Drag-and-drop File Sharing System Report

Ming Ge
Department of Electrical Engineering, Columbia University
mg2703@columbia.edu

Abstract

Nowadays, people have many choices of operation systems to work on computers, like Windows, Linux and Mac. Meanwhile, kinds of network environments are provided for people to get and share information, like traditional wired network, wireless network and Mobile Ad Hoc Network (MANET). "Drag-and-drop files sharing system" is developed to allow users to sharing files in different kinds of network environments and cross different operating systems. Furthermore, users running this system can automatically detect each other on the local network without any configuration. The system is also designed to be disruption-tolerant to adapt to dynamic network topologies.

1. Introduction

Information is more valuable than ever in information age today. Quick and convenient access to information has become more and more important. Computers and computer networks have been playing an important role in spreading information by providing information platforms and sharing services. While different kinds of operating systems and network environments have provided people more choices in diverse situations, they also bring in limitations. For example, some information sharing tools require specific system environment to run and many people must have experience when they need to transfer files but their computer can not connect to Internet and they have to use USB disks which brings many inconveniences.

"Drag-and-drop file sharing system" (DnD) is designed to provide a tool for sharing files in different network environments, including wired, traditional wireless and Mobile Ad Hoc Network (MANET) [1] , as well as on different platforms, including Windows, Linux, Mac OS.

To work in traditional wired and wireless networks, TCP/IP protocols must be used for DnD to communicate between computers. It should also work without DHCP, DNS servers since MANET has no fixed infrastructure and centralized server but a self-organizing and configured topology.

In both traditional network and MANET, we need to know the destination's IP address (although the schemes for allocating addresses may be different for them) to perform file transfer. The IP addresses may keep changing. It can be either reallocated by a DHCP server in a wireless network or by address allocation protocols or schemes in MANET. So hosts running DnD should also be able to automatically detect each other and get their addresses. In addition, as mentioned before, in MANET, the network topology is keep changing, network members can enter and leave the network at any time, so if we consider hosts running DnD forming a group on the local network, each group member should be notified when other members leave or join the group.

Apple's Bonjour [2] [3] technology is used to maintain this network group including updating the member list and informing each member other members' information.

To work on different operating systems, DnD should be platform independent. Java is used as the programming language.

The following parts of this paper are organized as following: part2 background and related work, part3 architecture, part4 user interface and tests, part 5 task list and appendix in the last.

2. Background and related work

Known as zero-configuration networking, Bonjour [2] [3] can enable automatic discovery of computation devices over an IP network. There is no need to enter IP address or DNS servers on those devices to perform host discovery. (To obtain that, those devices must be able to allocate IP address without a DHCP server, perform translation between name and addresses without a DNS server and locate or advertise services without using a directory server.) So Bonjour can be

really useful in MANET where a DHCP and DNS server are not available. The Java package of Bonjour we will use is com.apple.dnssd [4]. Some main methods we are using will be introduced later. There are some tools making Bonjour's library easier to use.

BonSwing [5] is a JAVA written framework for building GUI P2P network applications based on Bonjour on a regular always-on network, an ad-hoc wireless network or even in a disconnected scenario. It includes a list and tree component.

There is a drag-and-drop file exchange application built on BonSwing [6]. It implements basic drag-and-drop behavior and file transfer but is not designed to deal with the disruptions during transfer. We reuse part of the code to build DnD.

BonAHA [7] is another JAVA written library based on Apple's Bonjour to provide a framework for developing P2P applications. Compared to BonSwing, it provides more flexible and extensible structure for application development. BonAHA library is composed of the following Classes/Interfaces: BNode is defined to represent and store the host's information including host name, IP address and port number in the network. BListener is an interface with two method serviceUpdated and serviceExited defined to deal with the host leaving and entering, it will be implemented by developers. BService is the most import class in BonAHA which implements RegisterListener, BrowseListener, and ResolveListener interfaces defined in Bonjour's library. Developers will create instance of BService class and use the public methods it provided to make use of Bonjour library. Hash map is used to store the network member's information. DNSSD.register method in Bonjour is used to register a service (of certain registration or service type) that can be discovered via DNSSD.browse and DNSSD.resolve methods. DNSSD.browse method in Bonjour is used to browse service (of certain registration type) and add itself as a listener to get called when services are discovered or disappear. DNSSD.resolve method in Bonjour is used to resolve a new service name discovered via DNSSD.browse or a lost service to a target host name, IP address, port number and txt record. Then it will call the methods defined in BListener interface to deal with the service leaving and entering event.

3. Architecture

We define some rules for system design. First, "hosts" used here are not necessarily different computers, instead, they are identified by a IP address and port number pair. There can be multiple hosts on one machine. Second, we define the members of the group are the hosts that have registered (use DNSSD.register method in BonAHA) the same registration or service type. We will use "_7ds_filexchange._udp" for DnD. [3] has described the details about the type name. The global view of the network is shown in Figure1 (we use A._UDP for short in the figure), Host1,2,3 are in the same local network, but only host 1 an 2 are in the same service group and can see ("see" here means in the same group) each other. The service name registered here is used to provide a unique member name in the group and this name will be translated to the a hostname (IP address) and port number pair that registered the service and shown to each group member, and those information will be used to start TCP connection between members.

Figure2 is what happen from the host aspect when host 1 enters the group (host2 already in the group) and when host2 leaves the group.

Java is used to build this cross-platform system. Three aspects must be considered for design: GUI for the member lists in the group to drag-and-drop files to, showing file saving options, transfer progress and other information, sockets that are used to transfer files, interface to work with the BonAHA and Bonjour library. The modules and classes for the system are shown as in figure 3.

Java's Swing library is used to build GUI. The user's view of the network members is stored in tree structure, with the service type of the group as the root name, and all members are direct children of the root. TransferHandler is used to drag-and-drop file to the member nodes.

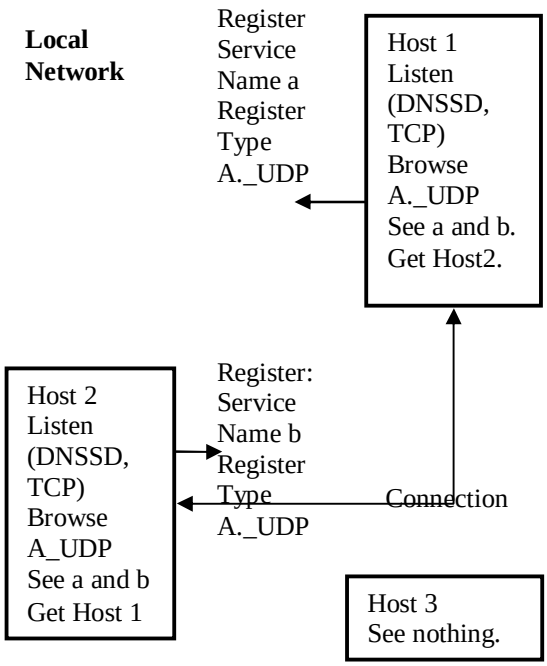

Figure 1 Global view of the network

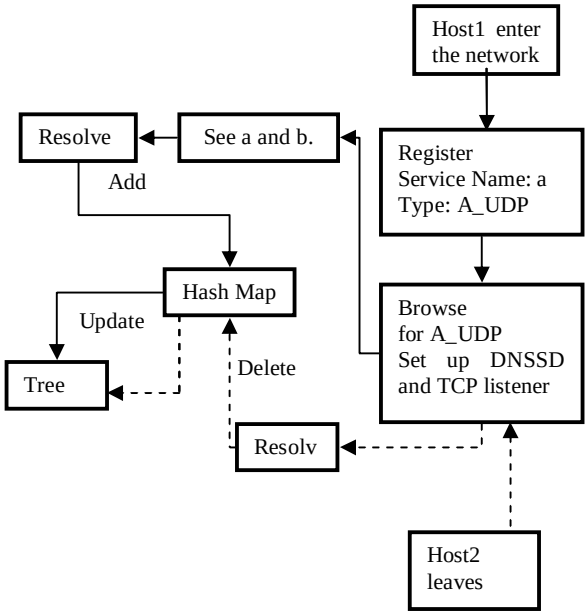

Figure 2 Local View of Host1

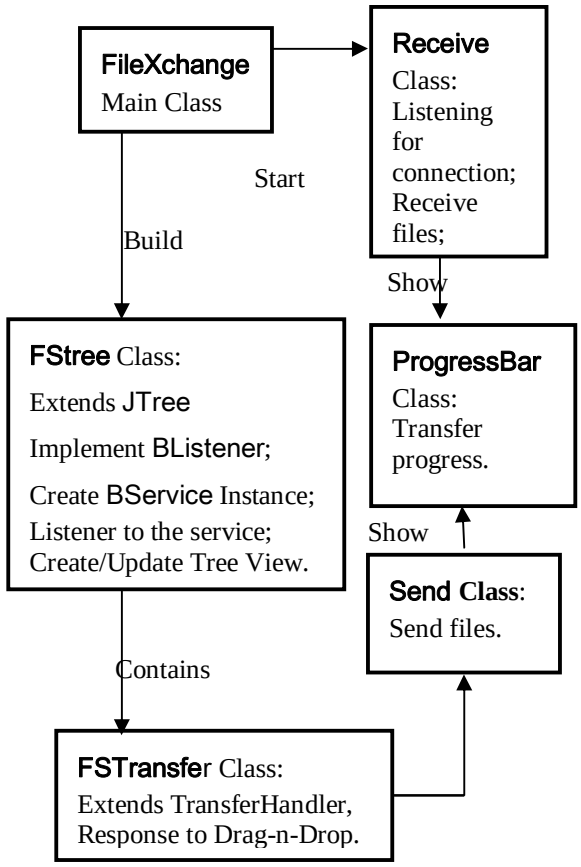

Figure 3 System Modules and Classes

To use the BonAHA, we need to define class (FStree in figure 3) that implements the interface BListener and create instance of BService class in BonAHA. The instance of BServcie will be added as the listener for the services on the network and does all the work to register, resolve service, and call functions in the BListener interface to update the tree view. When new services are discovered, the information of the hosts providing those services will be first encapsulated to BNode structures, and put in the hash map with the hostnames as keys, and add nodes in the tree view. When hosts are leaving the network, the corresponding BNode instances will be removed from hash map and so as the nodes in the tree view.

TCP socket is used to provide connection oriented data transfer service. Two kinds of data will be transferred for each transferring file: control and information data including the file's name, length, sender's information, recipient's response and file data, the control and information data will be transferred

before file data, so the recipient can choose to accept or decline the file and select a directory to save the file. There are five kinds of disruptions we need to concern during the file transfer: 1 Sender cancels the transfer. 2 Sender leaves the network during the transfer. 3 Receiver accepted the transfer but fails to choose directory to save the file within a certain time (before data transfer). 4 Receiver cancels the transfer during data transfer. 5 Receiver leaves the network during transfer. All of these can be handled by setting socket timeout, socket reset exception and comparing the length of the file with the length received. We also design progress bar based on the information of the received file length and total length. If multiple files are dragged and dropped at one attempt, they will be transferred one by one, but transfer of directory is not supported yet. Multiple drag-and-drop actions at the same time or during transfer progress are not permitted. File transfer procedure is shown in figure 4.

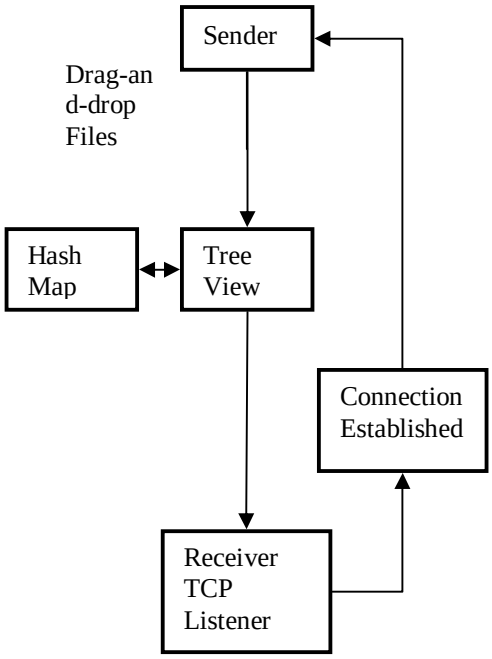

Figure 4 File Transfer Procedures

4. Tests and user interfaces

The system (including all libraries used) is written in Java. So it can run on Windows, Linux and Mac OS platforms with Java virtual machine and apple's bonjour installed. Windows Vista and Mac OS X 10.5 are selected to represent Windows and Mac OS platforms. Linux-based operation system Ubuntu (version 8.4) is chosen to represent the Linux platform. Tests include the system's functions of host discovery, files sending and receiving, as well as handling the exceptions during transfer. Tests are run both locally (multiple services on one machine) and remotely (multiple machines), as well as between different platforms, like between Windows and Ubuntu, Windows and Mac OS, etc.

The result shows the all functions of DnD work well on Windows Vista and Mac OS X 10.5 However on Ubuntu 8.4, due to likely faulty implementation of the Java Transferhandler for drag-and-drop on Linux, we can only receive files, and other functions are working well.

The tree view is shown in Figure 5:

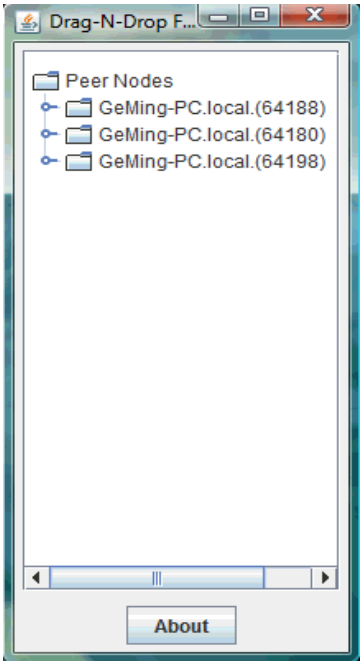

Figure 5 GUI—Group list

Once running, the services of the same register type (here the register type for this program is "_7ds_filexchange._udp", the service name is the name of the <host, port> that provide the service) registered on the network will be listed as nodes under the "Peer Nodes" list. File can be dragged and dropped to the target node. (Notice: a node represents a <hostname, port> pair on the network that is running DnD, the pair provides the information we need to start TCP connections to send files).

Upon receiving a file, the recipient can choose to accept or decline as well as the directory to save the file. Both recipient and sender can cancel the file transfer or leave the group and the other side of the transfer will get the corresponding information. During transfer, a progress bar will be shown at both sender

and recipient's sides indicating the percentage and data size that have been sent or received.

Figure 6 shows the scenario during file transfer, and figure 7 shows the information popped on recipient's side after the sender canceled the transfer.

The following figure is snapped during transfer.

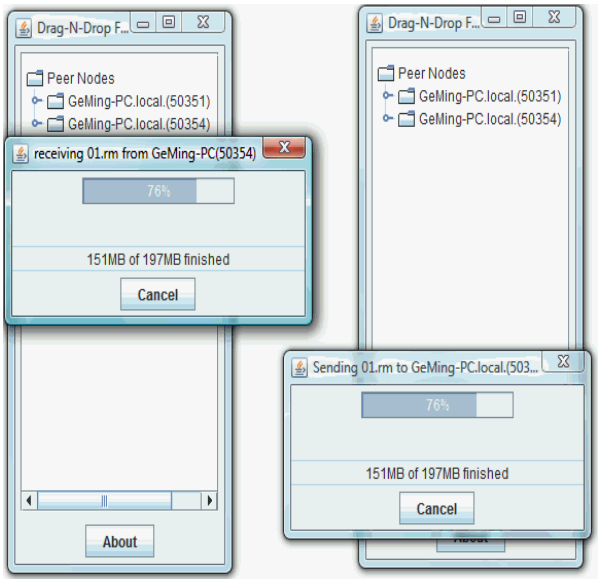

Figure 6 GUI--Transfer files

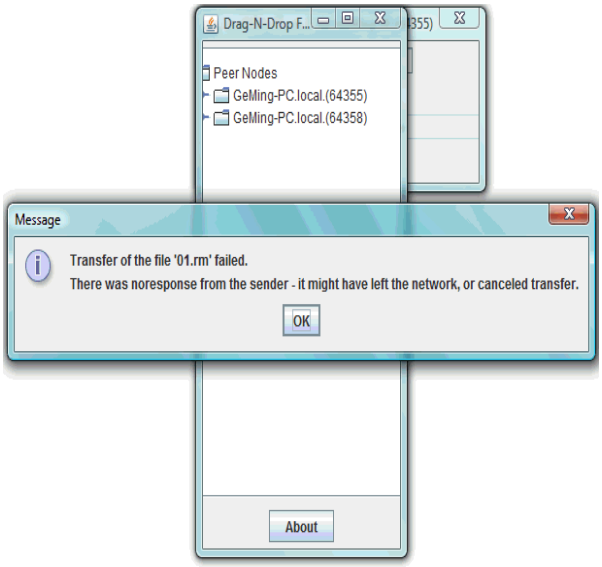

Figure 7 GUI--Sender canceled the transfer

5. Task list

DnD is built based on BonAHA library. Some of the codes of the file sharing system build on BonSwing [5] which implements basic drag-and-drop action and file transfer are reused. The tasks for this system includes: Use BonAHA to manage the member lists of the group and retrieve information of them; Build mechanisms to make transfer disruption-tolerant; GUI is improved for interactive messages for user's operation and file transfer.

References

[1] Roberto Beraldi, "Unicast Routing Technique for Mobile Ad Hoc Networks", the Hand Book of Ad Hoc wireless networks, Edited by Mohammad llyas,2003, chapter 7.

[2] http://images.apple.com/macosx/pdf/MacOSX_Bonjour_TB.pdf

[3] http://developer.apple.com/networking/bonjour/index.html

[4] http://developer.apple.com/documentation/Java/Reference/DNSServiceDiscovery_JavaRef/com/apple/dnssd/DNSSD.html

[5] http://bonswing.sourceforge.net/

[6] http://bonswing.cvs.sourceforge.net/viewvc/bonswing/7DS_J_filexchange/

[7] http://bonaha.sourceforge.net/

.

Appendix

Appendix A is the program documentation. Appendix B provides a program manual.

Appendix A
Program Documentation

Authors

Ming Ge
Department of Electrical Engineering
Columbia University
mg2703@columbia.edu

Name

Drag-and-drop File Sharing System (DnD).

Introduction

The System is built to provide files sharing service in traditional wired, wireless or Ad-Hoc network environment. Services on the local network can be automatically discovered without user's configuration.

System requirement

1. Java Virtual Machine/JDK 1.6 or newer version.
2. Apple's Bonjour is installed.

Installation instructions

An open source Java based software—IzPack [1] is used as the installation tool.

Installations for Windows Platform, Linux, Mac OS (DnD_install.jar) are available now.

For windows platform, double click DnD_install.jar to start the installation and following instructions to install DnD.

For Linux and Mac platform, use "$java -jar DnD_install.jar" to start the installation and following instructions to install DnD.

Operation

On windows platform, run Drag-and-Drop.jar by double clicking it in the installation directory, for Linux and Mac platform, go to the installation directory and use "$java -jar Drag-and-Drop.jar" to run the application.

Program internal operation

Once running, the services of the same register type (here the register type for this program is "_7ds_filexchange._udp", the service name is the name of the <host, port> that provide the service) registered on the network will be listed as nodes under the "Peer Nodes" list. File can be dragged and dropped to the target node. (Notice: a node represents a <hostname, port> pairs on the network that is running DnD, the pair provides the information we need to start TCP connections to send files).

Upon receiving file, the receiver can chose accept or decline and choose the directory to save the file. Both receiver and sender can cancel the file transfer or leave the group, and the other side of the transfer will get the corresponding information.

During transfer, a progress bar will be shown at sender and receiver indicating the percentage and data size that have been sent or received.

If multiple files are dragged and dropped at one attempt, they will be transferred one by one, but transfer of directory is not supported yet. Multiple drag-and-drop actions at the same time or during transfer progress are not permitted.

Figures 5-7 in Part 4 of the report are the screen dump for the system.

Limitation

Due to likely faulty implementation of the Java Transferhandler for drag-and-drop on Linux, sending files is not working. However files can still be received.

Future work and enhancement

Security features can be added, like group password needed to register the service on the network, SSL and Digital Signature can be used to secure the file transfer.

Acknowledge

Application is developed using the framework of BonAHA. BonAHA is developed by Suman R. Srinivasan. BonAHA is a framework built on Apple's Bonjour library and used to develop P2P applications. Some codes from an earlier version (by Suman R.

Srinivasan) of drag-n-drop file sharing application which can be found in [2] are reused.

Reference

[1]
http://izpack.org/
[2]
http://bonswing.cvs.sourceforge.net/viewvc/bonswing/7DS_J_filexchange/.

Appendix B
Program Manual

Name

Drag-and-drop File Sharing System.

Synopsis

N/A.

Availability

Installations for Windows, Linux and Mac OS (DnD_install.jar) are available now.

Description

The System is built to provide disruption-tolerant file sharing service in wired, traditional wireless or Ad-Hoc network environment. Services on the local network can be automatically discovered without user's configuration.

Features

1. Cross-platform. The system can run on Windows, Linux and Mac OS platforms.
2. Zero configuration. Hosts that run the application can automatically discover each others to form a network group. Hosts can leave or join the group, all the members will be notified and automatically update the member list.
4. Disruption-tolerant. The system can handle all exceptions during file transfer.
5. Multiple files transfer. Multiple files can be selected and transferred by one drag-and-drop action.

Configuration

N/A.

Option

N/A.

Notes

Due to likely faulty implementation of the Java Transferhandler for drag-and-drop on Linux, sending files is not working. However, files can still be received.

See Also

See Program Documentation in Appendix A.

Authors

Ming Ge
Department of Electrical Engineering
Columbia University
mg2703@columbia.edu

Acknowledgements

Application is developed using the framework of BonAHA. BonAHA is developed by Suman R. Srinivasan. BonAHA is a framework built on Apple's Bonjour library and used to develop P2P applications. Some codes from an earlier version of drag-and-drop files sharing application (by Suman R. Srinivasan) which can be found in [1] are reused.

Copyright

action of contract, negligence or other tortuous action, arising out of or in connection with the use or performance of this software.

Reference

[1]
http://bonswing.cvs.sourceforge.net/viewvc/bonswing/7DS_J_filexchange/.