© Aapo Kyrölä / Sulake Oy, 20000503
aapo@taivas.com

FUSE Server Technology

What is it all about?

Brief Description

FUSE technology is client-server system for graphical multiuser web-communities and games. Server technology is 100% Java™, while FUSE clients have been implemented using Macromedia© Shockwave™-software. Client technology is not restricted to Shockwave, since communication between client and server uses text-based *fuse-protocol,* which runs on top of TCP/IP.

Currently the best example of a fuse-application is MobilesDisco, a virtual chatroom with pseudo-3d-graphics and animation: http://www.mobilesdisco.com

---

FUSE Server – Core Features

Virtual Space

FUSE server manages the virtual space(s) where avatars are able to move and do things.

- space is divided into a matrix of sectors which can have different altitudes. A sector may be either empty or occupied by objects. The server takes care of finding moving paths around furniture and other people, thus preventing collisions. Because intelligence is fully in the server, clients are not able to do illegal actions, like walk through a table.
- hearing range: normal chatting can be heard only by the people near the source, but *shouting* is heard by everyone in the room. Clients receive only the chat messages they really hear – however, one may *see* that people in the other side of the room are talking, because their mouths are moving!

Server-driven intelligence

Basically, all intelligence is in the server. Placements of figures, their actions, watching directions etc. are all decided by the server. Responsibility of the client is only to display and animate the situation in the virtual space and broke user's requests to the server. For example, if the user is willing to go from place A to B, (shockwave-)client tells this to the server using fuse-protocol, server calculates the path and then, on each *cycle*[1] tells all the clients where the figure is currently. Benefits of this server-centric approach are:

- server has always full understanding of the overall situation in the space. This prevents clients from going 'out of sync'.
- poor network performance of individual users do not affect other
- security reasons: clients cannot do illegal actions without server's permit.

---

[1] Server broadcasts (usually) every 1/2 second the status of each figure in the space. Status contains the location, direction and actions of an individual.

- client-programming becomes easier and straight-forward. Also the footprint of the client remains smaller due to small amount of application code needed.

Handling of network problems and slow connection

FUSE-server is designed to work well with slow and instable network connections (..leave fast action games out). Some technical design details:

- status-packet which is sent regularly (usually every 1/2 second) is sent only after confirmation of receiving the previous packet has been received. Because each status-packet fully overrides the previous, leaving some packets out from the pipe is not critical: client may see jerky animation but it still works. And if the network lag is only temporary, the situation stabilizes immediately when possible.
- every status packet is compared to the last one and only modifications are sent.
- every network connection is monitored automatically: if a connection is silent over 60 seconds, it is disconnected.

For example in MobilesDisco network data rate average is approximately 500 bytes / second when high traffic. Maximum is little more than 1 kilobyte / second, which is adequately handled even by normal modems.

Security

Although the fuse-protocol is not encrypted (due to limitations of Shockwave) and is human-readable, connecting to the server requires decrypting a secret code. This prevents other people from easily doing their own clients without our permission.

Also, when connecting to the server, the client must tell which version of the client (s)he is using. If the version is not compatible with the server, the user is asked to reload new version (it happens often that browsers cache old version of the client shockwave-movie).

Integration to relational databases

FUSE technology incorporates a robust and efficient java database object-model, which uses JDBC. Database-operations can be distributed to a separate machine by using RMI, if needed. Database-object architucture is designed to clearly separate SQL-programming from the programming of application logic. It is then easy to share work between programmers: database programmers can concentrate on the thing they do better.
Currently we have used MySQL (http://www.mysql.com) as our database engine – integration to other databases should be no problem.

FUSE Server also collects basic usage statistics automatically.

Distributable

On high-traffic sites, there must be several “lobbies” or separate virtual spaces in order to split users into smaller groups. FUSE Server makes this possible by using Java™ RMI-technology. The solution is a light *entry-server* which the user first contacts. This entry-server manages information about the fuse-servers online, which host the actual lobbies. User can then choose which lobby to log in.
FUSE servers tell regularly the entry-server using RMI that they are alive and it is possible to dynamically increase or reduce the number of active servers.

This technology was used in Radiolinja Snowball (Lumisota)-game, where each FUSE-server hosted two mountain cottages, each having capacity of 25 simultaneous connections.

Command-line admin

As a standard module, FUSE server contains a password-protected command-line admin tool. Admin tool can be used, for example, for throwing users seeking trouple out or for controlling AI-driven bots etc.

FUSE Server – modular features

This section describes the additional features which can be included in the fuse server, but are usually so project-specific that cannot be counted into the "core"-features.

Intelligent bots

One of the most fascinating possibities in FUSE applications are artificially intelligent bots. For example, in MobilesDisco the bartender "Maarit" is computer controlled and knows how to react to users' verbal requests.
Bots can be like normal users, like Maarit or something else, like animals. Bots may have more freedom in their actions and movements than regular users because they are run inside the server.

Future developments:

• we are planning to implement a mobile software agent-layer to Fuse Server which would make possible for third parties to implement their own intelligent bots. Java gives good possibilities for doing this in secure ways, for example by using Voyager™-technology (http://www.objectspace.com/products/prodVoyager.asp).
• LISP-interpreter may be also integrated into FUSE server making bot-programming more efficient

FUSE Show System

With FUSE Show System it is possible to control spotlights, discoball, ambient light or launch different kinds of performances in the virtual space. For example, the disco-room in MobilesDisco uses FUSE Show System for doing all the light effects.
Show programmes can be scripted beforehand or controlled in realtime with the admin-tool.

Games

Of course, it is possible with fuse to implement many kinds of single- or multiplayergames, like chess, solitaire, action games such as the Radiolinja Lumisota-game etc.. FUSE Server is very modular and provides lot of freedom for the programmer.

FUSE Server – requirements

Operating systems

Since the server is 100% Java™, it runs on virtually every server platform. We have succesfully tested it on following environments: Linux (x86), Solaris (SPARC), MacOS X (PowerPC), AIX, Windows 95/98/NT.
Unix-platforms are strongly recommended because of the ability to administrate then remotely.

Performance

We have not yet made systematic test of FUSE Server performance. In practice, server has been able to handle hundred of simultaneous connections on a Linux-server without any problems. We except it to be able to handle 200-300 simultaneous connections on an efficient Linux-server.

Network performance is often more crucial than processing power. Problem is not the amount of data needed to transfer (data-rate only 0.5 – 1.5 kb /second) but so called “network lag”, which means that data doesn not transfer quickly enough between client and server.

It should be also remembered that FUSE server is very easy to distribute on multiple servers. Usually people will be splitted into rooms of maximum 30 users and thus it is easy to share user load simply by sharing rooms with servers.

Memory

FUSE server has small memory footprint. To host 50 users the same time you need approximately 15 megabytes of memory.