Smart-Assist: A Reminder System

Yuebin Li

A Major Report

in

The Department

of

Computer Science

Presented in Partial Fulfillment of the Requirements

For the Degree of Master of Computer Science at

Concordia University

Montreal, Quebec, Canada

March 2002

0-612-68472-5

ABSTRACT

Smart-Assist : A Reminder System

Yuebin Li

The objective of this project is to develop a reminder system to remind people of important events. Reminder systems are very important and popular. However, there is no general reminder system currently available. Almost all existing reminder systems are developed for special tasks. This report focuses on developing a reminder system called Smart-Assist, which has all the features of current existing reminder systems.

The major distinctive features of Smart-Assist are:

- An Open Source based system. The use of existing Open Source reduces the cost and promotes its use.
- A database based application. All the information is stored in a MySQL database and hence persistent.
- A system which runs either locally or though the web, but data are made persistent by sharing the same database.
- A real time system. Event trigger and timing depends on the system time.
- An automatic event driven alarm system with audio playing feature.

Acknowledgment

First, I would like to express my deepest respect and gratitude to my supervisor, Dr. Bipin C. DESAI, who spent many hours in discussion with me and gave me valuable suggestions, encouragement and motivation to help me in all stages of this report. I have benefited greatly from his knowledge, patience and disciplined approach. I very much thank him for giving me the opportunity to work with him on my project for over a year.

Secondly, I would like to thank M. S. Weiqi and Dong for providing me with new ideas during my work.

Thirdly, I would like to thank Dr. Zonglinag Mu, who patiently listened to me and helped me in editing and formatting this report.

Finally, I would like to thank my wife Qi Liu. It was her support that made this project possible.

Table of Contents

List of Figures

List of Tables

Chapter 1. Introduction

1.1 Overview of reminder system

Reminder system is an event reminding agent. It simplifies time management by enabling people to register important events for future reminding. Basically, it includes three aspects: it can be a time scheduler; it can be a notebook to write down important events for future review; it also can be an event alarm clock to alert you to an event.

1.2 Objective and Scope of the project

In this project. a reminder system called Smart-Assist is developed. With this system, users can input and edit events that they want to be reminded of, and select how they want to be reminded. Open Source is used to reduce the cost of development. The major features of Smart-Assist are:

- An Open Source based software system. The use of existing Open Source reduces the cost and promotes its use.

- A database based application. All the information is stored in MySQL database and hence persistent.

- A system which runs either locally or through the web, but data are made persistent by sharing the same database.

- A real time system. Event trigger and timing depends on the system time.

- An automatic event driven alarm system with audio playing feature. Smart-Assist provides an interface through which an alarm is set at event time. The alarm is set automatically by Smart-Assist and triggered by a system clock. The alarm sound is not simple audio sound; it is mp3 music played with RealPlayer, which can be selected by the user when the event is configured. The RealPlayer is integrated into the system and no separate installation is necessary.

1.3 Organization of the report

Chapter 2 provides a review of current reminder systems. Two main reminder systems will be introduced: "Setltor's Lexacom Enterprise Calendar" which runs on Unix platforms, and "DesktopSticker" which runs on Windows.

Chapter 3 presents the distinctive features of Smart-Assist and its GUI interface.

Chapter 4 describes the design and implementation of Smart-Assist. It introduces the architecture of the system and the design of the data sharing interface, discusses details of

Smart-Assist design, describes the major techniques used in this system and its implementation.

Chapter 5 gives some tests and results of Smart-Assist to show how it meets the goals of this system.

Chapter 6 presents the conclusion and future potential for implementing Smart-Assist.

Chapter 2. An introduction to current reminder systems

2.1 Overview of current reminder systems

Reminder systems are widely used in order to manage our schedules well and help us work effectively. A number of reminder systems have been developed to help in appointment management, daily business management, and medical services, etc. They share the similar functions such as a Calendar function which schedules events with date and time. However, almost all current reminder systems are platform dependent. Some of them have an alarm function, but most of the alarms are simple sounds. Many reminder systems use files to store information. Therefore, it is difficult for them to deal with a large amount of records. Besides, current reminder systems are developed for commercial purposes and are intended for special tasks.

2.2 Two popular reminder systems

One popular reminder system is Setltor's Lexacom Enterprise Calendar [21]. It is a web server application, which runs on Unix platforms.

Lexacom Enterprise Calendar is a platform-independent time management solution for Business-to-Business (B2B) service providers. Users can access their Agendas from any desktop computer or network computer equipped with a Web browser. Quiet sign-in allows users quick access to their calendars.

Lexacom Enterprise Calendar must be installed on a machine with an HTTP server running mod_fastCGI. Although it can run on the same machine as Lexacom Enterprise Calendar Server, this is not the optimal configuration unless the user base is very small. Ideally, both Lexacom Enterprise Calendar Server and Lexacom Enterprise Calendar should be installed on dedicated hosts, tuned for the expected usage on the site. Using Lexacom Enterprise Calendar, users can create their events in it for future review and be allowed to modify certain preferences, such as time format, date format etc.

Another example of a reminder system is DesktopSticker[23], developed for Windows98. It runs only on Windows98 platforms. It has an alarm function, but the alarm time has to be set for events.

For DesktopSticker, a user can create some events and store them in the system for future review, set the alarm time for the system, and edit the event note.

2.3 Features of current reminder systems

From the reminder systems above, we summarize the major shared features as follows:

- Most of them have basic event calendar function
- Most of them are platform dependent
- Some of them have alarm function with simple sound
- All of them are developed for special tasks.

Chapter 3. The need for Smart-Assist

3.1 Requirements and specifications

The primary objective of this work is to develop a general reminder system which is a real time interactive system with a calendar function.

The alarm function is a basic function which need to be included in Smart-Assist. In Smart-Assist, the alarm function is designed to be different from a general alarm function, such as a clock alarm. It is an event driven alarm and controlled by event time automatically. The alarm is set off at event time by Smart-Assist. The sound of the alarm is also designed to be configurable mp3 music rather than simple sounds. It is played and controlled by RealPlayer. The event's description also appears in the RealPlayer. The RealPlayer is independent and executed by Smart-Assist.

Smart-Assist is designed to be capable of running both locally on Windows or through the web. It can be used as a local application or as a web application which can be accessed from any computer using a web browser.

Smart-Assist is developed using Open Source software which is freely available and thus reduces the cost of development. Open Source software such as GNU's GTK+, MySQL and PHP are seamlessly integrated into the system. Smart-Assist is implemented in C.

3.2 Components of Smart-Assist

According to the requirements of this project, Smart-Assist should be a locally enabled system on Windows and a web enabled system. It is designed to include two sub-systems: a local application sub-system and a web based sub-system. Since both the local application and web application belong to the whole system and they share the same database, it is reliable to use a flag in a database table to control the persistence of the two sub-systems.

GTK+ and C are used in implementing the local application sub-system, since GTK+ is a C based GUI toolkit, it supports any user program written in C.

PHP, JavaScript and HTML are used in implementing the web based sub-system. JavaScript and PHP are tightly combined in HTML to take advantage of their features. PHP is the server side script programming language embedded in HTML [1,5,16]. It makes the web page more dynamic by using the general web page in the server side with dynamic data information. JavaScript can make the interface appearance attractive on the client side.

By using PHP, JavaScript and HTML, the web based sub-system is also made into a cross platform system because PHP, JavaScript and HTML are all portable programming languages and have standard functions that can run on both Windows and Unix.

Since Smart-Assist is a multi-user system, security is at risk, all personal information should be prevented from being viewed and changed by others. This security issue is solved by using an administration function to group the users by User_ID and Password. Security check is ensured that any personal information cannot be viewed or changed by others without permission.

On the other hand, for web based sub-system, all the pages are separate files on the server which can be easily seen by everyone. It is possible that an unauthorized user can directly access web pages without security checking so that some personal information can be seen and changed. Therefore the security between web pages must be implemented.

PHP session control is used with a user name and password to ensure the security check. After the user name and password are confirmed by the system, a session will be used for user id to identify who is the current user, then this session variable is checked by every page to prevent an unauthorized user from accessing the web page without passing through session control.

The database MySQL used in Smart-Assist resides on a Unix platform. The application program may not be on the same machine as the database. The connection between an application program and database that are on different machines should be considered.

Fortunately, MySQL has native API for C and PHP to make it easy to connect from the application program, whether they are on the same machine or not.

Playing audio sound is simple because there are many well-written sound playing applications (RealPlayers) available. However, passing the event information into these applications is difficult.

For the alarm function, the solution is to modify a RealPlayer to include event information which can be displayed while it is playing the music.

Due to the requirements, local application sub-system and web based sub-system perform similar tasks. However, since the software which is used in developing Smart-Assist in the two sub-systems is different, the design uses a different approach for local application sub-system and web based sub-system.

Development of the local application sub-system is a traditional programming task. All function handling is within the entire program, and the data is transferred between the program and database. Development of the web based sub-system is server side programming, task function handling is in each page, and each page is a different PHP program.

Flexibility is very important in software design. Smart-Assist uses Object Oriented Design methodologies to group the same kinds of tasks in classes, and treats everything as object. The local application sub-system of Smart-Assist is divided into two main tasks (two classes): the Calendar function, and the audio process for RealPlayer. Each has its own complex functions and is associated with the alarm function. The Calendar function part executes the RealPlayer part for whatever alarm time is coming and writes the event information and audio file name into two text files. When RealPlayer is executed, it reads the event information and audio filename from the text files, and concurrently displays the event information and plays the audio.

In the local application sub-system, the RealPlayer is a separate part from the system. As an independent part, RealPlayer can be executed by command line function, and then it gets upcoming event information from files. Hence, any future change of RealPlayer will not affect other parts of the local application sub-system.

3.3 Software used

3.3.1 MySQL

Database management plays a central role in computing. MySQL is a very fast, efficient and well featured database system, which can handle large amounts of data. It handles most standard SQL constructs [3,15] (SQL stands for "Structured Query Language").

MySQL's speed and flexibility comes from the fact that it stores data in separate tables rather than putting all the data in one big storeroom [15]. The tables are linked by defined relations, which makes it possible to combine data from several tables on request.

Because MySQL is an Open Source Software, anybody can download MySQL from the Internet and use it without cost, and anybody can study the source code and change it to fit their needs. MySQL uses GPL (GNU General Public License http://www.gnu.org) which defines what is permitted and what is restricted in using the software.

UNIX is far more stable than Windows in a shared-server, or multi-user environment. Windows NT/2000 was not developed for the multi-user, shared-server hosting environment (which can be a highly demanding environment at times), whereas UNIX was developed under these conditions, and MySQL is developed for UNIX operating systems under the GNU license [6].

3.3.2 GTK+

The GUI is very important for the local application sub-system. It is the key that can render a project usable. A good GUI has a better look and feel, and can be easily used with many popular programming languages.

Since C and C++ are so popular, Most of the toolkits have C or C++ interfaces because object-oriented programming seems particularly applicable to GUI code. However, some GUI system, such as Fresco, attempt to be language neutral and potentially provide an interface to almost any language. These toolkits come as libraries that must be compiled and linked for each target system. On the other hand, interpreted languages, such as Smalltalk, Tcl and Java, can run on multiple systems without having to be compiled for each one. Toolkits written in C, such as GTK, can be called from many other languages such as Scheme, Python or Perl.

GTK (GIMP Toolkit) is a library for creating graphical user interfaces[7,8,14]. It is licensed using the LGPL license, the user can develop open software, free software, or even commercial non-free software using GTK without having to pay for licenses or royalties.

GTK was written for developing the GNU Image Manipulation Program (GIMP), but it has now been used in many software projects, including the GNU Network Object Model Environment (GNOME) project. It is built on top of GDK (GIMP Drawing Kit), which is basically a wrapper around the low-level functions for accessing the underlying windowing functions (Xlib in the case of the X windows system).

GTK is an object oriented application programmers interface (API) to implement using the idea of classes and callback functions (pointers to functions). There is also another component called GLib to replace some standard calls, such as some additional functions for handling linked lists, etc. It is used to increase GTK's portability, such as for some of the functions which are implemented but are not available or are nonstandard on other unixes such as g_strerror().

GTK is an event driven toolkit [14], which means it will sleep in gtk_main until an event occurs and control is passed to the appropriate function. It is done by using the idea of "signals". When an event occurs, such as the press of a mouse button, the appropriate signal will be "emitted" by the widget that was pressed. GTK does most of its useful work in this way. The signals are all widgets inherent, such as "destroy", and are widget specific, such as "toggled" on a toggle button.

For developing under other GUIs, GTK+ is really more different and very good in some cases, and GTK+ is a far simpler API for programmers working in C to get along with than anything else [7].

3.3.3 PHP

PHP is a server-side HTML-embedded scripting language [1,2,4,5]. It is different from a CGI script written in other languages such as Perl or C. Within HTML script, some embedded code of PHP does something instead of writing a program with lots of commands to output HTML. The PHP code is enclosed in special start and end tags to allow jumping into and out of PHP mode.

PHP is different from client-side JavaScript because the PHP code is executed on the server. All the things are done on the server; the client would receive only the results of running that script.

At the most basic level, PHP can do anything that CGI program can do, such as collect form data, generate dynamic page content, or send and receive cookies. But the strongest and most significant feature in PHP is its support for a wide range of databases to make it incredibly simple to write a database-enabled web page.

The following databases are currently supported by PHP:

Adabas D	InterBase	Solid
dBase	mSQL	Sybase
Empress	MySQL	Velocis
FilePro	Oracle	Unix dbm
Informix	PostgreSQL	DB2

Table 3.1 Database supported by PHP

PHP also has support for talking to other services using protocols such as IMAP, SNMP, NNTP, POP3, or even HTTP [4], and can also be used to open raw network sockets and interact using other protocols.

3.4.4 Comparison of PHP with other server side script languages

The four main server side script languages are PHP, ColdFusion, Active Server Page (ASP), and JavaServer Page (JSP). They are widely used because of their popularity, flexibility, compatibility and potentiality, and all feature tag-based server-side scripting.

By comparison, the advantages of PHP are:

- Both PHP and Perl were designed for scripting on the web but PHP is less confusing (stricter) and easier to integrate in existing HTML.

- PHP is faster and more efficient, more stable and less resource intensive than Cold Fusion and runs on many platforms.
- PHP is faster and more stable than ASP.
- PHP is a better and more complete object model than Java servlets/(JSP). PHP can instantiate and use Java classes or servlets.

Let us see their features in the following table:

	PHP	ColdFusion	ASP	JSP
Cost	Free	$0 (Express version)-$5000 (Enterprise version)	Free with Windows NT, cost for 3rd-party software	Free
Language In Page	PHP	CFML	VBScript, JScript	Java
OS Platform	Unix (Linux), Windows, MacOS, OS/2	Windows NT, Solaris, Linux	Windows 9x, NT, other platforms	UNIX, Linux Microsoft Windows, Mac OS,
Supported Database	MySQL, mSQL, ODBC, Oracle, Informix, Sybase, etc.	ODBC, OLE DB, DB2, Oracle, Informix, Sybase, etc.	any ODBC-compliant database	any ODBC- and JDBC-compliant database
Speed	Fast	slower	slower	fast

Table 3.2 Comparison of PHP, ColdFusion, ASP, and JSP

3.4 Comparison

Reminder systems are used widely for many applications, such as appointment management, childcare, e-mail reminder, and event scheduler. Smart-Assist is a more generalized reminder system. Its distinctive features are:

- Smart-Assist is designed for general purpose.
- Smart-Assist is local enabled on Windows and web enabled.

- Smart-Assist is a cross platform system for web based sub-system.
- Smart-Assist is developed with Open Source.
- Smart-Assist has a complex alarm function: events are notified with music played by independent RealPlayer.

A table to show the differences between current reminder systems.

Reminder System	Smart-Assist	Lexacom Enterprise Calendar	StickerDesktop
OS	Unix, Windows	Unix	Windows98
Install configuration	No	Complex	No
License	Free	Free	Not Free
Multiple users	Yes	Yes	No
Web application	Yes	Yes	No
Alarm function	Yes	No	Yes
Enable wireless	No	Yes	No
administration	Yes	Yes	No
Security level	high	high	No
Audio	Yes	No	No
Appearance changeable	No	Yes	No
Information store	MySQL database	File	File
Environment required	Very complex	Complex	Simple

Table 3.3 Comparison of three reminder systems

Chapter 4. Design and Implementation

In order to meet the design goal that this reminder system be both local and web enabled, the whole system is divided into two parts: local application sub-system and web based sub-system. They share the same database, which resides on Unix.

The local application sub-system and the web based sub-system are two independent sub-systems of Smart-Assist which each performs their own functions. They maybe install on the same machine or on different machines. The database to which they connect is kept persistent for database updating by a controlling flag in a database table. In other words, by means of a flag in the database table, it is impossible for a local application sub-system and a web based sub-system to use the database at same time.

4.1 Architecture of Smart-Assist system

The local application sub-system and the web based sub-system are two independent sub-systems to connect to the same database. The architecture of Smart-Assist is shown in Figure 4.1.

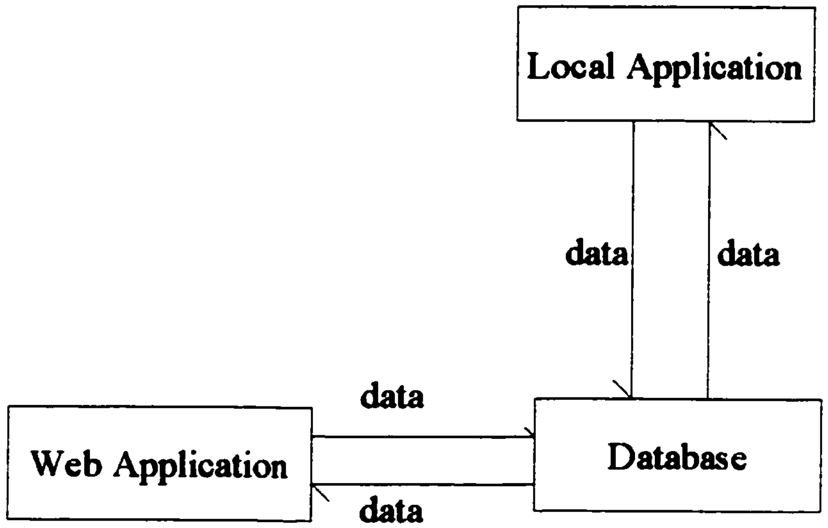


Figure 4.1. Architecture of Smart-Assist system

4.2 Design of database for Smart-Assist

"ReminderSys" was created in order to store information that users input. ReminderSys database includes 3 tables: Users, Events, and AudioFile.

Table Users: Stores all the information about users, User_ID, User_Name and Password to identify the user and user's permission. Logoff_Time identifies the last log off time of this user. Sys_Status is a flag to control that the user cannot use a local application sub-system and a web based sub-system at the same time.

Attributes:

User_ID	INT NOT NULL AUTO_INCREMENT;
User_Name	VARCHAR(20) NOT NULL;
Password	VARCHAR(20) NOT NULL;
Logoff_Time	DATETIME NOT NULL;
Status	INT NOT NULL;

Methods: used with Users

confirm_login();

add_new_user();

update_password();

delete_user();

check_user();

Table Events : Stores information about events. Event_ID and Event_Name is used to identify the events, Event_Time is the time of event. Location is where the event would happen. Description is a brief note about the event. User_ID identifies who the event belongs to. Audio_ID identifies the audio that is attached to the event notification.

Attributes:

Event_ID	AUTOINCREMENT INT NOT NULL;
Event_Name	VARCHAR(60) NOT NULL;
Event_Time	DATETIME NOT NULL;
Location	VARCHAR(100) NOT NULL;

Description	VARCHAR(220) NOT NULL;
User_ID	INT NOT NULL;
Audio_ID	INT NOT NULL;

Methods: using with Events

get_last_eventid();

get_event_list();

get_cur_eventque();

get_description();

get_audioid();

get_event();

add_event();

Table AudioFile : Stores information about audio file. Audio_ID and Audio_File_Name are used to identify the audio, Last_Used is the last time that the audio was used, Used_Times is the number of times the audio was used.

Attributes:

Audio_ID	AUTOINCREMENT INT NOT NULL;
Audio_File_Name	VARCHAR(60) NOT NULL;
Last_Used	DATETIME NOT NULL;
Used_Times	INT NOT NULL;

Methods: using with AudioFile

get_last_audiofileid();

get_audio_list();

get_audio_file_name();

check_exist();

add_audio();

By having User_ID and Audio_ID in table Events, the relation between table Users and Events are one-to-many, and the relation between table AudioFile and Events are one-to-many. Because one User may have many Events and one audio file can be used by many events, as shown in Figure 4.2.

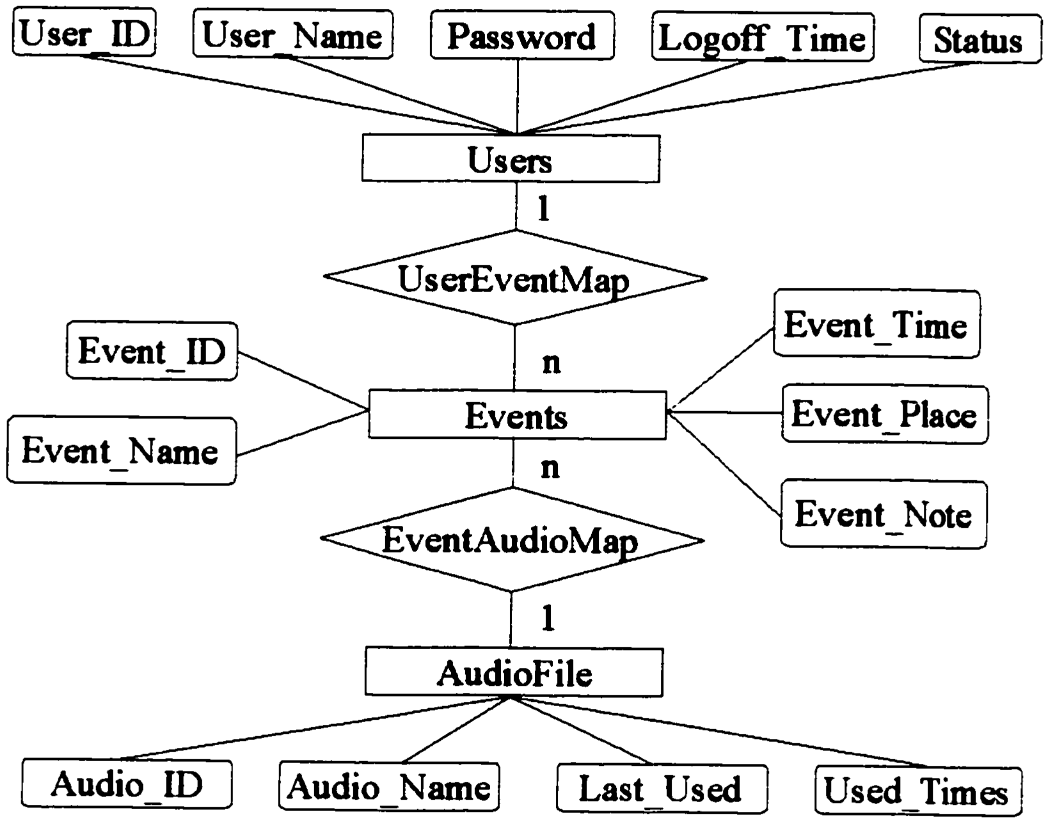


Figure 4.2 Entity-Relationship

4.3 Local application sub-system Design

The local application sub-system is non-web based and provides services for Windows users who do not like to use this system through the web browser or who want the fast speed of local application. If the database server is on the Internet, the local application sub-system can be installed on a different machine as the database.

In Smart-Assist's local application sub-system, there are five major interface functions through which users interact with the system. They are "Start", "Add Events", Add Audio", "Check Event" and "Help". With an event time queue to keep track of the occurrence of upcoming events, a RealPlayer is launched to play the selected music and display the information for the event. The architecture is shown in Figure 4.3.

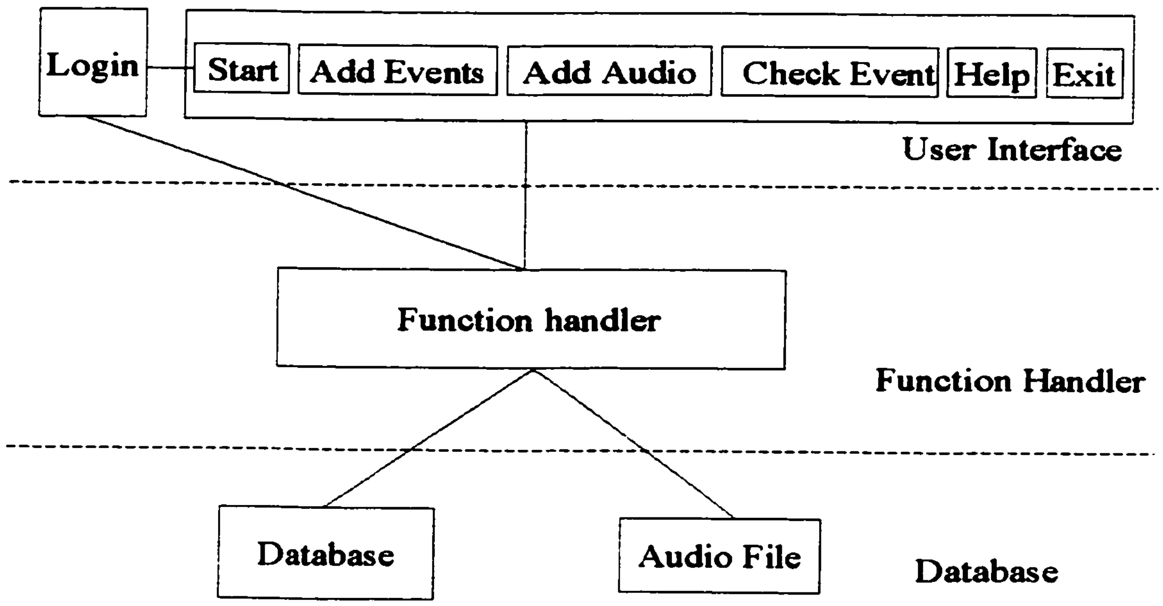
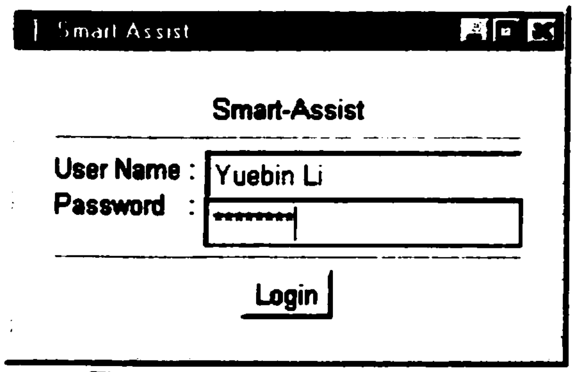


Figure 4.3. Architecture of local application sub-system

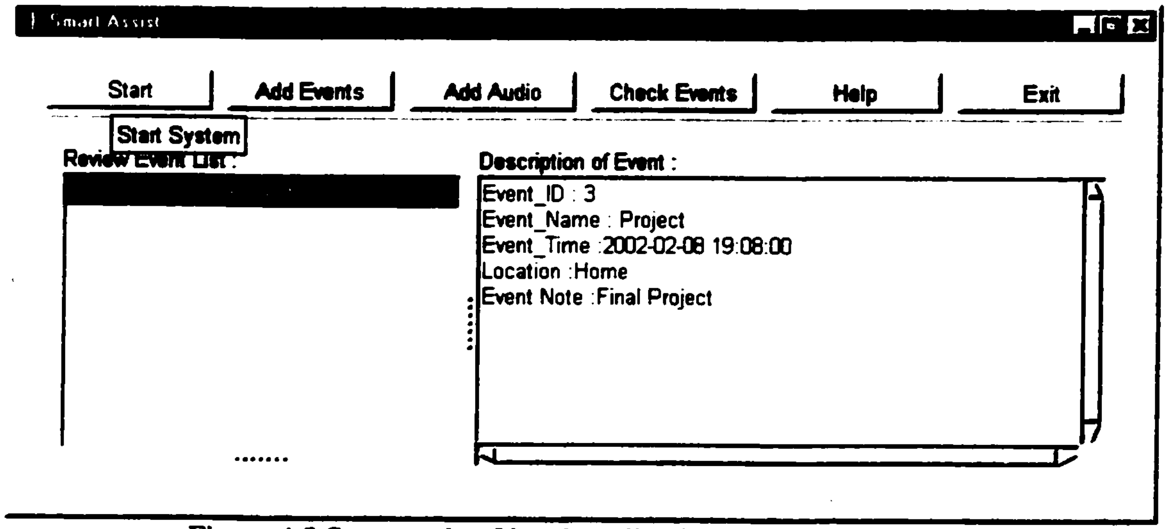
On launching Smart-Assist, it goes into the login mode and allows the user to input user name and password, as shown in Figure 4.4.



User Name : Yuebin Li
Password : ********

Figure 4.4 User Login

Once a user is logged in, Smart-Assist goes into "Start" mode. "Start" lists events and gets a description of a selected event. The "Start mode" is shown in Figure 4.5.



Event_ID : 3
Event_Name : Project
Event_Time :2002-02-08 19:08:00
Location :Home
Event Note :Final Project

Figure 4.5 Start mode of local application sub-system

Within the "Start mode", the system first checks the system status to make sure that this user is already logged in. If not, it gets the time of last log off and the current time so we can determine the events that would have occurred between the last log off time and the current time to display them. Furthermore, it gets a list of events which would occur after the current time but within the current day. At the remind time of an event in the current event list, the system executes a RealPlayer (Figure 4.17) to display a description of the event and plays an audio attached to the event. The architecture is shown in Figure 4.6.

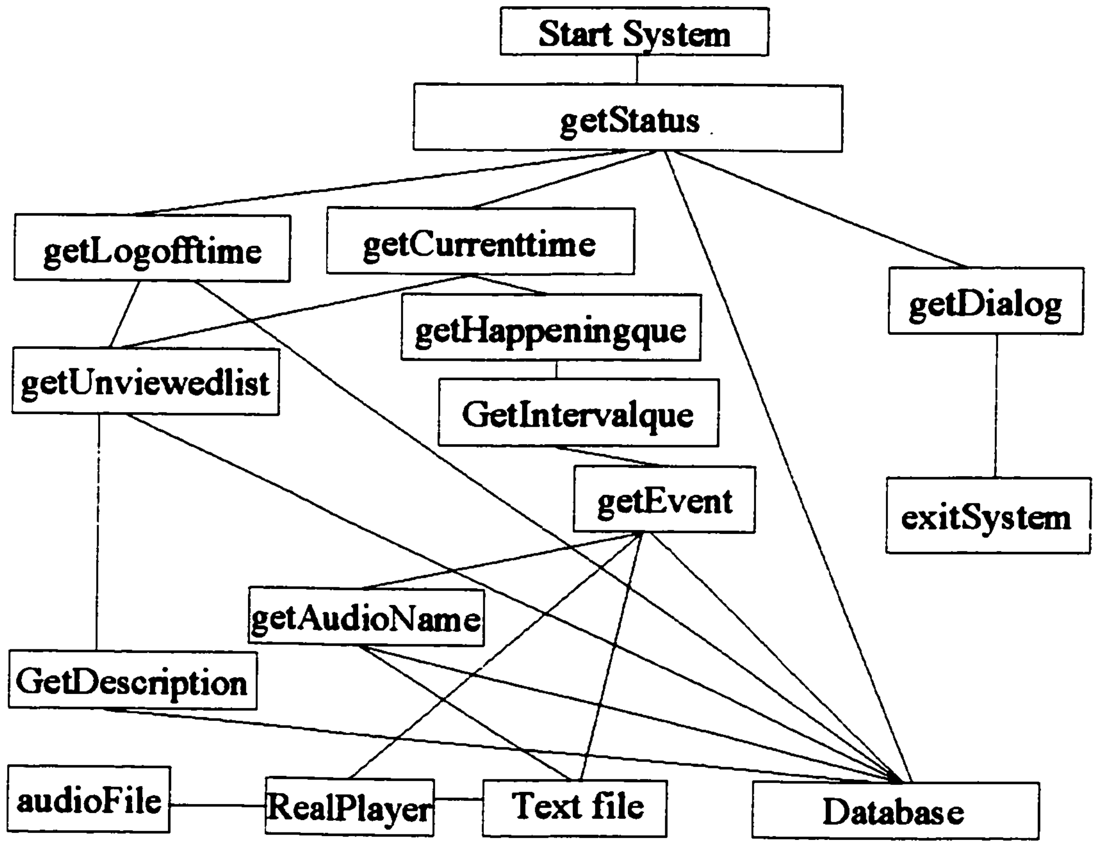
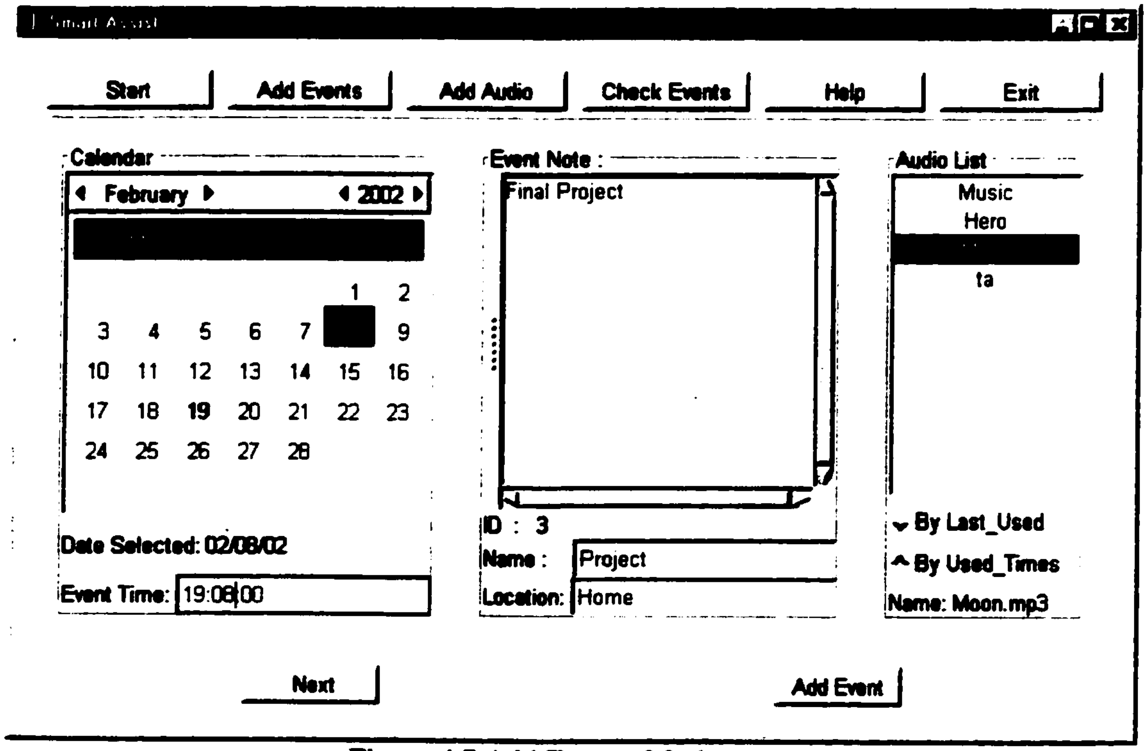


Figure 4.6 Architecture of "Start Mode"

In "Add Events Mode", the user can store information about events into the database. This is shown in Figure 4.7.



1 2
3 4 5 6 7 9
10 11 12 13 14 15 16
17 18 19 20 21 22 23
24 25 26 27 28

Date Selected: 02/08/02
Event Time: 19:08:00

Final Project

ID : 3
Name : Project
Location: Home

Music
Hero
ta

Name: Moon.mp3

Figure 4.7 Add Events Mode

In the "Add Events Mode", the system allows the user to select a date from a calendar, input the event time, name, location and description. The audio to be associated with the event can be selected from the list which can be ordered by the time of last use or frequency of use. This information about the event is added to the database using the next event id.

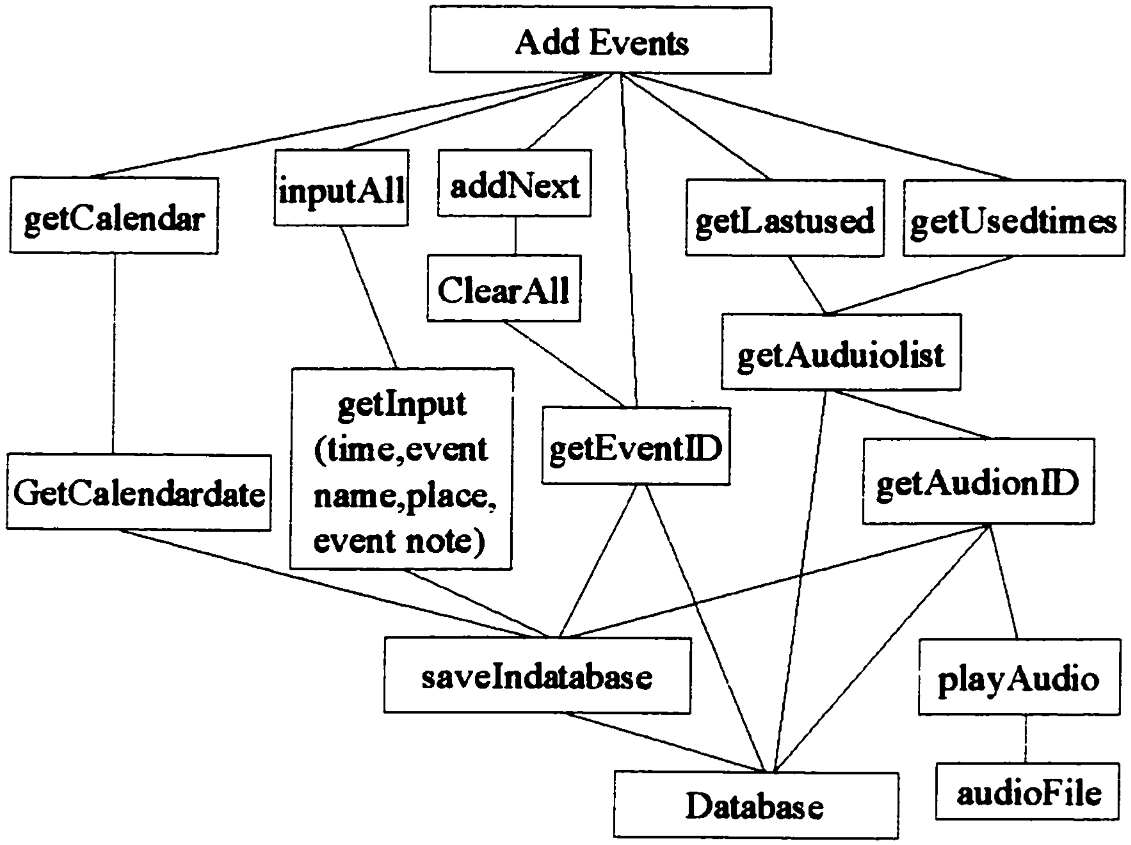
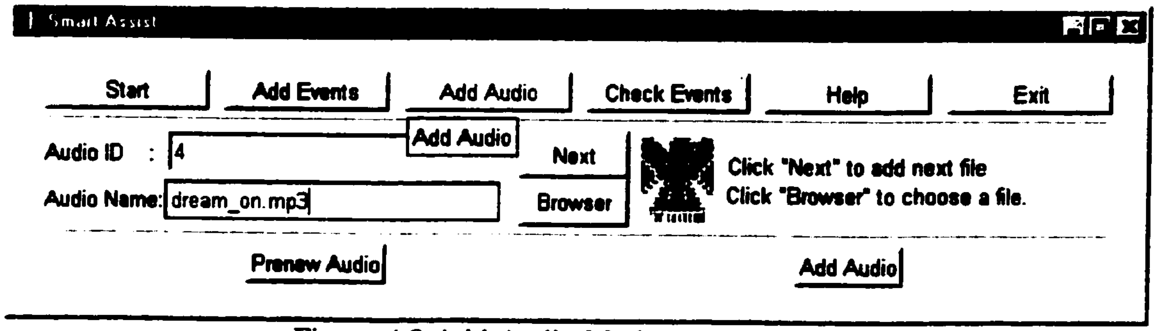


Figure 4.8 Architecture of "Add Events"

In "Add Audio", the user can save information about the audio file into the database. This is shown in Figure 4.9.



Audio ID : 4

Audio Name: dream_on.mp3

Click "Next" to add next file

Click "Browser" to choose a file.

Figure 4.9 Add Audio Mode

In "Add Audio", the system allows the user to input audio file name which is added to the database using the next audio id. The architecture is shown in Figure 4.10.

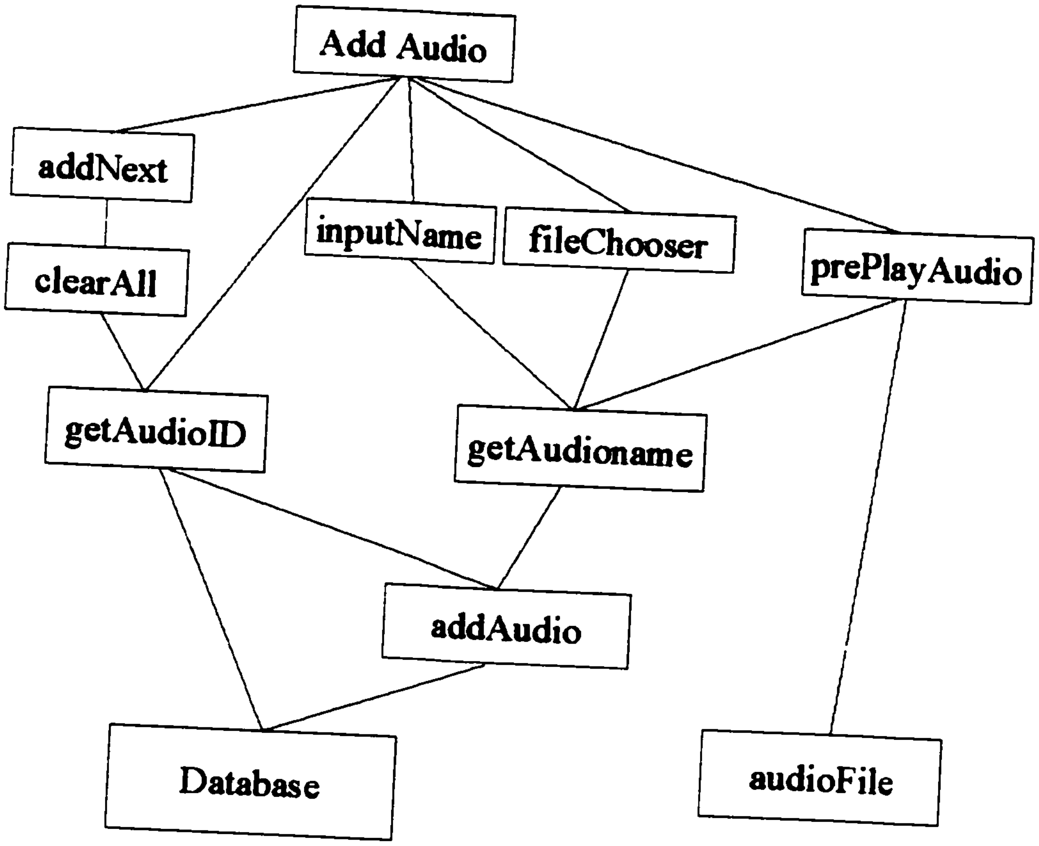
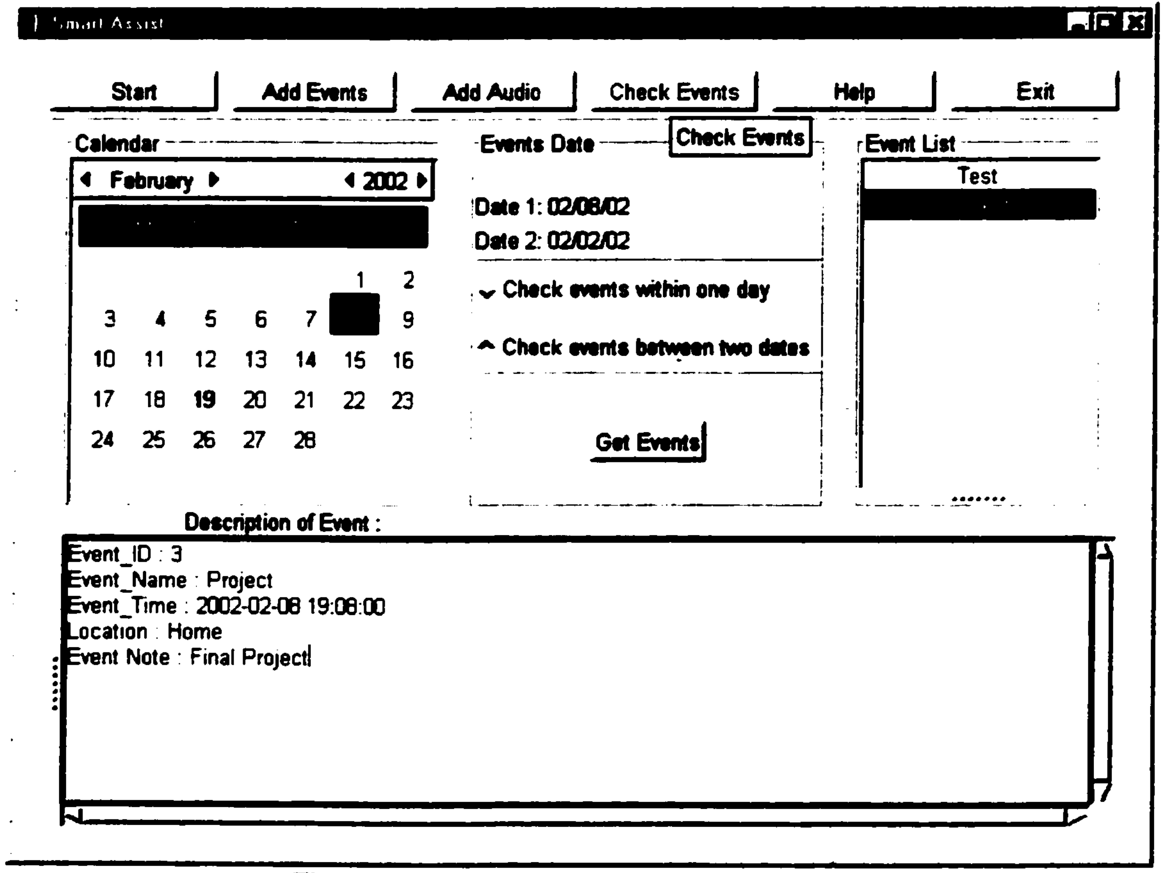


Figure 4.10 Architecture of "Add_Audio"

In the "Check Event", the user can review event by date. There are two choices: the events occur within one day or the events between two dates can be displayed. The "Check Events Mode" is shown in Figure 4.11.



Event_ID : 3
Event_Name : Project
Event_Time : 2002-02-08 19:08:00
Location : Home
Event Note : Final Project

Figure 4.11 Check Events Mode

In the "Check Events", the system gets the dates the user has selected and displays the events between these two dates. If the two dates are the same, this means the event time is within one day; otherwise, by clicking the radio button "Check events between two dates" and selecting two dates from the Calendar, the events that are between two dates are listed. The user can display the details about the event in the list by clicking on it. The architecture of "Check Events Mode" is shown as Figure 4.12.

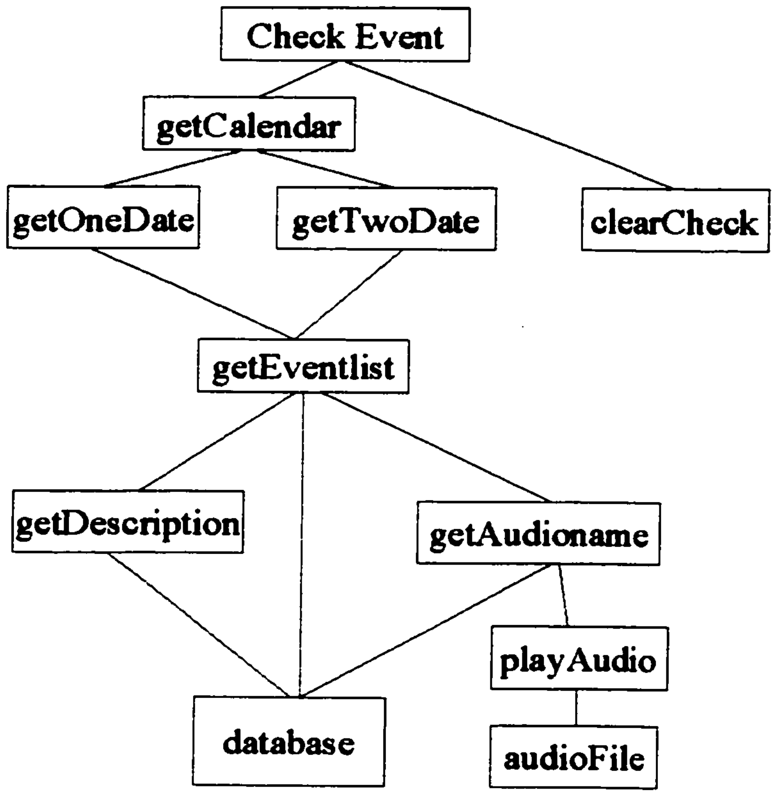
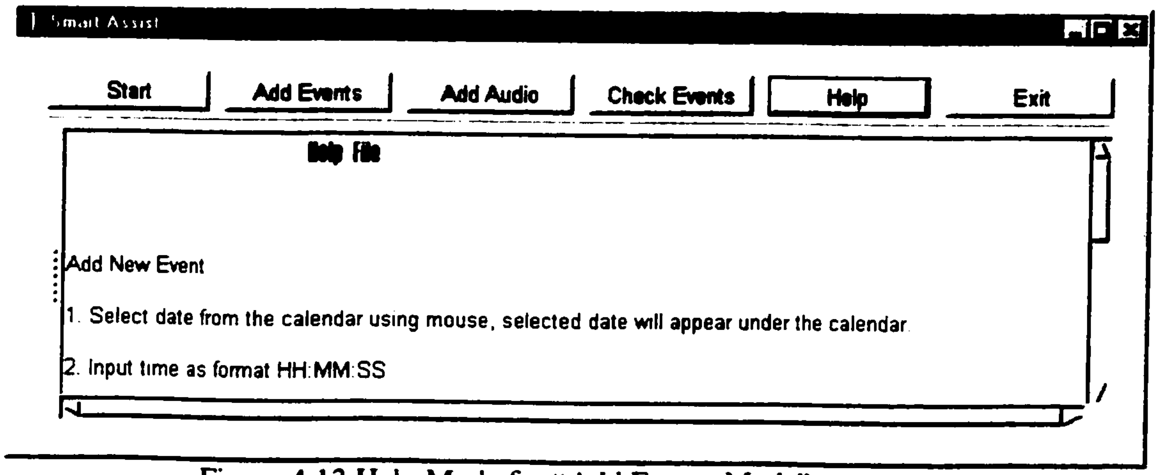


Figure 4.12 Architecture of "Check_Event"

Whenever you need help, you can get the help file for the mode that you need. The "Help Mode" is shown in Figure 4.13.



Add New Event

1. Select date from the calendar using mouse, selected date will appear under the calendar.

2. Input time as format HH:MM:SS

Figure 4.13 Help Mode for "Add Events Mode"

When the user is in a mode and needs some help, the system will display help file for this mode. The architecture of "Help Mode" is shown in Figure 4.14.

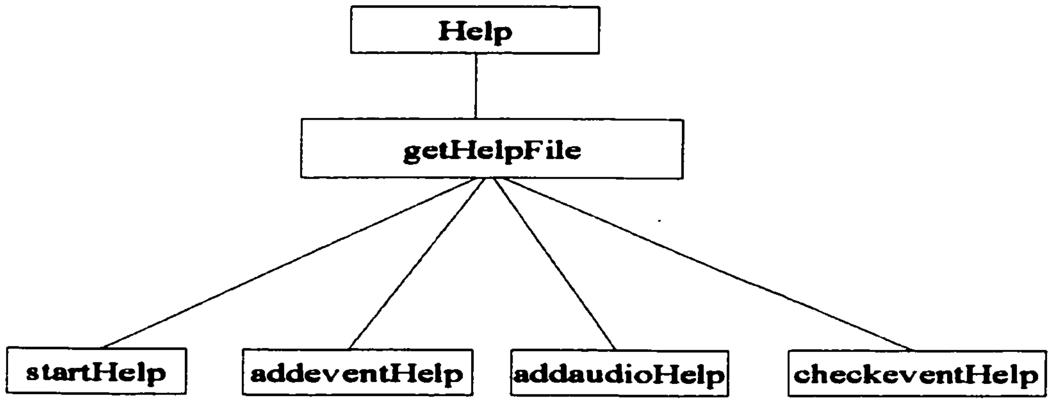


Figure 4.14 Architecture of "Help"

When the user wants to exit the system, the system will update the status flag, close the database and quit the system. The architecture is shown in Figure 4.15.

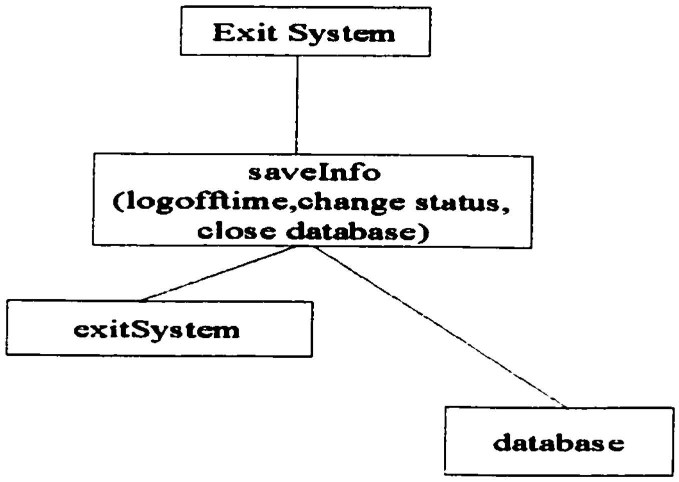


Figure 4.15 Architecture of "Exit"

Before an event happens, a RealPlayer (Figure 4.17) will appear with the event description and play music as an audible reminder. The RealPlayer is an independent part of the local application system of Smart-Assist. It is executed by a system command and gets information by reading information about the event and music name from files, which are created by the system, so that any change in this part will not affect any of the other functions. The architecture of RealPlayer is shown as Figure 4.16.

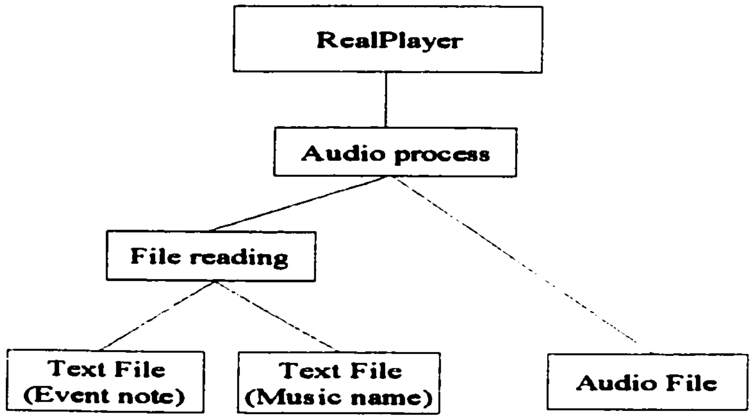
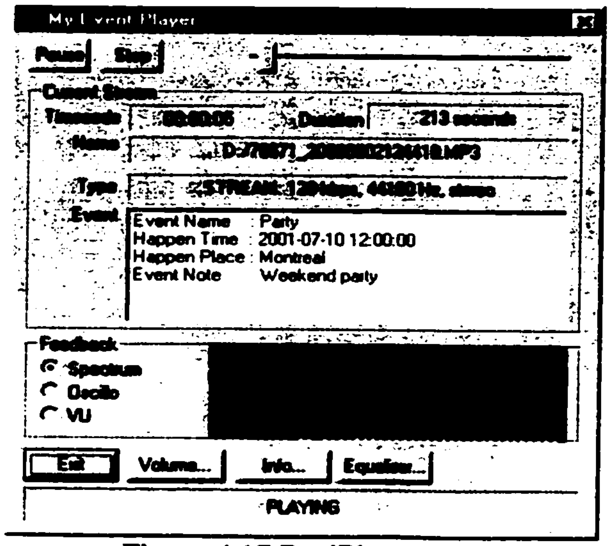


Figure 4.16 Architecture of RealPlayer



Timecode: 00:00:05
Duration: 213 seconds
Name: ...D:\[illegible].MP3
Type: STREAM: 128 kbps, 44100 Hz, stereo
Event:
Event Name : Party
Happen Time : 2001-07-10 12:00:00
Happen Place : Montreal
Event Note Weekend party

[✓ Spectrum] Oscillo VU

PLAYING

Figure 4.17 RealPlayer

4.4 Web based sub-system design and implementation

The web based sub-system of Smart-Assist is a web database application [9,16]. With the advent of Web database technology, Web pages are no longer static, but dynamic with connection to a back-end database. Web sites are now able to display updated information on-the-fly and support user interactivity. Furthermore, one can use the same page to display information of thousands of products in a database. Data-driven pages take the emphasis off managing the site itself and allow a business to focus on the content [22].

To build a Web database, usually we need to use a three-tier model: Web server, Web application server (middleware), and database [22]. The trinity is best illustrated by the figure below:

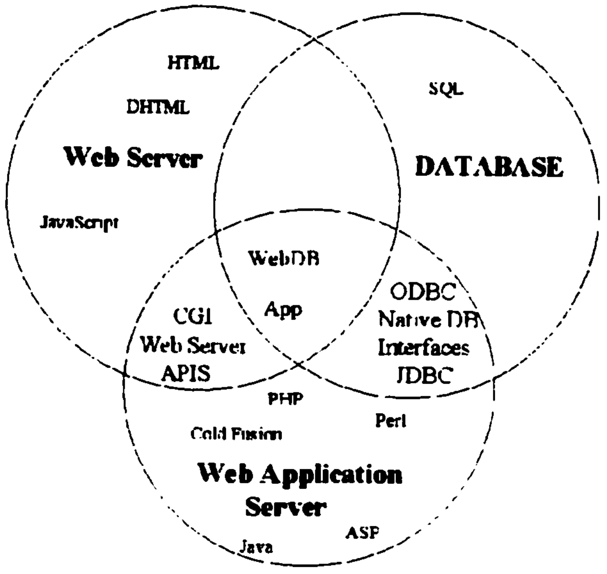


Figure 4.18 Three-tier model

The interface between the system and a Web user is a Web browser.

1. In a Web browser, a user submits a request to the Web server.
2. The Web server passes it onto the middleware
3. The middleware writes the request in SQL queries and sends it to a back-end database.
4. The retrieved data are handed back to the middleware
5. The middleware generates a Web page for the data
6. The Web server sends the Web page to the browser
7. The browser displays the Web page in front of the user

A simplified flow chart is shown below:

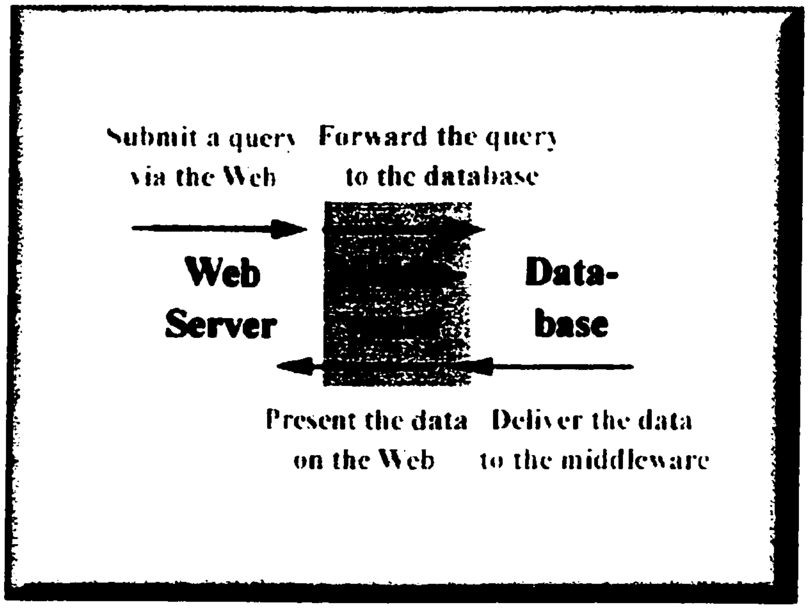

Submit a query via the Web
Forward the query to the database
Web Server
Data-base
Present the data on the Web
Deliver the data to the middleware

Figure 4.19 Interact with web database

The web based sub-system of Smart-Assist uses server side programming. The tasks are separated and implemented in every page. A server side script language PHP is employed to deal with the data transfer to or from MySQL database. JavaScript is embedded in HTML to improve user interface appearance and HTML table [12] is used to control the layout of user interface. PHP and JavaScript are tightly embedded in HTML. The web based sub-system includes 19 web pages and interacts with each other with links. The architecture of the web based sub-system is shown in Figure 4.20.

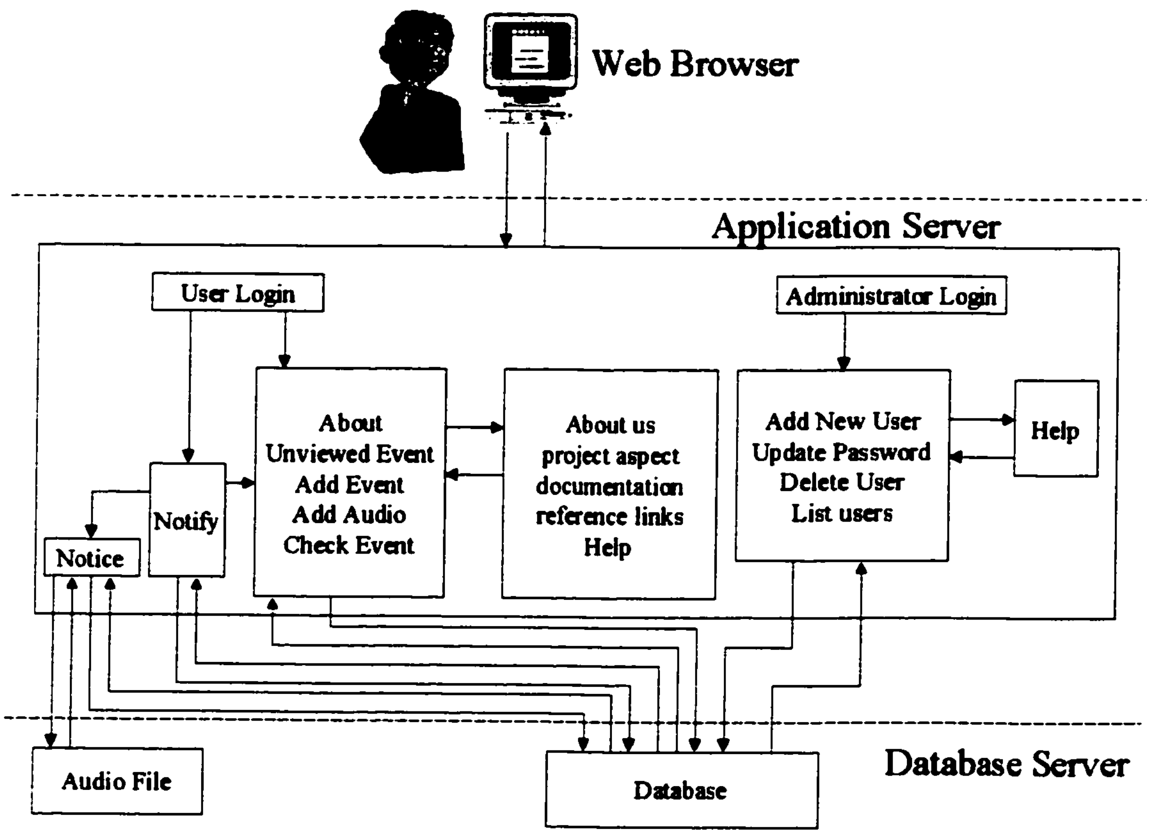


Figure 4.20 Architecture of web application part

In the User Login page, user's identification and password are required to authenticate the user. It is the entry point of the web based sub-system. This page is shown in Figure 4.19.

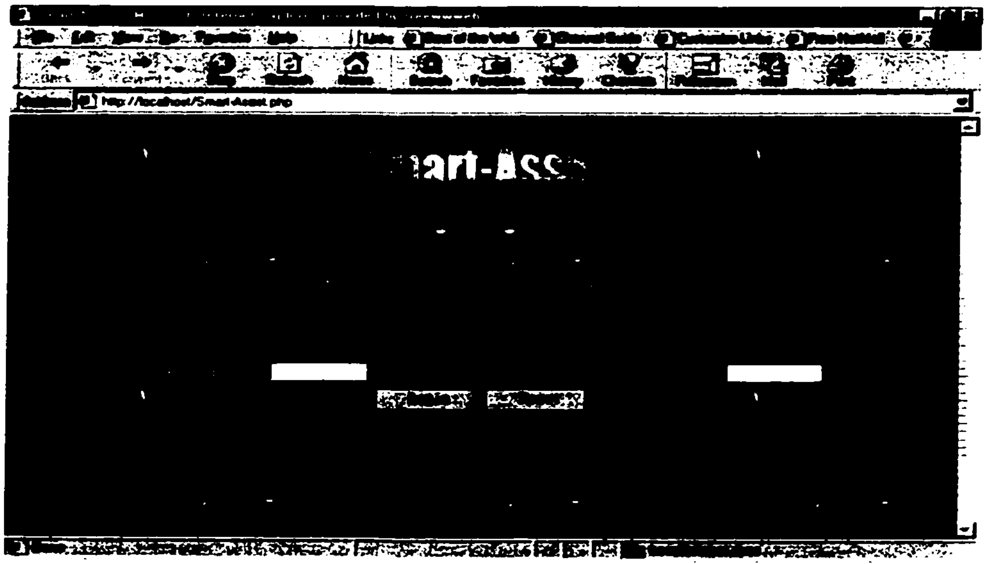


Figure 4.21 User Login

Once a user is logged in, Smart-Assist displays the About Smart-Assist page (Figure 4.22). The PHP system will register the user id as a session variable to identify the current user and displays a brief introduction of Smart-Assist. Currently, the "Notify" page (Figure 4.23) is popped up, which keeps track of events for the alarm function.

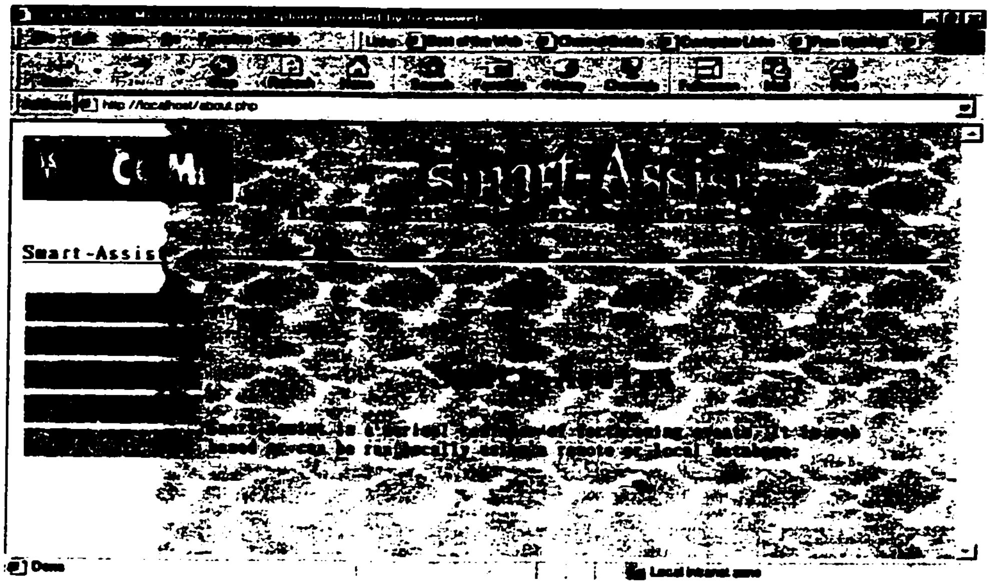


Figure 4.22 About Smart-Assist

This page retrieves the user id from security check, and gets all the event time of forthcoming events which would occur within the current date, puts them in the event time queue and waits for the upcoming event in this queue by timeout function. When an event time occurs, the "Notice" page pops up as an alarm function.

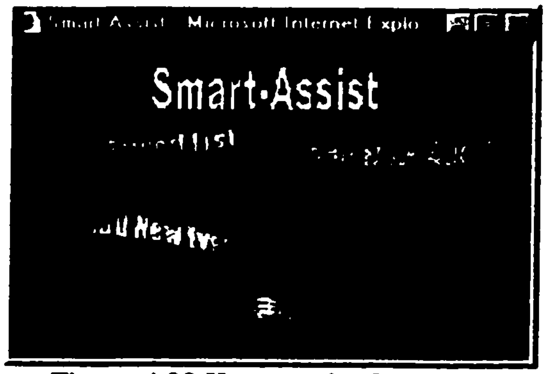


Figure 4.23 Keep track of events

The "Notice" page is the reminder and acts as an alarm function which displays the information about the event. At the same time, it gets audio id and plays the audio, which is identified by this id, in an embedded web RealPlayer. The web page of "Notice" is shown in Figure 4.24.

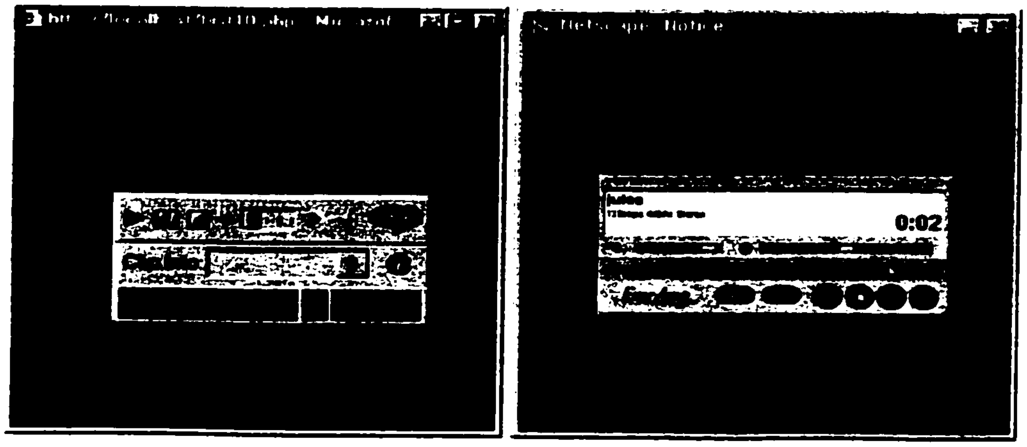


Figure 4.24 Notice page(From Windows and Unix)

The "unviewed events" page displays a list of events that have occurred since the last visit. These events are the event time between the last visit time and the current time. The web page is shown in Figure 4.25.

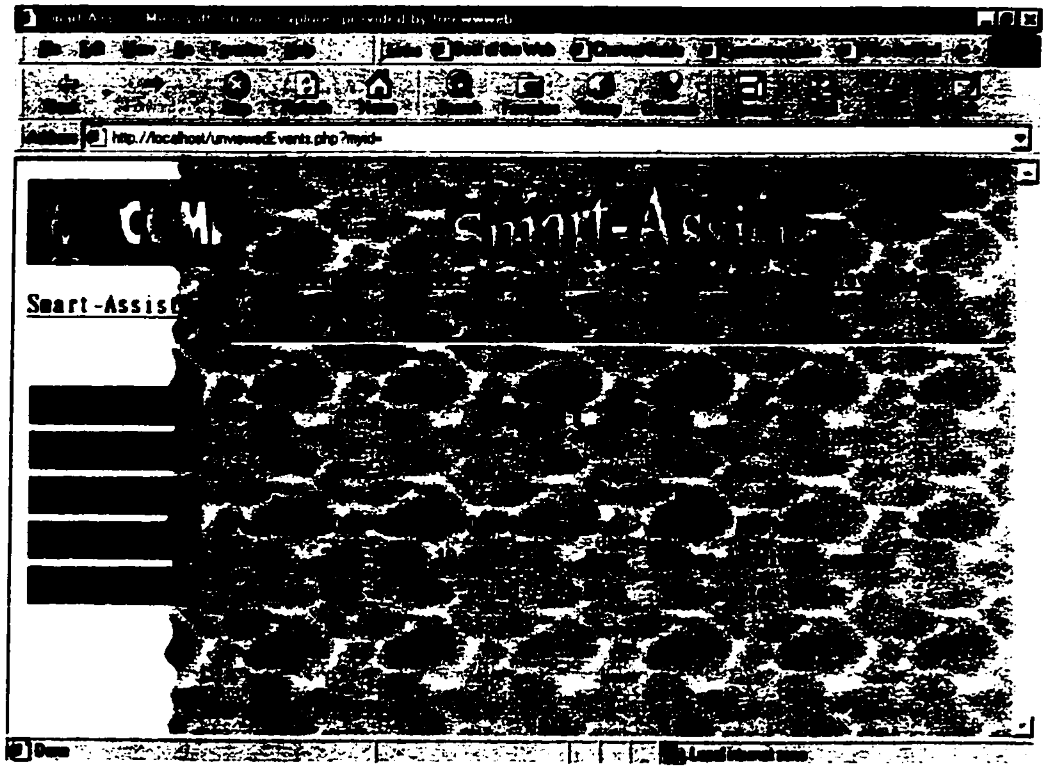


Figure 4.25 Event List

The "add events" page is used for adding events. It allows the user to input information about an event and select the audio to be associated with it, as shown in Figure 4.26.

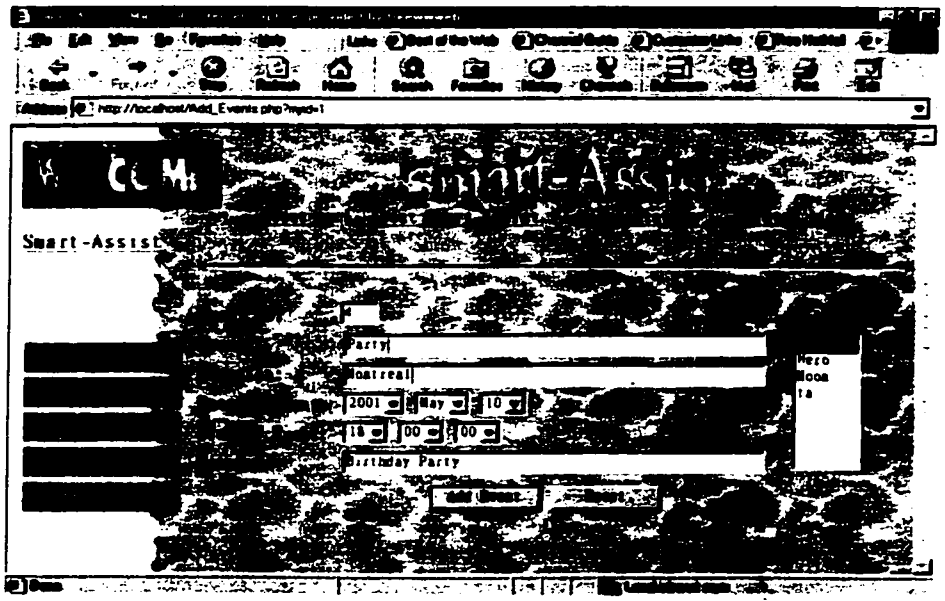


Figure 4.26 Add Event

The "add audio" page is used for adding an audio file. The user provides the audio file through input and the audio id is auto-increased, as shown in Figure 4.27.

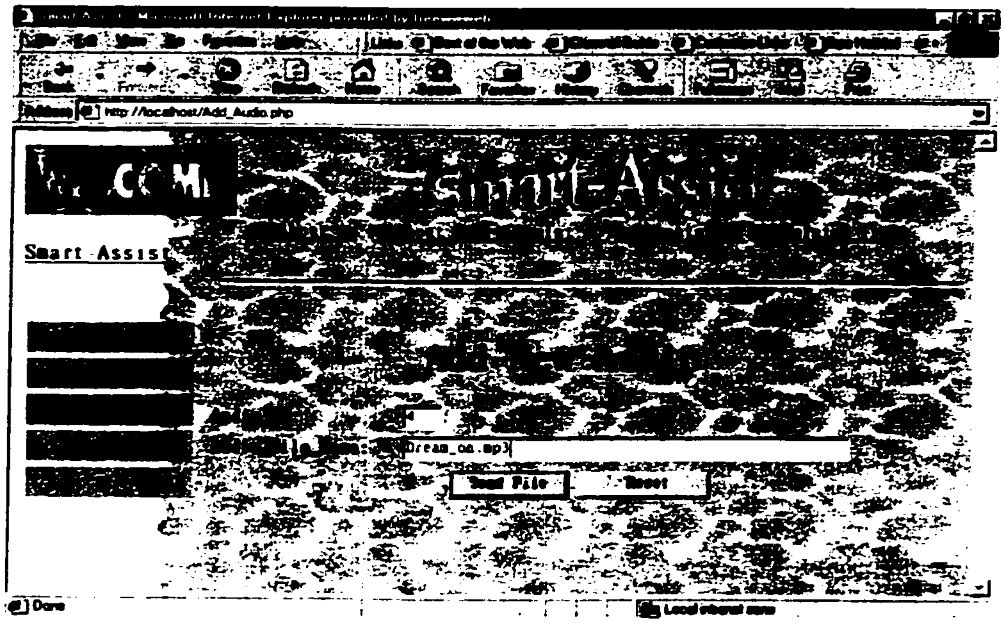
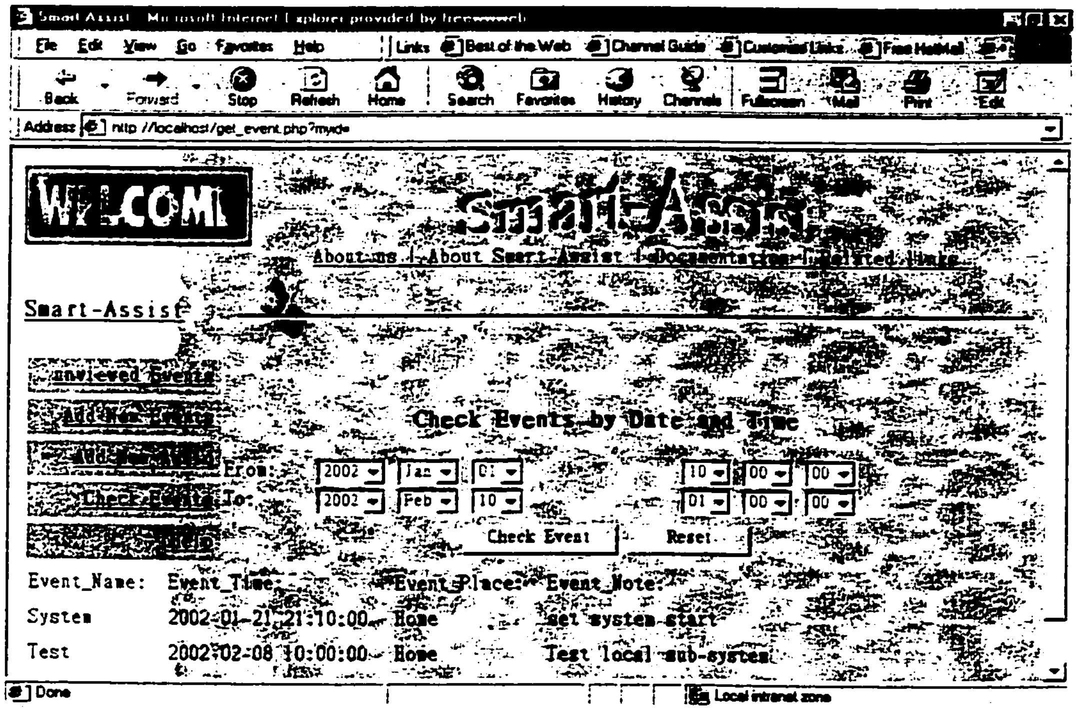


Figure 4.27 Add Audio

The "check events" page is provided to review events by date and time. The user provides one date and time as a start time and another date and time as an end time. When "Check Event" is clicked, all the user's events between these two date and times will be listed. This page is shown in Figure 4.28.



Check Events by Date and Time

From: 2002 Jan 01 10 00 00
To: 2002 Feb 10 01 00 00

```
Event_Name:  Event_Time:          Event_Place:  Event_Note:
System       2002-01-21 21:10:00  Home          set system start
Test         2002-02-08 10:00:00  Home          Test local sub-system
```

Figure 4.28 Web page of Check Events

Help is provided via a "Help" page, as shown in Figure 4.29.

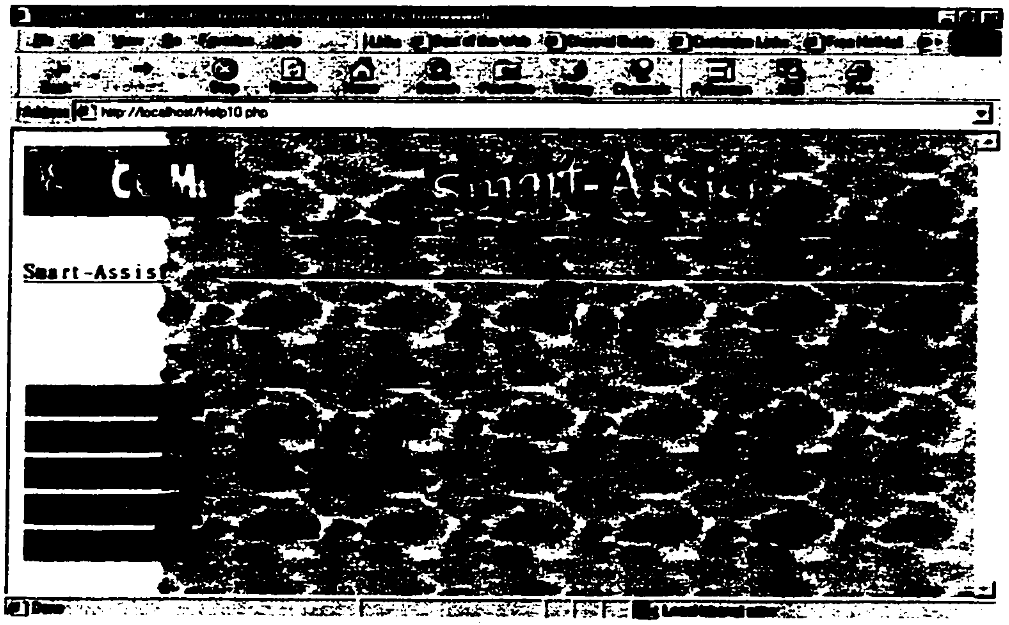


Figure 4.29 User Help File

Figure 4.30 shows the login page for Smart-Assist administrator.

Figure 4.30 Administrator Login Page

One administration function is to add a new user for the system, as shown in Figure 4.31. Here the user name and password is entered and associated with the next available user id.

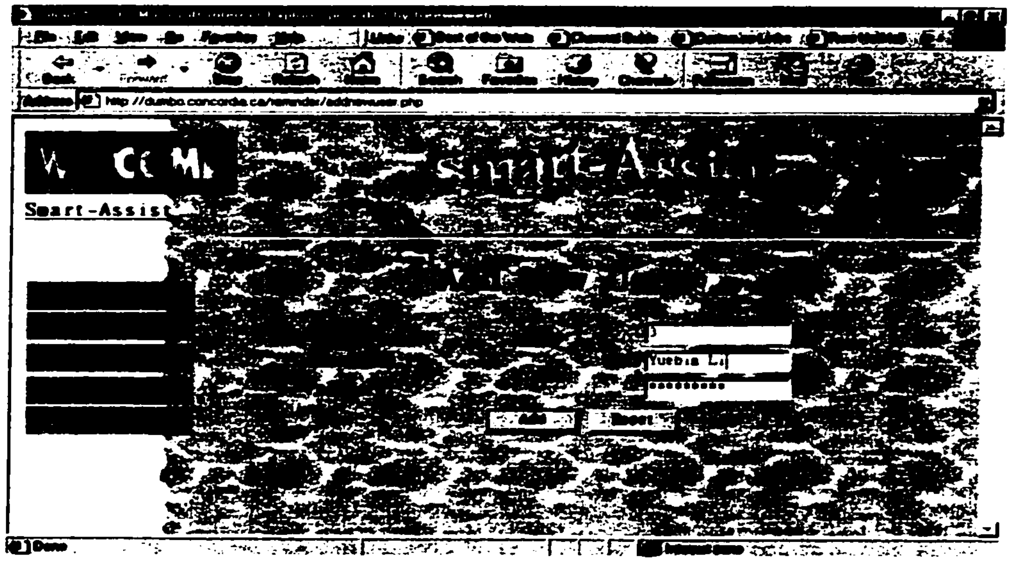


Figure 4.31 Add New User

Another administration function is to change user's password, as shown in Figure 4.32.

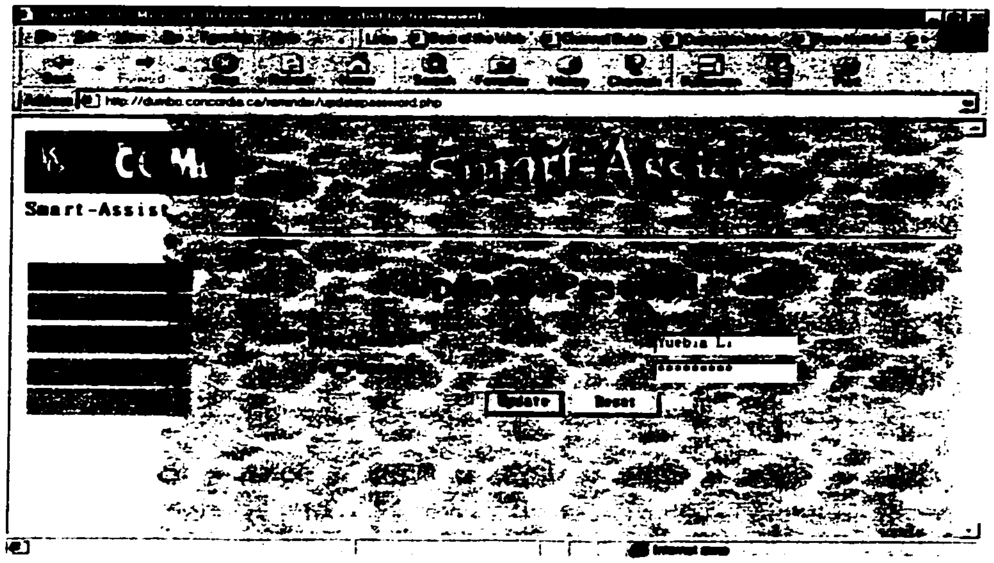


Figure 4.32 Change User Password

The "delete user" page is used for deleting the user from the system, as shown in Figure 4.33.

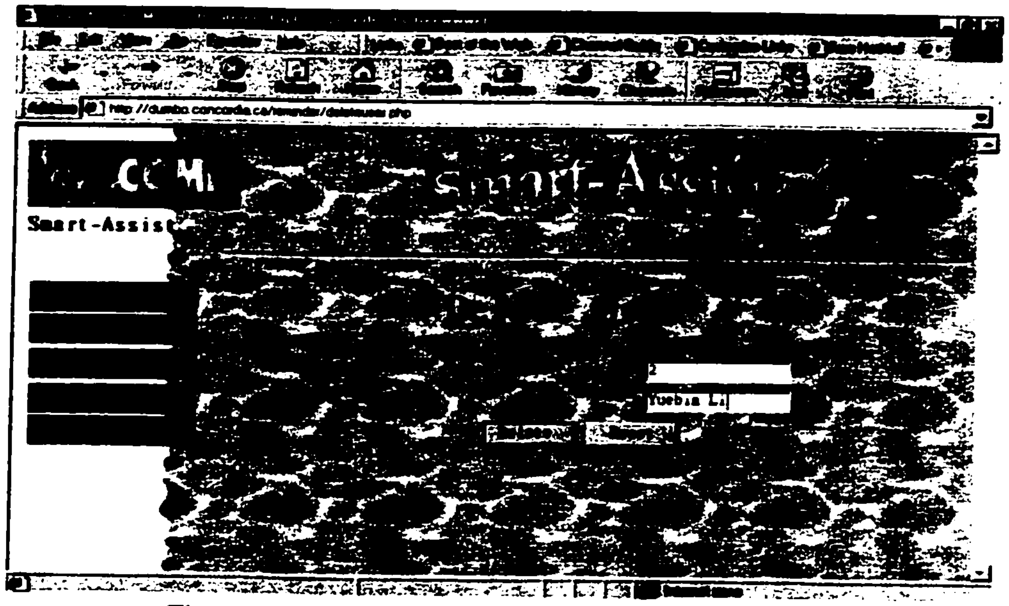


Figure 4.33 Delete User from System

The "list user" page is used for the administrator to view the users of the system. All the user ids and user names will be listed. The web page is shown in Figure 4.34.

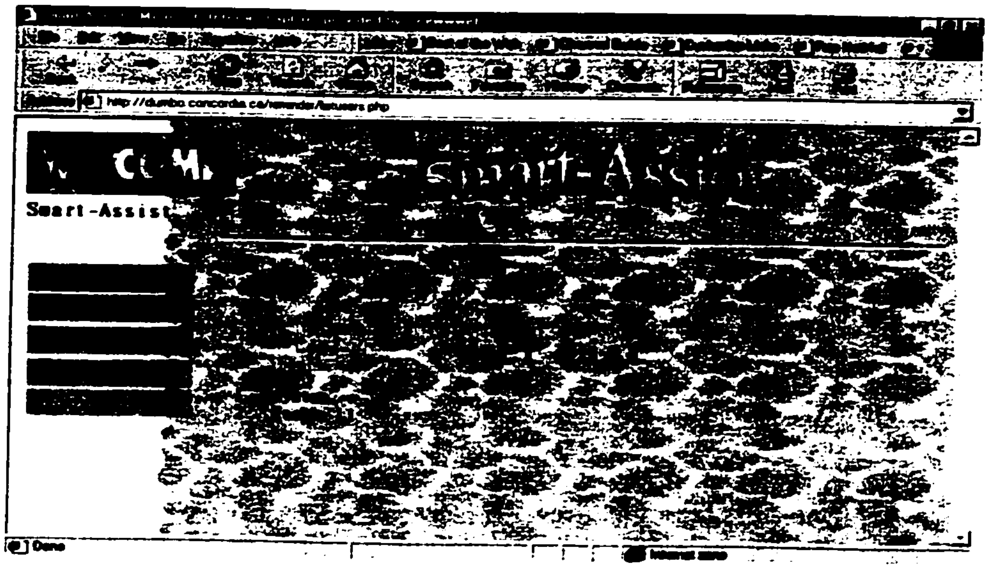


Figure 4.34 User List Page

"Administrator Help" is the help page for the administrator. This web page is shown in Figure 4.35.

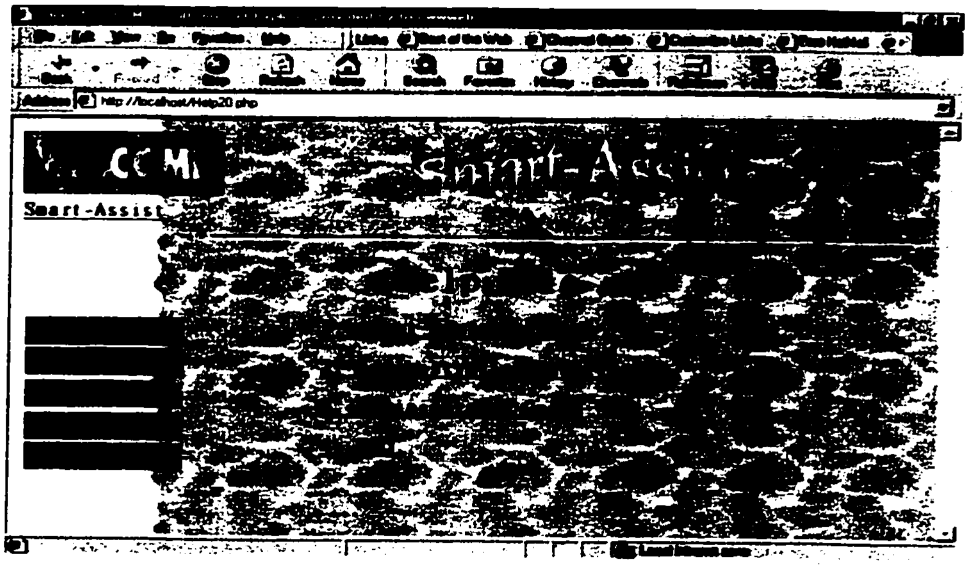


Figure 4.35 Administrator Help Page

The web page "About us" gives brief information about the developer of Smart-Assist. This page is shown in Figure 4.36.

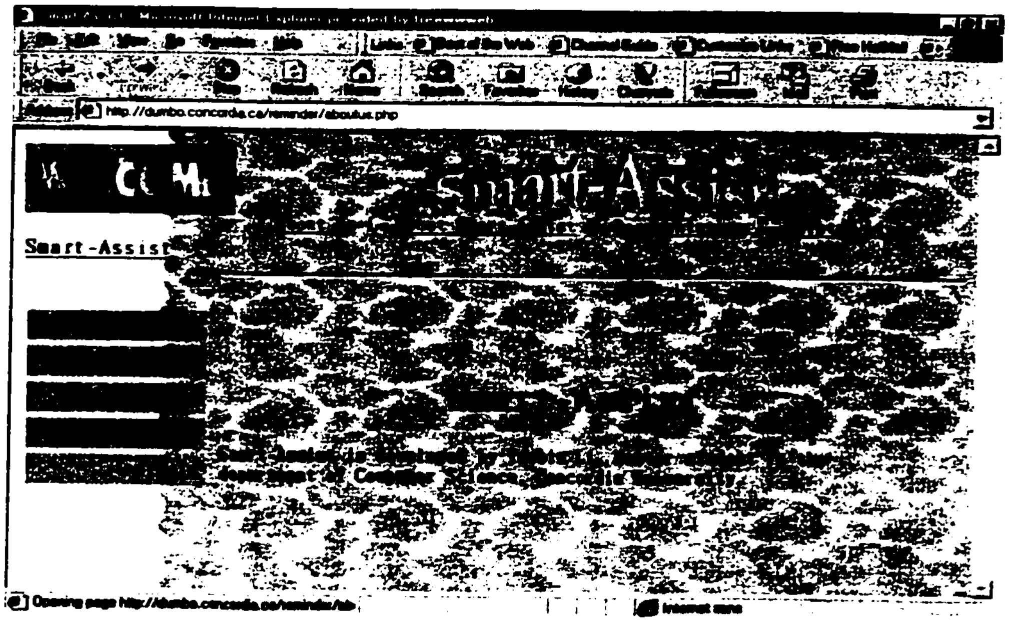


Figure 4.36 About us Page

The web page "Documentation" is for project documentation. The web page is shown in Figure 4.37.

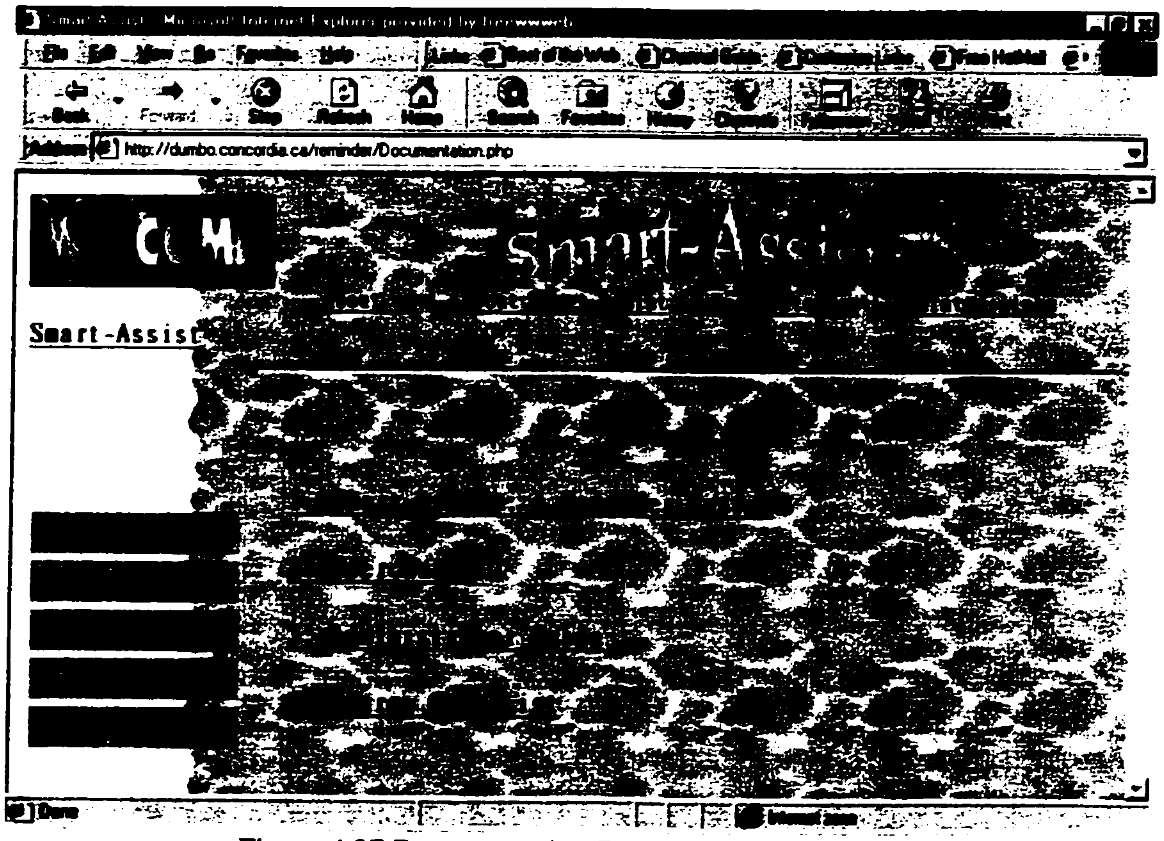


Figure 4.37 Documentation Page

The "Related Links" page gives some resource links related to Smart-Assist. The web page is shown in Figure 4.38.

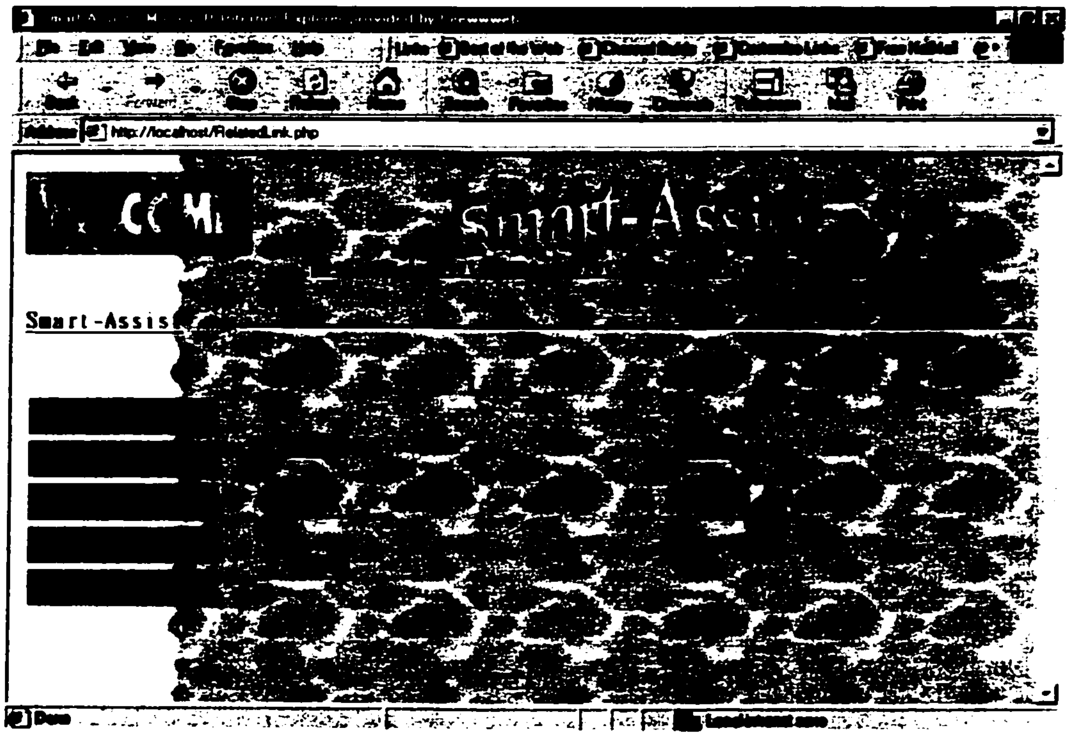


Figure 4.38 Related Links

Chapter 5. Smart-Assist Tests and Results

Smart-Assist is designed to include a local application sub-system and a web based sub-system. They share the same MySQL database.

The local application sub-system is implemented in GTK+, C, and the web based sub-system is implemented in PHP, JavaScript, and HTML. Currently, the local application sub-system is limited to a Windows application.

Smart-Assist includes many Open Source components, such as GTK+, PHP, MySQL and Apache Server [10,13]. The source code for Smart-Assist can be downloaded from http://dumbo.concordia.ca/reminder/sourcecode.

Smart-Assist is designed to be general purpose. It can be used as an appointment reminder system, a scheduler for daily work, etc. The general calendar function of both a local application sub-system and a web based sub-system has been tested from "add new event user interface". With Smart-Assist, the user can create new events in the user calendar and store them for future review. The GUI of a local application sub-system and a web based sub-system are shown in Figures 4.7 and Figure 4.26.

Smart-Assist has been tested for a complex event alarm function. This alarm function is different from a general alarm function. The alarm time doesn't need to be set separately;

it is set automatically by event time when the user starts the system. A RealPlayer will get the event description and play mp3 music as an audio reminder.

Smart-Assist's MySQL database can be hosted on either a Windows platform or a Unix platform, and can be local or on a remote system.

The web based sub-system has been tested. It could be installed on either a Windows or Unix platform. The local application sub-system has only been implemented and tested on WindowsNT and Windows98.

Chapter 6. Conclusion and future work

6.1 Conclusion

Smart-Assist is a very useful tool to remind users of important events. By saving event information into a database, the user can easily review events. Smart-Assist can give automatic notice to the user when events are occurring. It is a distributed, interactive system and supports multiple users; it is a real time system integrated with an mp3 music playing application, combining local application and web application in one system for the Windows platform.

From the user's point of view, Smart-Assist tries to include all features of other reminder systems, such as:

- Calendar function.
- Alarm function: displaying information of an event and playing audio to remind the user.
- Web application function: you can use it on any computer.

From the software developer's point of view, it tries to focus on:

- How to meet the users' special requirements.
- How to reduce development cost and make the application reliable.
- How to design and develop one system in different ways.

6.2 Limitation and Future work

Smart-Assist's local application sub-system implemented in C. However, to develop a cross platform reminder, some special functions which closely depend on the OS are not used. Smart-Assist is different from DesktopSticker which only runs on Windows98 and can use some special functions which belong to OS.

GTK+ is a good GUI toolkit, but it is still under development. It is not like Java Swing which has full components for GUI design. Furthermore, poor documentation and examples presents a big problem for developers.

In future development, the administration functions in Smart-Assist should be improved because they are not strong enough. Some functions need to be added, such as functions that give users more choice to choose GUI appearance and colors. On the other hand, the local application sub-system should be both tested and tuned up on Unix to make it a cross platform system.

References

1. PHP Fast and Easy Web Development by Meloni, Julie C. Prima Publishing, 7/00

2. PHP Developer's Cookbook by Hughes, Sterling / Zmievski, Andrei Sams, 12/00

3. MySQL by DuBois, Paul New Riders, 12/99

4. PHP/MySQL Tutorial

   http://hotwired.lycos.com/webmonkey/99/21/index2a.html

5. PHPBuilder

   http://www.phpbuilder.com/

6. Unix vs. Windows by Siteserver, Inc.

   http://www.siteserver.com/services/hosting/UnixVSWindows.asp

7. Beginning GTK+ and GNOME by Peter Wright. Wrox Press Inc 05/00

8. GTK+/Gnome Application Development by Havoc Pennington. New Riders 08/99

9. Build Your Own Database Driven Website Using PHP & MySQL By Kevin Yank.

   Publisher http://www.SitePoint.com Sep 2001

10. Apache Server Unleashed by Rich Bowen, Ken Coar. Sams 02/00

11. The Complete Idiot's Guide to JavaScript by Scott J. Walter and Aaron Weiss

12. Create html design.2 by Lynda Weinman and William Weinman

13. Apache Tutorial: Introduction to Server Side Includes

http://httpd.apache.org/docs/howto/ssi.html

14. GTK+ 1.2 Tutorial

http://www.gtk.org/tutorial/

15. Beginning MySQL Tutorial By W.J. Gilmore April 03, 1999

http://www.devshed.com/Server_Side/MySQL/Intro/page1.html

16. Website Database Basics With PHP and MySQL By Thomas Kehoe January 11, 2000

17. The Importance of the GUI in Cross Platform Development *by Michael Babcock*

http://www.iar.unlp.edu.ar/~fede/revistas/lj/Magazines/LJ49/2723.html

18. A Beginner's Guide to JavaScript by Rajesh Vijayakumar

http://www.javascriptguide.com/

20. Web Design in a Nutshell By Jennifer Niederst. O'Reilly November 1998

21. Setltor home page. http://www.fastcgi.com.

22. Web-Database Connectivity: Middleware by Jian-Qing Wu 01/01

http://www.ils.unc.edu/~wuj/webDB/home.shtml

23. DesktopSticker home page http://desktop-sticker.net/