# Scalability

Scalability is the second most important quality attribute of the buzz space system. It is thus prioritized as being critical to the system. Scalability can be achieved on the buzz space system on many fronts including load, functional, administrative and generation scalability.

Load scalability will be realized through the use dynamic data storage structures which can grow rapidly and also be traversed rapidly during operations such as searching and filtering. Coupled with recursive algorithms (that perform optimally on large data structures), the data storage algorithms (for transferring, searching, encoding/decoding etc.) will increase 10 fold in performance for every 100 users actively making use of the data storage structures.

Functional scalability will be realized by continually improving the buzz system features. Features include allowing for the upload of virtually any types of files (if it’s not supported then archiving them before uploads would also work), using new thread pool API’s for threads and implementing methods that limit unnecessary features based on the bandwidth of different users (while still offering the maximum functionally of the buzz space). Users connected to up-to 1Mbps will receive the least amount of the buzz features, users using a 2-4Mbps line will receive above average features and users with 5Mbps+ lines will receive all of the buzz space features

Administrative scalability is not a major concern at the moment because the buzz system is limited to the members & students of the UP Computer Science department, but it will be realized by the Integrability of the buzz system (which is discussed further on in this document).

Generation scalability will be realized by using efficient MIME encoding that will complement the communication protocols used to provide swift feedback to multiple users on the system. The encoding will also assist with the transfer and storage of resources uploaded by the users.

# Maintainability

Maintainability of the buzz space system is considered a low priority, but necessary none the less. Maintainability can be achieved by making use of design patterns that promote modularity within the system. Defects, errors and updates in the buzz space system will then be easily isolated, allowing effortless repairs/replacements to be done on those modules without effecting the whole buzz system.

Backtracking and transmission error & detection algorithms (Auto repeat requests) will maintain the systems efficiently, reliability and safety. This will allow for uncomplicated roll-backs during unavoidable system restores. This ensures a probability of approximately 75% when it comes to retaining/restoring threads and resources to a specified condition within a given time period.

# Testability

Testability is also approximately leveled with maintainability when it comes to its prioritization within the buzz space system. This quality requirement will be achieved by firstly implementing testing early on in the development phase of the buzz system so that it can naturally extend (via maintainability) with the system as it scales up. The actual tests will be initially run on mock objects against a simple structured solutions. The modularity of the system will greatly assist the testability of it. The tests conducted on the system include input & output tests to check consistency (of resource transfer for example) and variation, there will also be network tests that monitor data rates and data packet activity to optimize the system’s network performance. Tests will utilize probing and debugging tools to achieve their purpose.