World2World

An API for distributed simulations

Example of a tool for creating DVE

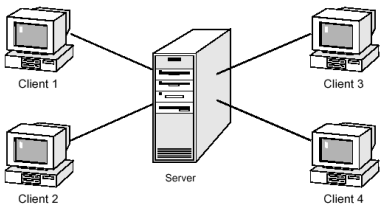
- DVE - Distributed Virtual Environments
- World2World from Sense8
  - What is it?
  - How is it built?
  - How do you use it?

World2World

- Client/Server based networked solution for interactive simulations
- Client is integrated into WorldToolKit and WorldUp
- Complies to the object/property/event paradigm of Wxx
- A simulation is composed of objects (e.g.. a car) and object properties.
- A change in a property is called an event.
- To allow several users participate in the same simulation they need to receive certain updates made by other users.
- Properties needs to be shared!

Why client/server?

- Multicast
  - Efficient on dedicated networks when every machine communicates its data to the network and listens to everything put on the network
  - More importantly, forcing every participant to receive every update for every other participant severely limits the number of potential participants, especially over low-bandwidth connections.
- Client/server
  - A client/server approach resolves the problem by creating a central point of communication through which all updates are routed.
  - Every participant sends update information to a central server and the server routes this information appropriately.



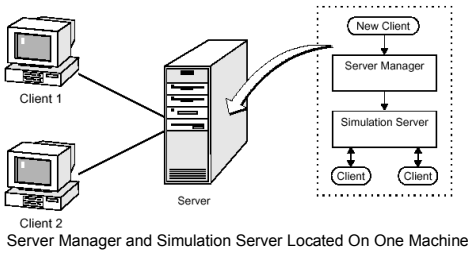
Typical client/server setup

Servers

- There are two layers of servers
  - The Server Manager
    - manages the initial point of contact for a client connecting to a simulation. Only one Server Manager can exist in a World2World server system
  - The Simulation Servers
    - store and broker access to shared simulation data. You can have several Simulation Servers in your World2World server system

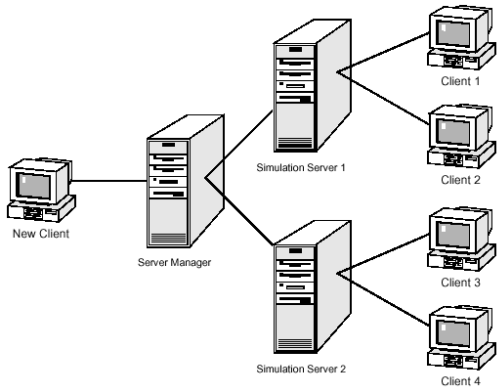
Servers

- A new client goes through the Server Manager upon connection,then interacts directly with the Simulation Server (bypassing the Server Manager) for the duration of the connection.
- The Server Manager and Simulation Server can be located on the same or different machines.
  - This makes the client administration transparent for the simulation.
  - That is, simulations that are currently running will not be disrupted when a new client connects to the Server Manager.



Server Manager and Simulation Server Located On One Machine

Servers



Server Manager and Simulation Server Located On Different Machines

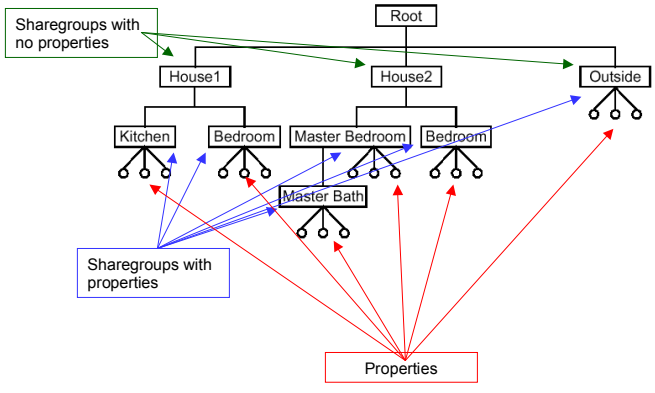
- W2W is written on top of UDP to avoid some unnecessary overhead imposed by TCP
- W2W determines a global simulation time
  - System clocks on client machines used in a simulation is probably not synchronized
  - This synchronized time enables you to:
    - set the value of a property at a specified time and have that time be in synch for all clients
    - using the timestamps from each property change event
    - incorporate *dead reckoning algorithms* into your simulation

Property - Information holder

- Properties
  - Can be one of:
    - WTINT, WTUINT, WTFLOAT, WTDOUBLE. WTP2, WTP3, WTQ, WTSTRING, WTPOINTER
  - Clients can share properties
  - A client requests an interest in a property
    - Only then the client will be notified if changes occur to the property
  - If the property changes, the server makes sure that all clients that are interested is notified
  - The changed property will eventually result in a callback call at the clients side of things.

Sharegroups - Structuring properties

- Sharegroups is a way of structuring properties in a hierarchical way



Sharegroups - Structuring properties

- Sharegroups
  - Clients can share sharegroups
  - A client requests an interest in a sharegroup
  - If any of the property in the sharegroup (or in the sub-tree), the server makes sure that all clients that are interested (in that property) is notified

Locks - Ensuring your alone!

- What if you are sharing properties with another user?
- Who can move it, delete it, …?
- The solution is to lock the data:
  - Only one user can lock a sharegroup or a property.
  - When property/sg. is locked by a user, no-one else can change it until he releases the lock.

Persistency

- What if you want parts of a simulation to last longer then the execution time of the clients?
- Sharegroups and properties can be created using PERSISTENCY flag.
- Making a sharegroup persistent will also make all the properties in that sharegroup persistent.

Network connection

- To make the simulation distributed a network connection has to be established
  - Synchronous mode
    - Client has to wait while requests are being fulfilled by the simulation server
  - Asynchronous (default)
    - Client can continue processing other tasks while requests are fulfilled and get a notification later when the request is finished.

Network connection

- Update rate
  - Keep as low as possible, although a low update rate may require you to employ dead-reckoning algorithms.
  - Possible to update only when changes occur.
- Connection callbacks
  - Changes to the simulation server will also generate events that can be handled by clients
    - E.g. a user connects/leaves, creating/deleting sharegroups ...

Using W2W

- Creating a new workspace in MSDEV6.0
  1. New workspace.
  2. Set C/C++ > Code Generation > Use run-time library to: *Debug* – Debug Multithreaded DLL *Release* - Multithreaded DLL
  3. Set C/C++ > Preprocessor > Additional include directories to: **c:\sense8\w2w\include**
  4. Set Link >Input > Additional library directories to: **c:\sense8\w2w\lib**
  5. Add Link > Input > Object/library modules: **wtkos.lib, w2w.lib**
  6. Set Link > Ignore libraries to: **libc.lib,libcmt.lib**
- In your code:
  - #include <w2w.h>
  - #include <wtkos.h>
- Check out the demos in c:\sense8\w2w\