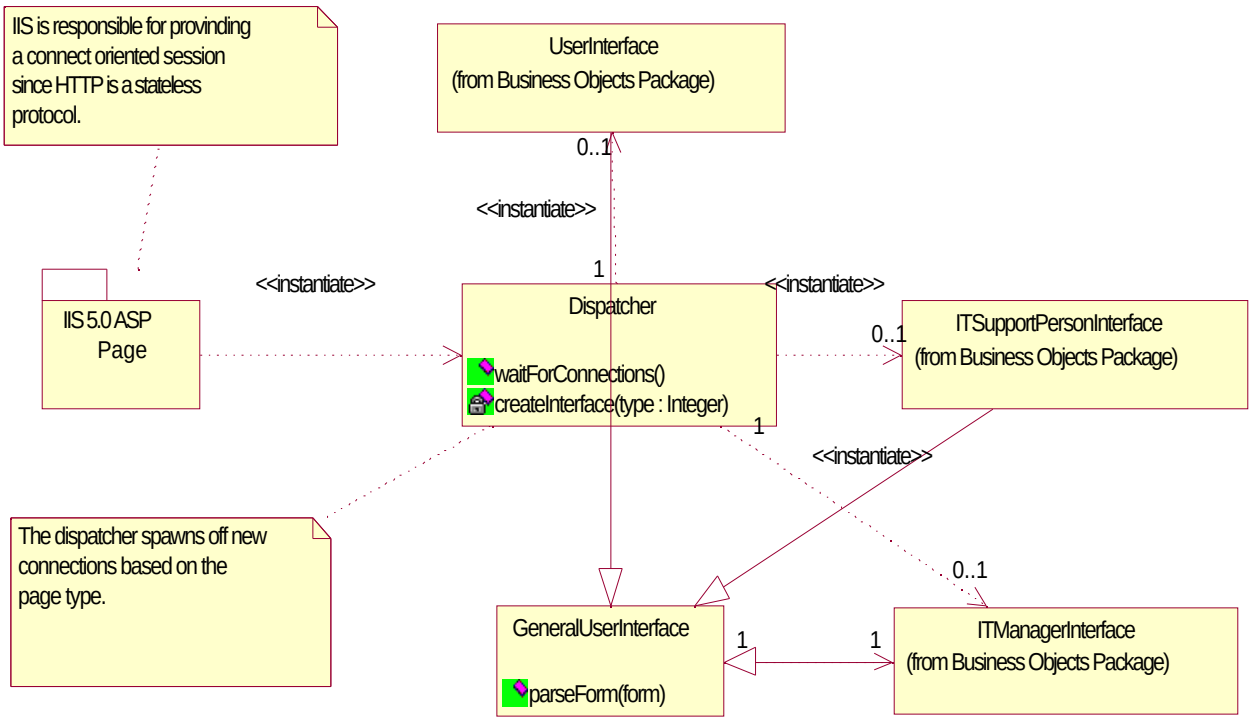


Figure 5 User Interface class diagram

---

3. Database Diagrams

3.1. Database Conceptual Class Diagram

The following is the class diagram that describes all data to be stored in the database.

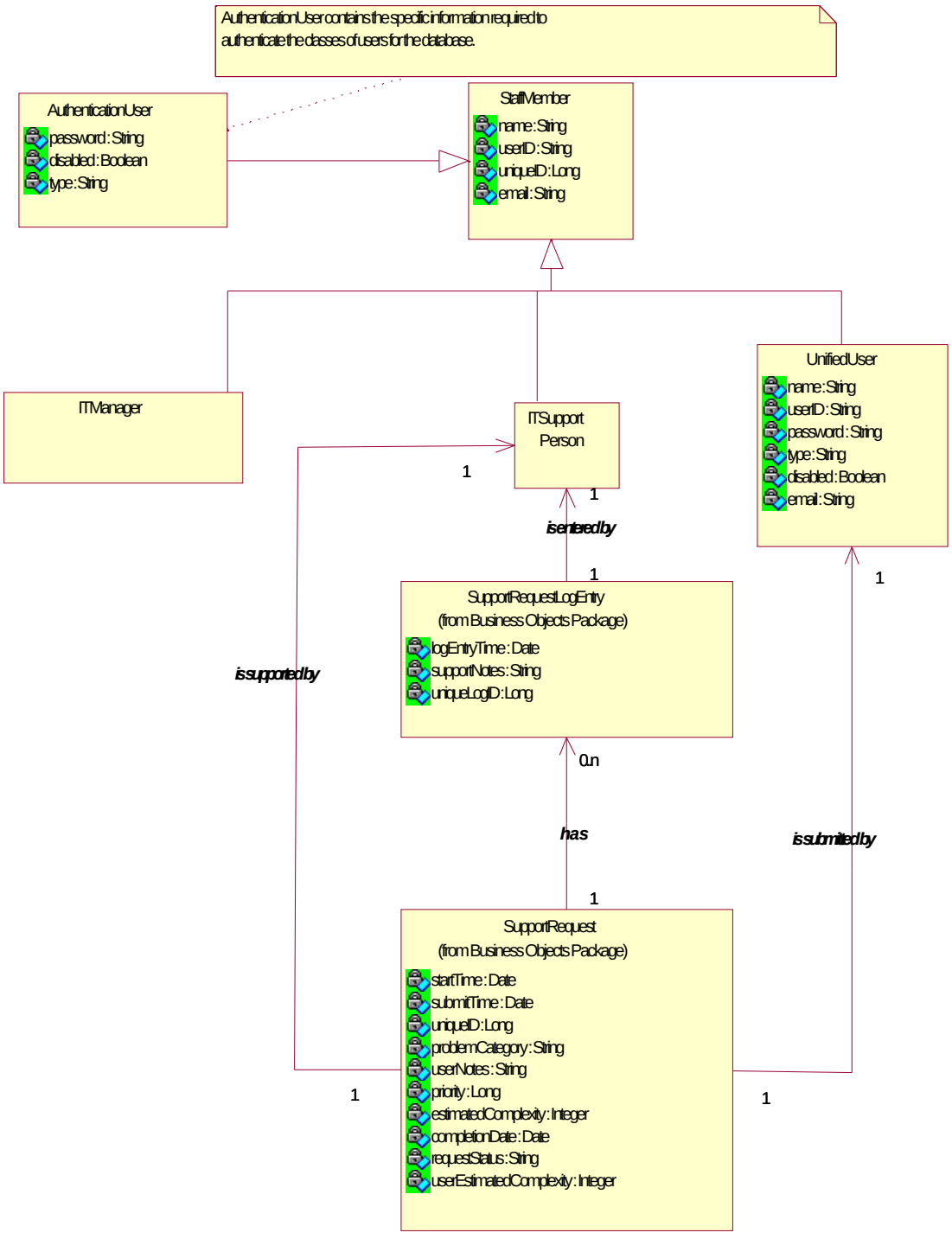


Figure 6 The conceptual class diagram

We selected the 10 most common operations to be performed on our database.

3.2. Typical operations:

1. **Operation 1.** Add support request
2. **Operation 2.** Check the status of the support request
3. **Operation 3.** Change the status of the support request
4. **Operation 4.** Verify the user's identity (authentication)
5. **Operation 5.** List all pending requests
6. **Operation 6**. List all requests currently processing
7. **Operation 7.** List all completed requests
8. **Operation 8.** List all the requests submitted by a particular user
9. **Operation 9.** Add an entry log for a request
10. **Operation 10**. Modify the entry log for a request

3.3. Workload Data

3.3.1. Table of volumes

Concept	Type	Volume
StaffMember	Interface	
User	Class	80
ITManager	Class	1
ITSupportPerson	Class	5
AuthenticationUser	Class	86
SupportRequest *	Class	20
SupportRequestLogEntry **	Class	100 000

* per day
** the number of SupportRequestLogEntries defines the capacity of the DB

3.3.2. Table of operations

Note: Based on interview with Tom Metaxas [5] we assumed that there would be 20 support requests per day on average.

Operation	Type	Frequency
Operation 1	I	20 per day
Operation 2	I	40 per day
Operation 3	I	20 per day
Operation 4	B	20 per day
Operation 5	I	40 per day
Operation 6	I	10 per day
Operation 7	I	10 per day
Operation 8	I	20 per day
Operation 9	I	20 per day
Operation 10	I	20 per day

3.4. Redundancies:

In our class diagram we found the following redundancies:

[5] See Appendix (Supporting Documentation)

1. Class AuthenticationUser and interface StaffMember can be merged into one class.
2. The logEntryTime attribute in SupportRequestLogEntry can be calculated from the startTime attribute in SupportRequest class.
3. CompletionDate in SupportRequest class can be calculated from startTime attribute in SupportRequest class.
4. UserID and uniqueID attributes in StaffMember can be merged.
5. Generalization: ITManager, ITSupportPerson and User classes share common attributes such as name, userID, uniqueID, email and do not have any unique attributes.

3.5. Tables of Accesses

These tables evaluate the cost of each operation, using the table of volumes and the navigation schema.

Operation 1

with redundancies

Concept	Type	Accesses	Type
User	Class	1	R
ITSupportPerson	Class	1	R
SupportRequest	Class	1	W

without redundancies

Concept	Type	Accesses	Type
UnifiedUser	Class	2	R
SupportRequest	Class	1	W

Operation 2

with redundancies

Concept	Type	Accesses	Type
User	Class	1	R
SupportRequest	Class	1	R

without redundancies

Concept	Type	Accesses	Type
UnifiedUser	Class	1	R
SupportRequest	Class	1	R

Operation 3

with redundancies

Concept	Type	Accesses	Type
User	Class	1	R
SupportRequest	Class	1	RW

without redundancies

Concept	Type	Accesses	Type
UnifiedUser	Class	1	R
SupportRequest	Class	1	RW

Operation 4

with redundancies

Concept	Type	Accesses	Type
AuthenticationUser	Class	1	R

without redundancies

Concept	Type	Accesses	Type
UnifiedUser	Class	1	R

Operation 5

with redundancies

Concept	Type	Accesses	Type
SupportRequest	Class	1	R

without redundancies

Concept	Type	Accesses	Type
SupportRequest	Class	1	R

Operation 6

with redundancies

Concept	Type	Accesses	Type
SupportRequest	Class	1	R

without redundancies

Concept	Type	Accesses	Type
SupportRequest	Class	1	R

Operation 7

with redundancies

Concept	Type	Accesses	Type
SupportRequest	Class	1	R

without redundancies

Concept	Type	Accesses	Type
SupportRequest	Class	1	R

Operation 8

with redundancies

Concept	Type	Accesses	Type
User	Class	1	R
SupportRequest	Class	1	R

without redundancies

Concept	Type	Accesses	Type
UnifiedUser	Class	1	R
SupportRequest	Class	1	R

Operation 9

with redundancies

Concept	Type	Accesses	Type
SupportRequest	Class	1	R
SupportRequestLogEntry	Class	1	W

without redundancies

Concept	Type	Accesses	Type
SupportRequest	Class	1	R
SupportRequestLogEntry	Class	1	W

Operation 10

with redundancies

Concept	Type	Accesses	Type
SupportRequest	Class	1	R
SupportRequestLogEntry	Class	1	RW

without redundancies

Concept	Type	Accesses	Type
SupportRequest	Class	1	R
SupportRequestLogEntry	Class	1	RW

3.6. Cost Comparison

Now for each operation, with redundancies and without, we estimate its cost in terms of accesses. For detailed calculations, see the Appendix [6]. There are differences in cost only for operations 1, 2, 3, and 4.

3.7. Deciding About Redundancies

Although some of the eliminated redundancies increase the access time (for operations 1, 2, 3, and 4 in particular), there are major advantages to our proposed restructuring. First, we decrease the complexity of the code and structure by collapsing 4 tables into a single UnifiedUser table. Secondly, this has the positive effect of the decreasing the total storage space used since we do not have to store duplicate copies of the authentication table. Thirdly, update times are reduced since we only have to update one record. Finally, scalability is increased since the addition of a type field allows the addition of new classes of users if necessary.

Therefore, the following redundancies can be considered as disadvantages and will be removed from the revised class diagram:

1. Class AuthenticationUser and interface StaffMember can be merged into one class.
2. UserID and uniqueID attributes in StaffMember can be merged.
3. Generalization: ITManager, ITSupportPerson and User classes share common attributes such as name, userID, uniqueID, email and do not have any unique attributes.

The following redundancies can be accepted as advantages since they are not used in most common operations, do not occupy much space, but are very useful for the instance accesses:

1. The logEntryTime attribute in SupportRequestLogEntry can be calculated from the startTime attribute in SupportRequest class.
2. CompletionDate in SupportRequest class can be calculated from the startTime attribute in SupportRequest class.

[6] See Appendix (Database Diagrams)

3.8. Removing Generalizations

The relational model does not allow the direct representation of generalizations in the class diagram. We need, therefore, to transform these constructs into other classes that are easier to translate.

For generalization on Staff Members:

- The relevant operations make no distinctions between the child classes and these classes have no specific attributes;
- We can therefore delete child classes (User, ITManager, ITSupportPerson) and add an attribute Type to the parent class.

3.9. Partitioning and Merging of Concepts

- The attributes uniqueID and userID in the User class can be merged
- Five classes User, ITManager, ITSupportPerson, AuthenticationUser and StaffMember can be merged into class UnifiedUser with the union of all their attributes.

3.10. Selecting Primary Identifiers

UnifiedUser class:

- There are three identifiers that can serve as primary identifiers: Name, Email and UserID;
- It makes more sense to select *UserID* as a primary identifier since it is guaranteed to be a unique non-null field; it is also shorter than the two other potential identifiers.

SupportRequestLogEntry:

- There are no good candidates for the primary identifier for this class, hence, we will introduce a new identifier *UniqueLogID*.

SupportRequest:

- We already foresaw a problem with the primary identifier for this class and thus, provided a *UniqueID* attribute, which will serve as a primary identifier.

3.11. Transformation into Relational Model

```
UnifiedUser (UserID, Name, Type, Email, Password, Disabled)

SupportRequestLogEntry (UniqueLogID, ITPersonName, SupportNotes, LogEntryTime)

SupportRequest (UniqueID, StartTime, SubmitTime, UserName, ProblemCategory,
         UserNotes, Priority, EstimatedComplexity, CompletionDate,
         RequestStatus, UserEstimatedComplexity, ITPersonAssigned)
```

3.12. Normalization

Now we normalize our relational model:

In First Normal Form (1NF):

Our relational model is already in 1NF since each relation does not include any multi-valued attributes or any composite attributes.

In Second Normal Form (2NF):

A relation is in 2NF if it is in 1NF and all non-key attributes depend on all elements of its key rather than a subset.

Our model is already in 2NF since for each relationship the following condition holds:

- There is only one primary key
- All the non-key attributes depend on this key

In Third Normal Form (3NF):

A relation is in 3NF if it is in 2NF and none of it non-key attributed depends on any other non-key attribute.

The following two relations are already in 3NF

```
UnifiedUser (UserID, Name, Type, Email, Password, Disabled)
SupportRequestLogEntry (UniqueLogID, ITPersonName, SupportNotes, LogEntryTime)
```

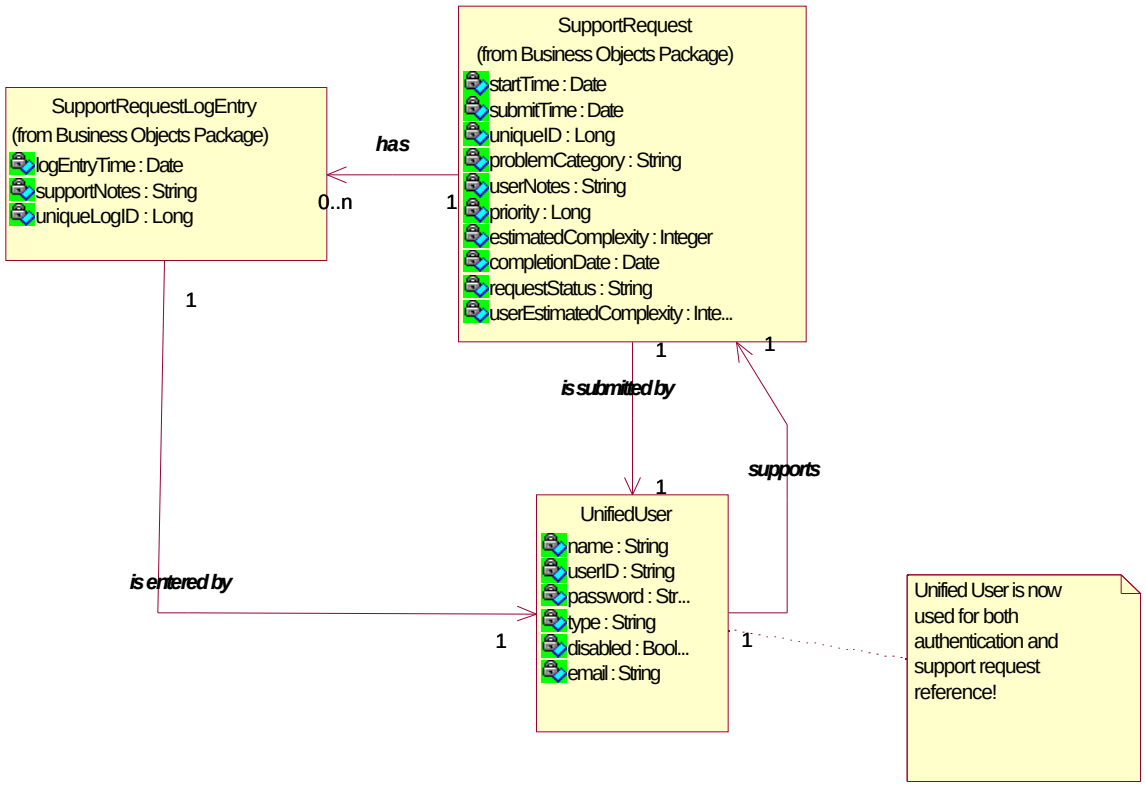
Now, we consider the last relation:

```
SupportRequest (UniqueID, UserName, ProblemCategory, UserNotes, Priority,
EstimatedComplexity, RequestStatus, UserEstimatedComplexity, ITPersonAssigned)

Timing (SubmitTime, StartTime, CompletionDate)
```

Therefore, our relational model is fully normalized.

3.13. Class Diagram After Restructuring



4. User Interface Design

4.1. Description of User Groups

In this system, there will be three main user groups. These are the IT Support Personnel, the IT Manager and the general staff at Mandrake who use computers (termed Users/General Staff). For more details on what each user group can do, see 'Justification that the Interface Design Meets Relevant Requirements' in the forthcoming section.

General users at Mandrake will be able to access the local intranet and use the form provided to submit their IT support requests. Feedback will take place in the form of an email confirmation when a user has submitted their support request. General users have the least amount of accessibility and authority in terms of changing elements with the system.

IT Support Personnel are responsible for providing the technical support to Mandrake's users. They take all requested support jobs and try to resolve them. They have more accessibility and authority than general users and IT Manager. For example, they have the ability to reclassify a problem into a lower priority level and can delete requests.

The IT Manager is responsible for managing the IT Support Personnel. He/She will look at reports that will be generated daily/weekly/monthly. By doing so, he/she can keep track of how the system is working – are there any particular jobs in a special area that are being requested more than others (for example, website failure)? If this is the case, then he/she will be responsible for resolving the problem - perhaps by improving the connection providing the website to the company computers. The IT Manager has less accessibility and authority than the Support Personnel since he/she will only be interested in general aspects of how the system is running. For example, he/she will be responsible for recommending strategies on allocating resources (such as time) more efficiently after reviewing statistical data output from the database. They, in contrast with Support Staff, cannot delete requests or anything of the likes.

The interface for general computer users will reside on the local intranet. When a general computer user experiences a problem with IT, support will be requested through the web-based form (hence, a web-based interface). When the form with the problem description is submitted, the information is then stored into the database. IT support staff will then look at the information in the database and from there, can modify, delete or do anything he/she wishes. Thus, a friendly interface to the database is required. In addition, the IT Manager is allowed access to the database. He/she will be able to click on certain options that allow generation of weekly/monthly reports and these operations require a similar interface to that of the IT Support Staff – a friendly interface to the database. However, their interface will be in whole part easier to understand and navigate.

4.2. State Diagrams Describing the Dialogues Supported by the Interface [7]

The dialogue mode that seemed to be most appropriate for this system is the Graphic-based Dialogue Structure. This is true since all user interfaces will be heavily dependent on mouse and monitor capabilities. There will be user menus that allow options to be chosen and the system will carry out those actions, updating the database accordingly, displaying the next menu, and so forth.

Note that all users will be required to 'login' or authenticate themselves. This assists in the determination of who gets more accessibility and authority in terms of changing elements of the system.

Two of the dialogues composed are the Dialogue Structure for the IT Support Personnel Interface and the Dialogue Structure for the General User. These two main groups of people are the ones who drive the input since they fill and modify the database with support requests, respectively. The last dialogue is the Dialogue Structure for the IT Manager. He/She is the drive behind the output since he/she is responsible for viewing the weekly/monthly-generated reports.

[7] See Appendix (Dialogue Structures for User Groups)

4.2.1. General User Dialogue Structure

After authentication/login, the General User has the option to make a new support request when an IT problem occurs, or he/she can view the list of submitted requests. In making a new request, there is confirmation that the system received the request. In listing the submitted requests, a general user has the option to choose a particular request and cancel, modify or just view the status of that particular request. On cancellation and modification, there is a system confirmation/feedback notifying the user that there has indeed been a cancellation and modification recorded. Once a user has finished with his/her query, he/she can conduct more queries. Otherwise, he/she can just leave the web-based system.

4.2.2. IT Support Staff Dialogue Structure

After authentication, there are options for the Staff to list all requests in the database, list the pending requests (those that have not started), list the current requests (those that have begun), or to search for a particular request. For each item in each of these lists, there is in turn the option to view it or to delete it from the database. Once the query has been accomplished, there is again the option of conducting more queries. A particular point to note is the search request: if details have been given but the database does not include the searched request, then there is an announcement to the user that the request could not be found and that the user should try again. After all queries are complete, the IT Support staff can exit the database.

4.2.3. IT Manager Dialogue Structure

After authentication, the IT Manager will be able to access information from the database system pertaining to statistics that have been gathered - problems that IT Support Staff have experienced, financial information and the history of allocations and resources that the IT Department has invested or has yet to invest. After a query has completed, the Manager can choose to either quit the system or continue conducting more queries into the system.

4.3. Mock-ups of Windows [8]

By no means are these prototypes exhaustive, but simply 'a taste' of what the real system interfaces may look like. Each main input/output session has its own dialogue structure.

The *Authentication Window* greets all users of the system. It determines whether the user is a valid user to the company and if so, the user level then determines how much accessibility and authority the user enjoys.

After authentication, a user can then enter one of three windows: *General User Introduction, IT Support Staff Introduction* or *IT Manager Introduction.*

From a *General User Introduction*, one can choose to either make a *New Support Request* or *List All Submitted Requests*.

A *New Support Request* contains all the details that a user must enter if he/ she wishes to make a query to the IT Support Staff. A 'Name' and 'Description' are mandatory. In this page, the 'Notes' section would be grayed out/ blocked for the General User. The reason for this is that it would be sensible to have the 'Notes' added by IT Support Staff, but only viewable for informative purposes by the General User. The General User then submits, cancels or resets the support request form.

The *List Submitted Requests* page allows users to view all of his/her submitted requests. From this list, he/she has the option of checking the request's status, cancelling it, or modifying it.

From picking one of the requests, the *Check Request Status* page lets users view the status of the request. The status can be*: Pending* (job has not been begun by IT Support Staff), *In Progress* (job has begun, but is not complete) or *Done* (job has been fixed).

[8] See Appendix (Mock-ups of Windows)

After picking a request, the user can also use the Cancel Submitted Request and Modify Submitted Request pages to cancel or modify the request, respectively. These pages will allow the user to see any confirmations that the system has indeed processed the information.

An IT Support Staff member can do one of four things in the *IT Support Staff Introduction*: *List All Requests, List Pending Requests, List Current Requests or Search for a Request.*

In *List All Requests*, an IT Support member may view all the requests that have been submitted by users. It also lists who (if anyone) has begun working on it. From here, a member may choose a particular request and View and Modify Request or Delete Request.

The *View and Modify Request* page allows the IT Support Staff to view the full details of a user's request and if he/she so desires, to change the details. Here, the 'Notes' are free for the Staff to change or add anything they think is of importance to this particular job. The 'Start Time' is a field that is filled in once a Support member has begun work on it. Once this is set, it cannot be changed and the 'Status' automatically sets to 'In Progress'. Once a job has been completed, it is up to the IT Support Staff to set the 'Time of Completion'. Once this has been set, 'Status' changes to 'Done'. After the IT Support member is satisfied with the request, he/she may save or cancel the changes he/she made.

The *Delete Request* page allows IT Support members to delete the request that they have chosen. A reason for the desirability of this feature is that a Support member may have had informal contact with the submitter of the request, who incidentally may have already fixed the problem.

The *List Pending Requests* and *List Current Requests* allows IT Support Staff to view all those requests that have not begun and those requests that have begun (but have not been completed).

The *Search for a Request* page allows IT Support Staff to search for a particular request. He/She must fill in at least one field of: the date that the request was sent, submitter's name, the type of problem or the keywords in the description.

The *IT Manager Introduction* allows the IT Manager to gather information about the IT Department's support history, finances and allocations of resources.

4.4. Website Design [9]

4.4.1. Site Map

The website was designed with ease of navigation in mind and this is evident in the ease with which one can form a mental model of the entire website and the intuitive nature of the model. A site map is provided to show the overall structure of the website, showing the links between each page.

Note that once a user has been identified, he/she will have his/her own path of divergence that is distinct from a user from another user group. For example, if an IT Manager authenticates him/herself, he/she can only use the pages designated specifically for him/herself. If he/she wishes to gain access to the system as a General User, then he/she will have to log off and re-authenticate him/herself as a General User.

However, once a user is logged on/authenticated, the website is designed so that the user can enjoy easy navigation. This is achieved by having all sibling pages linked together. For example, if IT Support Staff has finished searching for a request in 'Search for a Request' and wishes to 'List All Requests', he/she can easily do so by clicking on a single link on the 'Search for a Request' window.

A very helpful feature that was incorporated in designing the websites was the feature of knowing exactly where in the site map one is located. This is possible through the banner (which indicates what the exact operation or purpose

[9] See Appendix (Website Design)

of the page is) and the top right hand corner logo (that indicates which of the user groups is able to use the page). For example, if one were to see the 'New Support Request' page, one could easily tell that the function of the page was to make a new support request and that the user group is General User.

4.4.2. The Page Schema in the ADM Schema

Navigation in this website is made simple by the intuitive (but not redundant) links throughout the website. An example of these links is shown in the Appendix; this example shows the manner in which one can access the full details of a given request. From the list of all requests, one can arrive at the full page of the request by following a link given only partial information.

4.4.3. The Heterogeneous Union and Form in the ADM Schema

There are various instances where we will need a mechanism for searching for a particular problem. Given partial details on a certain request (such as date, name of submitter, type of problem or keywords in the description), the system will search the list of all its problems and return the full details of the matched problem.

4.5. Input/Output Design

4.5.1. Input Design

Since this information system is based on a local intranet where it is safely assumed that everyone in Mandrake has access to a computer, there are no foreseeable or extraneous input modes that have to be designed.

4.5.2. Output Design

The most obvious need for output design comes in with the role of the IT Manager. He/ She will not have the time to be too concerned with learning how to use the particular system, but will want to keep track of certain statistics. The output in question is *internal output* - output (such as summary reports) that never leaves the system and is useful for information gathering and management purposes. The output that this system uses will be on the paper medium generated weekly/monthly for the IT Manager and the archival version of the output will be put onto tape.

For the IT Manager, there will be charts (most appropriately on paper and printed to his/her preference) such as those shown in the Appendix[10]. Various charts will be printed. The first in the Appendix, Percentage of Types of Problems Taking up IT Support Staff Time, indicates the time taken by each category of problems on a quarterly basis throughout the year. As indicated by a quick glance at the chart, the IT Manager can conclude that more money or resources should be invested into the improved performance of the Web-related infrastructure for the company. This is because Web-related problems are at the forefront of problem categories occupying IT Support Staff time in each quarter.

The second example in the Appendix, Average Percentage of Problems Fixed by Different Support Staff, shows that throughout the year, Chantelle Gero, fixed most of the problems submitted to the IT Department. A conclusion to this is that she may get a pay raise, or have other members of the team improve their performance.

4.5.3. Internal Controls for Inputs/ Outputs

There will be various data validation checks for inputs throughout the website. An example of input control is the graying out/ blocked feature of the 'Notes' in the General User -> New Support Request. This allows users to see any notes that an IT Support person may have written, but disallows the user to modify any of those notes. Others include checks by JavaScript, an embedded language that acts as an internal filter for simple errors. An example of such error checks is in the 'Name' and 'Description' fields of General User -> New Support Request. These fields are mandatory and should they not be filled on submission, an error message controlled by JavaScript will report failure of submission. A typical example of this can be found in the Appendix [11].

[10] See Appendix (Output Design Charts)
[11] See Appendix (Error Message Popup for Internal Controls for Inputs)

As for internal controls for outputs, there will be exact specifications with respect to time, volume and destination. Output will be generated for the IT Manager each day/ week/ month, depending on preference. Volume of output will be produced according to the preference of the IT Manager also. The destination is quite obviously, the IT Manager him/herself.

4.6. Justification that the Interface Design Meets Relevant Requirements

The Authentication Window greets all users. This window verifies the amount of information to the company's support request database a user logging in would have in that session. For example, the general user would only have access to his/her submitted requests or could only make a new support request. On the other hand, an IT Support member would have access to more information, such as a list of all user requests and a list of all pending requests. Thus, the requirement that the system must take the user's name and ID in order to verify if that he/she is authorized to use the system is justified in the Authentication Interface.

The interface for general computer users allows the general staff to reset the form and/or to enter the required information, submit their requests by pressing a single button and receive a confirmation that their message is being processed or an appropriate error message as to why their submission cannot be processed. In addition, it allows users to check the status of their requests, cancel or modify the submitted ones.

The IT Support Personnel Interface allows support staff to view requests, parse information submitted by users, add notes, list pending requests, list all the requests, check what requests are being currently processed, schedule the time to process the selected requests and choose the action with which to proceed. In addition, it allows support staff to store the processed requests in the archived database, modify them, (optional: to delete them), view and list them, search the database by a given key (e.g. by date, by name, by problem type, alphanumerically, etc.), select certain subsets of required items stored in the database.

The IT Manager Interface allows the manager to generate monthly or weekly reports based on the data stored in the archived database.

From both the IT Support Personnel Interface and the IT Manager Interface (and partially the interface for general computer users), there is access to detailed information as to whether the handling of the request was completed, as well as information about the starting time of the support process, unique ID assigned to the problem, priority, estimated complexity, problem category, completion date, and all the details about the user and his/her interpretation of the problem.

For the above reasons, the interfaces for each user group and the Authentication Interface provide complete functionality in terms of requirements [12].

In addition, all the interfaces are well designed with respect to the characteristics of well-designed interfaces. By this, we mean that the interfaces keep in mind affordances, obvious mappings, mental models, feedback, forcing functions and learning. Since every user at Mandrake is competent at web browsing, affordance, automatic learning and mapping instances are provided very easily. They would all be able to navigate themselves with ease, given the simplicity of the website design and the similarity of this website to general websites. A mental model and feedback is provided through data validation techniques (error messages, etc.) and through confirmation through the system that, for example, the user has submitted a request. Forcing functions are made possible by the blocking of or graying out of particular fields. An example of this is the General User's inability to use the 'Notes' field of their New Support Request interface.

---

5. Conclusion

Our report has been comprehensive in detailing the design for the information system for our previous reports – the Feasibility Study and Requirements Analysis. Our detailed design encompassed selecting hardware, networking and

[12] See Appendix (Functional Requirements)

software for the new system, designing a global architecture, proposing a detailed design for the classes that are part of the system, as well as defining a relational database schema and suitable I/O procedures and interfaces. Our report has brought us well on our way to the next step – the implementation step.

6. Appendices

6.1. Appendix: Introduction

6.1.1. Related Factors and Constraints

The major constraint for Mandrake IT is person-power. The director of IT, Tom Metaxas, is responsible for both Network Administration and IT Support for 80 people. A part time student also provides IT support. Both members are constantly busy, Tom especially so since he fulfills a dual role. All of the above indicates either lack of managerial structure in the company's hierarchy or about an inappropriate scheduling and request processing mechanism. We chose the latter to analyze.

6.1.2. Evaluation Process and Criteria

The primary goal of IT support is to ensure the efficient and smooth operation of a firm's IT resources. Although IT is critical to Mandrake's operations, IT itself does not generate direct revenue for Mandrake. Thus, any improvement in IT support is measured in reduction of costs. Thus, the primary criterion we chose to use to evaluate the various alternatives was cost reduction.

We used a rather detailed analysis of each of the alternatives to conservatively estimate the monetary savings from each of the alternatives. Although, it is difficult to precisely measure all the factors, the two major factors affecting the monetary benefits were decreased consultant and associate "down time" due to IT failure and increased IT support efficiency. Our team of crack analysts identified several metrics affecting these two costs: IT incidents per week, average length of IT incident, average IT Request-to-Processing (excluding IT staff travel time), average overhead for IT staff arrival, and user resolvable instances. The feasibility study contained a detailed excel spreadsheet with projections.

After we completed all the analysis, we chose the most cost effective options to recommend to the client. Although, theoretically all of the alternatives might be selected, in practice it is much better to choose only a few to reduce the complexity of the implementation task.

6.1.3. Recommendations

Our **non-software development** recommendations are as follows:

1) Hire an additional IT person. Although the straight monetary analysis does not provide strong evidence, a combination of tangibles and intangibles make a good case. First, the current Director of IT spends far too much time doing technical support. Second, a large backlog of tech support work exists that needs to be cleared. Third, implementing the recommendations will require some support staff.
2) Install Remote Control Software. Further research will need to be done on this topic to determine the exact product, but common sense and the financial estimates strongly support this option.
3) Institute mandatory training. The tangible, monetary and intangible benefits are enormous. In-house training is preferred, but during busy times it can be supplemented with external courses.
4) Introduce a written policy regarding priority of IT Support and make sure users are aware of this policy.

6.1.4. Selected Alternative: Web Based Support Request Form

According to the conducted feasibility study, the alternative of processing technical support requests through the web based form showed itself financially efficient ($11,870). This is the only alternative requiring significant software development. For these two reasons we selected it for requirements analysis.

In general, Web based processing system allows users to enter all the information into the predetermined by support people text fields and enter the complexity level of problem on a scale so that prioritization can be automatically computed. The interface of this form (a sample HTML form is included in appendix 10), implemented in the standard Internet fashion, will be familiar to users, who are used to work with Internet on daily basis. The limited number of fields and specific subcategories of fields contribute to the structured and concise layout of the request

received by a technical support specialist. Hence, this minimizes cognitive load to decode the submitted document and allows starting technical support process much faster.

The priority level of the request is a useful feature helping a tech support person to manage his/her time resources efficiently, since he/she can queue all the requests in more optimized order. There is also a possibility that a user might underestimate or overestimate a level of difficulty of the problem, but this can be balanced by analyzing the level of difficulty in conjunction with subject matter or locality of the problem (details are left to implementation).

This web form allows "twenty-four-seven" (all-day and night) submission and can be expanded or modified based on its performance. On the other hand, this alternative solution is also vulnerable to technical failure, i.e. it won't help if a user has problem with Internet access.

Overall, this alternative (since our other recommendations did not require requirements analysis) can improve the performance of the existing system by introducing a standard way to organize the requests processing routine.

6.1.5. Requirements Details

6.1.5.1. Functional Requirements

The system should be able to perform the following functions:

- Take the user's name and ID in order to verify if that he/she is authorized to use the system. The verification part includes the check for the user's accessibility to the specific company's resources.
- Manage the traffic by waiting for the users' connections and creating specific interface sessions for them.
- Allow users to reset the form and/or to enter the required information (see input section about data description), submit their requests by pressing a single button and receive a confirmation that their message is being processed or an appropriate error message as to why their submission cannot be processed.
- Allow users to check the status of their requests, cancel or modify the submitted ones.
- Allow support staff to view requests, parse information submitted by users, add notes, list pending requests, list all the requests, check what requests are being currently processed, schedule the time to process the selected requests and choose the action with which to proceed.
- Provide detailed information as to whether the handling of the request was completed, as well as information about the starting time of the support process, unique ID assigned to the problem, priority, estimated complexity, problem category, completion date, and all the details about the user and his/her interpretation of the problem.
- Allow support staff to store the processed requests in the database, modify them, (optional: to delete them), view and list them, search the database by a given key (e.g. by date, by name, by problem type, alphanumerically, etc.), select certain subsets of required items stored in the database.
- Allow the designated person to generate monthly or weekly reports based on the data stored in the archived database.

6.1.5.2. Non-functional Requirements

6.1.5.2.1. Interface requirements

There will be two interfaces for this information system: one is for users who need technical support, and another one for technicians. They both should be user-friendly, well structured and tested with each set of users. Note: the interface for the users should be simpler and more straightforward.

6.1.5.2.2. Performance requirements

e) *time/space bounds:* The system should be able to support requests from all users simultaneously (in the worst case). The response time of the system will be instantaneous in the case a user asks for the status of his/her request. The system should also have enough space to store all the submitted requests as well as the history log or archive of all the requests in the past.

f) *reliability:* The web form has a high reliability level since it behaves consistently in a user-acceptable manner when operating within the environment for which it was intended. The Internet connection should be at least 99% reliable (at least for the hosting server where the form and data are stored). High reliability from the Internet connection is not that crucial for the remote workstation since the web form can be accessed through another neighbour computer. The form may break down one hour per year on average. The Mean Time To Repair must not exceed one to two hours. The restart time should be five minutes. No more than two bugs should be found yearly. If the system continuously breaks down, the implementation of this alternative should be revised.

g) *survivability:* The system will survive any catastrophes as long as they do not affect the server, where the web form and all the relevant data are stored.

h) *efficiency:* The capacity of the system should be such that the company network system is able to handle 80-90 terminals (appropriate for the number of employees) with the ability to expand if necessary. The degradation of service may occur when load on the system exceeds its capacity. In terms of timing requirements,
   - *stimulus-response:* e.g. the system will submit a request within a half to five seconds after it was completed by the user
   - *response-response*: e.g. the system will deliver a submitted request to the appropriate category/account based on calculated priority level within one minute.
   - *stimulus-stimulus:* e.g. a user may check for the status of his/her request two minutes after it was submitted.
   - *response-stimulus:* e.g. a user must wait for the technical support after he/she receives confirmation that his/her request is being processed.

6.1.5.2.3. Operating requirements

k) *physical constraints:* The size of the web form must not exceed 5 MB. The size of the support database should not exceed twenty GB (the size of the average contemporary hard drive).
l) *portability:* The degree of portability is high since the system relies on standards wherever possible (e.g. HTML, SQL database interface, and HTTP/TCPIP). All standards used are currently supported by most operating systems.
m) *security*: The system must be protected from the loss of information. Therefore, it is better to make it visible only to the company's employees. If this form is located on the official web site, then a password to access a form for internal technical support is required. One hundred and twenty eight (128) - bit encryption is optional but recommended due to the fact that the information submitted through the form is private but not highly secretive. Since the system is not prone to viruses, no special protection is required.
n) *personnel availability:* There should be enough people to process all the technical support requests.
o) *skill level considerations:* The front user interface should require minimal skill level, i.e. no additional training. The support personnel interface may require more advanced skills, i.e. knowledge on how to process a request.
p) *accessibility for maintenance:* The system should be maintained by a centralized unit or a single person (a network administrator in Mandrake's case).
q) *environmental conditions*: no special environment is required.
r) restart requirements: The system shall perform an automatic restart in the event of fatal software error, to be completed within 5 minutes.
s) *backup requirements*: optional
t) *fallback requirements*: the system should display an appropriate error message and terminate the current submission in the event of fatal error.

6.1.5.2.4. Lifecycle requirements

c) *quality of design*: The level of maintainability, enhancement and portability must be set with high standards from the beginning since we do not want an increase in any maintenance cost.
d) *limits on development*: There should be none, or even if there are, these limits should be reasonable on development since the model to be implemented is not very complicated.

6.1.5.2.5. Platform requirements

g) *memory:* 64 MB (-28 MB preferable) in order to run Windows, MS Office and the browser.
h) *disk space*: 20 GB (smallest hard drive offered nowadays) is sufficient to store both the form and submitted data.
i) *operating system*: Any OS will suffice but Windows 98/2000/NT is preferable since Mandrake is already a Microsoft client.
j) *CPU*: 300 MHz and more
k) *peripheral:* none (printer recommended for those items that are interesting or complicated enough that the support staff might want to print)
l) *network*: Internet connection (minimum Cable speed; DSL, T1 or T3 are preferable).

6.2. Appendix: Program Design

6.2.1. Business Objects Package

This package contains all objects for the business logic of the system

6.2.2. ActiveUser

This represents the active user information for an established session. There is *NO* assurance that user either exists or has entered the correct password.

6.2.2.1. Private Properties:

password: String
The password associated with the username.

userName: String
The username that the remote client has reported to the system.

6.2.2.2. Public Methods:

getUserName();
Gets the username for the object.

setUserName();
Sets the username for the object.

getPassword();
Gets the password for the object.

setPassword();
Sets the password.

6.2.3. ITManagerInterface

This is the interface designed specifically for the IT Manager. He/she conducts all human-computer activity on his/her part through this.

Derived from GeneralUserInterface

6.2.3.1. Public Methods:

String getSupportReport();
When called, this returns any detailed information that has logs of previous users, dates and times of submissions, as well as any other useful information.

chooseAction();
Directs the object to proceed along the right path based on whether the object is an ITManagerInterface, ITSupportPersonnelInterface or SupportRequestUserInterface.

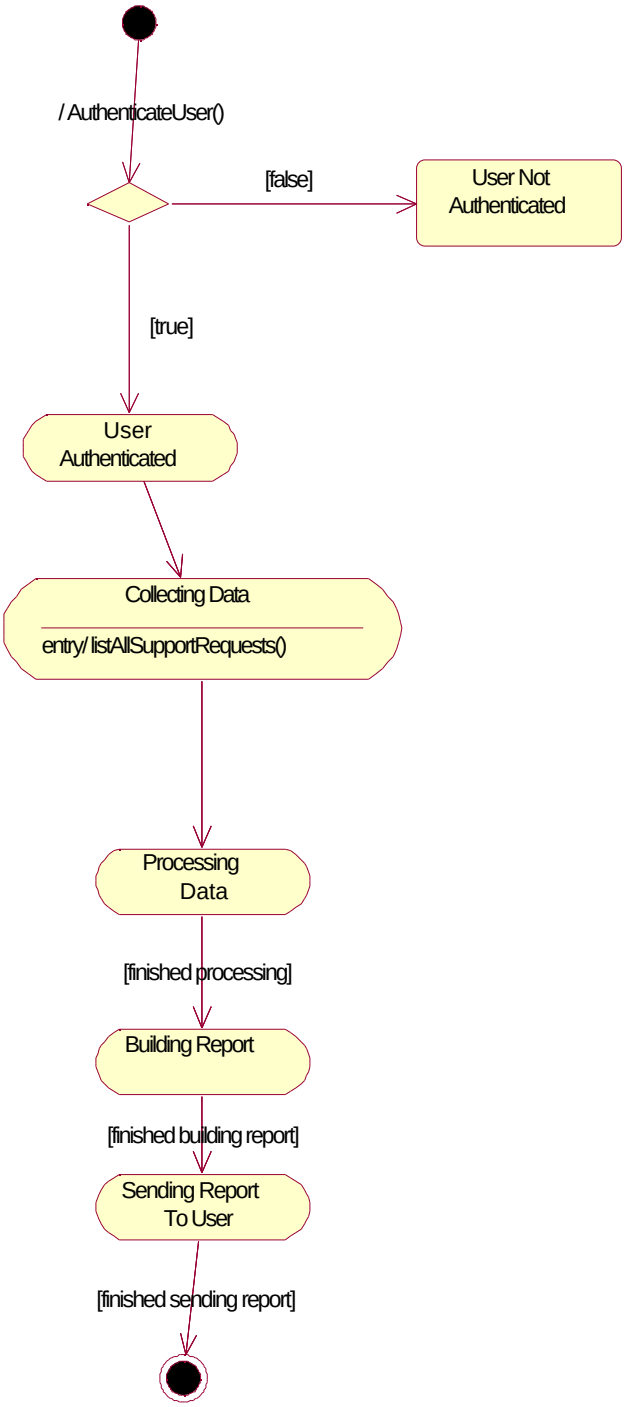


Figure 7 getSupportReport() Diagram

6.2.4. ITSupportPersonInterface

Represents the actions that an IT Support Perosonnel can do with the system. It is basically a one-to-one mapping with the use cases listed before.

Derived from GeneralUserInterface

6.2.4.1. Public Methods:

viewSupportRequest();
Views the inteformation of a single support request and displays the information appropriate for an IT Support Personnel in some manner for the user.

addEntryLogToRequest();
Adds an entry log by the IT Support Personnel to the support request.

scheduleRequest();
Schedules the starting time for when a request job is being handled.

beginProcessingRequest();
This flags the request as currently in progress.

listPendingRequests();
Shows the list of requests that are still to be tended.

listSupportRequestsInProgress();

Shows the list of requests that have begun but have not completed.

AllRequests();
Shows the list of all requests in the database.
chooseAction();
Directs the object to proceed along the right path based on whether the object is an ITManagerInterface, ITSupportPersonnelInterface or SupportRequestUserInterface.

modifySupportRequest();
Modify an existing support request.

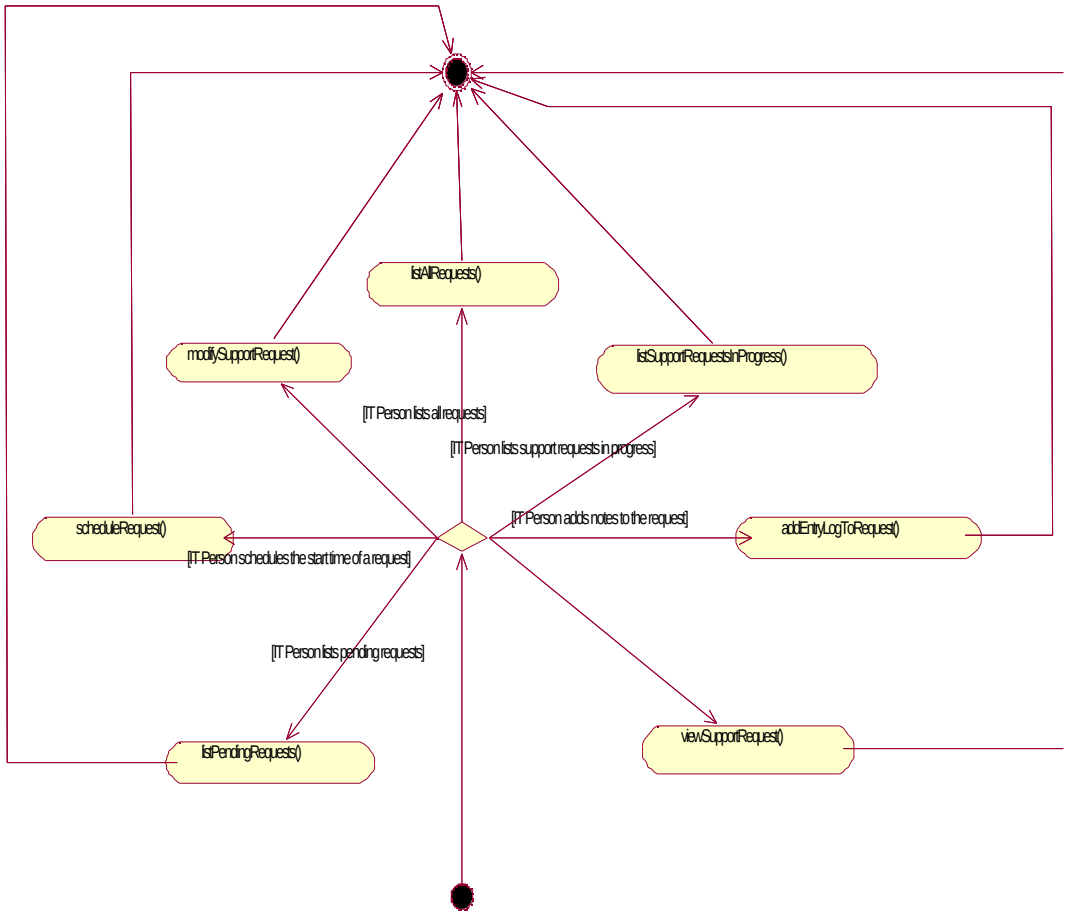


Figure 8 ChooseAction() State Diagram

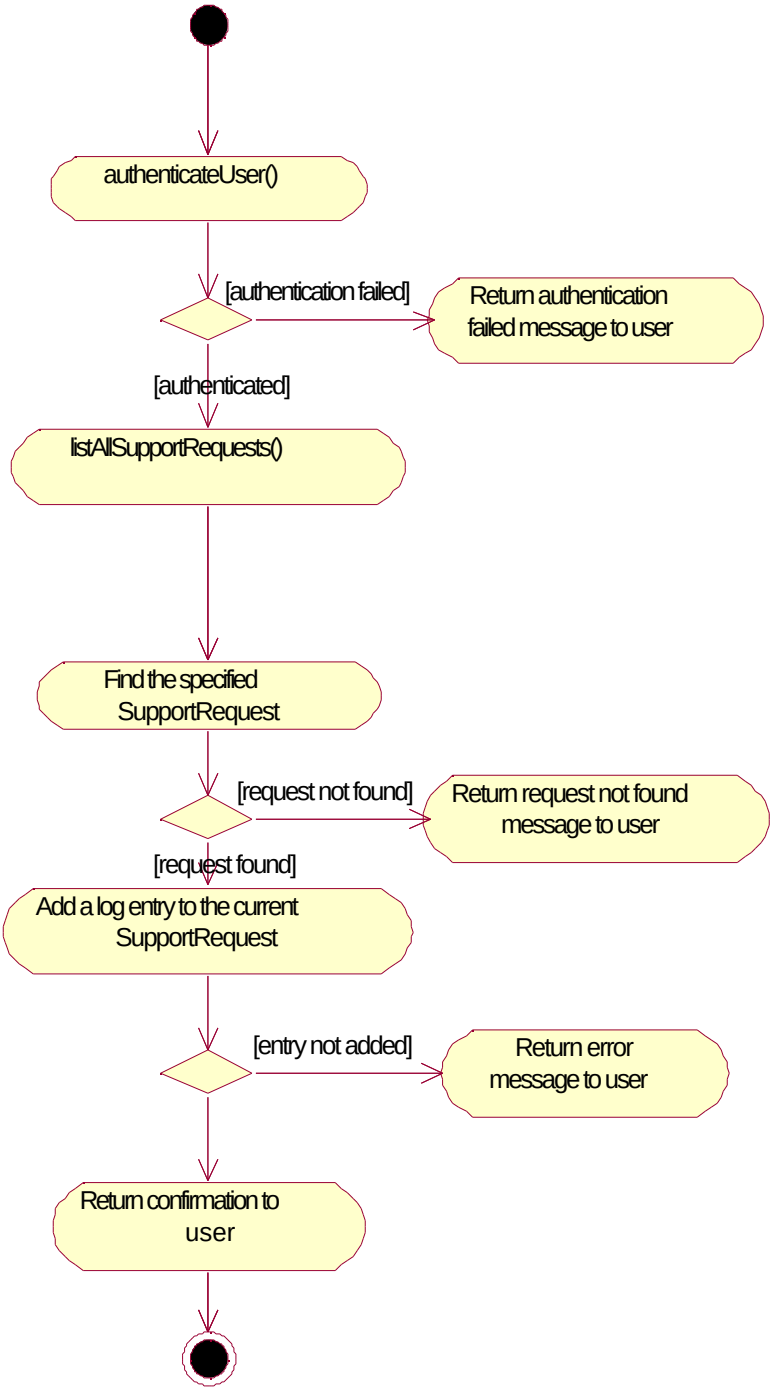


Figure 9 addEntryLogToRequest() Diagram

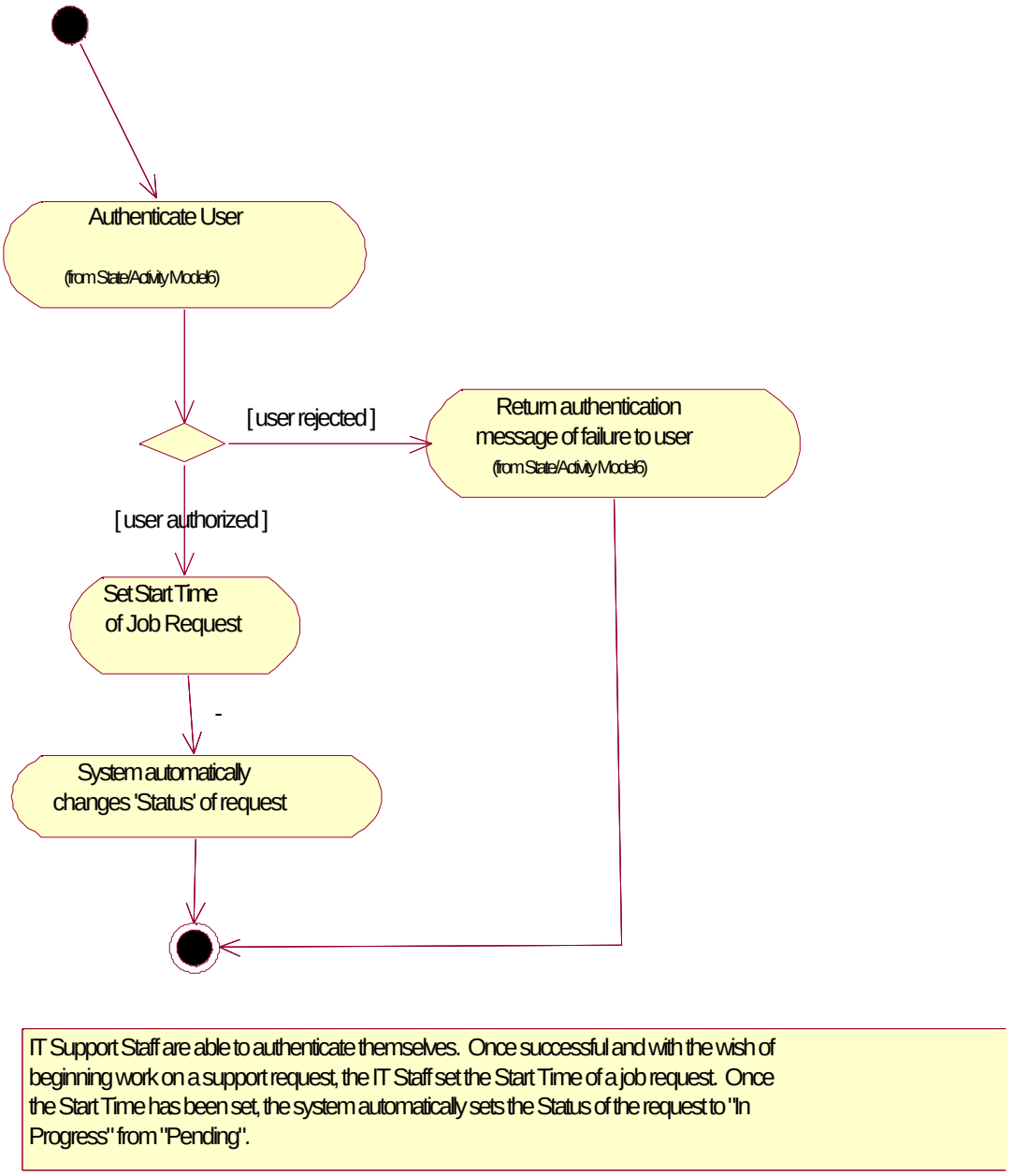
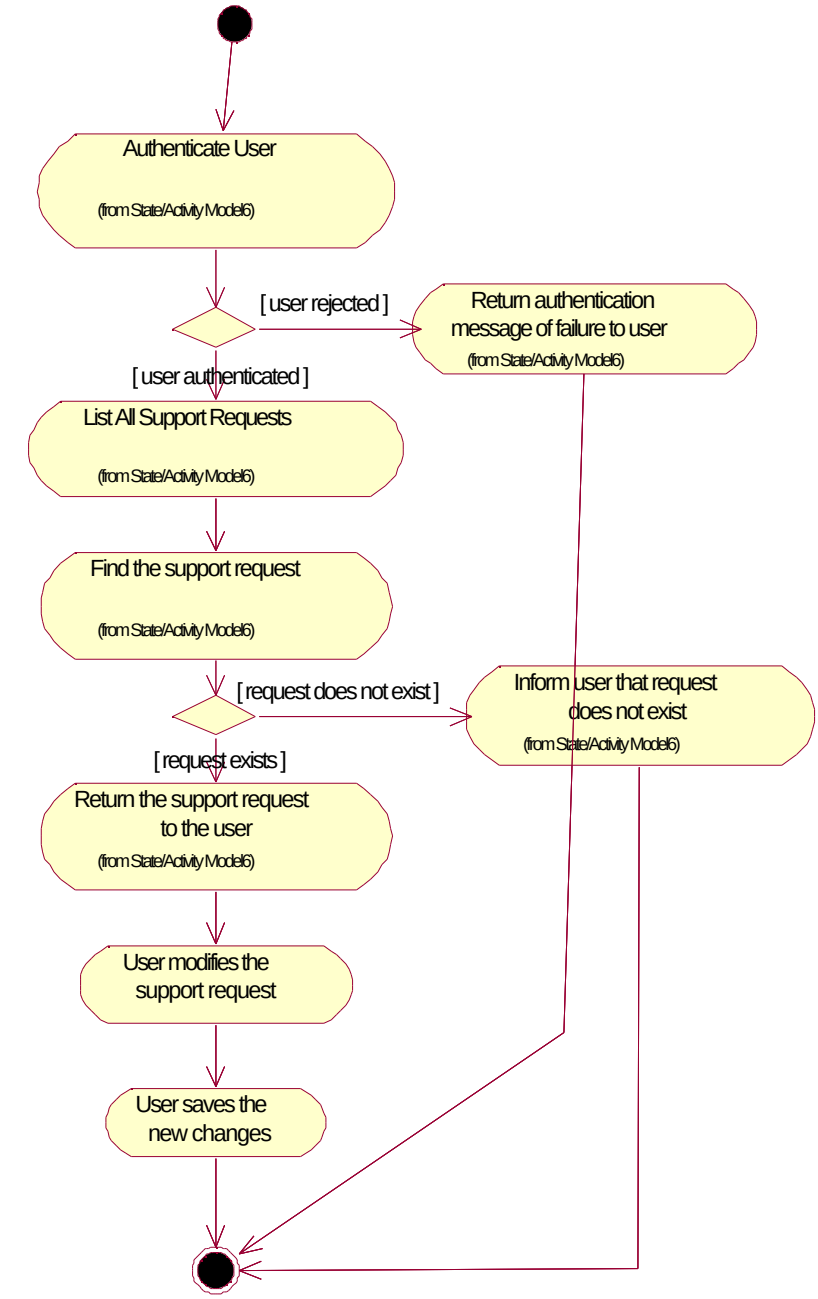


Figure 10 beginProcessingRequest()



After authentication, the IT Support Staff are allowed to list all support requests. Subsequently, they are able to choose from the list which of the requests they would like to make changes to. After they have made the relevant changes, they would save the changes to the database.

Figure 11 listAllRequests()

6.2.5. Mailer

Provides functionality to send simple email messages.

6.2.5.1. Public Methods:

Boolean sendEmail(String to, String from, String body);
This sends an email to the specified address from the specified address in plain text.

6.2.6. SupportRequest

This is a representation of a single support request.

6.2.6.1. Private Properties:

startTime: Date
Time that actual job handling or fixing of the request starts. This is needed since it will make implications on what the status of the request is.

submitTime: Date
The actual time the support request was submitted by the user.

uniqueID: Long
ID of request. Needed for easy reference for support staff as well as users.

problemCategory: String
This is the category under which the request is placed under. This is useful since, say for example, a support staff might want to view the list of hardware problems, or network problems, etc.

userNotes: String
When user makes a request, he/she may enter complementary, general notes that are not within any specified fields in the request form.

priority: Long
This is the priority that informs support staff how major a problem is. The user implements his/her own discretion and integrity in filling this out. (Major problems would be those that likely impede a user from doing his/her work).

estimatedComplexity: Integer
Estimated complexity by the support staff of request tells the support staff how difficult a problem might be to fix.

completionDate: Date
Date of completion has implications on what the status of the request is.

requestStatus: String
This is the status of the request that any user can see at any time.

userEstimatedComplexity: Integer
The complexity estimated by the user.

6.2.6.2. Public Methods:

Boolean isCompleted();
Determines if the support request has been completed.

Date getStartTime();
Returns the estimated start time of the support process. This is so the user will know when he can expect support to be received. It is set by the IT Support Personnel when he reviews the support request.

Date getSubmitTime();

Long getUniqueID();
Retrieves the unique ID of this particular request. Needed so that later references to this request can be easily fetched.

String getUserName();
Returns the user name of the user actually submitting the request.

String getProblemCategory();
This is a string in the form defined by BNF below:

PROBCAT ::= <CATS>,<CATS> | <CATS> | Empty

<CATS> ::= <CATS>.<CATS> | <CATS>

The string can present hierarchical category arrangement and multiple categories. e.g. Hardware, Mouse, Software, Software, Windows, Internet, Windows, TCPIP

String getNotes();
Returns general notes on the problem

Integer getPriority();
Returns the submitting user assesed priority of the problem

Integer getEstimatedComplexity();
Returns the estimated complexity of the problem.

Date getCompletionDate();
Gets the actual completion date of the request

String getRequestStatus();
Gets the status of a request.
Integer getUserEstimatedComplexity();
Gets the estimated complexity according to the user.
Boolean setUniqueID(Long s);
Sets unique ID.

Boolean setUserName(Long s);
Sets user name.

Boolean setProblemCategory(String s);
Sets Problem Category.

Boolean setStartTime(Date s);
Sets start time.

Boolean setPriority(Integer s);
Sets priority of request.

Boolean setCompletionDate(Date s);
Sets date of completion. Needed since this will have implications on the status of the request.

Boolean setUserEstimatedComplexity(Integer s);
Sets the user's estimated complexity of the request.
Boolean setRequestStatus(String s);
Sets the request status (Pending, In Progress, Done)
String getAssignedToITPerson();
Gets the name of the IT Support member who is working on the request.

setAssignedToITPerson(String username);
Assigns a request to a member of the IT Support team.

6.2.7. SupportRequestCore

This class contains the core functionality for dealing with support requests.

6.2.7.1. Public Methods:

Boolean submitSupportRequest(SupportRequest request);
Submits the completed support request form to the Support request data base.

Boolean deleteSupportRequest(Long uniqueID);
Deletes the specified support request.

SupportRequestList listAllSupportRequests();
Lists all support requests in the database.

SupportRequestList listAllSupportRequests(String userName, Date startRange, Date endRange);
An overloaded version that gives the option of a particular user or a date range.

SupportRequestList listPendingSupportRequests();
Lists all the pending support requests.

SupportRequestList listPendingSupportRequests(String userName, Date startRange, Date endRange);
An overloaded version that gives option to filter based on user or dates.

SupportRequestList listCompletedSupportRequests();
Lists all completed support requests.

SupportRequestList listCompletedSupportRequests(String userName, Date startRange, Date endRange);
An overloaded version that gives option to filter based on user or dates.

SupportRequestList listSupportRequestsInProgress();
Lists all support requests currently in progress.

SupportRequestList listSupportRequestsInProgress(String userName, Date startRange, Date endRange);
An overloaded version that gives

modifyRequest(Long uniqueID, SupportRequest request);
Allows user to change or edit any request items.

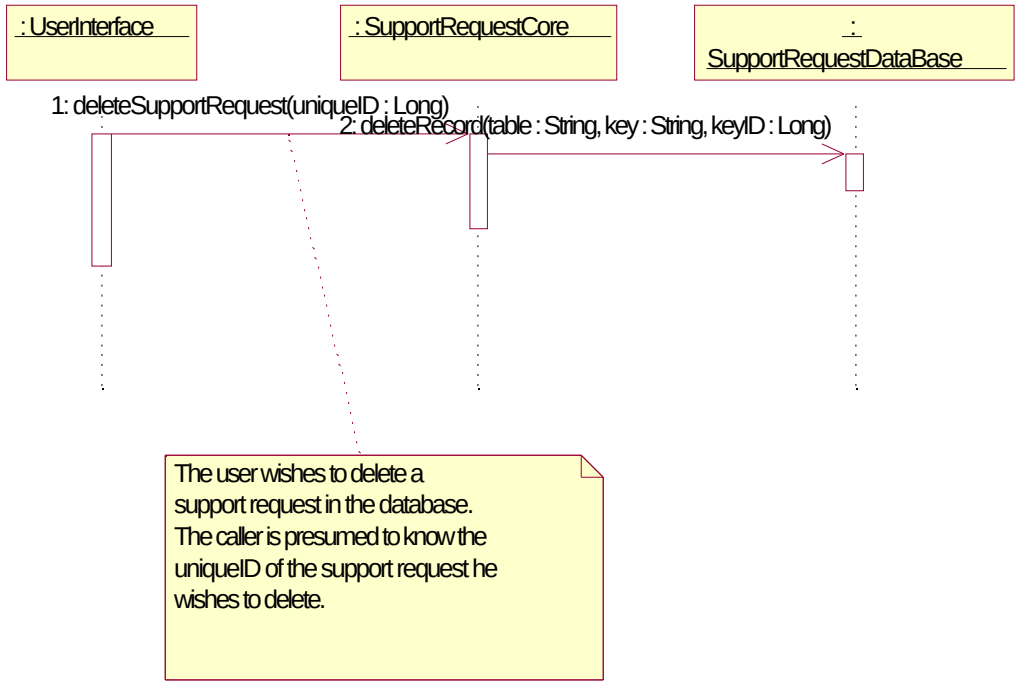


Figure 12 deleteSupportRequest() Sequence Diagram

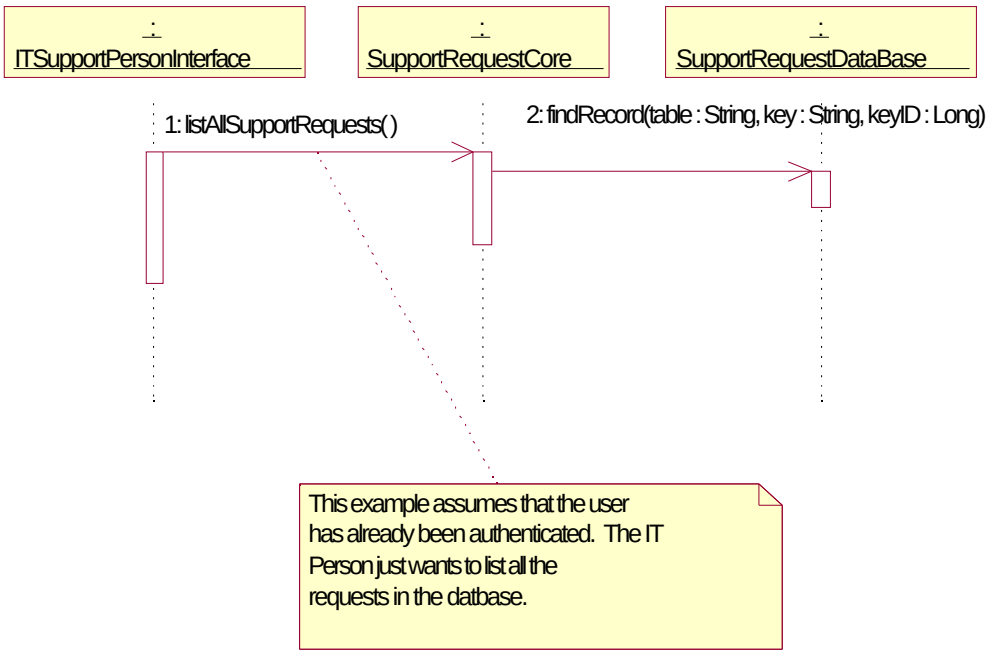


Figure 13 listAllSupportRequests() Sequence Diagram

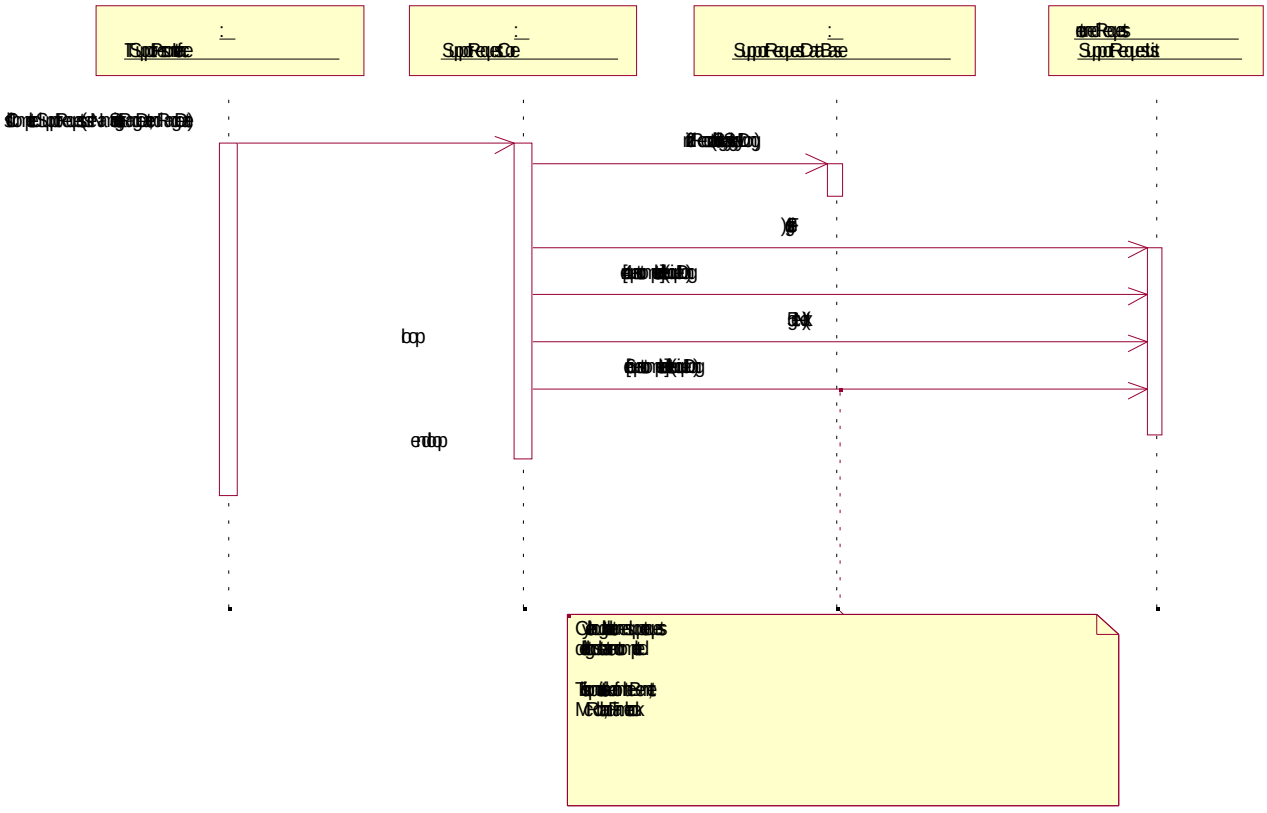


Figure 14 listCompletedSupportRequests() Sequence Diagram

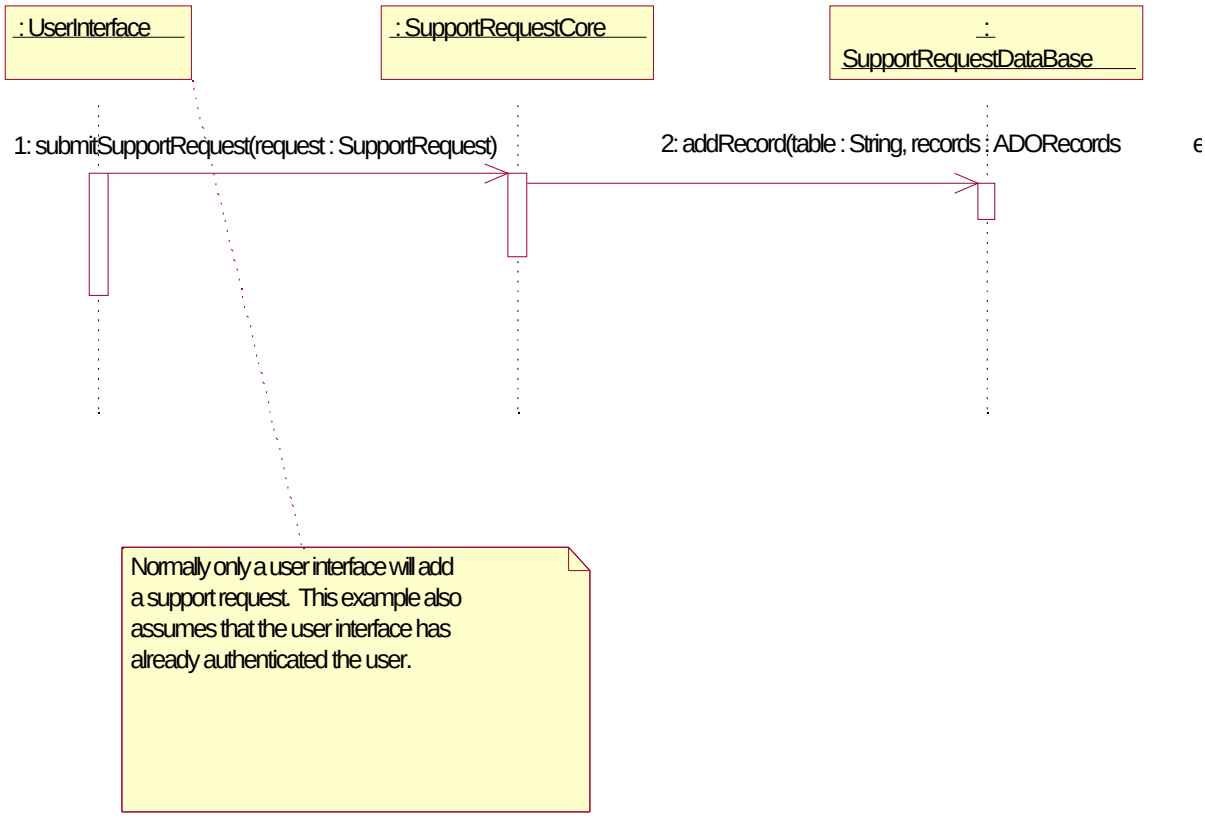


Figure 15 submitSupportRequest() Sequence Diagram

6.2.8. SupportRequestList

This represents a list of Support Requests. Normally, this is used when the IT Support staff wish get a list of the support request items.

6.2.8.1. Private Properties:

index: Long
Index is the current request that is being referenced in the list.

6.2.8.2. Public Methods:

sort(sortType, sortKey);
Sorts the given list according to the given sortKey (what field to sort) and sortType (how to sort it - for example: alphanumerically, by date, etc.).

SupportRequest getFirst();
From the list, get the first request.

SupportRequest getNext();
From the list, get the next request.

SupportRequest getNth(Long n);
From the list, get a request that is identified by number.

Boolean addSupportRequest(SupportRequest request);
Adds a support request to the list of support requests.

Boolean delete(uniqueID : Long);
Deletes a particular request identified by the unique ID of the request.

Boolean deleteAll();
Returns true if all the requests in the list have been successfully deleted.

6.2.9. SupportRequestLogEntry

This class represents a single log entry entered for a support request. There can be many log entries for one request.

6.2.9.1. Private Properties:

logEntryTime: Date
The creation time of the log entry.

supportNotes: String
The notes for the particular a log entry.

uniqueLogID: Long
The unique identifier of this support request log entry.

6.2.9.2. Public Methods:

Date getLogEntryTime();
Get the log entry time.

String getSupportNotes();
Get the support notes from this entry.

getUserName : String();
Get the IT Person username who entered this log entry.

setLogEntryTime(Date time);
Sets the time for this log entry.

setSupportNotes(String notes);
Sets the notes for this log entry.

setUserName(String name);
Set the IT Person user name for this entry.

6.2.10. UserInterface

This class is responsible for the human-computer interaction on the part of the users who have technical difficulties.

Derived from GeneralUserInterface

6.2.10.1. Public Methods:

chooseAction();
Directs the object to proceed along the right path based on whether the object is an ITManagerInterface, ITSupportPersonnelInterface or SupportRequestUserInterface.

submitSupportRequest();
Submit a new support request.

cancelSupportRequest();
Cancel an existing support request.

listSubmittedRequests();
List the submitted support requests by the user.

modifySubmittedSupportRequest();
Modify an already submitted support request by the user.

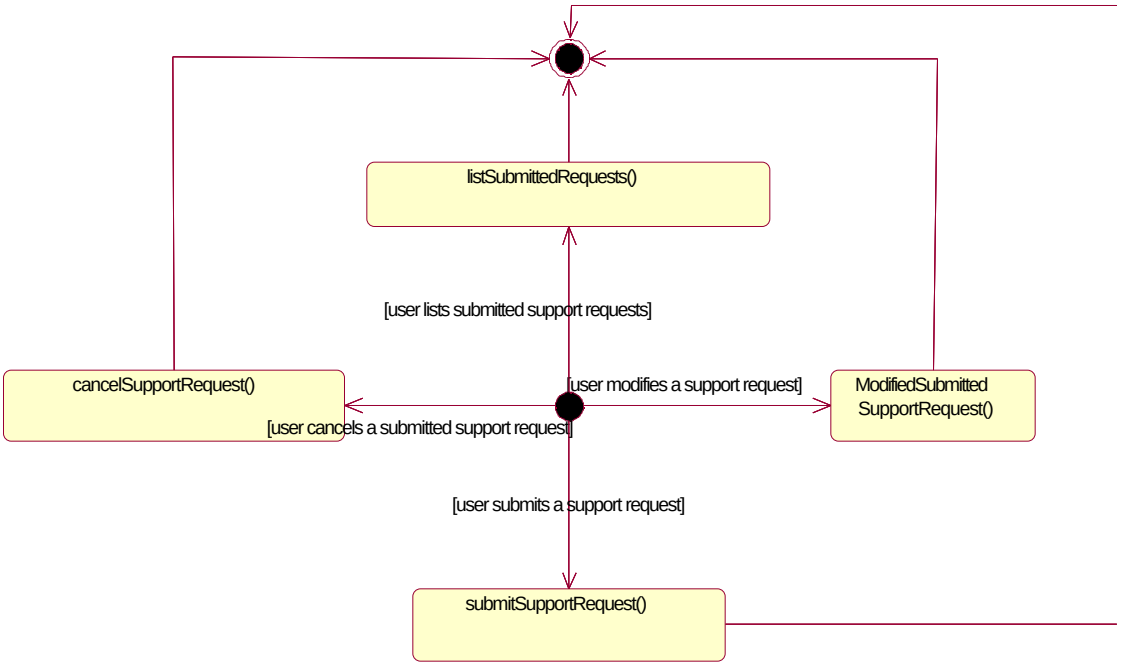


Figure 16 chooseAction()

6.2.11. Database Package

6.2.12. ADORecordset

This is merely a placeholder for the Microsoft supplied ADORecordset class.

6.2.13. AuthenticationUser

Derived from StaffMember

6.2.13.1. Private Properties:

password: String
disabled: Boolean
type: String

6.2.14. ITManager

Derived from StaffMember

6.2.15. ITSupportPerson

Derived from StaffMember

6.2.16. StaffMember

6.2.16.1. Private Properties:

name: String
userID: String
uniqueID: Long
email: String

6.2.17. SupportRequestDataBase

This class is responsible for holding all support requests.

6.2.17.1. Public Methods:

ADORecordset findRecord(String table, String key, Long keyID);
Finds a particular record in the database according to the key and which item based on that key.

Boolean deleteRecord(String table, String key, Long keyID);
Deletes a particular record from the database by first finding the request using given key and keyID, then finally removing it from the list.

boolean addRecord(String table, ADORecordset records);
Adds a number of records contained in records to the table specified in the database.

6.2.18. UnifiedUser

This is the unified user that exists *AFTER* restructing the conceptual schema diagram. This class does not actually exist in the program itself.

Derived from StaffMember

6.2.18.1. Private Properties:

name: String
userID: String
password: String
type: String
disabled: Boolean
email: String

6.2.19. IIS 5.0 ASP Page

This represents the calling .ASP file. This is actually somewhat of an abstraction since it is more complex than that.

6.2.20. Security

This package contains the security objects required to enforce restrictions on the business logic.

6.2.21. Authentication

This class is responsible for taking users' names and ID's in order to verify their being able to use the network.

6.2.21.1. Public Methods:

Boolean authenticateUser(String name, String password);
Returns true if user is authorized to use this system. Otherwise, the user is rejected.

Integer determineUserLevel(String name);
Determines on what level the user is. This is needed for verifying how much of the system the user has access to.

6.2.22. Dispatcher

This class is the traffic controller for users as they initiate new sessions.

6.2.22.1. Public Methods:

waitForConnections();
This waits for incoming connections from users and then creates interface sessions for them when they connect.

6.2.22.2. Private Methods:

createInterface(Integer type);
Based on the specific section of the web site, one of three interfaces will be created to process the submission of the web form. This is called by waitForConnections() as a private method.

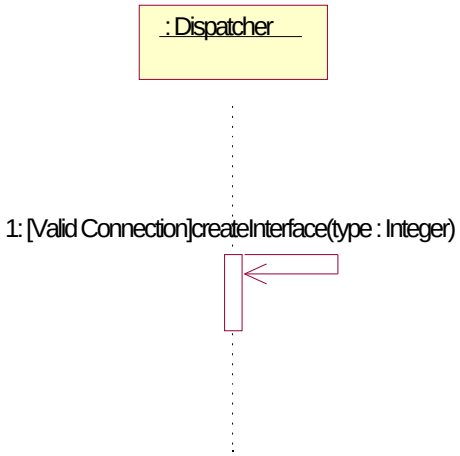


Figure 17 waitForConnections() Sequence Diagram

6.2.23. GeneralUserInterface

This is the general user interface used for all interaction between humans and the software. It will actually abstract the HTTP protocol submission of the software.

6.2.23.1. Public Methods:

parseForm(form);
Allows any call of this to parse the (filled out) form to the main database.

6.3. Appendix: Global Architecture & Supporting Evidence

Shopping Cart: This is an example of the price for the Oracle Database Server.

Product	**Store Price** **# of Units** **Licensing Unit**	**Amount**
	remove	
Oracle Database Standard Edition		15.00
	50	
UNIVERSAL POWER UNIT		7,500.00

Sub Total
7,500.00

Your Volume Discount
375.00

Your Total*
7,125.00

*(Estimated Taxes and a Shipping Charge may be added at final checkout.)

6.4. Appendix: Database Diagrams

Please see the descriptions for each operation in the Main Report.

Operation 1

Cost with redundancies:
R: (1 * 80 + 1 * 5) * 20 = 1700
W: 1 * 2 * 20 = 40 (per day)
Total: 1740 accesses per day

Cost without redundancies:
R: 2 * 86 * 20 = 3440
W: 2 * (1 * 20)= 40 (per day) counting from now on each write access twice.
Total: 3480 accesses per day

Operation 2

Cost with redundancies:
R: (1 * 80 + 1 * 20) * 40 = 4000
W: none
Total: 4000 accesses per day

Cost without redundancies:
R: (1 * 86 + 1 * 20) * 40 = 4240
W: none
Total: 4240 accesses per day

Operation 3

Cost with redundancies:
R: (80 * 1 + 1 * 20) * 20 = 2000
W: 1 * 20 * 20 * 2 = 800
Total: 2800 accesses per day

Cost without redundancies:
R: (86 * 1 + 1 * 20) * 20 = 2120
W: 1 * 20 * 20 * 2 = 800
Total: 2920 accesses per day

Operation 4

Cost with redundancies:
R: 80 * 1 * 20 = 1600
W: none
Total: 1600 accesses per day

Cost without redundancies:
R: 86 * 1 * 20 = 1720
W: none
Total: 1720 accesses per day

Operation 5

Cost with redundancies:
R: 1 * 20 * 40 = 800
W: none
Total: 800 accesses per day

Cost without redundancies:
R: 1 * 20 * 40 = 800
W: none
Total: 800 accesses per day

Operation 6

Cost with redundancies:
R: 1 * 20 * 10 = 200
W: none
Total: 200 accesses per day

Cost without redundancies:
R: 1 * 20 * 10 = 200
W: none
Total: 200 accesses per day

Operation 7

Cost with redundancies:
R: 1 * 20 * 10 = 200
W: none
Total: 200 accesses per day

Cost without redundancies:
R: 1 * 20 * 10 = 200
W: none
Total: 200 accesses per day

Operation 8

Cost with redundancies:
R: (1 * 20 + 1 * 100 000) * 20 = 2,000 400
W: none
Total: 2,000 400 accesses per day

Cost without redundancies:
R: (1 * 20 + 1 * 100 000) * 20 = 2,000 400
W: none
Total: 2,000 400 accesses per day

Operation 9

Cost with redundancies:
R: 1 * 20 * 20 = 400
W: 2 * 1 * 100 000 = 200 000
Total: 200 400 accesses per day

Cost without redundancies:
R: 1 * 20 * 20 = 400

W: 2 * 1 * 100 000 = 200 000
Total: 200 400 accesses per day

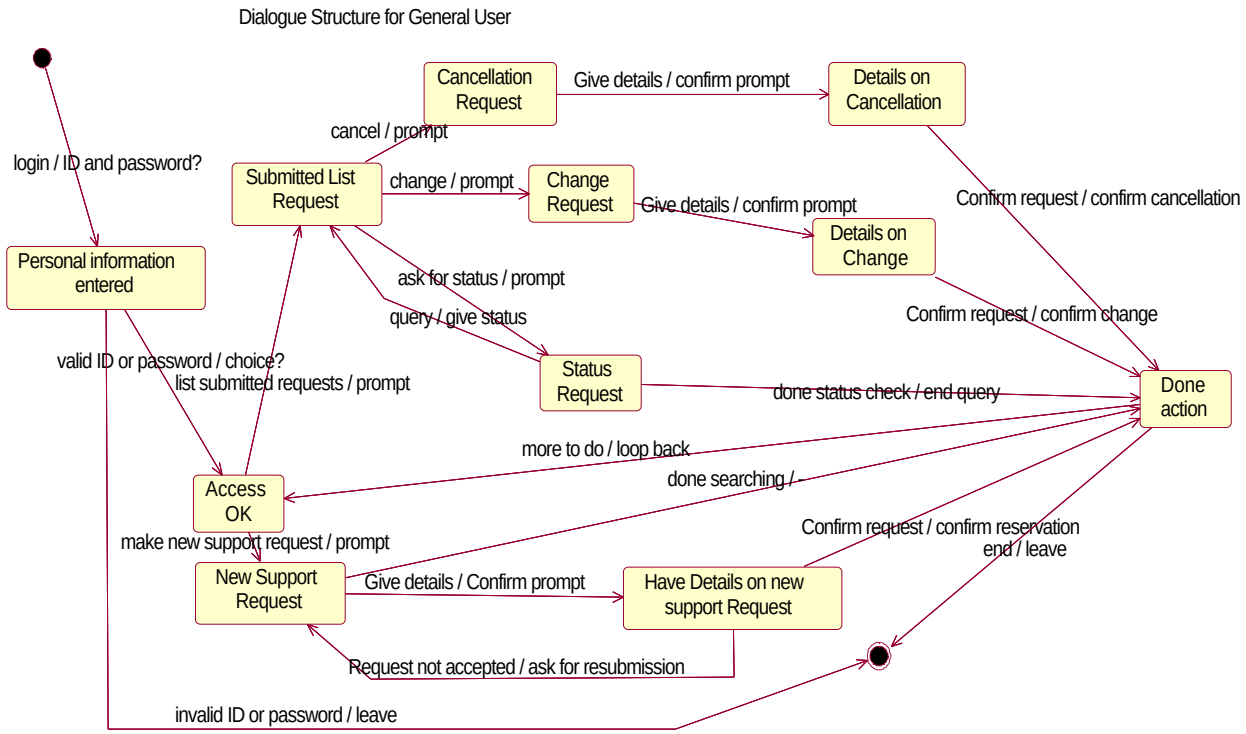
Operation 10

Cost with redundancies:
R: (1 * 20 + 1 * 100 000) * 20 = 2,000 400
W: 2 * 1 * 20 * 100 000 = 4,000 000
Total: 6,000 400 accesses per day

Cost without redundancies:
R: (1 * 20 + 1 * 100 000) * 20 = 2,000 400
W: 2 * 1 * 20 * 100 000 = 4,000 000
Total: 6,000 400 accesses per day

6.5. Appendix: User Interface Design

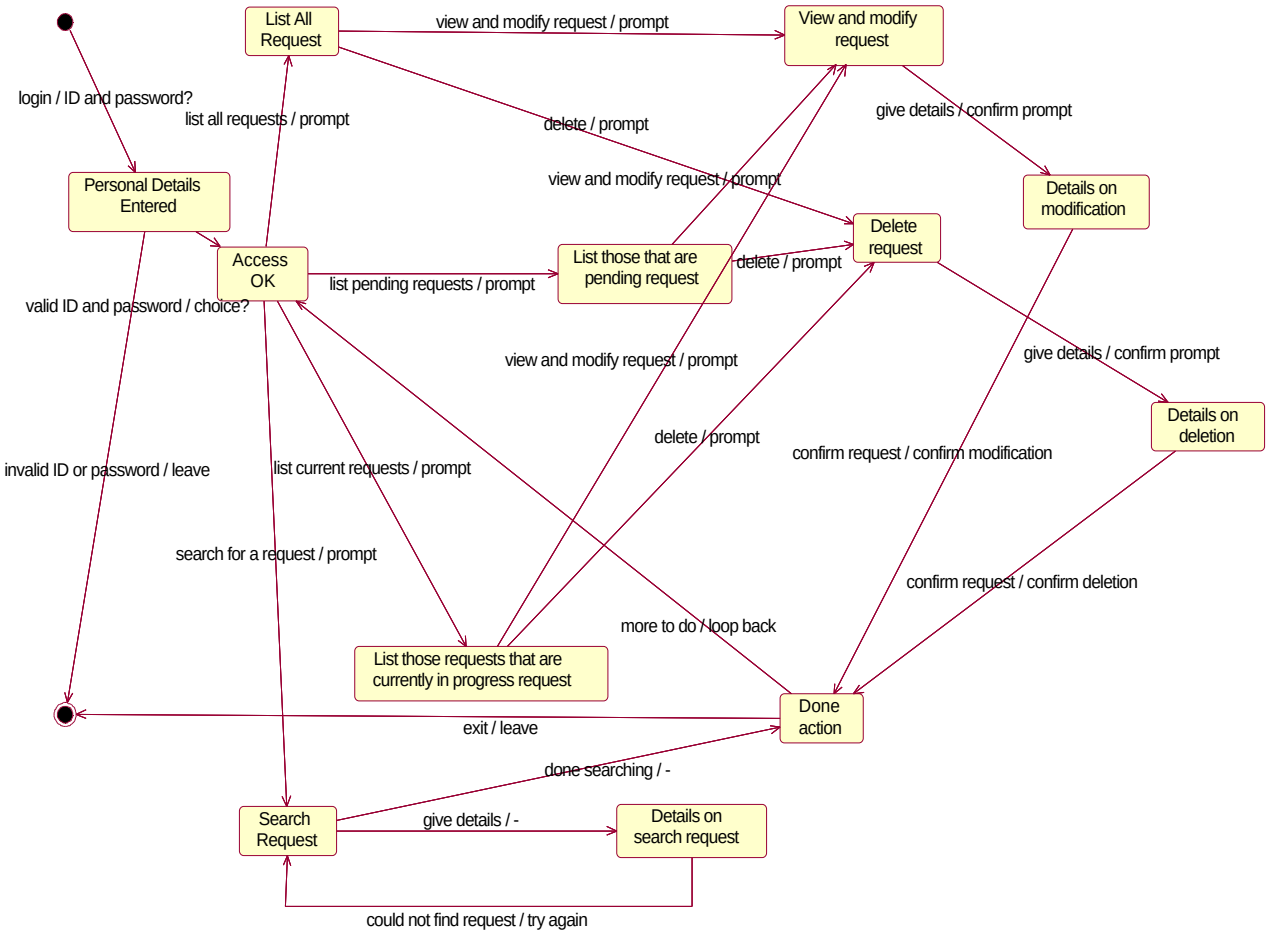
6.5.1. Dialogue Structures for User Groups



The general user is the general staff worker who uses a computer. They are the main drivers behind the input of the system. After authentication/login, they have the option to make a new support request when an IT problem occurs, or they can view the list of submitted requests.

In making a new request, there is confirmation that the system received the request. In listing the submitted requests, a general user has the option to choose a particular request and cancel, modify or just view the status of that particular request. On cancellation and modification, there is a system confirmation/feedback notifying the user that there has indeed been a cancellation and modification recorded. Once a user has finished with his/her query, he/she can conduct more queries. Otherwise, he/she can just leave the web-based system.

Dialogue Structure for the IT Support Personnel Interface



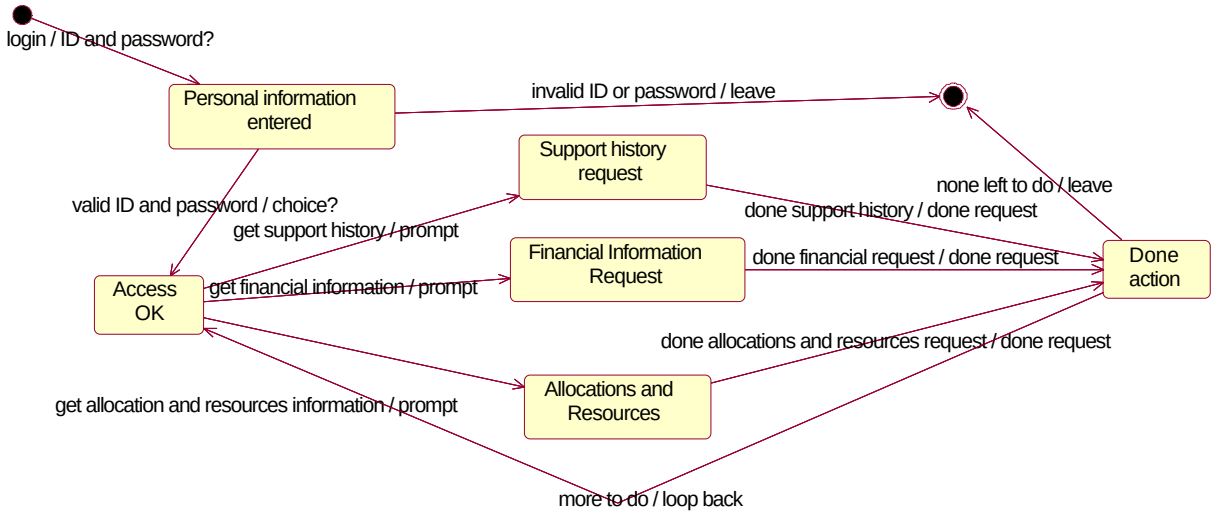
The IT Support Staff fix all the problems that have been submitted by the user.

After authentication, there are options for the Staff to list all requests in the database, list the pending requests (those that have not started), list the current requests (those that have begun), or to search for a particular request. For each item in each of these lists, there is in turn the option to view it or to delete it from the database. Once the query has been accomplished, there is again the option of conducting more queries.

A particular point to note is the search request: if details have been given but the database does not include the searched request, then there is an announcement to the user that the request could not be found and that the user should try again.

After all queries are complete, the IT Support staff can exit the database.

Dialogue Structure for ITManagerInterface



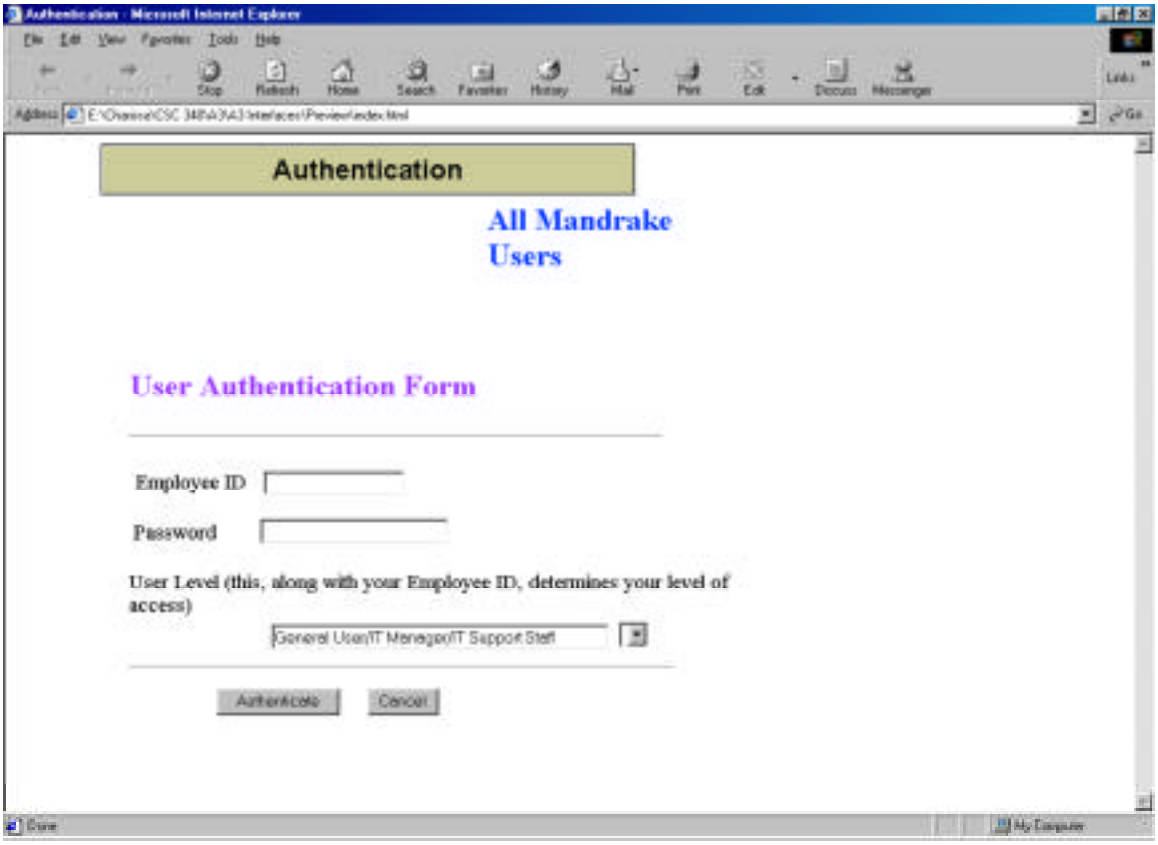
The IT Manager is involved in the management aspect of the system. He/she will be responsible for those resources and allocations that maintain the system. For example, he/she will review weekly/monthly reports on the efficiency of the system; how much time is spent on fixing hardware problems; what is better strategically in terms of investing money into different areas (such as more hardware, or more support staff, etc.).

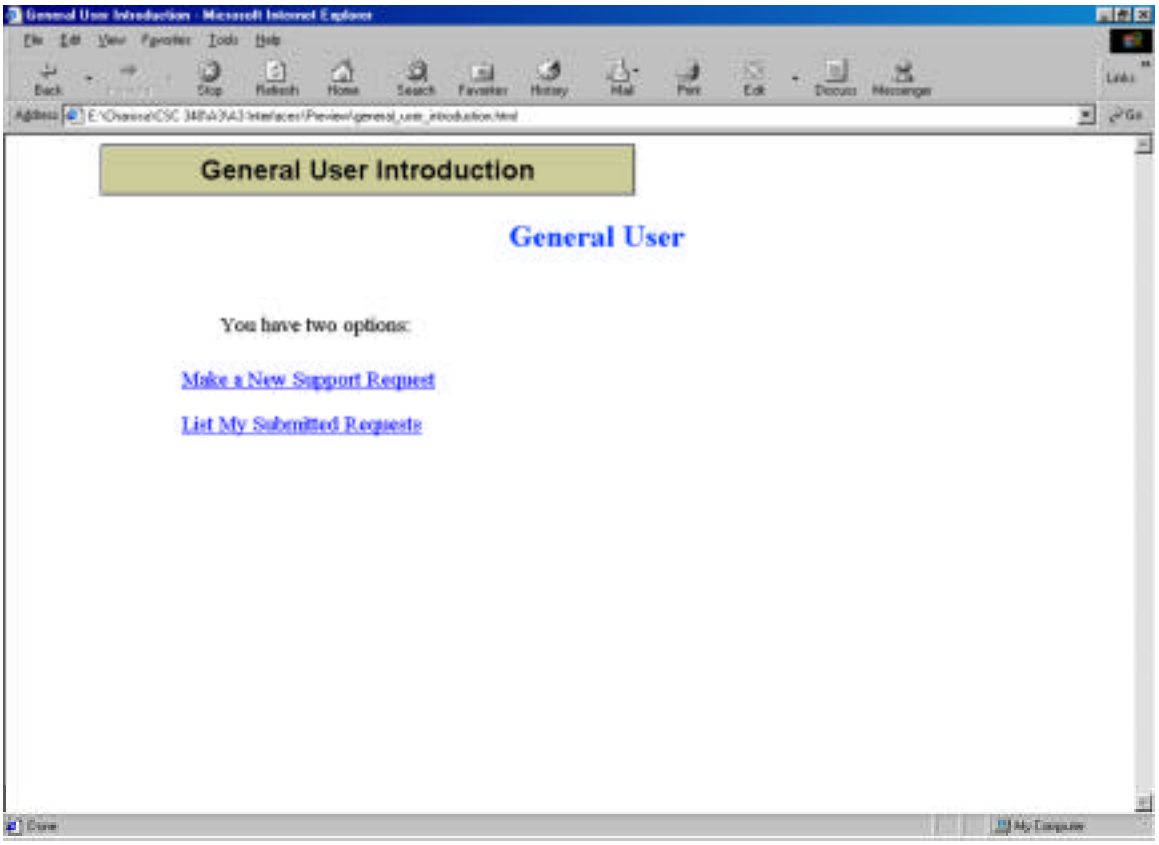
After authentication, the IT Manager will be able to access information from the database system pertaining to statistics that have been gathered - problems that IT Support Staff have experienced, financial information and the history of allocations and resources that the IT Department has invested or has yet to invest.

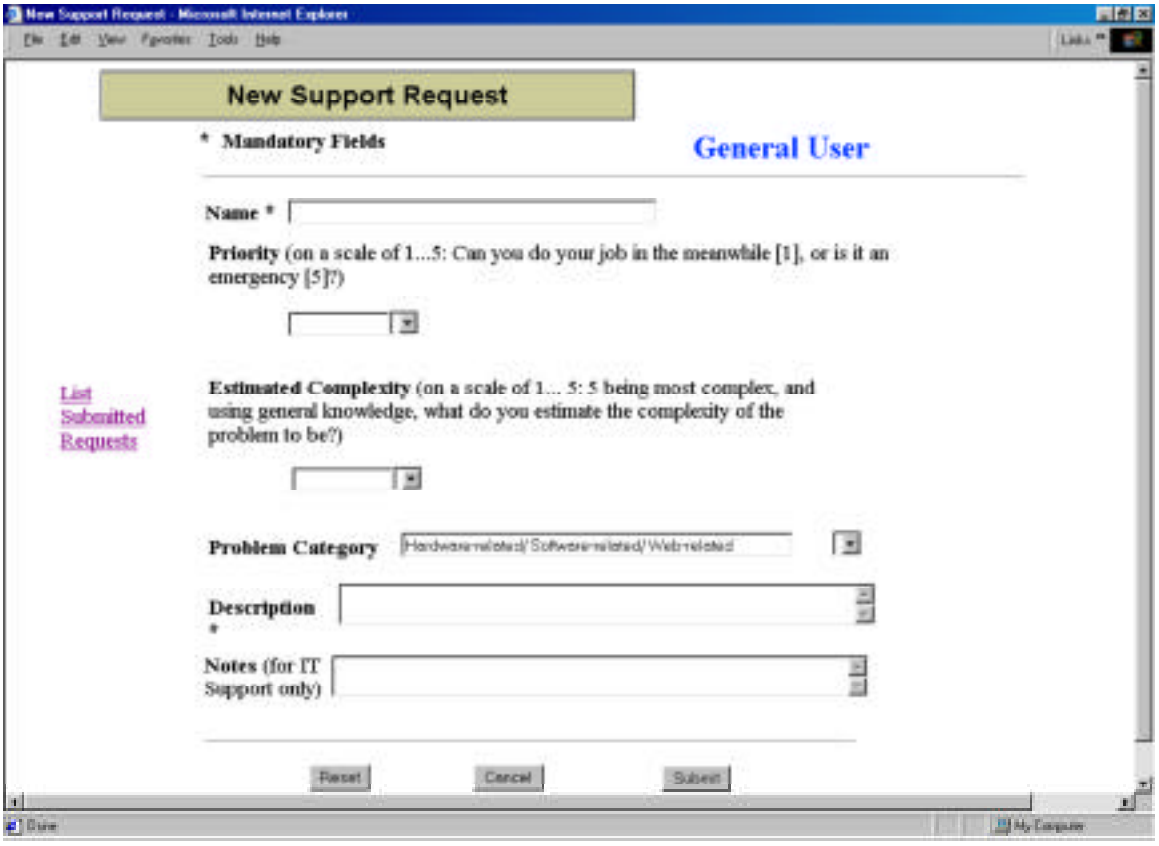
After a query has completed, the Manager can choose to either quit the system or continue conducting more queries into the system.

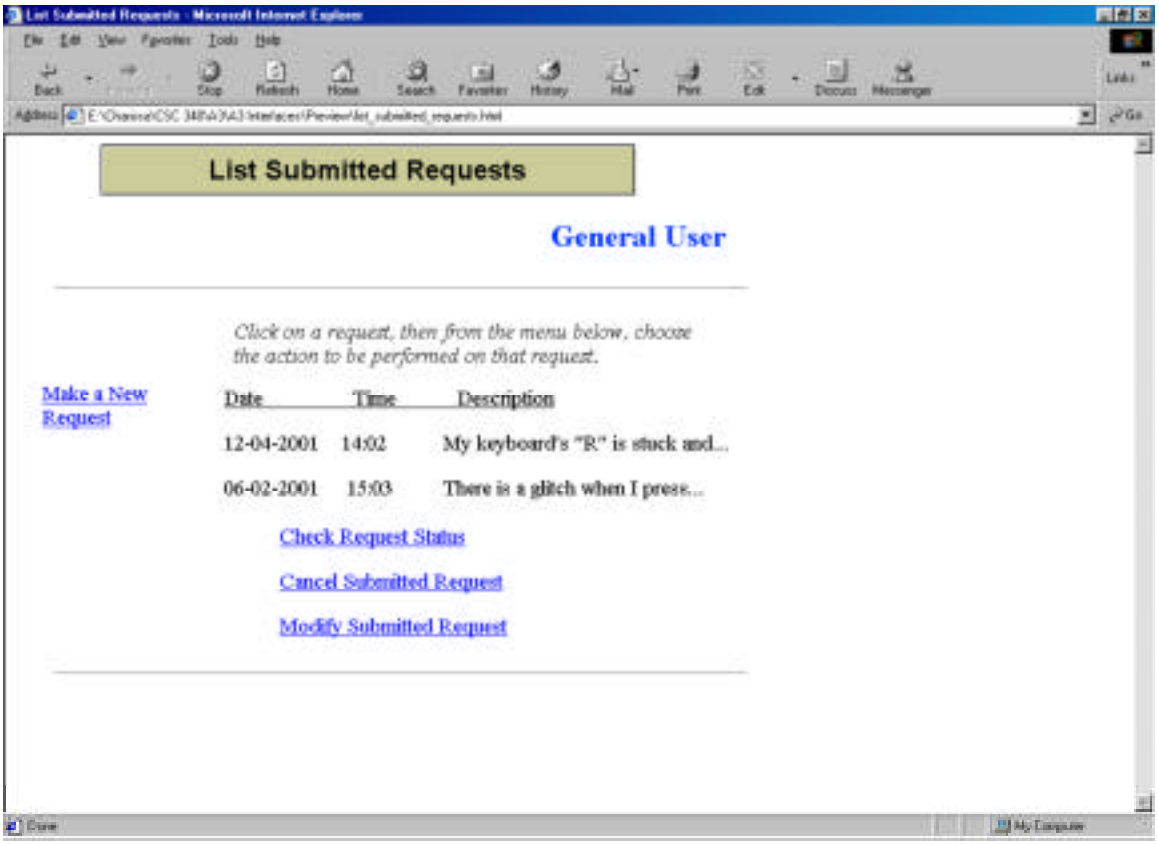
6.5.2. Mock-ups of Windows

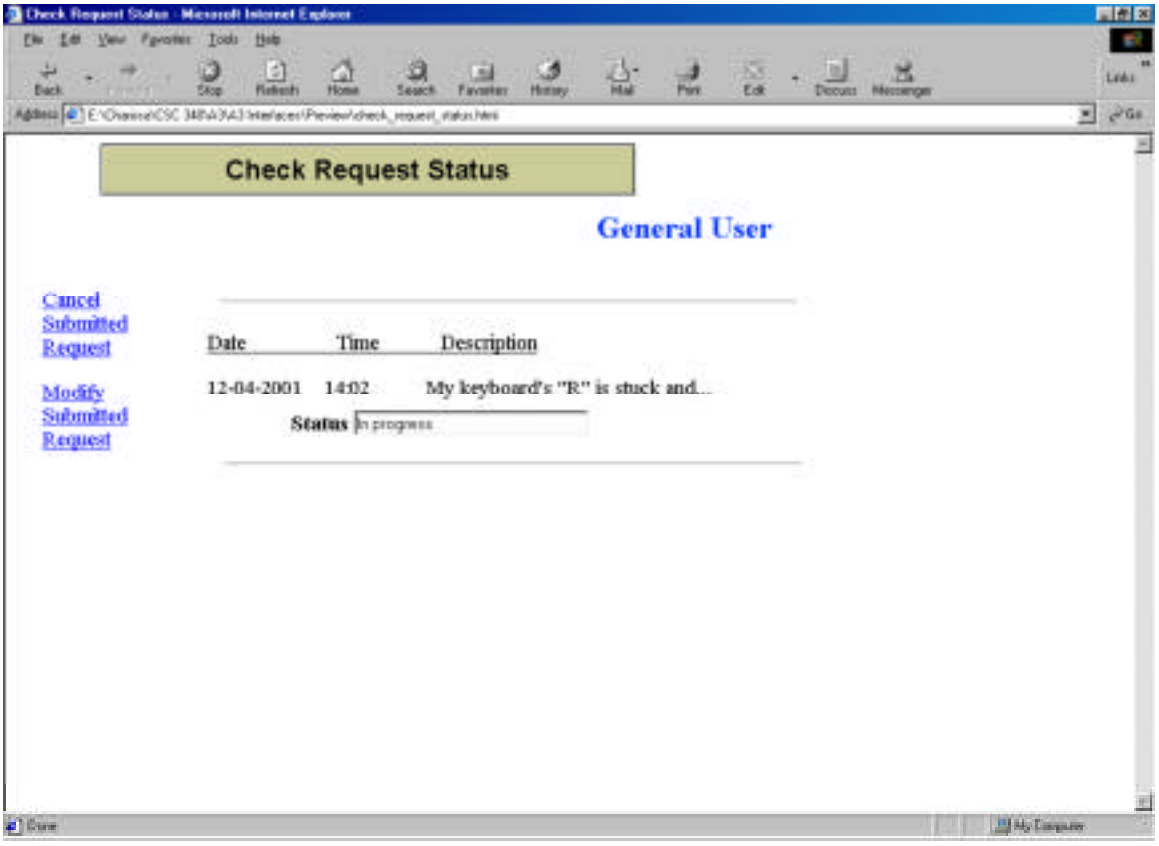
Please see Main Report section for full descriptions.

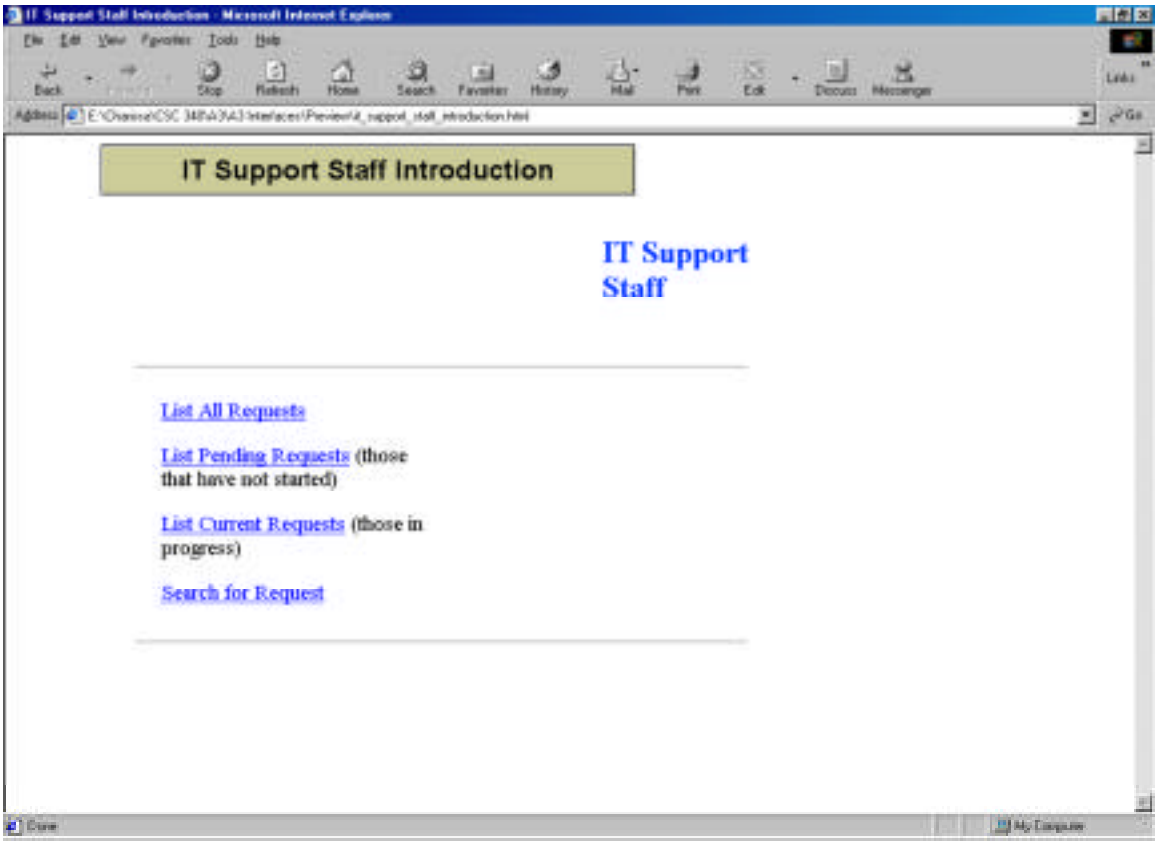


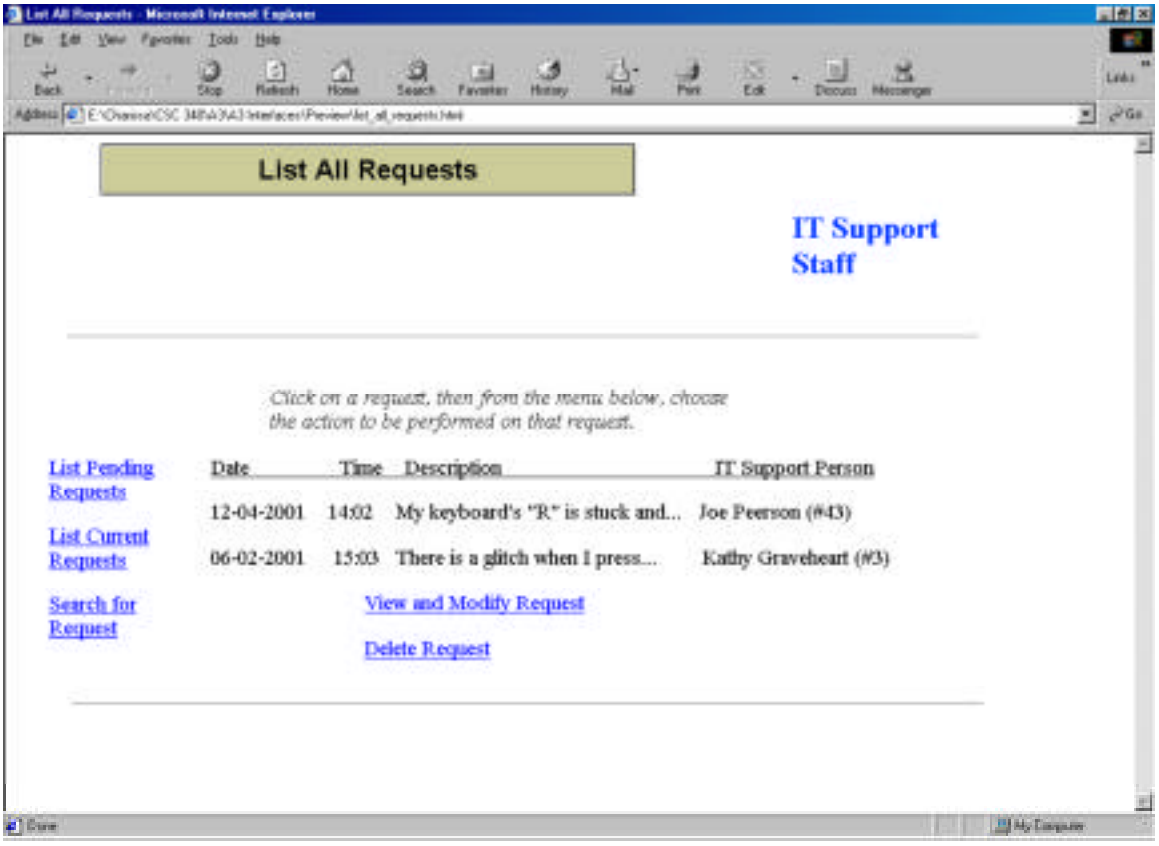

General User Introduction
General User
You have two options:
Make a New Support Request
List My Submitted Requests

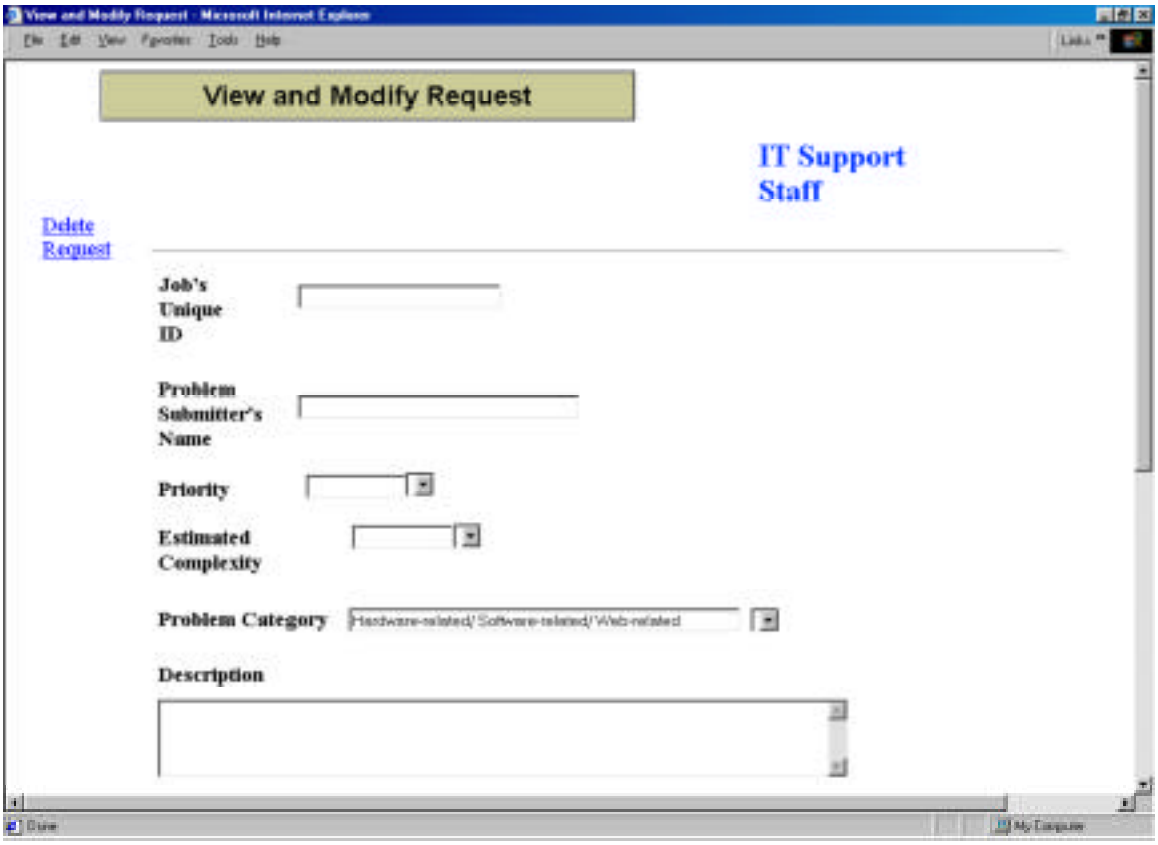

New Support Request
* Mandatory Fields
General User
Name *
Priority (on a scale of 1...5: Can you do your job in the meanwhile [1], or is it an emergency [5]?)
List Submitted Requests
Estimated Complexity (on a scale of 1... 5: 5 being most complex, and using general knowledge, what do you estimate the complexity of the problem to be?)
Problem Category
Description *
Notes (for IT Support only)
Reset
Cancel
Submit


List Submitted Requests
General User
Click on a request, then from the menu below, choose the action to be performed on that request.
Make a New Request
Date Time Description
12-04-2001 14:02 My keyboard's "R" is stuck and...
06-02-2001 15:03 There is a glitch when I press...
Check Request Status
Cancel Submitted Request
Modify Submitted Request

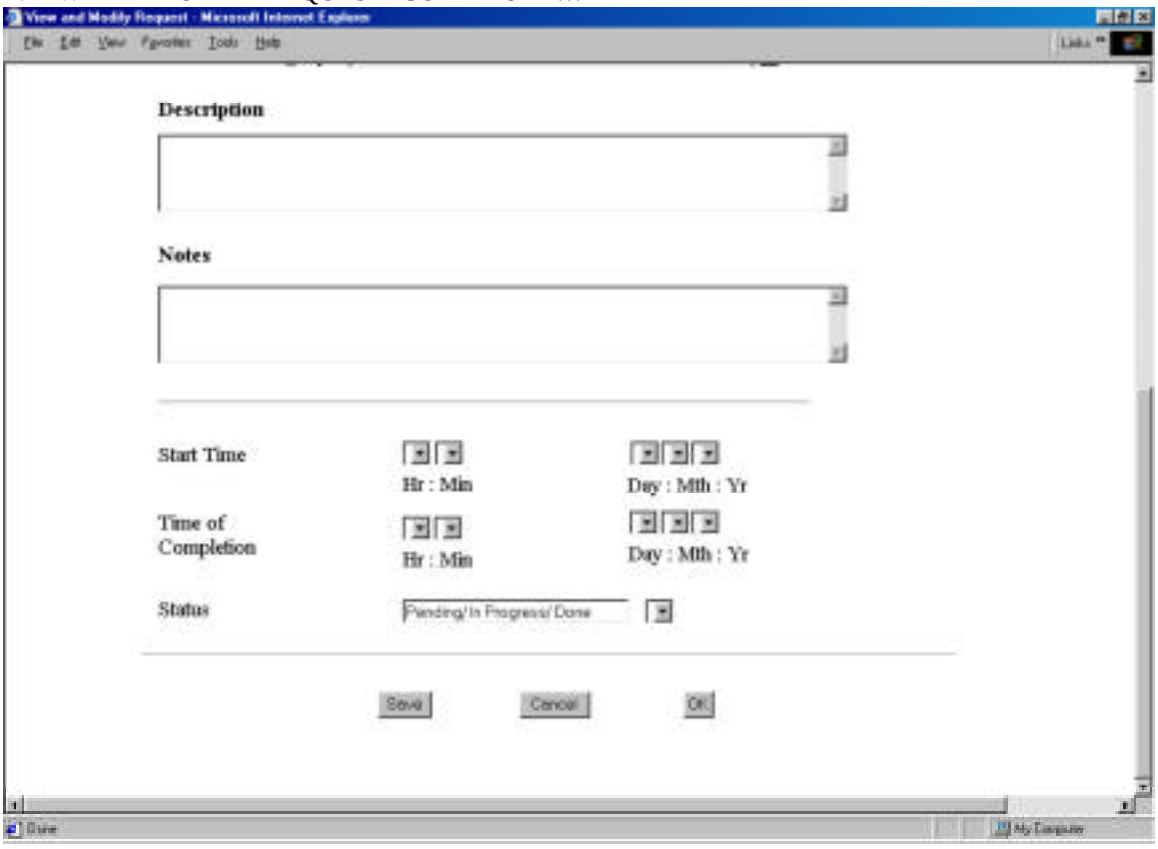

Check Request Status
General User
Cancel Submitted Request
Modify Submitted Request
Date Time Description
12-04-2001 14:02 My keyboard's "R" is stuck and...
Status In progress

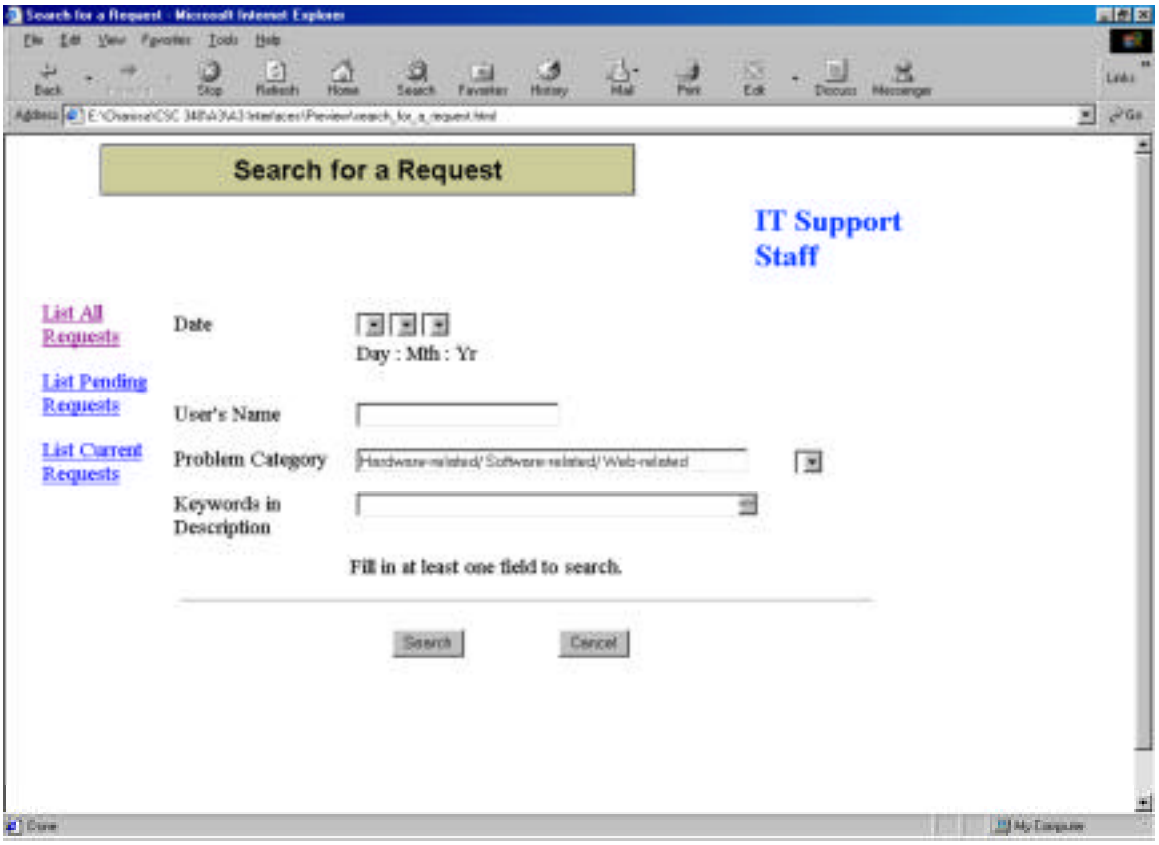

IT Support Staff Introduction
IT Support Staff
List All Requests
List Pending Requests (those that have not started)
List Current Requests (those in progress)
Search for Request

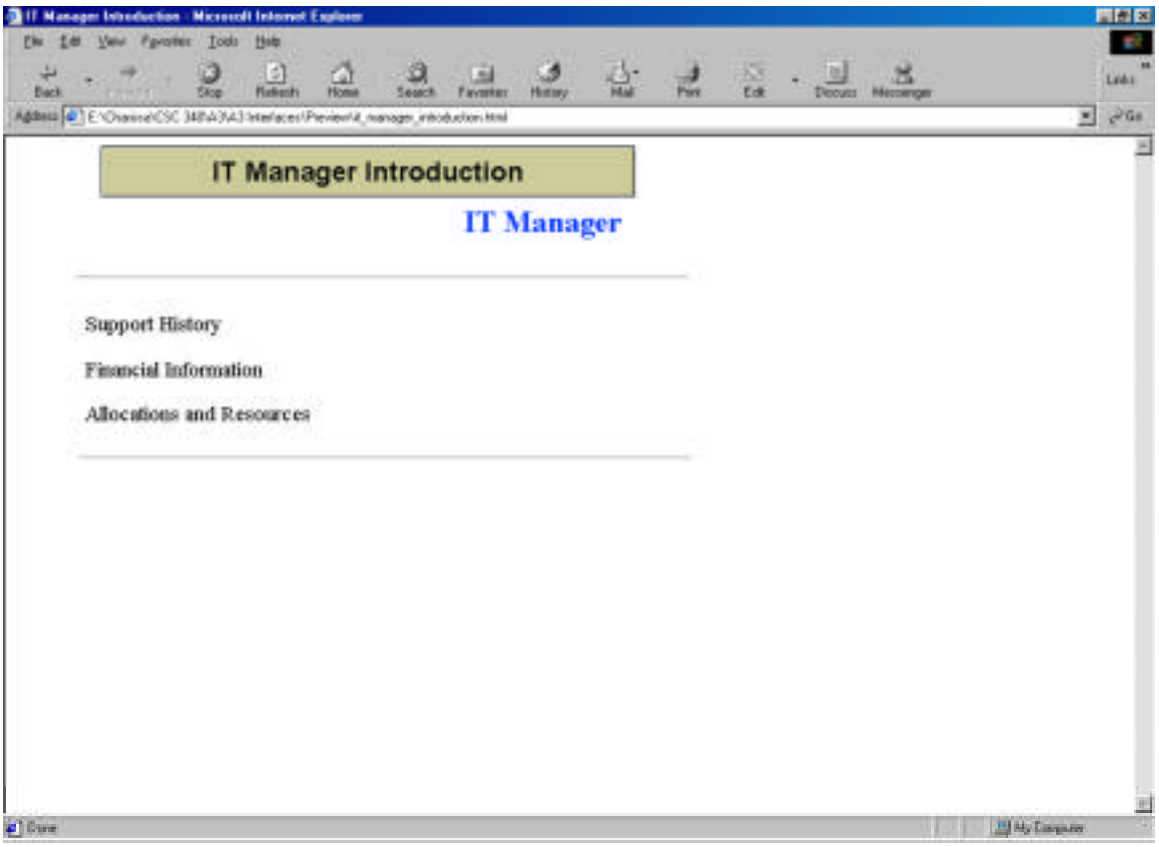

List All Requests
IT Support Staff
Click on a request, then from the menu below, choose the action to be performed on that request.
List Pending Requests
List Current Requests
Search for Request
Date Time Description IT Support Person
12-04-2001 14:02 My keyboard's "R" is stuck and... Joe Peerson (#43)
06-02-2001 15:03 There is a glitch when I press... Kathy Graveheart (#3)
View and Modify Request
Delete Request



'VIEW AND MODIFY REQUEST' CONTINUED…



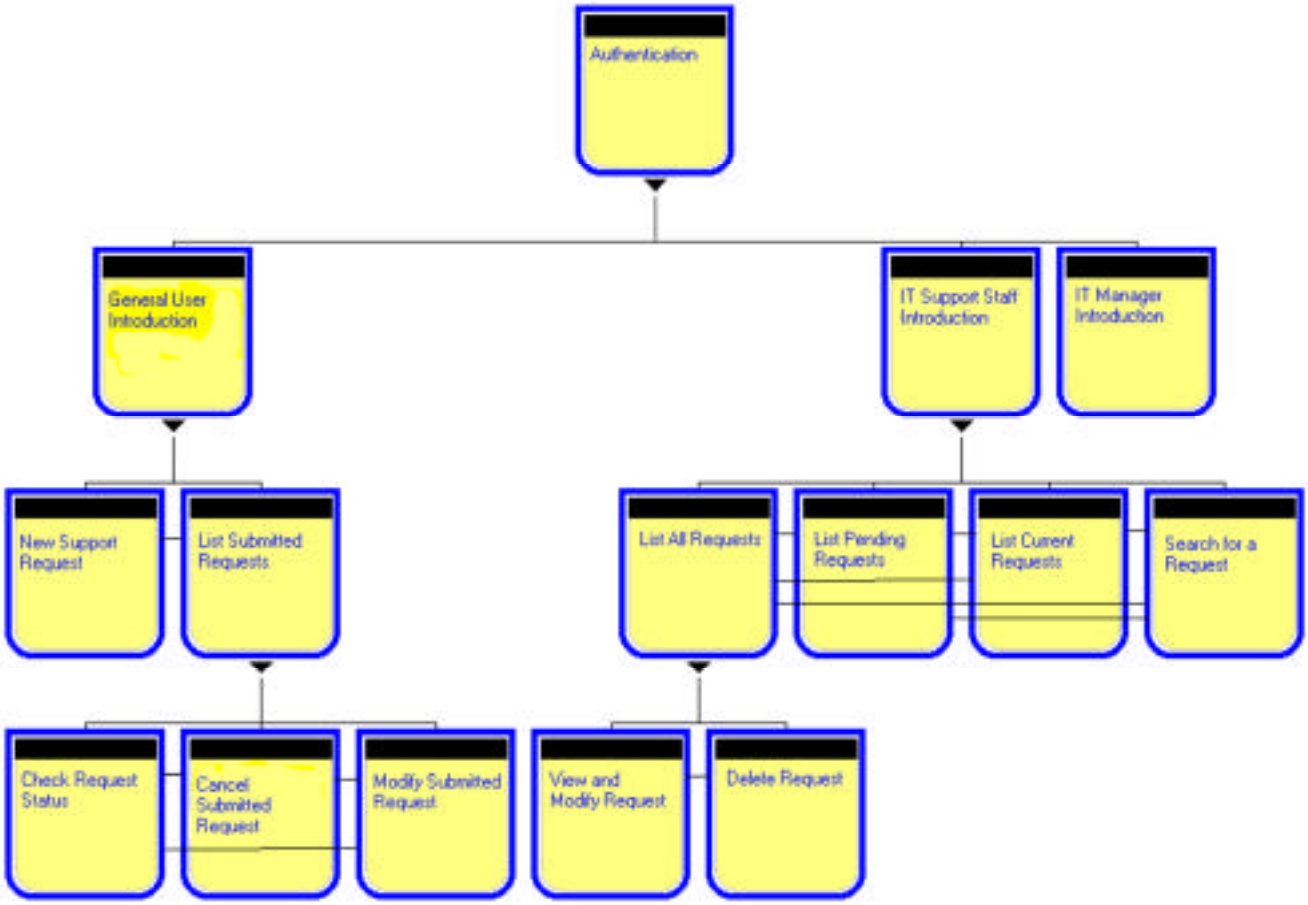

Search for a Request
IT Support Staff
List All Requests
List Pending Requests
List Current Requests
Date
Day : Mth : Yr
User's Name
Problem Category
Hardware-related/ Software-related/ Web-related
Keywords in Description
Fill in at least one field to search.
Search
Cancel


IT Manager Introduction
IT Manager
Support History
Financial Information
Allocations and Resources

6.5.3. Website Design

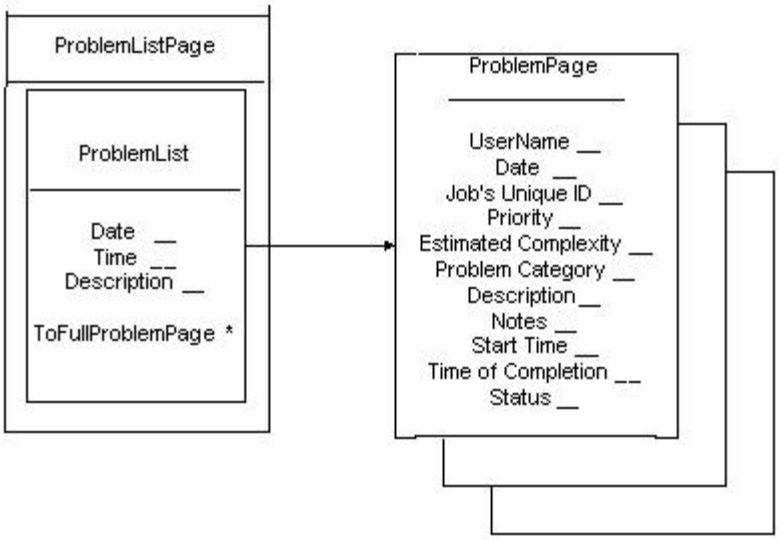
6.5.3.1. Site Map

This site map shows the overall structure of the pages and their relationships.



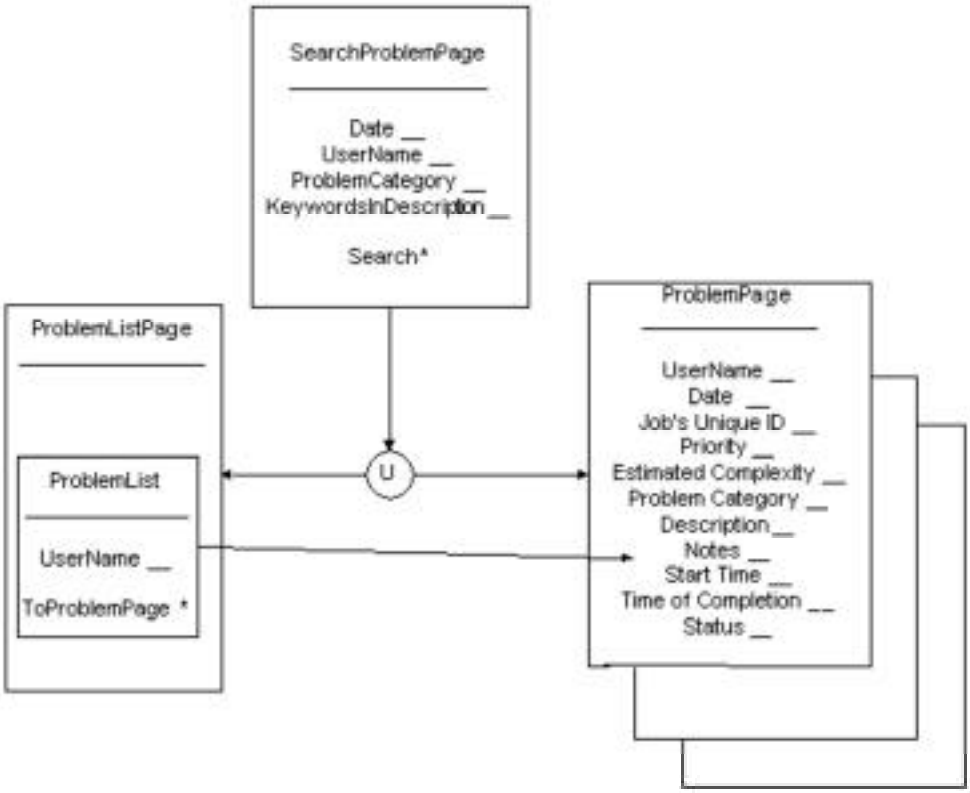
6.5.3.2. The Page Schema in the ADM Schema

This page schema indicates the manner in which one can access the full details of a given request. From the list of all requests, one can arrive at the full page of the request by following a link given only partial information. IT Support Staff would use this mechanism in the *List All Requests* page to get to the full details in the *View and Modify Request* page.



6.5.3.3. The Heterogeneous Union and Form in the ADM Schema

This illustration shows the mechanism used in searching for a particular problem. Given partial details on a certain request (such as date, name of submitter, type of problem or keywords in the description), the system will search the list of all its problems and return the full details of the matched problem. The IT Support Staff will use this mechanism at Mandrake when they are in the *Search for a Request* page.

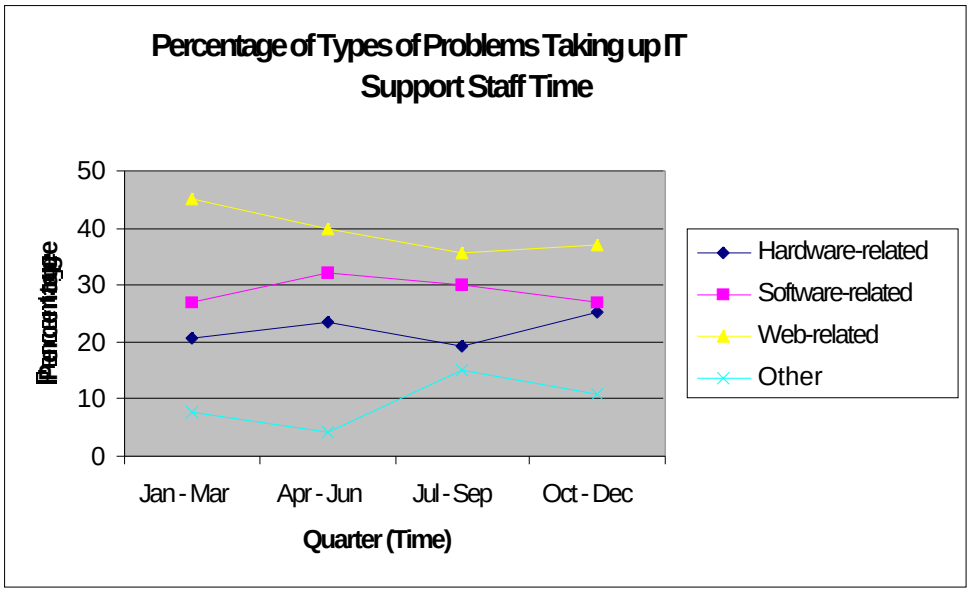


6.5.4. Output Design Charts

Percentage of Types of Problems Taking up IT Support Staff Time

		Problem Category			
		Hardware-related	*Software-related*	*Web-related*	*Other*
Quarter	*Jan - Mar*	20.5	26.8	45	7.7
	Apr - Jun	23.6	32.2	40	4.2
	Jul - Sep	19.2	30	35.8	15
	Oct - Dec	25.3	26.8	36.9	11

The above chart translates into the following graph:

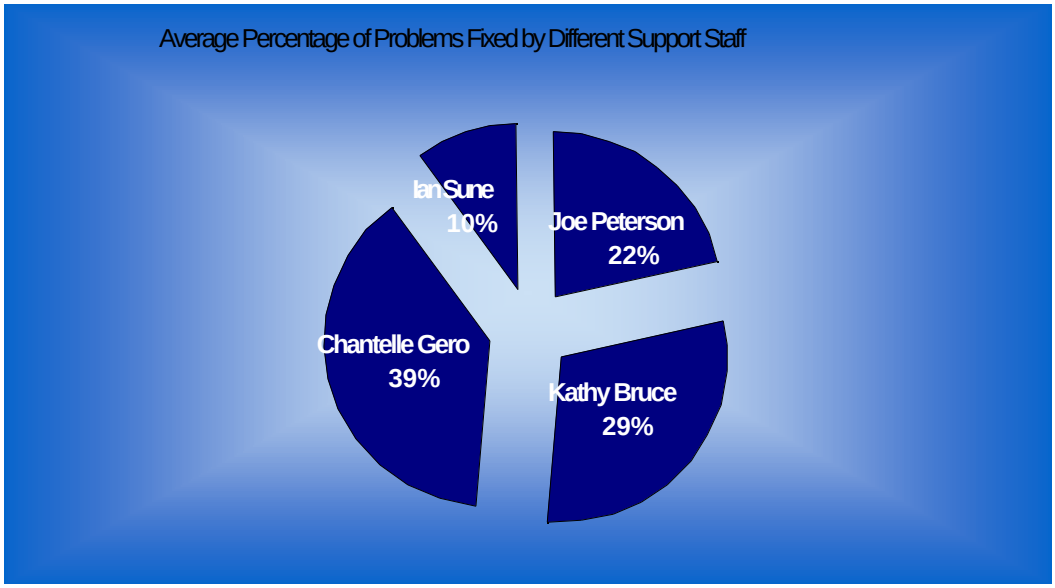


The chart of Percentage of Types of Problems Taking up IT Support Staff Time indicates the time taken by each category of problems on a quarterly basis throughout the year. Web-related problems are at the forefront of problem categories occupying IT Support Staff time in each quarter. The IT Manager can conclude that more money or resources should be invested into the improved performance of the Web-related infrastructure for the company.

Average Percentage of Problems Fixed by Different Support Staff

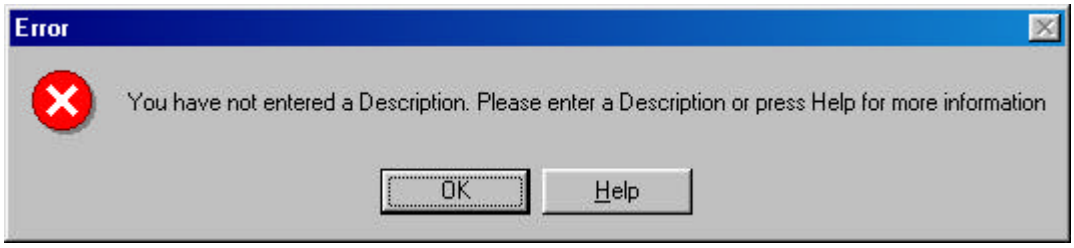
		IT Support Staff Worker			
		Joe Peterson	Kathy Bruce	Chantelle Gero	Ian Sune
Time Duration	Jan - Dec	22.2	30	39.9	10.1

The above chart translates into the following graph:



The second chart, Average Percentage of Problems Fixed by Different Support Staff, shows that throughout the year, Chantelle Gero, fixed most of the problems submitted to the IT Department. A conclusion to this is that she may get a pay raise, or other members of the team must be asked to improve their performance.

6.5.5. Error Message Popup for Internal Controls for Inputs



An example of data validation for input or error checks is in the 'Description' field of the General User -> New Support Request web form. These fields are mandatory and should they not be filled on submission; an error message controlled by JavaScript will report failure of submission.

As is evident, messages such as the above should be helpful and intuitive for the computer user to guide him/herself along. A 'Help' feature is often a useful feature for eradicating user confusion.

6.6. Appendix: Supporting evidence

6.6.1. Interview with Tom Metaxas

Wednesday, April 11, 2001

CMR: -- Hi Mr. Metaxas, the third phase of our project is to do detailed system design and to build a database to store all the support requests in particular.

TM: -- Well, definitely, a properly designed database would be a very important feature for us. How can I help you?

CMR: -- To estimate the capacity of our database, we have to specify some quantitative parameters…

TM: -- Of course, I know what are you talking about. Roughly there should be about 80 employees, only one IT Manager (that's me), and about two support people. No, put five because we are hiring some new specialists at the end of the month.

CMR: -- Now lets talk about load. Are there any changes between the current situation and the situation at the time of our first interview in terms of how many requests you have to handle?

TM: -- Not really, every week I support 100 problems on average, which is about 20 per day. It takes more than 20 hours per week for me to deal with all the requests, while my available time is at most 2 hours a day.

CMR: -- Assume you have a fully working database. How often do you want to generate statistical reports?

TM: -- Well, weekly is a good frequency since every Monday we have a brief meeting, so this report might be an interesting issue to discuss on those meetings.

CMR: -- For how long do you want to keep old records in the database?

TM: -- Three years maybe…

CMR: -- Then according to our approximate calculations (100 requests per day * 365 days * 3 years) the database will be able to hold about 100 000 records…

TM: -- Yes, that's more than enough.

CMR: -- And the last question. How often, do you think, users will check the status of their requests and support people will list all the pending requests?

TM: -- Probably a couple of times per support request. Users definitely will be checking the status of their requests quite often [laughs].

CMR: -- Mr. Metaxas, thank you for your time and all the provided information.

TM: -- Thank you guys, hopefully the system you are designing will solve all our problems. Take care!

6.7. Supporting Documentation

Team meetings:

Sunday, March 18

Topics discussed: General structure of the assignment, task decomposition.

Plans for the next meeting: Generate ideas about different components of the assignment.

Sunday, March 25

Topics discussed: Global architecture, hardware and software selection, revision of requirements specifications.

Plans for the next meeting: Have requirements specifications, hardware and software selection ready. Work more on global architecture. Produce class, sequence and state diagrams.

Saturday, April 7

Topics discussed: Global architecture continued. Detailed Program Design.

Plans for the next meeting: Discuss interface design. Keep working on program design. Revise produced class, sequence and state diagrams.

Sunday, April 8

Topics discussed: Detailed interface design.

Plans for the next meeting: Make research for database section. Interview Tom Metaxas. Make changes to class diagrams.

Wednesday, April 11

Topics discussed: Pre-interview questionnaire

Interview with Tom Metaxas.

Plans for the next meeting: Have database class diagram and operations ready. Complete tables of volumes, operations and accesses.

Thursday, April 12

Topics discussed: Database load finished. Revised class diagrams.

Plans for the next meeting: Integration of all the parts of the assignment.

Friday, April 13

Topics discussed: Missing report parts are written.

Plans for the next meeting: Edit report.

Monday, April 16

Report is assembled together and proof read. Diagrams converted into Word format.

6.8. Appendix: Team Report Form

Description of roles and contributions of each team member:

This assignment is the result of equal intellectual work of all members of the group.

Name	% of team Effort	Signature
Charissa Lai	33	C. Lai
Martin Ross	33	M.Ross
Ruslana Goncharenko	33	R. Goncharenko

Date submitted: April 16, 2001