CSE 486/586 Distributed Systems

Programming Assignment 3

Simple Key-Value Storage

a.) Components Used:

In order to implement Simple Key-Value Storage, the components SimpleDhtProvider needs to be written. The provider needs to implement the insert() and query() functions, where the values are inserted into the chord based on the hash value of the corresponding files. Following scenario explains the working and implementation of Simple Key-Value Storage.

b.) Component functionalities:

1.)SimpleDhtMainActivity: The buttons Test, Ldump and Gdump are identified and are provided with their onSetClickListener, functionality. The code to publish the values of Ldump and Gdump, is also written. Thus on Test button click, the OnTestClickListener class is instantiated, whereas a click on Ldump or Gdump, would call upon the query() function of ContentProvider.

2.) SimpleDhtProvider: The client and server task are implemented in the SimpleDHTProvider, which would establish a connection between the incoming nodes and the parent node “5554”. SimpleDhtProvider has a genHash(), method which does hashing of the file and returns the hash value, as a string. The genHash() method, takes the filename as parameter and generates SHA-1 hash as a hexadecimal string. This string is then compared to hash value of the active AVD's portStr, as well as to the hash value which is generated for the predecessor node. Here, Insert() and Query(), methods are called when Test button is clicked and thus the values obtained from the files are inserted into the chord, if they satisfy the condition.

a.) onCreate(): Here, the server task and client task are instantiated.

b.) returnVal(): Here, the search for the position of the key value to be inserted in the chord is performed.

c.) insert (): Here, the key and value are fetched from the ContentValues which are passed as an argument to the insert() function. The key obtained is then passed to the returnVal(), method, where calculation of hash is performed over key. The genHash() method is called and hash value for predecessor, key and current active portStr, are calculated. If the hash value of the key falls in the range of its predecessor and active AVD, then the value for the corresponding key is written, on that location, if not, the successor node is called and the key and value are passed as parameters until an appropriate location for the specified key and value is found. Basically, the concept of port forwarding is implemented here.

d.) query(): Here, the key to be searched is passed as an argument to the query(), which needs to return the corresponding cursor having key and the value pair. The key is passed as selection parameter which needs to be passed to the returnVal() function, which finds the corresponding position, where the record is stored and returns a boolean value, either true(in case of position found) or false(in case of position not found). In case of a false value returned back, the port forwarding needs to be performed and once the value is obtained, this value along with the key would form a cursor, which now needs to be backtracked, to the original caller. In order to do so, we need to keep track of predecessors, so that it becomes easy to reach the caller once the value is obtained. Here, we us the while() loop, which would act as busy waiting, which would end when the value reaches back to the originator or the caller.

1. Ldump: In order to get values on the click of Ldump, the list of files is retrieved using the getContext().fileList() method, and the file values are then hash and compared in order to determine whether, these fall into the range of Ldump, if yes, the values are added into the cursor and this cursor is returned to the SimpleDhtMainActivity, which then prints the value on the screen of the AVD.

2. Gdump: In order to get values on the click of Gdump, the list of files belonging to a particular AVD are retrieved and forwarding is performed, till the source is obtained, the values are added into the cursor and this cursor is returned to the SimpleDhtMainActivity, which then prints the value on the screen of the AVD.

e.)getPortno(): this function is used in order to return current AVD's port number.

Client Task and Server Task, are implemented, in order to establish connection, between the nodes and also to implement the insert() and query() functions successfully.

c.) Updation of predecessor and successor pointers:

The predecessor and successor pointers are updated as and when new nodes join the chord, and the referencing between the nodes also occurs, when the insert() and query() functions are invoked.
i.e. when hash values are calculated and comparisons are peformed.