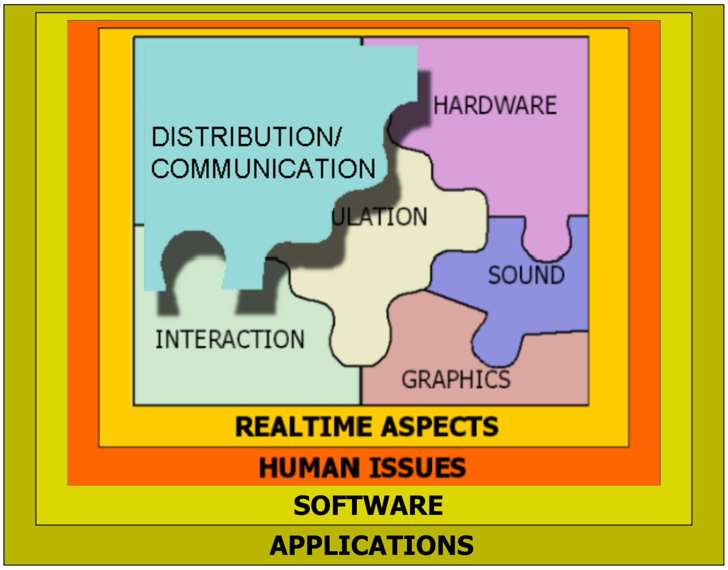
# Networked Virtual Environments






# Networked Virtual Environments

- Net-VE
  - Software system in which multiple users interact with each other in real-time.
  - Common features
    - A shared sense of space
      - Illusion if being located in the same place, same room, building, …
    - A shared sense of presence
      - Entering the net-VE each participant can see avatars of other participants.
    - A shared sense of time
      - Real-time interaction
    - A way to communicate
      - By gestures, by typed text, by voice…
    - A way to share
      - Share the VE in its full sense
      - Pick up, move and manipulate objects in the VE

## Net VE

- Net-VE:s are:
  - Distributed systems
    - Contend with managing network resources, data loss, network failure and concurrency
  - Graphical applications
    - Smooth, real-time display frame rates.
  - Interactive applications



## Challenges

- Network bandwidth
  - 14.4 Modem – Internet2
  - More users – more data to be sent.
- Heterogeneity
  - Different equipment at different locations
  - Reducing to ”lowest common denominator”
  - Take use if all available resources
    - Fair play?
- Distributed interaction
  - Mask artifacts that might arise due to the distributed nature
  - Messaging required to exchange information within the net-VE.
  - Different messages may incur different delays.
  - Collision detection, agreement, ownership
- Real-time system design and resource management
  - Every CPU second has to be accounted for

## Challenges

- Failure management
  - System stop
    - Failure may cause the entire net-VE to terminate.
  - System closure
    - Prevent new user to arrive
  - System hindrance
    - Degrade of the system. E.g.. weather server breaks down, no weather information.
  - System continuance
    - A failure might have almost no noticeable affect on the net-VE.
    - Redundant servers, …

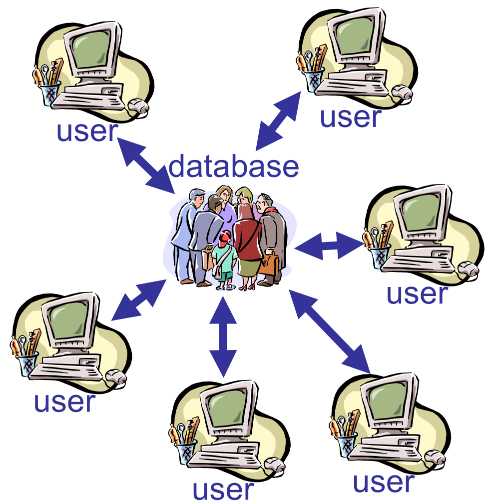


## Challenges

- Scalability
  - Different ways of measure it
    - Number of *entities* simultaneously participate in the system
    - Number of *hosts* connected to the net-VE.
    - Physical distance between participants in the net-VE.
  - Depends on
    - Network capacity
    - Processor capability
    - Rendering capacity
    - Capacity of shared servers
- Deployment and configuration
  - How do we share the net-VE
    - Large and monolithic, inappropriate for downloading
    - Small core, dynamically loading of content more suitable.
  - Security

# Architectures

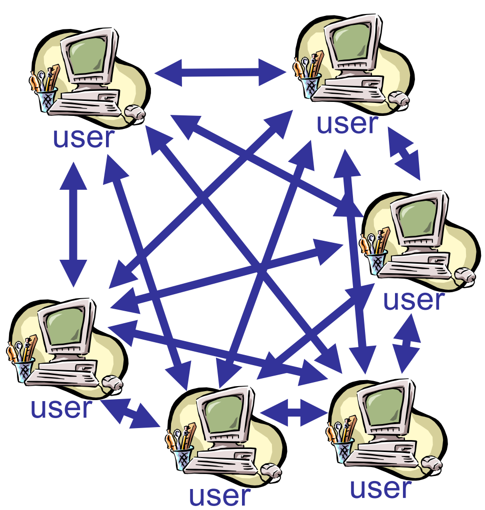
- Centralized model
  - Client-server
  - One computer (database) collects all data and sends updates to the users.
  - Simple structure.
  - Not scalable, the database is the bottleneck.





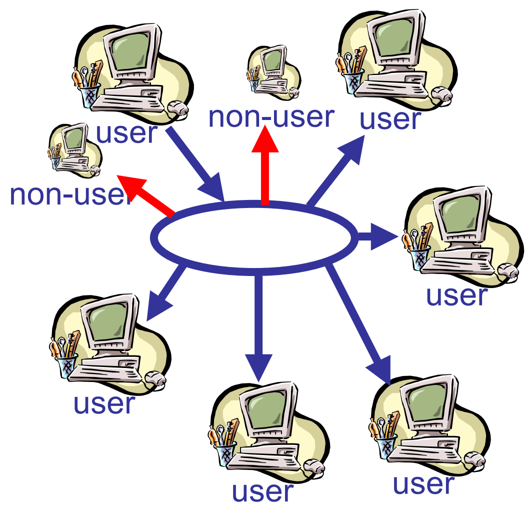

# Architectures

- Distributed model
  - Peer-to-Peer
  - Each user maintains its own copy of the database.
  - Updates are send to other users.
  - Not scalable, the network is the bottleneck.



## Architectures

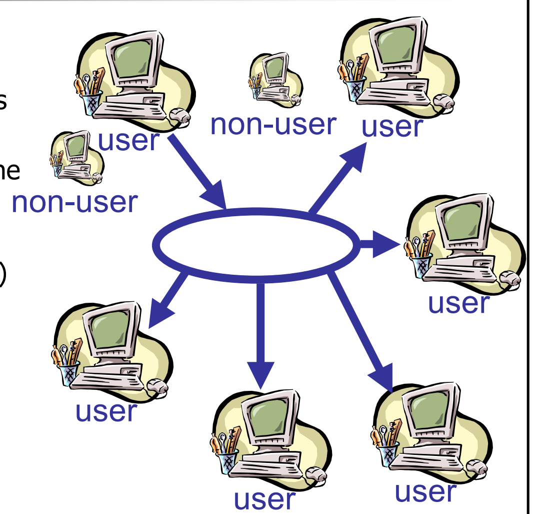
- Broadcast model
  - For n users, instead of n-1 messages only one message.
  - The message is send to all users (and non-users).
  - Not selective.
  - Quake I – Flooded LAN with packages.






## Architectures

- Multicast model
  - A multicast group includes all users.
  - The message is send to the group and therefore to all users (group members).
  - Non-users (non-members) do not receive messages



# How to avoid bottleneck

- Better communication models - reduce number of connections and messages.
- Better database models – distributed databases.
- Better decision making – make it distributed but any given decision is made in only one place.
- …



# Dead Reckoning

- Problem
  - Entities on a distributed system must have an "accurate" image of each other's location
  - Updating each other for all entity movements requires a high communication overhead
- Solution
  - Formalized in DIS 1993.
  - Both local and remote entities maintain an "approximate" image of the entity
  - If local entity detects that it has deviated from expected location by more than an acceptable range, it then updates its location to the remote entity
- An algorithm to reduce number of messages.
- Instead of sending frequent updates on object's position, it is calculated locally using a last-known velocity and position.

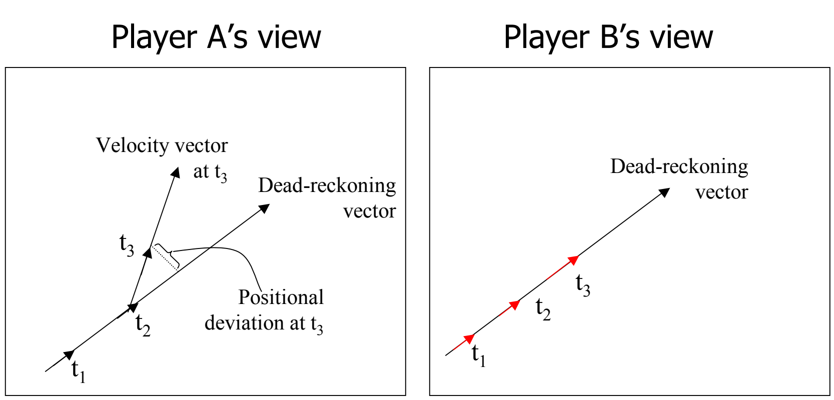
# Dead Reckoning

- A player controls his or her own objects – *Live/player objects.*
- Objects controlled by other players are know as *ghosts*.
- A player use the ”game application” to control his or her objects. E.g. Steering a car.
- And a dead-reckoning algorithm to control the ghost objects.
- Player A sends his current velocity *v* and acceleration *a* to player B (and vice versa).
- If A do not change *a* or *v*, then B can calculate A:s future position. B uses dr. to control A:s ghost object.
- When A decides that A:s position has his velocity has changed by an amount sufficiently significant to send a message to B.
- A makes this decision from running the same DR. on his live object and measuring the difference between the DR:s position and the actual pos.



# Dead Reckoning

- All players simulate their competitors´live entities by extrapolation.



## Dead Reckoning

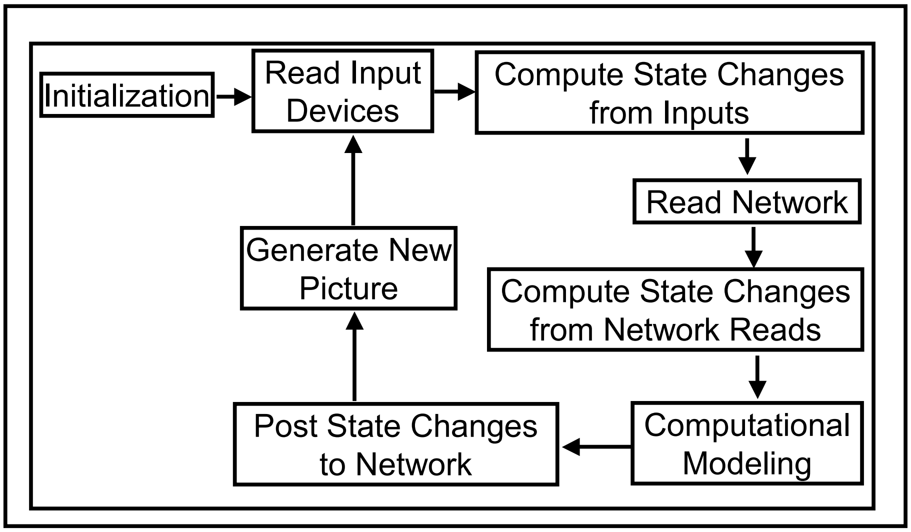
- Problem
  - Trade-off between accuracy and number of messages sent
    - Large deviation threshold – jerky movement in ghosts
    - Small deviation threshold - more messages will be sent.
  - What if player A destroys a bridge and stops in front of it?
    - B will extrapolate the position over the non-existing bridge before A have transmitted the new position (and also transmitted information of the bridge).
  - Sensitive to sudden changes in velocity acceleration.



## Heartbeat

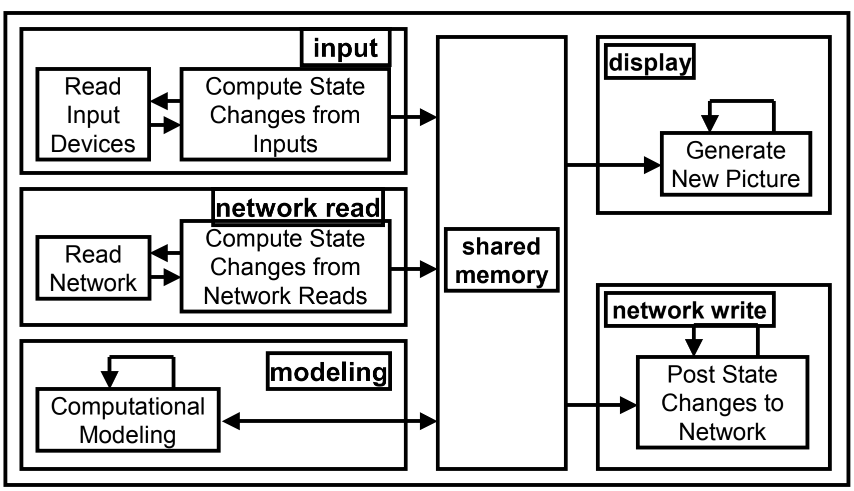
- Some systems (DIS) uses Heartbeat:
  - Each user periodically sends a a message called a **heartbeat** informing everyone of its status.
  - Helps recovery from lost messages (network reliability issues).
  - Helps users who just joined.

# Single threaded design



# Multithreaded design




 ©Anders Backman, Dept. Computing Science, VR00 - Net-VE

## Sample systems

- NPSNET (86-)
  - Made as a part of a CG class.
  - Implemented on a SIG IRIS-2400 system (500 polygons/sec).
- Distributed Interactive Simulation (DIS) (89-)
  - IEEE 1278 standard.
  - Contained
    - Object-event architechture
      - Objects
        - Vehicles, peoples
      - Events
        - Messages
    - Autonomous simulation nodes.
      - Each player, vehicle, … is responsible to place messages representing their current state.
    - Dead reckoning algorithms
  - Uses Protocol Data Unit (PDU) for exchanging messages.
    - Entity state, fire, detonation and collision



## Sample systems

- NPSNET-1, -2, -3 and NPS-Stealth (90-)
  - 1990, U.S. Army project.
- NPSNET-IV (93-)
  - Built upon Performer.
  - A player can be a fully articulated human, air vehicle, ..
- PARADISE 93- (Performance Architechture for Advanced Distributed Interactive Simulation Environments)
  - IP multicast (simulates as Multicast did not exist at the time).
  - Transmits fewer packages for objects moving slow
  - Position History-Based Dead Reckogning (PDBDR)

# Sample systems

- DIVE 93-
  - SICS (Swedish Institute of Computer Science)
  - Distributed Virtual Environments
  - Simulating a large shared memory over a network.
  - Uses reliable multicast protocol and concurrency control via distrited locking mechanism.
  - Does not scale over 32 participants.
  - Solving problems of collaboration and interaction.
- BrickNet 94-
  - No replicated database model.
  - Partitions the Virtual World among the various VE clients.
  - Request for objects/communication is mediated by servers.
  - Interactions on distant objects are accomplished by an object-request broker on a server, a server that knows which client owns the distant object.



# Sample systems

- BAMBOO (98-)
  - Dynamically configurable (during run-time)
  - Based on plug-ins, dynamically loaded from the web.
    1. Eventually there will be a persistent VE shared simultaneously by billions of participants
    2. There can never be a global reboot
    3. All modifications must happen on the fly
- VERSE (00-)
  - Interactive Studio, Stockholm
  - Client-server

# References

- *Networked Virtual Environments. Design and Implementation*, Singhal, Zyda, Addison-Wesley, 1999
- *3D Games, Real-time rendering and Software Technology*, Watt, Policarpo, Addison-Wesley, 2000
- *Developing Shared Virtual Environments,* Course 42, Siggraph2000, New Orleans.