Agent-Based Modeling for Physical Behavioral Entities with Umbra

SAND2014-3383C

Complex Systems Fundamentals, Methods and Applications (CSYS300)

Fred Oppel 6134
Ken Summers 6134

Training Outline

- Agent Based Physical Modeling Approach
  - Umbra Overview
  - Building Blocks for Complex Systems
- Dante (Umbra Application)
  - Force on Force Modeling
  - Scenario Setup
  - Post Processing Batch Runs
  - Create an example Scenario
- Dante Demo
  - Create a simple scenario
  - Execute Batch Runs
  - Analyze Data

Sandia National Laboratories

Embodied Physics Agent Based Modeling

Dept 6134

- Systems analysis and software engineering
- Simulation & Gaming Terrain Team
- Embodied Agents (Physics, Behaviors, 3D environ.)
- Live Virtual Constructive Simulations

Impact

- DOE Physical Security – Safety of NW Complex
- DOE Facility Design - Sensor layouts
- DoD Warfighter Missions – Operational Information

Software Tools

Modular Software Framework

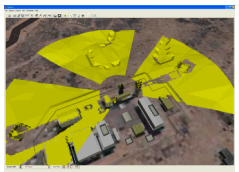
Terrain Generation & Gaming

Force-on-Force Constructive and Tabletop

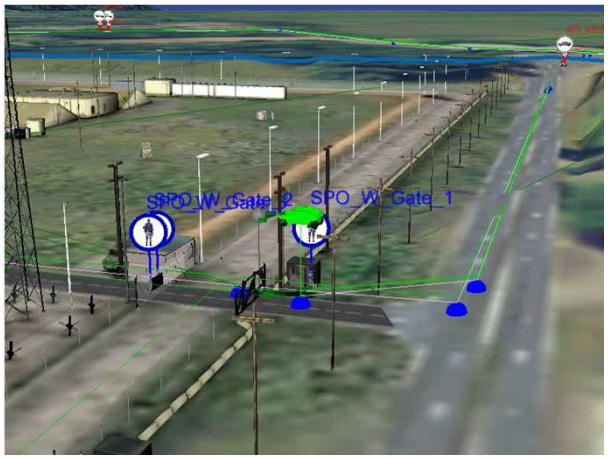
Sensor Operations & Path Planning

Examples

OpShed Sensor Analysis



Dante Tabletop



Dante Constructive Simulation
Small Arms Engagement

Umbra Models
Multi Fidelity Physics and Environments

Sandia National Laboratories

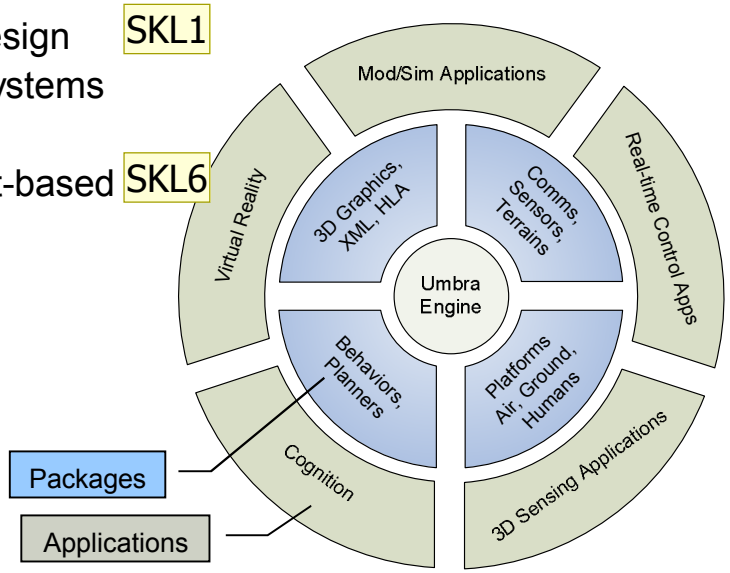
What is Umbra?

Umbra Engine (Software Framework)

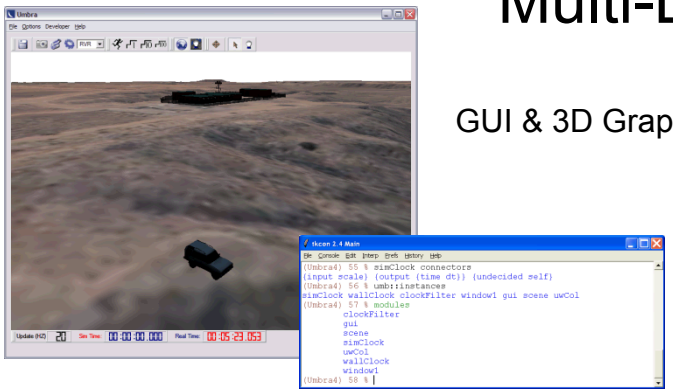
- Modular C++ Core based on Object Oriented Design
- Data Directed Update Graph for Modules and Systems
- Flexible Scripting (Tcl) for Module Construction
- Enables both Physics-based (time-step) & Event-based Models to co-exist
- Supports Batch & 3D Interactive Mode
- Executes on Windows, Linux, and Mac OS X
- Uses Open Scene Graph, XML, Boost, GDAL
- Optimized Computational Geometry Package
- Umbra Worlds support non-linear interactions

Umbra Packages (Current Libraries)

- Platforms (Air, Ground)
- Communications, Sensors, Terrains
- Behaviors, Planners
- LVC Interface for Real and Virtual Entities
- HLA Distributive Computing

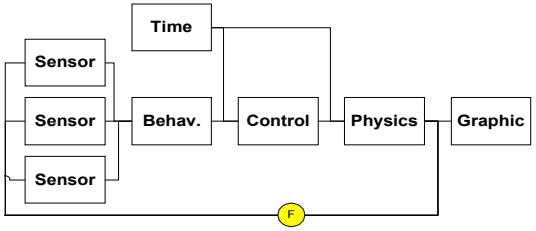


Multi-Layer



GUI & 3D Graphics

Command Shell



Umbra System

Slide 4

SKL1 We might update this slide a little
- to include C#,
- take out Mac OS X (we used to support it, but I don't think we could deliver a build today),
- eliminate the reference to OSG, XML, Boost, GDAL (who cares?).

Summers, Kenneth L, 8/12/2013

SKL6 Something missing in this presentation is an explination of this. Umbra is a time-slice simulator. This is really the only place where that is introduced (and as an aside, at that), but that is key to following explinations.

Summers, Kenneth L, 8/12/2013

Types of Problems

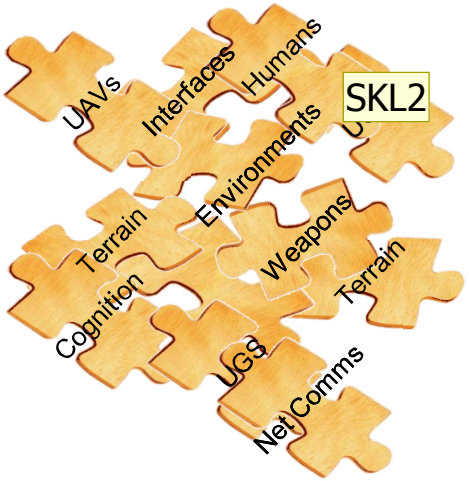
Umbra was designed for

- **Robotics technologies**
  - Sensors, Controls, Mobility, …
- **Network-centric communications**
  - Ad-hoc wireless, full communication OSI layer
- **Human behaviors**
  - Decision making within mission operations
- **Small unit combat operations**
  - DOE physical security
  - DoD convoy operations (IED)
  - Border/Port security
  - Incident response
- **Live-Virtual-Constructive (LVC) interfaces**
  - Augmented Reality training

Sandia National Laboratories

Umbra's design approach

Enable rapid analysis of complex systems using the best models regardless of origin



Umbra Simulation

Portfolio of interrelated projects leading to comprehensive libraries of capabilities and tools

Rapid insights into tech impacts on mission effectiveness

Sandia National Laboratories

Slide 6

SKL2 I don't understand why this slide is here. It is definite marketing speak, and it either should be move way up (because it is extremely abstract) or eliminated altogether (because it is really more of a sales slide than a technical slide).
Summers, Kenneth L, 8/12/2013

Umbra Building Blocks

Sandia National Laboratories

Umbra Simulation Method

- The simulation is built up of sub-components
  - Complex systems built out of many simple components
- Time is sliced into small chunks
  - As time advances all sub-components asked to recalculate their outputs
    - Based on the current time and their inputs
  - Slice size can be set to any value (typically 0.01s to 0.1s)
    - Or Umbra can vary the slice size dynamically to give any multiple of real-time

Sandia National Laboratories

Modules (use as talking points for next slide)

SKL8

- Umbra applications consist of
  - Modules that perform the state calculations for each time step
  - Connectors that transfer data between modules
- Modular, object-oriented architecture facilitates independent development of systems
  - Allows problem and implementation decomposition
  - Independent implementation
  - Promotes code reusability
  - Enhances the flexibility of scenario development
- Modular architecture allows "mix and match" control over fidelity
  - Some modules can be lower fidelity, while modules of particular interest to the simulation can be higher fidelity
- Umbra's architecture provides building blocks for modules from very simple to complex systems

Slide 9

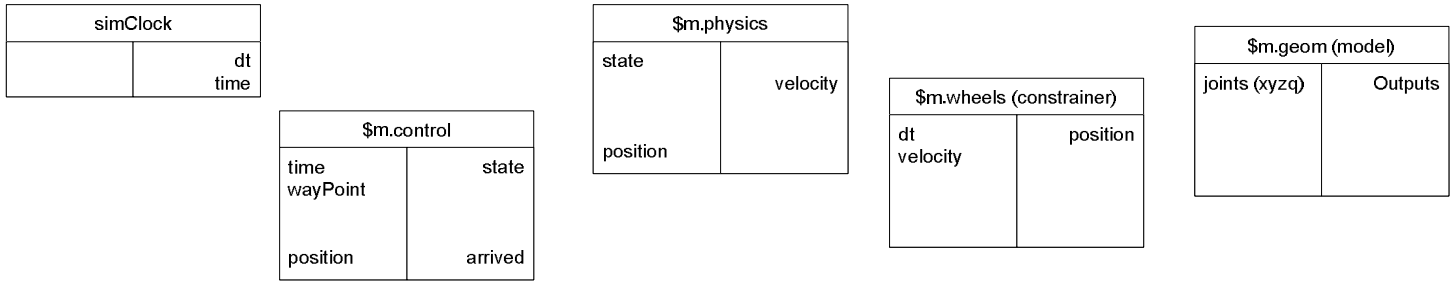
SKL8 This whole slide could be the notes for the following slide (and we could eliminate this one).

So, I'm thinking these are the talking points for the slide on the next page.
Summers, Kenneth L, 8/12/2013

Modules

- Represent system functional components like controllers, sensors, physics,…
- Instances of C++ classes
- Response to Outside Events (commands) from the environment

Example of Module Types



Module is Umbra's word for Component

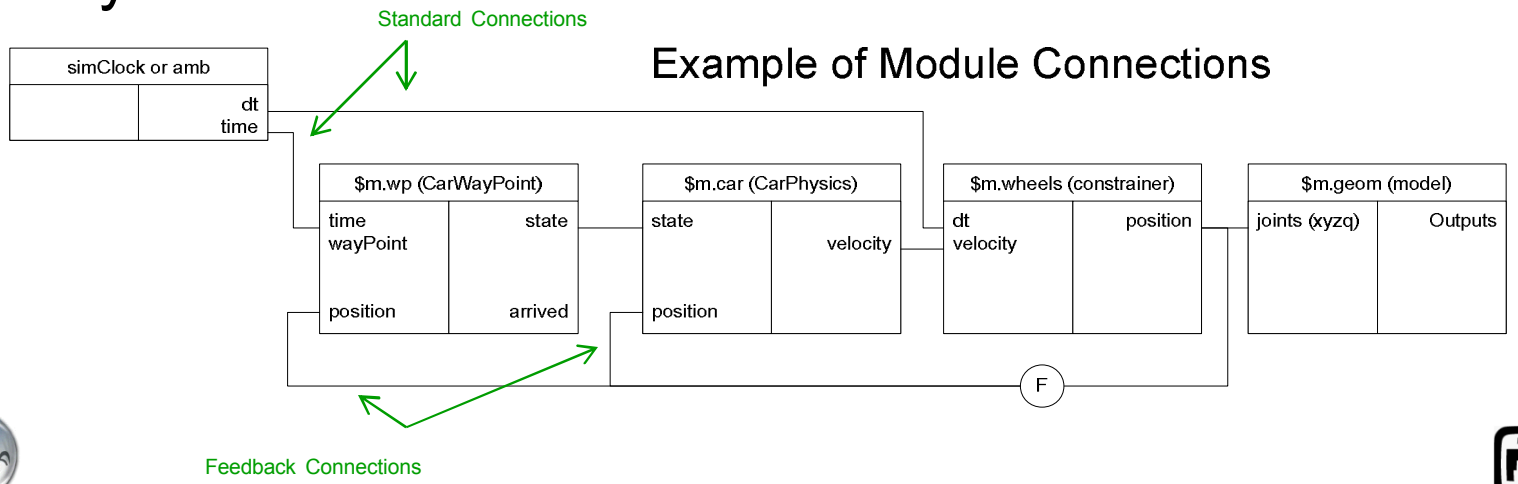
Connectors

- Connect modules to form a system function
- Enable data-flow connections between modules
- Connections between modules determine the update order of modules during each update cycle
- Data transferred can be standard types (double, int, etc.) or complex structs and classes.
- Allow 1 connection to an input and multiple connections from an output

Sandia National Laboratories

Connectors (cont.)

- Connector data can trigger events
- In Standard connectors data flows left to right
- In Feedback connectors data flows right to left
- Feedback connectors provide input data from previous update
- Feedback connectors enable modeling a control system



Graphs

- Determine the Umbra update order when a tick occurs
  - Sequential graph (sorted list) where update order is determined by connection predecessor enforcement or explicit predecessor relationships
- Calls the “update” method on each module that resides on the graph.
- Separates modules into levels where modules at the same level can update in any order

SKL9

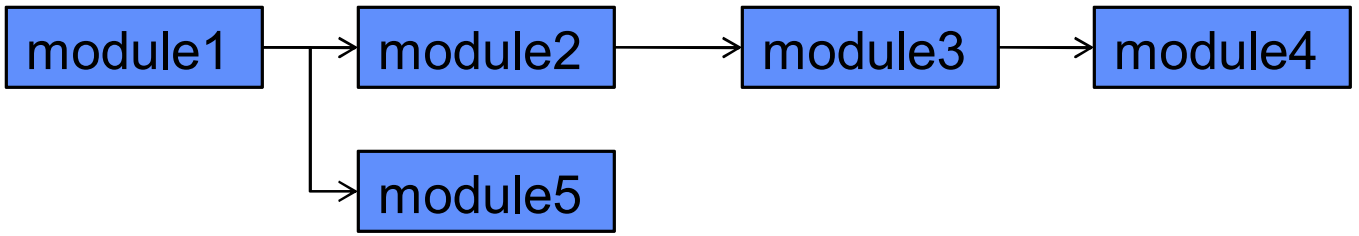
Sandia National Laboratories

Slide 13

SKL9 I *think* I know what this means, but it sounds redundant to me (what is the difference between "connection predecessor enforcement" and "explicite predecessor relationships"?). In any case, it's confusing.
Summers, Kenneth L, 8/12/2013

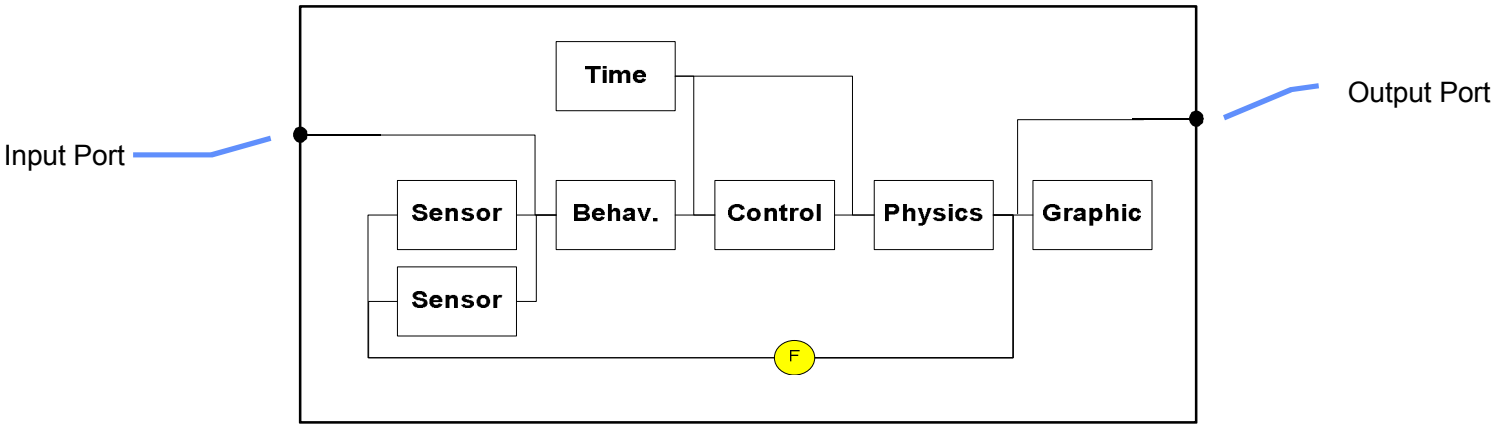
Umbra Update Order

- Basic update order for 4 modules
  - The output of module1 is connected to the input, which means module1 updates BEFORE module2 each step in the simulation engine.
  - Module5 updates at the same level as module2 because module1 is a predecessor



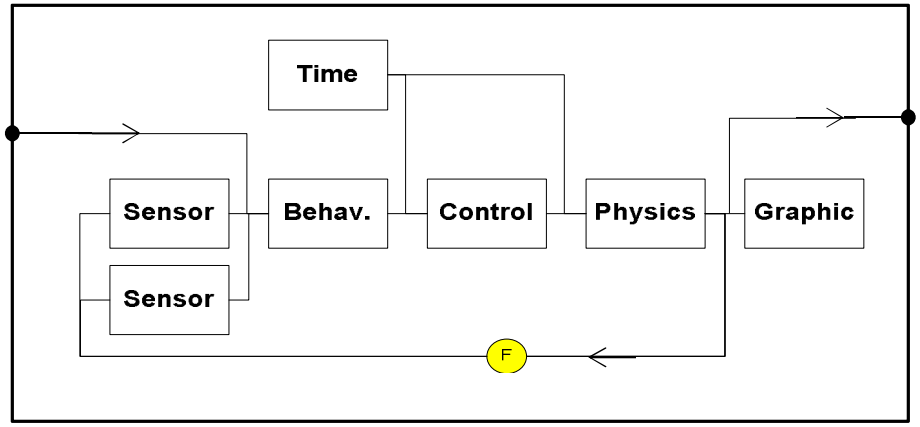
Systems

- A System is a module AND a self-contained Graph that contains its own list of sub-modules
- Systems are “black boxes” that provide complex functionality through the creation of a single module.
- Systems expose “ports” which pipe the data of their sub-modules for external consumption
  - Systems look like “fat” modules from the outside



Systems (cont.)

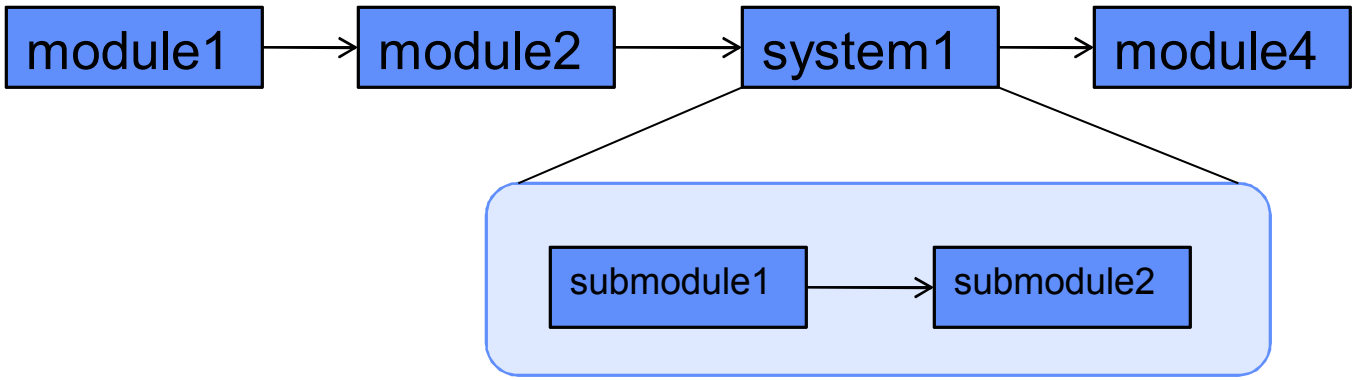
- A System also contains three additional capabilities relative to an ordinary module
  - Constructed of lower-level modules to form a system function
  - Maintain the update graph for the modules in the System
    - It is only the System, not its modules, that is part of the global Umbra update graph
    - The System has the responsibility to update its own modules
  - Update the modules in the System with each time step
    - System can do multiple updates within a single Umbra update
    - This allows multiple levels of time granularity



Umbra System

Umbra Update Order (with Systems)

- Update order when a system is introduced
  - System1 is connected to modules 2 and 4, but its submodules are contained entirely within the system's graph and do not interact with other modules directly



Worlds

- World modules represent phenomena that have simulation-wide impacts (such as communication, collisions, sensors, etc.)
- Worlds encapsulate physical phenomena by abstracting them into a world instead of distributing them throughout the simulation
- Worlds allow Umbra to integrate with other systems and higher-level simulations SKL11
- Worlds are factories that produce "product" modules (*i.e.*, children of the World) which act as probes or proxies for world-specific effects

Basis of the Umbra Patent
(US 7,085,694)

Sandia National Laboratories

Slide 18

SKL11 Need to explain this, probably verbally.
Summers, Kenneth L, 8/12/2013

Worlds (cont.)

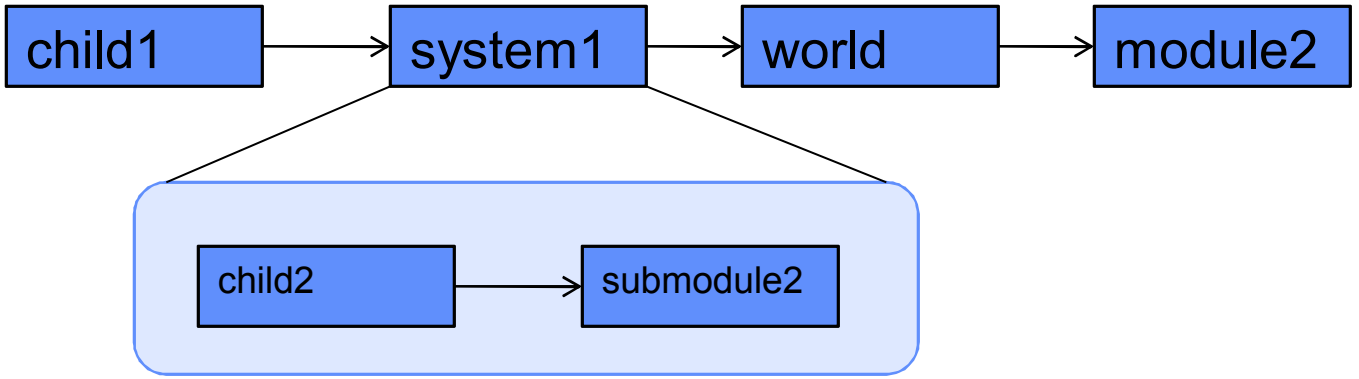
- World modules let developers break the problem into easily understood pieces, thus reducing the complexity of simulation development
  - Modules partition the problem space at the entity level
  - Worlds are special modules that modularize the *interactions* between entities
- Worlds provide the physics for their children
- Examples:
  - Various kinds of sensor worlds (RF, acoustic, radiation)
  - Communication World contains transmitters and receivers
  - HLA World contains objects and interaction modules

Enables modeling complex systems

Sandia National Laboratories

Umbra Update Order (with World)

- Update order with world/children
  - World updates AFTER its children or the systems containing their children.
  - Module2 can update after the world because it has no relationship to world.



Enables a fair fight in Force on Force

Capability Interface

- Umbra Systems contain a capability interface to easily add capabilities to compose complex systems SKL5
- Reduces coding complexity
  - Removes inheritance issues
  - Simplifies library dependencies
- Allows a simple system base
  - Additional capabilities added thru xml or interactive scripts
- Example: Entity Character
  - Base System
    - Contains a position, simple behavior, and graphic module
  - Optional Capabilities
    - Tactical behaviors, Shooting, Sensing, Detectable Properties, Mobility, etc…

Slide 21

SKL5 This sounds confusing: "systems contain capabilities to compose systems" ???
Summers, Kenneth L, 8/12/2013

Callbacks

- Callbacks implement the event mechanism of Umbra
- A callback is a method (C++, TCL, etc.) called when an event occurs
- Events can be user-defined, but there are a number of “standard” events built-in to Umbra
  - Connectors can call events for changes in data values or connectivity
  - SimClock has callbacks for numerous time event types: onTime, onDeltaTime, onPeriod, etc.
  - Graph has callbacks for pre and post graph updates

Parameters

- Externally accessible data that is part of a Module but does not feed into other modules
  - Operates as an easy way to access a piece of data in a module, providing built-in functionality of get/set
  - Can be read-write or read-only
  - Can be mapped to callbacks on data value changes

Sandia National Laboratories

Dante

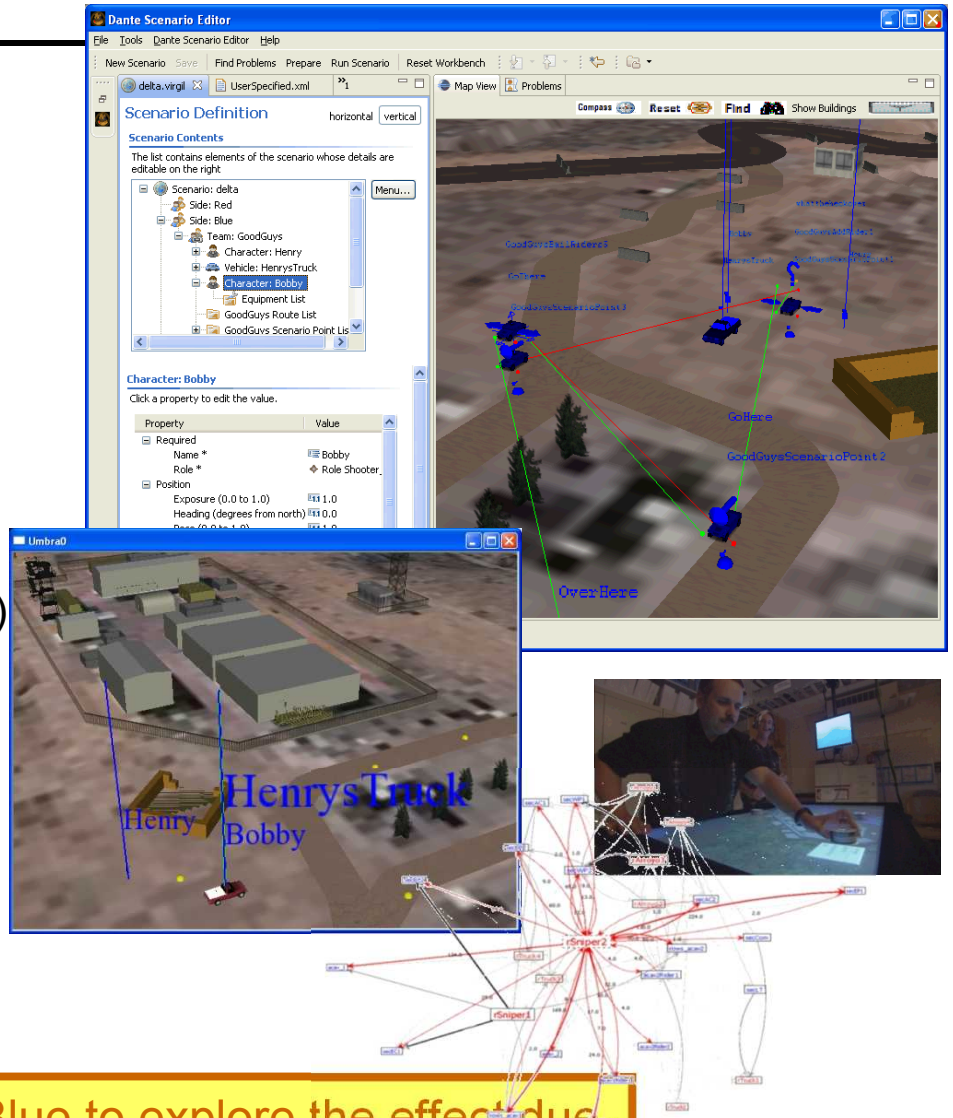
Combat Modeling

Analysis Tool

Sandia National Laboratories

Dante

- Physical security evaluation
  - Concept of Operations (CONOPS)
  - Tactics, Techniques and Procedures (TTPs)
  - Weapons systems
  - Protective systems (barriers, fences, sensors, etc.)
- Force-on-force engagement simulation
  - Automated behaviors and perception
  - Weapon modeling (direct and indirect)
  - Combining physical interactions with statistical modeling
  - Human-in-the-loop not required
- Applied to:
  - DOE Physical Security
  - DoD Operations

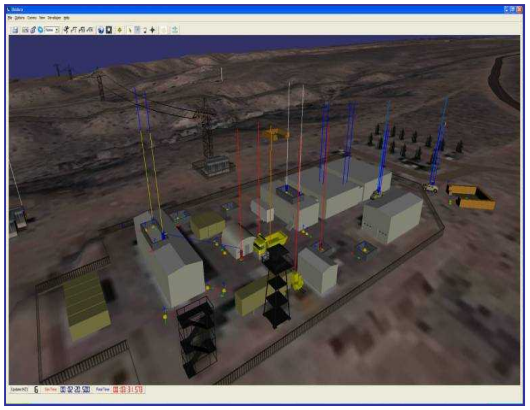


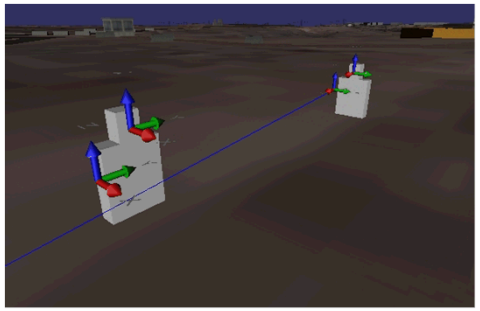
DANTE allows Red and Blue to explore the effect due to changes in CONOPS and technology insertion.

Sandia National Laboratories

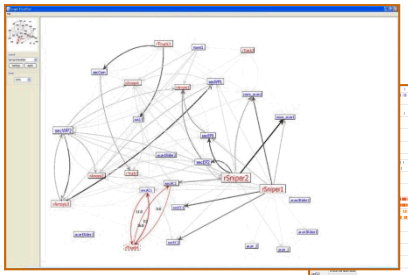
Dante Tool Suite
Combat simulation & gaming

- Scenario Editor
- Run-time execution and visualization
  - Physics calculations
  - Enhanced Ph/Pk
- Batch run management
- Data analysis and visualization
  - Statistical analysis
  - Scenario Replayer





Example of a missed shot where PH said a hit would occur, but geometric orientation caused a miss. Shot lines terminate at the entity/terrain they hit.



Sandia National Laboratories

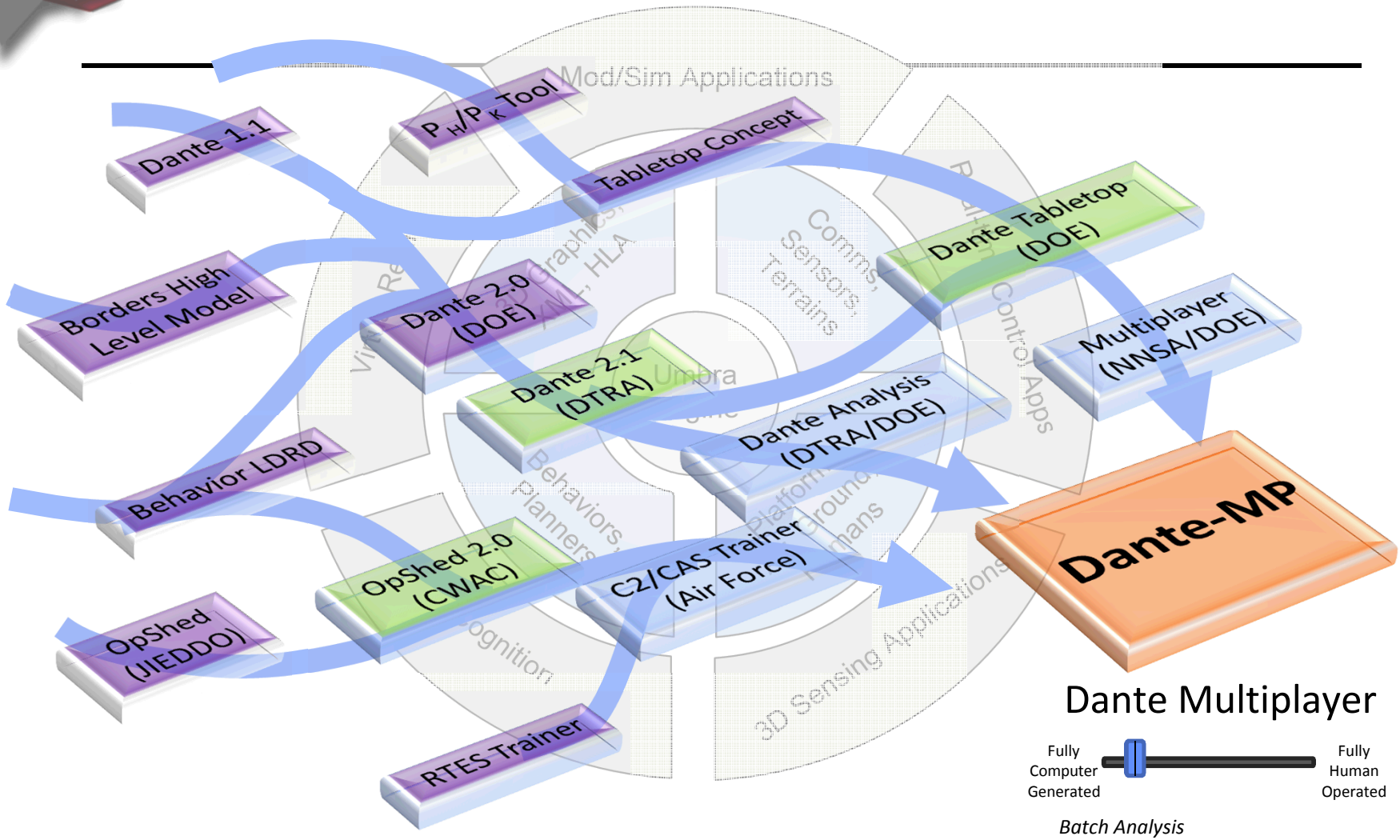
Combat Modeling Concepts

Battle outcomes between entities comes from automated behaviors and perceptions

- **Behavior**
  - Set of selectable character behaviors
  - Behaviors are driven by the entities plans (sequence of planned activities
  - Based on perception (both visual and acoustic) entity will react.
- **Visual perception**
  - Probability of detection based upon line-of-sight obscuration, range and pose.
- **Auditory perception**
  - Sounds are classified, i.e., footsteps, vehicle, explosion, shot (distant), crack (bullet aimed at entity).
- **Engagement**
  - If reaction is to engage, Ph/Pk tabular data is used.

Sandia National Laboratories

Dante Path & Future



Dante Multiplayer

Fully Computer Generated — Fully Human Operated

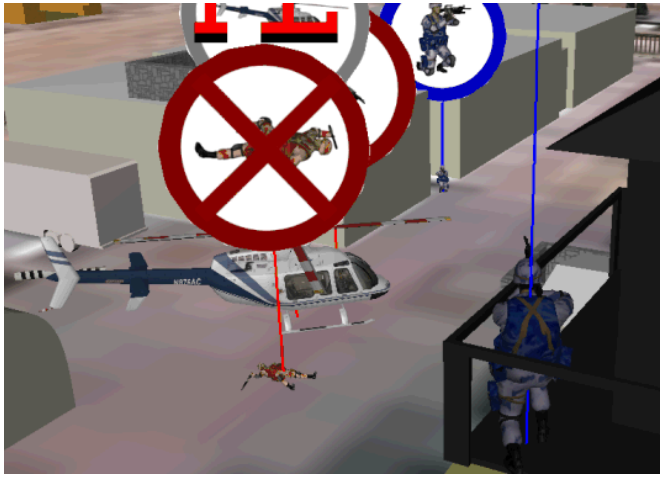
Batch Analysis
TableTop "Turn-Based" Game
Multiplayer Real Time Game

Past Work | Active Work | Future Work

Sandia National Laboratories

Dante Tabletop

- Operation (Current Beta version)
  - Strategy “Turn-Taking” Game
  - Coordinated simulations for Red & Blue
  - Line-Of-Sight in 3D terrain
  - Ph/Pk Engagements
  - “Fog of War” concealing hidden entities
  - Records Events and Notes
  - Referee/Spectator view
- Next Version
  - Leverage Dante batch tool
  - Team Behaviors and Path Planning
- Trainer mode (Future)
  - Red side automated
  - Batch Execution



3D Overview

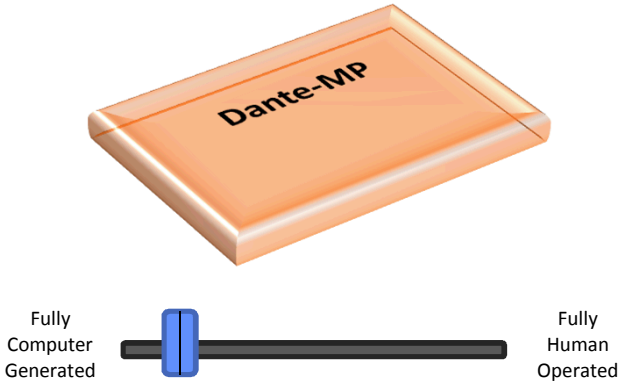
1st person view

Sandia National Laboratories

Dante Mulitplayer Future

Distributed multi-player neutralization tool

- Sliding functionality from fully computer generated to fully human operated
- Scenarios derived / developed from multiplayer game (and run in batch mode for analysis)
- “Gaming” factors mitigated by improved interface
- Active communications simulated in the game
- LVC capable
  - CAS operations
  - Active sensor models
- Insider Threat
- Government-owned software
  - Leverage other government capabilities



Dante Demo

- Create Simple Scenario
- Execute Batch Runs
- Analyze Runs

Sandia National Laboratories

Review

- Umbra is a powerful modeling and simulation framework that allows multiple modules to be incorporated to solve complex problems
- Umbra has been applied to a wide array of engineering problems: robotics, planning, systems integration, etc.
- Umbra's key features are modularity, scriptability, hybrid time management, mixed fidelity, and the concepts of worlds and systems

Sandia National Laboratories