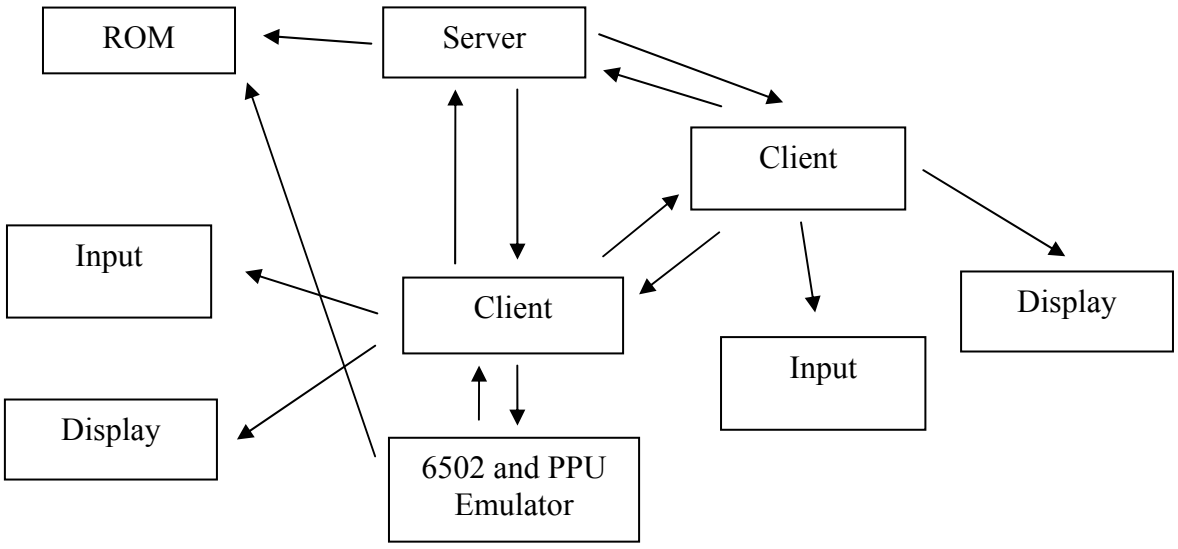
Bernard Peng
bpeng
netNES

Description
The Nintendo Entertainment System is a gaming console from the 1980's that had hundreds of games for it. Using current hardware, we can emulate the NES and play the old games on Linux computers. One of the most fun aspects of the NES system was the ability to play 2-player games with a friend. However, in emulators, this aspect is often neglected. Many emulators do support play over a network, but you need to contact that person beforehand to specify the game, agree who's hosting, and specify an IP to connect to. netNES will be an emulator that aims to focus on the 2-player networked aspect of the games. It will support a main server for storing ROMS and organizing games between players. Individual clients can connect to the main server to choose the game they want to play or meet up with a friend. Starting a game will be as easy as choosing the game, and another player to play with. The server will determine which client's computer will do the emulation work, and start a game with the 2 players.

System Model Diagram



Components
ROM – This component is a class representation of an NES game. It includes data structures to hold all the contents on a ROM file.

Server – This component serves as a library that holds information for all the ROMs as well as a place for players to meet and start a game. It keeps track of which games it has in its library, which players are connected, and what game they are interested in playing. When 2 players decide on a game, the server chooses which client will handle the emulation, and tells the two clients about each other. The server sends the ROM data over to the client that will handle the emulation, and then the server no longer plays a role between those clients until

they are finished and wish to find a new game to play.

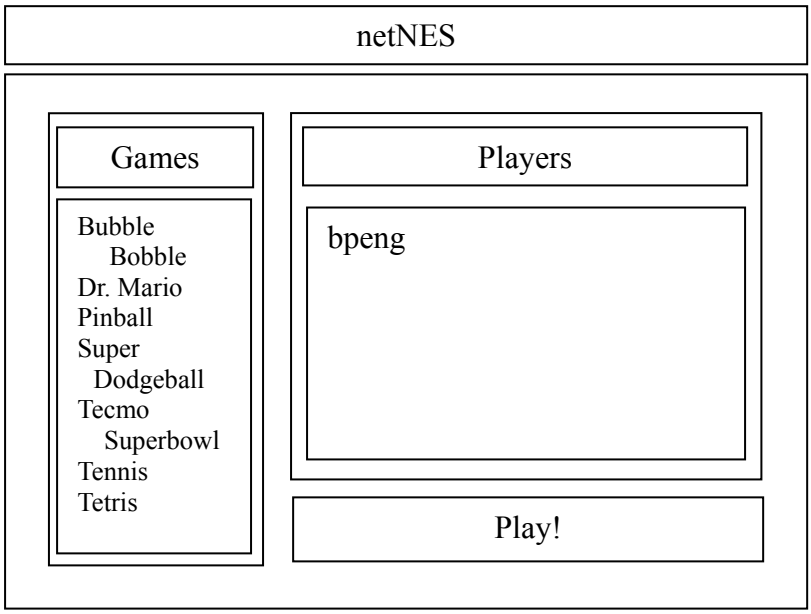
Client – This component is used to first connect to the server and allow the user to browse available games and players. Once a game and other player is decided upon, the client is able to communicate with the other client involved. One of these clients will call upon its Emulator component to begin emulation of the game. The Emulator will relay back to the client should send to its Display, and the Display of the other client (both clients will display the same image). The client will also accept data from the Input (and the Input of the other client) and send that to the Emulator component.

Display – This component is used to display the NES portion of the program. It will have the tile set stored in its memory (since the NES has tile-based graphics). It will accept input from the Client specifying which tile should be placed where on the screen.

Input – This component takes the players keyboard (or game pad or mouse) input and relays it to the Client.

6502 and PPU Emulator – This component takes the data of a ROM (which is sent from the server to the client to this component) and emulates each instruction of the game, as well as extracting other information (such as the tile set) from the ROM. It sends the output of the game (the display information) to the Client, and also takes the input of the game (the players controls) from the Client.

User Interface Diagrams and Descriptions



This diagram is of the Client's GUI. The left column is a list of the available games stored on the central server. The user clicks on a game from this list, and the right side of the window loads the other players waiting to play that game. The user then clicks on the user he/she wants to play with and hits the "Play!" button. The other user would get a simple dialog box asking if he/she will accept or decline the offer. Once a game is started, a new

window pops up with the NES display.

Requirements

Features
High Priority
-emulation of the 6502 processor (the NES' core processor) (high priority)
-emulation of the PPU (the NES' graphics processor) (high priority)
-loading of ROMs
-centralized server to organize games
-local clients with GUI to connect to server
-keyboard input

Medium Priority
-support for sound
-better compatibility

Low Priority
-save states
-joystick input support
-mouse input support
-filters to enhanced display quality

Performance
The performance requirement for netNES would be to have it able to run a NES game at full speed in real time.

Testing
The Emulator component should be able to be tested separately from the rest of the components.

Reliability
Although emulation is often imperfect, this program should not crash if an error occurs, and should be able to safely detect whether or not it will be able to run a given game.

Easy of Use
Since there is not a lot for the GUI to do (choose a game, choose a player and begin) this program should be very easy to use. There will always be only 3 clicks required to start a game (click on a game, click on a player, click on play button).

Risky Parts
The core part of this project (the Emulator component) is the riskiest part. I do not have any experience with emulation, so I am not sure if the NES processor would be easy enough to emulate in the time given. The legality issue is also another big risk, since copying and storing ROMs isn't legal.