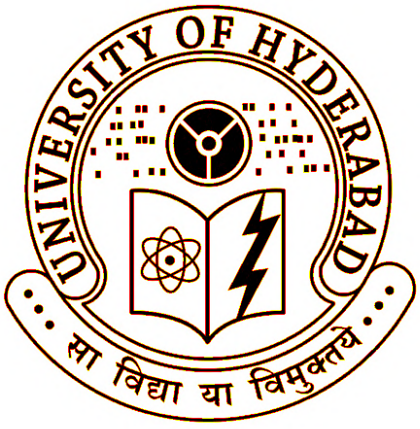
Next Hop Finder

G. Hanish
P Navaneeth Kumar
K.Venkateswara Rao

Advisor: Ms. Anupama Potluri

Date: April 5, 2012



Dept. of Computer Science and Information Sciences
University of Hyderabad

Contents

Chapter 1

System Requirements Specification

1.1 INTRODUCTION

1.1.1 Product Overview

In this project we build a IP forwarding-table lookup.When we transmit a packet through Internet it may travel through various intermediate nodes (ex: routers,hosts). Each router and each host keeps a routing table which tells the System how to process an outgoing packet.
MAIN COLUMNS IN ROUTING TABLE:

1. Destination address: where is the IP datagram going to?
2. Next hop: how to send the IP datagram?
3. Interface: what is the output port?

In this project we will determine "Next hop IP address" given a "destination IP address" by searching in the table. We also provide facility of modifying a table, but some of the conditions such as same prefix with different subnet mask are not handling in this project.

The data structure we are using here to store the IP address and corresponding next hop is binary trie and it uses longest prefix matching algorithm to find the matching IP in routing table.

1.2 SPECIFIC REQUIREMENTS

1.2.1 External Interface Requirements

User Interfaces

The product must be usable from the Graphical User Interface(GUI), particularly under operating systems that support GUI.

Hardware Interfaces

The project should be implemented in a hardware-independent fashion and should not rely on any particular hardware interfaces.

Software Interfaces

The software interfaces required for this project is java development kit.
Name:Java development kit.
version:1.7.

Communications Protocols

No communications protocols are mandated.

1.2.2 Software Product Features

- Provides next hop address and the Longest matched prefix for a given IP address in Search functionality.
- Parse functionality, fills the table with the file contens.
- Verifies the IP address given by the user.
- User can update the routing table tuple.
- User can delete a tuple in the table.
- Provides help for a new user.

1.2.3 Software System Attributes

Reliability

The product should always give the correct next hop IP address based on the data provided. This functionality should always be reliable.

Maintainability

The code of the product will be well commented. There will be enough modularity to accomadate future changes.

Portability

Usage of Machine independent programming style, can increase portability of the system.

Performance

The product supports single user with single temrminal in use. The product can handle large data to form the required data structure.

1.2.4 Database Requirements

The product uses files to fill its data strucutre. These files are used once to form the data structure. All the files are opened in Read-only mode. The format of the files being used is IPAddress SubnetMaskBits NextHopRouterIPAddress.

1.3 ADDITIONAL MATERIAL

None.

Chapter 2

High Level Design

2.1 Modules in the system

1. **GUI Module**:
   This module provides interface for all the functionalities of the product like parse, search, delete and update by generating appropriate input fields. This module also provides results of various operations performed.
2. **Parser Module**:
   This module parses the input file and send its contents to the Forwarding Table module.
3. **Forwarding Table Module**:
   This module stores the data in binary trie and searches the input IP address using Longest Prefix Matching algorithm. It also provides the delete and update functionality on the table.

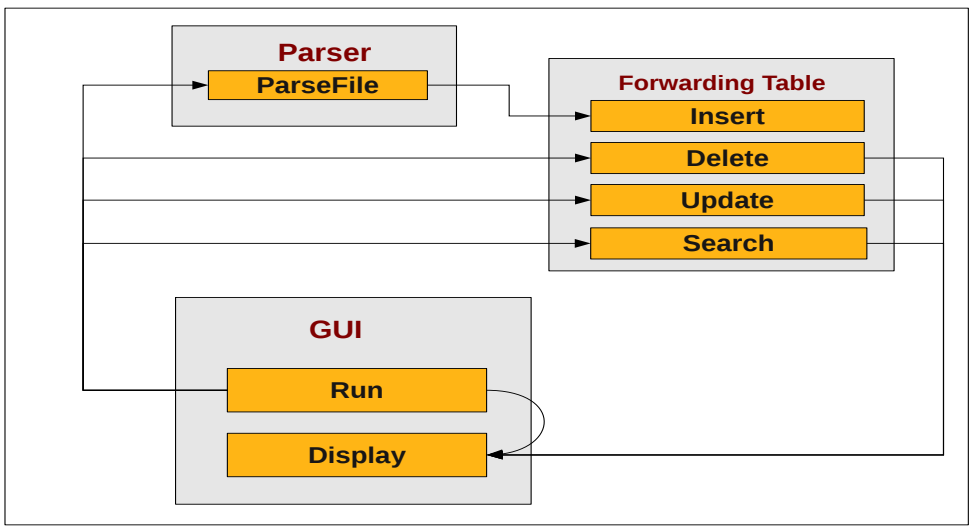


Fig: Interfaces between modules

2.2 DataFlow Diagram

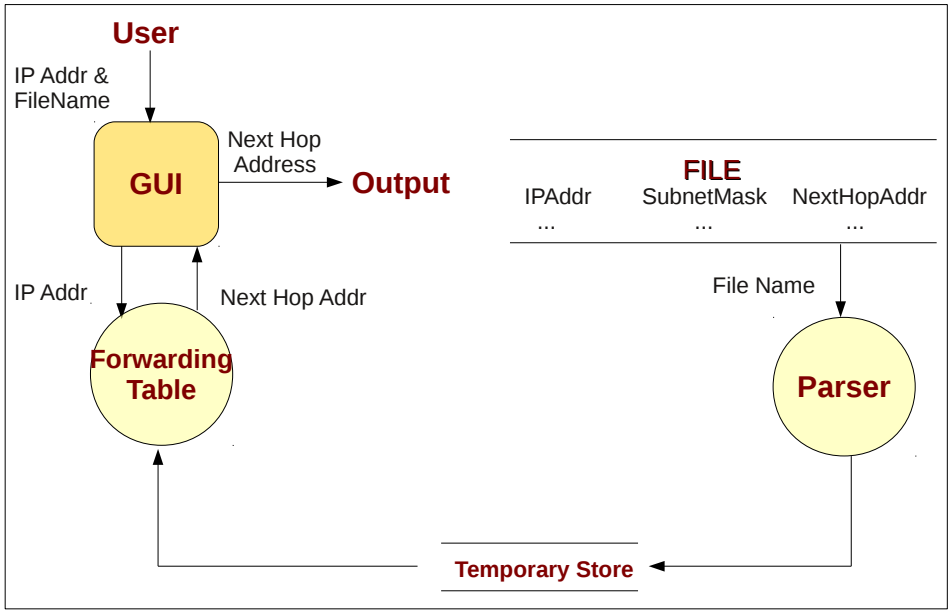


Fig: DataFlow Diagram

2.3 API Specification

2.3.1 Module of the architecture

As mentioned above there are three modules in this system.

Parser Module

- **Functionality**:
  The goal of this module is to parse a text file, containing information about IP addresses, subnet and corresponding next hop addresses.Input of this module is text file and the output of this module will be given to the Forwarding Table module.
- **Interface Description**

  – **parse()**

  Purpose : To parse the given input text file containing prefix and next hop information.
  Input Parameter : FileName.
  Output Parameter : None.

Return Value : Returns OK on success; ERRNO on error,ERRNO set to various negetive integers for different errors like FILENOTFOUND: -1, EMPTYFILE:-2, INCOMPLETEINFORMATION:-3 and so on.
Calledby : GUI module.
Calls : Insert API in the Forwarding Table module.

GUI Module

- **Functionality**:.
  The goal of this module is to read the IP address and text file name from the user and displays next hop address in a convenient format. It also provides interface for update and delete functionality.
- **Interface Description**

  – **Filesubmit actionListener**

  Purpose : To invoke the parse module once the file is submited.
  Input Parameter : Action event.
  Output Parameters : None.
  Return values : None.

  – **IPsubmit actionListener**

  Purpose : To invoke the search function in the Forwarding Table module once the IP is submited.
  Input Parameter : Action event.
  Output Parameters : None.
  Return values : None.

  – **update actionListener**

  Purpose : To invoke the update function in the Forwarding Table module once the required IP address and next hop IP address are submited.
  Input Parameter : Action event.
  Output Parameters : None.
  Return values : None.

  – **Delete actionListener**

  Purpose : To invoke the delete function in the Forwarding Table module once the IP address is submited.
  Input Parameter : Action event.
  Output Parameters : None.
  Return values : None.

– **Run**

Purpose : To invoke the Graphical interface once the program starts.
Input Parameter : None.
Output Parameters : None.
Return values : None.

– **DisplayTable()**

Purpose : To display the IP address, Next hop IP address from forwarding table.
Input Prameters : List containing the IP address and Next hop Information.
Output Parameters : None.
Return Value : None.
Calledby : Both GUI and Forwarding Table modules.

Forwarding Table Module

- **Functionality**:.
  The functionality of the module includes insert, search, update and delete on the forwarding table.
- **Interface Description**

  – **search()**

  Purpose : To find the longest prefix match IP address for a given IP address.
  Input Parameters : Ip address that we are tring to find out the next hop address.
  Output Parametes : Longest prefix match of IP Address found in the binary trie.
  Return Value : Next Hop information on successful search and ERRNO incase of unknown IP.
  Calledby : GUI module.

  – **Insert()**

  Purpose : To store IP address and corresponding next hop information in the binary trie.
  Input Prameters : Ip address and corresponding next hop information.
  Output Parameters : None.
  Return Value : Returns OK on successful insetion and ERRNO on error.
  Calledby : Parser module.

– **Update()**

Purpose :Updating the forwarding table, this can be done in two ways,.
1. updating next hop information.
2. Inserting new IP address and next hop information into the forwarding table.

Input Prameters : Ip address and corresponding next hop information.
Output Parameters : None.
Return Value : Returns OK on successful updation, NEWINSERTION incase of new insertion and ERRNO on error.
Calledby : GUI module.

– **Delete()**

Purpose : To delete some IP address information from forwarding table.
Input Parameters : Ip address that we want to delete.
Output Parameters : None.
Return Value : Returns OK on successful deletion and ERRNO on error.
Calledby : GUI module.

Chapter 3

Detailed Design

3.1 Parser Module

This module is invoked by the GUI module to read the data from the input file given by the user and pass it to the Forwarding table module to construct a prefix table for the given input.

3.1.1 Interface Data Structures

1. TableRecord

Table Record :

The TableRecord data structure contains three data types.

- IP address – unsigned integer
- Subnet mask –unsigned short integer
- NextHop IP address – unsigned integer

This data structure is exported when the parse function passes it to Insert function of the Forward table module.

3.1.2 Internal Data Structures

NONE

3.1.3 Interface Functions

– int Parse(String filepath)

Description : This function opens the given file, reads the data and parse it into the required format. Call the calculateSubnet function take its return value, along with this the next hop information which was read from file are passed to the insert in the forwarding table module. The data read from the file line wise, each line in input file should be of this format "IPAddress SubnetMask NextHopAddress".

Input Parameters : The path of the file given by the user from the GUI module, complete path should be specified.

Output Parameters : NONE

Return Values : It returns the Number of lines successfully inserted in to the trie on successful completion of parsing and ERRNO assigned to a spcefic error when the parse is not successful.

Pseudo Code :

```
PARSE(FilePath)
1 fp=openFile(Filepath,"r")
2 if fp = NIL
3 return FileNotFoundError
4 Nline=0
5 sucess=0
6 InsertErr=0
7 while str=fp.readLine!=EOF
8 Nline++
9 // TR - Reference to the table row
10 TR.IPAddr=nextToken(str)
11 TR.Prefix=nextToken(str)
12 TR.NextHop=nextToken(str)
13 if TR.IPAddr >=0
14 rval=INSERT(TR)
15 if rval == OK
16 sucess++
17 else
18 InsertErr++
19 if InsertErr > 0
20 print InsertionError
21 if Nline == 0
22 return EmptyFileError
23 else
24 return success
```

3.2 GUI Module

The GUI module is useful for taking input from the user and giving results to the user. This module provides interface for entering the input. At first it asks file name as input from the user

and passes it to the parse function in parse module. Then it displays a screen with various buttons specifying the functionalities of the system such as search, update and delete. Upon on selection of functionality by the user the corresponding screen is presented to the user with various input text fields. The following diagram shows the finite state machine model for GUI.

Finite State Machine – GUI :

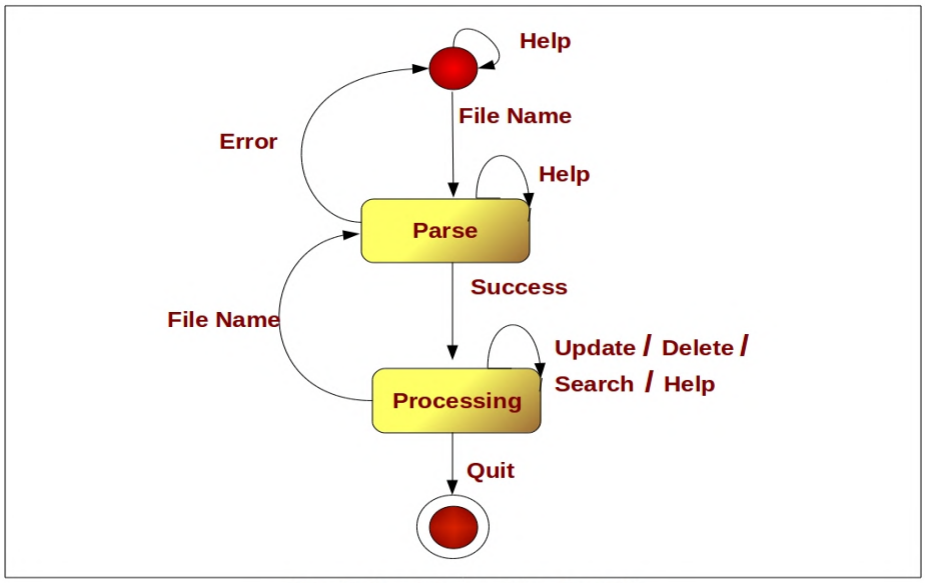


Fig: Finite state machine for GUI

Prototype – GUI :

Prototype for Help window :

3.2.1 Interface Data Structures

1. TableRecord
2. String
3. Unsigned Integer

Table Record : The TableRecord data structure contains three data types.

- IP address – unsigned integer
- Subnet mask –unsigned short integer
- NextHop IP address – unsigned integer

This data structure is exported when the GUI calls Search, Update and Delete functions of the Forward table module.

String : String is used to represent the file path and it is exported when the GUI calls Parse function of the Parser module.

Unsigned Integer : Unsigned Integer is used to represent IP address and it is exported when the GUI calls search function in the GUI module

3.2.2 Internal Data Structures

1. ArrayList

ArrayList : ArrayList is used to represent all the records in the table and it is exported when run method calls DisplayTable function in this module.

3.2.3 Interface Functions

– void DisplayTable(ArrayList list)

Description : This function is called to display the look up table on the GUI for user convenience to search and is updated after each successful and exact delete, update operation on the table.

Input Parameters : List containing Table Record objects.

Output Parameters : NONE

Return Values : NONE.

Pseudo Code :

```
DisplayTable(ArrayList list)
1 While list has more records
2 rec = list.nextRecord
3 IPAddr = dottedDecimal(rec.IPAddr)
4 Subnet = rec.Subnet
5 NextHop= dottedDecimal(rec.NextHop)
6 Display on screen
```

3.2.4 Internal Functions

– **void Run()**

Description : The purpose of this function is to invoke GUI and call the appropriate functions based on the user choice.

Input Parameters : NONE

Output Parameters : NONE

Return Values : NONE.

Procedure :

```
Run()
1 Create the components required for GUI
2 Add the components to GUI
3 Call appropriate function based on component clicked
```

3.2.5 Action Event Functions

– **void ParseListener(ActionEvent ae)**

Description : This function is invoked if the parse button is pressed in the GUI, we must parse the file before performing any action. It calls the PARSE function in the Parser module and handles the return values.

Input Parameters : ActionEvent

Output Parameters : NONE

Return Values : NONE.

Pseudo Code :

```
ParseActionListener(ActionEvent ae)
1 //TR - Reference to the Table Record
2 FilePath = ae.READ(FilePath)
3 rval = PARSE(FilePath)
4 if rval > 0
5 print rval 'number of lines inserted'
6 else if rval == FileNotFoundError
7 print 'File Not Found Error'
8 else if rval == EmptyFileError
9 print 'Empty File Error'
```

Prototype for Parse window :

– **void SearchListener(ActionEvent ae)**

Description : This function is invoked if the search button is pressed in the GUI. It calls the search function in the Forwarding table module and handle its return values.

Input Parameters : ActionEvent

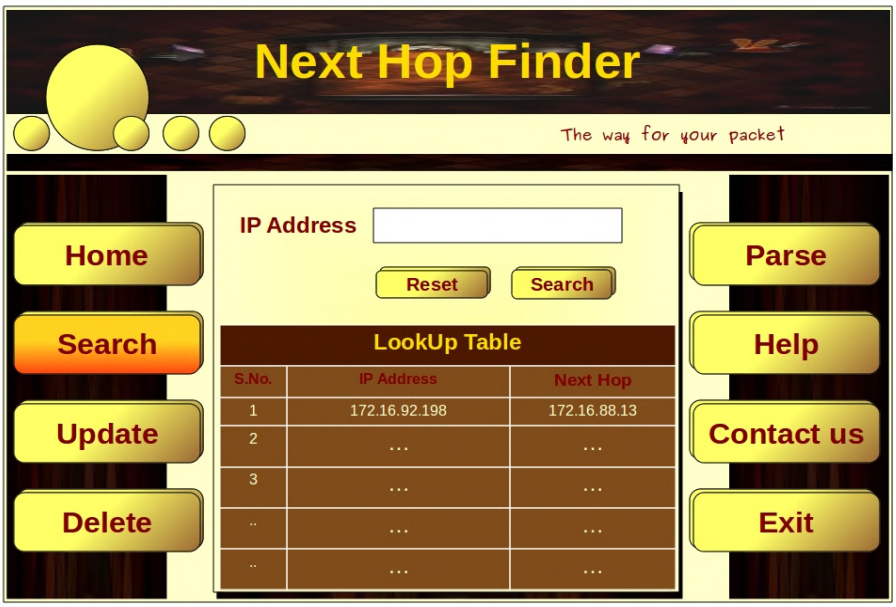
Output Parameters : NONE

Return Values : NONE.

Pseudo Code :

```
SearchActionListener(ActionEvent ae)
1 //TR - Reference to the Table Record
2 TR.IPAddr = ae.READ(IPAddr)
3 rval = SEARCH(TR)
4 if rval >= 0
5 print NextHopAddr
6 else
7 print 'IP Not Found'
```

Prototype for Search window :



– **void UpdateListener(ActionEvent ae)**

Description : This function is invoked if the update button is pressed in the GUI. It calls the update function in the Forwarding table module and handle its return values.

Input Parameters : ActionEvent

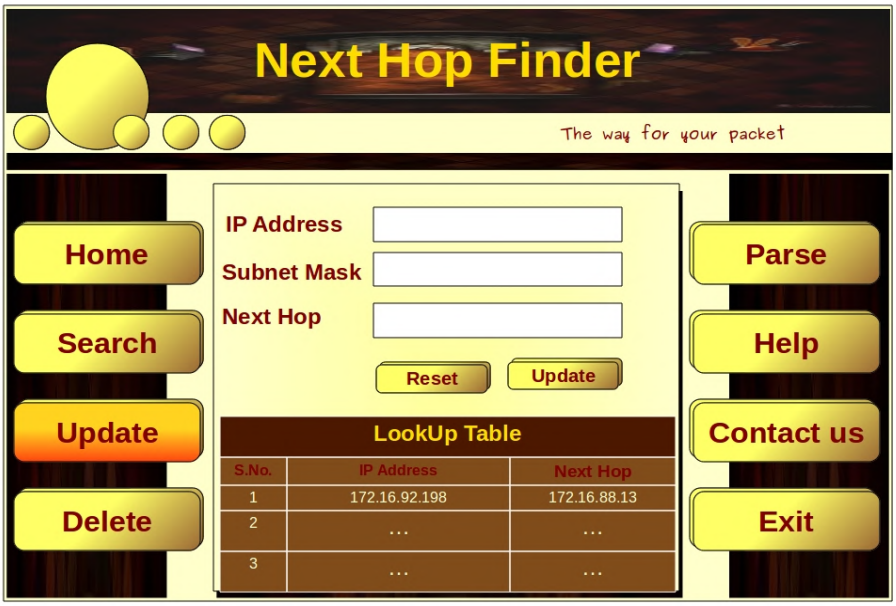
Output Parameters : NONE

Return Values : NONE.

Pseudo Code :

```
 UpdateActionListener(ActionEvent ae)
 1 //TR - Reference to the Table Record
 2 TR.IPAddr = ae.READ(IPAddr)
 3 TR.Subnet = ae.READ(Subnet)
 4 TR.NextHop = ae.READ(NextHop)
 5 rval = UPDATE(TR)
 6 if rval == OK
 7 print 'Sucessful Updation'
 8 else if rval == NEWINSERTION
 9 print 'Sucessful Insertion'
10 else
11 print 'Updation Error'
```

Prototype for Update window :



– **void deletelistener(ActionEvent ae)**

Description : This function is invoked if the delete button is pressed in the GUI. It calls the delete function in the Forwarding table module and handle its return values.

Input Parameters : ActionEvent

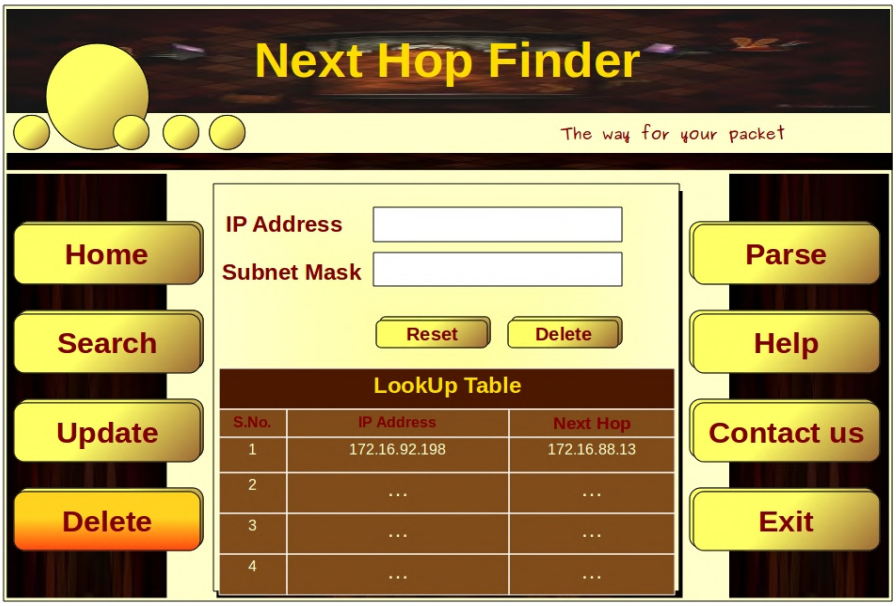
Output Parameters : NONE

Return Values : NONE.

Pseudo Code :

```
DeleteActionListener(ActionEvent ae)
1 //TR - Reference to the Table Record
2 TR.IPAddr = ae.READ(IPAddr)
3 TR.Subnet = ae.READ(Subnet)
4 TR.NextHop = ae.READ(NextHop)
5 rval = DELETE(TR)
6 if rval == OK
7 print 'Sucessful Deletion'
8 else
9 print 'IP Addr Not Found '
```

Prototype for Delete window :



3.3 Forwarding Table Module

This module creates and represents the prefix table as a binary trie in which the nodes store the next hop information and the path from root node to the any node is a prefix of binary IP address. The functionalities include, searching for a next hop information given IP address, Inserting a new IP address and next hop information into the trie, Updating the next hop information in the trie and deleting the information from the trie.

3.3.1 Interface Data Structures

1. TableRecord
2. PrefixRecordMatch

Table Record :

The TableRecord data structure contains three data types.

- IP address – unsigned integer
- Subnet mask –unsigned short integer
- NextHop IP address – unsigned integer.

This data structure is exported when update and delete functions calls the Calc_Subnet function of the parser module.

PrefixRecordMatch :

This data structure is a string to represent the longest prefix match and it is exported when the GUI calls search function in this module.

3.3.2 Internal Data Structures

1. Binary trie

Binary Trie :

Binary trie is used to store the IP addresses and next hop IP address. The next hop IP address is stored in the nodes, left and right links of the nodes represent the binary '0' and '1' respectively and the path from the root node to any internal node represents the prefix for binary form of IP Address.

3.3.3 Interface Functions

– int INSERT(TableRecord TR)

Description : This function is used to insert the IP address and the corresponding next hop IP address in to the binary trie.

Input Parameters : The input parameters of this function is the TableRecord data structure containing IP address,next hop IP address and subnetmask.

Output Parameters : NONE

Return Values : It returns OK on sucessful insertion of data and ERRNO if the insertion fails.

Pseudo Code :

```
INSERT(TR)
//TR - Reference to Table Record
1 if root == NULL
2 root=InitTrie()
3 T=root
4 IPAddr = Calc_Subnet(TR)
5 while IPAddr != 0
6 if MSB(IPAddr) == 0
7 if T.left == NULL
8 T.left=InitTrie()
```

```
 9 T=T.left
 10 else
 11 if T.right == NULL
 12 T.right=InitTrie()
 13 T=T.right
 14 if T == NIL
 15 return InsertionError
 16 IPAddr=IPAddr << 1
 17 T.data=TR.NextHop
 18 Add TR to the list
 19 return OK
```

– **uint32 SEARCH(uint32 IPAddr, PrefixRecordMatch Prefix)**

Description : This function is used to search the Next Hop information given IP address. It returns the next hop information in the longest prefix match node.

Input Parameters : The input parameter of this function is TableRecord structure which contains IP address,next hop IP address and subnetmask and the irrevalent data elements are ignored.

Output Parameters : Longest prefix Match for the given IP Address is the output parameter.

Return Values : It returns the next hop IP address in the unsigned integer format and it returns ERRNO in case of unknown IP address.

Pseudo Code :

```
SEARCH(uint32 IPAddr, PrefixRecordMatch Prefix)
 1 T = root
 2 HopAddr = NIL
 3 Prefix = ""
 4 tempPre=""
 5 IPAddr = TR.IPAddr
 6 while IPAddr!=0 and T!=NIL
 7 if MSB(IPAddr) == 0
 8 T = LEFT(T)
 9 tempPre+='0'
10 else
11 T = RIGHT(T)
12 tempPre+ = '1'
13 if DATA(T) != NIL
14 HopAddr = DATA(T)
15 Prefix+= tempPre
16 tempPre=""
17 IPAddr = IPAddr << 1
```

```
18 if T!=NIL
19 return DATA(T)
20 else
21 if HopAddr != NIL
22 return HopAddr
23 else
24 return UnKnownIPError
```

– **int Update(TableRecord TR)**

Description : This function is used to update the next hop information given IP address and the corresponding next hop IP address, if IP Address not found it will consider it as a new entry and inserts into the binary trie.

Input Parameters : The input parameter of this function is TableRecord structure containing IP address, Next Hop IP address and subnetmask.

Output Parameters : NONE

Return Values : It returns OK if the IP Adress exist and updation done sucessfully, it returns NewInsertion(integer) if IP Address does not exist before and inserted successfully and it returns ERROR if the updation fails.

Pseudo Code :

```
UPDATE(TR)
 1 T=root
 2 IPAddr = TR.IPAddr = CALC_SUBNET(TR)
 3 while IPAddr!=0 and T!=NIL
 4 if MSB(IPAddr) == 0
 5 T = LEFT(T)
 6 else
 7 T = RIGHT(T)
 8 IPAddr = IPAddr << 1
 9 if T != NIL
10 DATA(T)=TR.NextHopAddr
11 Update TR in the list
12 return OK
13 else
14 rval=INSERT(TR)
15 if rval!=OK
16 return InsertionError
17 else
18 return NewInsertion
```

– **int Delete(TableRecord TR)**

Description :This function is used to delete the entry from the prefix table i.e delete IP address and the corresponding next hop IP address from the binary trie given IP address and subnet mask.

Input Parameters : The input parameter of this function is TableRecord structure containing IP address, Next Hop IP address and subnetmask and the irrevalent data elements are ignored.

Output Parameters : NONE

Return Values : It returns OK on sucessful deletion of entry and ERRNO incase of unknown IP Address.

Pseudo Code :

```
DELETE(TR)
1 T=root
2 IPAddr = CALC_SUBNET(TR)
3 while IPAddr!=0 and T!=NIL
4 if MSB(IPAddr) == 0
5 T = LEFT(T)
6 else
7 T = RIGHT(T)
8 IPAddr = IPAddr << 1
9 if T!=NIL
10 DATA(T) = NIL
11 Delete TR from the list
11 return OK
12 else
13 return IPNOTFOUNDERROR
```

Chapter 4

Unit Test

4.1 Introdcution

4.1.1 System Overview

The Next Hop Finder system has following modules

1. Parse module: Taking file name as input from the user for parsing that file and calls insert function in the Forwarding Table Module to fill the Binary Trie.
2. Forwarding table module has following functionalities
   (a) Constructs the binary trie provided IP address, subnet mask and next hop address.
   (b) User can search for the Next hop address by giving the IP address.
   (c) User can update the data structure by providing the IP address, the subnet mask and the new Next hop address.
   (d) User can delete the an existing entry in the binary trie by providing the IP address and the subnet mask.
3. GUI module Provides the user interface for all the functionalities provided by the other modules. Help facility is provided to the User for accessing the system.

4.1.2 Test approach

The test approach used at this phase is the Integration testing. In integration testing all the interface functions and their return values are tested. Based on their return values Proper messages are displayed and necessary computation is done.

4.2 Parser Module

4.2.1 Test Plan

Features to be Tested

1. Opening of the file for parsing.
2. Checking compatability of the format of each line in the file.
3. Validation of data type of the IP address, subnet mask and next hop address.

Features not to be Tested

1. The insert statement called in the module and its return values are not tested.

4.2.2 Test Cases

PARSE-1 : A Non-Existing File Name as Input

Purpose : To check whether file not found error condition was handled or not.
Input : File name or file path.
Output : No File Found message is displayed.
PassCriteria : It should return FileNotFound error.
TestProcedure : The file pointer is checked for the null condition.

PARSE-2 : An Existing File Name as Input

Purpose : To check whether the input file was opened properly.
Input : File name or file path.
Output : NONE.
PassCriteria : File pointer which is not null.
TestProcedure : The file pointer is checked for the null condition.

PARSE-3 : An Empty File as Input

Purpose : To check whether Empty File Error was handled or not.
Input : FileName or file path.
Output : A File Empty Error message is displayed.
PassCriteria : Number of lines read must be zero. It should return an Empty FileError.
TestProcedure : Number of lines read are counted.

PARSE-4 : Proper Input

Purpose : To check whether this module paring the correct data or not.
Input : FileName containing proper (well aligned and correct) record.

Output : Number of lines containing valid data.
PassCriteria : It should be stored in a table record structure with out fail.
TestProcedure : Print the table record structure information.

PARSE-5 : Misaligned and Correct Input

Purpose : To check whether this module handling the mis-aligned records(eg: improperly spaced records) or not.
Input : FileName containing mis-aligned record.
Output : Number of lines containing valid data.
PassCriteria : It should be stored in a table record structure with out fail.
TestProcedure : Print the table record structure information.

PARSE-6 : Improper Input

Purpose : To check whether this module handles improper (eg:Incompatible type) data or not.
Input : FileName containing Improper record.
Output : NONE.
PassCriteria : It should be skip the record with out storing.
TestProcedure : Check weather or not skipCount incremented.

PARSE-7 : NULL String as input file name

Purpose : To check whether Parse module behaves when no filname was given.
Input : NONE.
Output : "NO FILE WAS GIVEN" error message is displayed.
PassCriteria : Proper error message should be displayed.
TestProcedure : Give the unintialized string as file name, check for expected ouput.

4.3 Forwarding Table

4.3.1 Test Plan

Features to be Tested

1. Insert feature of this module.
2. Search feature of this module.
3. Delete feature of this module.
4. Update feature of this module.

Features not to be Tested

1. Display to the GUI is not tested.

4.3.2 Test Cases

FWT-1 : Inserting into Entirely New Path

Purpose : To check the insertion functionality.
Input : Record containing valid IP address, subnet and next-hop address.
Output : Inserted successfully message is displayed.
PassCriteria : The number of created nodes is equal to subnet.
TestProcedure : The count is incremented if a node is created, that count is equated to the subnet.

FWT-2 : Inserting in Existing Path

Purpose : To check the insertion functionality.
Input : Record containing valid IP address, subnet and next-hop address.
Output : Inserted successfully message is displayed.
PassCriteria : The number of created nodes are zero.
TestProcedure : The number of newly created nodes are counted.

FWT-3 : Insertion Including Taversing and creating path

Purpose : To check the insertion functionality.
Input : Record containing valid IP address, subnet and next-hop address.
Output : Inserted successfully message is displayed.
PassCriteria : The subnet is zero.
TestProcedure : The subnet is decremented for each move to next node. At the end it should be equal to zero.

FWT-4 : Search for an Existing IP Address

Purpose : To check the Search functionality.
Input : A valid IP address.
Output : The corresponding next-hop address and prefix are displayed.
PassCriteria : The path traversed should atleast contain one next-hop address.
TestProcedure : The next-hop addresses in the path are stored and at the comple-tion of path if it should not be null.

FWT-5 : Search for an Un-Known IP Address

Purpose : To check the Search functionality.
Input : A valid IP address.

Output : "UnKnown IP address message" is displayed.
PassCriteria : The path traversed should not contain any next-hop addresses.
TestProcedure : The next-hop addresses in the path are stored and at the completion of path if it is null then test is passed.

FWT-6 : Searching Empty Table

Purpose : To check the Search functionality.
Input : A valid IP address.
Output : "UnKnown IP address message" is displayed.
PassCriteria : The expected error message is returned.
TestProcedure : Add default route to the trie, and delete it. Now try to search for the any record in the trie it results in the error message.

FWT-7 : Updating Exactly Matched Record

Purpose : To check the Update functionality.
Input : Record containing valid IP address, subnet and next-hop address.
Output : "Record sucessfully Updated" message is displayed.
PassCriteria : At the end of traversing the trie, the node should contain the next hop address.
TestProcedure : The trie is traversed based on the IP address, at the end that node should be updated with the new next-hop address.

FWT-8 : Trying to Update Non-Existing Record

Purpose : To check the Update functionality.
Input : Record containing valid IP address, subnet and next-hop address.
Output : "A New Record is Inserted" message is displayed.
PassCriteria : At the end of traversing the trie, the node should point to null.
TestProcedure : The trie is traversed based on the IP address, at the end that nodeshould point to null and the insert method is called with this record and its return value is handled.

FWT-9 : Trying to update an internal node.

Purpose : To check the Update functionality.
Input : Record containing valid IP address, subnet and next-hop address.
Output : "Node successfully Updated" message is displayed.
PassCriteria : At the end of traversing the trie, the node is not null and it should not contain the next-hop address.
TestProcedure : The trie is traversed based on the IP address, at the end that node's next hop address is updated from 0 to new value.

FWT-10 : Deleting Existing Record

Purpose : To check the Delete functionality.
Input : Record containing valid IP address and subnet.
Output : "Record sucessfully Deleted" message is displayed.
PassCriteria : At the end of traversing the trie, the node should contain the next-hop address.
TestProcedure : The trie is traversed based in the IP address, at the end that node should be deleted.

FWT-11 : Trying to Delete Non-Existing Record

Purpose : To check the Delete functionality.
Input : Record containing valid IP address and subnet.
Output : "No Record with these Details" message is displayed.
PassCriteria : It should return an error message.
TestProcedure : The trie is traversed based in the IP address, at the end that node should point to null or the node's next-hop address is zero. Then "No Record with these Details" message is displayed.

FWT-12 : Trying to Delete when There are No Records

Purpose : To check the Delete functionality.
Input : Record containing valid IP address and subnet.
Output : "No Records Exists" message is displayed.
PassCriteria : It should return the expexted error message.
TestProcedure : The root node of trie is checked for null, if it does then the error message is displayed.

4.4 GUI module

4.4.1 Test Plan

Features to be Tested

1. Upon starting the module execution, only the parse option should be highlighted.
2. The search button functionality of the GUI.
3. The Update button functionality of the GUI.
4. The Delete button functionality of the GUI.
5. The Help button functionality of the GUI.
6. The Contact Us button functionality of the GUI.
7. The Exit button functionality of the GUI.

8. The Parse button functionality of the GUI.
9. Validations of all the Input fields provided in the GUI

Features not to be Tested

1. The search button functionality on the screen after the search button is selected.
2. The update button functionality on the screen after the update button is selected.
3. The delete button functionality on the screen after the delete button is selected.
4. The parse button functionality on the screen after the parse button is selected.

4.4.2 Test Cases

GUI-1 : Parsing Could be Done First

Purpose : To check only parse button is activated and all others are not enabled.
Input : An input action event.
Output : All the other operaions on the table are disabled and parse is enabled.
PassCriteria : No other button responds before parse completes.
TestProcedure : The home page buttons except parse are not enabled, once the parse functionality completes all the buttons are enabled.

GUI-2 : Parse Button Functionality

Purpose : To check the Parse button functionality.
Input : An action event of clicking the Parse button.
Output : The parse form is displayled.
PassCriteria : Parse Form should be generated upon clickng on Parse button.
TestProcedure : Click on parse button.

GUI-3 : Search Button Functionality

Purpose : To check the Search button functionality.
Input : An action event of clicking the Search button.
Output : The Search form is displayled.
PassCriteria : Search Form should be generated upon clickng on Search button.
TestProcedure : Click on search button.

GUI-4 : Update Button Functionality

Purpose : To check the Update button functionality.
Input : An action event of clicking the Update button.
Output : The Update form is displayled.

PassCriteria : Update Form should be generated upon clickng on Update button.
TestProcedure : Click on Update button.

GUI-5 : Delete Button Functionality

Purpose : To check the Delete button functionality.
Input : An action event of clicking the Delete button.
Output : The Delete form is displayled.
PassCriteria : Delete Form should be generated upon clickng on Delete button.
TestProcedure : Click on Delete button.

GUI-6 : Help Button Functionality

Purpose : To check the Help button functionality.
Input : An action event of clicking the Help button.
Output : The Help is displayled on the screen.
PassCriteria : Help Form should be generated upon clickng on Help button.
TestProcedure : Click on Help button.

GUI-7 : ContactUs Button Functionality

Purpose : To check the ContactUs button functionality.
Input : An action event of clicking the ContactUS button.
Output : The Contact Information is displayled n the screen.
PassCriteria : Contact us Form should be generated upon clickng on ContactUs button.
TestProcedure : Click on ContactUs button.

GUI-8 : Exit Button Functionality

Purpose : To check the Exit button functionality.
Input : An action event of clicking the Exit button.
Output : Quits from the system.
PassCriteria : Window must be closed upon clicking on exit button.
TestProcedure : Click on Exit button.

GUI-9 : Input Valid IP

Purpose : To check weather or not it accepting valid IP addresses.
Input : Valid IP address.
Output : IP address will be given as an input to the calling function.
PassCriteria : It should not output an Invalid IP error message (Every Byte of an IP address must be in between 0 and 255 for a valid IP address).

TestProcedure : Take each octet and verify that it is ≥ 0 and ≤ 255.

GUI-10 : Input Invalid IP

Purpose : To check weather or not it accepting invalid IP addresses.
Input : Invalid IP adress.
Output : Invalid IP Error message.
PassCriteria : It should give an error message.
TestProcedure : Take each octet and verify that it is ≥ 0 and ≤ 255.

GUI-11 : Input Valid Subnet

Purpose : To check weather it accepting valid subnet or not .
Input : Valid Subnet Mask.
Output : Subnet will be given as an input to the calling function.
PassCriteria : Invalid subnet error should not be displayed (range of a valid subnet is 0 - 32).
TestProcedure : Check for the subnet mask ≥ 0 and ≤ 32.

GUI-12 : Input Invalid Subnet

Purpose : Checking for the Invalid Subnet mask.
Input : Invalid Subnet Mask.
Output : Invalid Subnet Mask Error message.
PassCriteria : Error message should be displayed.
TestProcedure : The subnet is verified the following condition, that it is ≥ 0 and ≤ 32.

GUI-13 : Display the IP address and Next-hop in a table

Purpose : To verify the display functionality which is invoked after the update, delete and search.
Input : A List of IP addresses and Next-hop addresses.
Output : All the input list showed in a table.
PassCriteria : Total number of rows(list size) given as input should be present in the output.
TestProcedure : Check the display function with the various number of input length.

4.5 Additonal Material

4.5.1 TEST LOG

Table 4.1: Test log for the Unit test plan

CASE ID	CASE NAME	INPUT	OUTPUT	PASS/FAIL.
PARSE−1	Non Existing File Name	data_no	NO FILE EXISTED	Pass.
PARSE-2	Existing File name	data	Parse Successfully	Pass.
PARSE-3	Empty file	data_empty	Given File is Empty	Pass.
PARSE-4	Proper Content in File	data	Parsed Successfully	Pass.
PARSE-5	Misaligned Input	data1	Row Skipped	Pass.
PARSE-6	NoN Compatable data in File	data2	Row Skipped	Pass.
PARSE-7	NULL String as File name		No File is given	Pass.
FWT-1	Inserting into New Path	4294967295 31 1073741823	Inserted successfully	Pass.
FWT-2	Inserting into Existing Path	4294967295 30 1073741823	Inserted successfully	Pass.
FWT-3	Insertion including Traversing and Creating New Path	4294901760 31 1010101010	Inserted successfully	Pass.
FWT-4	Searching an Existing IP	4294901760	1010101010 1111111111111111	Pass.
FWT-5	Searching a Partially matched IP	4294901761	1010101010 1111111111111111	Pass.
FWT-6	Searching a Unknown IP	16843009	IP Not Found	Pass.
FWT-7	Searching an Empty Table	4294901761	IP Not Found	Pass.
FWT-8	Updating Exactly Matched Record	4294901760 16 1010101011	Updated Successfully	Pass.
FWT-9	Updating Non Existing Record	16843009 31 16843009	New Insertion	Pass.
FWT-10	Updating Internal Node	16843009 30 16843010	Updated sucessfully	Pass.
FWT-11	Deleting Existing Record	16843009 31	Deleted successfully	Pass .
FWT-12	Deleting Non Existing Record	16843009 31	IP NOT Found	Pass.
FWT-13	Deleting Record from Empty Table	16843009 31	IP NOT Found	Pass.
GUI-1	Parse should be done first	NONE	Only Parse Button functionality is highlighted	Pass.
GUI-2	Parse Button Functionality	Clicking Parse button	parse form	Pass .
GUI-3	Search Button Functionality	Clicking the Search button	Search form	Pass.
GUI-4	Update Button Functionality	Clicking the Update button	Update form	pass.
GUI-5	Delete Button Functionality	Clicking the delete button	delete form	pass.
GUI-6	Help Button Functionality	Clicking the Help button	Help form	pass.
GUI-7	Contact Us Button Functionality	Clicking the Contact Us button	Contact Us form	pass.
GUI-8	Exit Button Functionality	Clicking the Exit button	Close the applicaion	pass.
GUI-9	Input Valid IP	1.1.1.1	Valid IP address	Pass.
GUI-10	Input InValid IP	1..1.1.1, 1.1.1, 655.22.1.1,55.a.1.1	Invalid IP address	Pass.
GUI-11	Input Valid Subnet	24	Valid Subnet	Pass.
GUI-12	Input InValid Subnet	1.1, a, 45	Invalid Subnet	Pass.
GUI-13	Display the IP address, Subnet and Next-hop in a table	Records	Table of records	Pass.

Chapter 5

Integration Testing

5.1 Introdcution

5.1.1 System Overview

Next Hop Finder system implements IP forwarding table lookup.When we transmit a packet through Internet it may travel through vari- ous intermediate nodes (ex: routers,hosts). Each router and each host keeps a routing table which tells the router how to process an outgoing packet.
MAIN COLUMNS IN ROUTING TABLE:

1. Destination address: where is the IP datagram going to?
2. Next hop: how to send the IP datagram?

In this system we will determine "Next hop IP address" given a "destination IP address".User is also allowed to perform manipulations such as update, delete on the routing table.

5.1.2 Test approach

The test approach used at this phase is the Integration testing. In Integration testing the functionality of the system by giving various possible Inputs to the system and check whether it generates the expected results. The test cases are designed so that it covers the functionality of software and include all kinds(valid and invalid) of the Inputs from the user.

5.2 Test Plan

5.2.1 Features to be Tested

1. Parse Functionality.
2. Search Funcionality.
3. Update Functionality.

4. Delete Functionality.
5. Input validation.

5.2.2 Features not to be Tested

1. Internal details of the system will not be tested

5.3 Test Cases

5.3.1 SYSTEST-1 : File Not Exist

Purpose : To check the whether the file name given by user exists.
Input : File Name by the user.
Output : FILE NOT FOUND error message.
PassCriteria : The file name entered does not exist.
TestProcedure : The file name entered in the text box is checked for its existence. If it does not exist then error message is returned.

5.3.2 SYSTEST-2 : Empty File

Purpose : To check whether the given File is empty.
Input : File Name.
Output : Given File is Empty message is displayed.
PassCriteria : If the given file is empty.
TestProcedure : The file name entered in the text box is checked whether is empty. If it is empty the error message is returned.

5.3.3 SYSTEST-3 : Search Existing IP Address

Purpose : To check the search functionality for existing IP address.
Input : IP address.
Output : Next-hop address and Prefix.
PassCriteria : If the entered IP address exist in the table or atleast one next-hop address is present in its path.
TestProcedure : The IP address entered by the user is given to the search function, if its return value is next hop address then search is successful.

5.3.4 SYSTEST-4 : Search Non-Existing IP Address

Purpose : To check the Search Functionality for Non-existing IP address.
Input : IP address.
Output : IP address Not Found.
PassCriteria : The Path traversed has no next-hop addresses. .

TestProcedure : The Input IP address is given to the search function, If it returns the IP NOT FOUND
value then error message is displayed.

5.3.5 SYSTEST-5 : Update Existing IP Address

Purpose : To check the update functionality, if the IP address exist in the table.
Input : IP address, Next-hop address, subnet.
Output : Updated successfully message is displayed and table is updated.
PassCriteria : If the Update returns OK.
TestProcedure : The inputs from the user are given to the Update function after validation, Its return value should be OK.

5.3.6 SYSTEST-6 : Update Non - Existing IP Address

Purpose : To check the Update functionality for Non Existing IP addresses.
Input : IP address, Next-hop address, Subnet.
Output : New Insertion Occured Message and table is updated.
PassCriteria : If IP address does not exist.
TestProcedure : The inputs from the user are given to the Update function after validation, Its return value indicate the new insertion.

5.3.7 SYSTEST-7 : Delete Existing IP Address

Purpose : To check the Delete Functionality for Existing IP address.
Input : IP address, Subnet.
Output : Deleted record sucessfully Message and Table is updated.
PassCriteria : The delete function returns OK.
TestProcedure : The inputs from the user are given to the Delete function after validation, Its return value indicates the sucessful deletion.

5.3.8 SYSTEST-8 : Delete Non- Existing IP Address

Purpose : To check the Delete Functionality for Existing IP address.
Input : IP address, Subnet.
Output : No Record Found error message is dispayed.
PassCriteria : The return value of the Delete Function is ERRNO.
TestProcedure : The inputs from the user are given to the Delete function after validaion, Its return value indicates the Non Existing IP address.

5.3.9 SYSTEST-9 : Delete IP addresss at root node in the trie

Purpose : To check the Delete Functionality of IP addresss at root node in the trie.
Input : IP address, Subnet.
Output : IP address successfully deleted.
PassCriteria : The return value of the Delete Function is OK.
TestProcedure : The inputs from the user are given to the Delete function after validaion, In Delete function the root node value set to zero, it returns OK.

5.3.10 SYSTEST-10 : Validating IP Address

Purpose : To check whether the given IP address is valid or not.
Input : IP address.
Output : Error message if entered IP address is not valid.
PassCriteria : Every Byte of an IP address must be in between 0 and 255 for a valid IP address.
TestProcedure : The IP address read from Text Field and it is checked, whether each octet is ≥ 0 and ≤ 255.

5.3.11 SYSTEST-11 : Validating Subnet

Purpose : To check whether the given Subnet is valid or not.
Input : Subnet.
Output : Error message if entered Subnet is not valid.
PassCriteria : If the subnet is in between 0 and 32 the it is a valid subnet.
TestProcedure : The subnet is checked if it is ≥ 1 and ≤ 31.

5.3.12 SYSTEST-12 : No Value in Text Field (in all forms)

Purpose : To check the whether the Textfield is empty or not. .
Input : User Input.
Output : Can not be empty error message.
PassCriteria : The text field should not be empty.
TestProcedure : The text field in all forms are checked if they have any value if it is submitted. If the field is empty, error messages are displayed.

5.3.13 SYSTEST-13 : No Insertions Happened during Parse

Purpose : To check the Parse functionality and format of file.
Input : File Name by the user.
Output : No Insertions happend error message.
PassCriteria : The file given is not of required format.
TestProcedure : The file name entered in the text box is opened and parse takes place, if the number of successfull insertions are zero. Then this error is returned.

5.4 Addtional Material

None.