DOWNLOAD MANAGER

By

SHIVENDRA SHARMA-031208
VAIBHAV BHARDWAJ-031231
RAHUL RANJAN SINGH-031253
ANSHUL MAHAJAN-031263

MAY-2007

ACKNOWLEDGEMENT

In Accordance with our final project submission of 8th Semester (B.Tech Computer Science & Engg.), we were assigned to study, research and implement Download Manager . We would like to express our extreme gratitude to

Mr. Amol Vasudeva
Lecturer
Department of CSE and IT,
Jaypee University of Information Technology

For guiding us, being extremely helpful, patient and always being there whenever we were in doubt. Without his help, support and constant supervision, we would have never been able to complete our final project assignment successfully.

031253	RAHUL RANJAN	Rahul
031263	ANSHUL MAHAJAN	Anshul..
031231	VAIBHAV BHARDWAJ	Vaibhav Bhardwaj
031208	SHIVENDRA SHARMA	Shivendra Sharma

25/5/07

CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ACK	Acknowledgement
AWT	Abstract Window Toolkit
FTP	File Transfer Protocol
GUI	Graphical User Interface
HTTP	Hyper Text Transfer Protocol
HTTPS	Hyper Text Transfer Protocol Secure
IP	Internet Protocol
ISN	Initial Sequence Number
JDBC	Java Database Connectivity
SQL	Sequential Query Protocol
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
URL	Uniform Resource Locator
WWW	World Wide Web
RTSP	Real Time Streaming Protocol
MMS	Microsoft Media Service

ABSTRACT

Download Manager is specifically designed for management of downloading files. There is requirement for managing more than one file at a time. Different download speeds and cheaper rates during night time bring in the requirement for scheduling downloads. Resuming the broken download due to error in connection can be handled by building a database. Download manager is made platform independent for different operating systems.

CHAPTER 1:-OVERVIEW

1.1 What is a Download Manager

A download manager is a computer program designed to download files from the Internet, unlike a web browser, which is mainly intended to browse web pages on the World Wide Web (with file downloading being of secondary importance).

List of some popular download manager:-

- Aria2
- Download Accelerator
- Download Express
- downThemAll (Firefox extension)
- FlashGet
- Free Download Manager
- Internet download manager
- Mass download manager
- Speed download
- ReGet
- QuickDownloader
- TrueDownloader (free and open source)

1.2 General Information

Following are the general information, working protocols and their features are discussed

Download manager	Creator	Interface	Runs on LINUX	Runs on Windows	Runs on MAC OS X
aria2	**Tatsuhiro Tsujikawa**	**CLI**	**Yes**	**No**	**Yes**
Download Accelerator Plus	**SpeedBit**	**GUI**	**No**	**Yes**	**No**
Downloader for X	**Maxim koshelev**	**GUI**	**Yes**	**No**	**No**
Download Express	**Metaproducts**	**GUI**	**No**	**Yes**	**No**
DownThemAll	**Federico Parodi**	**GUI**	**Yes**	**Yes**	**Yes**
FlashGet	**AmazeSoft**	**GUI**	**No**	**Yes**	**No**
Free Download Manager	**Free Download Manager**	**GUI**	**No**	**Yes**	**No**
Internet Download Accelerator	**Westbyte Software**	**GUI**	**No**	**Yes**	**No**
Speed Download	**YAZSOFT**	**GUI**	**No**	**No**	**Yes**

Table 1

The maximum download managers have their interfaces on Graphical User Interface, supported either by LINUX, WINDOWS, MAC-OS or both.

1.3 PROTOCOL SUPPORT

	HTTP	HTTPS	File Transfer Protocol	MMS	RTSP	BitTorrent	Metalink Support	Podcast Support
aria2	Yes	Yes	Yes	No	No	Yes	Yes	No
Download Accelerator Plus	Yes	No	Yes	No	No	No	No	No
Downloader for X	Yes	?	Yes	No	No	No	No	No
Download Express	Yes	Yes	Yes	No	No	No	No	No
DownThemAll	Yes	?	Yes	No	No	No	No	No
FlashGet	Yes	Yes	Yes	Yes	Yes	No	No	No
Free Download Manager	Yes	Yes	Yes	No	No	No	No	No
iGetter	Yes	Yes	Yes	No	No	No	No	No
Internet Download Accelerator	Yes	Yes	Yes	No	No	No	No	No
ReGet	Yes	Yes	Yes	Yes	Yes	No	No	No
Speed Download	Yes	Yes	Yes	No	No	No	Yes	No

Table 2

The above chart shows the different types of protocols used by the Download Managers.

The protocols includes:-

- HTTP
- HTTPS
- FTP
- MMS
- RTSP
- Bit Torrent

1.4 TRANSMISSION CONTROL PROTOCOL (TCP):-

The Transmission Control Protocol (TCP) is one of the core protocols of the Internet protocol suite, often simply referred to as TCP/IP. Using TCP, applications on networked hosts can create *connections* to one another, over which they can exchange streams of data using Stream Sockets. The protocol guarantees reliable and in-order delivery of data from sender to receiver. TCP also distinguishes data for multiple connections by concurrent applications (*e.g.*, Web server and e-mail server) running on the same host. In the Internet protocol suite, TCP is the intermediate layer between the Internet Protocol (IP) below it, and an application above it. TCP does the task of the transport layer, in the simplified OSI MODEL of computer networks. The other main transport-level Internet protocol is UDP. The TCP checks that no bytes are corrupted by using a checksum; one is computed at the sender for each block of data before it is sent, and checked at the receiver.

TCP connections have three phases:

1. **Connection establishment,**
2. **Data transfer,**
3. **Connection termination,**

Before describing these three phases, a note about the various states of a connection end-point or Internet Socket:

1. LISTEN
2. SYN-SENT
3. SYN-RECEIVED
4. ESTABLISHED
5. FIN-WAIT-1
6. FIN-WAIT-2
7. CLOSE-WAIT
8. CLOSING

9. LAST-ACK
10. TIME-WAIT
11. CLOSED

LISTEN

Represents waiting for a connection request from any remote TCP and port. (Usually set by TCP servers)

SYN-SENT

Represents waiting for the remote TCP to send back a TCP packet with the SYN and ACK flags set. (Usually set by TCP clients)

SYN-RECEIVED

Represents waiting for the remote TCP to send back an acknowledgment after having sent back a connection acknowledgment to the remote TCP. (Usually set by TCP servers)

ESTABLISHED

Represents that the port is ready to receive/send data from/to the remote TCP. (Set by TCP clients and servers)

TIME-WAIT

Represents waiting for enough time to pass to be sure the remote TCP received the acknowledgment of its connection termination request.

Connection can stay in TIME-WAIT for a maximum of four minutes

1.4.1 Connection establishment

To establish a connection, TCP uses a three-way handshake. Before a client attempts to connect with a server, the server must first bind to a port to open it up for connections: this is called a passive open. Once the passive open is established, a client may initiate an active open. To establish a connection, the three-way (or 3-step) handshake occurs:

1. The active open is performed by sending a SYN to the server.
2. In response, the server replies with a SYN-ACK.

3. Finally the client sends an ACK (usually called SYN-ACK-ACK) back to the server.

At this point, both the client and server have received an acknowledgement of the connection.

1.4.2 *Data transfer*

There are a few key features that set TCP apart from User datagram protocol:

- Ordered data transfer
- Retransmission of lost packets
- Discarding duplicate packets
- Error-free data transfer
- Congestion control

1.4.2.1 *Ordered data transfer, Retransmission of lost packets & Discarding duplicate packets*

In the first two steps of the 3-way handshaking, both computers exchange an initial sequence number (ISN). This number can be arbitrary. This sequence number identifies the order of the bytes sent from each computer so that the data transferred is in order regardless of any fragmentation or disordering that occurs during transmission. For every byte transmitted the sequence number must be incremented.

Conceptually, each byte sent is assigned a sequence number and the receiver then sends an acknowledgement back to the sender that effectively states that they received it. What is done in practice is only the first data byte is assigned a sequence number which is inserted in the sequence number field and the receiver sends an acknowledgement value of the next byte they expect to receive.

1.4.2.2 *Error-free data transfer*

The TCP checksum is a quite weak check by modern standards. Data Link Layers with a high probability of bit error rates may require additional link error correction/detection capabilities. If TCP were to be redesigned today, it would most probably have a 32-bit cyclic redundancy check specified as an error check instead of the current checksum. The weak checksum is partially compensated for by the common use of a CRC or better integrity check at Layer 2, below both TCP and IP, such as is used in PPPor the Ethernet frame. However, this does not mean that the 16-bit TCP checksum is redundant: remarkably, surveys of Internet traffic have shown that software and hardware errors that introduce errors in packets between CRC-protected hops are common, and that the end to end 16-bit TCP checksum catches most of these simple errors. This is the end-to-end principle at work

.

1.4.2.3 *Congestion control*

The final part to TCP is congestion control. TCP uses a number of mechanisms to achieve high performance and avoiding 'congestion collapse', where network performance can fall by several orders of magnitude. These mechanisms control the rate of data entering the network, keeping the data flow below a rate that would trigger collapse.

Acknowledgements for data sent, or lack of acknowledgements, are used by senders to implicitly interpret network conditions between the TCP sender and receiver. Coupled with timers, TCP senders and receivers can alter the behavior of the flow of data. This is more generally referred to as flow control, congestion control and/or network congestion avoidance.

1.4.3 *Connection termination*

The connection termination phase uses, at most, a four-way handshake, with each side of the connection terminating independently. When an endpoint wishes to stop its half of the connection, it transmits a FIN packet, which the other end acknowledges with an ACK. Therefore, a typical teardown requires a pair of FIN and ACK segments from each TCP endpoint.

A connection can be "half-open", in which case one side has terminated its end, but the other has not. The side that has terminated can no longer send any data into the connection, but the other side can.

It is also possible to terminate the connection by a 3-way handshake, when host A sends a FIN and host B replies with a FIN & ACK (merely combines 2 steps into one) and host A replies with an ACK. This is perhaps the most common method.

Finally, it is possible for both hosts to send FINs simultaneously then both just have to ACK. This could possibly be considered a 2-way handshake since the FIN/ACK sequence is done in parallel for both directions

1.5 *User Datagram Protocol (UDP)*

The User Datagram Protocol (UDP) is one of the core protocols of the Internet Protocol suite. Using UDP, programs on networked computers can send short messages sometimes known as datagram's (using Datagram Sockets) to one another. UDP is sometimes called the Universal Datagram Protocol or Unreliable Datagram Protocol.

UDP does not provide the reliability and ordering that TCP does. Datagrams may arrive out of order, appear duplicated, or go missing without notice. Without the overhead of checking whether every packet actually arrived, UDP is faster and more efficient for many lightweight or time-sensitive purposes. Also, its stateless nature is useful for servers that answer small queries from huge numbers of clients. Compared to TCP, UDP is required for broadcast (send to all on local network) and multicast (send to all subscribers).

1.6 *File Transfer Protocol (FTP)*

FTP or **File Transfer Protocol** is used to transfer data from one computer to another over the Internet, or through a network.

Specifically, FTP is a commonly used protocol for exchanging files over any network that supports the TCP/IP protocol (such as the Internet or an intranet). There are two computers involved in an FTP transfer: a server and a client .The **FTP server**, running FTP server software, listens on the network for connection requests from other computers. The client computer, running FTP client Software, initiates a connection to the server. Once connected, the client can do a number of file manipulation operations such as uploading files to the server, download files from the server, rename or delete files on the server and so on. Any software company or individual programmer is able to create FTP server or client software because the protocol is an open standard. Virtually every computer platform supports the FTP protocol. This allows any computer connected to a TCP/IP based network to manipulate files on another computer on that network regardless of which operating systems are involved (if the computers permit FTP access)

There are three modes in FTP:

1. Active mode
2. Passive mode
3. Extended passive mode

In **active mode**, the FTP client opens a random port (> 1023), sends the FTP server the random port number on which it is listening over the control stream and waits for a connection from the FTP server. When the FTP server initiates the data connection to the FTP client it binds the source port to port 20 on the FTP server.

In **passive mode**, the FTP server opens a random port (> 1023), sends the FTP client the server's IP address to connect to and the port on which it is listening (a 16 bit value broken into a high and low byte) over the control stream and waits for a connection from the FTP client. In this case the FTP client binds the source port of the connection to a random port greater than 1023.

In **extended passive mode**, the FTP server operates exactly the same as passive mode, however it only transmits the port number (not broken into high and low bytes) and the client is to assume that it connects to the same IP address that was originally connected to.

The FTP protocol supports resuming of interrupted downloads using the REST command. The client passes the number of bytes it has already received as argument to the REST command and restarts the transfer. In some commandline clients for example, there is an often-ignored but valuable command, "reget" (meaning "get again") that will cause an interrupted "get" command to be continued, hopefully to completion, after a communications interruption.

Resuming uploads is not as easy. Although the FTP protocol supports the APPE command to append data to a file on the server, the client does not know the exact position at which a transfer got interrupted. It has to obtain the size of the file some other way, for example over a directory listing or using the SIZE command.

1.7 Features

Following are the features of various download manager in terms of checksum verification, resuming, download acceleration, cookies, speed limit, file browser etc.

	Web browser integration	Resuming	Download acceleration	Mirror search	Auto dial/ hang up	Categorized downloads	Cookies import	Speed limit	File browser	ZIP preview	Checksum verification
aria2	No	Yes	Yes	No	No	No	No	No	No	No	Yes
Download Accelerator Plus	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	?
Downloader for X	?	Yes	Yes	No	No	No	No	No	No	No	No
DownThem All	Firefox Only	Yes	Yes	No	No	No	No	No	No	No	No
FlashGet	Yes	Yes	Yes	Yes	Yes	Yes	?	All downloads	Yes	No	?
Free Download Manager	Yes	Yes	Yes	Yes	Yes	Yes	?	Yes	Yes	Yes	Yes
iGetter	Yes	Yes	Yes	No	Yes	No	No	Yes	No	No	No
Internet Download Accelerator	Yes	Yes	Yes	?	?	Yes	?	?	Yes	?	?
ReGet	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
Speed Download	Yes	Yes	Yes	No	No	Yes	?	?	?	?	Yes

Above are some critical parameters which are used to measure the flexibility, feasibility and efficiency of any download manager. Also gives the brief idea and description and scope of download manager.

CHAPTER 2:- DESCRIPTION

2.1 Project Description

A socket represents a connection point into a TCP/IP network between two computers; one of which is the server and the other is the client. Sockets are used to access the Web server. These sockets are available in the Java Socket class. Sockets act as virtual passageways for communication through which the data travels from one host to another. Sockets are of two types, connection-oriented or connectionless. The connection-oriented socket operates similarly to a telephone where the connection is established prior to the transmission of data. The second type of socket is the connectionless socket. This type of socket is called a Data gram. This socket operates similarly to the mail. A characteristic of the Data gram is that it allows only one message to be sent at a time. The delivery of data is not guaranteed because Data grams use the UDP protocol.

2.1 Main Features:-

1. Manage all downloads in one window (for each download you will see the rate, currently bytes received etc.)

2. An option to download your files now, or download your files at a given time (in the night when the phone, internet rate are chipper).

3. Resume a broken download from the place it stopped.

4. Supply various user settings (Timeouts, use don't use proxy etc.).

5. Supply a friendly GUI, which can combine with other Internet browsers (insert a new file to the list in one button click from the Netscape Navigator).

6. Download Manager will be platform independent i.e it will run on LINUX as well as on Windows.

2.2 Modules:-

2. ***The GUI*** - This module contains the interface of utility. It contains the information about the file to be downloaded in the form of its URL it also provides the information about the already downloaded file in terms of URL, size, type, name, etc.

3. ***The Download Manager*** - Download the files and manage them by implementing the specified features. It allows the user to select their required protocols, HTTP, FTP, and UDP accordingly to download a file. The input is in the form of URL specified by the user.

3. ***The Dialer*** - Establish the connection between client and server. It's a Sub-module of Download manager. It works on the input given by the download manager in the form of URL generally.

4. ***The Database Manager*** - Stores the URL's and other important information (such as the time, size, type, name, URL, etc...) in the database. In order to provide the above features this module is important.

5. ***The File-fetcher*** - It's a sub-module of Download manager. By interacting the Dialer this will fetch the file.

2.3 Methodology:-

- By using the Java foundation classes like Java Swings the GUI part of each module will be done.

- Other modules are covered by some Java packages like java.net, java.io, java.util etc.

2.3.1 *GUI Components*:-

- Following are the predefined classes defined in javax.swing package where javax stands for "Java Extension" that are used in making the GUI Components
    1. JMenuBar : Creates a menu bar.
    2. JMenu : Creates a pull-down menu
    3. JMenuItem : Creates a menu item
    4. JCheckBoxMenuItem : Creates an on/off menu item
    5. JComponent : A subclass of component that can hold other components
    6. JTable : Creates a table
    7. JPanel : Provides general purpose containers for lightweight components
    8. JScrollPane : A container that provides horizontal and / or vertical scroll bars for another component.
    9. JLabel : Creates a label that displays a string or image or both
    10. JTextField : Creates a single-line edit control
    11. JButton : Creates a push button control
    12. JCheckBox : Creates a check box control
    13. JFrame : Ceates a standard window that has a title bar, resize corners and a menu bar

2.3.2 *Events on Components*

- Following are the predefined classes defined in java.awt.event package where awt stands for "Abstract Window Toolkit" that are used to tackle the events occur on the components

1. ActionListener : Defines one method, actionPerformed(), to receive action events.
2. ItemListner : Defines one method, itemStateChanged(), that is invoked when the state of an item changes.
3. KeyListner : Defines three methods, keyPressed(), keyReleased() and keyTyped(), are invoked when a key pressed, released and typed.
4. MouseListener : Defines five methods, mouseClicked(), mouseEnetered(), mouseExited(), mousePressed(), mouseReleased(), to recognize when the mouse is clicked, enters a component, exits a component, is pressed , or is released.
5. WindowListener : Defines seven methods to recognize when a window is activated, close, deactivated, deiconified, iconified, opened, or quit.

2.3.3 *Dialer*

- Following are the predefined classes in java.net package, where net stands for "Networking" , which provides support for networking and are used for data transferring and downloading the files in form of packets. These classes are used by the dialer to establish the connection between client and server :-

1. HttpURLConnection: It is an abstract class which extends the URLConnection class in order to get the support for HTTP specific features.
2. URL: Class `URL` represents a Uniform Resource Locator, a pointer to a "resource" on the World Wide Web.

3. URLClassLoader : This class loader is used to load classes and resources from a search path of URLs referring to both JAR files and directories.
4. URLConnection : The abstract class `URLConnection` is the superclass of all classes that represent a communications link between the application and a URL. Instances of this class can be used both to read from and to write to the resource referenced by the URL.

2.3.4 *Scheduling*

Scheduling is done by using the predefined classes in java.util package, where util stands for utility and java.text package where java.text package provides classes and interfaces for handling text, dates, numbers, and messages in a manner independent of natural languages.. Now following are the classes which are used to schedule the downloads:-

1. Calendar: Calendar is an abstract base class for converting between a Date object and a set of integer fields such as YEAR, MONTH, DAY, HOUR, and so on. A Calendar object can produce all the time field values needed to implement the date-time formatting for a particular language and calendar style.
2. GregorianCalendar : GregorianCalendar is a concrete subclass of Calendar and provides the standard calendar used by most of the world.
3. Observer: It is an interface provides by this package and it can implement the Observer interface when it wants to be informed of changes in observable objects.
4. Observable: This class represents an observable object, or "data" in the model-view paradigm. It can be sub classed to represent an object that the application wants to have observed. An observable object can have one or more observers. An observer may be any object that implements interface Observer. After an observable instance changes,

an application calling the Observables notifyObservers method causes all of its observers to be notified of the change by a call to their update method.

5. StringTokenizer : The string tokenizer class allows an application to break a string into tokens. The StringTokenizer methods do not distinguish among identifiers, numbers, and quoted strings, nor do they recognize and skip comments.
6. SimleDateFormat : SimpleDateFormat is a concrete class for formatting and parsing dates in a locale-sensitive manner. It extends the DateFormat class.
7. Timer: A facility for threads to schedule tasks for future execution in a background thread. Tasks may be scheduled for one-time execution, or for repeated execution at regular intervals. Corresponding to each Timer object is a single background thread that is used to execute all of the timer's tasks, sequentially.
8. TimerTask : A task that can be scheduled for one-time or repeated execution by a Timer. It needs Runnable interface to work.

2.3.5 *File-fetching*

File fetching is done by using the predefined classes of java.io package, where io stands for input and output. Provides for system input and output through data streams, serialization and the file system. Following are the classes used in file-fetching -

1. InputStream: This abstract class is the superclass of all classes representing an input stream of bytes. Applications that need to define a subclass of InputStream must always provide a method that returns the next byte of input. It is extends the Object class.
2. RandomAccessFile: Instances of this class support both reading and writing to a random access file. A random access file behaves like a large array of bytes stored in the file system. There is a kind of cursor, or index into the implied array, called the *file pointer*; input operations

read bytes starting at the file pointer and advance the file pointer past the bytes read. If the random access file is created in read/write mode, then output operations are also available; output operations write bytes starting at the file pointer and advance the file pointer past the bytes written. Output operations that write past the current end of the implied array cause the array to be extended. The file pointer can be read by the getFilePointer method and set by the seek method. It extends the Object and implement the DataOutput and DataInput class.

2.3.6 *Connectivity to Database*

Connectivity to Database is done by some predefined classes of java.sql class. It also provides the API for accessing and processing the data store in a data source. Now following are the classes which are used to connect to Database:-

1. Connection : A connection (session) with a specific database. SQL statements are executed and results are returned within the context of a connection. A Connection object's database is able to provide information describing its tables, its supported SQL grammar, its stored procedures, the capabilities of this connection, and so on. This information is obtained with the getMetaData method.
2. DriverManager : The basic service for managing a set of JDBC drivers. As part of its initialization, the DriverManager class will attempt to load the driver classes referenced in the "jdbc.drivers" system property. This allows a user to customize the JDBC Drivers used by their applications.
3. PreparedStatement : An object that represents a precompiled SQL statement. It is an interface. A SQL statement is precompiled and stored in a PreparedStatement object. This object can then be used to efficiently execute this statement multiple times.

2.4 ***Flow Diagram***:

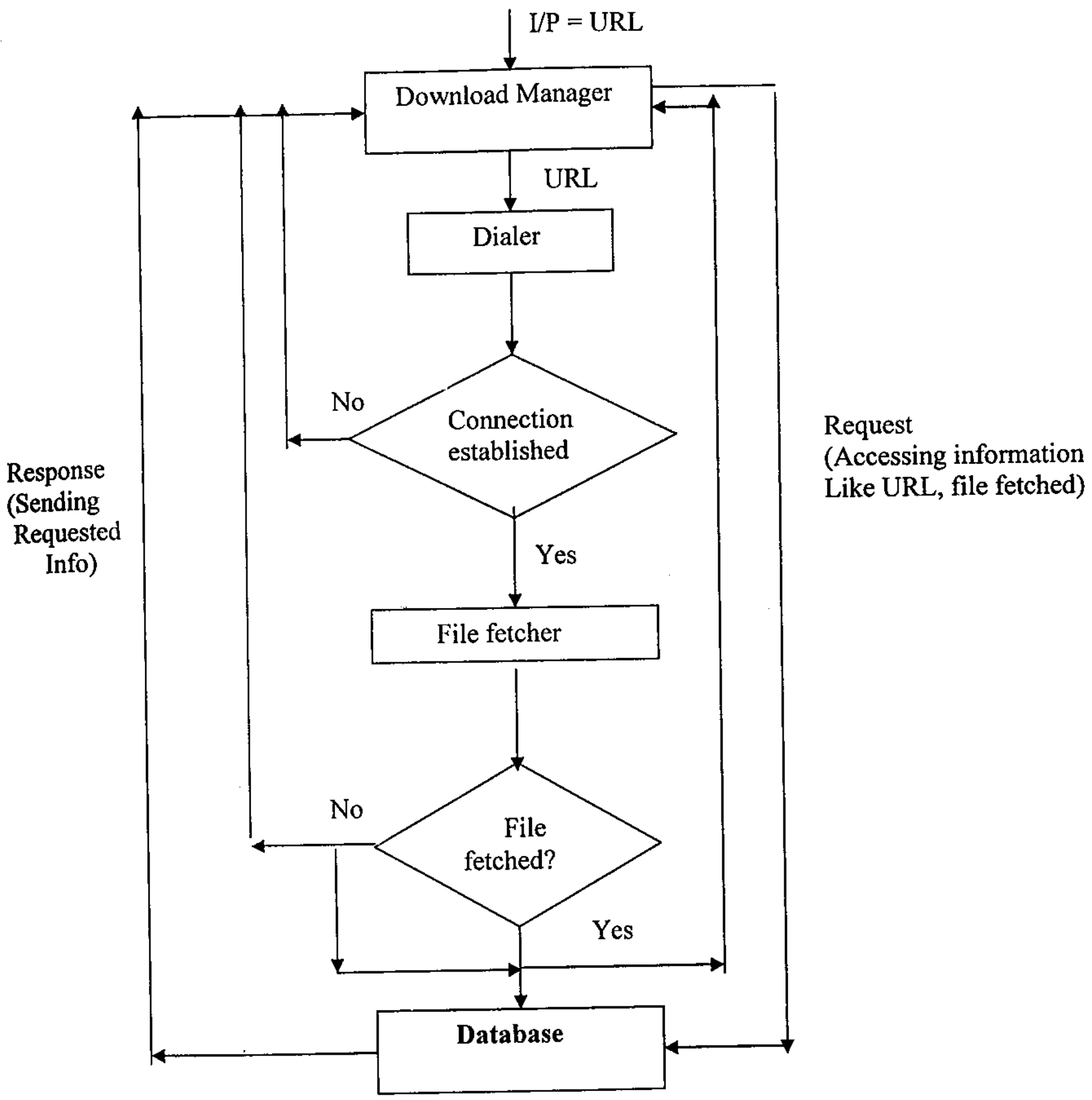


Figure 2.1

CHAPTER 3- DESIGNING

3.1 *Statement of Purpose*

The download manager downloads the files from URL provided the end user and manages downloading process. It can download files from the web server using Http and Ftp protocols. It can schedule a download of a particular file, resume a download from the instance where the connection breaks, pause in between the downloading process and cancel a download

3.2 *Context diagram*

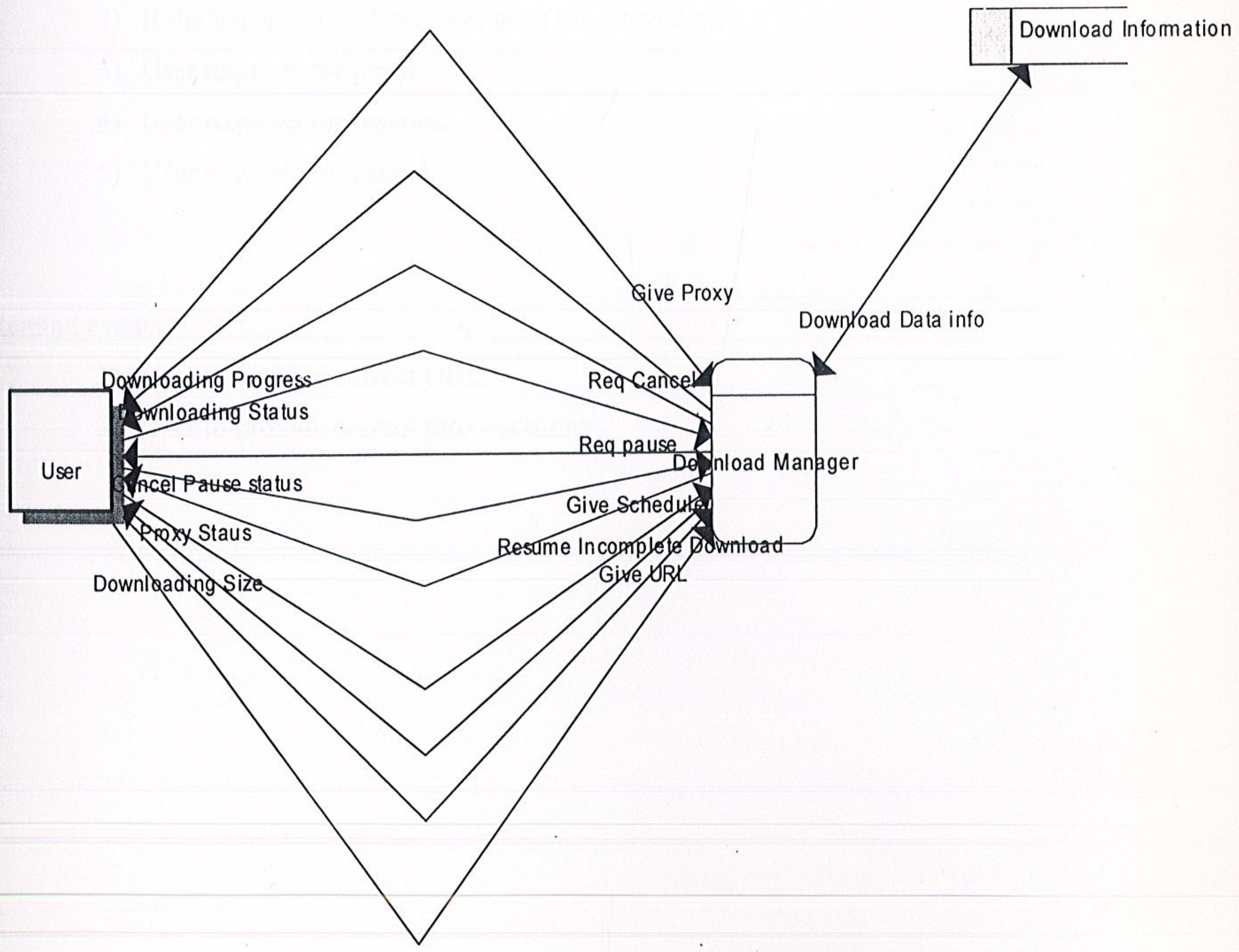


Figure 3.1

3.3 Event List

Flow events

1) User provides URL.
2) User gives schedule
3) User provides proxy.
4) User requests for incomplete downloads..

Control Flow event

1) If the downloading stops due to connection error. User asks for status.
2) If the net speed is slow. User goes for scheduling.
3) User requests for pause.
4) User requests for resume.
5) User requests for cancel.

Remind events

1) User to provide correct URL.
2) User to provide correct proxy settings.

3.4 *Cartesian Hierarchy*:

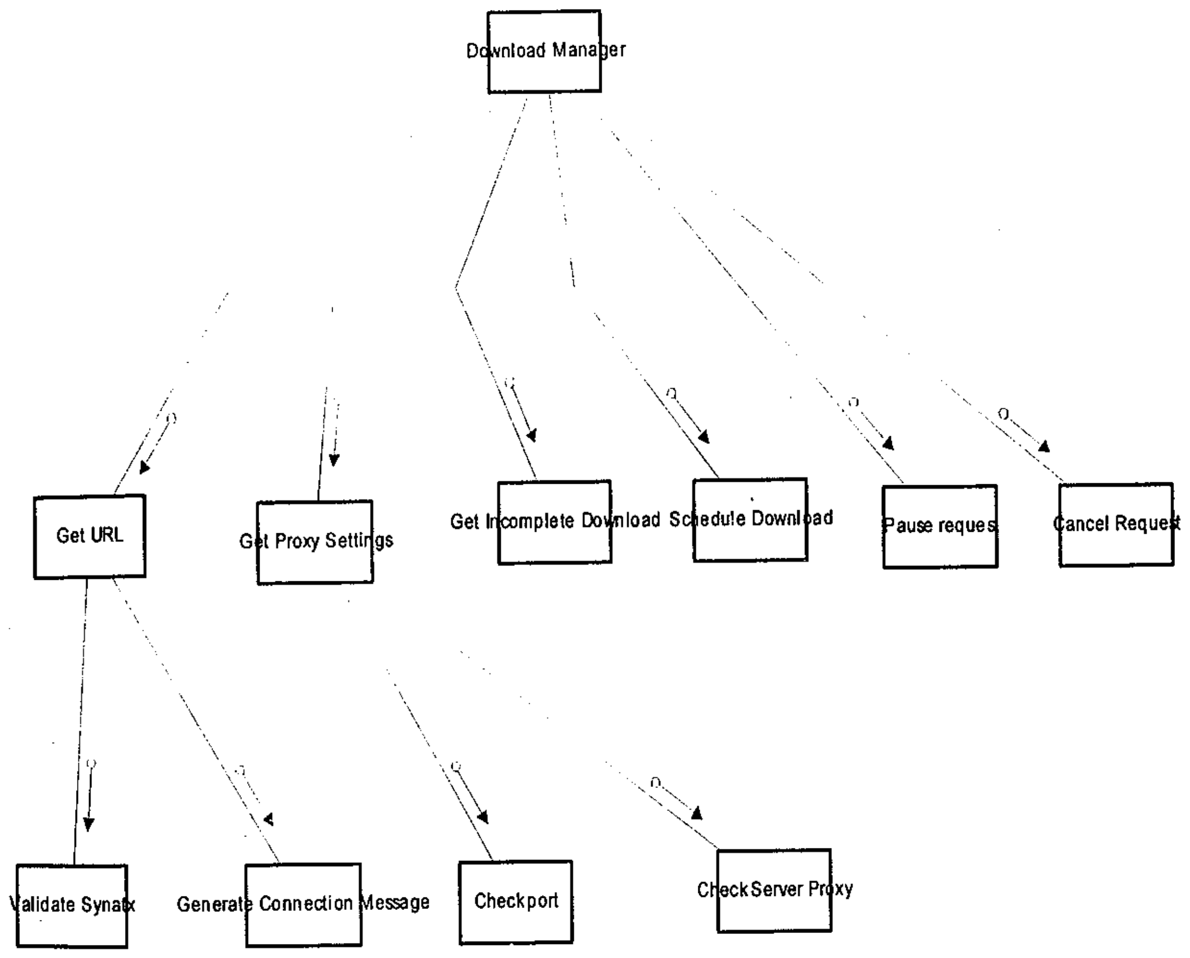


Figure 3.2

3.5 Data Flow Diagram:

Download a File

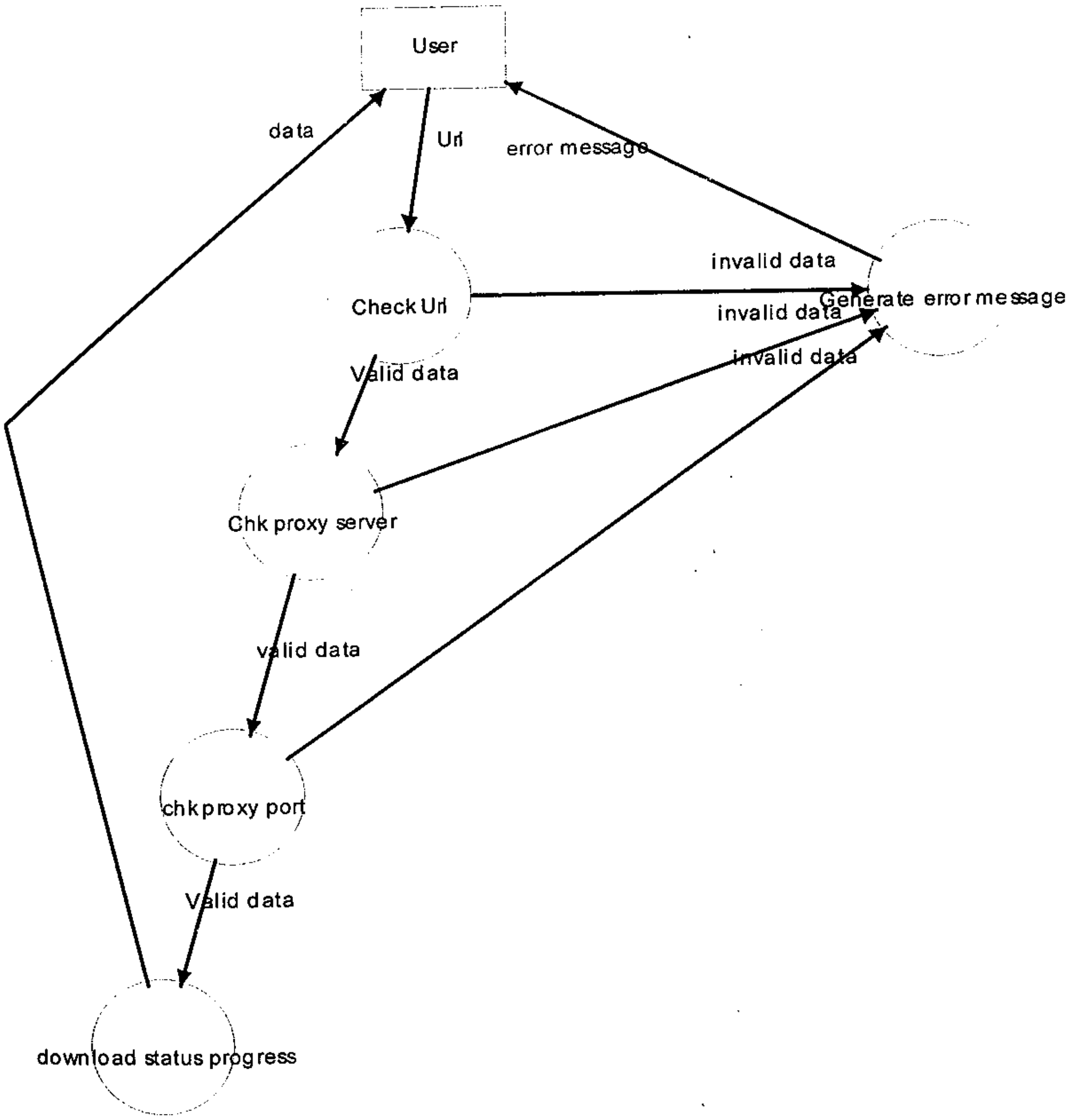


Figure 3.3

Schedule a download

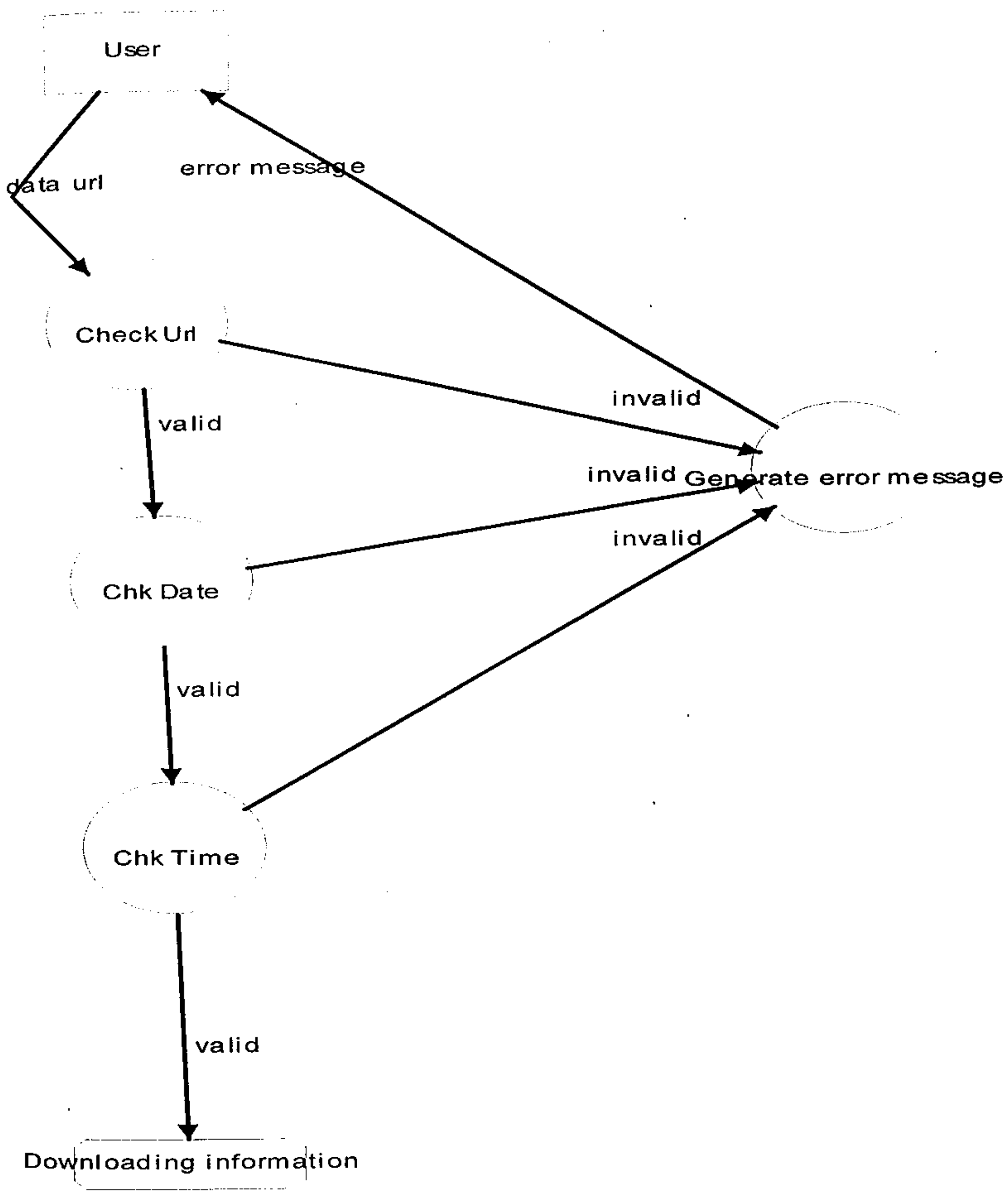


Figure 3.4

To pause a download

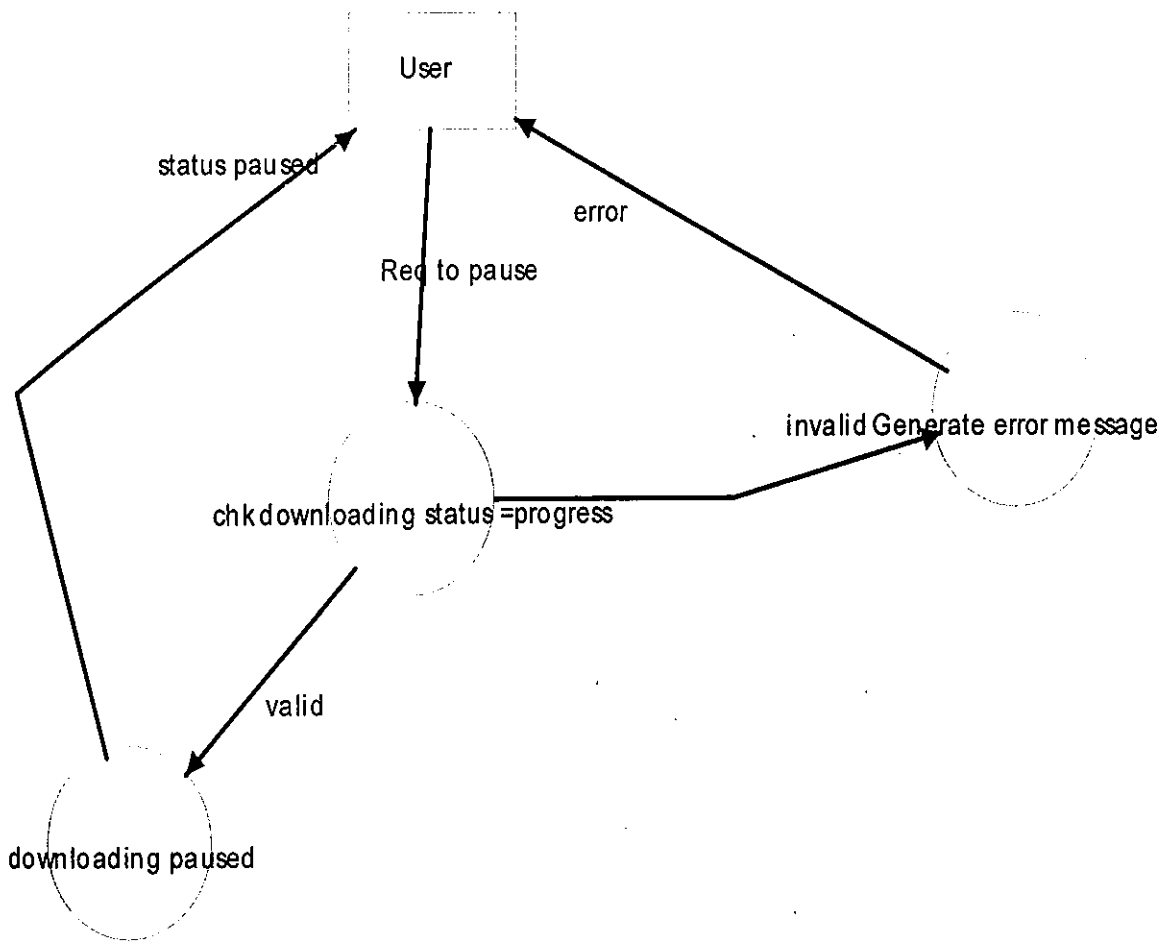

User
status paused
error
Req to pause
invalid Generate error message
chkdownloading status =progress
valid
downloading paused

Figure 3.5

To cancel a download

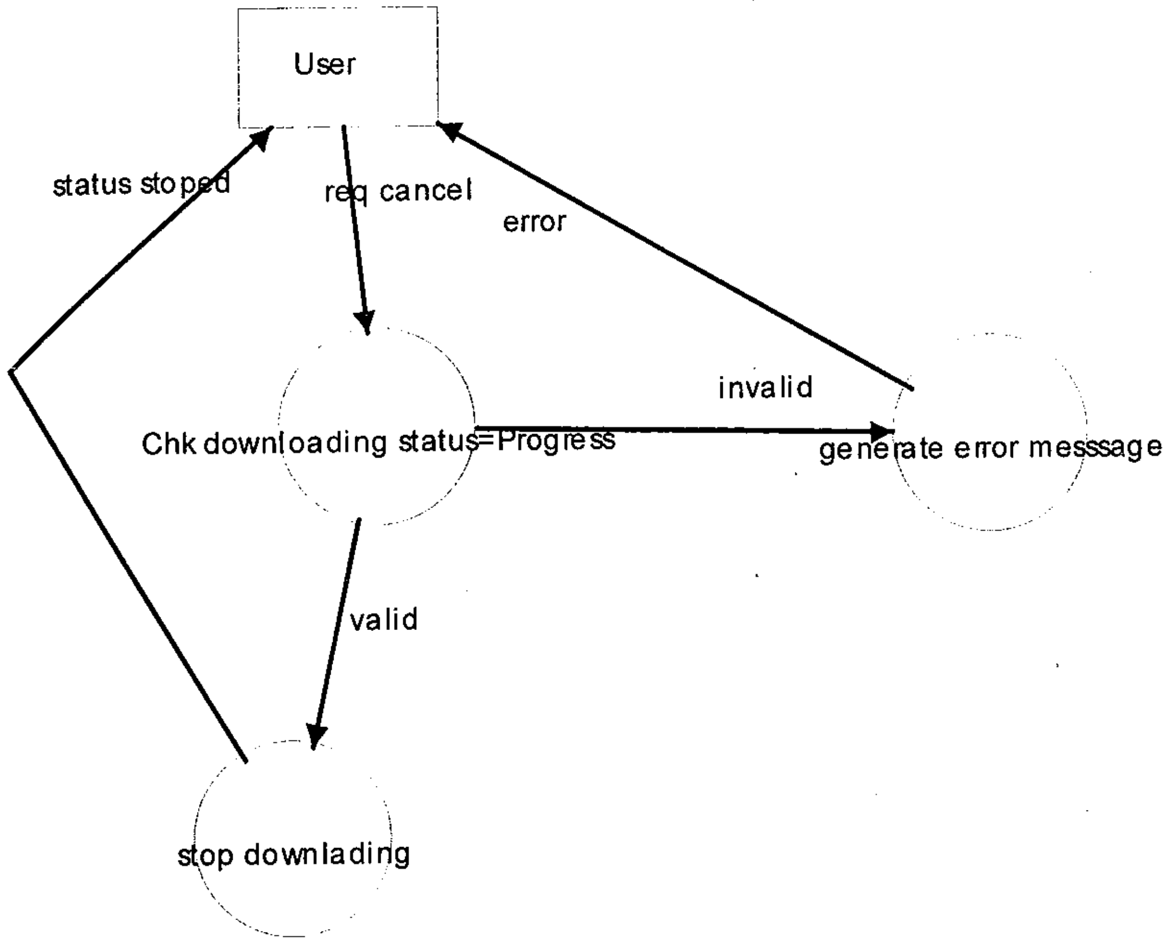


Figure 3.6

For Incomplete download

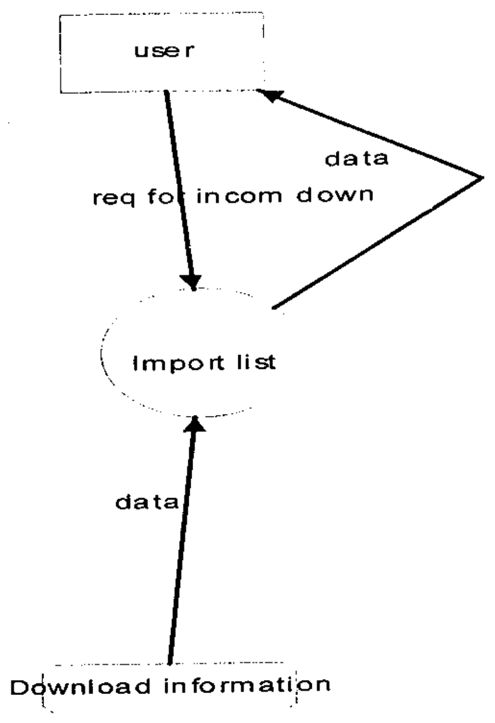


Figure 3.6

3.6 ***Process Specifications***

Precondition1:-

URL is valid.

Postcondition1:-

Establish the connection.

Precondition2:-

URL is invalid.

Postcondition2:-

Generate an error message.

Precondition3:-

Connection is established.

Postcondition3:-

Start downloading the file.

Precondition4:-

Connection is not established.

Postcondition4:-

Status=Error

Precondition5:-

Status=Error

Postcondition5:-

Save the download to database

Precondition6:-

Check downloading status =Progress

Postcondition6:-

Downloading is paused or cancelled.

Precondition7:-

Date not before current date

Postcondition7:-

Invalid scheduling

Precondition8:-

Time not before current time

Postcondition8:-

Invalid time

Precondition9:-

Date after current date

Postcondition9:-

You picked the right date

Precondition10:-

Time after current time

Postcondition10:-

Scheduling successful

3.7 ***Data Dictionary***:

URL = Protocol + DNS + Host name

Host = 2{Letter} 3

DNS = {Letter} 3

Proxy Server = A + B + C + D

A= 1 {digit} 3

B= 1 {digit} 3

C= 1 {digit} 3

D= 1 {digit} 3

Proxy port = 2 {digit} 4

Name = 1 {letter | digit} 20

Progress = 1 {digit} 10

Date= Day + Month + Year

Day= 1{digit} 2

Month= 1{digit} 2

Year= 4{digit} 4

Time=Hour + Minute + Second

Hour= 1{digit} 2

Minute= 1{digit} 2

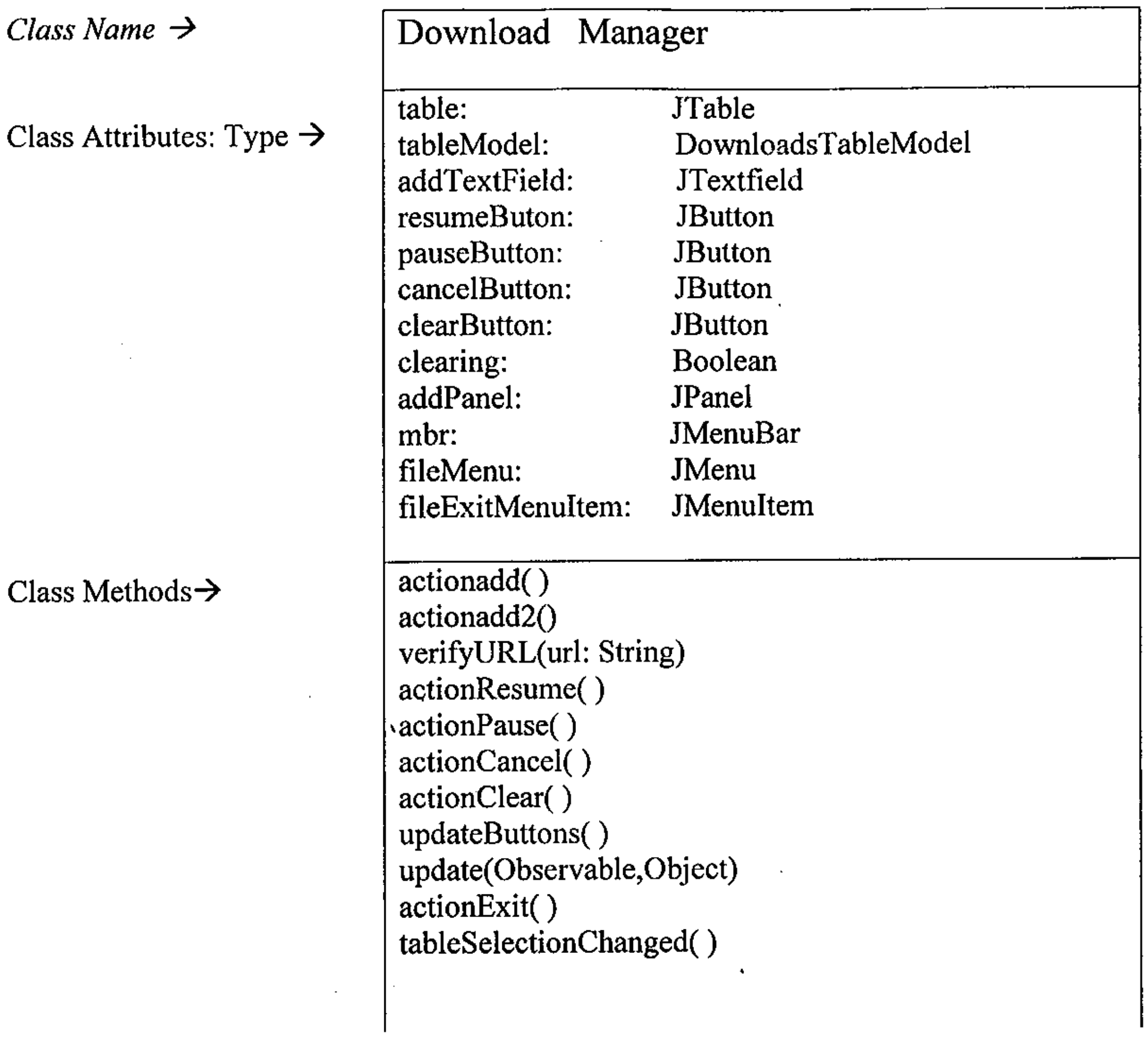
Second= 1{digit} 2

CHAPTER 4- IMPLEMENTATION

Why Java?

- Java applications are typically compiled to bytecode, now this bytecode can be run on any platform.
- Includes string and multi-thread support in the language.
- Automatic memory management.
- Data type sizes and arithmetic behavior are fixed and fully defined for all platforms.
- Have useful standard OO libraries.
- Documentation can be extracted from the source code.
- Security checking is built in to the libraries and virtual machine.
- Supports Unicode for ease of internationalization.
- Write once, run anywhere, any platform (no porting, no client configuration ... well almost!)
- Vast amount of supplier and programmer support and acceptance.
- Loads and runs over the WWW, 40 million potential clients.
- Extensive run-time type information and safe dynamic link-loading is available

4.1 Class Diagrams



Class Name →

Download Manager	
table:	JTable
tableModel:	DownloadsTableModel
addTextField:	JTextfield
resumeButon:	JButton
pauseButton:	JButton
cancelButton:	JButton
clearButton:	JButton
clearing:	Boolean
addPanel:	JPanel
mbr:	JMenuBar
fileMenu:	JMenu
fileExitMenuItem:	JMenuItem

Class Attributes: Type →

Class Methods→

actionadd( )
actionadd2()
verifyURL(url: String)
actionResume( )
actionPause( )
actionCancel( )
actionClear( )
updateButtons( )
update(Observable,Object)
actionExit( )
tableSelectionChanged( )

Figure 4.1

actionadd():- This function is called when JButton "Add Download " is pressed after providing the URL by the user. It will add the download if the URL is valid otherwise generate an error message. Return type is void.

actionadd2():- This function for the schedule URL. Return type is void verifyURL(String url):- Its input is the URL provided by the user. Verify the URL and return the object of URL type.

update(Observable,Object):- This function is automatically called as the selected download is changed or when download notifies its observers of any changes. Return type is void.

actionPause( ):- When user wants to pause a download this function is called and after this there is need to update the buttons on the provided GUI. Return type is void.

actionResume():- Paused download is resumed by this function and after it again there is need to update the buttons in the GUI. Return type is void.

actionCancel():- This function is called when the user wants to cancel the download. Return type is void.

actionClear():- User can easily manager the download list by this function. It will erase the download from the list. Return type is void.

updateButtons():- When download is managed(pause, resume, cancel, clear) by the user, there is need to update the buttons in GUI . This is done by this function. Return type is void.

actionExit():- When window closing event is occurred i.e..user wants to close the window then this function is called. Return type is void.

tableSelectionChanged():- As the user select any download from the download list, there is need to add and delete the observer. Return type is void.

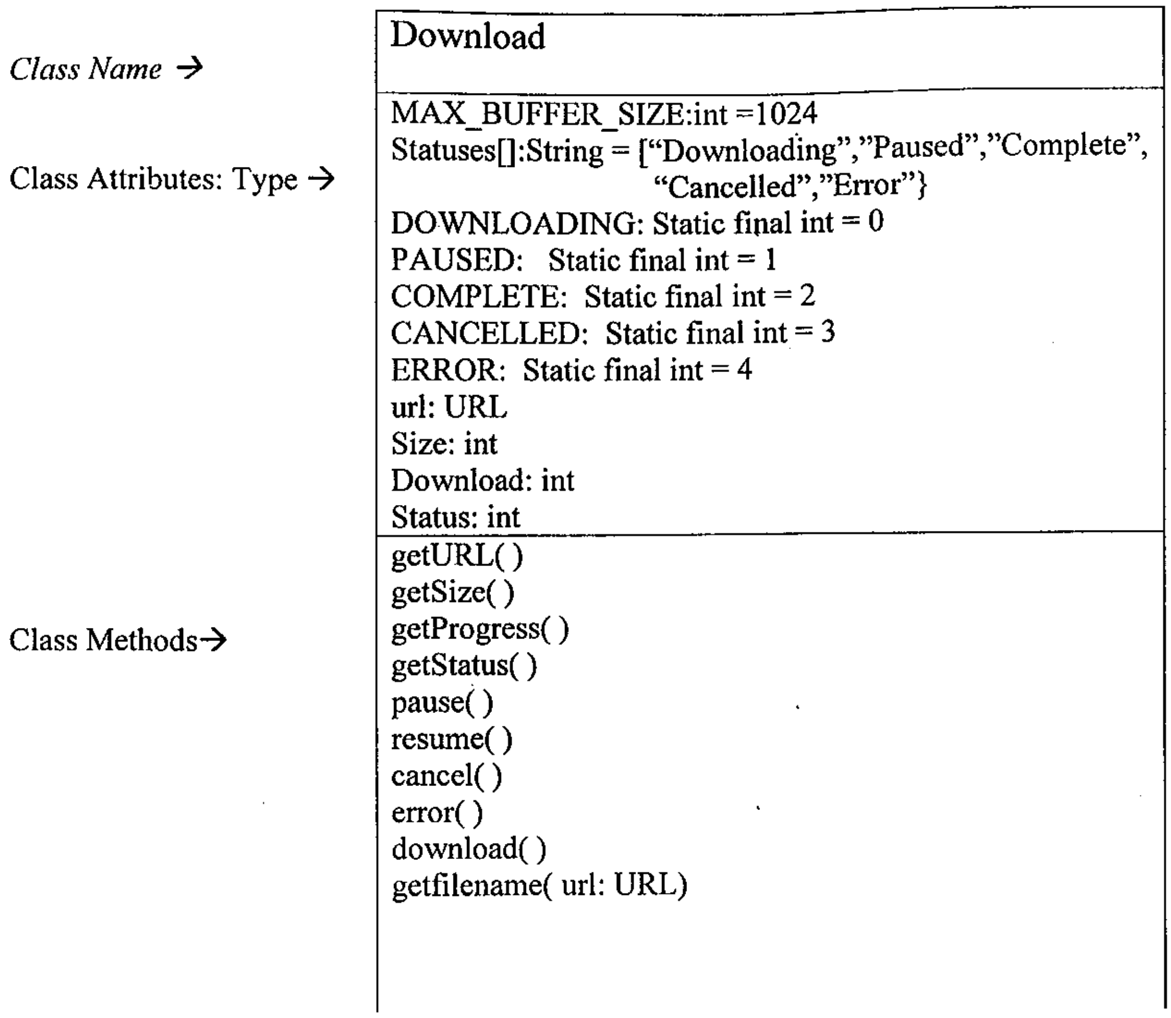

Class Name →
Class Attributes: Type →
Class Methods→
Download
MAX_BUFFER_SIZE:int =1024
Statuses[]:String = ["Downloading","Paused","Complete", "Cancelled","Error"}
DOWNLOADING: Static final int = 0
PAUSED: Static final int = 1
COMPLETE: Static final int = 2
CANCELLED: Static final int = 3
ERROR: Static final int = 4
url: URL
Size: int
Download: int
Status: int
getURL( )
getSize( )
getProgress( )
getStatus( )
pause( )
resume( )
cancel( )
error( )
download( )
getfilename(url: URL)

Figure 4.2

download():- This function will create new threads and start them as the new download is added by the user. Return type is void.

getURL():- It will return the URL as String.

getSize():- It will return the size of the download as an integer.

getProgress():- Progress is calculated and progress is returned as float.

getStatus():- This function return the status of download as an integer, its can be completed, Downloading, Error.

pause():- This function will set the status as PAUSED and then called the stateChanged() is called which set the changes and notify the changes to observer. Return type is void.

resume():- This function will set the status as RESUMED and then called the stateChanged() is called which set the changes and notify the changes to observer. Return type is void.
cancel():- This function will set the status as CANCELLED and then called the stateChanged() is called which set the changes and notify the changes to observer. Return type is void.

error():- This function will set the status as ERROR and then called the stateChanged() is called which set the changes and notify the changes to observer. Return type is void.

getfilename(URL url):- It provide the file name from the URL provided by the user. Return type is String

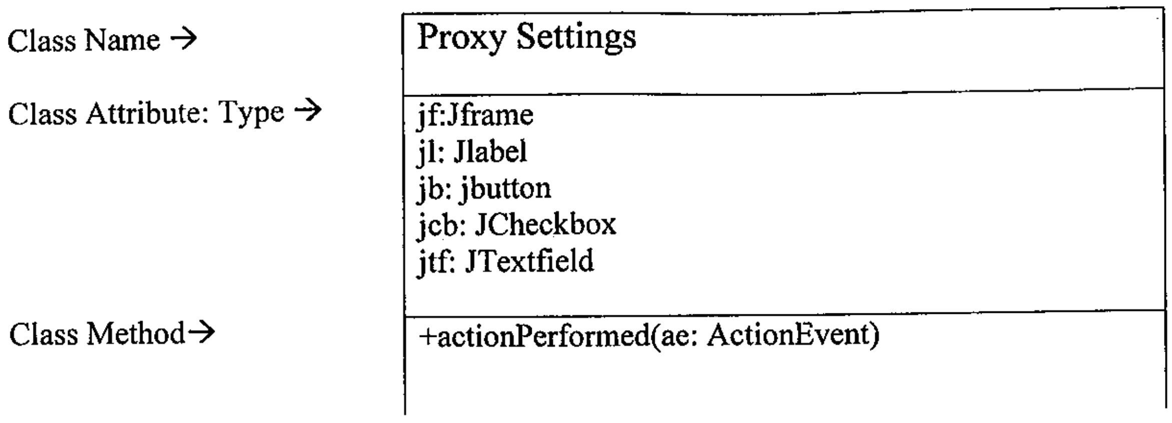


Figure 4.3

actionPerformed(ActionEvent):- This function is automatically called when any event is happened in JComponents

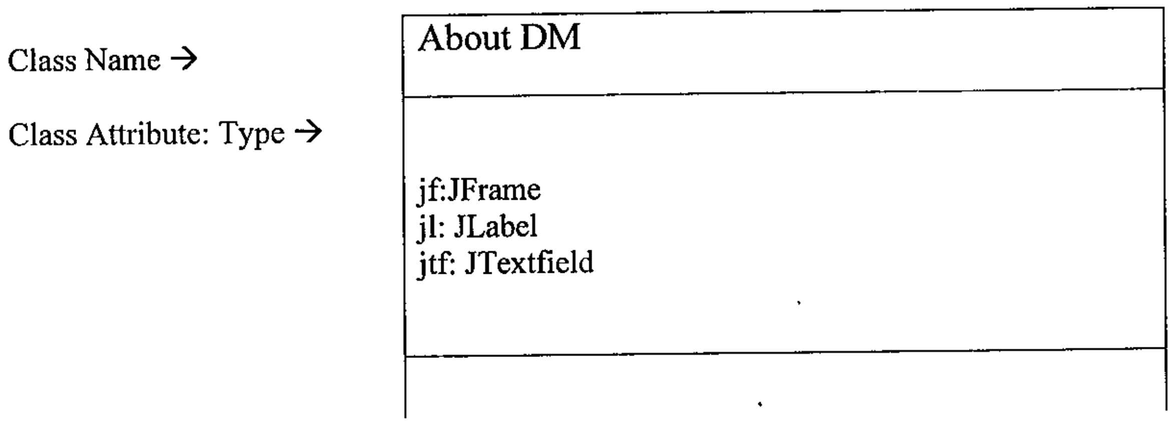


Figure 4.4

This class is interacting with Download class. Having the attributes of type JFrame, JLabel and JTextField.

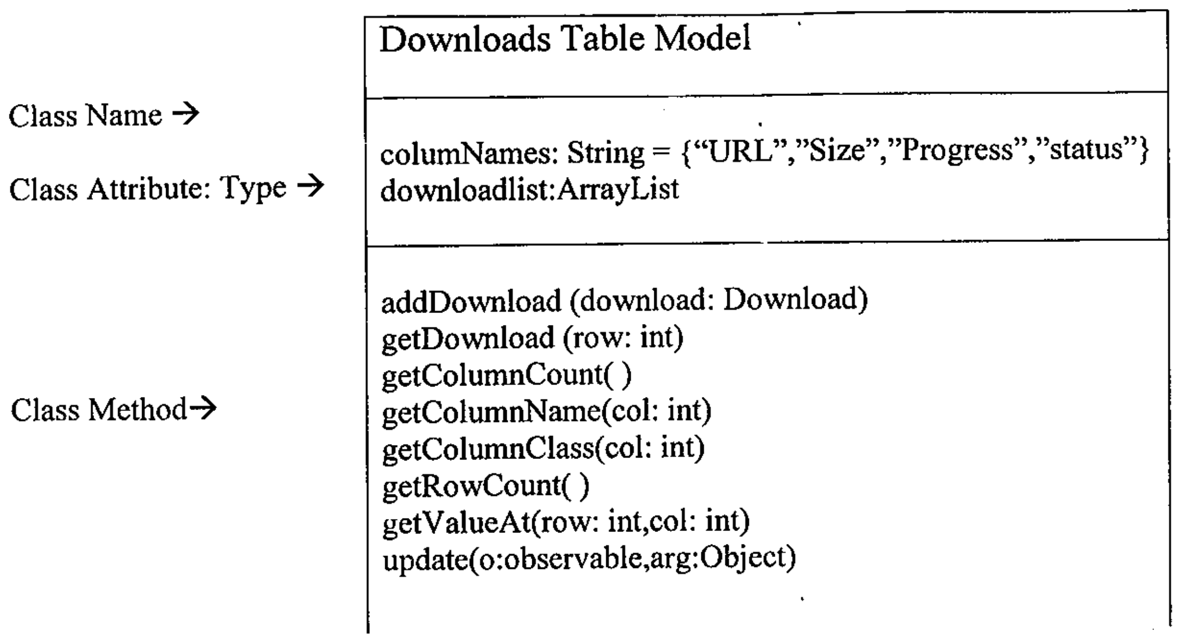


Figure 4.5

addDownload(Download download):- As the user click on JButton “Add Download” this function is called. It will add the observer to the download inorder to observe the changes and add the download to the download list. Return type is void.

getDownload(int row):-

getDownload(int row):- This function adds the download in the passed row. Return type is the object of download class.

getColumnCount():- It returns the column count as an integer.

getColumnName(int col):- This function returns the column name as String of the passed column.

getRowCount():- This function returns the row count of the download list as an integer.

getValueAt(int row, int col):- This function returns the value of downloads of specified row and col. Returns the object of Object.

update(observerable o, Object arg):- This function is automatically called as the selected download is changed or when download notifies its observers of any changes. Return type is void.

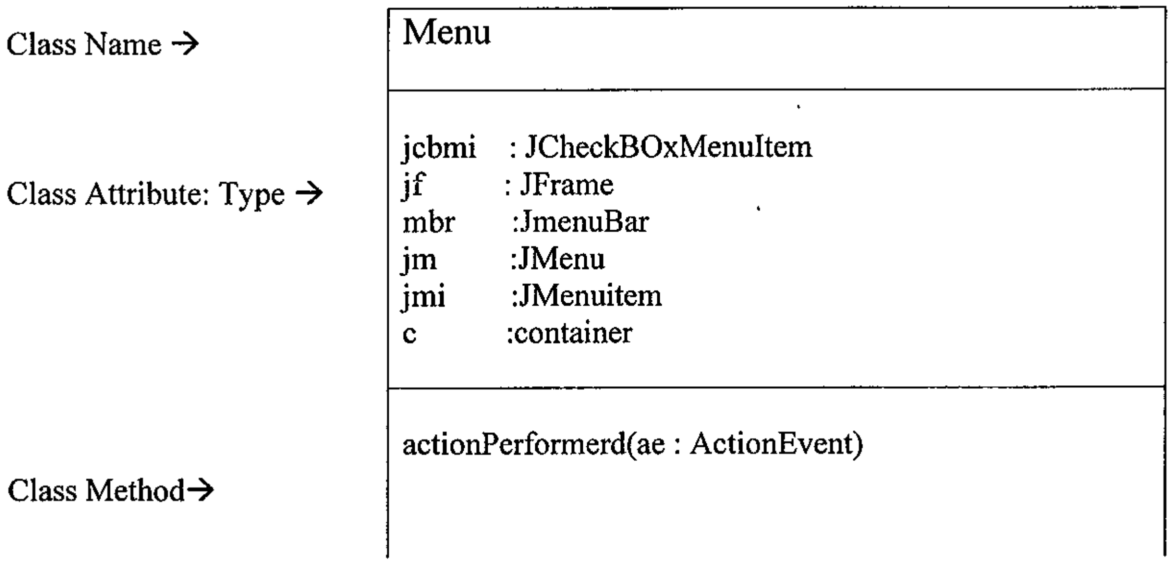


Figure 4.6

This class is interacting with JtabbedPaneDemo class. It contains the attributes of type JCheckBoxMenuItem, JFrame, JMenuBar, JMenu, JMenuItem , Container.

actionPerformed(ActionEvent):- This function is automatically called as any event occurs on the menu items in the menu on the menu bar. Return type is void.

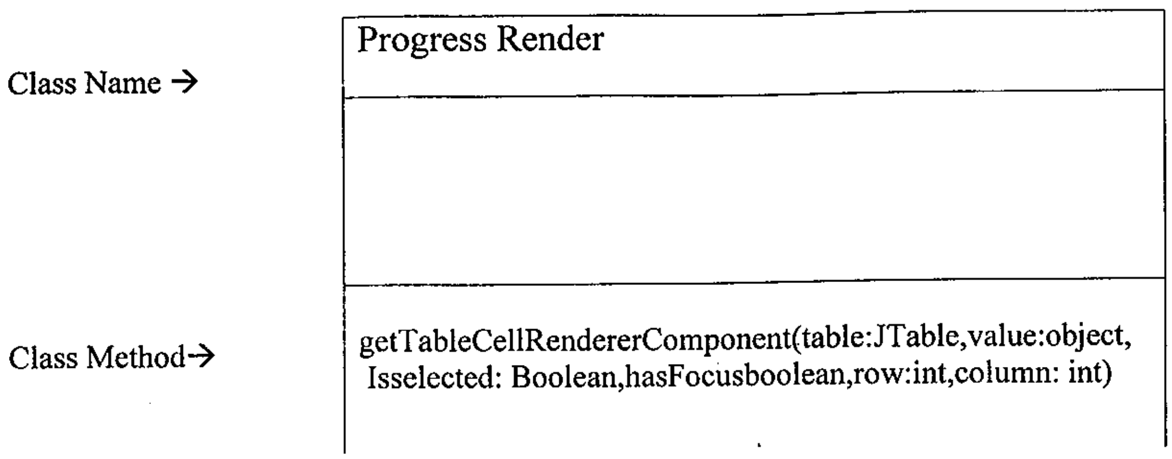

Class Name →
Progress Render
Class Method→
getTableCellRendererComponent(table:JTable,value:object,
Isselected: Boolean,hasFocusboolean,row:int,column: int)

Figure 4.7

getTableCellRendererComponent(JTable, Object):- Returns the JProgressBar as the renderer for the given table cell and set JProgressBar's percent complete value.

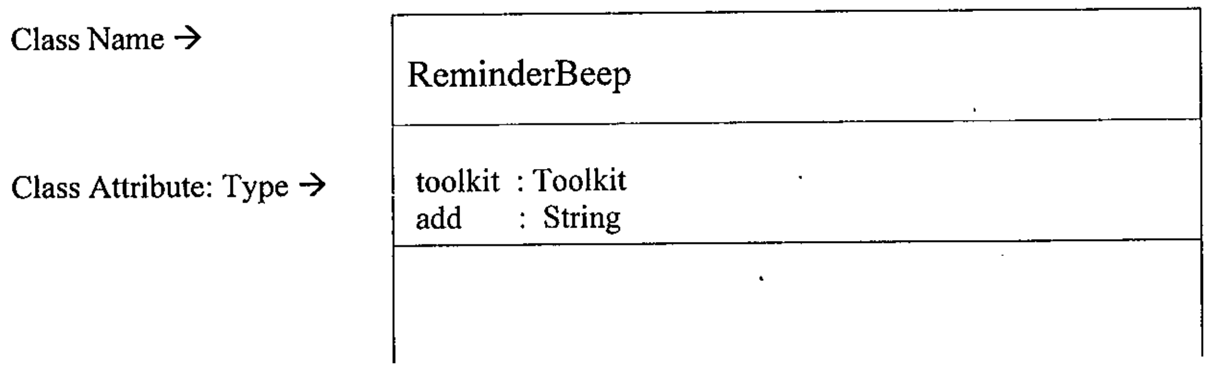


Figure 4.8

This class reminds as the scheduled download is going to be executed. It has attributes of type Toolkit and String. It interacts with the RemindTask class as creating its object.

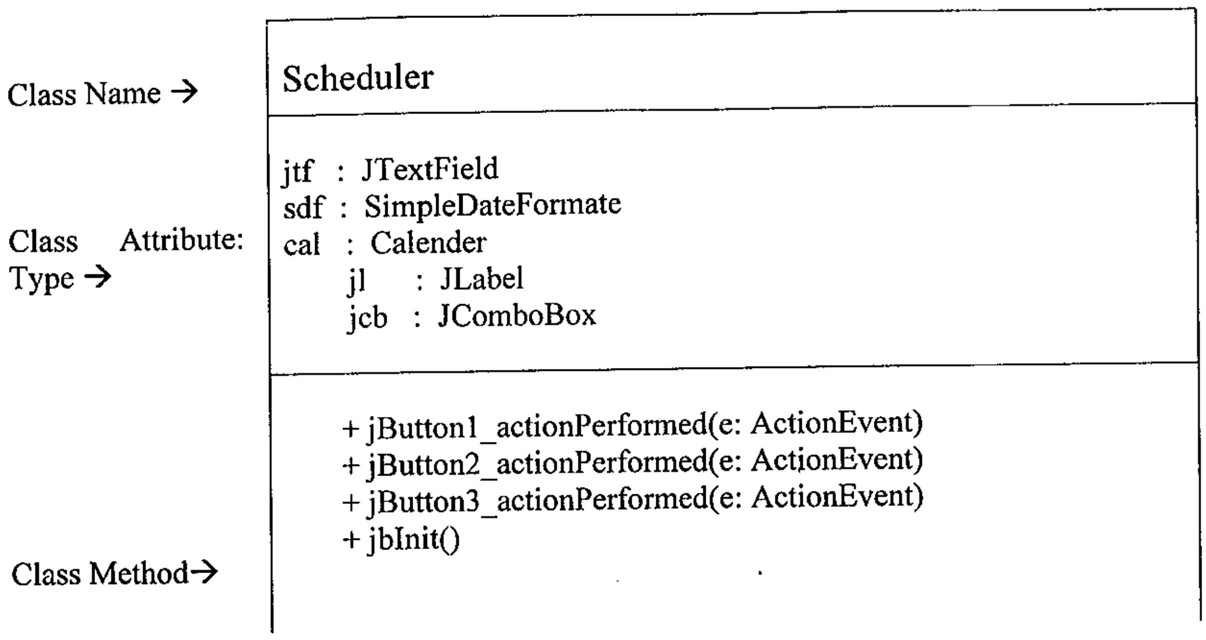


Figure 4.9

This class interacts with ReminderBeep and DatePickerSimple classes. It has attributes of type JTextField, SimpleDateFormate, Calender, Jlabel, JComboBox.

jButton1_actionPerformed(ActionEvent e):- This function is automatically called as the user click on the JButton "Date Picker". Then it calls the DatePickerSimple class and pick the date. Return type is void.

jButton2_actionPerformed(ActionEvent e):- This function is called as the user click on the JButton "Schedule". Provide the way to select the day, month and year. And interacts with ReminderBeep class. Return type is void.

jButton3_actionPerformed(ActionEvent ae):- This function is called as the user click on the JButton "Close" and performs the action. Return type is void.

jbInit():- This function calls as the this class object is created. It takes the input from the user as URL and description and provide the option of date picking, scheduling. Return type is void.

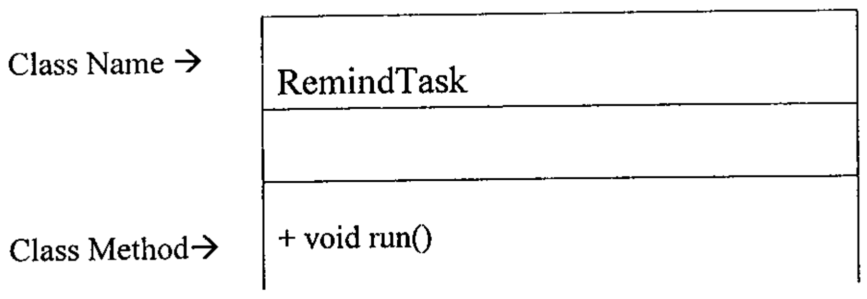


Figure 4.10

void run():- This function is called when download manager wants to run the schedule downloads. Now this class interacts with the DownloadManager class by creating its object. Rteurn type is void.

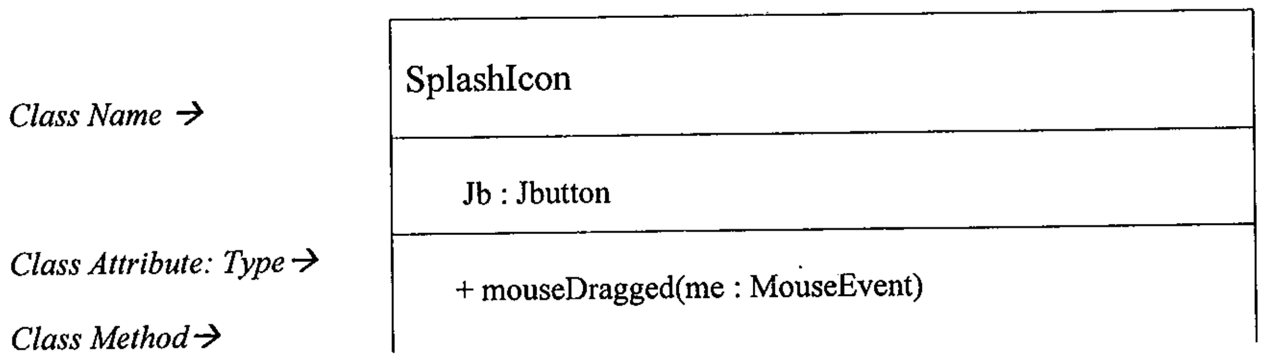


Figure 4.11

This class provides a JButton which shows the speed of download in bytes per sec.

mouseDragged(MouseEvent me):- This function is automatically called as the mouse goes on the JButton.

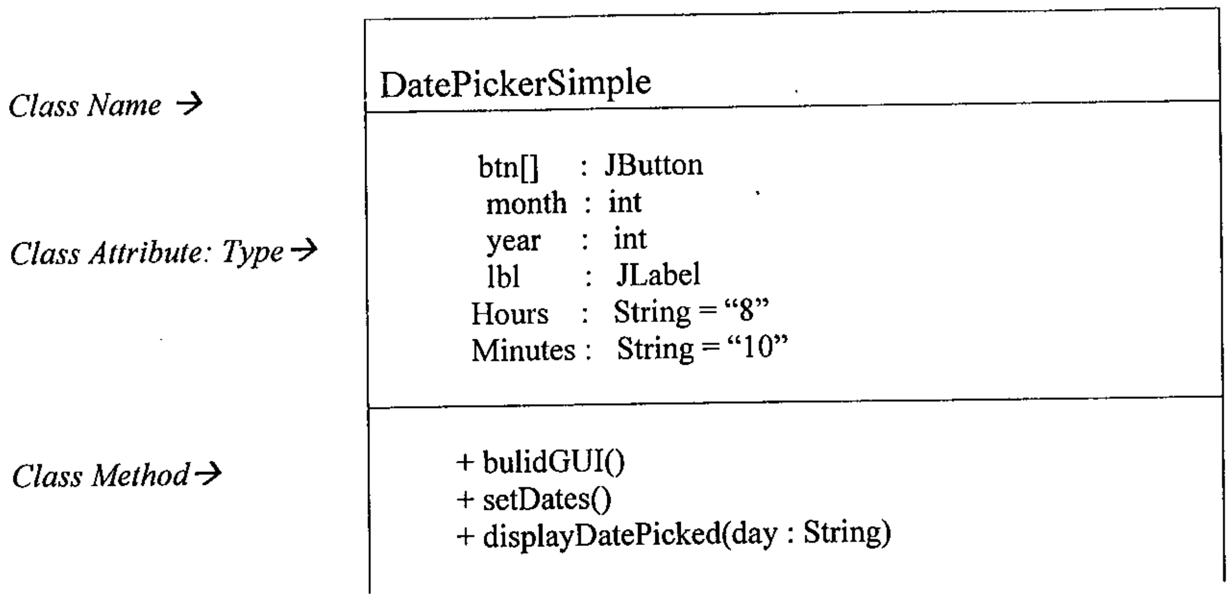


Figure 4.12

This class has attributes of type JButton, JLabel, String and integer.

bulidGUI():- This function provides the GUI from which the user can choose the date. Return type is void.

setDates():- This function sets the date which is selected by the bulidGUI(). Return type is void.

displayDatePicked(String day):- This function displays the date picked by the user. Return type is void.

4.2 *Interaction of Classes*

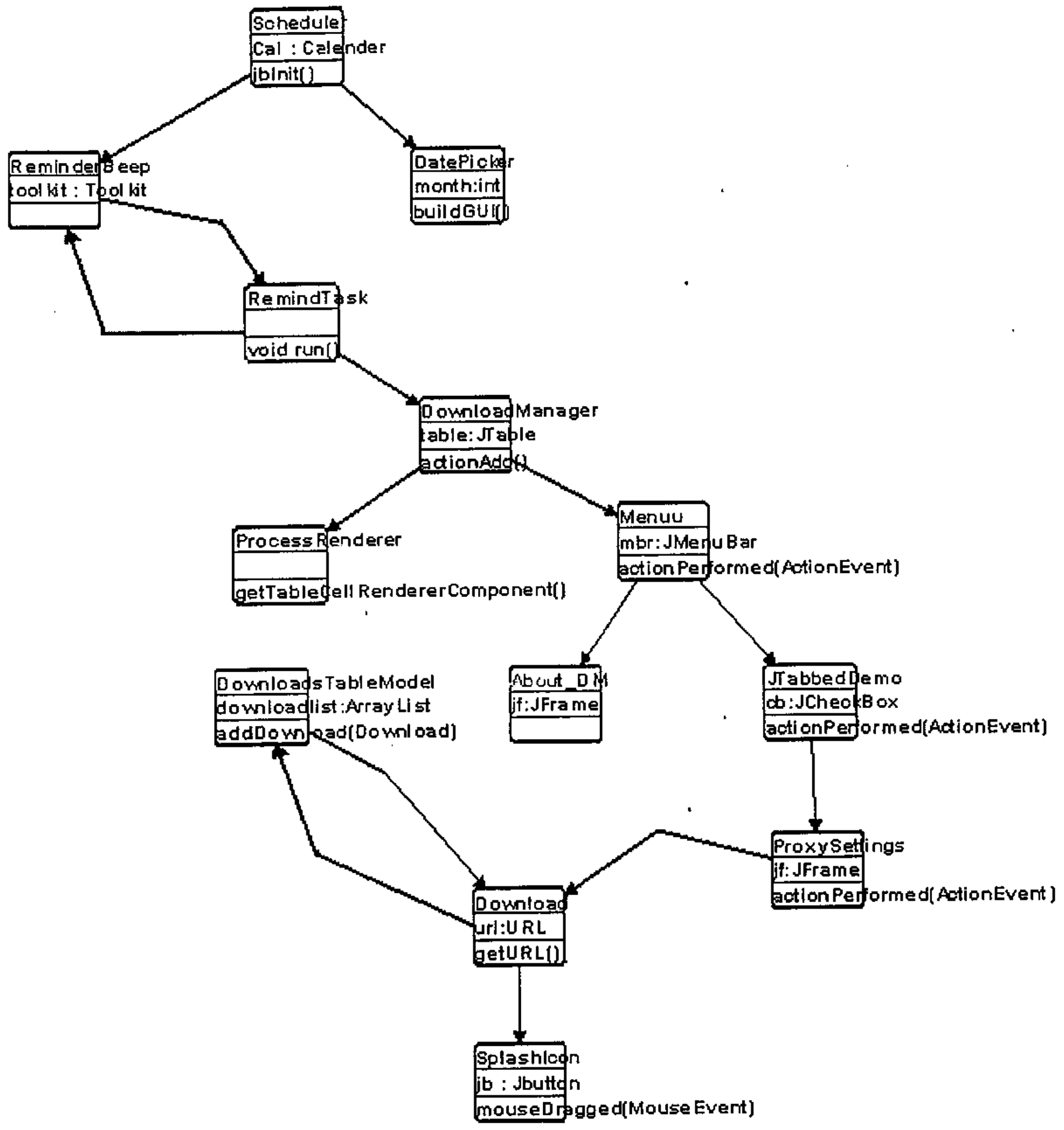


Figure 4.13

4.3 Developed a GUI prototype

4.3.1 Main Window

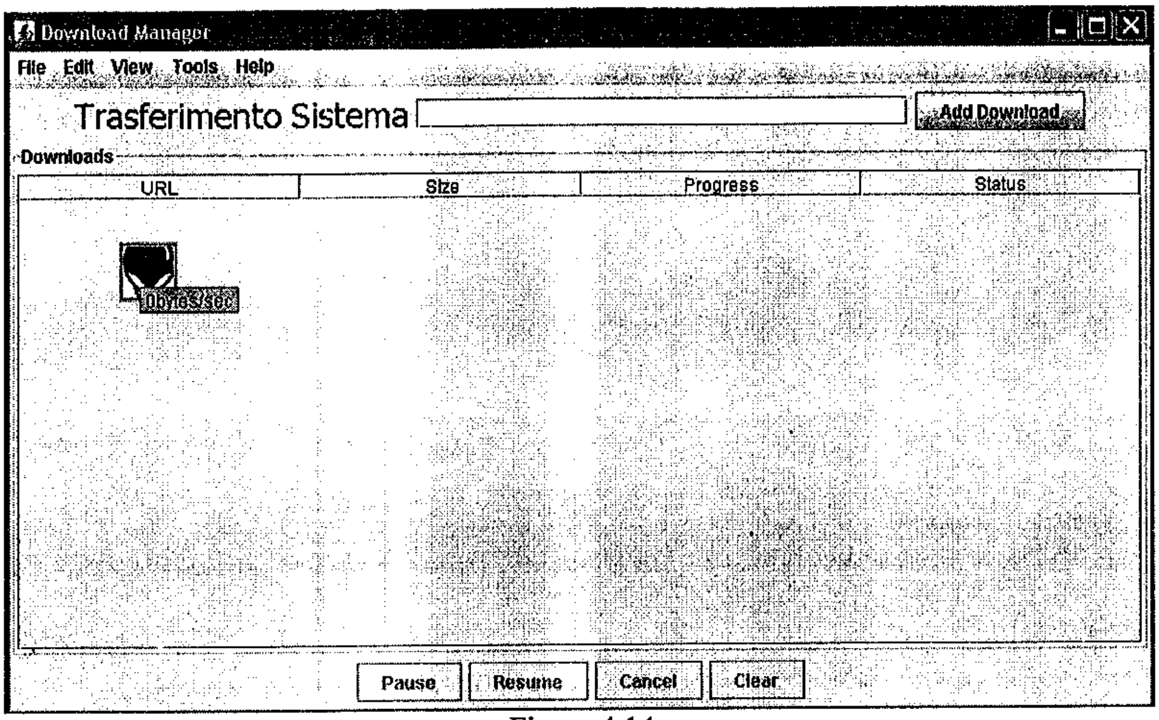


Figure 4.14

Working
How to download a file?

1. First connect to the internet.
2. Go to tool menu and then click Connect option
3. Click Set proxy, give proxy server and port
3. Write or paste the URL in the text box provided.
4. Click Add download

These are shown in the in the figures below in the form of steps.

Step 2

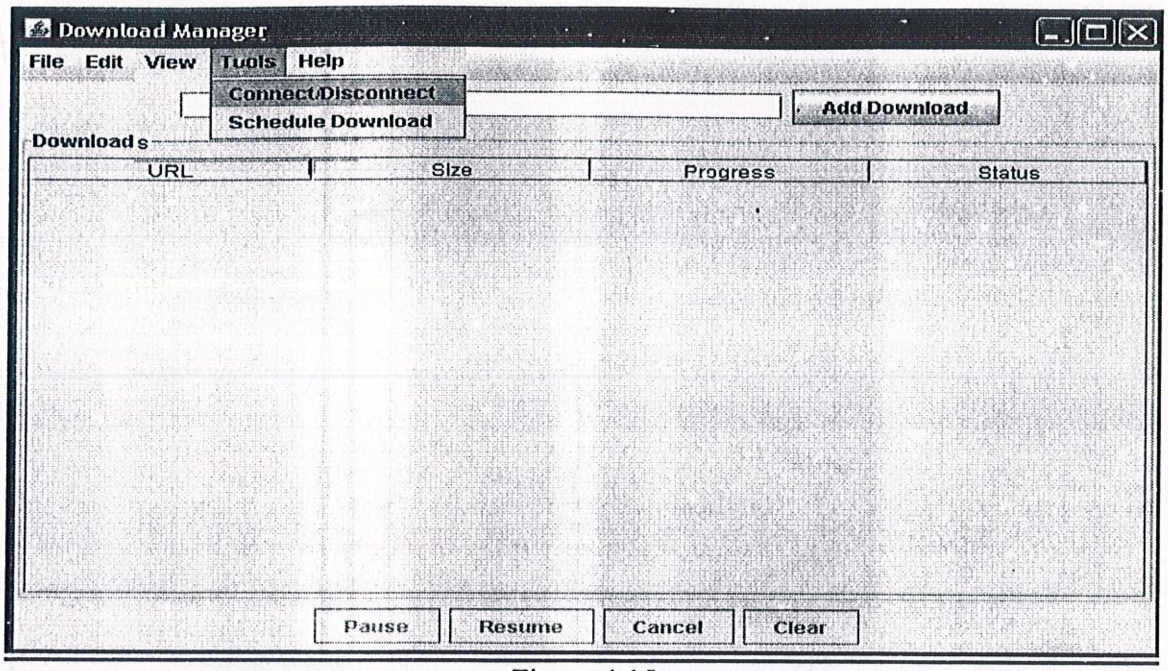
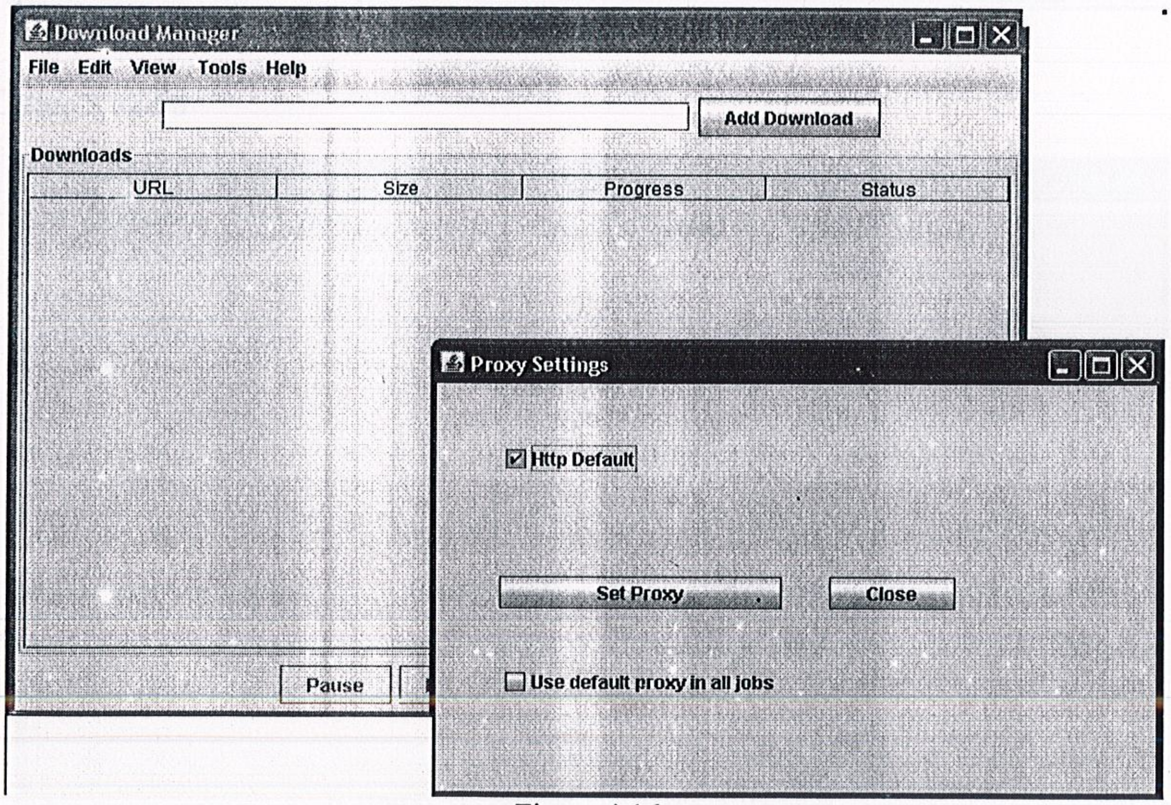


Figure 4.15

Step 3

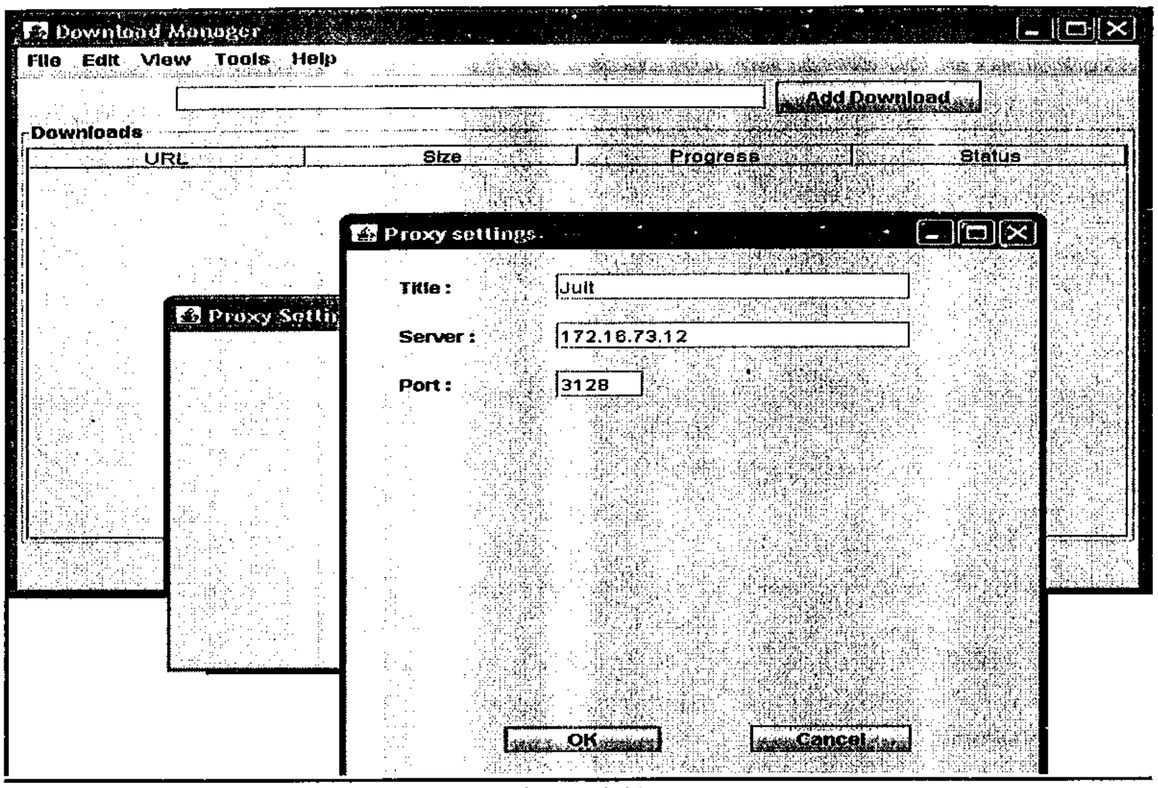


☑ Http Default

☐ Use default proxy in all jobs

Figure 4.16

Step 4



Title : Juit

Server : 172.16.73.12

Port : 3128

Figure 4.17

Step 5 and 6

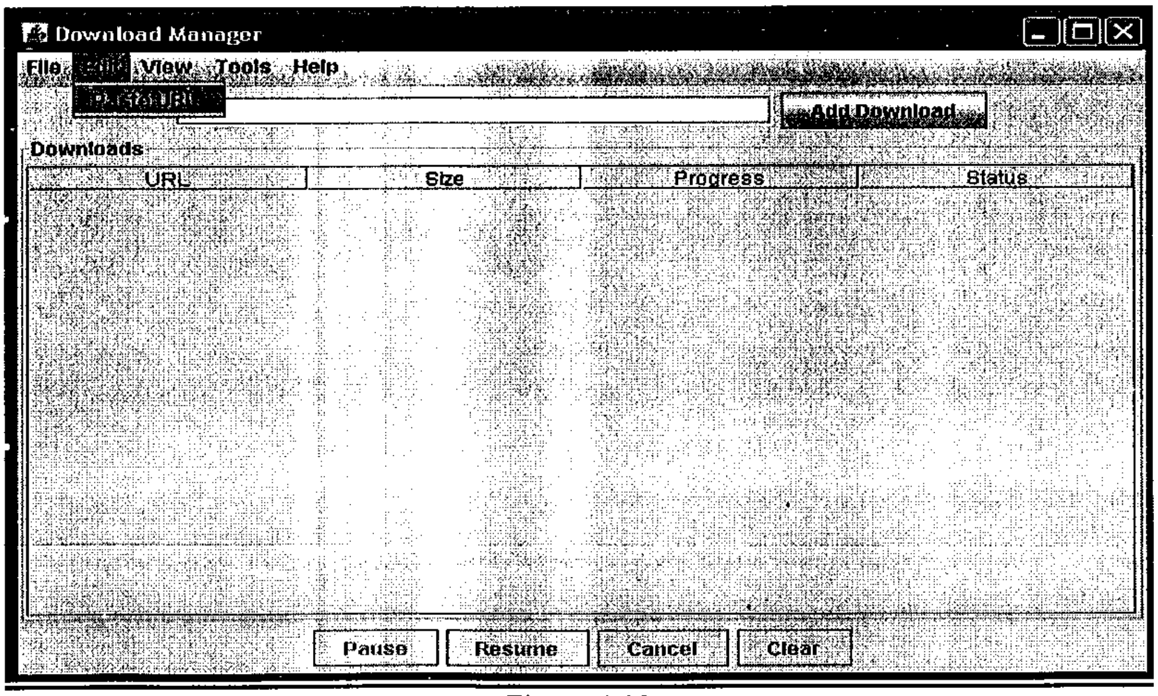
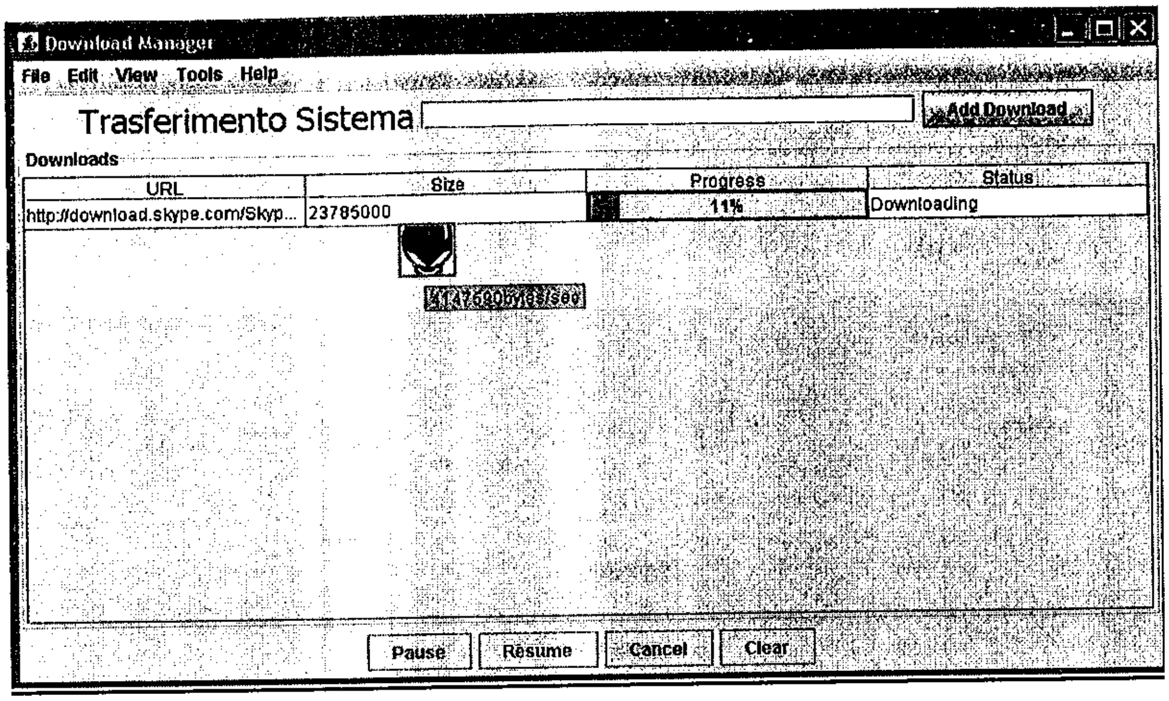


Figure 4.18

Result – The download manager starts downloading the file from the URL address provided by the user.



URL	Size	Progress	Status
http://download.skype.com/Skyp...	23785000	11%	Downloading

Figure 4.19

What actions user can perform after downloading starts?

1. User can pause a particular download by selecting that download.
2. User can cancel a particular download by selecting that download.

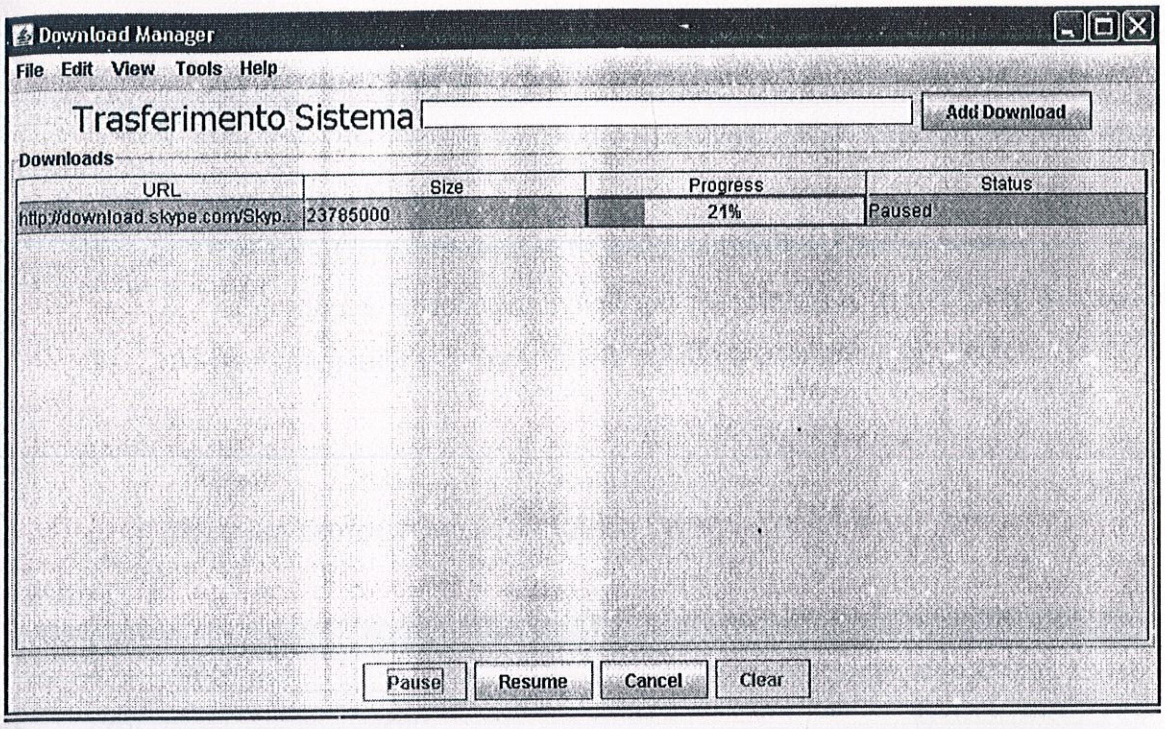
What happens after user pause a particular download?

Resume and cancel option gets highlighted as shown in the figure below.

Now the user can either resume or cancel the download.

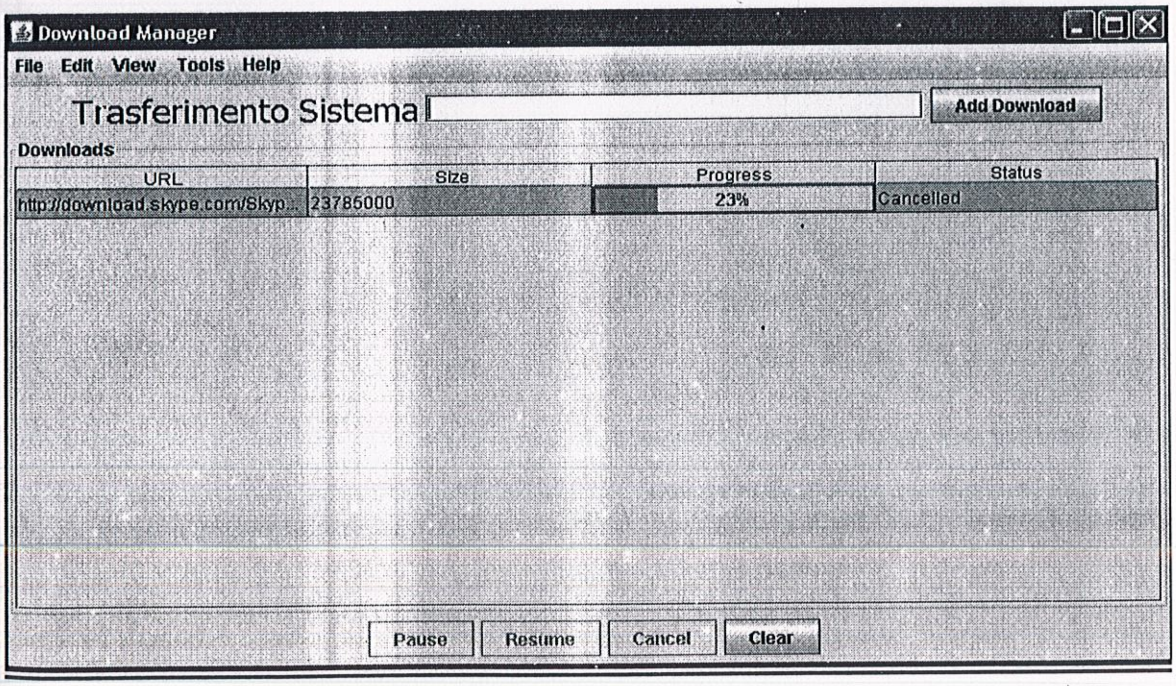
If the user decides to resume the download then the file again starts again downloading and the status "paused" is changed to "downloading". This is shown is the figure below.

If the user decides to cancel the download then Clear button gets highlighted and status is changed to "cancelled". Now the user press Clear button to clear off the download from the download table.



URL	Size	Progress	Status
http://download.skype.com/Skyp...	23785000	21%	Paused

Figure 4.20



URL	Size	Progress	Status
http://download.skype.com/Skyp...	23785000	23%	Cancelled

Figure 4.21

How to schedule a download?

1. Go to the tools menu and select schedule download.
2. Then the scheduling window will appear. Write the URL address of the file download user wants to schedule and the description of the download.
3. Click on the date picker button to select the date on which the download should be scheduled.
4. Click on the time at which the download should start on the date which was picked before.
5. Then click schedule button to schedule the download.
6. After step 5 is done, user will get the message "Scheduling successful. Executing after x years y month's z day's u hour's v minutes".

These are shown in the in the figures below in the form of steps

Step1

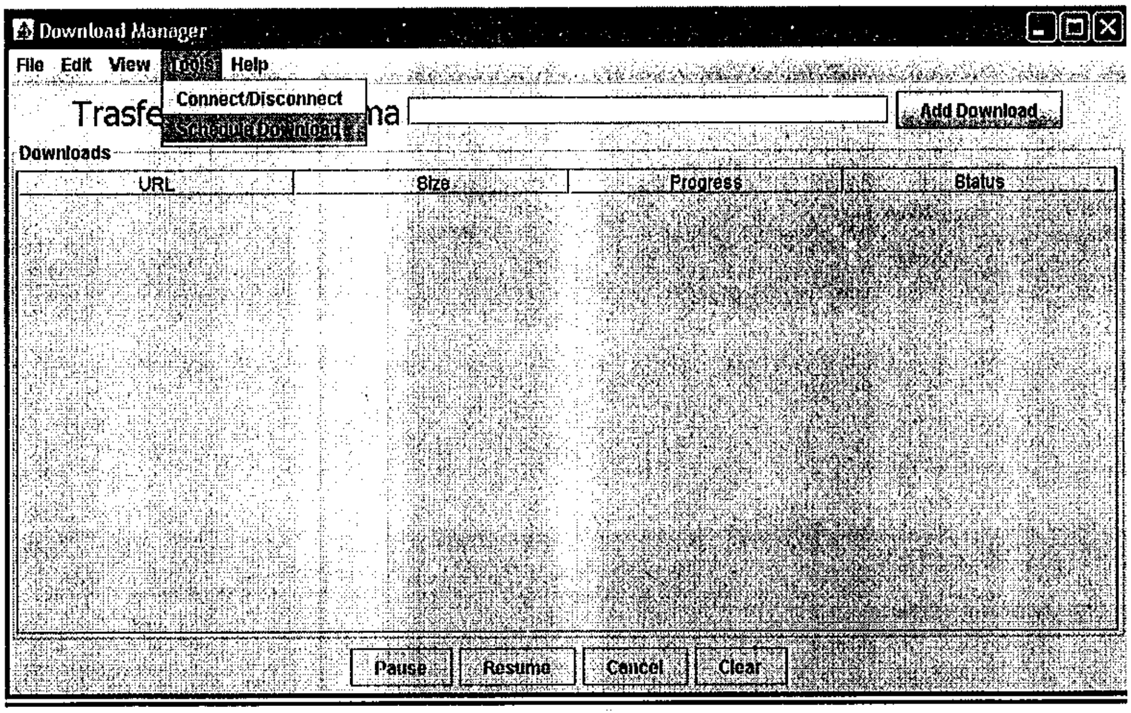


Figure 4.22

Step2

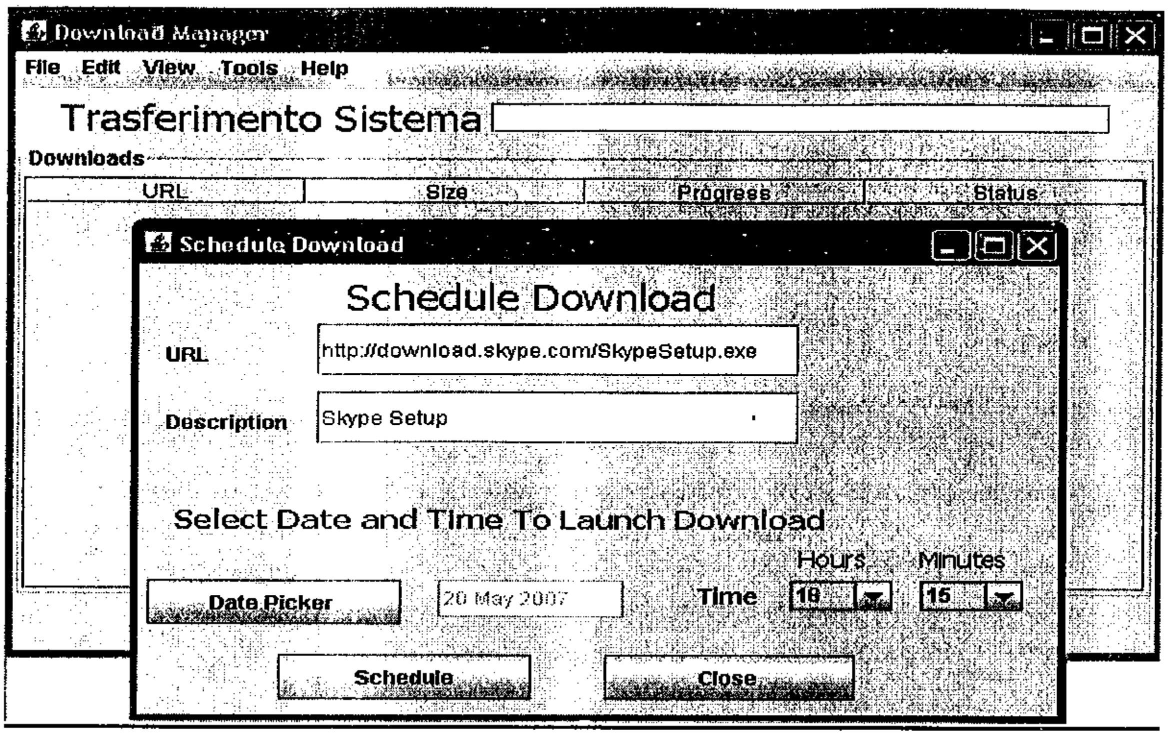


Figure 4.23

Step3

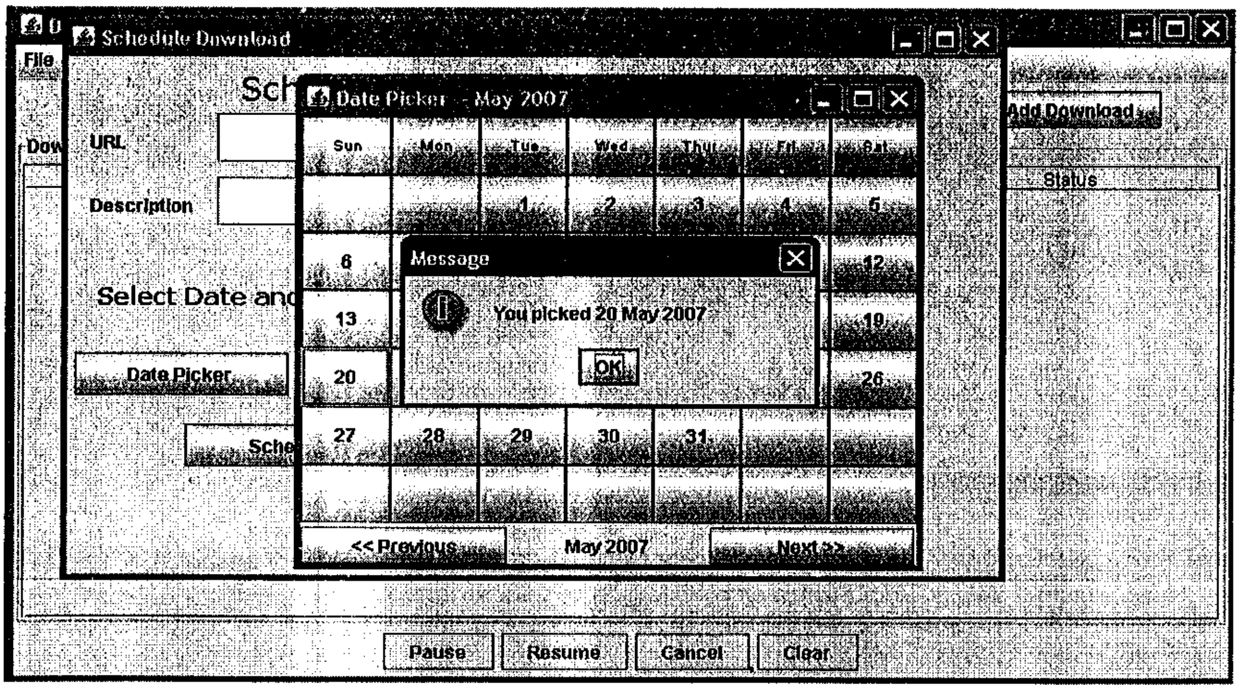
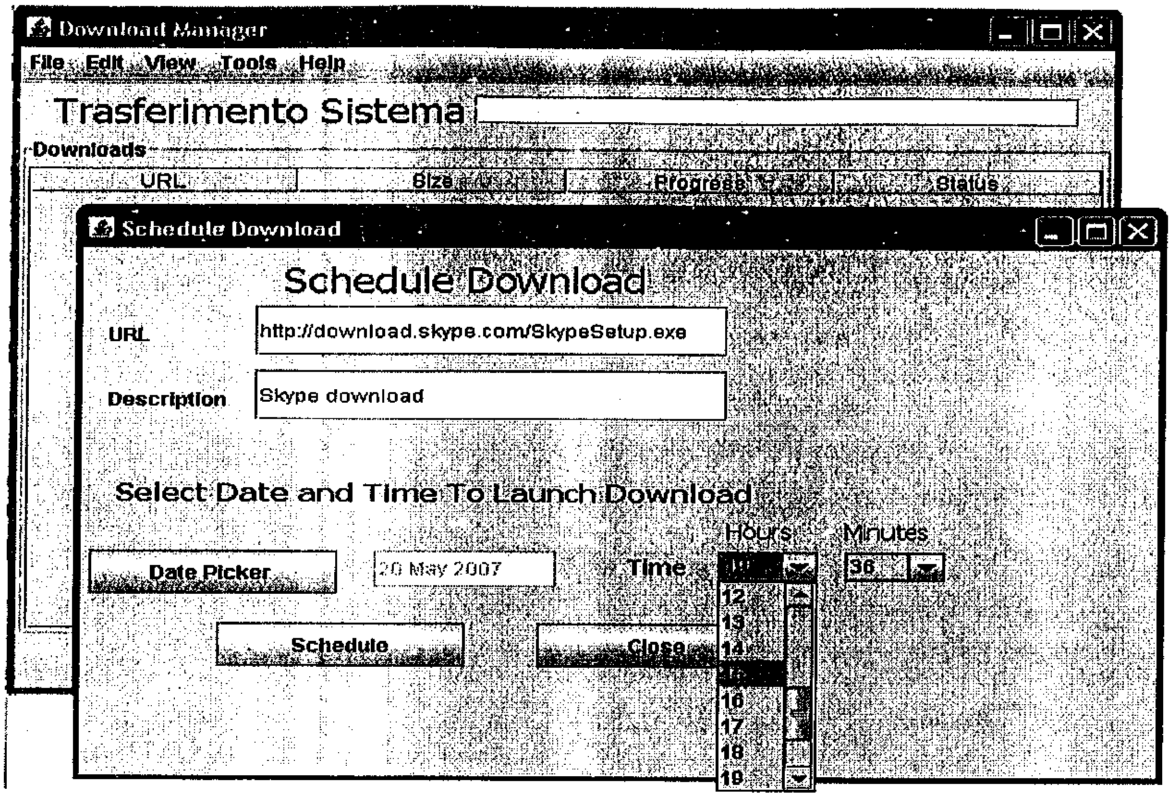


Figure 4.24

Step4


Download Manager

File Edit View Tools Help

Trasferimento Sistema

Downloads

URL	Size	Progress	Status

Schedule Download

Schedule Download

URL http://download.skype.com/SkypeSetup.exe

Description Skype download

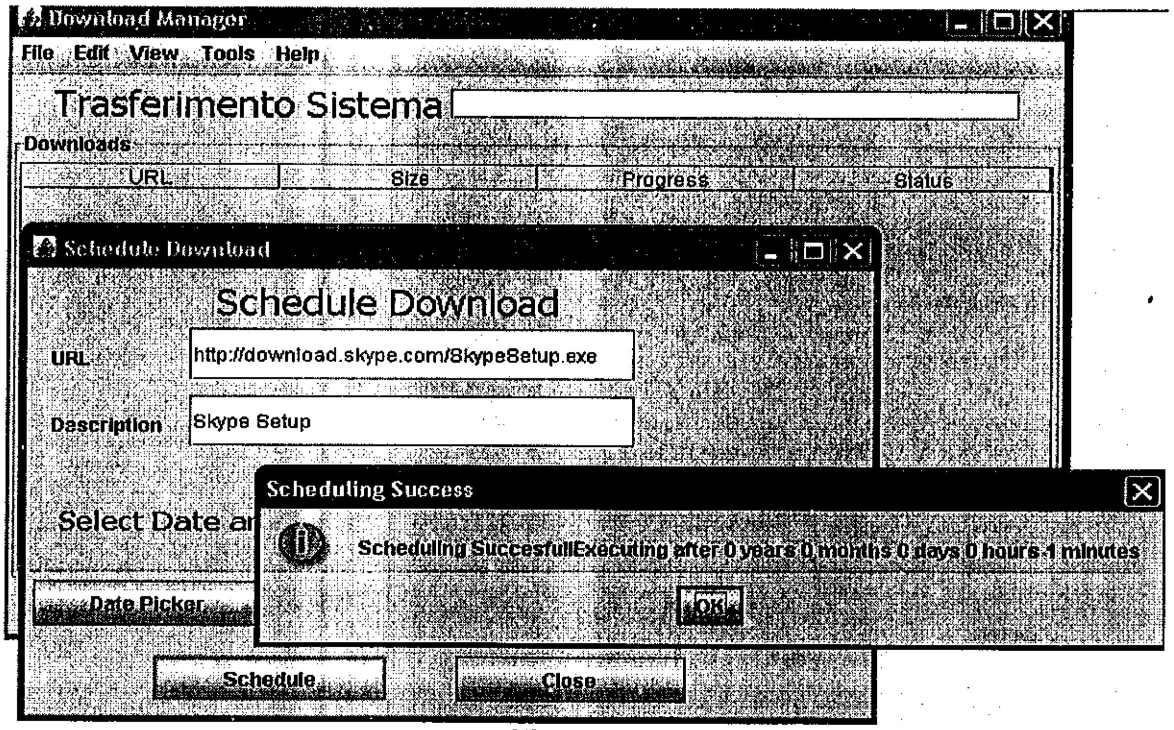
Select Date and Time To Launch Download

Date Picker 20 May 2007 Time Hours 15 Minutes 36

Schedule Close

Figure 4.25

Step 5 and 6


Download Manager

File Edit View Tools Help

Trasferimento Sistema

Downloads

URL	Size	Progress	Status

Schedule Download

Schedule Download

URL http://download.skype.com/SkypeSetup.exe

Description Skype Setup

Select Date ar

Date Picker

Schedule Close

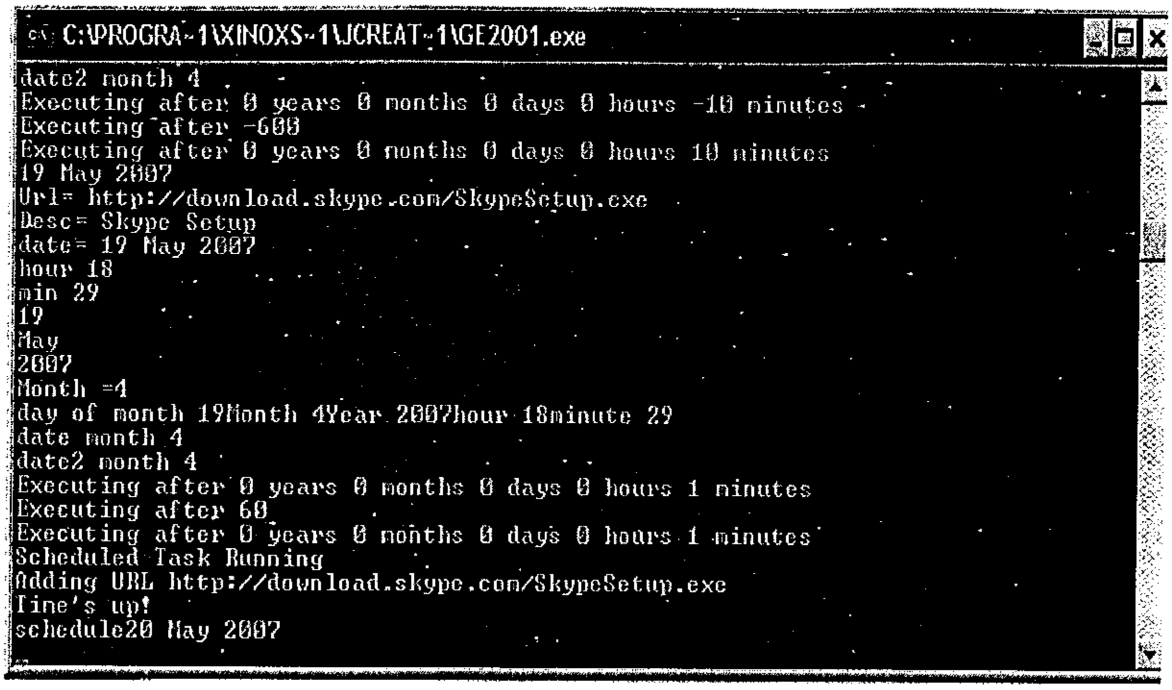
Scheduling Success

Scheduling SuccesfullExecuting after 0 years 0 months 0 days 0 hours 1 minutes

OK

Figure 4.26

Result – The download starts at the scheduled day and time as shown in the figure.



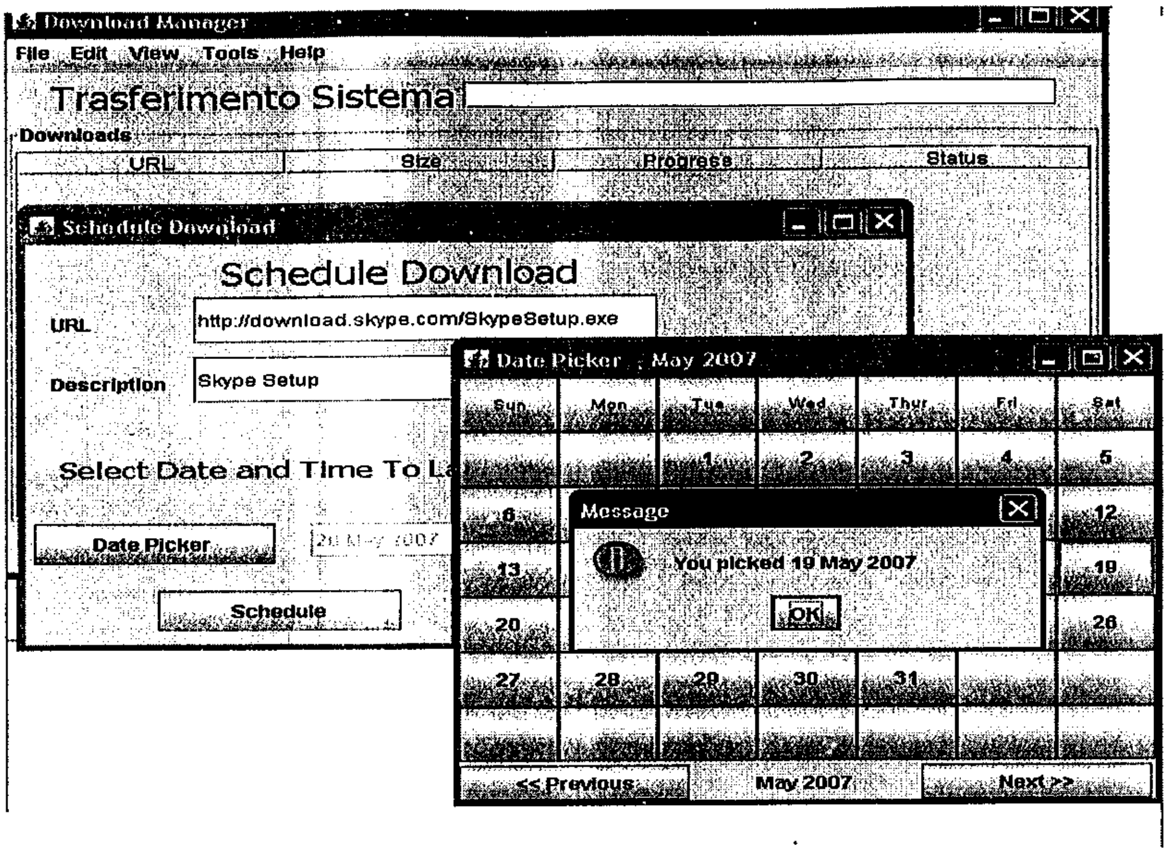
```
C:\PROGRA~1\XINOXS~1\JCREAT~1\GE2001.exe
date2 month 4
Executing after 0 years 0 months 0 days 0 hours -10 minutes
Executing after -600
Executing after 0 years 0 months 0 days 0 hours 10 minutes
19 May 2007
Url= http://download.skype.com/SkypeSetup.exe
Desc= Skype Setup
date= 19 May 2007
hour 18
min 29
19
May
2007
Month =4
day of month 19Month 4Year 2007hour 18minute 29
date month 4
date2 month 4
Executing after 0 years 0 months 0 days 0 hours 1 minutes
Executing after 60
Executing after 0 years 0 months 0 days 0 hours 1 minutes
Scheduled Task Running
Adding URL http://download.skype.com/SkypeSetup.exe
Time's up!
schedule20 May 2007
```

Figure 4.27

What happens when the user schedules the download after the current day and/or time has passed?

This is explained by the help of an example.

Suppose the current date is 20 may 2007 and the user schedules the download on 19 may 2007 at time 20:00 hours. This is shown in the figures below



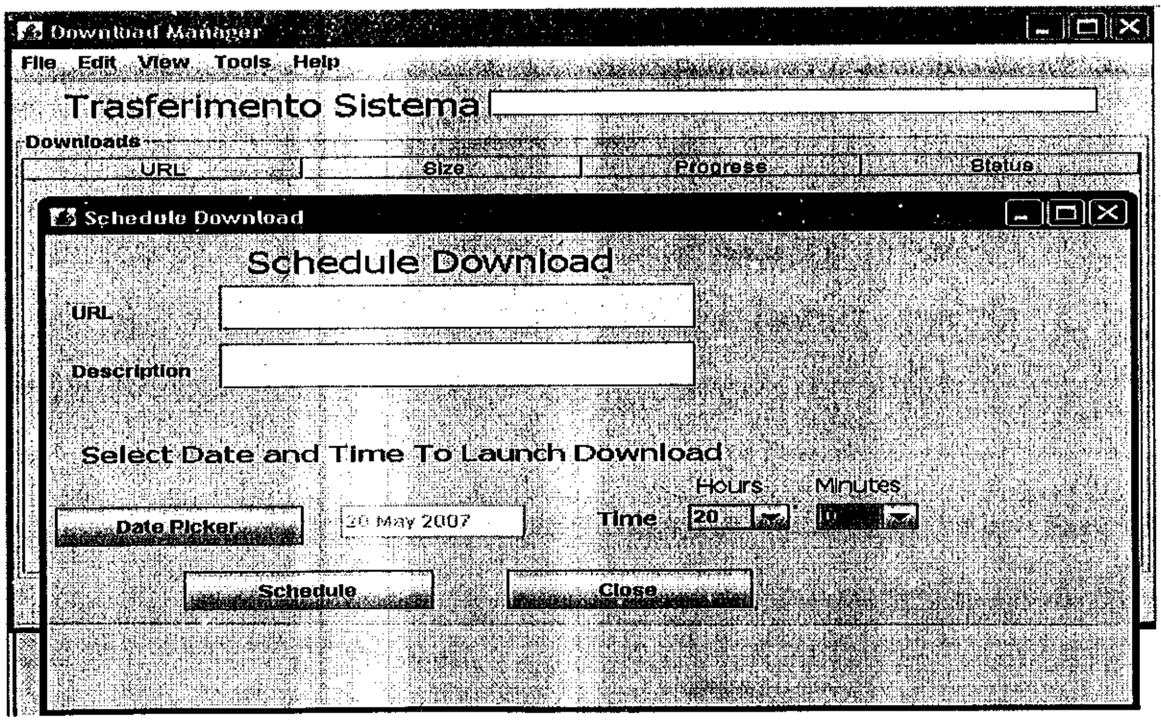
URL: http://download.skype.com/SkypeSetup.exe

Description: Skype Setup

20 May 2007

You picked 19 May 2007

Figure 4.28

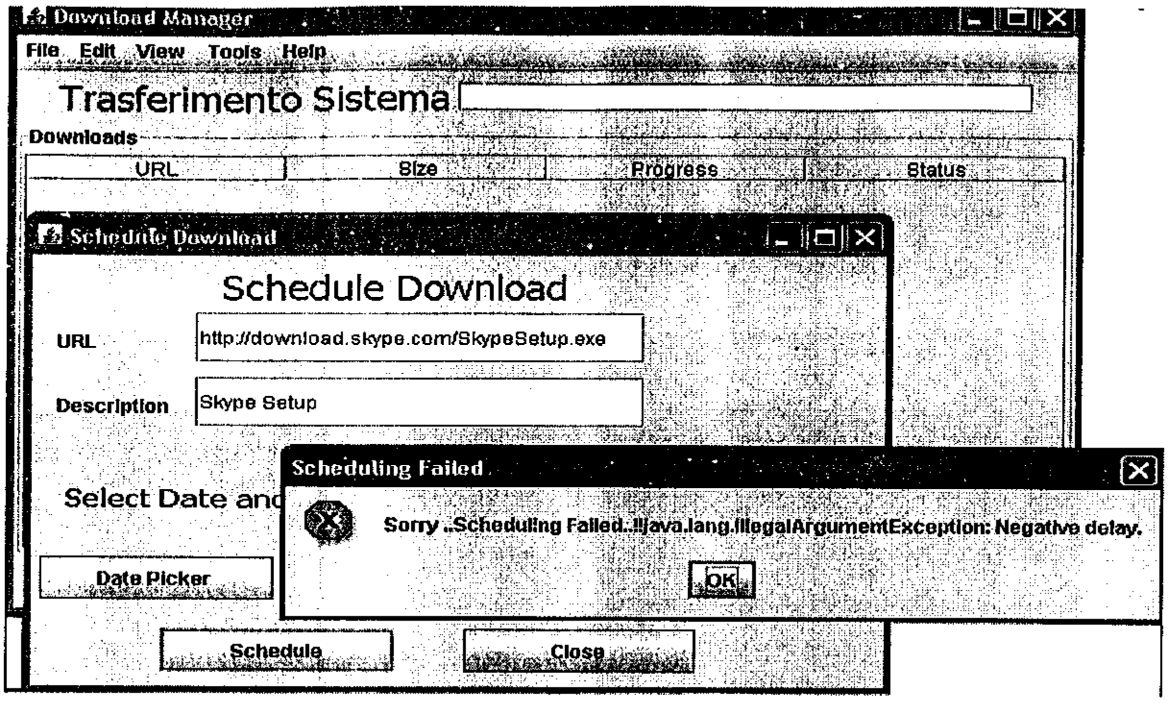


20 May 2007

Time: 20 : 0

Figure 4.29

Result – Sorry scheduling failed .Negative delay. As shown in the figure below.



http://download.skype.com/SkypeSetup.exe

Skype Setup

Sorry ..Scheduling Failed..!!java.lang.IllegalArgumentException: Negative delay.

Figure 4.30

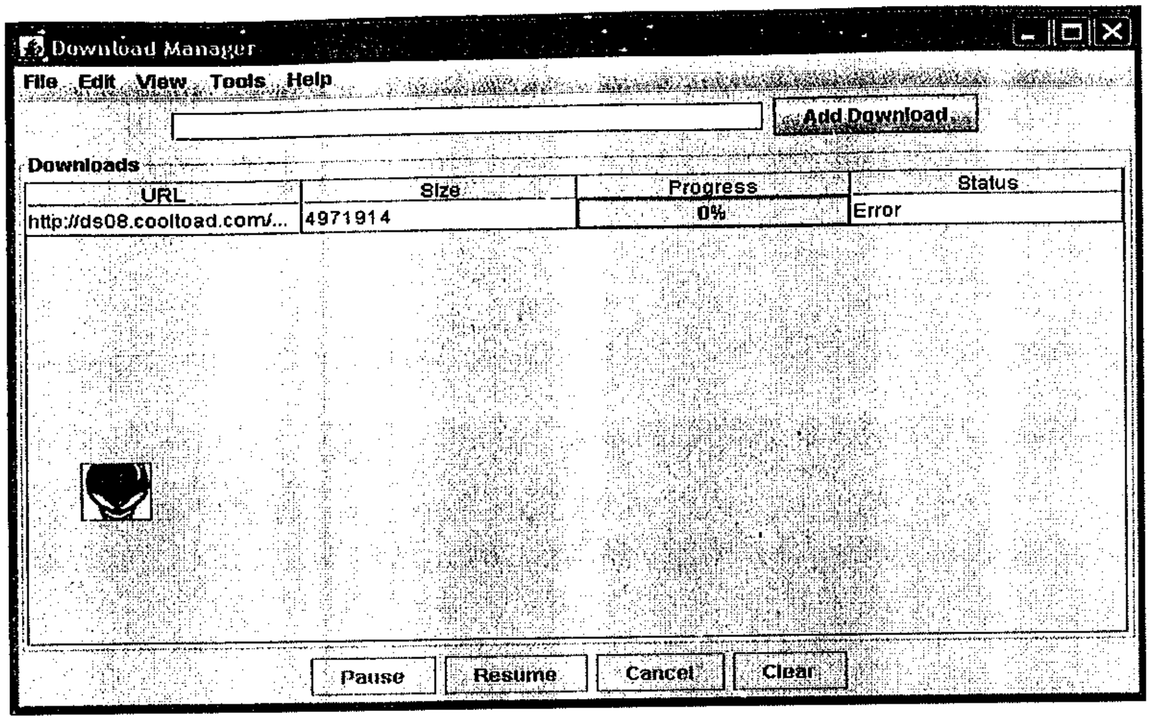
What happens to the downloads that were incomplete?

1. If due so some reason connection is lost then the file which are downloading stops and downloading status changes to "Error". There can be many cases when error comes such as

a) Internet connection gets lost when the downloading is in progress.

b) When the required file is not available in the server then the error 403 occurs.

This is shown in the figure below



URL	Size	Progress	Status
http://ds08.coolload.com/...	4971914	0%	Error

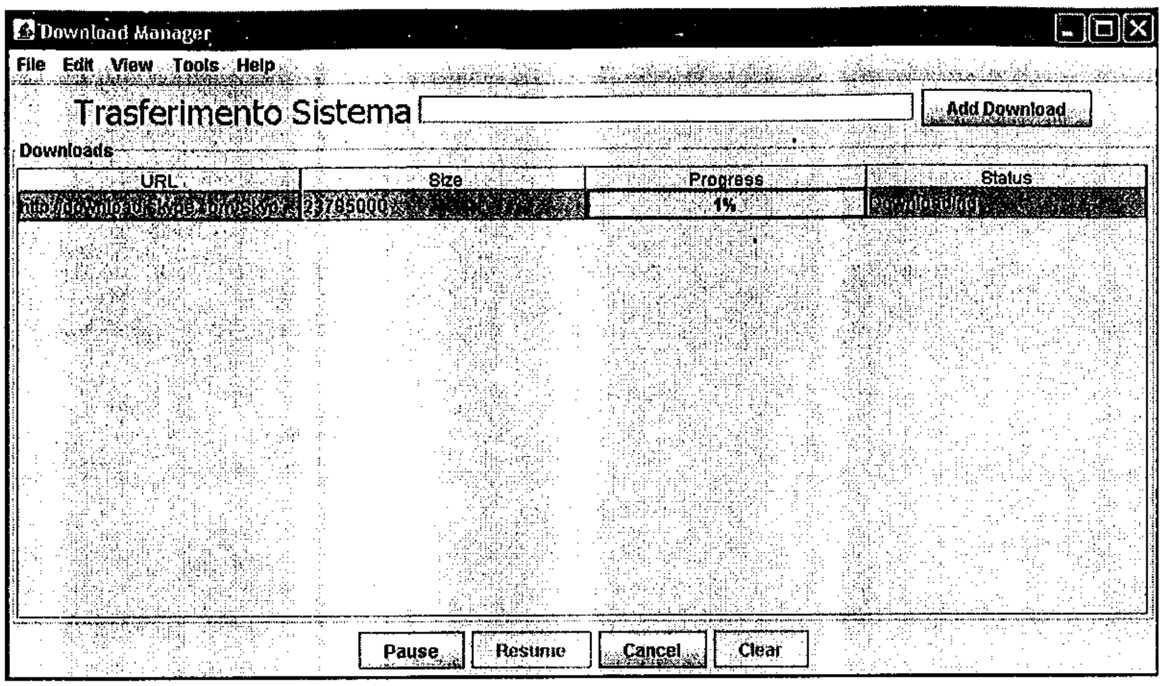
Figure 4.31

Now when the connection is lost in between downloading process the URL address is saved in the database so that the user can resume the download from the state where the connection was lost.

In order for the user to access the database the user can import list through file menu.

This can be explained by the help of the figures given below.

The file is in downloading progress.



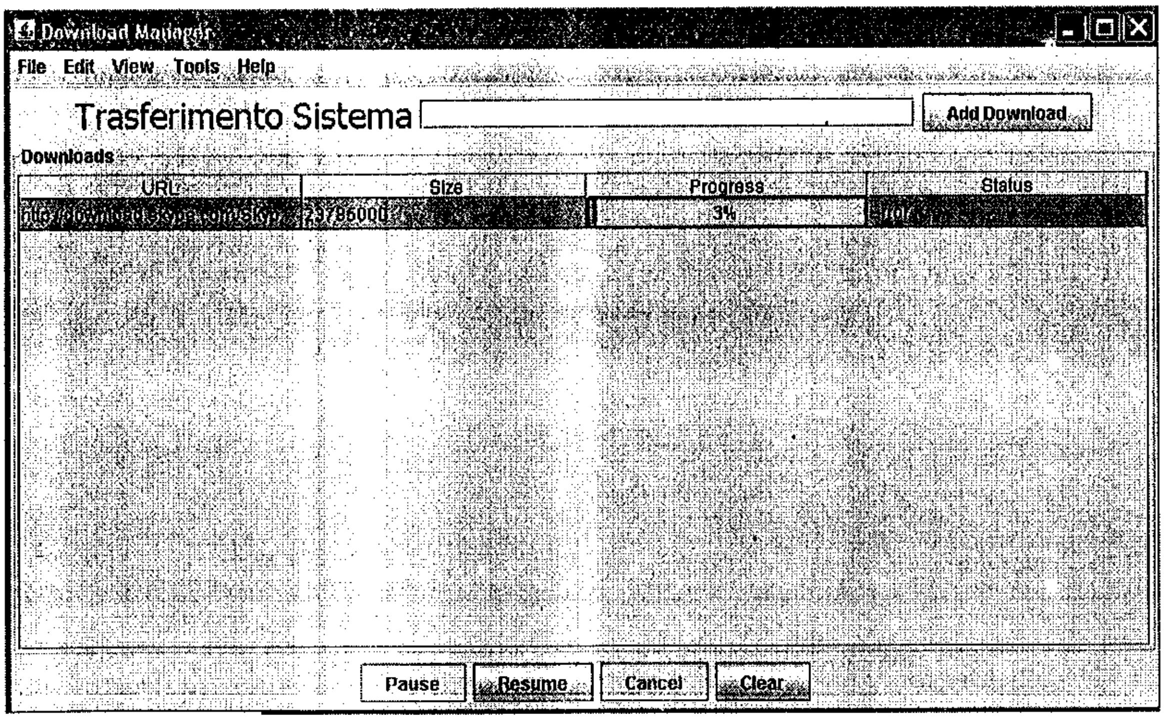
Trasferimento Sistema

Downloads

URL	Size	Progress	Status
http://download.skype.com/Skyp	23785000	1%	Downloading

Figure 4.32

The connection is lost.



Trasferimento Sistema

Downloads

URL	Size	Progress	Status
http://download.skype.com/Skyp	23786000	3%	Error

Figure 4.33

User access the database by importing list through file menu

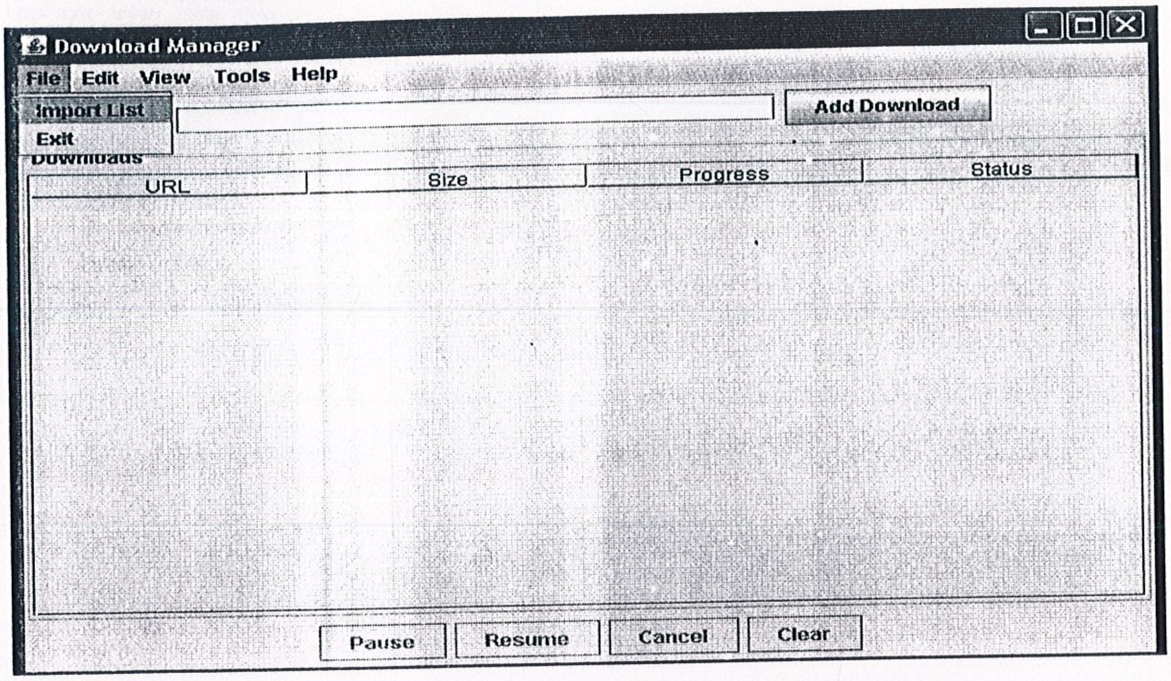
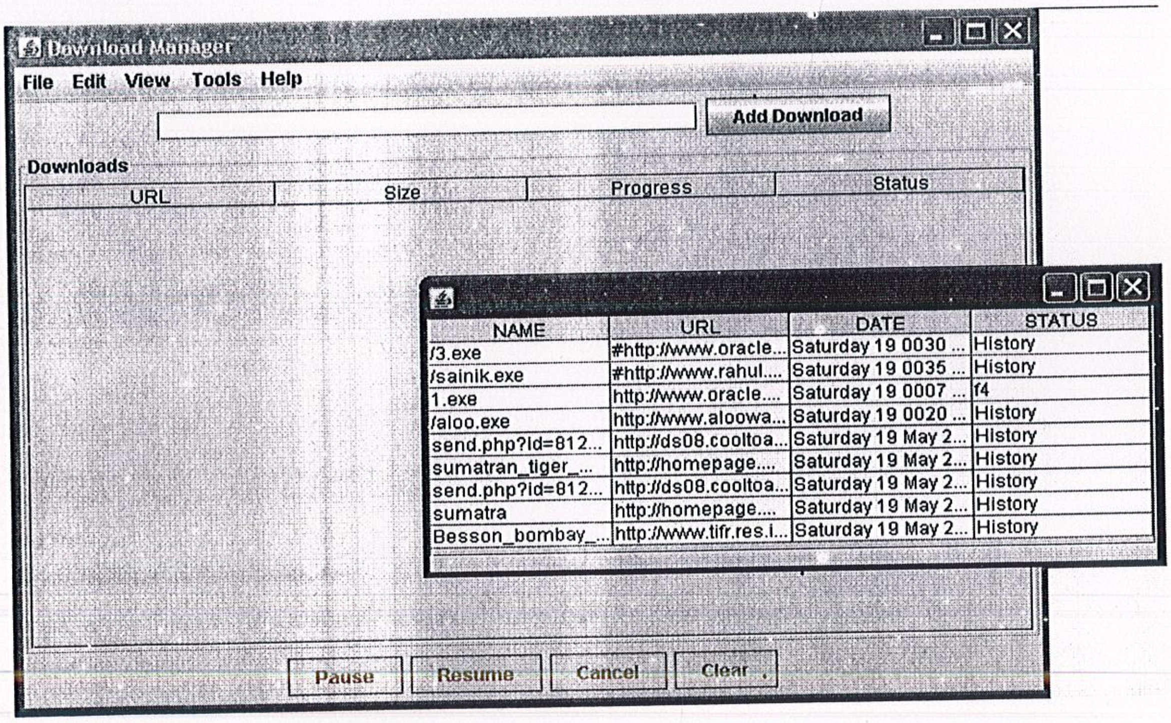


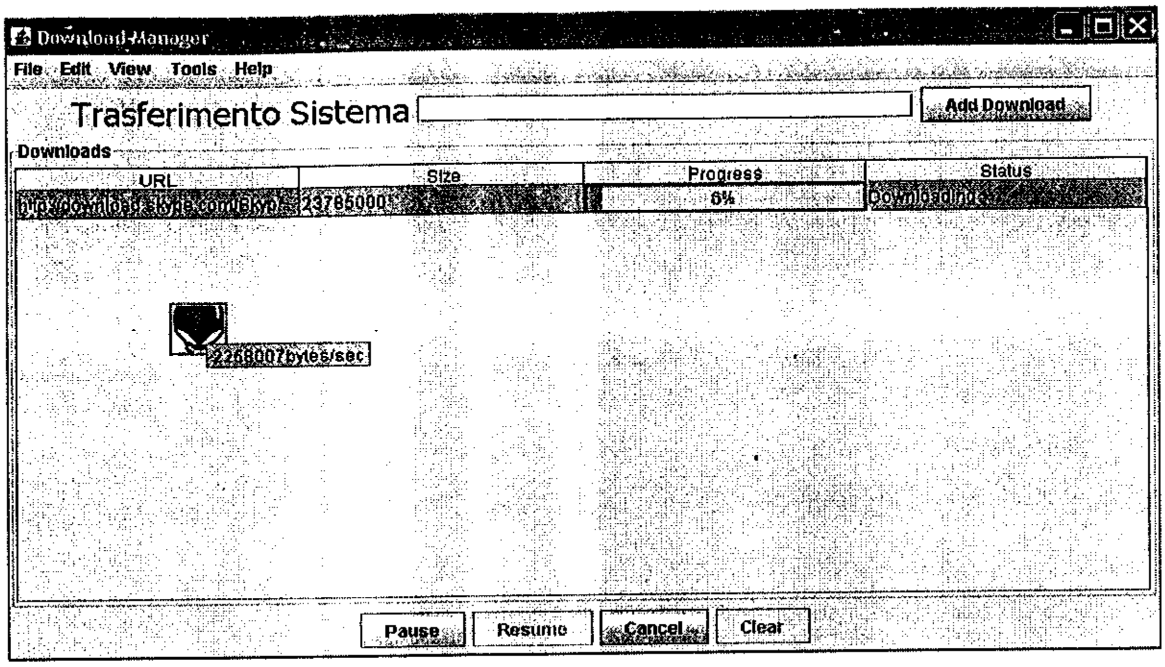
Figure 4.34



NAME	URL	DATE	STATUS
/3.exe	#http://www.oracle...	Saturday 19 0030 ...	History
/sainik.exe	#http://www.rahul....	Saturday 19 0035 ...	History
1.exe	http://www.oracle....	Saturday 19 0007 ...	f4
/aloo.exe	http://www.aloowa...	Saturday 19 0020 ...	History
send.php?id=812...	http://ds08.cooltoa...	Saturday 19 May 2...	History
sumatran_tiger_...	http://homepage....	Saturday 19 May 2...	History
send.php?id=812...	http://ds08.cooltoa...	Saturday 19 May 2...	History
sumatra	http://homepage....	Saturday 19 May 2...	History
Besson_bombay_...	http://www.tifr.res.i...	Saturday 19 May 2...	History

Figure 4.35

User copies the URL address from the database and file starts downloading from the state where error occurred.



URL	Size	Progress	Status
http://download.skype.com/Skyp/	23785000	6%	Downloading

Figure 4.36

4.4 ***Database***

In order to provide the facility of scheduling and resuming to user, we need to store some information of the downloading file such as URL, file-name, date of downloading, Status etc. So it is implemented by using databases

Field name	Data Type	Description
Name	Varchar	Gives the file name.
URL	Varchar (PK)	Provides source location.
Date	Date	Gives the date & time.
Status	Varchar	History

Table 3

CHAPTER 5 - TESTING

5.1 <u>*Questions addressed while testing*</u>

1. How is the functional validity tested?

- Tested by providing
- Good Input Values like correct URL Syntax
- Correct Server and proxy port
- Syntax of Server and proxy port

2. What Classes of Input make good test cases?

- URL address/Syntax
- Proxy Server and port
- Web address
- Only HTTP Server

3. Is the System particularly sensitive to some input values?

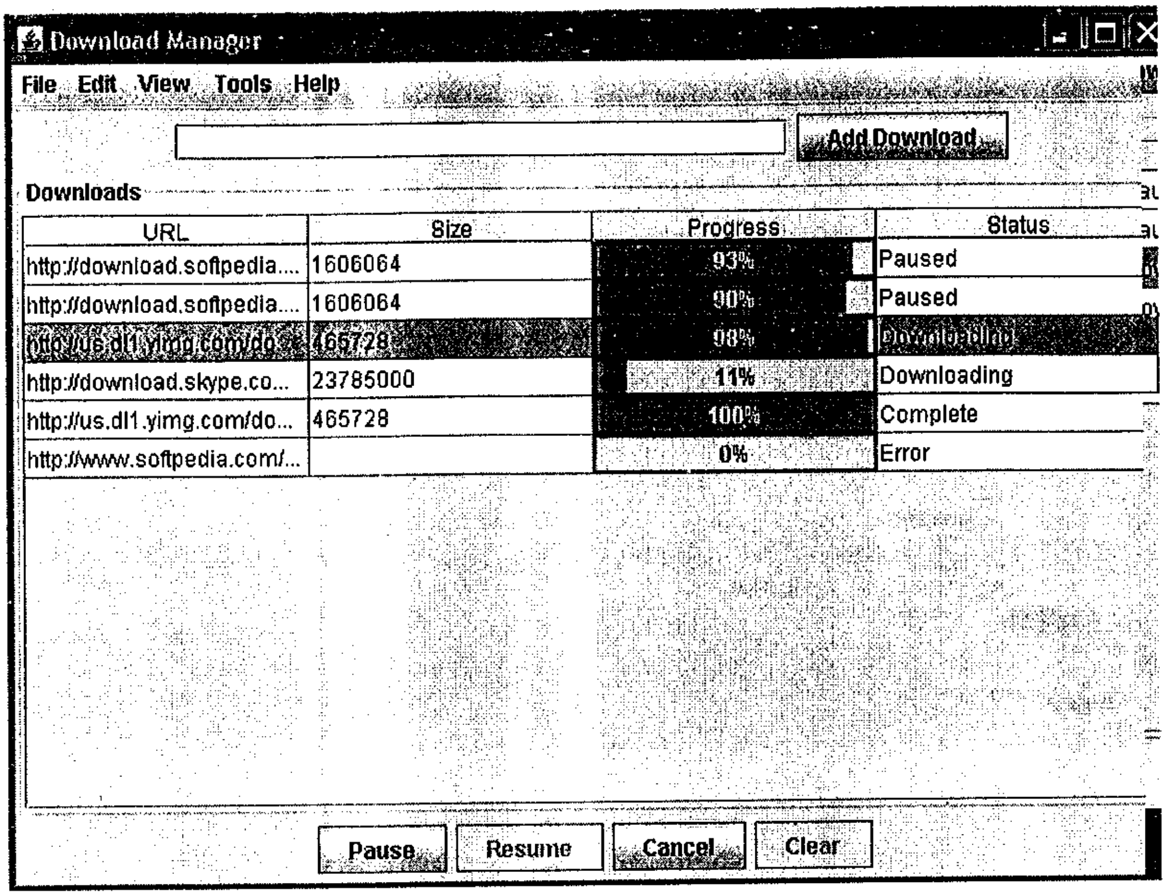
   No such Values exist.

4. What data volumes and data rates can be tolerated?

   <u>Data Volumes</u>

   Since it will depend on how many downloads are there running simultaneously. Every download will be regarded as an individual thread, hence there will be no buffer size overflow.

Here we can see that large amount of data volumes are received by the system .Simultaneously 5 download are being carried out.



URL	Size	Progress	Status
http://download.softpedia....	1606064	93%	Paused
http://download.softpedia....	1606064	90%	Paused
http://us.dl1.yimg.com/do...	465728	98%	Downloading
http://download.skype.co...	23785000	11%	Downloading
http://us.dl1.yimg.com/do...	465728	100%	Complete
http://www.softpedia.com/...		0%	Error

Figure 5.1

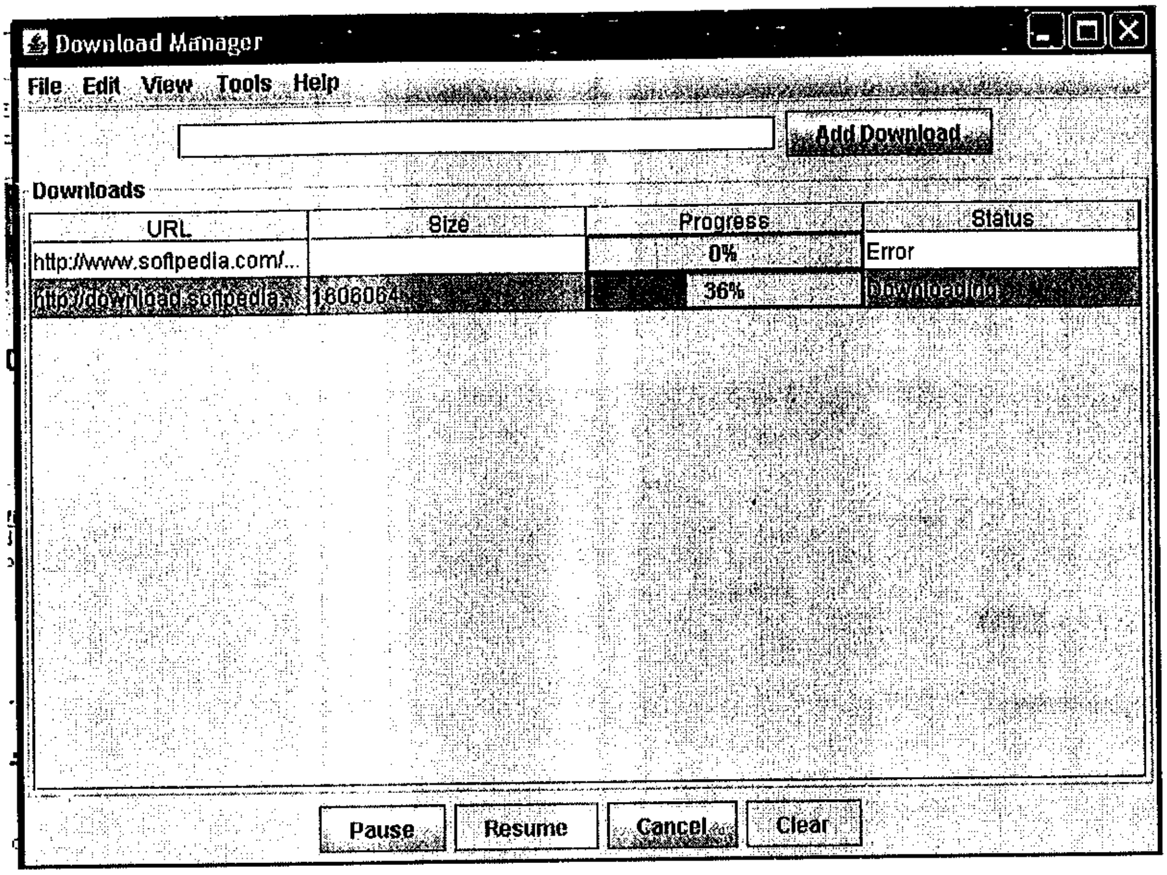
Observation – System working correctly.

Result - No buffer overflow.

Data Rates

Since downloading speed will depend upon the Internet speed so more the files to be downloaded lesser the speed will be of the files to be downloaded.

Download rate when download is being carried out:-



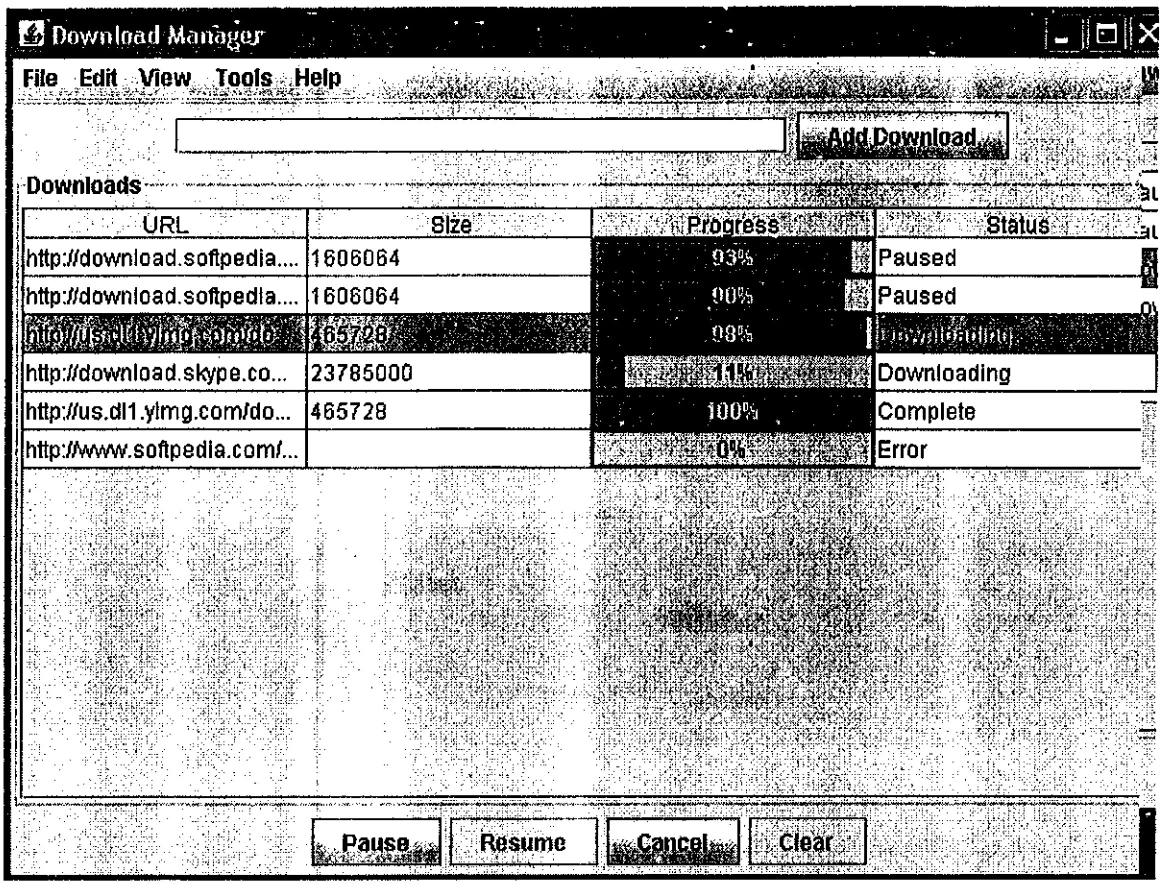
URL	Size	Progress	Status
http://www.softpedia.com/...		0%	Error
http://download.softpedia...	1606064	36%	Downloading

Figure 5.2

Observation:-

Avg Speed - 2175999 bytes/sec

Download rate when 5 Downloads are simultaneously being carried out.



URL	Size	Progress	Status
http://download.softpedia....	1606064	93%	Paused
http://download.softpedia....	1606064	90%	Paused
http://us.dl1.yimg.com/do...	465728	98%	Downloading
http://download.skype.co...	23785000	11%	Downloading
http://us.dl1.yimg.com/do...	465728	100%	Complete
http://www.softpedia.com/...		0%	Error

Figure 5.3

Observation:-

Avg Speed – 688592 bytes/sec

Result – Avg Speed drops as the number of downloads increase.

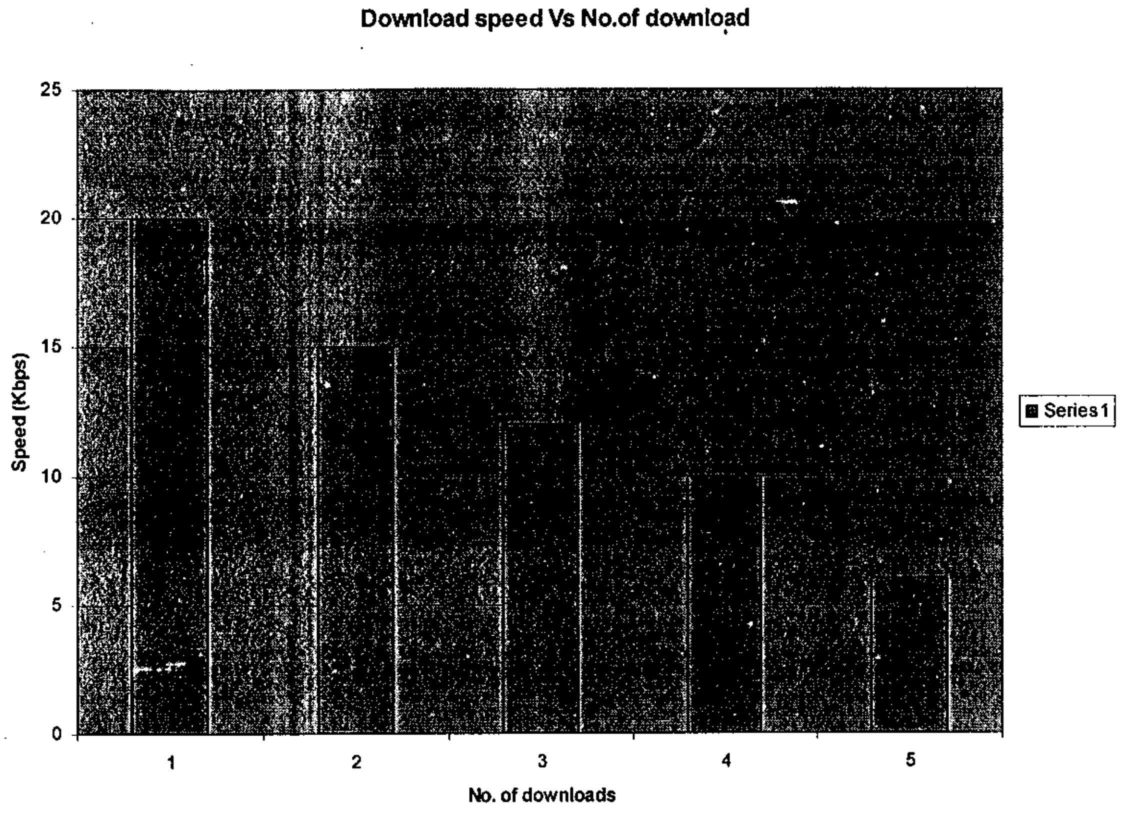


Figure 5.4

5.2 ***Black Box***

Download File

Requirement:-

The system should send the message about file downloading progress, status, size, and speed at which the file is getting downloaded.

Function:-

Download (Connection established, Correct URL)

Processing:-

Function downloads the file from the URL address provided by the user.

Output:-

Complete

Error

Invalid Syntax

Scheduling:

Requirement:-

The file should start downloading at a given scheduled time provided by the User.

Function: - Schedule (take date, take time, take URL)

Processing: - Downloading should start on a scheduled time.

Output: - Downloading

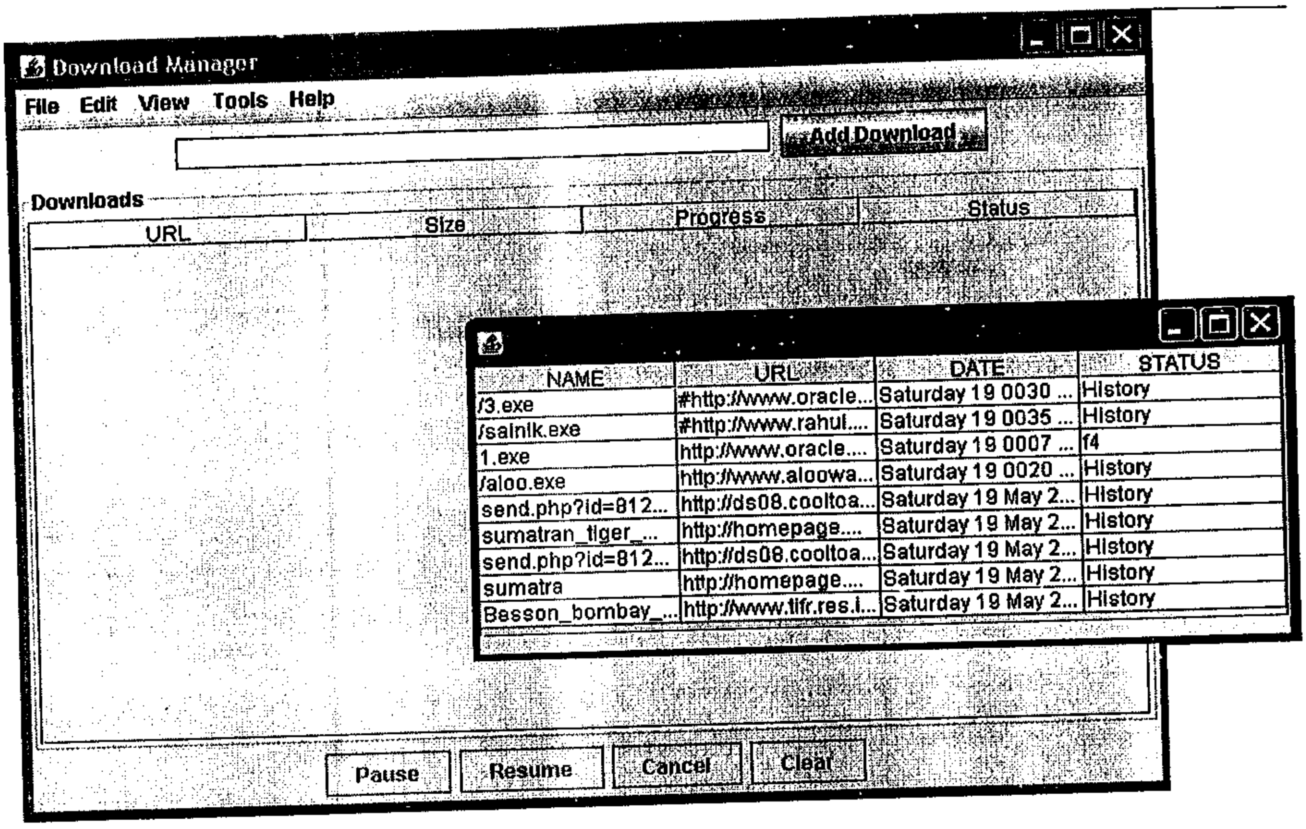
Error

Invalid date

Invalid time

In-Complete download

Requirement: -	System should retrieves the list of the in-complete downloads from the Database.
Function: -	Import (URL, Status, size)
Processing: -	Provides with the history of all downloads.
Output: -	Filename
	URL
	Date
	Time



NAME	URL	DATE	STATUS
/3.exe	#http://www.oracle...	Saturday 19 0030 ...	History
/sainik.exe	#http://www.rahul....	Saturday 19 0035 ...	History
1.exe	http://www.oracle....	Saturday 19 0007 ...	f4
/aloo.exe	http://www.aloowa...	Saturday 19 0020 ...	History
send.php?id=812...	http://ds08.cooltoa...	Saturday 19 May 2...	History
sumatran_tiger_...	http://homepage....	Saturday 19 May 2...	History
send.php?id=812...	http://ds08.cooltoa...	Saturday 19 May 2...	History
sumatra	http://homepage....	Saturday 19 May 2...	History
Besson_bombay_...	http://www.tifr.res.i...	Saturday 19 May 2...	History

Figure 5.5

Observation – Output is correct.

Result - Functionality tested

FUTURE SCOPE OF WORK

Download Manager Helps in managing the downloads. Management of multiple downloads and scheduling of downloads have been achieved. The future scopes with download manager are:

1. DOWNLOAD ACCELERATOR Best utilization of the bandwidth. Files can be split into sections and download each split simultaneously. Now opening multiple connections to each file will help reduce download time.

2. PROTOCOL Adding support for various other protocols can be additional feature.

3. BROWSER Download Manager can be embedded with different browsers to start downloading directly.

CONCLUSION

The software performs the basic tasks such as downloading, pause a download, resume a paused download, cancel a download, scheduling successfully but there is still scope for improvement in terms of additional features, various complexities etc. The scope and scale of this application can be extended by future developers. It can be used by both professional and home users according to their own wish.

BIBLIOGRAPHY

Books

- Schildt, Herbert .The Complete Reference JAVA2. McGraw Hill Publishers, 2002.
- Ramakrishnan, Raghu & Gehrke, Johannes .Database Management System
- O Neil, Joseph. Teach Yourself Java 2.Tech Media Press,2004.
- Hortan, Ivor. Beginning Java 2 (JDK 5 Edition).Willey Publishers India, 2005
- Steven Holzner et al .Black Book Java 2 (JDK 5 Edition) Programming, 2006
- Tanenbaum, Andrew S. Computer Networks, 3rd Edition, 2004.

Internet Sources

- http://www.wikipedia.org/
- http://java.sun.com/docs/books/tutorial/
- http://www.wrox.com/dynamic/books/download.aspx
- http://developer.java.sun.com/developer/support/
- http://java.sun.com/j2se/1.5/docs/index.html