RAPID SIMULATION TO SUPPORT OPERATIONS ANALYSIS-M

March 8, 2013

OVERVIEW

This paper describes tools for analysts to rapidly produce large scale simulations in support of analysis of complex operations. Figure 1 illustrates an example of dynamic operations to be analyzed. The simulation environment described here allows an analyst to create high fidelity simulations using "point and click" interactive facilities and panels. It eliminates the need to build software when analyzing large numbers of concurrent missions that include sensors, communications, C2, EW and weapons systems.

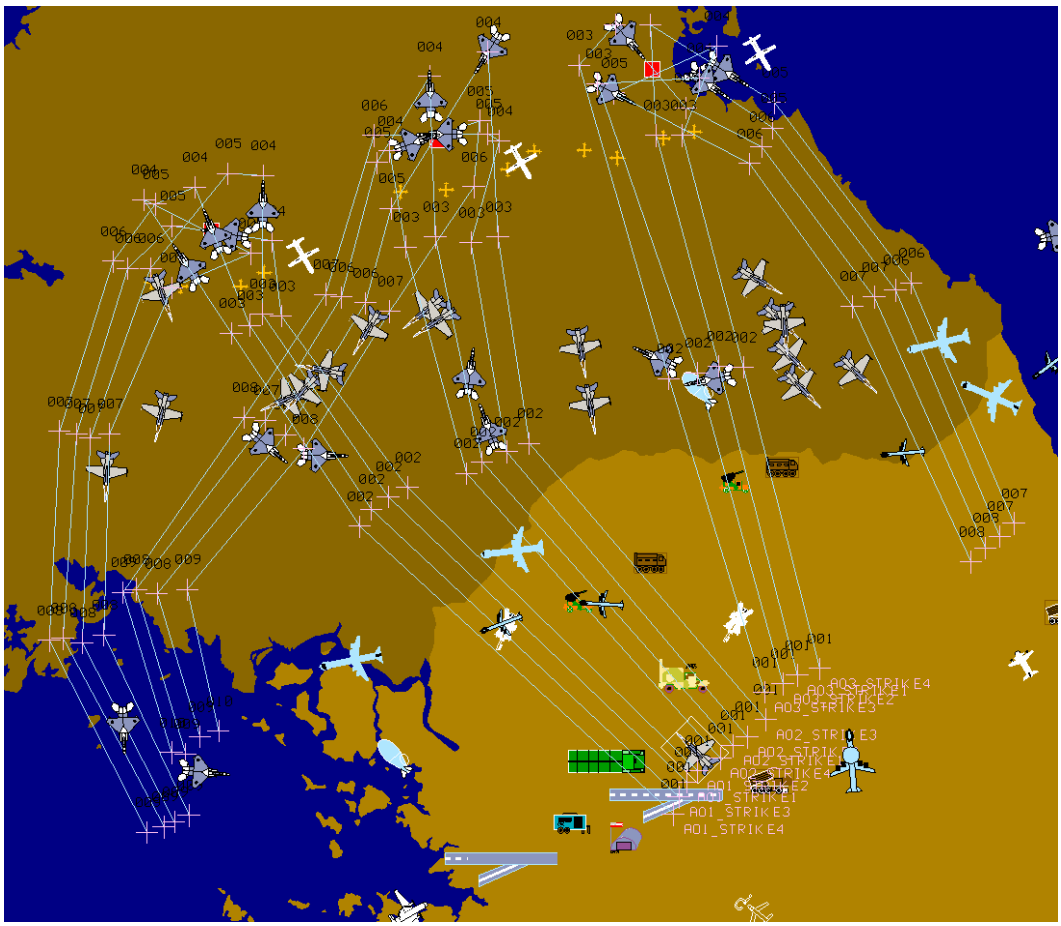

Figure 1. Example of dynamic operations to be analyzed.

This capability provides a library of models and simulation infrastructure that allows analysts to select the specific models and corresponding parameters needed, and to create the dynamic scenarios required to analyze mission effectiveness and performance of platforms and equipment. It also contains a CAD environment for rapid development or modification of models without a knowledge of computer programming. It allows analysts to perform variational analyses to determine probabilities of mission success and envelopes on platform and equipment performance.

UNDERSTANDING THE PROBLEM

To understand the problem of analyzing mission and system susceptibilities to support integrated defense planning, one must investigate the dynamics of unfolding missions containing critical systems that may cause failure. Once identified, one would like to derive approaches to counter the susceptibilities. To accomplish this, the approach presented here uses detailed models of the systems and decision processes in a multi-service multi-theater simulation. This has proved to be extremely successful in obtaining an understanding of potential problems in the systems of interest, and specific approaches available to counter susceptibilities.

The scenario vignettes described here have a common pair of mission types that can be used to analyze the susceptibility of mission threads. These are:

- Find, Fix, Track, Target, Engage, Assess (F2T2EA) Kill Chain - This chain must be closed within a given time frame for missions to be successful.
- Persistent Stare - This requires that targets be tracked over a given time frame for missions to be successful.

CHARACTERIZING SUSCEPTIBILITY BASED ON MISSION THREADS

Missions may evolve in different ways, spawning different mission threads. One must analyze threads that are potentially susceptible to failure, e.g., when selected assets are degraded or denied. In the F2T2AE mission, multiple platforms are required to interact to complete a mission. Each platform may have multiple functions to be performed to contribute to success. Failure of one mission thread can result in failure of a mission.

The following mission scenarios are expected across most Phases of Crisis Action Planning, e.g., (Phase 0/1 – Phase 4/5-6). These scenarios offer specific missions that include threads that are similar, but differ during execution based on platforms, weapons, and C2 systems. These are:

- Joint Integrated Air & Missile Defense (Phase 0/1/2)
- Time Sensitive Targets (Phase 2/3/4)

These scenarios are described below.

Joint Integrated Air & Missile Defense (JIAMD): (Phase 0/1/2 Operations)

This scenario is offered to focus on the Air Superiority mission area, central to the role of the Combined/Joint Functional Air Component Commander. It has force protection implications, as well as providing an operating environment essential for the success of air, land and sea operations. This scenario includes multiple defensive & offensive systems that are land, sea, air, & space based.

These systems include weapons, ISR capabilities, and C2 platforms. It is the most complex & dynamic of the scenarios. It includes representative systems involved in Defensive Counter-air, Offensive Counter-air (to include counter Theater Ballistic Missile operations), Suppression of Enemy Air Defenses, Electronic Warfare, air & space based ISR, air & space based C3.

The following systems are included:

1. Space based IR detection sensors for missile launches with inherent C2
2. Patriot/Thad land based missile defense systems with inherent C2
3. Aegis Cruiser in counter TBM role
4. F-22, F-15C, F-16, F-18 in Defensive Counter-air CAPs
5. Land/Sea based radars for Air Defense (CRCs, Marine TACC, Navy Aegis)
6. AWACS and Navy E-2C for Air Surveillance/Control (data-link capable)
7. Space based SIGINT
8. Air Breathing SIGINT/IMINT (Rivet Joint, Navy EP-2, Joint Stars, U-2, Global Hawk, etc (current data-link capability)
9. F-15E, B-2 in CTBM role (data-link capable) (Scud Hunter)
10. F-16 CJ, EF-18, EA-6B, EC-130 Compass Call, etc, in SEAD, EW roles (data-link capable)

Threat systems include SSM(TBMs: 150-700 NM range); 4th Generation fighters; medium range bombers with Air launched cruise missiles; Surface and air based GPS jammers; Surface base SATCOM jammers; long range Early Warning Radars; modern SA-10/20 SAMs, SA-15s, etc; to protect TBM launch locations, and an Integrated Air Defense System structure.

The focus of this scenario is to define mission threads and C2 communications that defend against a TBM attack, including SIGINT (space and air based) to detect precursor to attack; space based IR sensor to determine launch; C2 to determine threat area; C2 to alert Patriot/Thad/Aegis/targeted base/sector defenses; C2 to the bases to cover up; and lastly, the C2 messaging between Patriot/Thad, Aegis, and CRC/AOC to intercept the inbound missiles.

On the offensive side, missile launches trigger counter-attack operations involving space & air breathing ISR to identify the launch platform, e.g., space based LEO/MEO sensors, Rivet Joint, Joint Stars, etc., and an attack package comprised of F-15E, F-16CJ, EA-6B/EF-18, Compass Call, etc.. Automated and man-in-loop message traffic must be captured for message threads that could be impacted by interference of GPS and SATCOM.

Simultaneously, Defensive Counter-air capabilities must deal with air breathing threats to Blue Forces and bases. Threats include medium range bombers, escort fighters and EW aircraft; and cruise missiles likely launched from other stand-off bomber aircraft. Susceptibility of the Blue Integrated Air Defense network to GPS and SATCOM jamming must be determined.

Time Sensitive Targets (Phase 4 High Value Targets)

Time Sensitive Targets represent a variety of high value targets in Phases 2/3/4 of Crisis Operations. This scenario focuses on classic counter-terrorism missions in Iraq & Afghanistan. Mission and communications threads will be similar across most all TST types. Like the first scenario, this mission touches all parts of the F2T2EA kill chain. It involves a mix of space and air based collection capabilities to Find, Fix and Track the HVT. The Targeting Engagement and Assessment platforms include manned aircraft and UASs.

In this scenario, an environment is assumed where Blue possesses air supremacy at medium to high altitudes. However, one must provide the adversary with land based fixed and mobile GPS and SATCOM jammers. This represents terrorist organizations supported by a sophisticated state-sponsor. The systems involved include:

1. Spaced based ISR, (SIGINT, IMINT, MASINT, etc.)
2. Threat supported by commercial (Iridium like) communications capability, as well as commercial land based cell phone and internet capability, and a representative collection of tactical radios.
3. Air based ISR (Rivet Joint, EP-3, Senior Scout, MC-12, Predator, U-2, Global Hawk).
4. Engagement capability to include, F-15E, F-16CJ/F-16G, Reaper UAS, A-10C, B-1B (Lightening Pod)

This scenario looks at suspected terrorist headquarters within a high CDE area. Meeting site is protected by GPS jammers in multiple locations, fixed and mobile. SATCOM jamming can be introduced, (possibly deceptively).

The analysis must highlight impact of interference on ability to fuse ISR, susceptibility of all platforms to degraded PNT, the impact on weapons, and the balance between risk to mission and collateral damage. Mission and communication threads will impact timeliness of action. Spaced based capabilities can be fixed. All listed air breathing ISR and engagement capabilities are not needed, but can be selected to explore different capabilities in this scenario.

These scenarios are provided as an Operational/Tactical overview. They should be reviewed by USAF organizations, like NASIC and the AWFC. If USAF developed operational scenarios already exist that meet the requirements of JEFX-11, those should be used.

CORRESPONDING MISSION THREADS

Defensive Surface-To-Surface Missile Threads

These mission threads represent part of a defensive surface-to-surface missile engagement from Joint Integrated Air & Missile Defense (Phase 0/1/2 Operations) above. They start with the ISR Collection laydown of Space and Air breathing systems to identify/geolocate suspected launch areas. Surveillance platforms such as RC-135, U-2, C-130 Senior Scout, EC-130 Compass Call, etc, with spaced based ISR systems will attempt to correlate COMMINT, SIGINT, etc., to identify C2 communications to execute a missile launch, enabling geolocation of the launch site. If unsuccessful, the launch of Red surface-to-surface missile event, ①, illustrated in Figure 2, would begin with the launch event triggering a sighting event by an IR sensor on board a Blue satellite. This is followed by two simultaneous mission threads. First, a defensive mission thread will be initiated by a Blue aircraft reacting to the launch to intercept the missile if inbound to a Blue location on the Defended Asset List. Second, an offensive mission thread, (TST), will be initiated by a Blue aircraft reacting to conduct a counter-attack against the Missile site.

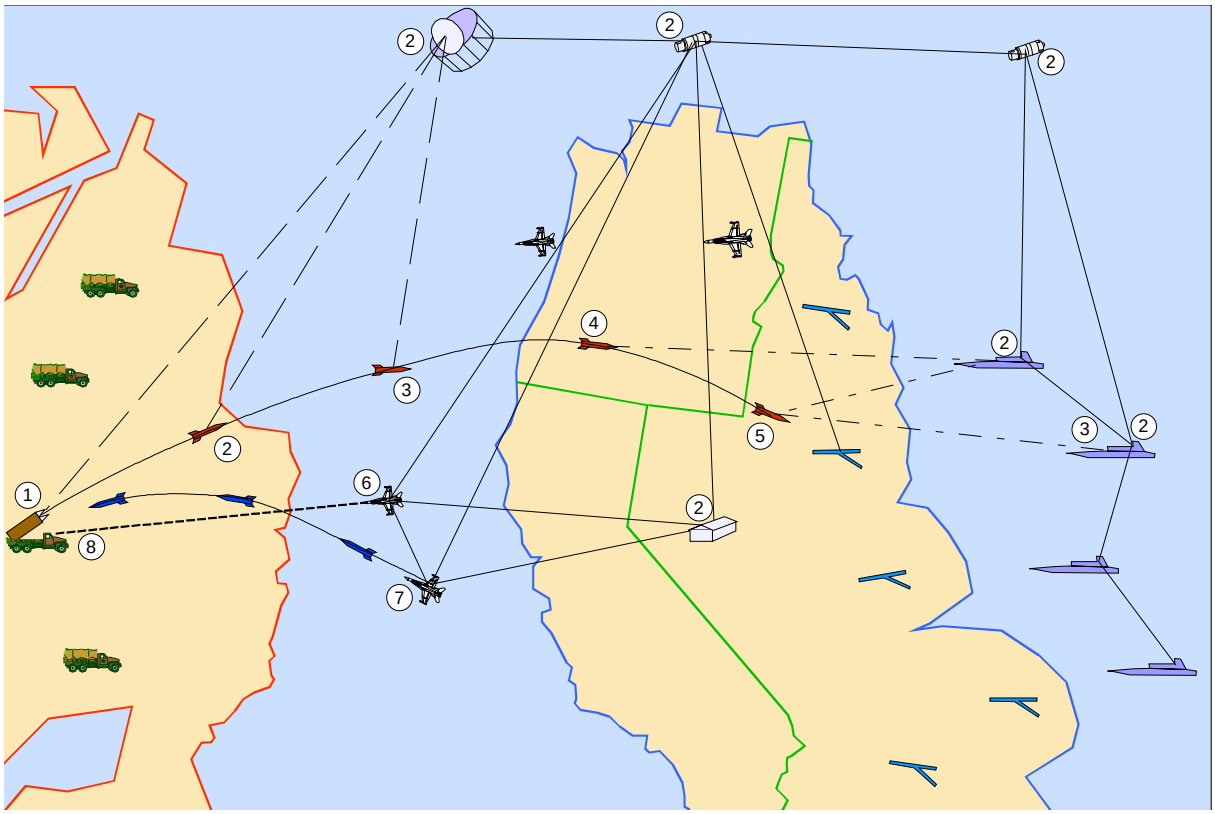

Figure 2. Surface-to-surface missile engagement threads.

An initial sighting event causes multiple communications to multiple platforms in multiple locations (ground, sea and air). Ground Based & Aegis radars take trajectory data from satellites and other airborne sensors, (Cobra Ball, etc) to locate, track missiles in terminal phase, and enable defensive weapon systems, (Patriot, Thad & Aegis) to engage inbound missiles.

Sensors communicate, exchanging information with C2 centers to determine the launch point as well as the final hit point. Threads involve Aegis cruisers and the Patriot system in a defensive surface to air missile engagement. The tracking and targeting involves significant coordination and decision processes across services supported by various communications networks (satellite, airborne and ground). It requires accurate state information regarding position and motion of incoming Red missiles, including GPS updates to moving platforms tracking it, see Figure 3. Both requirements are susceptible to jamming and spoofing threats.

When missiles and paths are identified, assignments are made to engage using surface-to-air missiles. Engagement requires controlling assigned weapons, through the intercept. This requires communications to support position updates. Jamming threats must be considered.

Various communication networks may be used to support the above requirements, including geo-synchronous orbiting satellites, network constellations, Link 11, Link 16, EPLRS, and others. In the related JIAMD offensive mission thread, communications and coordination similar to the Time Sensitive Target thread are necessary.

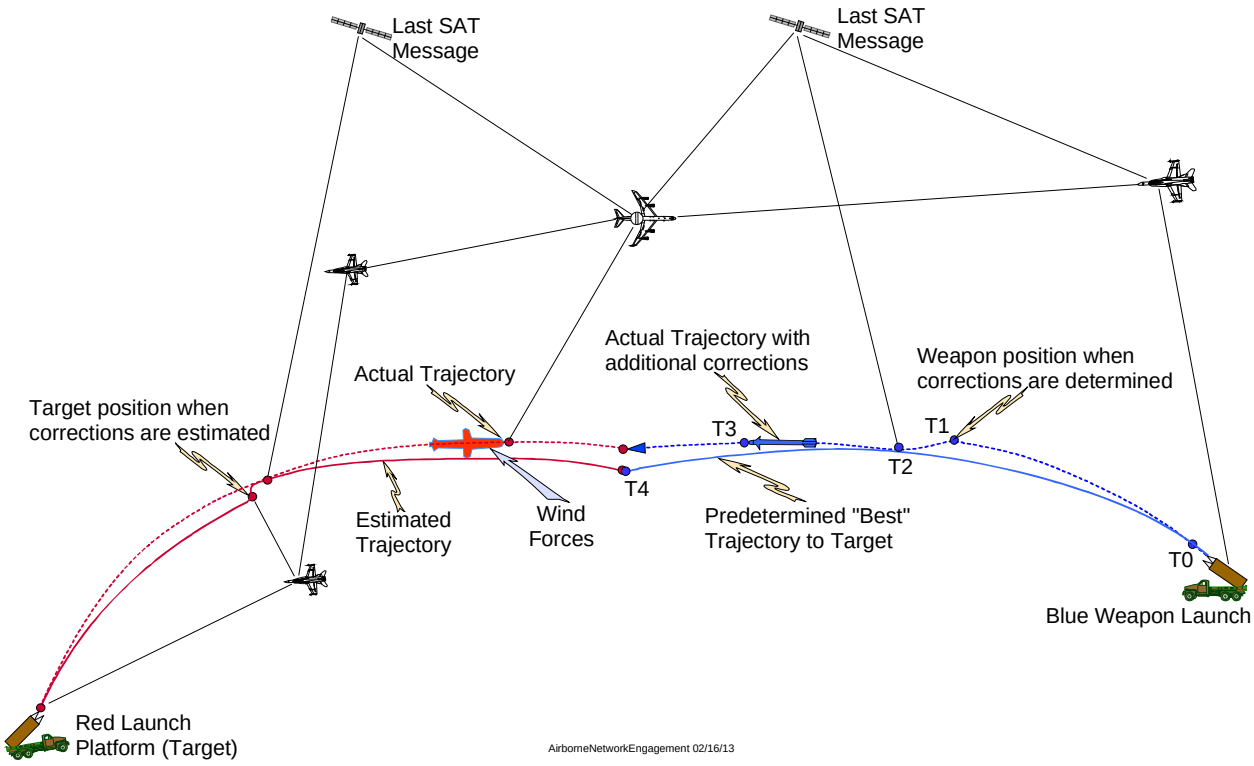

Figure 3. Surface-to-air missile state updates.

Time Sensitive Target Threads

As adjunct threads, consider that once the satellite IR sensor detects a launch, it may coordinate with airborne platforms to determine the launch site. This presents a Time Sensitive Target since the launcher immediately starts to shut down and hide. The launch site target may be assigned to a fighter or bomber equipped with Link 16 radios and GPS guided weapons.

These threads start when a satellite IR sensor detects the launch, followed by a downlink to a ground station. The ground station talks to weapon and sensor airplanes (e.g., F-15E/Rivet Joint/Senior Scout) to fly toward the target, as well as a radar-equipped Joint Stars to start looking for the moving target as it tries to hide. The J-Stars will try to track the moving target while updating the airplanes flying toward it. Communications from the launcher, e.g., C2 messages, will be picked up by the sensor aircraft. Updates on position of the target will be passed to attack aircraft, (F-15Es) to enable location of the TBM target with its onboard sensors, then complete the attack. If there are SAMs protecting the area, F-16 CJ and EA-6B/EF-18 would support this mission, and require sensor updates on SAM locations.

If the TBM launcher is tracked to a fixed hide sight, the decision to engage it from a standoff distance may be made for survivability. In this case, standoff weapons are launched after receiving targetable geographic coordinates, e.g., TLAMS from Aegis or JASM from an F-15E or B-52. These weapons may need GPS updates to hit the target. GPS jammers may be used to protect the hide sight by attempting to degrade the position information on the weapons themselves, as well as the launch platforms.

Thread Error Budgets

When dealing with this type of problem, one must estimate the error budgets for all contributing components so that specific error constraints for each unit are met by design. These budgets depend upon trade-offs that are affected by factors such as cost, size, availability within time frame of interest, etc., as well as the worst case scenarios of interest. Typical factors include the following:

- Required (worst case) accuracy at the target, with maximum allowed error specified in 2 or 3 spatial dimensions and time
- Position of target
- Terrain, foliage, buildings, atmosphere and other factors surrounding target
- Distance from weapon launch to target
- Path to target
- Velocity of weapon
- Velocity of launch platform
- Sensors on weapon
- Sensors on launch platform
- Receivers on weapon
- Receivers on launch platform

THREAD SUMMARY

The above threads have been derived from the selected mission scenarios described above. These threads can be tailored to evaluate the susceptibility of the systems under analysis. They can also be tailored to various terrain environments. This tailoring is best supported using the simulation tools described below. The treads are complex by nature, since they represent the detailed processes, platforms and equipments required to ensure successful missions. Valid analyses of these thread requires substantial simulation assets, including models that have been proven to run fast as well as provide a valid representation of the systems and environments.

OVERALL APPROACH

Based upon PSI's prior work with all of the DoD services using its simulation and planning tools, we propose to deliver a world class simulation facility to AF/ACC, Langley AFB, to support immediate needs for planning, optimization, and analyses of the types of systems and scenarios described in the threads above. This can start with high-fidelity models and simulations already developed for prior DoD clients. An overview of the proposed capabilities is illustrated in Figure 4.

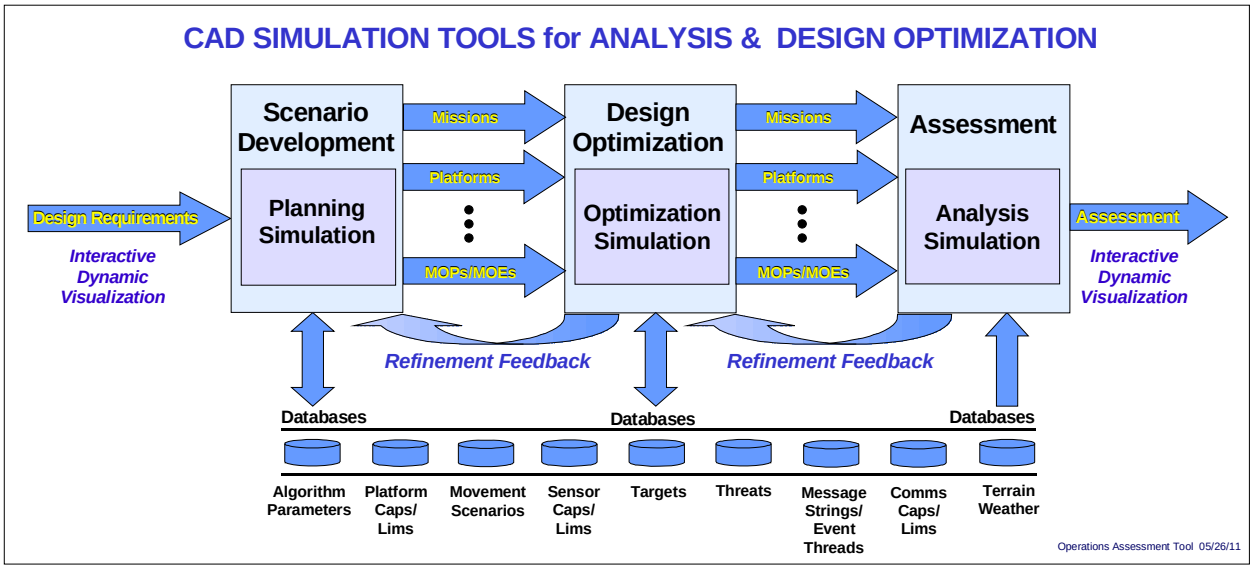

Figure 4. Proposed simulation capability for Navy analysis & optimization.

Figure 4 shows three simulations, one for complex multi-service, multi-theater scenario development (Planning Simulation), one for parameter design optimization (Optimization Simulation) and one for assessment (Analysis Simulation). The planning simulation is used to support interactive creation and modification of scenarios. The optimization simulation is used to determine optimal design parameters, e.g., those used for optimal sensor placement. The analysis simulation is used to perform general analysis, producing various MOEs and MOPs using large dynamic scenarios, and to perform parametric, sensitivity, and Monte Carlo analyses. The features of the proposed simulation capability are described below relative to functions to be performed when supporting an analysis effort.

USING SCENARIO VIGNETTES TO SUPPORT ANALYSIS

Figure 5 illustrates how selected vignettes (parts of a scenario) can be used to evaluate MOPs and MOEs. In this example, there are two simple sensor-to-shooter vignettes. Hostile radars radiate, ①, and these target radars are picked up by networked aircraft sensors, ②. A call for weapon assignment, ③, is followed by an engagement agreement,④. A weapon is fired, and guided, ⑤, while in flight.

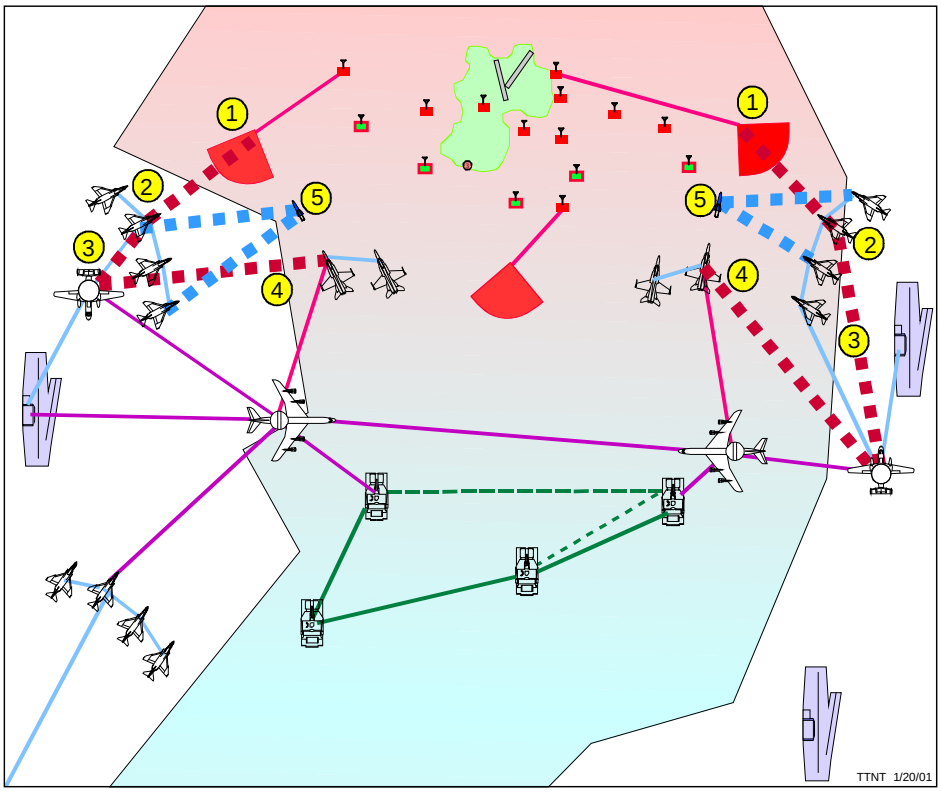

Figure 5. Scenario vignettes can be used to analyze message traffic requirements.

From these vignettes one can track the mission threads using the sequence of events and corresponding message strings that are required from the initiation of a mission (target sighting) to its successful completion (target neutralization). These mission threads must be completed in a time frame that ensures successful completion of the missions. If one of the messages in the thread is jammed and does not get through, or is too slow getting through, the mission may not be successful. By analyzing a sufficient number of representative vignettes, one can derive the probability of mission success. Given that mission success probabilities must be above some minimum, one can compare different design approaches.

The vignettes in Figure 4 are simplified relative to those needed to analyze sensor and communication requirements. An improved illustration is offered in Figure 6. An original mission plan is to address targets T1, T2, and T3. Along the way, targets T4 and T5 are sighted. Rapid decisions must be made as to how to address these new targets. Information must be shared as to which platforms can track which targets with what accuracy, and what weapon systems should be used with what guidance. This illustrates how timing and bandwidth requirements may be derived.

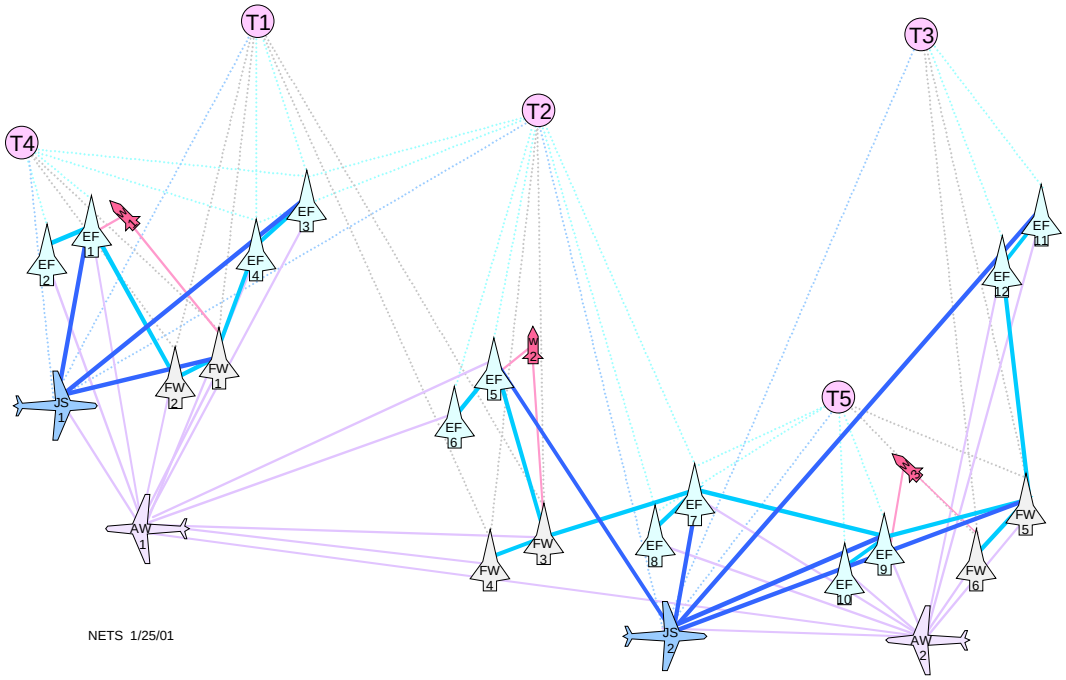

Figure 6. Illustration of the interactions between sensors and weapons systems.

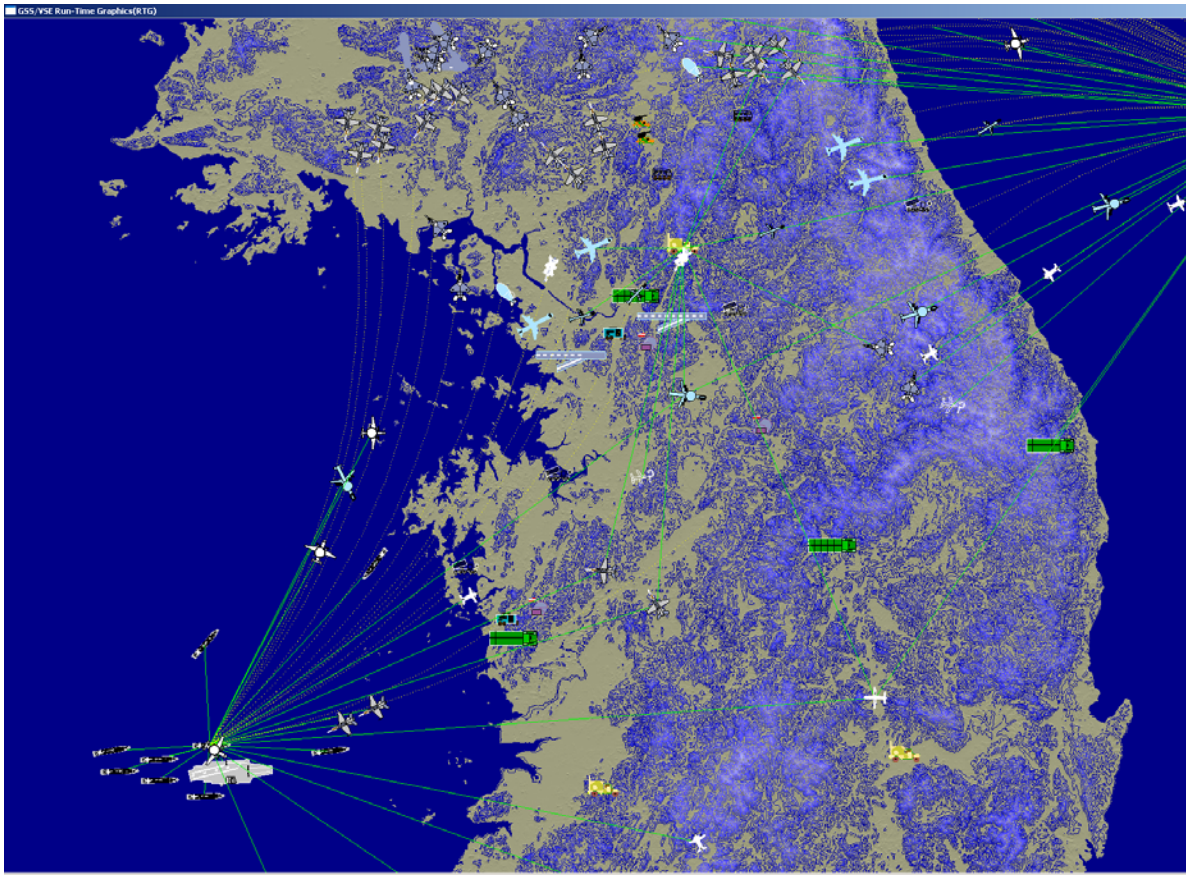

Figure 7. A more realistic illustration of interactions of sensors, C2, and weapons systems.

The screen shot in Figure 7 illustrates a larger number of moving platforms sharing different information concurrently. This figure points out the need for simultaneous many-to-many communications in a very dynamic environment. Figures 6 and 7 illustrate the need for a more detailed approach to determine what messages must be sent and received by whom, and the determination of their probability of reception on a highly dynamic basis. All of these details must be modeled and measured in the simulation tool.

Target Tracking, C2, and Weapon System Control Traffic

To understand the types of vignettes illustrated above, one must analyze the pertinent details affecting sensor, C2, and weapon systems traffic. Based upon past efforts, this is most easily done using a simulation that plays all of the desired details. One must model the event driven mission threads that represent the rules that compose the sensor, C2, and weapon system control loop(s).

The following are examples of factors that may affect the timing as well as the rules of these mission threads:

- Target sighting/tracking events
- Resulting track information messages sent from sensor platforms
- Reception events and latencies at other sensor, C2, and weapons platforms
- C2 command/control messages - target tracking and weapon assignments, etc.
- Sensor to control console/display message events and latencies
- Engagement determination and control messages
- Other sensor, C2, and weapon control messages

This information must be used to build detailed scenarios.

Using Event Driven Simulations With Smart Models To Support Analysis And Design

One of the most time-consuming processes faced in simulation is scripting and synchronization of events and message traffic. Simple modifications of a scenario can be tedious when working with huge scripts that must be synchronized. Using event driven simulations and smart models that contain representative decision processes, a relatively small number of events and messages can be scripted as an input scenario that causes the unfolding of very complex vignettes containing huge numbers of events.

For example, one can easily lay down paths for the surface ships and flights shown in Figure 7 using the movement models in PSI's existing LINK 16 simulation. This is illustrated below in the section on Movement Models, where flight paths may be copied, moved, and modified graphically, *while the simulation is running*. Ground unit paths can be created using the same movement model facilities.

Using C2 Table-Driven Mission Thread Models

As flights fly, emitters emit, and movers move, sensor models will detect targets. Target detection can initiate mission threads that represent the system's rules of engagement. Decision processes can determine target priorities and weapon system assignments. All of the messages to support these actions can be processed as they would in a real engagement. PSI has built C2 table-driven mission thread models that can be used easily and quickly to represent this complex process. Tables can be built that contain the set of events that are caused by the occurrence of each single event. Mission threads are built by loading the C2 tables - instead of changing the logic in the models, or the timing and tables in a tedious script.

Determining Mission Thread And Operational Net Latency Budgets

The initial models can assume that, if A can hear B with a good signal at the receiver and just ambient noise, then all messages from A to B will get through. This will provide a capacity stress based upon good but realistic connectivity. On the other side, if connectivity is poor, and relays are needed, delays can increase or messages may be lost.

Average delay per unit can be modeled parametrically to determine the mission thread traffic under optimal conditions. Results can be rolled up into measures of performance and effectiveness. These can be used to determine latency budgets for the threads. Once this is accomplished, individual unit delays can be varied parametrically to determine latency budgets on the units. Probability of mission success can be tied to the unit delays. Other factors affecting mission success can be accounted for in the parametric analysis. This data can be used to determine operational net requirements.

A Sufficiently Expanded Scenario Of Multiple Vignettes

A scenario must be created containing a sufficient number of vignettes, running simultaneously, to represent realistic stress cases. This is a scenario that one could expect to occur over a large geographic area, where groupings of missions are important to ensure realistic stress conditions.

Create & Modify Scenarios Fast

Creating and modifying complex scenarios can be a time-consuming process. When planning detailed operations, one wants to observe the dynamics - graphically - and interact with the simulations *while they are running* - to make necessary changes. The CAD system behind these simulations contains an advanced run-time graphical environment that allows analysts to *modify* - as well as observe - scenarios on the fly, and save numerous versions for future use. It has many facilities for rapid entry and modification of databases required to produce a scenario.

The planning simulation supports connectivity analysis as illustrated in the screen shot in Figure 8 while running much faster than real-time. Users may interact with the simulation to modify scenarios. Flight paths or segments of flight paths may be created, copied, or modified - on the fly - *while the simulation is running*. This allows analysts to observe the effects of changes in position to changes in connectivity interactively.

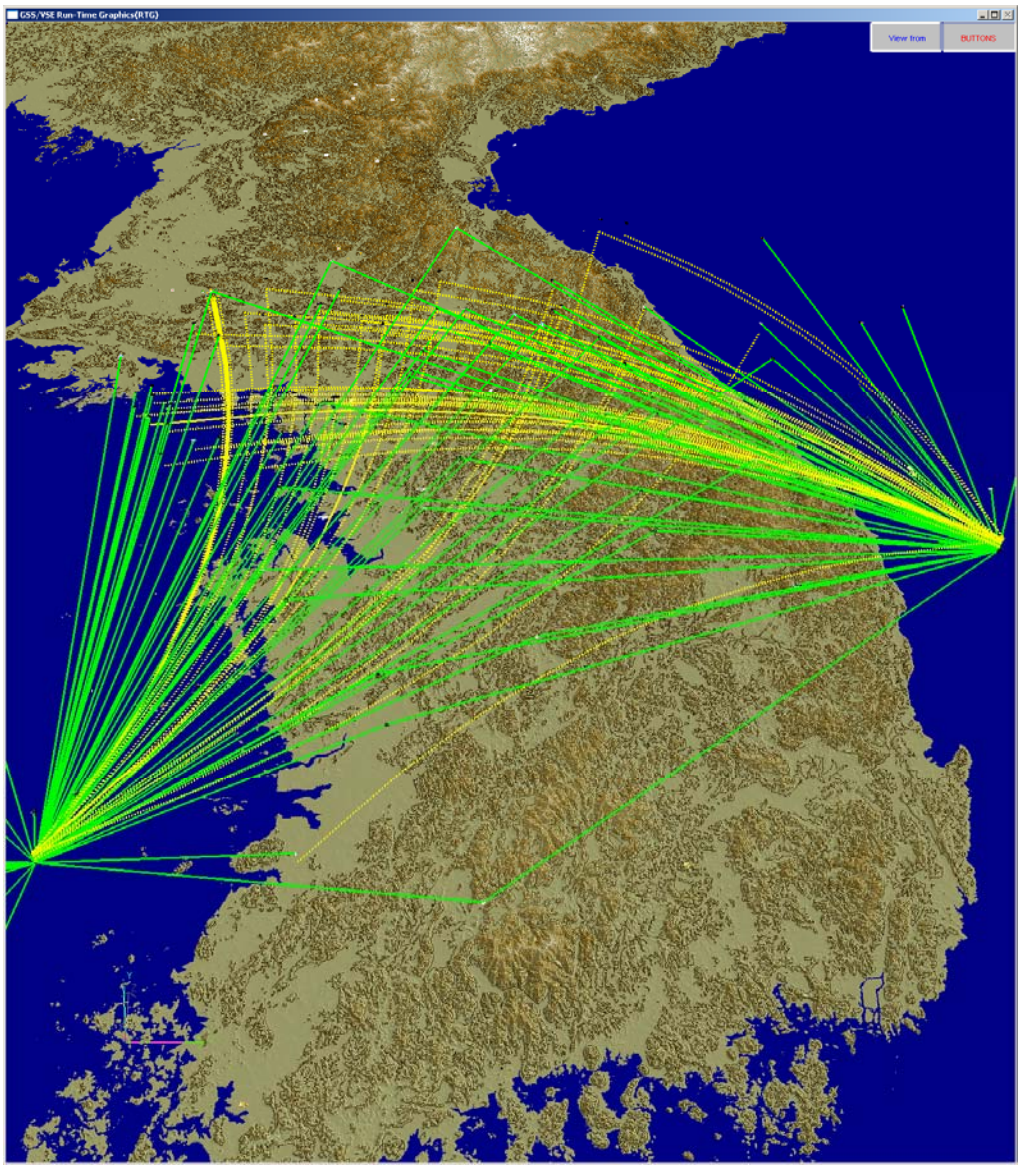

Figure 8. Example of communications connectivity between multiple platforms.

PSI's simulations also contain smart C2 models that recognize events and changes in status, and take care of timing and synchronization based upon C2 decision processes. This determines how sequences of events unfold - on the fly. This eliminates the need for huge script files that are used to determine the synchronization of events in a complex scenario. Using this approach, significant scenario modifications may be accomplished in minutes instead of hours.

OVERALL MODEL ARCHITECTURE

ADDING & MODIFYING MODELS

Figure 9 provides an illustration of the types of models that must reside within a simulation to support various analyses. Platform models include ground, sea, air and space. All of these platform and equipment models reside in the PSI library which has been growing since 1982, starting with TACSIM Line-Of-Sight (LOS) and sensor models.

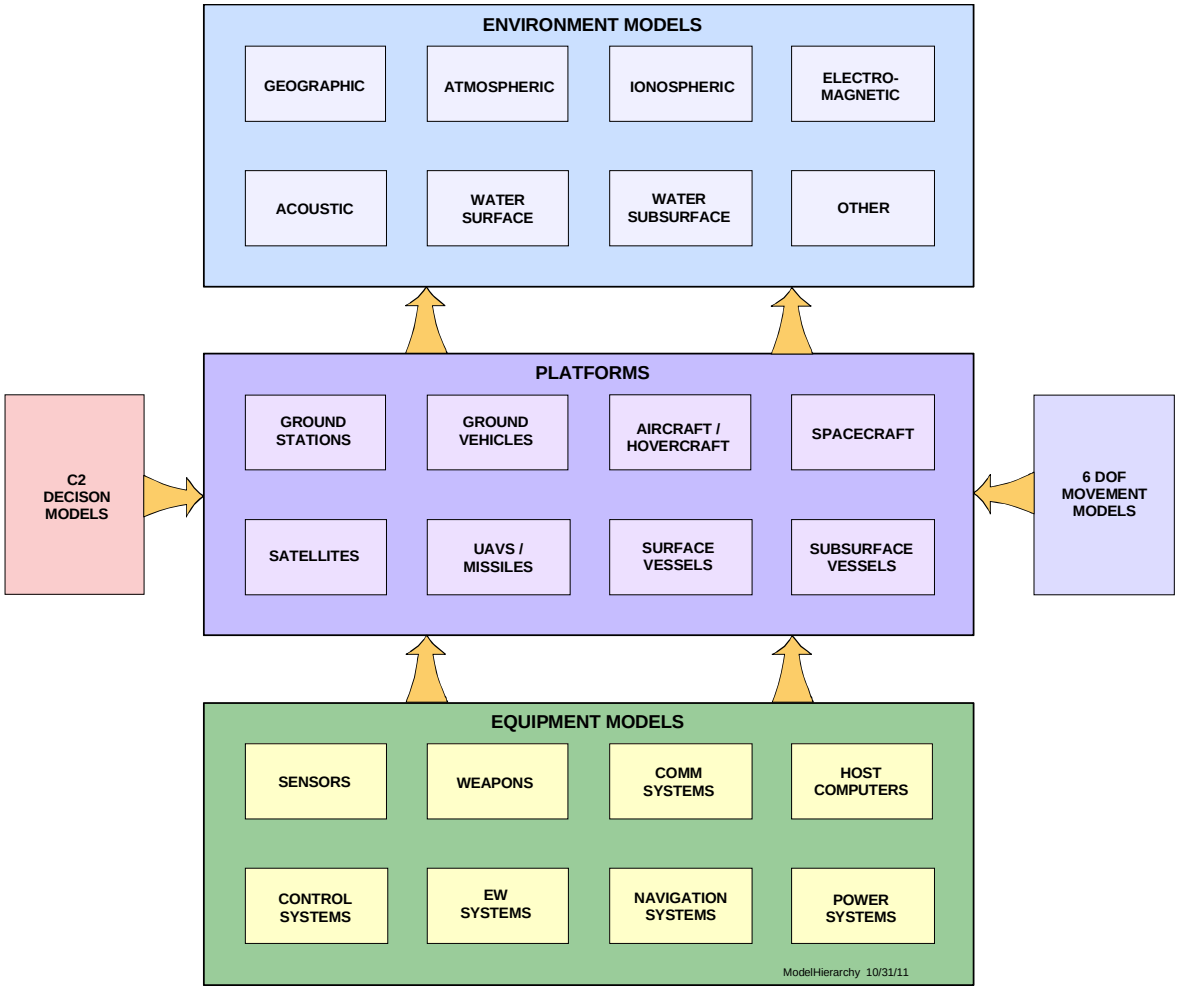

Figure 9. Overall all architecture of the models.

Underlying this modeling facility is a software architecture technology that has evolved as a CAD facility for simulation. This CAD facility was designed to support the rapid development of models and simulations by subject area experts without help from programmers. The technology behind this facility has dramatically increased productivity, including ease of reuse and modification of models. This facility can be enhanced to provide for selecting specific types of models by clicking on check boxes in a user-friendly panel.

Figure 9 provides an overview of the model architecture that supports rapid creation of complex simulations. This facility may be used to evaluate detailed equipment designs as well as comparative assessments of platforms.

The system described here contains a large library of high fidelity models that may be used to create a simulation with most if not all of the platforms and equipments one would need. If a model does not already exist in the library, it is relatively easy to take a model of an existing piece of equipment and modify it to obtain a different piece of equipment.

The difficult models are already well tested in many different simulations. These include Line-Of-Sight (LOS) and propagation prediction models using standard DTED databases.

Assigning Equipment To Platforms

Platforms that are inserted within the simulation may have specific equipment associated with the platform type. If there is no equipment associated with the platform type, a panel like the one shown in Figure 10 may be supplied for one to specify the equipment to be loaded. Antennas may be selected via a panel such as the one shown in Figure 11. The user may select antennas for use with the radio equipment that has been loaded. One may want to select a simple omni-directional antenna or refer to a separately loaded antenna pattern. Antenna patterns may be provided as full 3D specifications. For existing platforms, platform attributes can be viewed through a Platform Attributes Panel while the simulation is running.

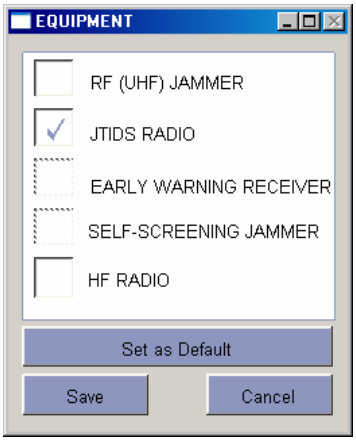

Figure 10. Equipment Selection Panel

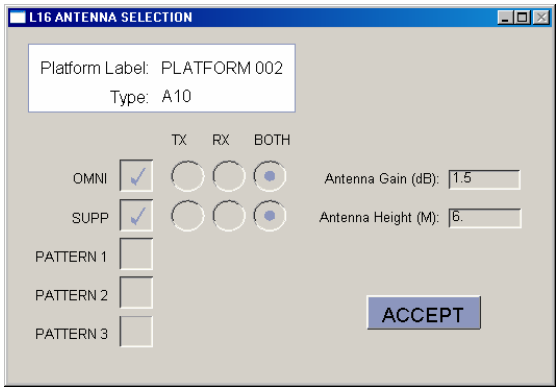

Figure 11. Antenna Selection Panel

OVERALL SIMULATION ARCHITECTURE

The simulation architecture allows analysts to select those platforms to be incorporated into a simulation, as well as the equipment on each platform for rapid creation of complex simulations. Figure 12 provides an illustration of different platforms each carrying different equipments. The simulation architecture takes care of automatically integrating different equipment into specified platforms, and platforms into specified missions. Missions are defined by mission threads, including the movement of all platforms as well as sensor sightings and targeting events. Events are defined by decision tables in the C2 models as well as messages, and all may be defined by input files including the movement paths to be used by each platform.

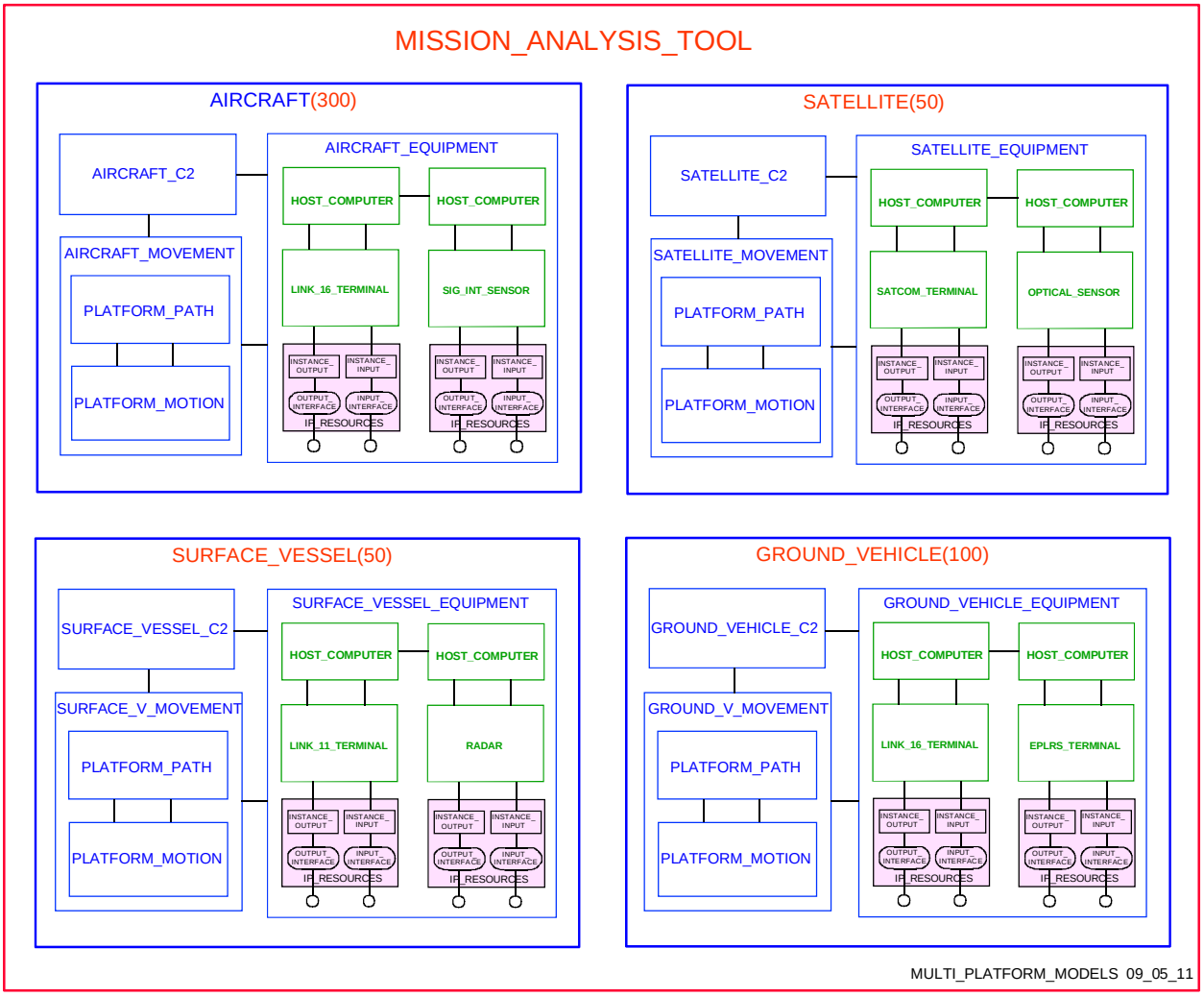

Figure 12. Simulation with aircraft, satellite, surface vessel and ground vehicle models.

MODEL ARCHITECTURES

Platforms contain the following basic model types.

- **Equipment Models** - These models include host computers, radio terminals, sensors, communications and EW equipment. They are all tied to the C2 model and may be tied directly. For example, a radio terminal may receive a message from another platform whereupon it may send that message to its own C2 model as well as retransmit that message or a new message to a different platform. Upon receipt of the incoming message, the C2 model may initiate a sensor model to track a target or a weapon model to prosecute a target.
- **Movement Models** - These models determine platform movement, e.g., following a given path or moving from a given path to another path based upon a C2 event. For example, the C2 model on a given platform may receive a message from another platform whereupon it determines that the given platform must change its movement path to initiate a new mission thread.
- **C2 Models** - These models manage mission threads based upon event sequences. Events may be initiated at predetermined times based upon event files that are read in at initialization time. Once a scenario starts to unfold, events will normally occur based upon the outcome of prior events or the exchange of messages from equipment on a given platform or between platforms.
- **Environment Models** - These models are used to exchange information between platforms as well as define the environment in which platforms and equipment operate. For example, radio messages are put into the environment in a given frequency band and thus transmitted to other radios listening in that band. Jamming signals are also transmitted to receivers using environment models.

All of the above models are initialized using input files that determine their behavior. Special facilities exist for populating and changing the content of these files quickly and easily. These files are stored as a scenario or variations on a scenario. These files may be modified interactively during the course of a simulation and stored under a new scenario name.

EQUIPMENT MODELS

Various types of equipment models may be incorporated into a platform. Some of these are provided below for examples. There are many more than those listed below. Equipment may be placed upon platforms interactively, including equipment type and model number to check for compatibility.

Host Computers

Host computers send and receive messages to and from one or more communication systems. Hosts may generate multiple messages based upon incoming messages, thus forming message strings. Hosts may generate events that are sent to a C2 system based upon messages received. Hosts may also generate messages based upon events received from a C2 system.

When messages are sent and received they are time-stamped and recorded for both immediate and subsequent analysis. This provides for measures of latency so that one can determine if response time and throughput requirements have been met. It allows the modeling of delays within a communication system to be modeled directly, protocol layer by protocol layer, along physical lines.

Communication Systems

Many different types of communication systems have been modeled to support C2ISR requirements, from fixed infrastructure networks to mobile networks of multi-hop radios. Connectivity, capacity, and latency are the primary communication effects of interest. Radio connectivity can be measured on a link basis in terms of the probability that a destination can receive the desired information within a given time increment. This is best done using the Receiver Operating Characteristic (ROC) of the radio knowing the signal power and noise power at the input to the receiver, accounting for all of the factors affecting signal loss, including specific jamming waveforms. Capacity is best measured by sending traffic, e.g., messages or packets of data, including overhead based upon the specific protocol layers used. Since network delay between locations directly affects mission effectiveness, message traffic must be time stamped to determine arrival times.

Sensors

Various types of sensors have been modeled. Sensors range from Acoustical to Optical, including RADAR, Signals Intelligence (SIGINT), Imagery Intelligence (IMINT), and Measurement And Signatures Intelligence (MASINT). Different sensors require different characteristics to represent their capabilities. The difference in requirements for supporting different sensor types warrants separate models for each. The focus on sensors will depend upon the scenarios selected. The attributes used to characterize sensors can expand or change as additional resolution is added. The characterization and resolution are also influenced by measures of merit required by the analyses. For example, when targets are detected, one may want to generate an event or message that causes a follow-on action. These messages and events may be recorded for immediate output or for future reference, such as when tracing causes of failure. As specific sensors are identified, additional characteristics may be required to more accurately represent a sensor.

Red Emitters

Red emitters are required to provide opportunities for detection. Red emitters need not be modeled in the same detail as blue communication systems to provide for detection. However, event threads and message strings may be defined to provide realism concerning the order and timing of transmissions. The criticality of the messages may then be established based on their mission role. This provides greater accuracy of traffic representation than simply generating random messages at the transmitter, since traffic is typically highly correlated with event threads. An example is timing of sensor data fusion at higher levels involving the determination of threat entity relationships and their intent.

Radars

Radars provide the ability to detect and track platforms. Types include pulse, continuous wave, and pulse Doppler radars. Each requires slightly different characteristics, e.g., continuous wave radars make use of two antennas. Vulnerability of platforms to radars is a function of their cross section. Radar Cross Sections (RCS) may produce radically different results for incident angles that are only a few degrees off. An RCS may be implemented similar to an antenna pattern but a single value representing the worst case RCS value may be used in many cases.

Jammers

Different jamming techniques are used to efficiently radiate RF energy to drive down the Signal-to-Noise-Ratio (SNR) at targeted RF receivers, lowering the probability of detection and reception of desired signals. These techniques may deny communications as well as sensor detections and processing of RF signals. Jammer models include broadband, sweep, and pulse jammers.

Weapon Systems

Weapon systems may be modeled at various resolutions. One is typically focused on representing the impact of the weapon system on various platforms based on the platform type, flight path, and flight time within the weapon's range. Characteristic sets of parameters for each weapon type can be used to specify their capabilities. This allows one or two basic models to support the representation of multiple weapon types. Weapons typically ride on a platform that may contain sensors, C2 and communication systems, and may be limited by fuel and PNT receptions. Weapon platforms may be launched from moving host platforms.

When modeling a weapon system, e.g., a red IADS, one may model the system sensors, the C2 centers, and the weapon launchers as well as the weapons. The sensors, launchers, and weapons may contain their own C2 models. Weapons ride on platforms with movement models and may be limited by fuel. They may have their own sensors, POS-NAV, and communication systems commensurate with their level of sophistication.

MOVEMENT MODELS

The major difference between platform types is their movement. Airplanes, surface vessels, ground vehicles, helicopters, and satellite platforms generally have different movement models. Movement models reside within platforms, moving the platform according to a predetermined path at specified times, or based upon messages received or C2 inputs. Different platforms can use the same or different paths.

Satellites generally move in predefined orbits, and these orbits are defined by their paths, including the orientation of the satellite as it moves along its path. When moving satellites, it is convenient to have the 6 DOF+ vector pre-calculated for speed since the orientation angles are generally predefined (although different) at each point along the path. Even though the orientation angles are stored, they can still be changed by the movement model associated with the platform in the special case where they may be changed by a message or C2 model.

Ships (surface vessels) generally have less DOF since their orientation angles are aligned with their direction of movement. One only needs to compute the pair of (LAT, LON) points to determine bearing and their direction. However, it is convenient to have the bearings pre-computed for a predetermined path for speed and accuracy on the WGS-84 globe.

Ground vehicles may have to follow terrain. In this case, given (LAT, LON) or (REL_X, REL_Y) points, it is convenient to store the terrain elevations with the path coordinates for speed. Orientation of ground platforms may be unnecessary since they will generally be aligned with the direction of the path.

Airplanes and helicopters will generally have 6 DOF+ movement, though the orientation of most flights will be aligned with their direction along a predetermined path. However, for analysis of connectivity, one may want to create paths with predefined roll angles to determine at what roll angle connectivity is lost. This is independent of banking in turns, although this can also be pre-computed.

Based upon prior efforts, each of the above types of paths are very difficult to create without tailored interactive path creation facilities, and control file inputs that are small compared to the resulting path files with all of the necessary points for movement. We note that run-time speed is a critical factor in defining the databases to be created in advance to support platform movement. However, creating scenarios can take days instead of minutes without a well designed interactive path creation and modification model as described below.

DEFINING PATHS

To define a *path*, one starts with a set of points, where platforms may follow "shortest distances" between those points while meeting other constraints. Consider the following examples. In the case of a path in the air, a plane may want to maintain a constant altitude above the earth. In the case of a ground vehicle, it will have to remain on the terrain. A surface vessel at sea will remain on the surface of the water. Unless it is geo-stationary, a satellite will follow a prescribed orbit relative to the earth.

If a flight path goes from one city to another, and then to another, sharp turns may be required. These turns may be defined simply by points along the overall path, where the path changes direction. Figure 13 illustrates sets of points where a path follows a given direction, then changes to another direction, and then to another, etc. We define these points of directional change as *pathpoints*, in that they define the overall path of a platform.

Given a set of pathpoints on the globe, one will likely have to break up the subpaths between them. There are two reasons for this. First, the distance between pathpoints may be long, see for example Figure 14. Yet, movement is accomplished by moving an icon from one point to another, and large jumps will appear unrealistic. Also, an airborne platform may have to maintain a constant altitude above the earth's surface. If it flies along intermediate points on a straight line between the path points, it will be changing altitude along the way, maybe going into the earth. In the case of a ground vehicle, it will have to follow terrain over short distances.

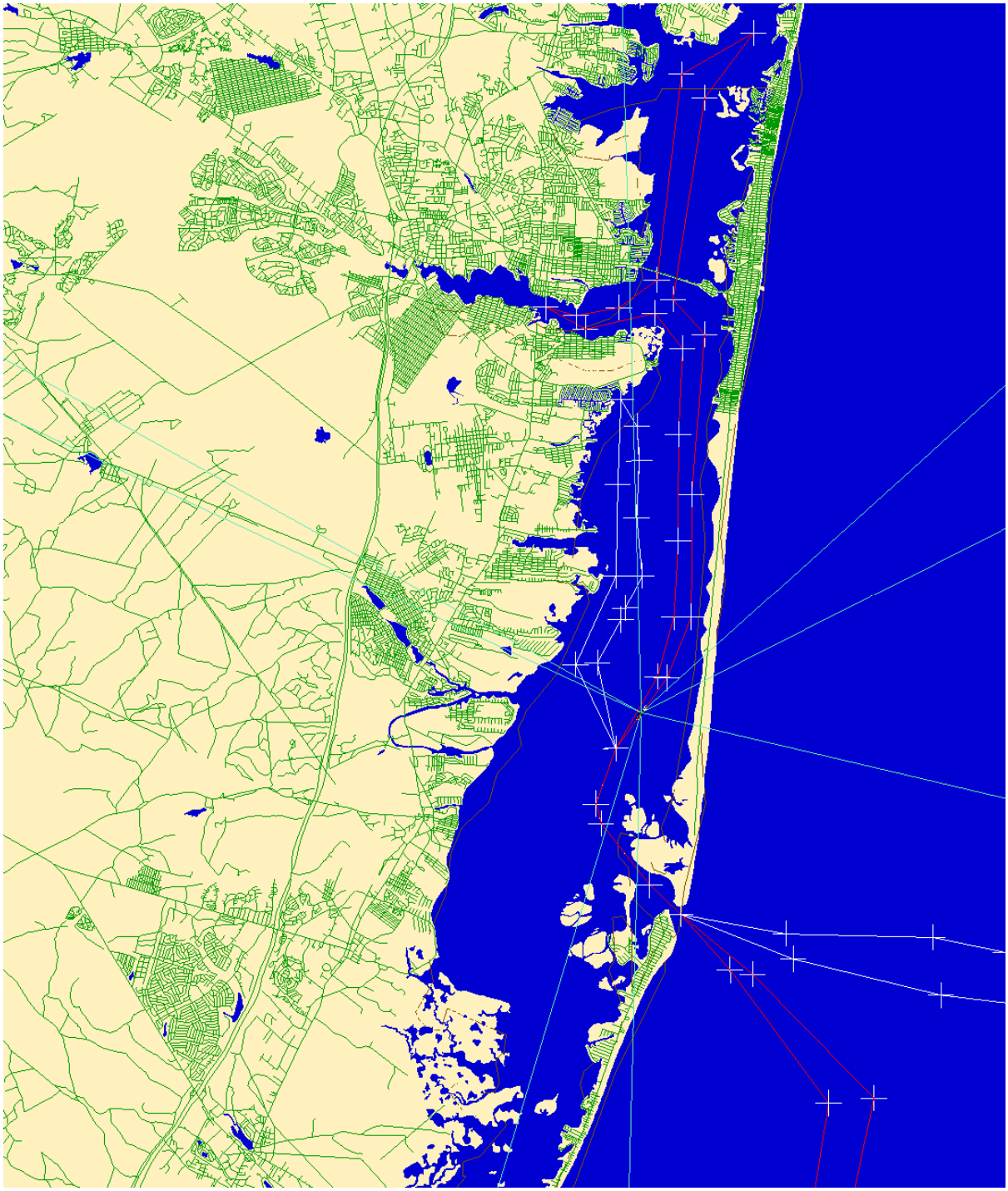

Figure 13. Pathpoints.

Second, when platforms are moving between two distant points over the surface of the earth, they must adjust their bearings to follow the shortest path, staying at a given altitude above ground or sea level. This implies that intermediate points must account for changes in bearing and altitude. These are termed *waypoints*. The distance between waypoints will depend upon the path and the application. Figure 14 may have many waypoints between the path points.

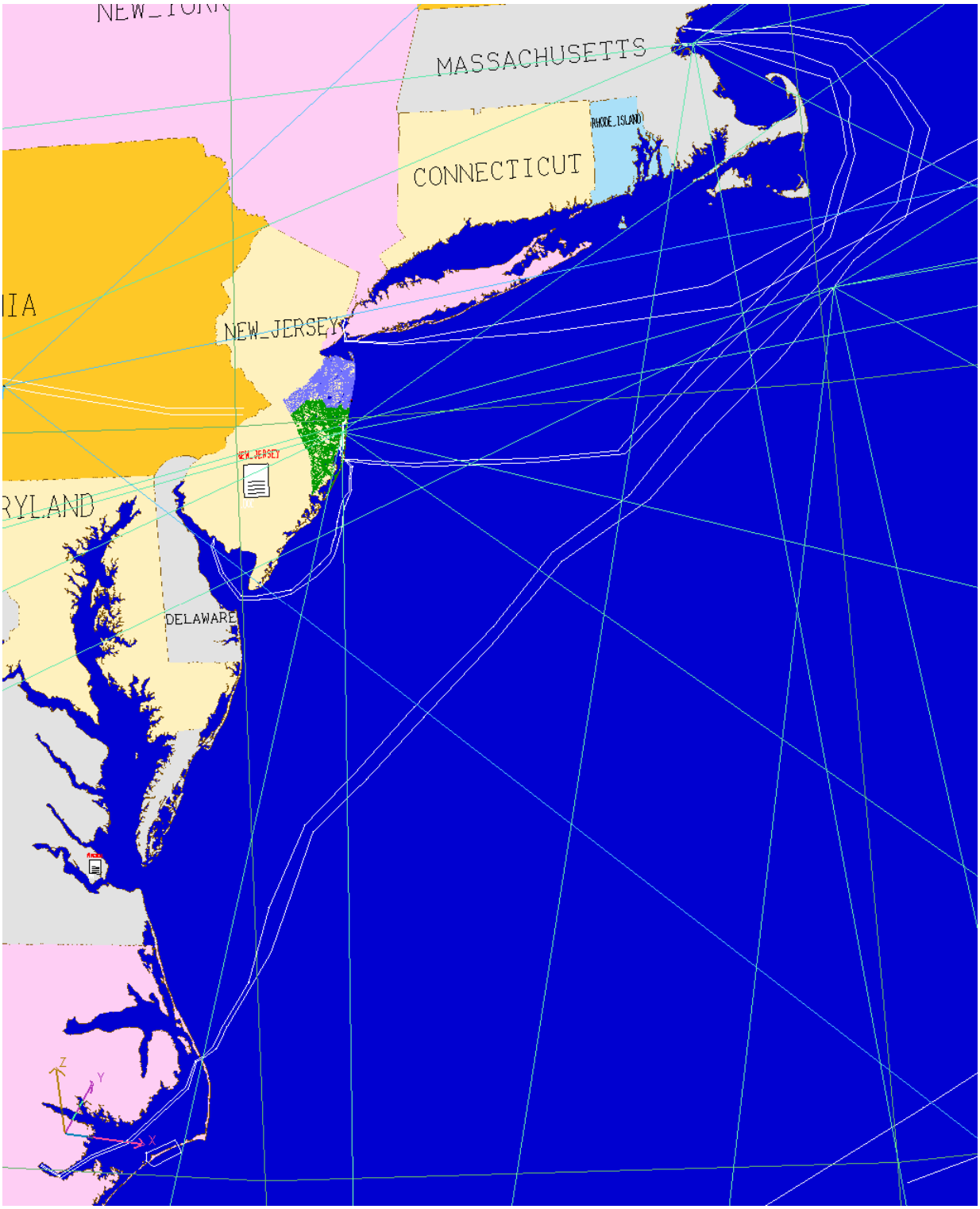

Figure 14. Satellites connected to platforms on paths.

In Figure 14, waypoints that are more than 100 Km apart will likely require a new bearing for each waypoint. This implies breaking up the path to obtain waypoints that are on the order of 100 Km apart, requiring 10 waypoints for 1000 Km, see Figure 15.

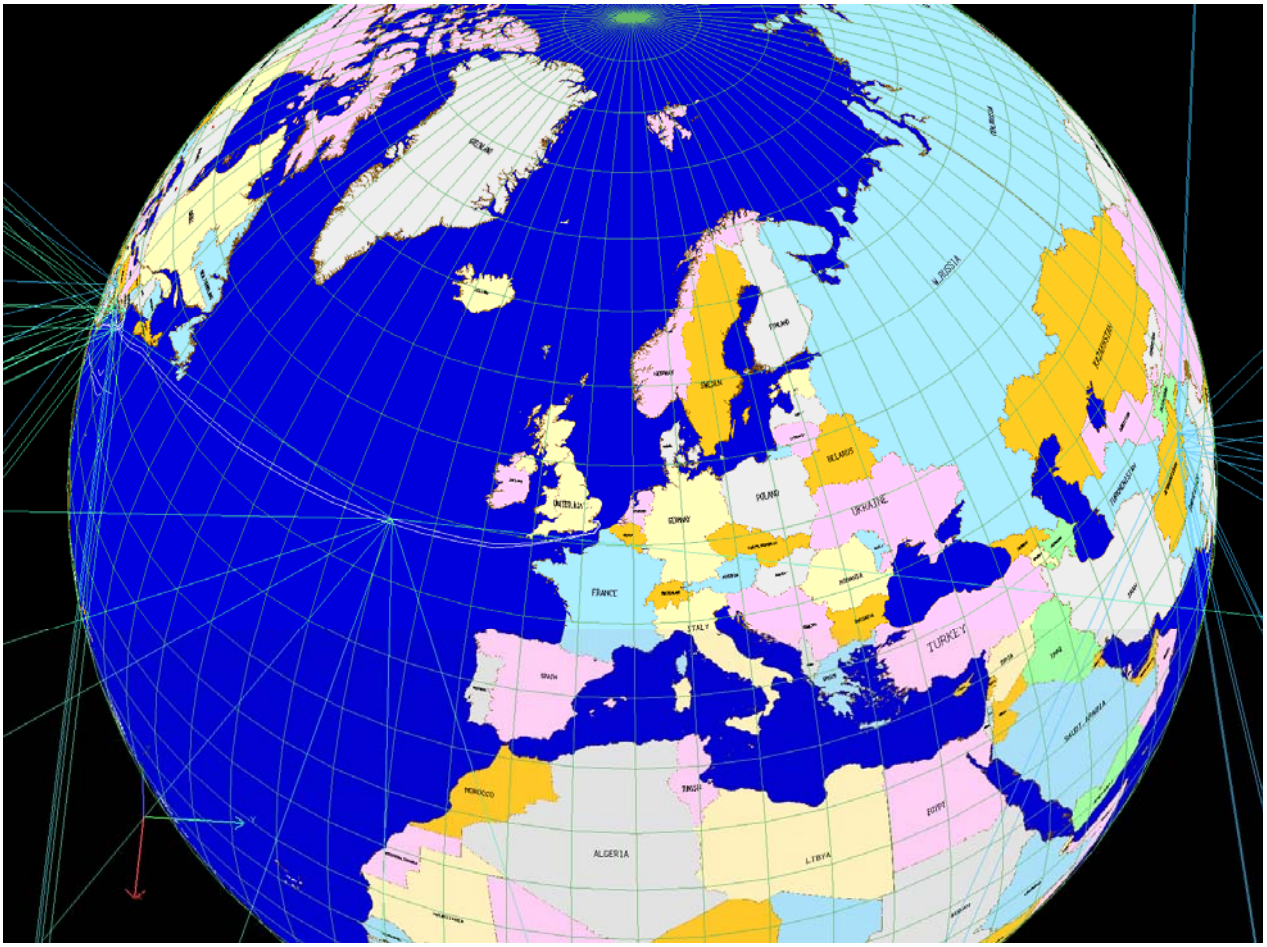

Figure 15. Longer paths on the globe.

If the distance between waypoints is less than 100Km, changes in bearing are likely unnecessary. However, it is also unlikely that one wants to move a platform in 100Km jumps. Even 10 Km jumps may be large depending upon the scenario. Thus it is likely that one needs additional *movepoints* between the waypoints as defined above. For example, one may want to move a platform in 1Km moves. Again, this will depend upon the platform, path and application.

When creating or modifying paths, one wants to click on the pathpoint and move it, or add points in between with a double click. When there are many paths that cross, e.g., in Figure 1 or Figure 13, it may be hard to select the desired pathpoint to be changed. This is resolved by highlighting the desired path (in red in Figure 13) to pick the desired points from those that are close together. An interactive path model with these features dramatically reduces the time to build scenarios (from days to minutes).

To summarize, depending upon the path, the analyst may have to define *pathpoints* where the path changes direction. Between the pathpoints, the analyst may require *waypoints* to provide for changes in bearing or other factors, e.g., maintaining a given altitude above the surface of the earth. In between waypoints, one may require *movepoints* to meet the dynamic visual requirements of a scenario.

Automatic Path File Generation

To generate path files automatically, the analyst may enter the pathpoints interactively by clicking a sequence of pathpoints down with the mouse. In addition, information on altitude (ASL or AGL) may have to be entered as it applies. For example, one may want to maintain the same altitude or change altitude at specified points. One may also want to enter orientation angles at a point, e.g., to represent a "bank" or roll. A simulation of planes flying around mountains quickly demonstrates these features. When planes flew in formation, doing rolls at predetermined points as they flew along the path, their actions are realistic. This is often observed at an Air Force base as planes do 90° rolls in formation while flying straight line or slightly curved paths.

Paths between pathpoints may be broken up automatically into waypoints. Additional information may be entered, e.g., if one wants to change altitude, velocity or orientation at a given point. Otherwise, these may be generated automatically when there is no change. This automatic breakup will follow great ellipsoids around the earth, placing waypoints when there is a sufficient change in the bearing angle, or at points within a predetermined maximum distance. At the same time, the bearing and distance between points will be calculated and put on the file.

Paths between waypoints may be broken up automatically into movepoints. One may also want to change altitude, velocity or orientation at a give point. Otherwise, these may be generated automatically when there is no change.

Distance And Velocity Along A Path

Velocity is either constant along a path or changing along a path. If it is constant for the entire path, then one can store the fixed velocity with the platform. The data in the path file, e.g., distance between points, can then be used to schedule the next move event, whereby the platform moves to the next point at a given time. To schedule the next move, one must know the distance to the next point and the velocity. The following cases must be considered when designing the models to support movement.

1. Velocity is constant along a path for every platform that uses that path.
2. Velocity is constant along a path but may be different for some platforms that use that path.
3. Velocity changes along a path depending upon the position, but each platform that uses that path follows the same trajectory (same velocity changes for each platform).
4. Velocity changes along a path depending upon the position, and some platforms that use that follow different trajectories (different velocity changes for each platform).

Based upon prior simulations, cases 1 and 3 are the cases of most use. In these cases, it is much more simple and much faster to calculate the distances, and schedule times (delta-times) in advance and store them with the path database to be used by the movement model for the platforms. This renders the movement models to be very simple. All they have to do is schedule the next move in the specified future delta-time. Calculations are unnecessary.

In cases 2 and 4, the velocity changes may only occur part of the time, and can be done in the movement model as an override to the default in the path file. In these cases, the corresponding calculations for scheduled delta-time must be done there as well.

Interactive Path File Generation

It is anticipated that the analyst will want to enter the path points interactively by clicking a sequence of pathpoints down with the mouse. Note that these points will be treated as pathpoints, with waypoints and movepoints being generated automatically after the pathpoints are entered. As pathpoints are entered, the analyst will have the option to enter altitude, velocity, and rotation angles for selected points. If this data is not entered, the data in the prior point will be used as the default.

When paths are generated automatically, readable files are produced that are easily changed on a point-by-point basis. Such a file is illustrated in Figure 16. Fields not shown in this figure include maximum waypoint and movepoint distances, bearing, and path name.

```
*GLOBE DEMO PATH FILE DATE: 2011/05/02
*
* WGS Coordinates (LAT, LON) are defined as follows:
* N99 99 99.9 E999 99 99.9  =  Ndd mm ss.s Eddd mm ss.s
* +99 99 99.9 +999 99 99.9  =  Ndd mm ss.s Eddd mm ss.s
* S99.9999999 W999.9999999  =  Sdegrees real Wdegrees real
* -99.9999999 -999.9999999  =  Sdegrees real Wdegrees real
*
*Altitude = meters
*Velocity = miles/hour
*Rotation = degrees (PITCH, ROLL, & YAW are incremental angles off the bearing)
*MAX_DISTANCE = 1 Km
*
*      PATH   WAY  MOVE                                POSITION   DISTANCE     ROT_X ROT_Y ROT_Z
*PATH POINT POINT POINT      LAT    LON    ALT     X      Y      Z NEXT PT VEL PITCH  ROLL   YAW
*PATH 1
0001   0001  0001  0001  +39.003 - 75.105   20   0.0   0.0   0.0    20    22   0.0   0.0   0.0
0001   0001  0002  0001  +38.901 - 75.004   20   0.0   0.0   0.0    20    22   0.0   0.0   0.0
0001   0001  0003  0001  +38.905 - 74.850   20   0.0   0.0   0.0    20    22   0.0   0.0   0.0
0001   0001  0004  0001  +38.908 - 74.705   20   0.0   0.0   0.0    20    22   0.0   0.0   0.0
0001   0001  0005  0001  +38.903 - 74.653   20   0.0   0.0   0.0    20    22   0.0   0.0   0.0
0001   0001  0006  0001  +38.953 - 74.600   20   0.0   0.0   0.0    20    22   0.0   0.0   0.0

                                          .
                                          .
                                          .

0001   0002  0018  0002  +39.831 - 74.108   20   0.0   0.0   0.0    20    22   0.0   0.0   0.0
0001   0002  0018  0003  +39.810 - 74.123   20   0.0   0.0   0.0    20    22   0.0   0.0   0.0
0001   0003  0001  0001  +39.794 - 74.131   20   0.0   0.0   0.0    20    22   0.0   0.0   0.0
0001   0003  0002  0001  +39.788 - 74.129   20   0.0   0.0   0.0    20    22   0.0   0.0   0.0
0001   0003  0003  0001  +39.771 - 74.111   20   0.0   0.0   0.0    20    22   0.0   0.0   0.0
0001   0003  0004  0001  +39.763 - 74.100   20   0.0   0.0   0.0    20    22   0.0   0.0   0.0
0001   0003  0005  0001  +39.743 - 74.063   20   0.0   0.0   0.0    20    22   0.0   0.0   0.0
0001   0003  0006  0001  +39.708 - 74.045   20   0.0   0.0   0.0    20    22   0.0   0.0   0.0
*PATH 2
0002   0001  0001  0001  +39.850 - 74.101   10   0.0   0.0   0.0    10    22   0.0   0.0   0.0
0002   0001  0001  0002  +39.843 - 74.115   10   0.0   0.0   0.0    10    22   0.0   0.0   0.0
0002   0001  0001  0003  +39.810 - 74.123   10   0.0   0.0   0.0    10    22   0.0   0.0   0.0
0002   0001  0001  0004  +39.794 - 74.131   10   0.0   0.0   0.0    10    22   0.0   0.0   0.0
0002   0001  0002  0001  +39.788 - 74.129   10   0.0   0.0   0.0    10    22   0.0   0.0   0.0
```

Figure 16. A sample path file.

Interactive Path File Creation & Modification

In addition to automatically generating waypoints and movepoints from pathpoints, one would like to create paths interactively using a graphical interface such as that shown in Figure 17. It is clear from the shape of points in the figure that they are all defined by pathpoints. Once these paths are created in terms of pathpoints, their waypoints and movepoints can be generated automatically.

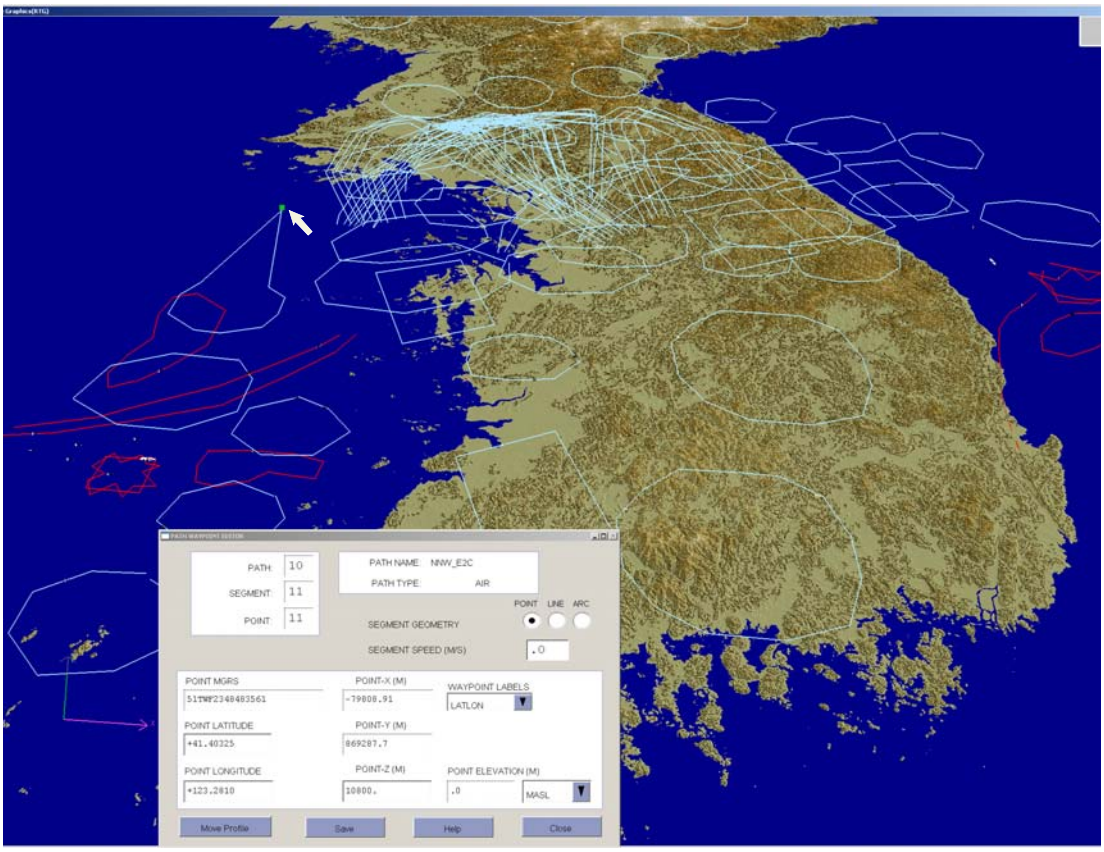

Figure 17. Interactive creation and modification of paths.

When the analyst indicates that he wants to create or modify pathpoints, a panel will be presented with options, e.g., selecting the waypoint distance for the path so that the waypoints can be generated automatically when the distance between pathpoints exceeds the waypoint distance. Similarly, the movepoint distance may be entered so that movepoints can be generated when the distance between waypoints exceeds the movepoint distance.

When platforms are moving on terrain, one must provide the altitude AGL if required by equipment, e.g., when an antenna resides on the platform at a given height AGL.

Satellite Path File Creation & Modification

Satellite path data is typically available as a function of date and Time Of Day (TOD). Depending upon the type of orbit, satellite paths can be generated automatically. Geostationary paths are easily modeled for reasonable accuracy. PSI has models to represent the GPS constellation at any point in time. As illustrated in Figure 18, users may view the GPS satellite constellation showing connections to receivers at the surface of the earth as well as other GPS satellites. Connectivity is modeled accurately, accounting for terrain on the earth as well as pertinent radio characteristics.

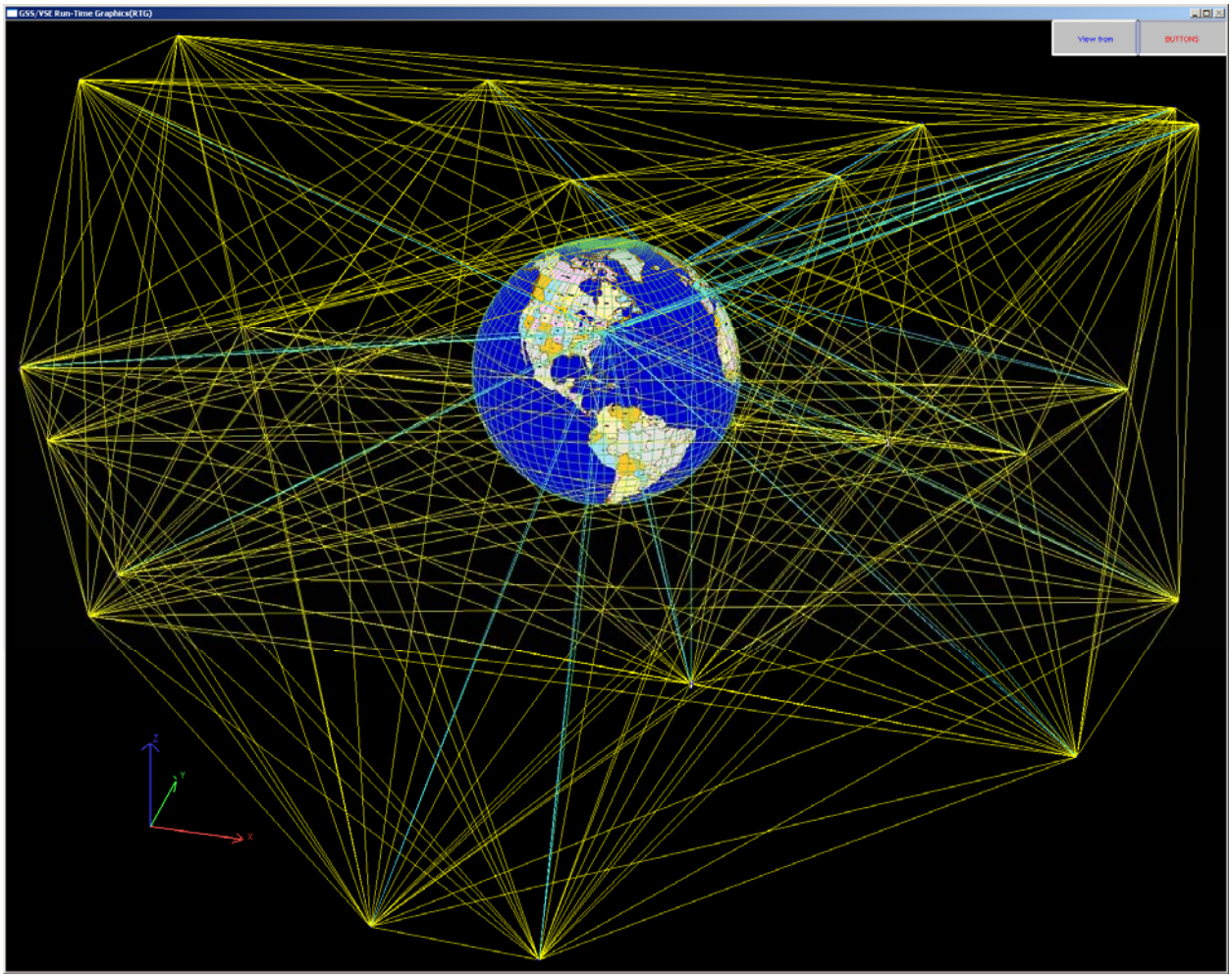

Figure 18. GPS satellite constellation.

C2 MODELS

The C2_MODEL operates on events, putting events in its event queue, and popping off events when they are to occur. It may also translate an event into a message or message sequence. Conversely, it may translate incoming messages into events to occur immediately or in the future.

Mission Event And Message String Sequences

When generating mission threads and message strings, various input files must be built. A notional illustration of how such model architectures are intertwined is shown in Figure 19. Communication requirements models must read in files to determine who must talk to whom and when. This includes files that determine the messages and nets that are used.

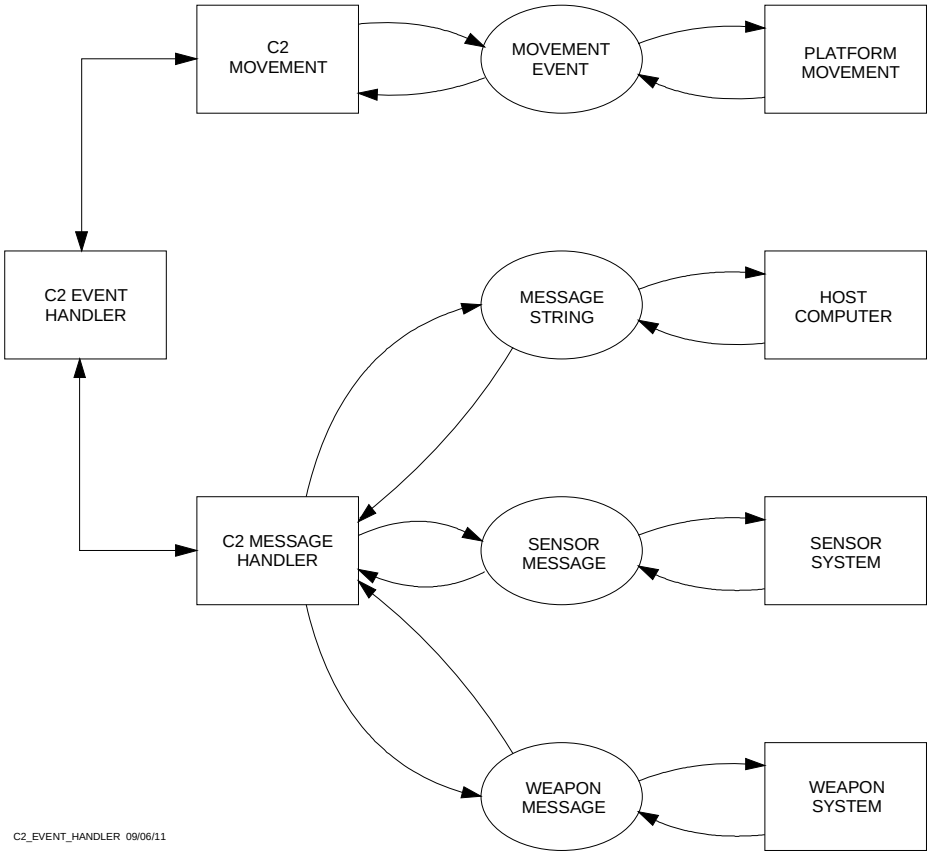

Figure 19. Illustration of intertwining of Message Strings and Mission Threads.

C2_MESSAGE_HANDLER Model

All platforms communicate via radios or other types of equipment via the appropriate environment models. Communications is achieved using messages as specified by the missions. The C2_MESSAGE_HANDLER takes in messages from the MISSION_ MANAGEMENT_ MODEL and distributes them to the HOST_MODELS. Messages received from the HOST_ MODEL may cause new events to be generated or messages to be sent from the same or different host. The C2_MESSAGE_HANDLER model may start the host on a script, terminate the script, or start, modify, or stop a statistical message generation process.

C2_MOVEMENT Model

The C2_MOVEMENT model will receive events and path designators from the MISSION_ MANAGEMENT_MODEL, and store these for future use. It may send actions to the MOVEMENT model to identify a path to be used, start movement, move to a new path, or stop. It will provide a starting point on the designated path. Points along the path may generate events that are sent back to it, and these events may be sent on to the C2_EVENT_ HANDLER for further processing. The MOVEMENT model may create new paths based on instructions from the C2_MOVEMENT model.

MISSION THREADS

Mission threads provide for scripted or dynamically generated events that cause subsequent *dependent* events. For example, one could script "target sighting" events that occur at predetermined times during the simulation. The scripted events are treated as *independent* of what is going on in the scenario. They can be used to cause other dependent actions to occur, e.g., messages being sent from a host on an ISR platform to a C2 center. Events can be sequenced in time so as to model the effects of a complex operation that is not explicitly modeled. Figure 20 illustrates a scripted sequence of independent events as a function of time.

```
***  SCRIPTED EVENT FILE
***
*EVENT#      TIME     EVENT              PRIORTY
 000001    001600     INIT_PLATFORM            4
 000002    001610     SEND_MESSAGES            2
 000003    001650     LAUNCH_WEAPON            4
 000004    001651     DESTROY_TARGET           0
 000005    001653     GENERATE_STATS           4
 000006    001700     STOP_MESSAGES            4
```

Figure 20. Example of a Scripted Event file.

Scripted events can be recognized in a model that triggers subsequent events as the result of their occurrence. Thus, the occurrence of event DRAM21 can cause events OUTJOK and DRAM39 to occur at specified delta-times in the future. An example is shown in Figure 21. Thus, a mission thread model senses different event types and causes other events to occur at predetermined times in the future. The information containing the subsequent events and times of occurrence can be put into a *mission thread* table that defines the sequence of events of a mission. Note that these events may be affiliated with messages.

```
***  MISSION THREAD TABLE
***
*EVENT#    EVENT  EVENTOUT   E/M  DELTA_T    PRIORTY
 000001   DRAM21    OUTJOK    E          0          4
 000002   OUTJOK    DRAM39    E          2          2
 000003   DRAM39      A_OK    M          0          4
 000004     A_OK    WRAPUP    M          4          4
 000005   WRAPUP    DRAM55    E          0          0
 000006   DRAM55      EXIT    E          4          4
```

Figure 21. Example of a Mission Thread file.

The scripted event file, Figure 20, is read in during the course of the scenario, causing events to occur at the times specified. The mission thread file, Figure 20, is read in during initialization to build a table that determines which events are generated when other events occur.

MESSAGE STRINGS

Message strings define messages that, when received, cause other messages to be sent. Figure 22 provides a simplified illustration of a Message String Table. Messages may be assigned to destinations or (in this example) to multiple nets. Depending upon the source (or input net), particular destinations (or output nets) are assigned to receive the messages. Message strings may be initiated by events and some messages may cause events to occur. Message string tables can be built independent of the scenario, and read in during initialization of the simulation.

```
***  MESSAGE STRING TABLE
***
*MSGID   INTYPE  NET_IN  OUTYPE  E/M PRIORTY   NET_1   NET_2   NET_3   NET_4   NET_5   NET_6
000001   DRAM21      22  OUTMES   M         4      30      35
                         WINGIT   M         4      60      65
000002   OUTMES     501  DRAM39   E         2     220     222      30       5
000003   OUTMES     601  DRAM39   E         2     320     322      30       5
000004   DRAM39      24    A_OK   M         4      62      55
000005     A_OK      30  WRAPUP   M         4     309      40
                         KILLIT   M         2     670
000006   WRAPUP     330  DRAM55   E         0     220     222      30       5
000007   WRAPUP     430  DRAM55   E         0     320     322      30       5
```

Figure 22. Example of a Message String Table.

Scripted event files, as in Figure 20, or dynamically generated events can be used to initiate messages by a host unit or subscriber. In this case, a message is generated based on inputs from outside of the network - a different input to the host. Messages generated from the message string table may be initiated based upon messages coming in from the network.

C2 MODEL ARCHITECTURES

Figure 23 illustrates how a C2_MODEL may fit into a simulation architecture where there are large numbers of complex synchronized events and messages that are intertwined. This is a very simplified illustration of a GSS model designed to handle the decision processes that determine the sequence of events. It is intended to serve to determine: (1) what resources in what models will store what data structures; and (2) what processes in what models will perform what operations. The models are outlined below.

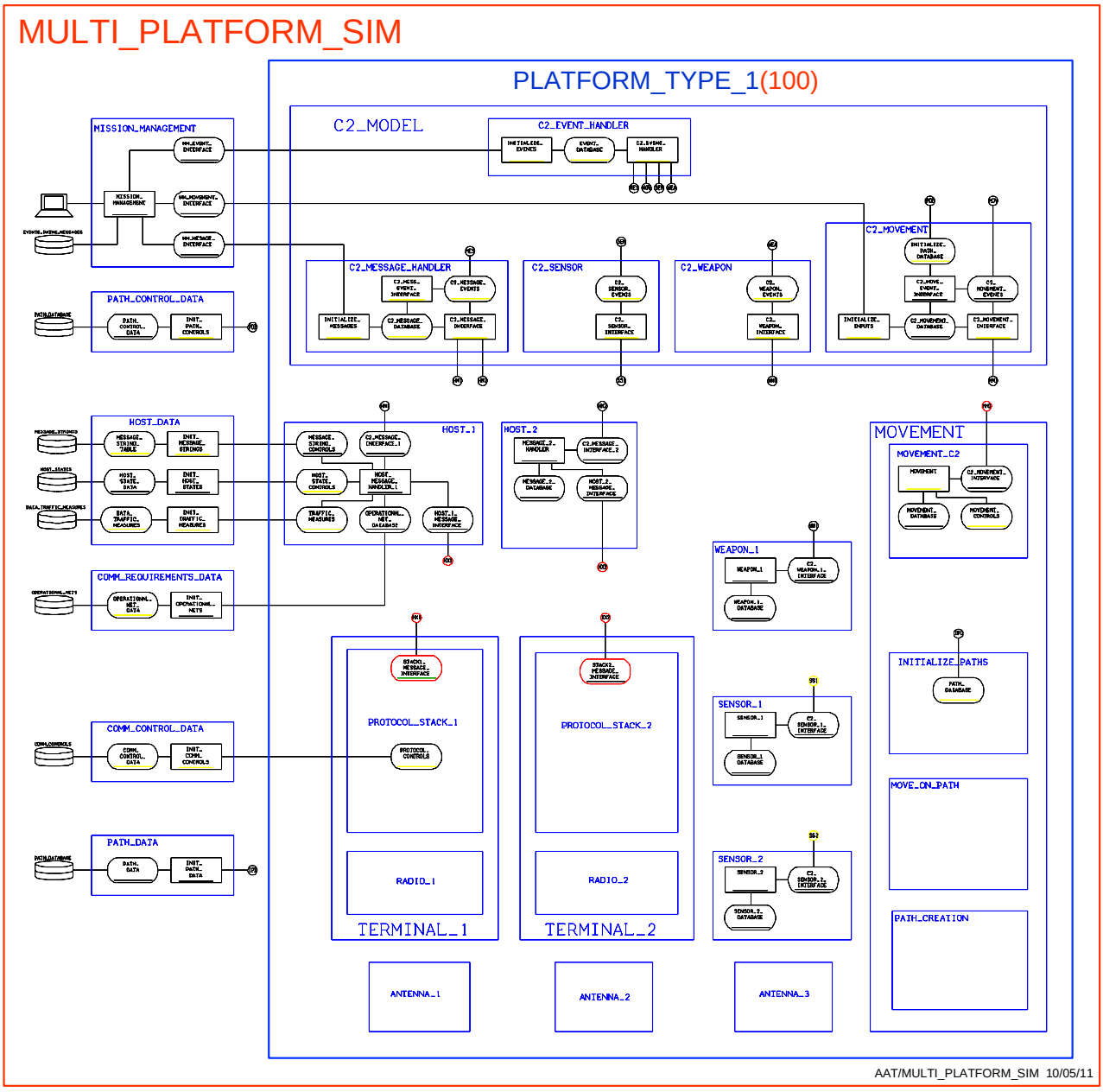

Figure 23. C2 and MOVEMENT models within a platform model.

MISSION_MANAGEMENT MODEL

The models shown in Figure 23 are a characterization of what may be in an interactive planning tool or a fast analysis simulation. In the planning tool, a MISSION_MANAGEMENT model will interface with the user to create path files, event files, and scripted traffic files. This model takes the databases created and initializes the platform model. Some of these databases may reside in the C2_MOVEMENT model or the C2_MESSAGE_HANDLER model. For example, the current MOVEMENT_MODEL may have a flight path stored within its database that was specified for use by the C2_MOVEMENT model.

ENVIRONMENT MODELS

PROPAGATION MODELING

PSI has been developing models for fast and accurate Electro-Magnetic (EM) wave propagation since the early 1980s. These models cover the spectrum from HF, and VHF-UHF to Optical Line-Of-Sight (LOS). These models use standard World Geodetic System (WGS-84) Digitized Terrain Elevation Data (DTED) with varying degrees of resolution. An illustration of PSI's 3D terrain data is shown in Figure 24, taken from the mountainous region of northern Afghanistan. PSI's propagation models can cover extremely large areas to support scenarios covering major parts of the globe with fast computations producing high degrees of accuracy.

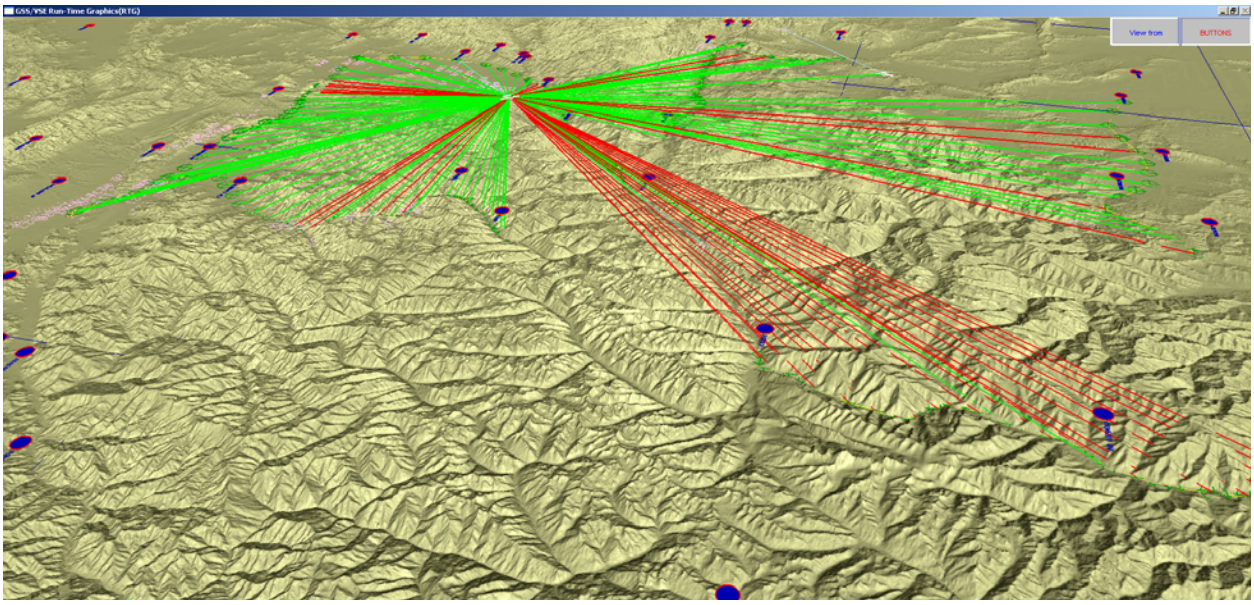

Figure 24. Illustration of propagation modeling to obtain connectivity.

Figure 25 illustrates the effect on GPS position accuracy of platforms on or close to the ground in Afghanistan. These models account for terrain in determining the number of satellites being received as a function of time-of-day. Note that the paths shown in the multiple colors used to determine the number of satellites received are on the terrain or at some height above the terrain. When these paths are generated using the path model, they are placed at a given height above the terrain. This implies having access to the terrain elevation database when generating these paths.

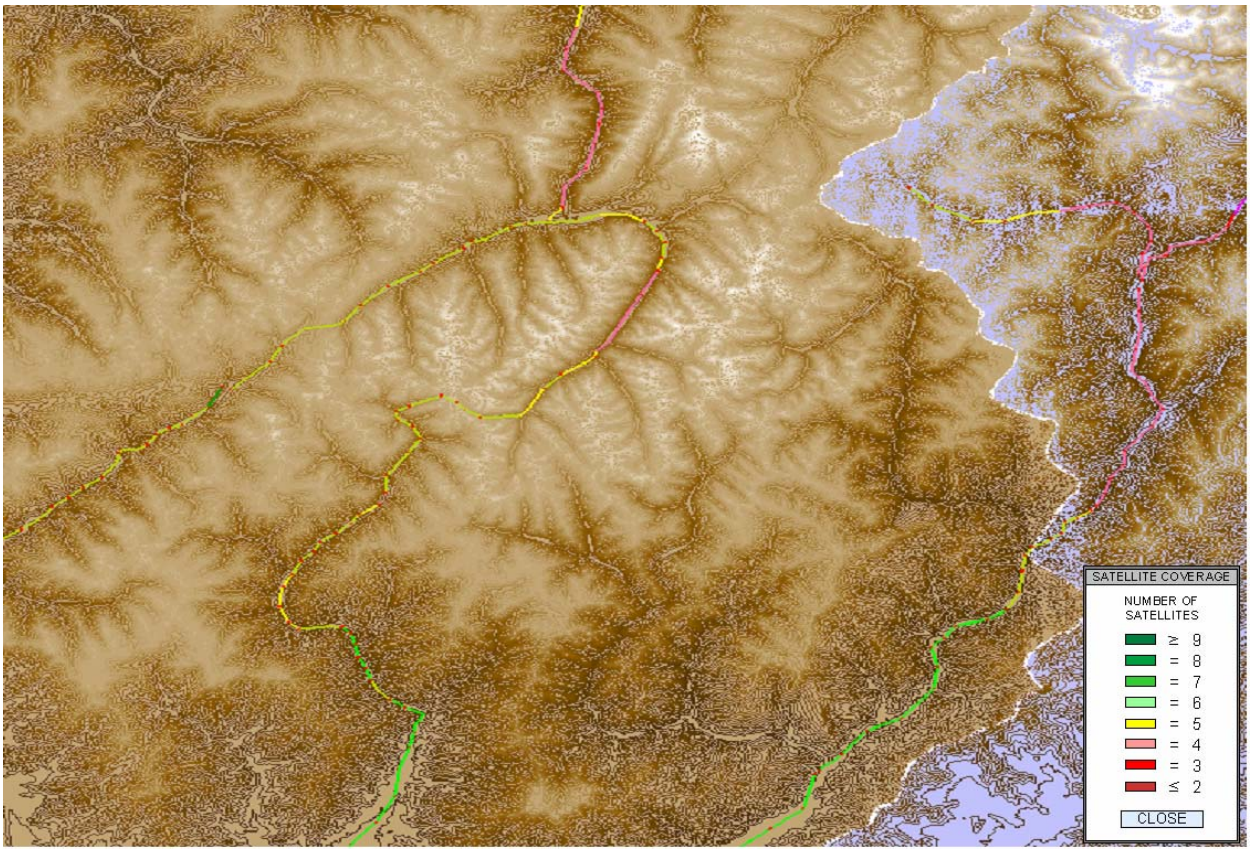

Figure 25. Ability to receive GPS satellites on or above roads at a give time in Afghanistan.

The Wide Area Terrain (WAT) System

Figure 26 illustrates the graphical environment for selecting REL zones to support the scenario area of interest. In this case, the figure illustrates a large theater area. In fact, it could be larger, covering major parts of the globe. The grid lines in the picture denote the 6° X 8° REL zones defined by the WGS-84 spheroid standards for DTED data. Each zone represents a Cartesian coordinate system to allow fast LOS or wave propagation calculations in X, Y, Z coordinates. The area covered by DTED need not be contiguous, but need only exist to support those areas where the calculations are needed.

The WAT system can be automated to support interactive selection of zones on the globe using the mouse, whereby a panel is presented for a selected zone indicating what data already exists in the WAT library and the corresponding resolutions for that data. The analyst may chose to produce additional databases at different resolutions using the interactive panel facilities.

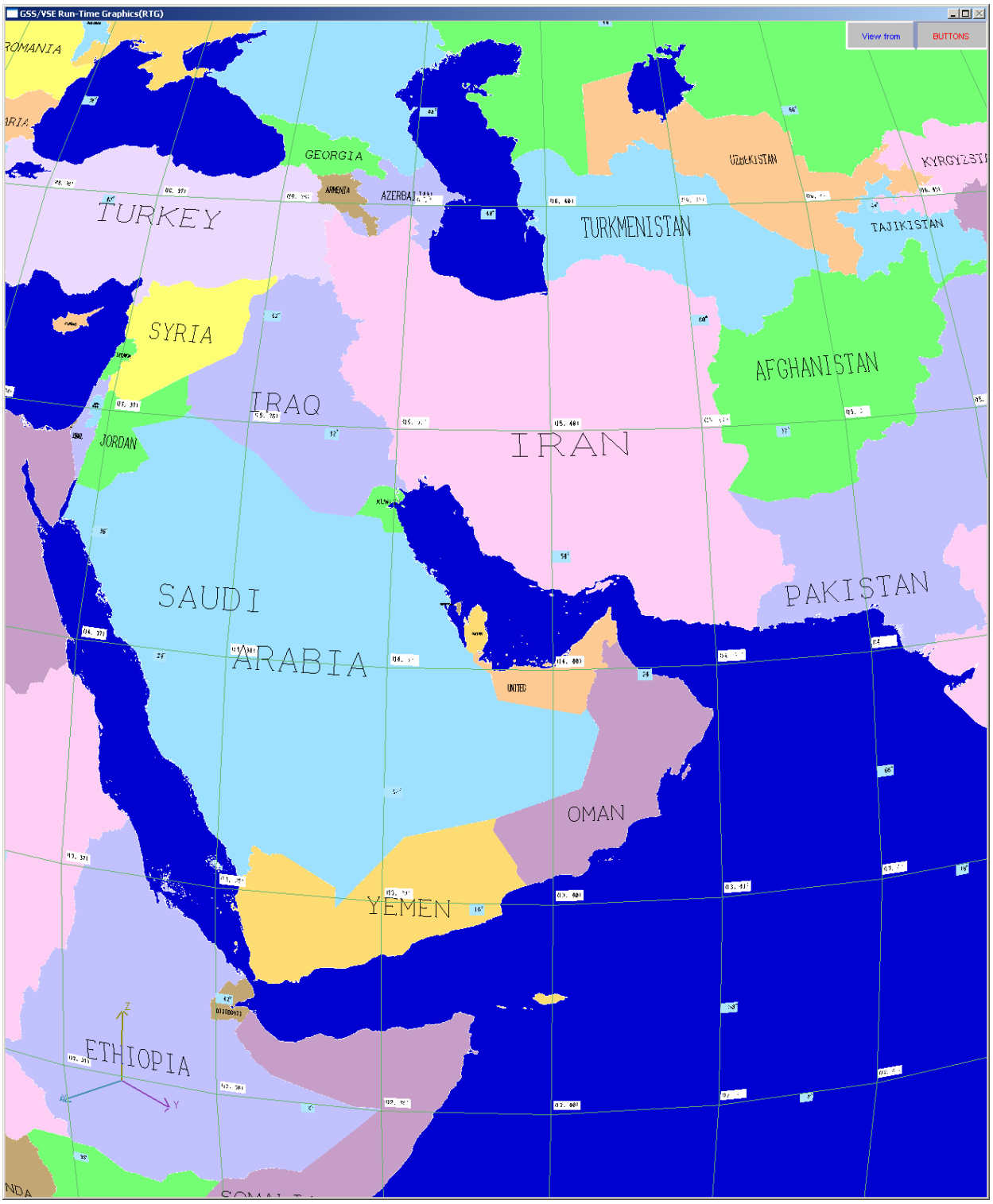

Figure 26. Estimating the coverage of a REL-NAV system.

DESIGNING ARCHITECTURES FOR SPEED

There are multiple considerations when handling the movement of platforms. These are the following:

- Speed on a single processor without graphics
- Speed on a single processor with interactive graphics
- Speed on a parallel processor without graphics
- Speed on a parallel processor with output graphics

Each of these cases is described below.

Single Processor Speed Without Graphics

This is likely to be the most stringent requirement on speed until moving to parallel processing. Maximizing speed requires that all of the computations required to accurately move a platform are minimized. This implies doing all computations that can be done prior to starting the simulation. For example, all of the parameters required for movement along a path may be done in advance, including the schedule times if desired.

This is particularly important when moving over the globe, since each move potentially requires a significant computation. This requires storing all of these parameters in the path database for access each time a move is scheduled to occur.

When platforms have radio transmitters and receivers, connectivity must be updated when a platform moves. This determination is best done by storing a connectivity column vector in each receiver as opposed to updating a shared connectivity matrix. The approach for doing this is described in **Building Reusable Models and Large Scale Simulations Using GSS**, Prediction Systems, Inc., Spring Lake, NJ, 16 APR 2011.

Single Processor Speed With Interactive Graphics

When drawing moving platforms on the screen, one must turn the scene off, move the platform, and turn the scene back on. Turning the scene off and on takes time, so all platforms that have been moved since the last draw should be moved together. Ships move slowly compared to airplanes and satellites, so all platforms need not be updated graphically together. This implies that the graphics scene will be updated at the rate necessary to handle the fastest platform movement. More rapid updates may be done without the slower updates. This may be done by putting an entry into an RTG_MOVEMENT_DATABASE in an RTG_INTERFACE module.

When platforms have connectivity lines between them, these lines will have to be updated when a platform moves. This determination is best done by each receiver as described in the prior section, i.e., using the connectivity column vector described in **Building Reusable Models and Large Scale Simulations Using GSS** referenced above. When these lines are updated, they may be put into an RTG_CONNECT_ DATABASE. This is illustrated in the parallel processor version of the MULTI_PLATFORM_SIM in Figure 27.

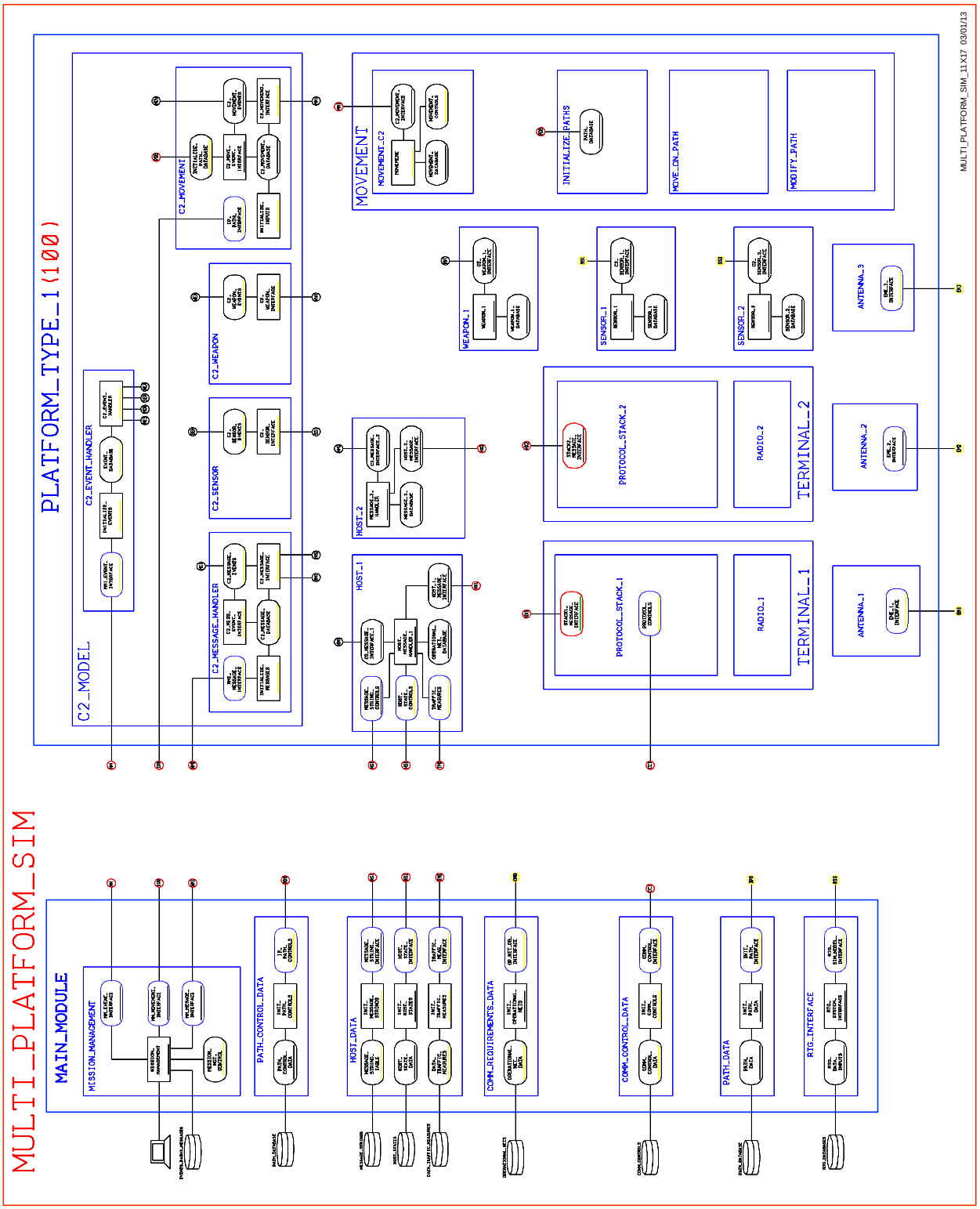

Figure 27. MULTI_PLATFORM_SIM for a parallel processor.

The entries put into the RTG_MOVEMENT_DATABASE and the RTG_CONNECT_ DATABASE in the RTG_INTERFACE module provide a list of elements to be drawn, i.e., ICONS and LINES. This list must contain the ICON names both for the icons to be moved and for the lines to be drawn between them. These ICON names must be obtained from the platform database in the simulation using the platform type and the corresponding instance. These names can then be used to obtain the RTG_ICON_POINTERs. This information may be shared with a DRAW_PLATFORM_MOVEMENT module and a DRAW_CONNECTIVITY_LINES module, both within an RTG_INTERFACE module. This module will draw the scene periodically using the updates in the databases. If nothing has changed, then no changes will be drawn.

Parallel Processor Speed Without Graphics

When using parallel processors, one wants to avoid centralized two-way databases (that each process can update) shared across multiple processors. This is accomplished by putting the initialization modules and graphical output modules on separate processors. Since the initialization provides one-way inputs into databases within the instances, these may be initialized prior to the run-time. Between radios, it is best to use the type of scheme used to track connectivity described above. This requires additional schemes to ensure data coherency between parallel processors as well as those described in the prior section, i.e., using the connectivity column vector described in **Building Reusable Models and Large Scale Simulations Using GSS** referenced above.

PSI is developing a parallel PC, where a 32 processor PC will run the types of simulations described here faster than a High Performance Computer (HPC) using 250 parallel processors. The many factors that go into this huge increase in speed are described in various papers. But this level of speed is required to run detailed models in complex scenarios of the type needed to do the planning defined here.

Parallel Processor Speed With Output Graphics

The approach to architecture using graphics on a single processor described above also applies to that on a parallel processor. This is accomplished by putting both the initialization modules and graphical output modules on separate processors. The initialization modules are one-way inputs. Since the assessment or analysis simulation is designed for multiple runs in a real-time environment, it will not contain interactive graphical inputs. In this mode, the graphical output modules are one-way outputs. This is why there is a planning simulation