

HollyShare

Sean McCarthy

Bijit Hore

Ilya Issenin

Shannon Tauro

Songmei Han

Project Overview

Project Goals

Architecture

Existing Architectures

Module Descriptions

Comparison

Survey

Project Specifics

Other stuff

Project Goals

- **Design an architecture of a peer-to-peer application for sharing large multimedia files among a small group of users**
- **Create a program that implements this architecture**
- **Survey existing file sharing architectures and compare with our implementation**

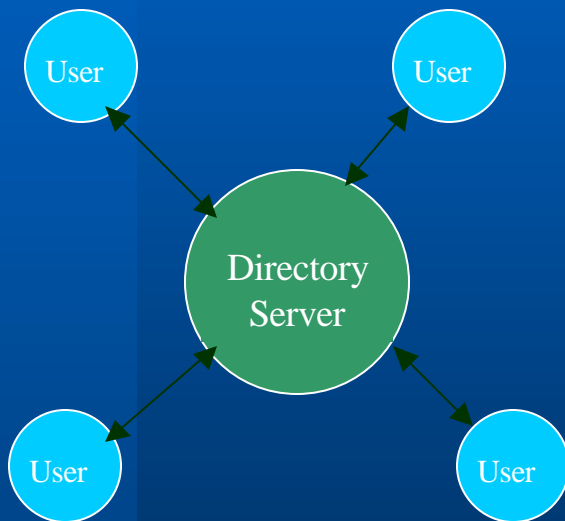
What we're doing

- **Catalogue of all shared files**
 - Permits browsing of available files and obviates the need for search
- **Select file(s) via the GUI and download**
- **No central server (distributed)**

Existing Architectures (I)

- **Napster**

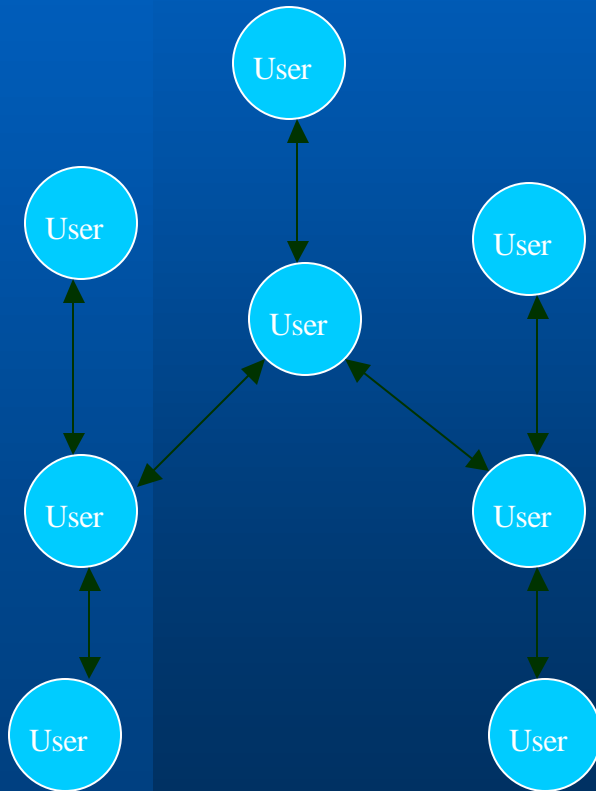
- File sharing using a directory server
- Server directs traffic between individual users



Existing Architectures (II)

- **Gnutella**

- File sharing using distributed search
- More robust than centralized model



Our architecture vs. Napster

- **Strengths of our system vs. Napster**
 - Unlike Napster, our application does not require a centralized directory server
- **Weaknesses of our system vs. Napster**
 - More information is stored at all nodes present in the system

Our application vs. Gnutella

- **Strengths vs. Gnutella**
 - No searching is necessary, since all available files can be seen – much more effective for small number of files
- **Weaknesses vs. Gnutella**
 - More node/file information must be stored in databases present in each node

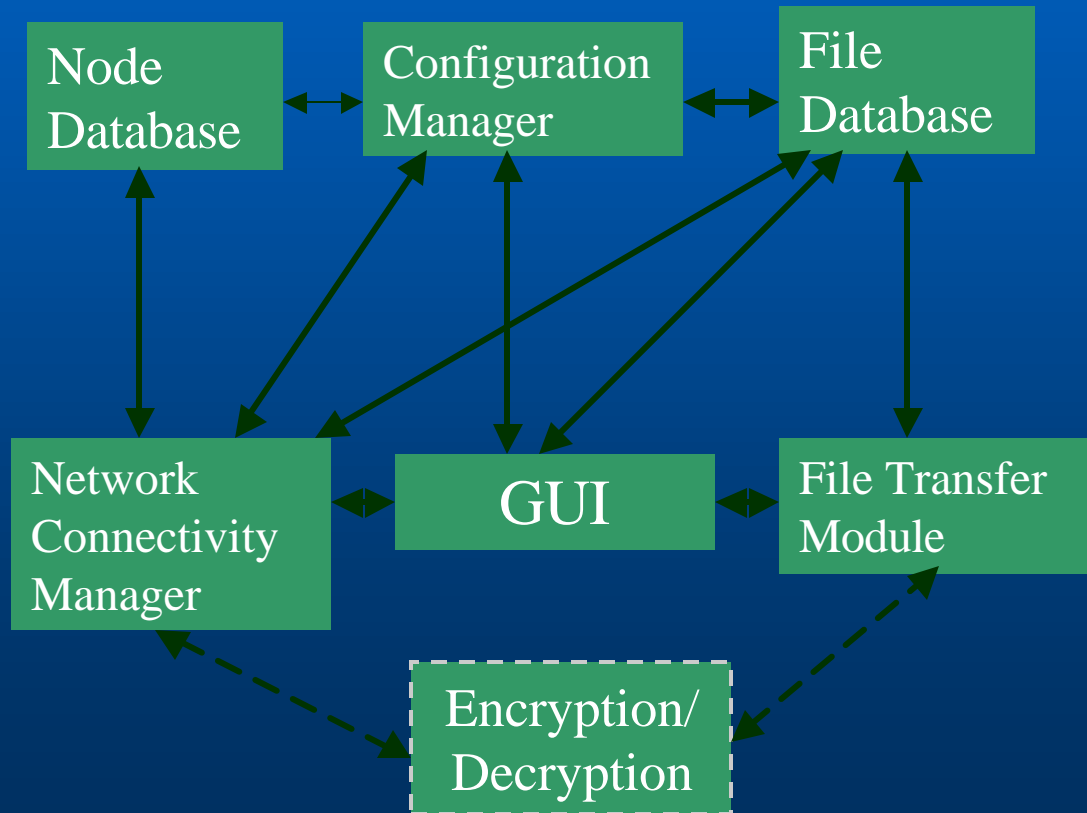
Application

- **The application is being designed in C++, using a development library called QT**
 - Provides GUI functionality and basic network functions
 - Library is platform-independent
 - Test platform: Windows machines, connected to the campus internet
- **6 modules are needed to provide the necessary methods and data structures for the application**

Application Architecture

- **Node Database**
- **File Database**
- **Configuration Manager**
- **Network Connectivity Manager**
- **File Transfer Module**
- **GUI**
- **Encryption module (optional)**

Architecture



Node database

- **Stores information about the nodes in the system**
 - IP address/port
 - ID (text name)
 - Date last logged on
 - Active flag
- **Provides functionality via several methods**
 - add node (create node/change node states)
 - delete node (remove nodes from database)
 - deactivate node (set node state as inactive)
 - get node list (returns list of nodes)
 - get next node (returns single nodes in a random sequence, used for connecting to other nodes)

File database

- **Stores information about the files in the system**
 - File name
 - IP Address/port
 - Size of file
- **Provides functionality via several methods**
 - remove file (remove from availability list)
 - get file (retrieve file information: address, port, etc.)
 - add file (new file available)

Configuration Manager

- **Stores information about the local system**
 - ID (text name)
 - IP Address/port
 - Shared directories
 - Path(s) for downloads
- **Provides functionality via several methods**
 - add share (add a shared directory)
 - remove share (remove a shared directory)
 - get shared directories
 - set download directory

Network Connectivity Manager

- **Provides all network communications except file downloads**
 - **Initialization: connecting a node to the network**
 - **Reconnection: maintaining a tree structure when one of the nodes goes off**
 - **Information routing: propagates information about new/deleted shared files/nodes across the network**

Network Connectivity Manager

- Implements HollyShare network protocol:

	Bytes	Field	Description
Message Header	0-1	Type	Type of the message, see next table
	2-5	Size	Message body length in bytes
	6-13	Reserved	Should be filled with 0s
	14-15	H_CRC	Message header checksum
Message Body	16 – (15+Size)	Body	Message body
	(16+Size) – (17+Size)	B_CRC	Message body checksum, 0 if message body is empty

Network Connectivity Manager

- Some of the message types:

Type #	Type	Description	Message body format
0	Res	Reset – first message after establishing TCP connection	Server port, ID (name) <i>int16, null-terminated string</i>
1	NShare	New share – have some new files to share	List of all shared files that should be added to the file list, each filename is augmented with the ID of the computer that holds it.
2	DShare	Delete file from the shared list	List of all files that should be deleted from the file list, each filename is augmented with the ID of the computer that held it.
6	MemOn	Member connected	IP, port#, ID <i>int32, int16, string</i>
7	MemOff	Member disconnected	IP, port# <i>int32, int16</i>
14	Ping	Request to send a Pong message	Empty
15	Pong	“I’m alive” message	Empty

File Transfer Module

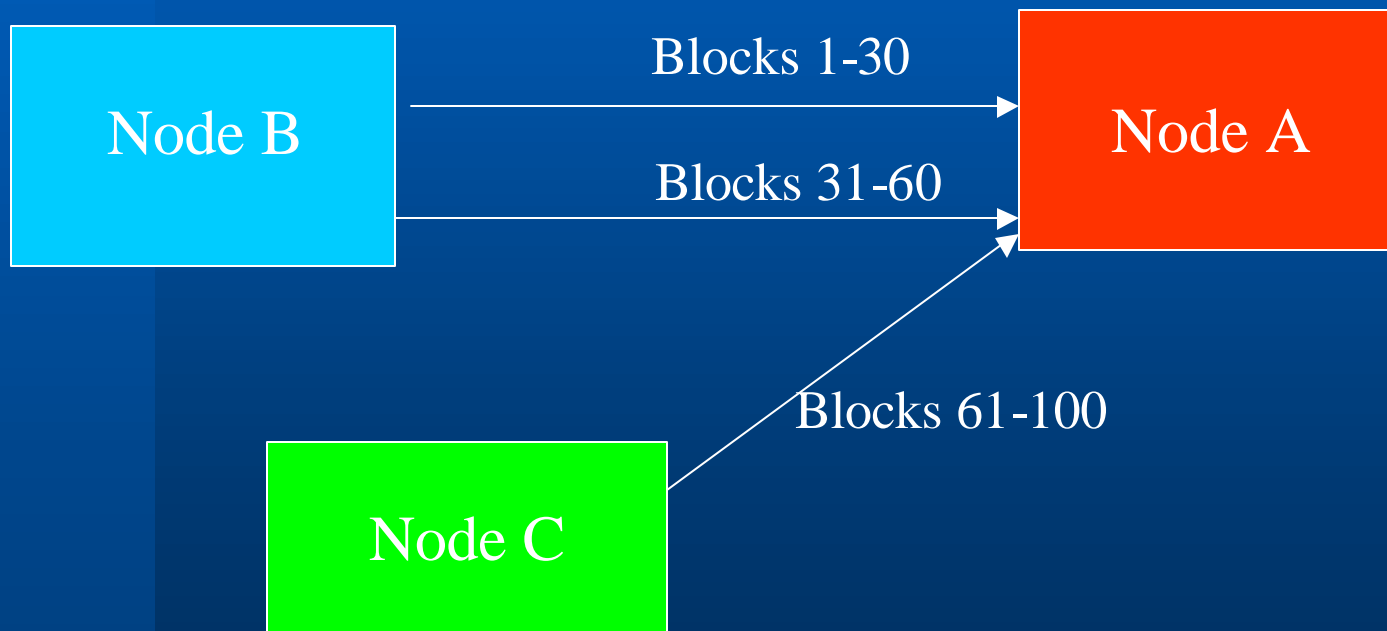
File Transfer module implements a file server and a file client :

The server caters to the requests from other active nodes on the network by serving specified blocks from the file in its shared directory.

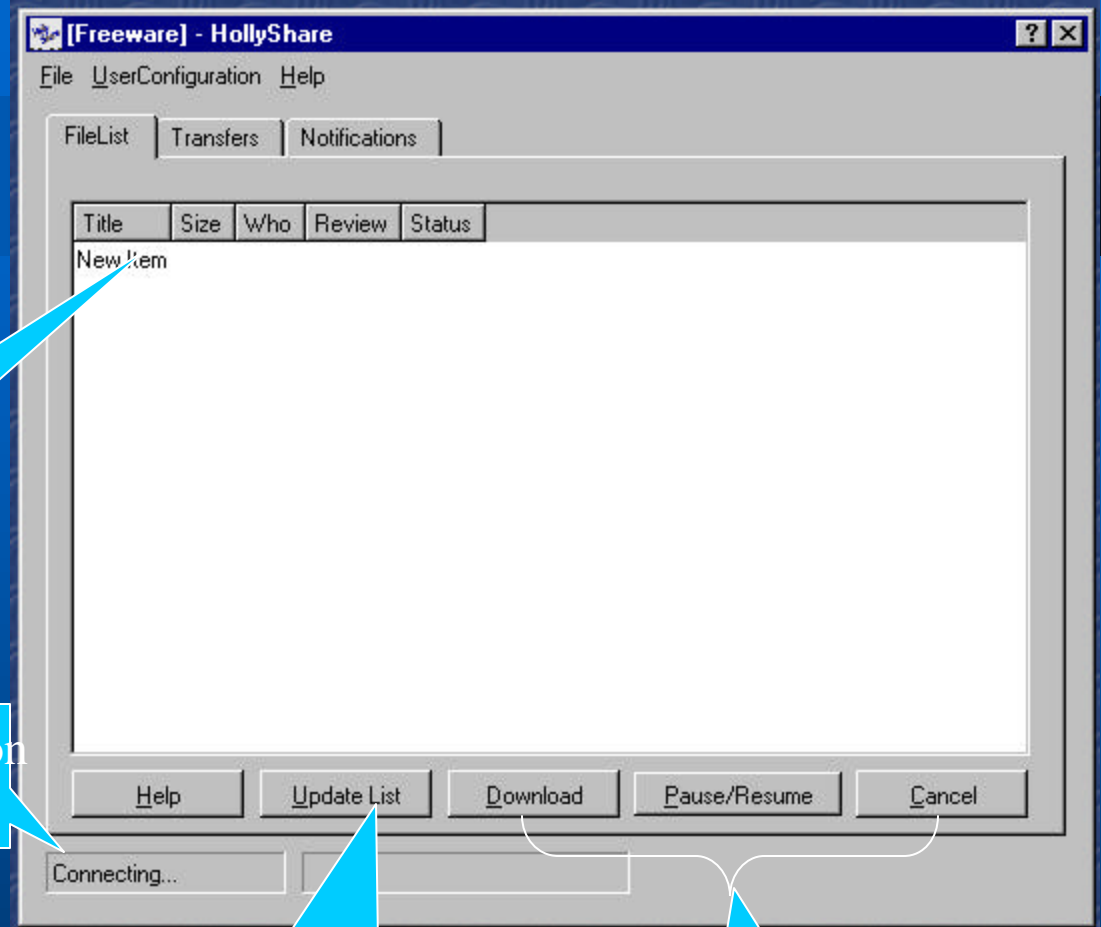
The client downloads files from other nodes. It also decides whether to setup a single or multiple connections to enable faster downloading of a file (perhaps from different sources).

Transferring a file

Shows a possible scenario of downloading a file (100 blocks long) from multiple locations



GUI



Movie Listings

Connectivity
Module

Receipt of
Connection Status

Application
Status

File
Database

Request & Receipt of
File Listings

File
Transfer

Request for File &
Receipt of progress

Notice to Update
Listings

Download Movie
Request(s)

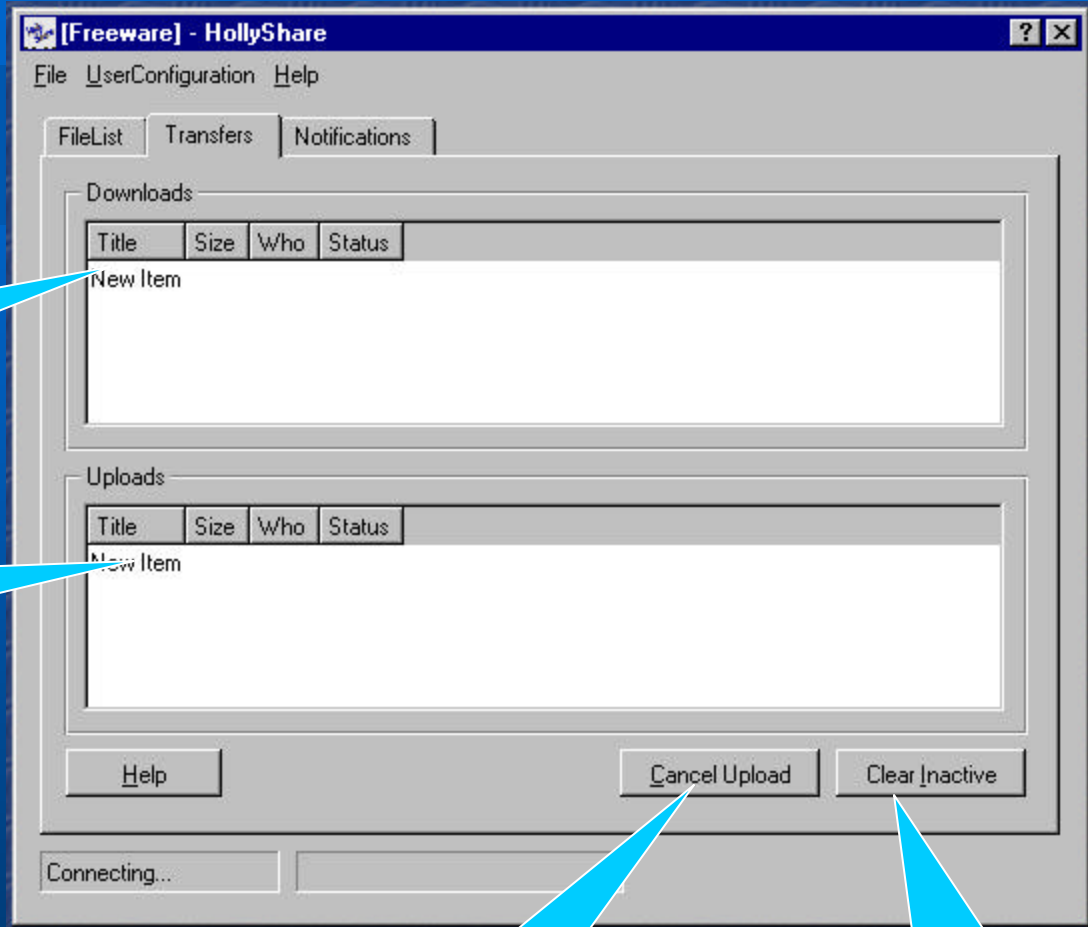
GUI

File Transfer

Updates of Progress of transfers

Current & Past Downloads

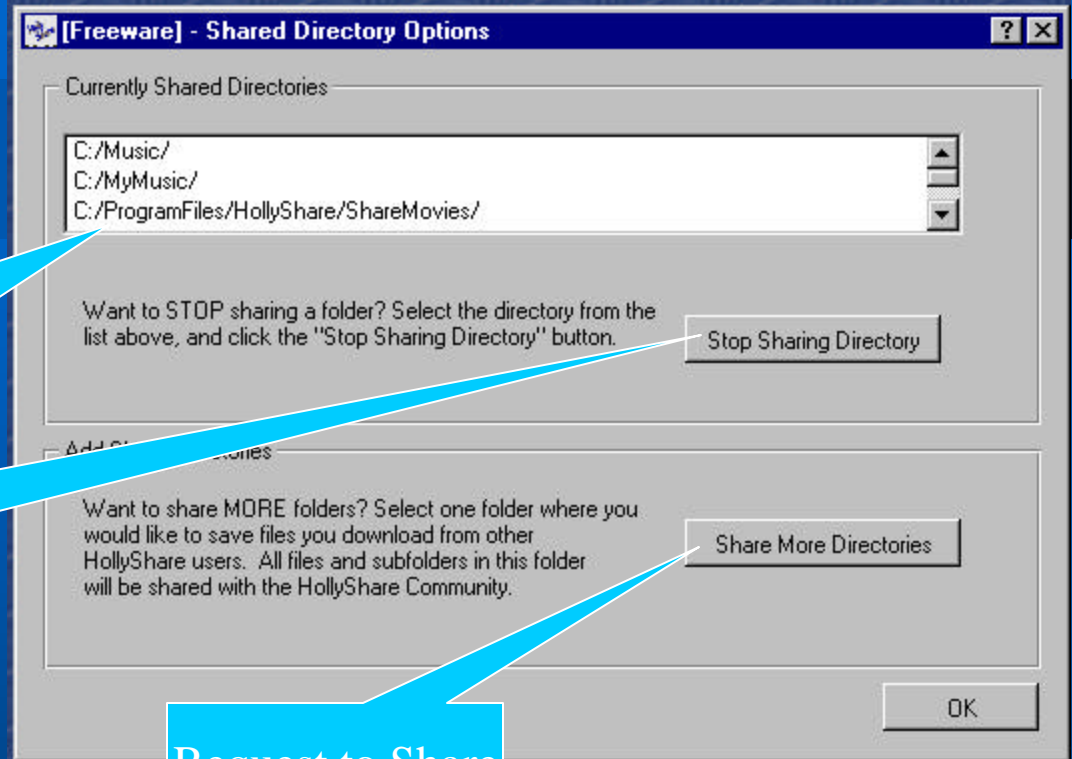
Current & Past Uploads



Cancel an Upload

Clear Lists of Complete Transfers

GUI



List of Currently Shared Directories

Request to STOP Sharing Directory

Request to Share More Directories

Receive Shared Listing

Remove Directory Entry

Update List of Shared Directories

Send Update to Database

Config Mgr.

File Database

Encryption Module (Optional)

- Provides security of all communications
- Together with checksums embedded in the HollyShare protocol, ensures the integrity of data
- All users of the group should have the same shared key
- Functions:
 - Encrypt the character stream stored in the buffer using ARC-4 stream cipher.
 - Decrypt the character stream with the same algorithm.

Survey on related literature

Papers that we have read up to now

1. Hurvicz, M. Groove networks: think globally, store locally. *Network Magazine*, vol. 16 (no.5), Miller, Freeman, May 2001. P 82 – 84.
2. Clarke, L.; Sandberg, O.; wiley, B. Hong, T.W. **Freenet: a distributed anonymous information storage and retrieval system.** *Design Privacy Enhancing Technologies (Lecture Notes in Computer Science*, vol. 2009), p. 46 – 66. International Workshop on Design Issues in Anonymity and Unobservability, Berkeley, CA, USA, July 2000, Proceedings
3. Fox, G. Peer to Peer Networks.
4. Parameswari, M.; Susarla, A. & Whinston, A. **P2P Networking: an Information-Sharing Alternative.**
5. Smart Decentralized Peer-to-Peer Sharing.
<http://www.ohaha.com/design.html>.

Survey on related literature (Continued)

6. Decentralized Resource Discovery in Large Peer-based Network.
<http://www.cubicmetercrystal.com/alpine/discovery.html>.

Papers that we are going to read.

1. Clarke, I. A Distributed Decentralized Information Storage and Retrieval System.
2. Forte, D. Peer-to Peer File Sharing is Here to Stay.
Network Security, 2001 (2), p 9 –11.
3. Zhao, W., Schulzrinne, H., & Guttman, E. mSLP- Mesh-enhanced Service Location Protocol. Proceedings Ninth International Conference on Computer Communications and Networks (Cat.No.00EX440) Las Vegas, NV, USA, 16-18 Oct. 2000.) Piscataway, NJ, USA: IEEE, 2000. p.504-9.
4. **Modern Peer-to-Peer File-Sharing over the Internet.**
<http://www.limewire.com/index.jsp/p2p>.

Survey on related literature

4. Modern Peer-to-Peer File-Sharing over the Internet. <http://www.limewire.com/index.jsp/p2p>.