SIMNET

- Technical challenges
  - how to fabricate high-quality, low-cost simulators
  - how to network them together to create a consistent battlefield
- Testbed
  - 11 sites with 50–100 simulators at each site
  - a simulator is the portal to the synthetic environment
  - participants can interact/play with others
  - play was unscripted free play
  - confined to the chain of command

SIMNET NSA

Basic components

i. An object-event architecture
ii. A notion of autonomous simulator nodes
iii. An embedded set of predictive modelling algorithms (i.e., 'dead reckoning')

i. Object-Event Architecture

- Models the world as a collection of *objects*
  - vehicles and weapon systems that can interact
  - a single object is usually managed by a single host
  - 'selective functional fidelity'
- Models interactions between objects as a collection of *events*
  - messages indicating a change in the world or object state
- The basic terrain and structures are separate from the collection of objects
  - if the structure can be destroyed then it has to be reclassified as an object, whose state is continually transmitted onto the network

ii. Autonomous Simulator Nodes

- Individual players, vehicles, and weapon systems on the network are *responsible* for transmitting *accurately* their current state
- Autonomous nodes do not interact with the recipients by any other way
- Recipients are responsible for
  - receiving state change information
  - making appropriate changes to their local model of the world
- Lack of a central server
  - single point failures do not crash the whole simulation
  - players can join and leave at any time (persistency)
- Each node is responsible for one or more objects
  - the node has to send update packets to the network whenever its objects have changed enough to notify the other nodes of the change
  - a 'heartbeat' message, usually every 5 seconds

iii. Predictive Modelling Algorithms 1 (3)

- An embedded and well-defined set of predictive modelling algorithms called *dead reckoning*
- Originally each change was reported
  - for some objects packets were generated as fast as possible (i.e., at frame rate)
  - flooded the network and overloaded the CPUs
- Objects and ghosts paradigm to reduce the packet traffic:
  - objects place packets onto the network only when their home node determines that the others can *no longer predict* their state within a certain threshold
  - the other nodes maintain 'ghost' copies
  - predict the current location using the last known direction, velocity, and location

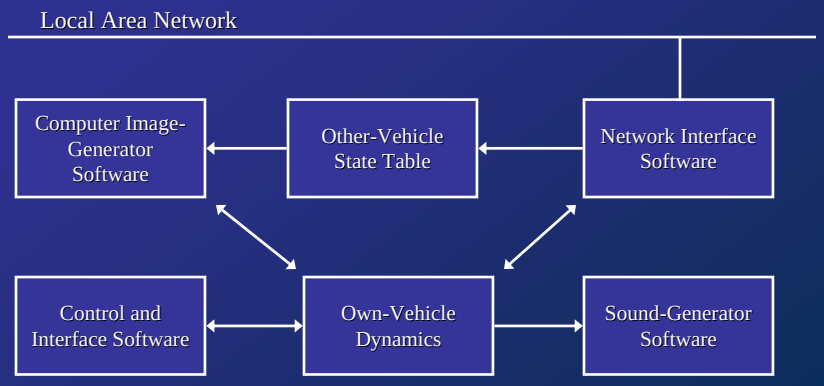
iii. Predictive Modelling Algorithms 2 (3)

- When new packets arrive, the ghost object is seen to move over or back slightly
- Larger thresholds mean fewer packets but larger jumps
- Helps also to cope with packet losses
- Current properties are good for prediction if the object does not move wildly
  - if it does, another packet is most likely to be received soon
- The heartbeat packets also update the object state

iii. Predictive Modelling Algorithms 3 (3)

- Average SIMNET packet rates:
  - 1 per second for slow-moving ground vehicles
  - 3 per second for air vehicles
- Some of the information is relatively static
  - single bit for indicating whether the vehicle is stationary
  - if stationary, the ghost computations are turned off
- Other packets
  - fire: a weapon has been launced
  - indirect fire: a ballistic weapon has been launced
  - collision: a vehicle hits an object
  - impact: a weapon hits an object

SIMNET Major Software Modules



Distributed Interactive Simulation (DIS)

- SIMNET protocol was designed to satisfy a contract with the U.S. government
  - did not define the NSA for general purpose simulation
  - lack of documentation
- DIS was an attempt to formally generalize and extend the SIMNET protocol
- First version of the IEEE standard for DIS appeared 1993
- Goals
  - to allow any type of player, on any type of machine
  - to achieve larger simulations

DIS NSA

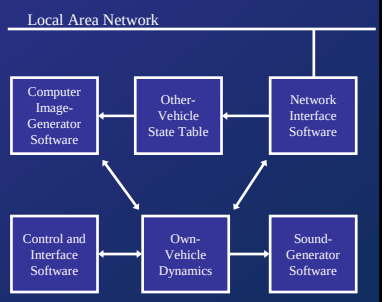
- Derived from SIMNET
  - object-event architecture
  - autonomous distributed simulation nodes
  - predictive modeling algorithms
- Covers more simulation requirements
- Protocol data unit (PDU)
  - determine when each vehicle (node) should issue a PDU
  - the DIS standard defines 27 different PDUs
  - only 4 of them interact with the environment
    - entity state, fire, detonation, and collision
  - the rest of the defined PDUs
    - simulation control, electronic emanations, and supporting actions
    - not supported and disregarded by most DIS applications

Issuing PDUs

- The vehicle's node is responsible of issuing PDUs
  - entity state PDU
    - when position, orientation, velocity changes sufficiently (i.e., others cannot accurately predict the position any more)
    - as a heartbeat if the time threshold (5 seconds) is reached after the last entity state PDU
  - fire PDU
  - detonation PDU
    - a fired projectile explodes
    - node's vehicle has died (death self-determination)
  - collision PDU
    - vehicle has collided with something
    - detection is left up to the individual node

Receiving PDUs

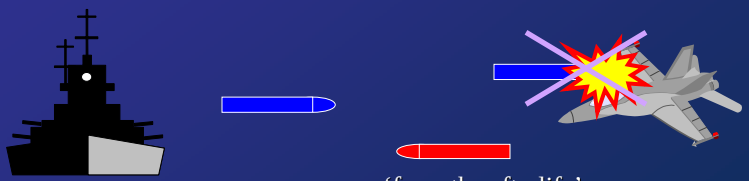
- The recipient is responsible for
  - receiving the PDU
  - changing the appropriate state tables
  - updating the display



  - entity state PDU
    - update the other-vehicle state table
  - fire PDU
    - create a new vehicle
  - detonation PDU
    - compute the effects
    - change the other-vehicle and own-vehicle state tables
  - collision PDU
    - process similarly

Lost PDUs 1 (2)

- Packets are sent via unreliable UDP broadcast
- State tables may differ among the hosts
- Lost detonation PDU



'from the afterlife'

Lost PDUs 2 (2)

- Lost entity state PDU
  - not a big problem
  - larger jumps on the display
- Lost fire PDU
  - receive entity state PDU for which no ghost entry exists
- Lost collision PDU
  - continue to display a vehicle as live
  - next heartbeat packet solves the situation

The Fully Distributed, Heterogeneous Nature of DIS

- Any computer that reads/writes PDUs and manages the state of those PDUs can participate a DIS environment
- The virtual environment can include
  - virtual players (humans at computer consoles)
  - constructive players (computer-driven players)
  - live players (actual weapon systems)
- Problem of the advantages of the low-end machines
  - the less details in the scenery, the better visuality
- Problems with modelling
  - dynamic terrain
    - soil movement
  - environmental effects
    - weather, smoke, dust,…

Additional Properties

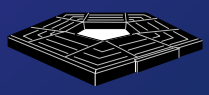
- DIS standard defines 9 dead reckoning algorithms
  - fields in the entity state PDU
  - dead reckoning will be discussed later in the course
- DIS packets are larger than SIMNET packets
  - the entity state PDU has a lot of static information
  - position of a vehicle can be expressed in a subnanometer resolution (64-bit co-ordinate field)
  - could be redesigned to 20% of the current size
- Designed for fewer than 300 units but DoD wants even 300,000 units
- Does not define how to define new types of information nor how to modify the DIS NSA

High-Level Architecture (HLA)

- Aims at providing a general architecture and services for distributed data exchange.
- While the DIS protocol is closely linked with the properties of *military* units and vehicles, HLA does not prescribe any specific implementation or technology.
  - could be used also with non-military applications (e.g., computer games)
  - targeted towards new simulation developments
- HLA was issued as IEEE Standard 1516 in 2000.

Academic NVEs

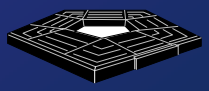
- DoD's projects
  - large-scale NVEs
  - most of the research is unavailable
  - lack-of-availability, lack-of-generality



- Academic community has reinvented, extended, and documented what DoD has done
  - NPSNET
  - PARADISE
  - DIVE
  - BrickNet

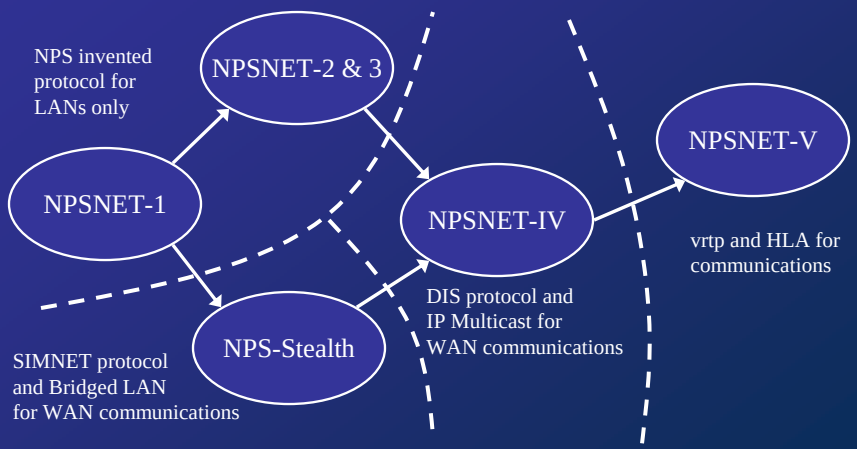
NPSNET

- Naval Postgraduate School (NPS), Monterey, California
  - 'Longest continuing academic research effort in NVEs'
  - 'The complete breadth of human-computer interaction and software technology'
  - …that can be useful for DoD



(Guess where our course books authors come from...)

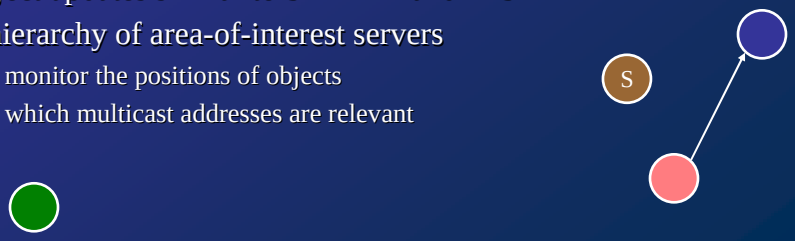
Evolution of NPSNET



PARADISE

- Performance Architecture for Advanced Distributed Interactive Simulations Environments (PARADISE)
- Initiated in 1993 at Stanford University
- Explicitly addressed NSA issues in the case of thousands users

- Assign a different multicast address to each active object
- Object updates similar to SIMNET and DIS
- A hierarchy of area-of-interest servers
  - monitor the positions of objects
  - which multicast addresses are relevant



PARADISE (cont'd)

- All objects, including terrain, are capable of transmitting state updates
- Recognizes that update need varies for objects
- Improved dead reckoning protocols
  - position history-based dead reckoning
- Support for multiple communication flows per object
  - unique dead reckoning for each flow
- Combine information about groups of objects
  - based on their location and on their type
- Support for slowly changing entities
  - to eliminate the heartbeat messages of DIS

DIVE

- Distributed Interactive Virtual Environment (DIVE)
- Swedish Institute of Computer Science
- To solve problems of collaboration and interaction
- Simulate a large shared memory over a network
- Distributed, fully replicated database
- Entire database is dynamic
  - add new objects
  - modify the existing databases
  - reliability and consistency

BrickNet

- National University of Singapore, started in 1991
- Support for graphical, behavioural, and network modelling of virtual worlds
- Allows objects to be shared by multiple virtual worlds
- No replicated database
- The virtual world is partitioned among the various clients