Networking for Games

CIS 399 – Introduction to Game Programming
Chris GauthierDickey
Summer '05
Why Networking

- There's this thing called the Internet, in case you haven't noticed...
- Everyone wants to play together!
- Networking adds another reason for someone to buy your game
- Interaction between users provides more interesting content than AI alone
Two Main Paradigms

- We commonly see two main paradigms in networking...
  - Client/server architecture
  - Peer-to-peer architecture
- Client/server is more of a centralized solution
- Peer-to-peer is more of a distributed solution
Client/Server Architecture

- The server keeps track of all state
- Clients send updates to the server
  - Server responds with current state of the game
- Clients are 'dumb', except for movement prediction
  - Movement prediction is called 'dead reckoning'
- Clients have to 'roll-back' state whenever the current game state doesn't match their view
Peer-to-Peer Architecture

- Fully distributed architecture
  - Age of Empires was P2P (google it for description of the protocol)
- Each peer keeps state of the game
- Each peer sends update to all other peers
- Each peer runs an algorithm to make sure that state remains consistent
  - AoE sent periodic hashes of the game state to all other peers and killed the game if it didn't match
P2P vs. Client/Server

- Why choose one over the other?
  - P2P eliminates the need to write a server
  - P2P simplifies some of the game logic
  - P2P requires a cheat-proof event-ordering protocol
  - Client/Server eliminates many kinds of cheats
    - Lookahead, timestamp, suppressed-update
  - Client/Server requires a trusted server, or perhaps a server farm in the case of MMOGs
Socket Programming

- Whichever way one chooses, one still needs to write the socket code.
  - Windows has its own high-performance networking way to program as sockets have some performance issues

- We can use TCP or UDP
  - TCP is totally ordered, reliable transport
  - UDP is best-effort