A Report on DBLike

## Name Student Numbers

Harsimran Singh Maan

Kuntal Joshi

Prabal Sharma 86932126

# Architecture of DBLike

We have implemented a drop box like feature in our project called DBLike. For this we have a server side and a client side. On client side any user can use our application irrespective of the operating system he or she is using. In order to implement this platform independent feature we have coded in Java programming language. The client connects to the server via Java RMI. This client is authenticated on server side by a server called Broker server. This broker server in addition to client authentication also tell the server in a cluster, which will process the clients request i.e. there will be one master server which will interact with the client. This master server is selected by selecting the lowest serial number of the server in the server database.

If a client performs any kind of modification on the file or he uploads a new file or deletes a pre existing file, in all the cases the file will get replicated on all the servers of a cluster. Our application also handles sharing among the servers of different clusters. This is implemented by using logical link. This is how our application is handling server-side failure. On the client side 'hash' of the file is compared with the existing file hash on the server side, before upload or deletion of file. If there is any kind of conflict, it handled by using versioning. For example, if there are two files with the same name i.e. test.txt ..... . This course of action ensures simultaneous editing feature in our application. We are implementing a time delay in our application before uploading a file on the server to allow retrieval of file. This ensures optimization in our application.



# Server side Implementation

We have a server to select a master server. This is implemented by DBoxBroker. It has class as FileServerMonitor which checks whether the selected main server is running or not. If the server fails then it deletes that server which was acting as the main server. It also has an Authenticator class which authenticates the user to access this application. When the DBoxLike is started DBoxBroker starts the connection using Java RMI prior to any user input.

Our DBOXServer has a AliveCheck class which sends its name and port and sends the heart beat to Broker telling that the particular server has not failed and thus should not be deleted. The server uses Java RMI to connect to the network. FileReceiver class................................

# Client Side Implementation

We have a class called watchDir; which tracks the directory on the users machine which he want to share. Any changes in the directory are monitored by this class. It captures events like entry create, modify and delete. It scans through the path of the directory given by the user and gets all the file's names in that directory and calculates the hashes for those files and saves it as a hash map.

file transfer

sync

# Rationale for the design choices

# Known bugs

# Possible extensions