Seamless Object Migration for Massively Multiplayer Online Games

Te-Yu Chen

University of California, Irvine

teyuc@uci.edu

Motivation

Two main methods currently used for distributing work to multiple servers in a massively multiplayer online game (MMOG)

User Based Load Distribution

- Assign players to servers when they login.
- The assignments do not change during the whole play session.
- **Disadvantage**
  - Players on different servers can never interact with each others.

Location Based Load Distribution

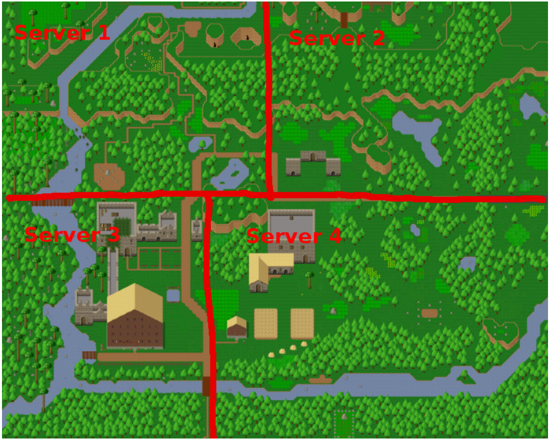
- Divide the game into several areas, each managed by a server.
- When players move around, they change their servers dynamically.
- **Disadvantage**
  - Long waiting time when moving from one server to another.
  - Players in different areas still can't interact with each others.

Goals

- Easy to use tools to do load distribution for game developers.
- Players can interact with each others without any limitation.
- Players do not need to wait when moving from one server to another.

Goals

- Easy to use tools to do load distribution for game developers.
- Players can interact with each others without any limitation.
- Players do not need to wait when moving from one server to another.

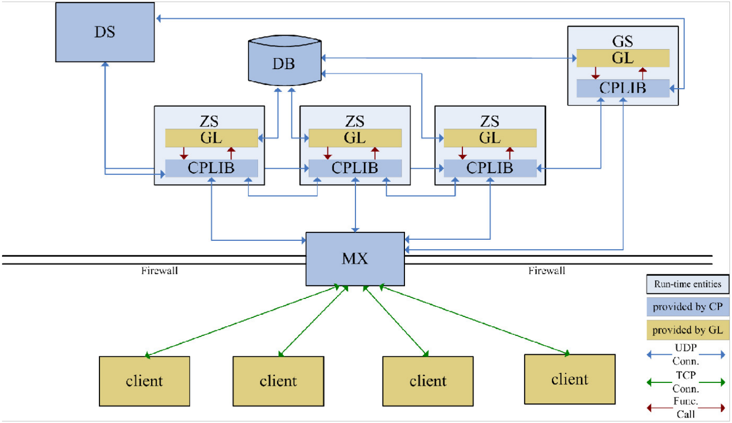


Related Works

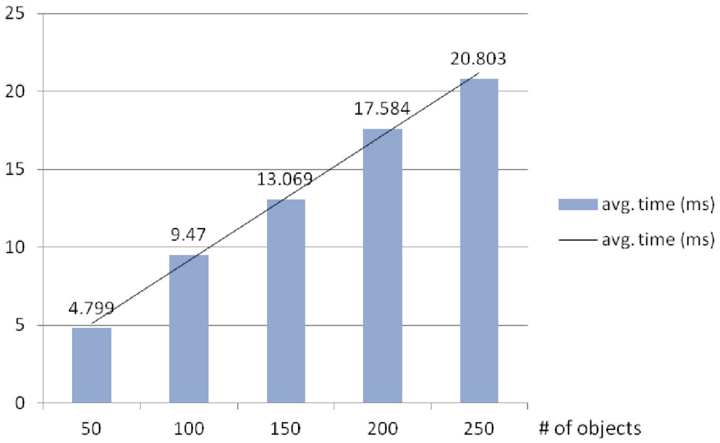
- D. Meilander, F. Glinka, S. Gorlatchand, L. Lin, W. Zhang, X. Liao, “*Bringing mobile online games to clouds*”, In Proc. of Computer Communications Workshops (INFOCOM WKSHPS), Toronto, 2014.
- F. Lu, K. Storey, G. Morgan, “*Message Oriented Middleware Services for Networked Games*”, In Proc. of the I3D 2005. ACM Symposium on Interactive 3D Graphics and Games, Washington DC, 2005.
- A. Bharambe, J. Pang, S. Seshan, “*A Distributed Architecture for Interactive Multiplayer Games*”, In CMU CS Technical Report Number CMU-CS-05-112, 2005.
- C. Greenhalgh, S. Benford, “*MASSIVE: a distributed virtual reality system incorporating spatial trading*”, In Proc. International Conference on distributed computing systems (DCS 95), Vancouver, 1995.

System Structure

- **Game Server**: central manager
- **Zone Servers**: calculating game states
- **Directory Server**: map ip address/port to server ID
- **Message Exchanger**: direct messages to/from clients



Simulation Results


# of objects
avg. time (ms)
avg. time (ms)