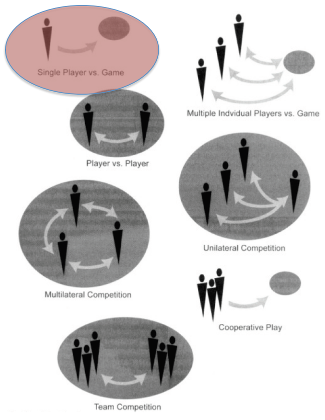
G54GAM - Games

- Multiplayer Games

Multiplayer Games

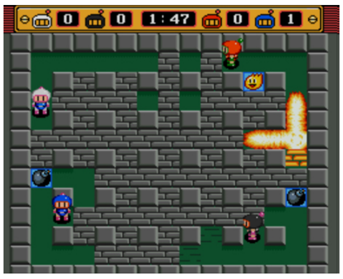
- Single Player
  - Pre-defined challenges, "campaigns"
  - Artificial Intelligence controlled opponents using Finite State Machines
  - Simple MVC design
  - Player vs "the game"
- Multi-Player
  - More than one player can play in the same environment at the same time
  - Interaction with other players forms a key challenge of the game play
  - How can we build such a system?
  - What are some of the issues that arise?



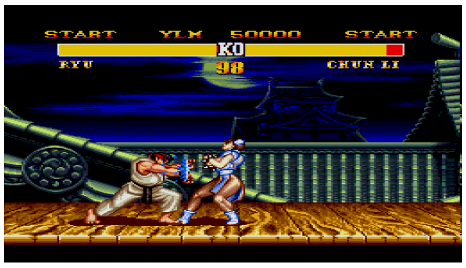
Where is the view?

- Local
  - Players are co-located
  - Share the same console / screen / pc
  - Share or split screen into two or four sections
  - Arcade games, racing, fighting, co-operative shooters
- Networked / Online
  - Players are physically separated
  - Game play is shared over the network / Internet
  - Many combinations of players 2 -> ??
  - FPSs, MMORPGs

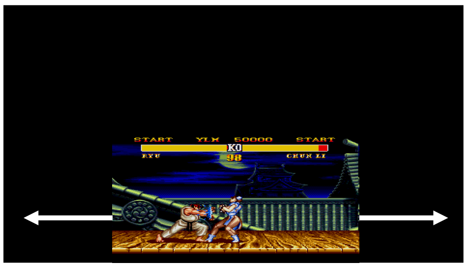
Static Shared View – Bomber Man

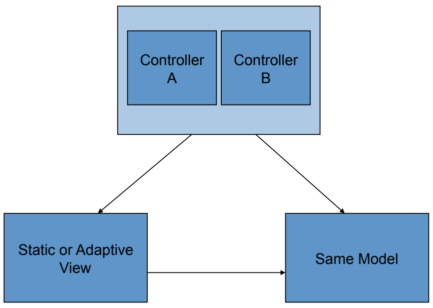


Adaptive Shared View – Street Fighter

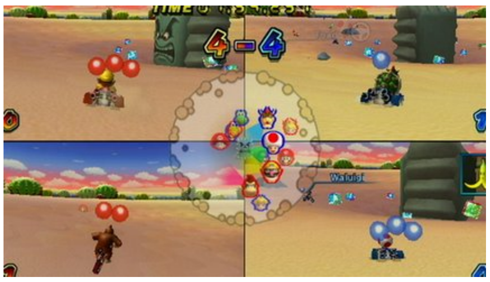


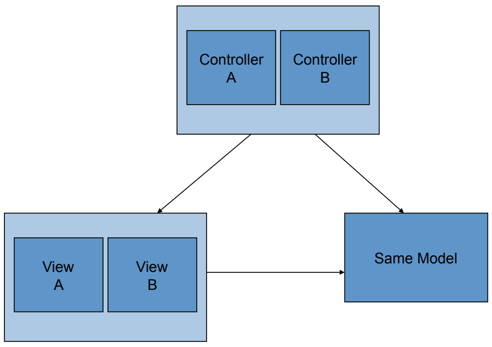
Adaptive Shared View – Street Fighter





Split-Screen View – Mario Kart

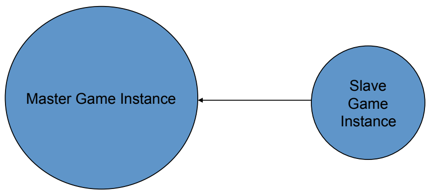


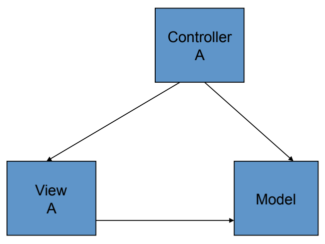


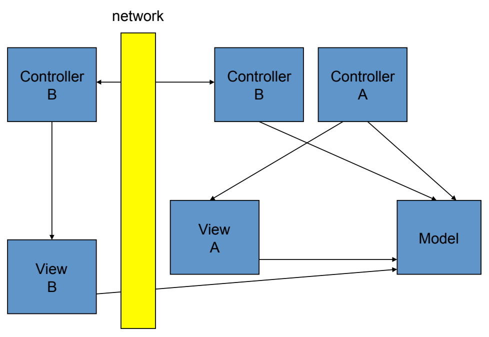
Networked / Online Game Play

- Players are physically separate
- Where is the game?
- **Master** and **slave**
  - Usually two players, local network
- **Dedicated server** and **Clients**
  - Multiple players, local network, internet
- **(Peer-to-peer)**
  - Largely theoretical

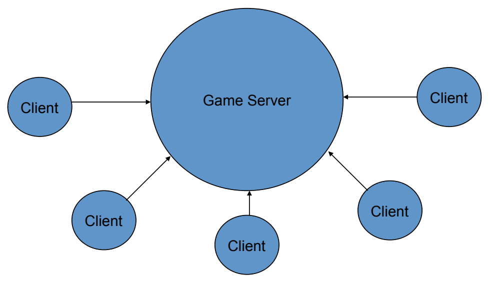
Master and Slave

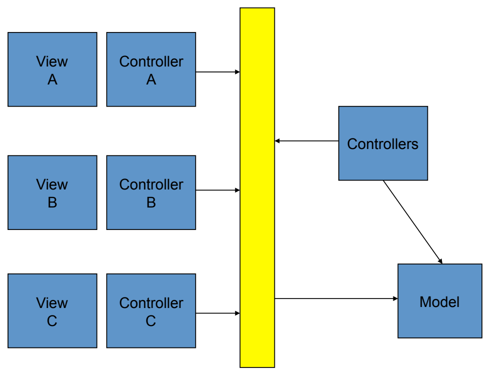




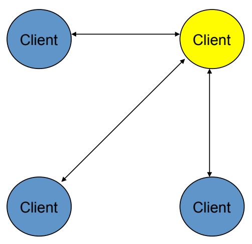


Client Server

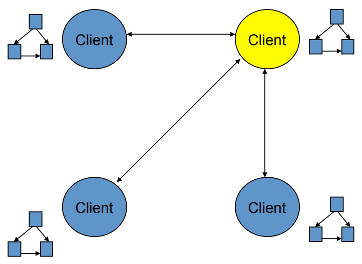




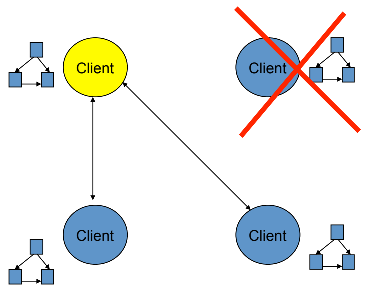
Dynamic Master-Slave

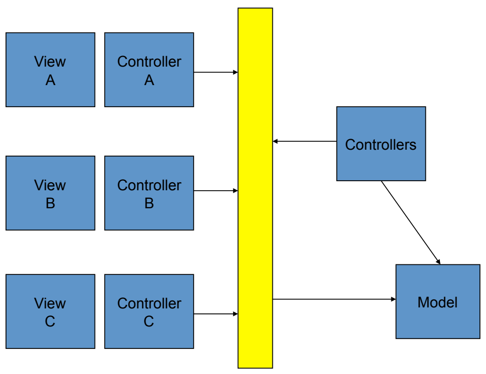


Dynamic Master-Slave



Dynamic Master-Slave

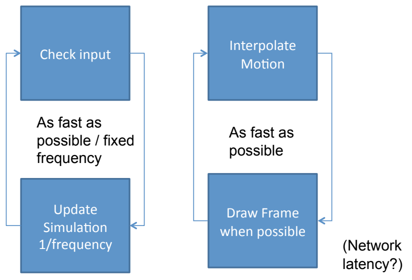




Sadly it's not that simple

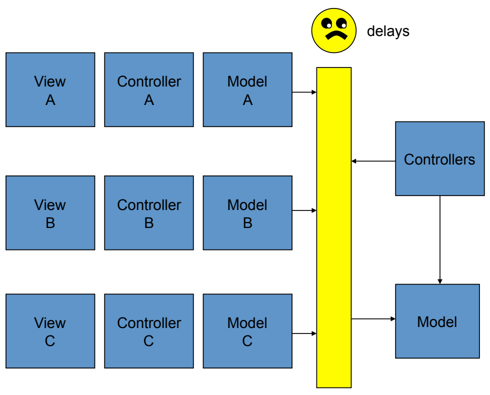
- Game loop runs at 30 Hz
- Renderer redraws at 50-100 Hz
- Each input has to…
  - Travel from the client to the server
  - Be processed by the server
  - Wait for the server loop to update the model
  - Travel from the server to the client
  - Be drawn onscreen
- For several players on the Internet playing a fast paced game
  - This is very slow
  - Requires a lot of bandwidth (each client needs to know about the current state of the model every loop)
  - Packets get lost or delayed or arrive out of order

Decoupling



Local Replication

- Give each client its own replica of the model to use
- The server is the authority on what is happening
- The Client
  - Receives regular snapshots of the server model
    - UDP (fast, but unreliable @~20Hz)
  - Has a local replica of the server's logic
  - Between snapshots uses its local replica of the model to calculate the current state to render the view
  - Tells the server what it is doing, so the server can update the master model



Lag

- Network delays lead to logical Inconsistencies
- A player shoots at another player
  - The player/client thinks they hit
  - The **fire** command takes some time to get to the server
  - The server thinks the player missed
- Two players try to pick up the same gold
  - We both arrived at the gold at the same time
  - Who picked up the gold?
  - We both did, we both saw that we did
  - Who gets to keep it?

Lag

- Implementation needs to have
  - Speed (otherwise it feels slow and jerky)
  - Synchronisation (to avoid logical inconsistencies)
  - Not use too much bandwidth (remember dial-up)
  - Cope with packet loss
- Client prediction
- Entity interpolation
- Lag compensation

Entity Interpolation

- Client is responsible for frame-to-frame movement simulation and rendering
  - X is moving at speed Y with direction Z, so move it a bit
  - 100Hz
- Server is responsible for the bigger picture, the context and the consequences
  - X has picked up the gold, Y hasn't
  - X has fired a gun and hit Y
  - X has pressed UP so start them moving
  - 20Hz

Lag Compensation

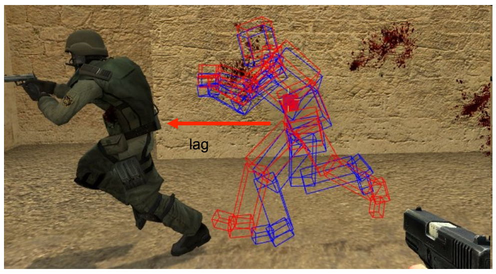
- Dead Reckoning
  - X is moving at speed Y
  - We haven't received a snapshot but need to draw a frame, so move it a bit ourselves
  - It's likely that X will continue to move at speed Y because there's nothing near it
  - …allows smoother graphics than the network allows
- Deltas
  - Only tell the client what has changed, rather than send the full model snapshot
  - Send full snapshot after a network delay so that the client gets back in sync
  - …reduces bandwidth use

Client Prediction

- The player pressed up
- Send the command to the server controller
- While we're waiting for the updated snapshot, start moving anyway
- Rewind if we got it wrong
- …the game feels more responsive than the network allows

Server-side Lag Compensation

- The server keeps a record of current round-trip-time for all the clients (time for a packet to travel from client->server->client)
- Remember where everything was up to a second ago
- When a new command is received, estimate when it was **sent** rather than when it was **received**
- Use a historical snapshot of the model to work out what happened
- …the command is executed hopefully as if there was no lag



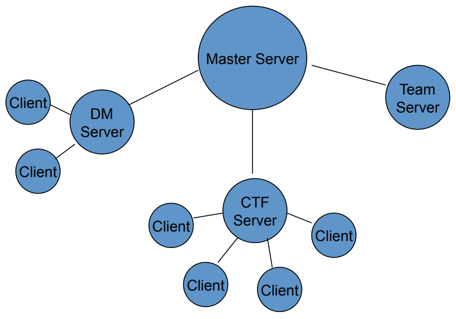
Scalability

- Basic client server model
  - Is reasonable for small numbers of players
- Second Life
  - 15 million registered accounts
  - Average 10000 players at any one time?
- World of Warcraft
  - 11 million paying subscribers

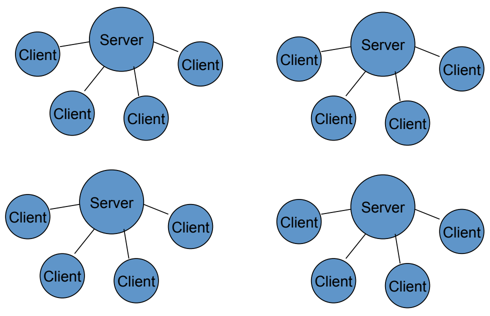
Scalability

- Moderated Awareness
  - Server(s) maintains the complete model for all players
  - Client receives only a subset of the model relevant to the player
  - Where the player is and what is nearby
- Lobbies
  - A master server maintains a browse-able list of all active servers that a player can join
- Instances
  - Multiple identical instances of the server model
  - Each player inhabits only one
  - Create new instances as required
- …can make the game bigger by adding more server hardware, does not affect the clients

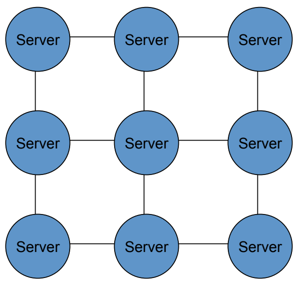
Lobby

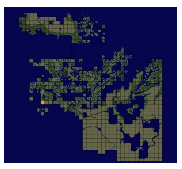


Instances



Second Life Grid





Social Challenges

Different Attitudes to the Game

- Degree of lusory attitude
  - Taking pleasure in playing the game
  - Accepting that rules lead to limitations
  - Accepting the rules as play is an end in itself
- Relationship to the rules
  - How much the player adheres to the rules
    - Operational – the formal rules
    - Implicit – the social rules
  - Acknowledges the authority of the rules
    - Can/should they be broken?
- Interest in winning
  - How much the player wants to win

Types of Players

- Standard Player
  - Possesses lusory attitude
  - Acknowledges authority of the rules
  - Typical interest in winning
- Dedicated Player
- Unsportsmanlike Player
- Cheat
- Spoil-Sport

Types of Players

- Standard Player
- Dedicated Player
  - Extra-zealous lusory attitude
  - Special interest in understanding and mastering the rules
  - Intense interest in winning
- Unsportsmanlike Player
- Cheat
- Spoil-Sport

Types of Players

- Standard Player
- Dedicated Player
- Unsportsmanlike Player
  - Sometimes dedicated, sometimes a cheat
  - Adheres to operational rules, violates implicit rules
  - Intense interest in winning
- Cheat
- Spoil-Sport

Types of Players

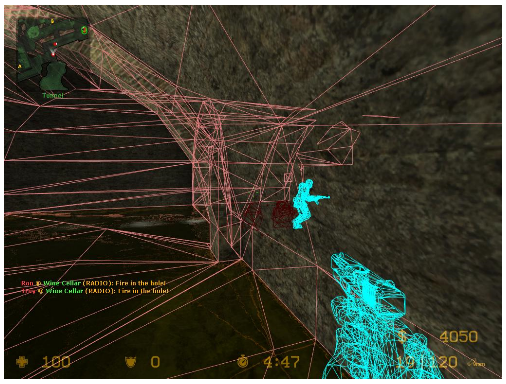
- Standard Player
- Dedicated Player
- Unsportsmanlike Player
- Cheat
  - Pretends to possess lusory attitude
  - Violates operational rules in secret
  - Intense interest in winning
- Spoil-Sport

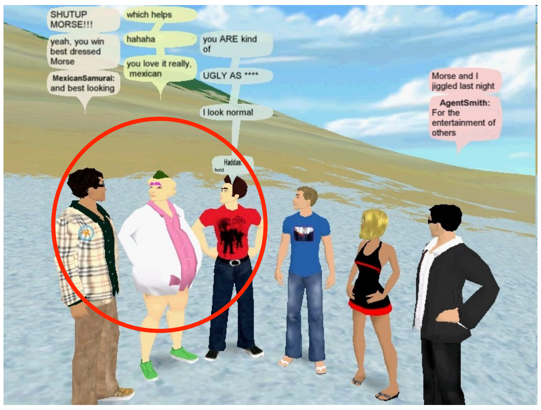
Types of Players

- Standard Player
- Dedicated Player
- Unsportsmanlike Player
- Cheat
- Spoil-Sport / Griefer
  - No pretense about lack of lusory attitude
  - No interest in adhering to any rules
  - No interest in winning

Breaking the Rules

- A big problem in online games
- Making use of dominant strategies in order to win
  - Standard player vs dedicated player
  - Trick jumping, camping, weapon imbalances, short cuts
- Breaking operational rules
  - Hacking the client
  - Using bots to automate play
- Breaking implicit rules
  - Multiple concurrent sessions
  - Killing team members / Spying for the other team
  - Subverting game economy ("gold farming")
  - Disrupting the play of other players


Ron @ Wine Cellar (RADIO): Fire in the hole!
Troy @ Wine Cellar (RADIO): Fire in the hole!

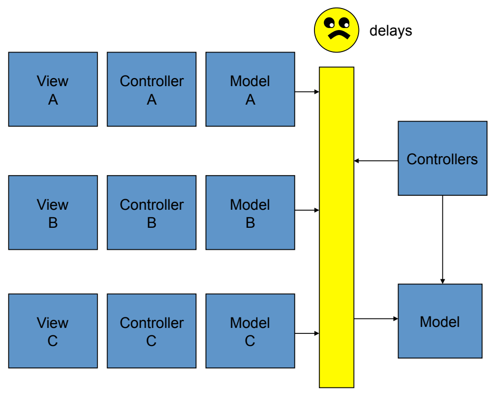

SHUTUP MORSE!!!
yeah, you win best dressed Morse
MexicanSamurai: and best looking
which helps
hahaha
you love it really, mexican
you ARE kind of
UGLY AS ****
I look normal
Haddas: hold
Morse and I jiggled last night
AgentSmith: For the entertainment of others

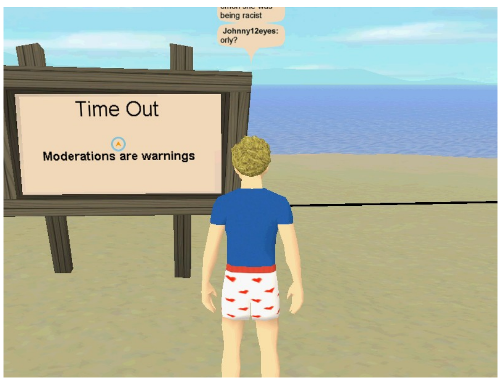
Reducing Rule Breaking

- Sanctioned Cheating
  - Make cheating a form of game play
  - Difficult to maintain meaningful play
- Formalise implicit rules as operational rules
  - Violations are punished within the scope of the game
  - Killing a team-mate = lose a point
  - Gold farming
    - Violates EULA ("operational rule")
    - Account is banned

Reducing Rule Breaking

- Technical Measures
  - Never trust the client
  - Attempt to hinder the use of bots / client hacks
    - PunkBuster / The Warden / Valve Anti-Cheat System et al search the clients memory for known hacks, scripts, behaviours
    - Impossible to completely stop client hacking
    - Occasionally punishes innocent players
  - Ban offenders by email, IP address and MAC address
- Social Measures
  - Player empowerment
    - Allow players to vote out or mute offensive players
  - Employ **moderators** to police the game





References

- http://developer.valvesoftware.com/wiki/Source_Multiplayer_Networking
- http://www.youtube.com/watch?v=0OiamBxxoXA