GAME 311
Network Programming

Chapter 1

Overview of Networked Games

Lecture 3
Objectives

- **A brief history of multiplayer games**
  - Discussion of the first networked multiplayer games

- **Starsiege: Tribes**
  - How did the classic first-person shooter architect its network model?

- **Age of Empires**
  - How do real-time strategy games differ in their networking requirements, and how did that influence the architecture of Age of Empires?

Local Multiplayer Games

- Some of the earliest video games featured **local multiplayer**, meaning multiple players could play on one machine.

- Examples:
  - Tennis for Two (1958)
  - Spacewar! (1962)

- Local multiplayer games are programmed much like single-player games.

Early Networked Games

- Most of the earliest networked multiplayer games were run on mainframes at universities.
- Examples:
  - Empire (1973)
    - a turn-based strategy game on the PLATO network.
  - Maze War (1973)
    - a first-person game
- Unclear which of these two games was first.

Early Networked Games

- On personal computers, the first networked games used **serial ports**, which can transmit 1bit at a time.
- These games started appearing in the late 70s and early 80s.
- Because most computers had only one (or two) serial ports, several computers would have to be **daisy chained** together to have more than two players in a game.
  - Ring of connected PC's.

Multi-User Dungeons

- A **multi-user dungeon** (MUD) is a (usually) text-based multiplayer role-playing game.
- First MUD was MUD (1978), developed by Rob Trushaw at Essex University.
- MUDs were very popular on **bulletin board systems**, where personal computer users with modems could connect and interact with other players in the game.

Local-Area Network Games

- The proliferation of the Ethernet protocol led to a large increase of **local-area network** (LAN) games
- Though not the first such game, the first big hit was the first-person shooter Doom (1993).
  - Supported up to 4 players
- In recent years, most networked games no longer support LAN play.

Online Games

- In an **online game**, players connect to a network with geographically distant computers.
- Today, this term is synonymous with an Internet game, though there have been other such networks in the past.
- Examples:
  - Quake (1996), id Software's follow-up to Doom
  - Unreal (1998), Epic Game's first major multiplayer game

Online Games

- Due to the distances traveled, online games have to worry about **latency**
  - the amount of time it takes data to travel over the network.

- Today, most players also expect there to be an online gamer service such as Xbox Live, Steam, or PlayStation Network.

Massively Multiplayer Online Games

- Most networked games support a small number of players; numbers between 4 and 32 are common.
- A **massively multiplayer online game** supports hundreds if not thousands of players.
- Examples:
  - Habitat (1986)
  - Ultima Online (1997)
  - Everquest (1999)
  - World of Warcraft (2004)

Mobile Networked Games

- Mobile networked games typically are asynchronous, meaning they are turn-based games that do not require real-time transmission of turn data.
- **Asynchronous** games have existed for a long time before mobile games; for example, many BBS games were asynchronous.

- Examples of networked mobile:
  - **Words with Friends (2009)**
  - Hearthstone: Heroes of Warcraft (2014)

Starsiege: Tribes (1998)

Starsiege: Tribes (1998)

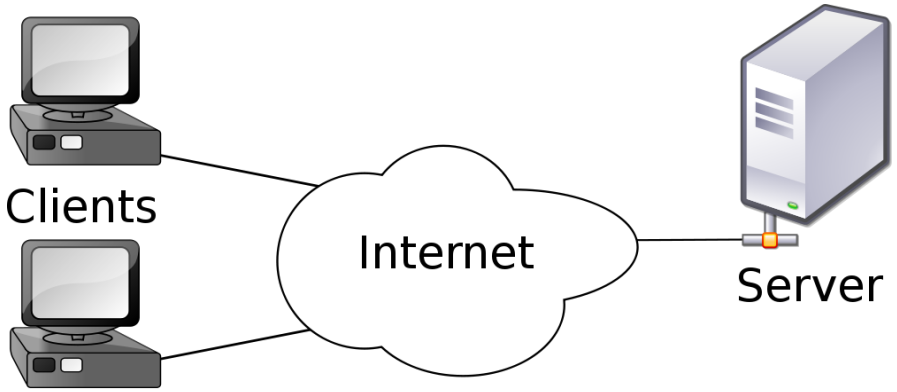
- Fast-paced first-person shooter from the late 90s.
- At the time, most players had Internet access with speeds maxing out at 56.6 or 28.8 kbps, which presented unique challenges.
- Many networked games still use models similar to the one used by Tribes.

Tribes: Reliability

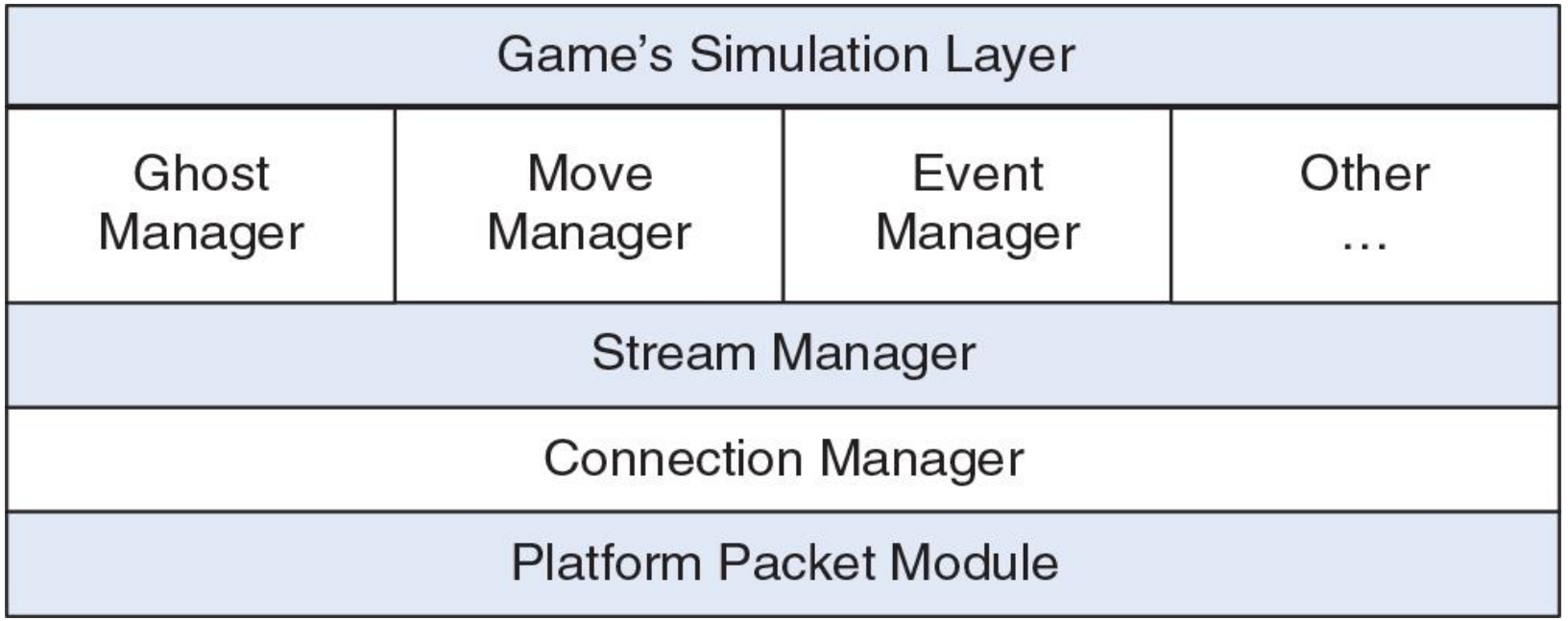
- *Tribes* uses an ***unreliable*** protocol, meaning that data is not guaranteed to be delivered to the destination.
- This presents challenges because some game data may be so important that it needs to be reliable.
- Developers split up data into four categories:
  - **Nonguaranteed**: Data that is nonessential
  - **Guaranteed**: Guarantees both the arrival and ordering of data, for data deemed important
  - **"Most recent state"**: For things like position, where only the newest data matters
  - **Guaranteed quickest**: Will try to arrive ASAP

Tribes: Client-Server

- Tribes uses a **client-server** model in which each player connects to a central server.
- Bandwidth requirements:
  - Each player requires a constant amount of bandwidth, regardless of the number of players.
  - Server requires $O(n)$ bandwidth, where n is the number of players.



Tribes: Layer Cake



Tribes: Platform Packet Module

- A **packet** is a formatted set of data sent over a network.
- The **platform packet module** is the lowest layer of the Tribes model, and is platform specific.
- It is a wrapper for standard socket APIs that know how to construct and send various packet formats.

Tribes: Connection Manager

- The connection manager abstracts connection between two computers.
- Delivery is *not* guaranteed, but **delivery status notification** is (the status of any request sent to connection manager can be verified).

Tribes: Stream Manager

- Sends data to the connection manager
- Determines the maximum rate of transmission, based on the user's connection
- Prioritizes transmission requests from the higher-level systems

Tribes: Event Manager

- Maintains a queue of events generated from higher-level systems.
  - For example, a player firing a weapon might generate a "player-fired" event.
- Events are sent from a client to the server, and then the server processes them.
  - Manages when to send events.
- Tracks the transmission requests for "**reliable**" events; so if necessary, events can be re-sent.

Tribes: Ghost Manager

- **Replicates** or "ghosts" dynamic objects to clients.
- The server sends to each client the information it needs to know about various game objects.

Tribes: Ghost Manager

- The game simulation layer determines how important objects are, in two priority levels:
  - Need to know: Client must know the status of these game objects
  - Should know: Ideally, the client should know the status of these game objects

Tribes: Move Manager

- Transmits player movement data as quickly as possible.
- Example of "most recent" data.
- The move manager data is high priority, because knowing the true position of other players and objects is critical to an enjoyable networked game experience.

Age of Empires (1997)

- A popular real-time strategy game (RTS).

Age of Empires (1997)

- As with Tribes, AoE had to be concerned with bandwidth.
- Implements a **deterministic lockstep model**:
  - All computers are connected to each other
    - peer to peer.
  - Each peer concurrently performs a *deterministic* simulation of the game.
    - *Lockstep* because the system guarantees synchronization between peers.
- Deterministic lockstep is still used in many RTS games, such as StarCraft II (2010).

Age of Empires: Challenges

- In Tribes, each client rarely needs to know about more than 20-30 objects at once.
- In Age of Empires, each player can control 50+ units.
  - A massive battle among 8 players means that there's nearly 400 units to keep track of.
  - Impractical to send all the unit object data.

Solution: Instead, send the commands issued by players, and independently execute these commands on each peer.

Age of Empires Turn Timers

- The **turn timer** queues up commands for a fixed duration (200 ms by default in Age of Empires).
  - Once the turn finishes, the commands are sent to all the peers.
  - Scheduled for execution two turns later.
  - So commands issued on turn *x* aren't executed until turn *x* + 2.
  - This adds to input lag, the amount of time it takes for a player's action to appear onscreen, but this is acceptable in an RTS.

Age of Empires: Synchronization

- It's extremely important that each peer simulates the game in an identical manner.
- Pseudo-random number generators (PRNGs) must be **seeded** consistently across all clients.
- It's also a requirement to validate the synchronization, in the event of either a programming error or a player attempting to cheat.

Chapter 1
Summary

- **A brief history of multiplayer games**
  - Discussed the first networked multiplayer games
- **Starsiege: Tribes**
  - Discussed How the classic first-person shooter created its network model
- **Age of Empires**
  - Discussed How real-time strategy games differ in their networking requirements, and it's influences on the architecture of Age of Empires