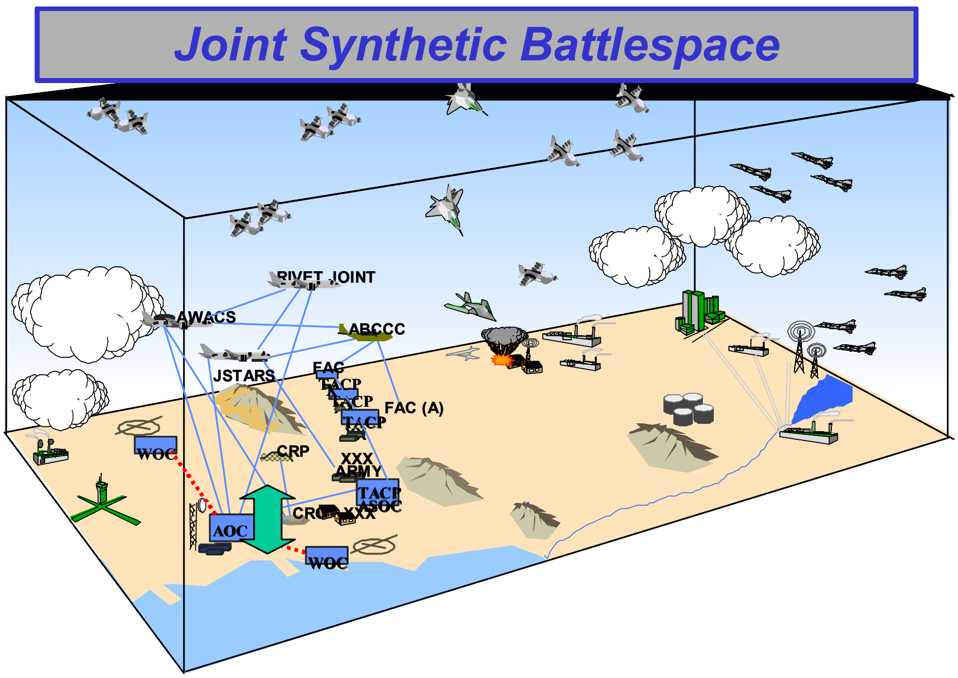
System/Subsystem Design Document
for
The Joint Synthetic Battlespace (JSB) Experiment



Version 1.0
March 5, 2002

Prepared for:
Electronic Systems Command/CXC
Hanscom Air Force Base, MA

DRAFT

System/Subsystem Design Document
for
The Joint Synthetic Battlespace (JSB) Experiment

Approved: [Project Sponsor] ______________________________ [designator] date	Accepted: [Software Support Activity] ______________________________ [Software Project Manager] date
Accepted: [Developing Activity] ______________________________ [Software Project Manager] date	Accepted: [Software Support Activity] ______________________________ [Branch or Division] date

RECORD OF CHANGES

Version Number	Date	Description

TABLE OF CONTENTS

LIST OF FIGURES/TABLES

Figure **Page**

This page intentionally left blank

1. SCOPE

1.1 IDENTIFICATION

The Joint Synthetic Battlespace for Simulation-based Acquisition (JSB (SBA)) initiative is part of the overarching concept for a Joint Synthetic Battlespace-Air Force (JSB-AF) sponsored by Headquarters AFMC (Air Force Materiel Command), the activity responsible for the acquisition for all hardware and software used by operational forces in the Air Force. The JOint Synthetic battlespace Experiment Federation (JOSEF) is an effort undertaken to understand technology gaps and explore requirements.

The JOSEF architecture does not mimic any current simulation architecture as applied within the training, operations analysis or engineering communities. It represents an engineering-based approach to modeling real-world stimuli to real systems – whether they are platforms, weapons, sensors or communications transmitters – within a credible simulated tactical environment.

1.2 SYSTEM OVERVIEW

1.2.1 Problem Statement

There are many problems in current state of the art simulation. In order to support efforts such as the Global Strike Task Force (GSTF) many improvements need to be designed and implemented. There are very few dynamic and credible representations of sensors. Many current sensor models embed in them their own synthetic environments and their own human behavior models. In addition these embedded environments are not dynamic. JOSEF separates the sensors from the environment. All sensors will use the same consistent and correlated environment. Additionally, the embedded human representations will be encapsulated in their own simulation federate.

Coupled with this problem is the fact that most current synthetic environments are not dynamic and are not correlated through the entire electromagnetic spectrum. In order to address this deficiency, JOSEF's Common Synthetic Environment (CSE) will provide a common, correlated and integrated environmental representation to three different legacy models covering the EO/IR/RF part of the spectrum. In addition, many synthetic environments do not have the capability of representing dynamic signatures for the entities contained within them. The CSE will provide dynamic signatures for most of the entities contained within it. These dynamic signatures will be influenced by ephemeral conditions (Sun position, time of day, etc…) and in future experiments by vehicle states. Other hard to model phenomenology, such as RF ducting, and the ability to produce high resolution geo-specific terrain and feature modeling are also addressed.

In order to make scenarios realistic, various types of clutter must be represented. Many current synthetic environments do a poor job of representing object clutter, and JOSEF will provide a varied clutter environment for accurate and dynamic stressing of the sensor models. A related issue is the ability of a synthetic environment to provide decoys and false targets that represent a realistic battlefield. The CSE will provide decoys and false targets which will stress the sensor models so as to give a more realistic representation of the sensor's capabilities.

Most military models and simulations, with the exception of limited performance analysis studies, do not account for process (from the perspective of C2 decision functions) and processing (from the perspective of C2 systems) latencies and workloads representations. Many current simulations don't capture battlefield process latencies and the human workload representations. JOSEF captures and models these latencies by modeling delays at the appropriate point in C2 systems and decision processes, and workloads through human and organizational behavior representation. This capability is used to model operators, decision makers – in particular the Time Sensitive Targeting (TST) Cell.

Other C2 related phenomenology, such as data "gridlock" and errors in message transmissions in combat communications systems need to be represented. Again, most modern simulations don't model this gridlock phenomenon and its causes. Although JOSEF will not model all aspects of this phenomenology in the Spring '02 instance, it will incorporate the necessary interfaces and capabilities to model this phenomenology.

1.2.2 System Use Case

The following discussion explains the challenges faced by real-world systems and operators which must be addressed by JOSEF. Consider an air-to-ground scenario consisting of five targets.

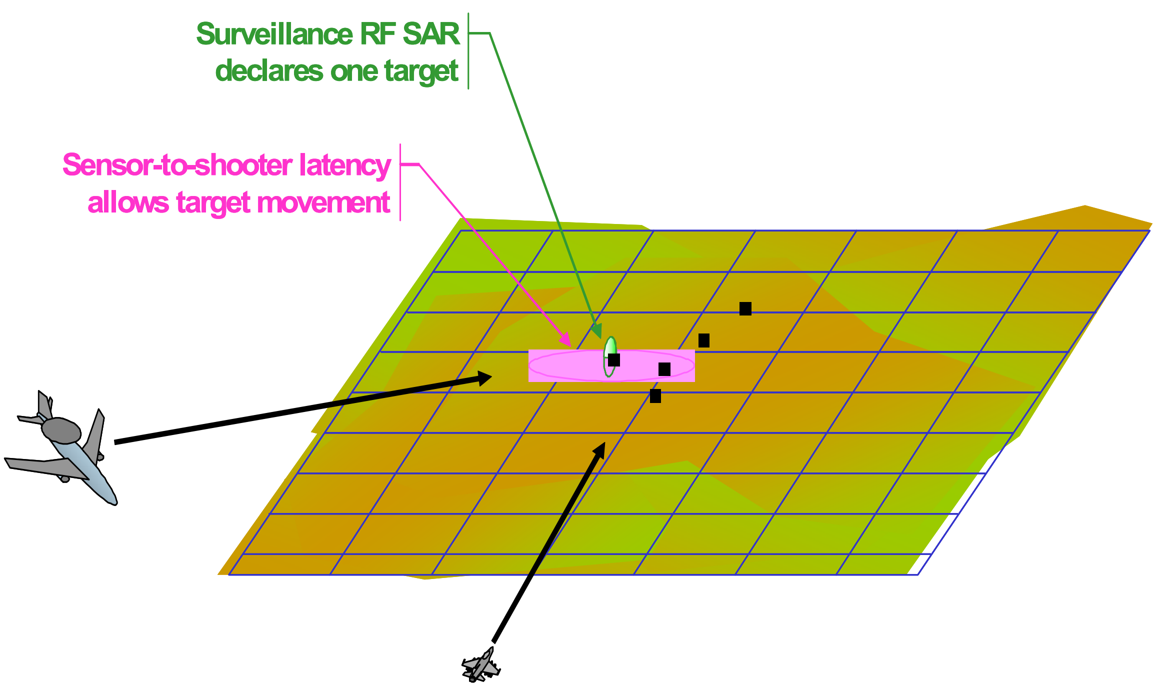


Figure 1-1. Surveillance RF SAR detects one target

Surveillance radio-frequency (RF) synthetic aperture radar (SAR), an imaging radar, identifies one target, as is shown in Figure 1-1, with the green ellipsoid representing SAR measurement error. Command and control (C2) processing latency allows target movement, thus the pink ellipsoid showing growing uncertainty of target location, before any other assets are allocated for prosecution.

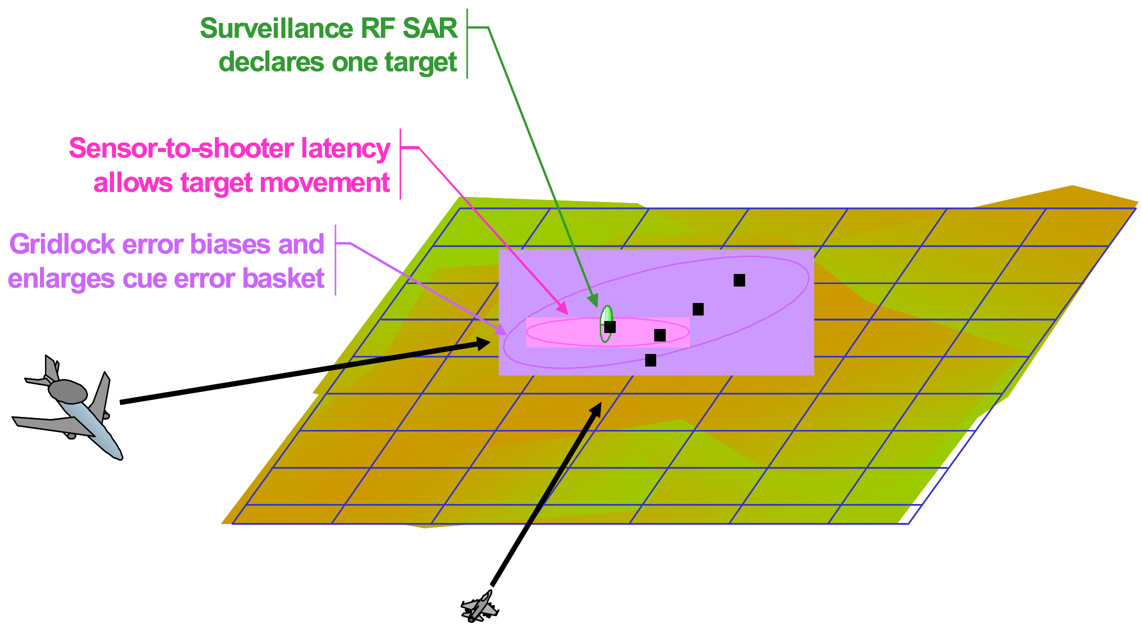


Figure 1-2. Gridlock error and process effects.

Dynamic and credible representations of sensor performance, processing latencies, workloads, and gridlock in current models and simulations are non-existent in nearly all of today's M&S; the very few exceptions to this rule could not be easily integrated with a common synthetic environment (e.g., OPNET). Figure 1-2 further illustrates the impact of real-world effects that allow the cue error basket to grow, the purple ellipsoid representing error due to resolution of differing platform coordinate systems. From a real-world perspective, errors are typically incurred not only within sensor measurements, C2 processes and their physical latencies, but also in message transmission (i.e. corruption of data contained in messages).

Following this example further down an operational timeline, the next step would be target validation prior to weapons pairing and strike aircraft assignment. Figure 1-3 demonstrates the use of ground motion target indicators (GMTI) and SAR to identify four targets (two moving) within the cue box. As a further refinement of target location and type, electro-optic/infrared (EO/IR) sensors are employed. Once again, respective inherent sensor measurement errors are represented by ellipsoids. The current state of M&S practice is challenged by replicating the temporal multi-spectral data correlation problem.

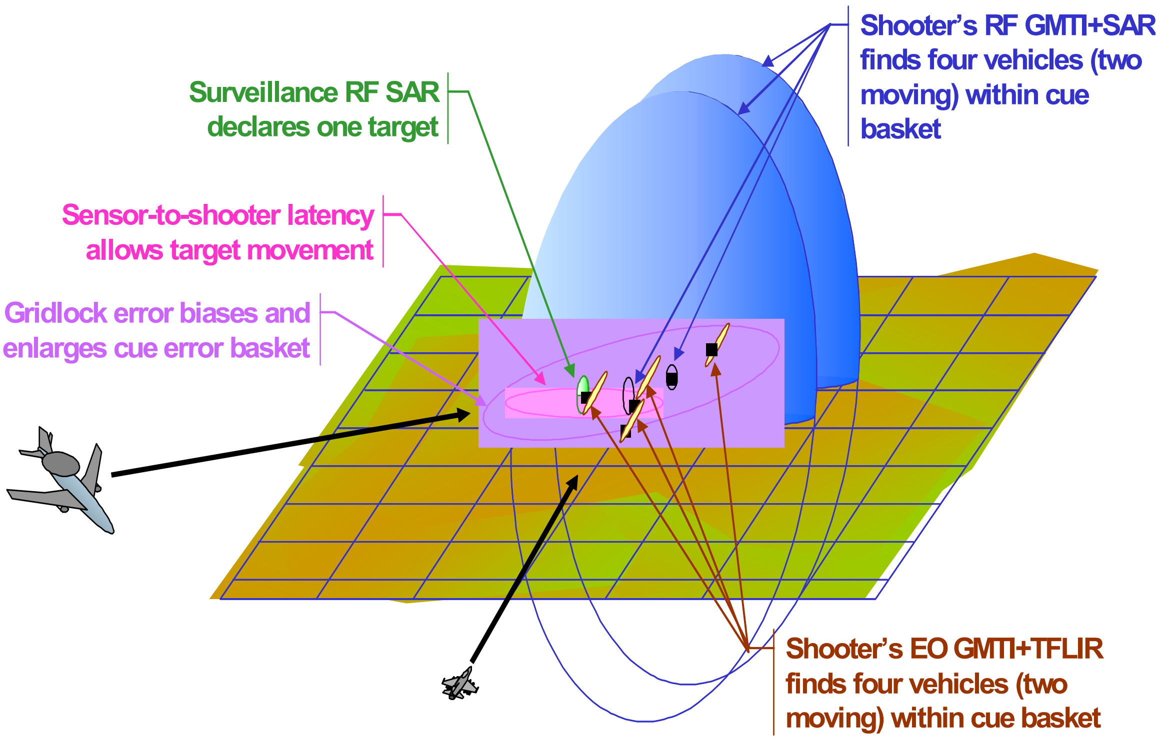


Figure 1-3. Multi-spectral correlation problem.

This page intentionally left blank

2. RELATED DOCUMENTS

The following documents, of the issue in effect at the date of publication of this plan, form the basis for the requirements and procedures described herein.

2.1 GOVERNMENT DOCUMENTS

(a) IEEE/EIA12207 - Software Lifecycle Processes, 1996.
(b) MIL-STD-498, Software Development and Documentation, 5 December 1994
(c) MIL-STD-973 - Configuration Management, 17 April 1992.
(d) DOD-STD-480A – Configuration Control, Engineering Changes, 29 Dec 1978.
(e) Department of Defense High Level Architecture Interface Specification, Version 1.3, DMSO, April 1998, available at http://hla.dmso.mil.

2.2 NON-GOVERNMENT DOCUMENTS

(a) Software Engineering Project Management, R. Thayer, IEEE, Computer Society Press, 1987.
(b) Software Engineering Risk Management, D. Karolak, IEEE, Computer Society Press, 1996.
(c) Software Engineering Institute, Capability Maturity Model, Version 1.1, Carnegie Mellon University, 1993.

This page intentionally left blank

3. SYSTEM-WIDE DESIGN DECISIONS

One of the key design decisions was a programmatic one – limit investment in time and resources by minimizing new modeling and simulation (M&S) development. This forced the ability to demonstrate reuse and integration of *best-of-breed* Air Force and DoD simulations and environments. Additionally, the desire to build on the Joint Combat Identification Evaluation Team (JCIET) 2002 event—Replicate scenario(s)/vignettes from the event; use event data for validating the Experiment Federation; and use the experience to identify the challenges migrating to future (To-Be) operational military architectures.

Another constraint was the desire to design for extension – thus the need to consider near-term support for Global Strike Task Force (GSTF) Command and Control (C2) Constellation studies and analysis and future SBA requirements. As part of this goal, the constraint to identify JSB (SBA) requirements drove the design to demonstrate JSB (SBA) concepts and elicit user and system requirements.

In terms of broad technical system-wide decisions is the issue of fully correlated representations. This concept applies in several different dimensions. For instance, in the multispectral sensor example previously discussed, it is not sufficient to model the RF and EO/IR sensors and the environment to stimulate them. The environment models and databases must be correlated across all the spectrum elements of interest in all phenomenology models.

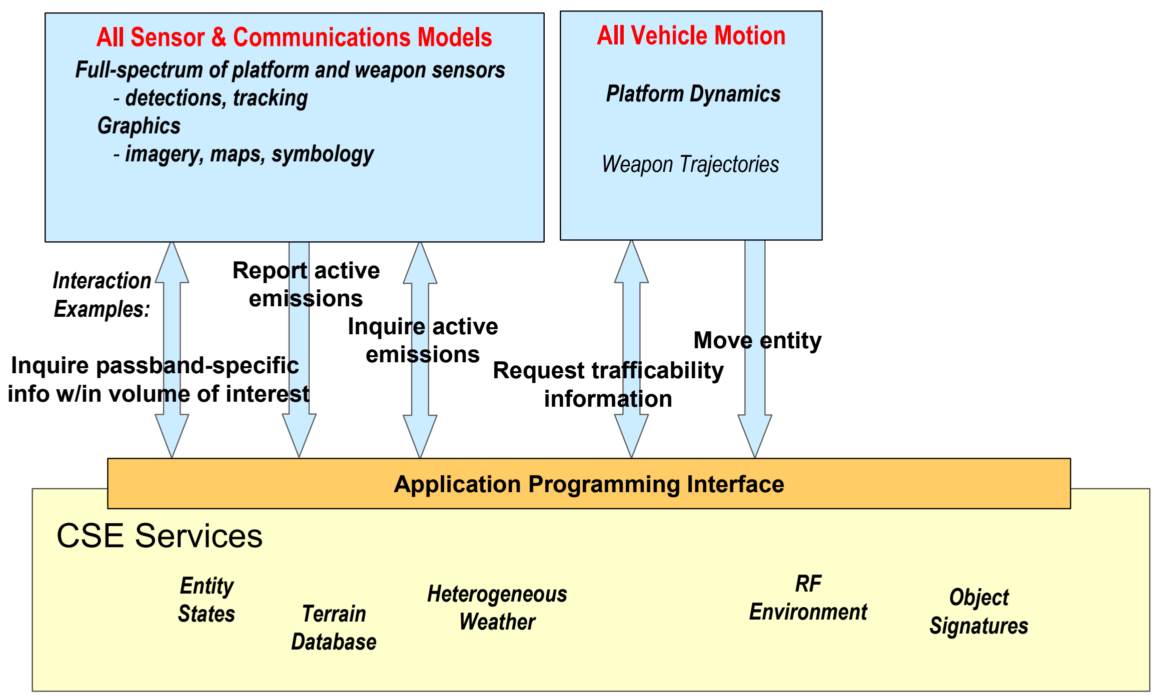


Figure 3-1. CSE relationship to phenomenological models.

In addition, when different levels of fidelity and resolution are mixed, care must be taken so as to conform to the data integrity requirements from model to model. For instance, many tracker algorithms and associated higher fidelity sensor models require platform dynamics and weapons trajectories to possess realistic acceleration and velocity vectors in order to function properly. This means that low-fidelity motion models with discontinuous accelerations and velocities (which are still quite common), must be upgraded.

The relationship of battlespace phenomenology models to those capabilities in the common SE envisioned for the JSB (SBA) is illustrated in Figure 3-1. This factorization of functionality ensures the level playing field capability. By encapsulating the environment and associated models and simulations (such as object signatures) that are required to be the same for all vehicle, weapons, sensors, and communications hardware of interest and providing a standardized access mechanism, an important architectural evolution step is taken. JSB (SBA) will establish a set of open standards and best practices for implementing a common synthetic environment, validated against live test results.

3.1 SE APPLICATION: BASIC SENSOR ARCHITECTURE

The shortfall exists where each model and simulation in a test, analysis, or exercise has its own representation of the world; this leads to a lack of correlation in key computations, such as multi-sensor fusion, that are essential to represent the operations in the battlespace of the future.

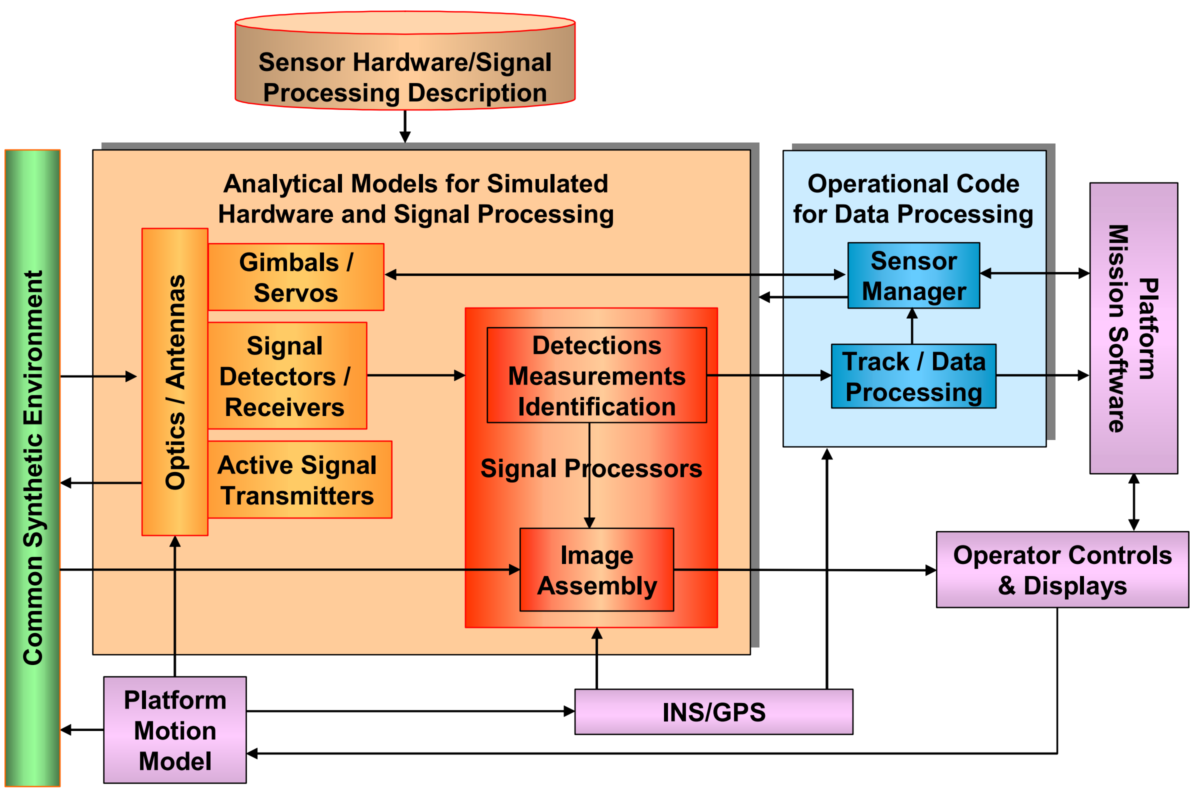


Figure 3-2. Basic Sensor Architecture.

As is shown in the Figure 3-2 above, achievement of a factorization that would separate the environment representation and signal propagation, the sensor representation, and the platform to which the sensor is attached was the goal. In addition, the ability of the sensor model to generate imagery or otherwise stimulate a tactical display needs to be separated from the signal receiver/processor. Greater flexibility is achieved due to the ability for specialists in each domain to model what they know best.

3.2 HARDWARE AND SOFTWARE

The collection of models and simulations that comprise JOSEF are all HLA compliant using various HLA integration approaches. The main federates are illustrated below in Figure 3-3. the basic functionality of each will be described in subsequent sections, and provide a detailed design description of the CSE federate in Appendix A.

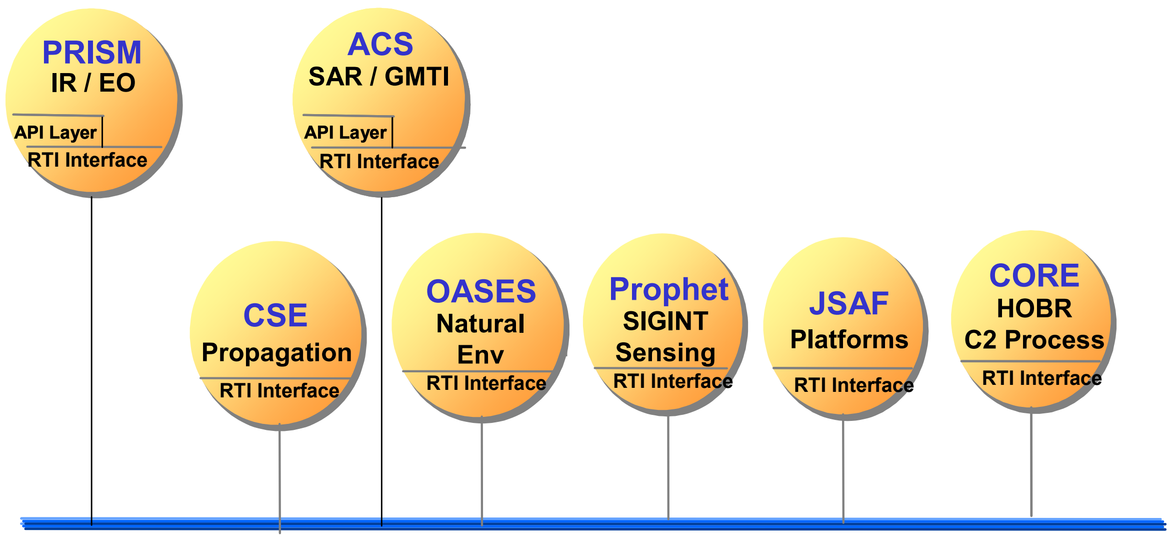


Figure 3-3. Primary JOSEF components.

The specification for the Basic PC is:

1.5+GHz CPU
512 MB RAM
40GB HDD
Min 32MB APG graphics board
Run Linux or Windows

The specification for the Max PC is:

2.0+GHz CPU
1 GB RAM
40GB HDD
NVIDIA GeForce 4 64MB APG graphics board
Run Linux or Windows

Software Component	Computer Type
ACS - GMTI/SAR	Basic PC

ACS - APG-70	Basic PC
PRISM – LANTIRN	Basic PC
PRISM - UAV Camera	Basic PC
Prophet	Basic PC
Prophet - Data Collector	Max PC minus graphics
OASES	Basic PC
CSE Service	Max PC minus graphics
JSAF	Max PC minus graphics
ModStealth	Max PC
JSAF / CORE UAV Operators	Basic PC
JSAF / CORE F-15 Operators	Basic PC
CORE TST Cell	Basic PC
CORE JSTARS Operators	Basic PC
Federate Control	Basic PC

This page intentionally left blank

4. SYSTEM ARCHITECTURAL DESIGN

The JSB Experiment's purpose is to illustrate the concept of accessible, authoritative, configuration-controlled capabilities for populating a synthetic battlespace to support specific needs of an acquisition assessment event. The Common SE, as shown below in Figure 4-1 (referred to as the CSE hereafter), is the organizing principle within which all carefully selected exercise participants must interact in order to lend credibility to the overall execution.

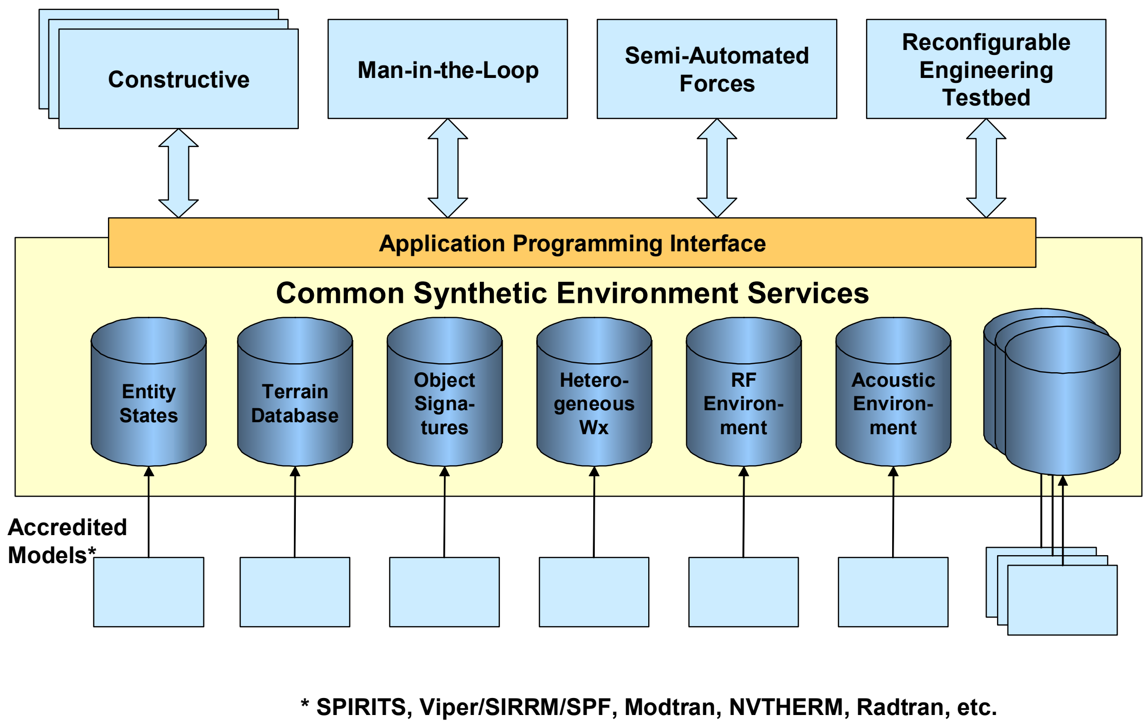


Figure 4-1. JSB Experiment infrastructure.

The JSB (SBA) initiative is embarking on a sequence of proof-of-concept experiments that are creating instantiations of Synthetic Environments (SE's) that improve the representation of the real world. The current JSB Experiment is providing an initial population of the above architecture, with emphasis on the CSE, Sensor, C2 process modeling, and human and organizational behavior modeling.

Future capabilities, such as the JSB Experiment and its successors, must provide increasingly complete battlespace representations to be available as a de-conflicted, level playing field to support acquisition decisions. **A critical objective is ensuring the correlation (modeled at the same level and using common databases) and resolution of Synthetic Environments to achieve consistent and accurate environmental representations between and among the models and simulations used within a test, analysis, or exercise.** Figure 4-2 details many of the capabilities required to represent the complexity of the environment in which aerospace forces must operate.

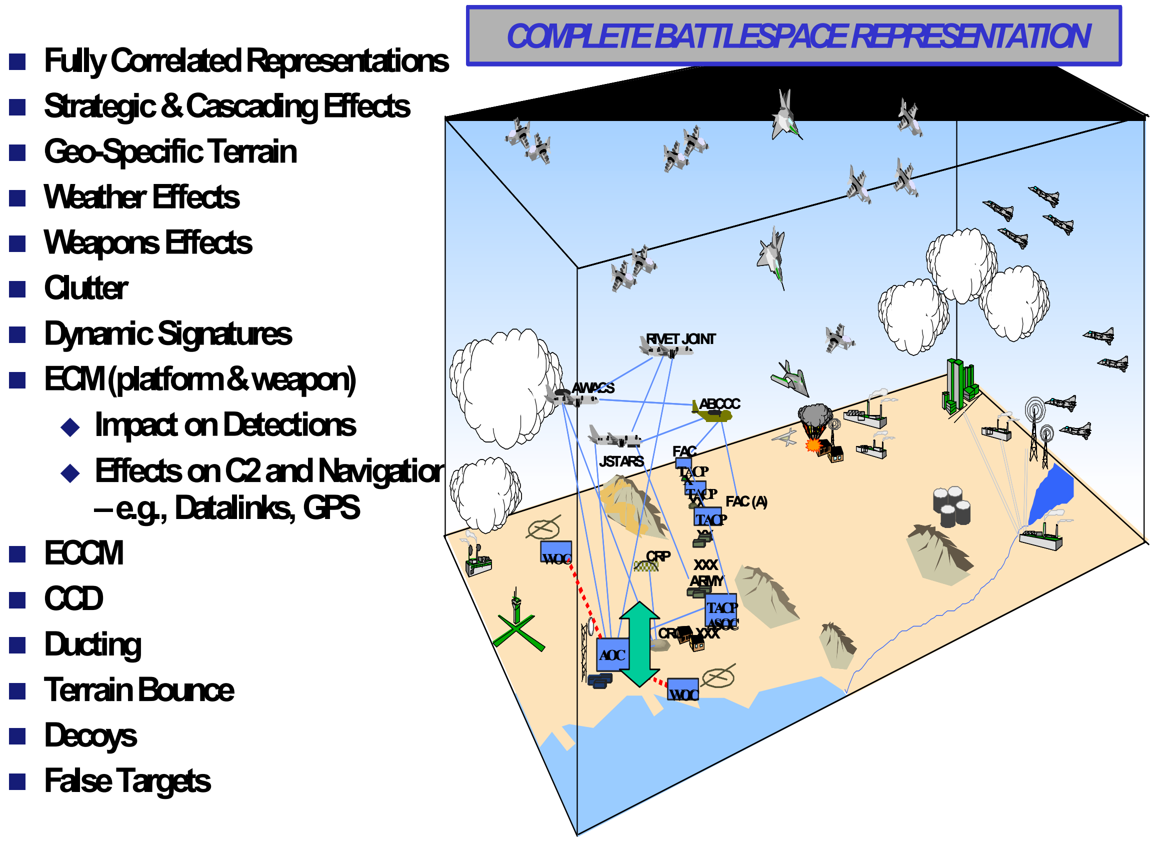


Figure 4-2. Characteristics of a complete battlespace representation.

4.0 SCENARIO AND MODELING APPROACH OVERVIEW

The JSB Experiment will be analyzed, verified, and validated by modeling a Time Sensitive Targeting vignette from the Joint Combat ID Evaluation Team (JCIET)/Joint C2ISR (JC2ISR) 2002 live event. This exercise will be played out as illustrated in Figure 6 below. The primary exercise regions are the Camp Shelby/ De Soto, Mississippi military operating area (MOA) and the Camden Ridge/Pine Hill, Alabama exercise area, as indicated in Figure 4-3 below. The JSB Experiment vignette will focus on Time Sensitive Targeting (TST) scenarios played out in the exercise, and in particular, the TST area is the Camden Ridge/Pine Hill exercise area.

Representing this area synthetically is a challenge for several reasons. First, creating a high-resolution multi-spectral database to simultaneously support high-fidelity sensor modeling in the EO/IR/RF spectrum using DMSO's Synthetic Environment Data Representation and Interchange Specification (SEDRIS) has never been done. Second, the variety of terrain, vegetation, and cultural features found in the Camden Ridge/Pine Hills area is challenging to represent and model. Finally, the climate in this area varies widely during the time of the year the event transpires, requiring a reasonable level of fidelity and detail in modeling the atmospherics correctly and consistently across all of the spectrum elements.

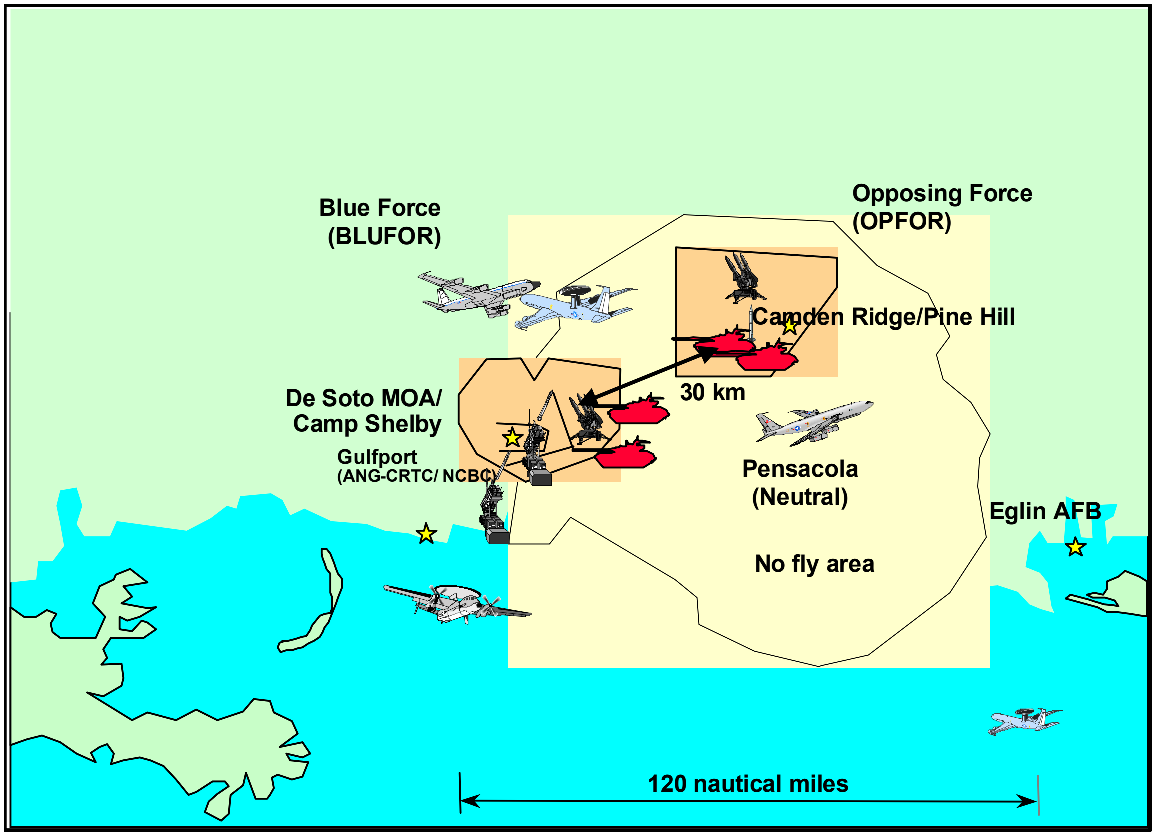


Figure 4-3. JCIET/JC2ISR JT&E 2002 exercise area.

The JSB Experiment scenario is described below.

1. Rivet Joint (RJ) detects an ELINT hit.
    1a. RJ computes target ellipse location
    1b. RJ contacts Time Sensitive Targeting Cell (TSTC) and passes target type and coordinates

2. TSTC directs Joint Stars (JS) to investigate
    2a. TSTC begins target validation process
    2b. TSTC identifies possible strikers for target

3. JS detects movers at location passed by RJ
    3a. JS detects movers along line of communication (LOC)
    3b. Based on previous intelligence and where detected confirms RJ's initial target type identification, if capable of doing so

4. TSTC tasks UAV to monitor target
    4a. UAV locates target with EO sensors
    4b. UAV passes its assessment of target type to TSTC
    4c. TSTC completes its target validation process with the data passed by the UAV

5. TSTC initiates J3.5 message on the JTIDS net on the target in question
    5a. TSTC contacts the Air Operations Center (AOC) which contains the JTIDS ground station with J3.5 info
    5b. TSTC confirms the appearance of the new ground target on the JTIDS net

6. JS loses GMTI track on the target in question
    6a. Target stops moving
    6b. Target stop could indicate it is preparing to "shoot"

7. TSTC re-tasks UAV to targets last known position
    7a. UAV finds target
    7b. UAV passes to TSTC target location and its activity
    7c. UAV continues to monitor target

8. TSTC issues J12.0 message (execution message) to F-15E strikers
    8a. If JTIDS is unavailable TSTC passes message to strikers via voice
    8b. TSTC cell passes on to strikers any other pertinent info about targets (9 Line type info)

9. F-15Es prosecute target attack
    9a. F-15E executes a SAR map of the target area
    9b. F-15E finds target on SAR map and refines target coordinates
    9c. F-15E cues the LANTIRN targeting pod to updated coordinates
    9d. F-15E locates and IDs target in LANTIRN

The complications inherent in developing an accurate prediction of this timeline involve solving, at some level, all of the major problems listed in Section 1.2.1. The development of dynamic and credible representations of sensors, as we have mentioned, is complicated by the diversity of the environment. Most current synthetic environments are not dynamic and are not correlated through the entire electromagnetic spectrum. JOSEF's CSE will provide a common, correlated and integrated environmental representation to three different legacy models covering the EO/IR/RF part of the spectrum.

Most synthetic environments do not have the capability of representing dynamic signatures for the entities contained within them. JOSEF's CSE will provide dynamic signatures for all of the targets of interest. These dynamic signatures will be influenced by ephemeral conditions.

Another complication provided by the Camden Ridge/ Pine Hill exercise area is the fact that it is not a restricted military-only area. As such, there will be many non-military entities that are detected by the sensors involved in JCIET/JC2ISR 2002. JOSEF will provide a varied clutter environment for accurate and dynamic stressing of the sensor models. As was noted in Section 1.2.1, most synthetic environments do not provide decoys and false targets that represent a realistic battlefield. JOSEF's CSE will provide decoys and false targets which will stress the sensor models so as to give a consistent and more realistic representation of the sensor's capabilities.

Due to the nature of the JCIET event – investigating new C2 tactics, techniques, and procedures – the C2 processes, human, and organizational behaviors usually implemented in existing models are not valid. A related issue is the representation of communications and C2 processing latencies and operator workloads. JOSEF captures and models these latencies and workloads through its human behavior model representation and its Time Sensitive Targeting (TST) representation and incorporation of human behavior.

4.0.1 Operational Architecture

The operational architecture being modeled is indicated in Figure 4-4. The F-15E and JSTARS will communicate via voice and Link 16 between each other the TST Cell. The TST Cell is part of the AOC. The Rivet Joint and UAV platforms communicate via voice communications only.

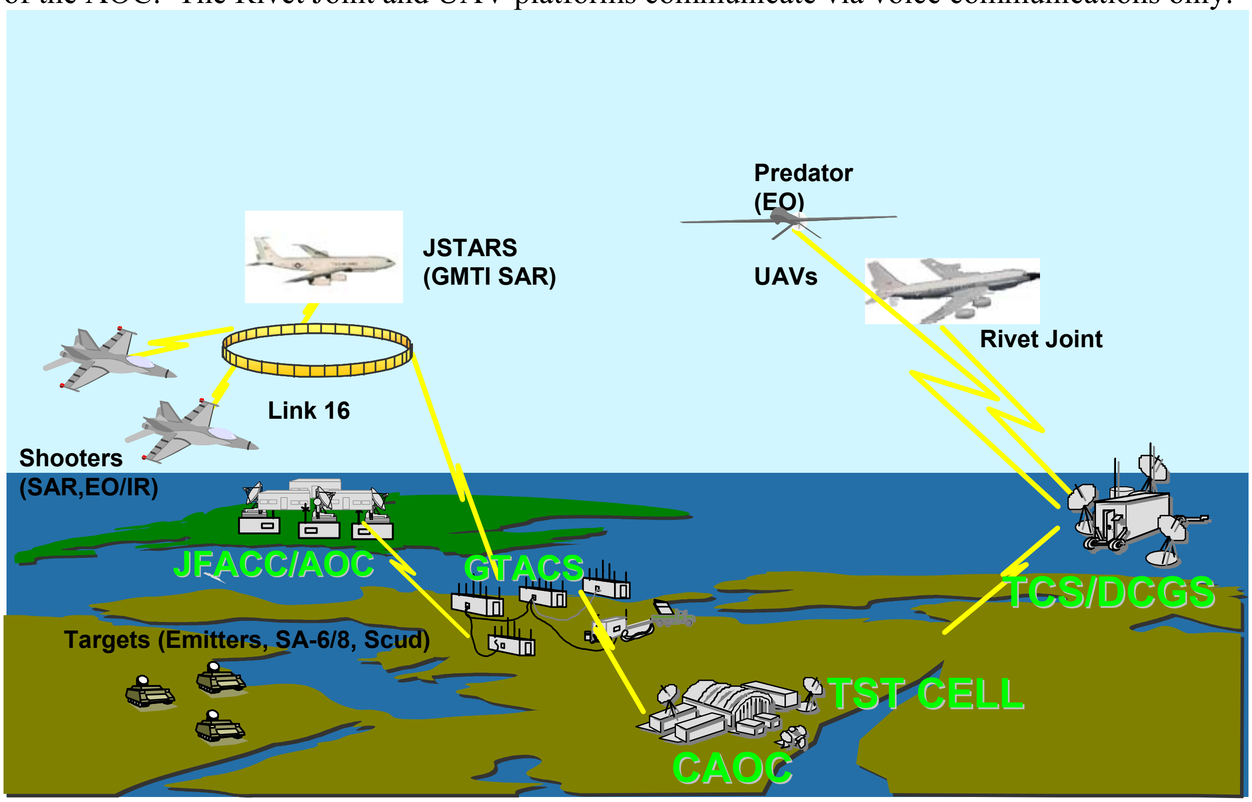


Figure 4-4. JOSEF Operational Architecture view.

The targets communicate with hand-held RF radios using authentic frequencies, and the targeting radars used with the SA-6 and SA-8 air defense systems use authentic frequencies also. This permits the maximum realism in the detection phase of the overall combat ID problem, and allows the scenarios modeled in JOSEF to be constructed (and hence modeled) more accurately.

4.0.2 Functional Decomposition

In order to understand what is being modeled, we refer to Figure 4-5. Each platform (F-15E, JSTARS, UAV, and Rivet Joint) has at least one sensor being modeled in such a way that it utilizes the CSE, and these interactions are indicated using solid colored lines in the diagram. Thus, all the targets (SA-6, SA-8, and Scud) have to have EO/IR/RF signatures.

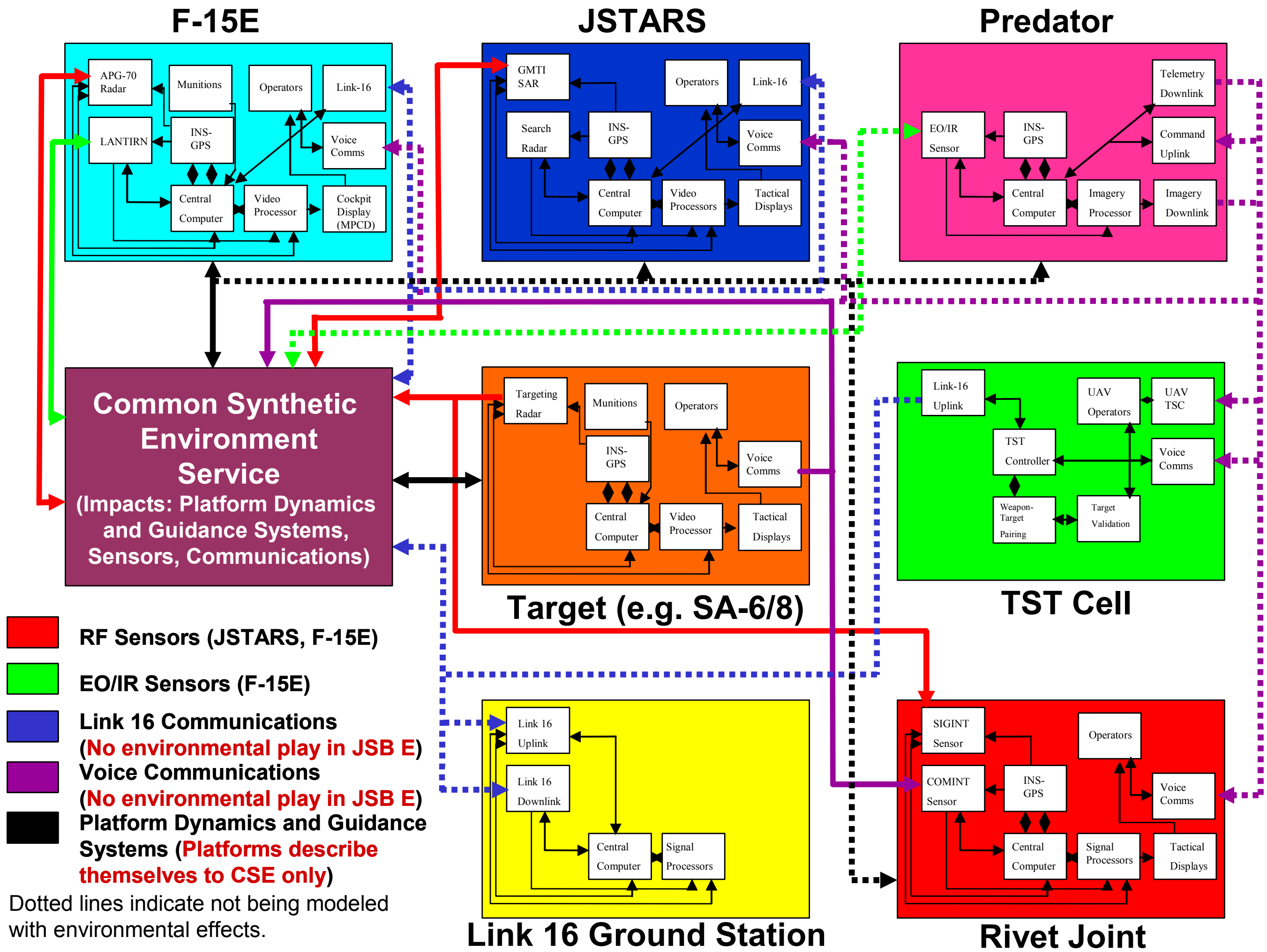


Figure 4-5. JSB Experiment functional decomposition.

In addition, it should be noted that all the platforms require some type of representation of operators. The design decision was made to model C2 processes, humans, and organizational behavior (e.g., the TST Cell), using a toolset that while permitting separation of behavior from sensors and vehicle dynamics representations, it can be embedded within other models and simulations or execute as a single computational entity.

The other point of interest is to note that the voice and Link 16 communications are not being modeled at a level of fidelity that takes into account moderation by the environment. Modeling this phenomenology is planned in future experiments, and the approach taken in the CSE implementation will permit this capability to be added modularly.

4.1 SYSTEM COMPONENTS

4.1.1 System Component Overview

The fundamental elements of JOSEF fall into the following categories:

- Distributed Simulation Architecture and Infrastructure
- Environmental Representations
- Vehicle/Platforms Representations
- Target Representations
- Human and Organizational Behavior Representations
- C2 modeling and simulation
- Sensors Representations
- COP Visualization
- Common Services Servers/Models
- Federation Control

4.1.2 Distributed Simulation Architecture – Infrastructure

There are two different components to the distributed simulation architecture/infrastructure. The HLA Runtime Infrastructure (RTI) provides distributed simulation services to a federate in a way that is analogous to how a distributed operating system provides services to applications, and the Synchronous Parallel Environment for Emulation and Discrete Event Simulation (SPEEDES) is used to provide a HLA integration middleware for some of the JOSEF federates.

Both the RTI and SPEEDES contain interfaces that are arranged into the service groups given below:

- Federation Management
- Declaration Management
- Object Management
- Time Management

The six service groups describe the interface between federates (or simulation components with SPEEDES) and the RTI, and the software services provided by the RTI for use by HLA federates (called the ambassador).

4.1.3 Environmental Representations

The Environment serves the dynamic 3D grid atmospheric data. It provides the effects on the EO/IR and RF spectra, which include weather changes. The Experiment Federation Environment will be provided by DMSO's Ocean, Atmosphere, and Space Environment Services (OASES) with extensions for RF and IR. The terrain, vegetation, and cultural features are stored in a Compact Terrain Database format file, and provide a polygonal representation with attributes and abstract features. JOSEF terrain will also contain RF and EO/IR extensions.

The CSE federate will model the environment elements that are used to compose the synthetic battlespace. These elements include the synthetic environment data, that includes the natural, cultural, and material-codes, and terrain data; meteorological, weather, and atmospheric data; and astronomic and stellar data; and the data services necessary to reflect the environment within which the systems and system components interoperate, at the appropriate level of detail for the simulation/federation. The data services include effects models that describe the interactions between the environmental elements and the specific system simulations. Effects models include aerodynamic drag, radar and optical backscatter, radio propagation, ballistics, etc. The CSE design is provided in detail in Appendix A.

4.1.4 Vehicle/Platforms Representations

The Platforms are the entities that move through time and space and provide entity position and motion information to the federation. The Experiment Federation Platforms are provided by the Joint Semi-Automated Forces (JSAF) application. JSAF will provide platforms for the Sensors and the Targets. The initial JOSEF platforms include JSTARS, F-15-E, UAV, and Rivet Joint air vehicles and the SA 6/8 and SCUD ground targets.

JSAF is a platform-level, real-time Computer Generated Forces (CGF) simulation system that is typically used to provide an operational context in support human-in-the-loop (HITL) experimentation and training, and through modifications made in its use of the HLA time management API calls for the JSB Experiment, JSAF will be able to support engineering and design. As shown in the JSAF Conceptual Model below, JSAF provides representations of the natural environment (e.g., clouds, wind, haze), environmental effects on military systems (e.g., obscuration effects of smoke, chaff, etc. on sensors), weapons effects impacts on the environment (e.g., runway cratering due to bombing), weapons systems dynamics, and tactical behaviors.

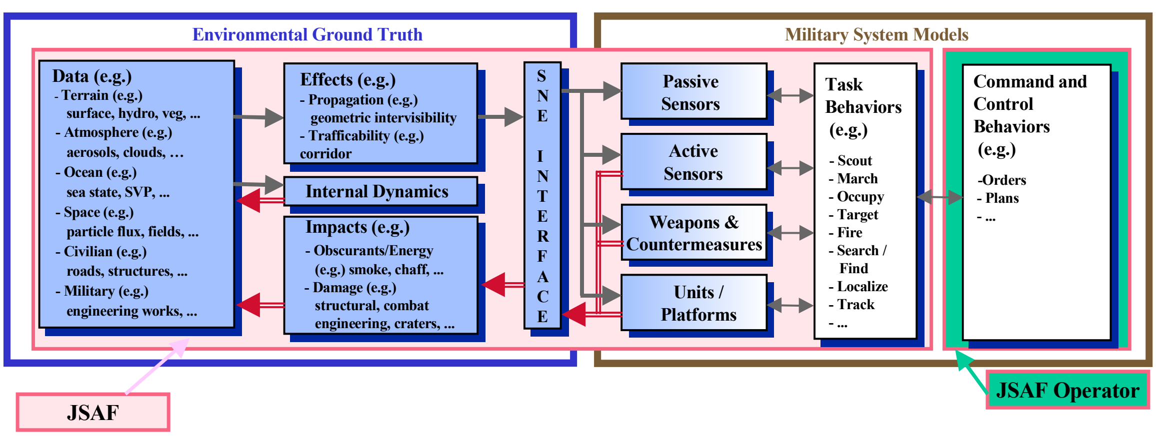


Figure 4-6. JSAF Conceptual Model.

The JSAF Operator provides supervisory control of the simulation, including functions such as placing units on the battlefield, creating missions for those units to execute, and monitoring or altering the execution of those missions during an exercise. The JSAF Graphical User Interface (GUI) provides mechanisms for the operator to influence the battle. The operator can create, modify, and delete platforms as well as assign tasks (e.g., move, shoot, withdraw) to them. Overall battlefield events can be closely supervised on the JSAF Plan View Display (PVD).

JSAF was originally developed under the multi-year DARPA Synthetic Theater of War (STOW) Advanced Concept Technology Demonstration program. It is a merger of the ArmySAF, NavySAF, AirSF, and Marine Corps SAF Computer Generated Forces systems that were developed for DARPA STOW. It includes models, behaviors and synthetic natural environment from beyond the "Army" domain that is the focus of ModSAF. It also captures the software developed under the Joint Countermine Operational Simulation (JCOS) and the USACOM J9 exercise J9901. Considerable "core" commonality is maintained between JSAF and ModSAF with respect to key infrastructure libraries (**`libpo`**, **`libtask`**, **`libsched`**, **`libenv`**, **`libctdb`**, etc.).

Ground Tasks in JSAF

Ground tasks in JSAF include the followings:

- Move
- Collide
- Terrain
- Stingray React
- Mount
- Indirect Fire Mission
- Supply
- Receive
- Backtrack
- MLRS

The **Move** task moves an entity along a route and around obstacles. An entity can selectively avoid rivers, lakes, buildings, trees, treelines, and other vehicles. This task determines when the entity should get on a route. Ground vehicles automatically avoid obstacles they encounter in their paths. The direction of travel and speed of other vehicles is calculated to ensure collisions do not occur.

The **Collide** task allows vehicles to recover from collisions and near-collisions. It directs colliding entities to wait a random amount of time and then back away. This task is also invoked when a vehicle cannot move around an obstacle

The **Terrain** task contains a local terrain map that vehicles use during route planning. This map also displays local obstacles that entities should avoid.

The **Stingray React** task simulates permanent damage received from a Stingray-equipped simulated vehicle on the network. If a vehicle receives permanent damage due to Stingray fire, it searches for a hidden position in the surrounding terrain. An entity's parametric data for this task determines whether to check for a Stingray hit and, if the vehicle is damaged, the speed at which it should move to a hidden position (in meters/second). Currently the following vehicles check for a Stingray hit: M1, M2, T72, BMP1, and BMP2.

The **Mount** task mounts DI onto its corresponding Infantry Fighting Vehicle (IFV). When the DI is dismounted, the Mount task waits for the appropriate IFV to get within a certain range. After a set time, the DI disappear. The DI are then considered mounted and the Mount task is in the mounted state. At this state it waits for a dismount request. After a set time, the Mount task causes the DI to reappear in a location behind the IFV. The task returns to the dismounted state.

The **Indirect Fire** task processes artillery radio fire requests for artillery vehicles. The gun fires the artillery. Vehicles that are running this task with turret traverse limits must adjust their hull orientation to the target line if the turret azimuth is out of range.

The **Supply** task implements resupply of a vehicle. The task processes the necessary issue and reception of supply protocol packets.

The **Receive** task requests supplies from a resupply vehicle if the:

Resupply vehicle is next to the vehicle.
Resupply vehicle is halted.
Vehicle is halted.
Vehicle needs supplies.

The **Backtrack** task implements a vehicle-level task that causes a vehicle to backtrack for a given distance. This is done by retrieving the history list (a list of points where the vehicle has previously been), reversing it, and setting the vehicle to move backward along the route described in the reversed list.

The Multiple Launch Rocket (**MLRS**) task implements the firing and mission acknowledgement characteristics of Multiple Launch Rocket vehicles. It loads the ammunition included on the vehicle, waits until the vehicle is in firing position, and then fires. The vehicles running this task are considered "one shot"; their munitions are expended during the mission.

Fixed Wing Aircraft (FWA) Tasks in JSAF

CAS (Close Air Support) - Allow ground attacks to be set up using radios.

Interdiction - Primarily designed for bombing bridges
.
Ground Attack - Attack ground units in an area.

Fly Route - Fly along a route.

Sweep - Fly along and attack other aircraft.

CAP (Combat Air Patrol) - Fly a racetrack pattern and look for targets to intercept.

RTB (Return to Base) - Fly back to a designated location.

Jam - Basically a CAP for FWA with jamming equipment.

Air-to-Air Attack - Attack enemy air vehicles.

Collision Avoidance - avoid colliding with something else.

Bingo Fuel - Forces a Return To Base when fuel runs low.

Formation Keeping - Allows flights of 2-4 FWA to maintain formation while flying.

Spot Report - Reports enemy positions.

4.1.5 Target Representations

The representation of targets is done with different system components. In general, the vehicle dynamics and basic tactical behavior is modeled using JSAF. The targeting radar and communications radios are also modeled in JSAF. The EO/IR signatures are modeled in SigSim, and the RF signature is generated using Xpatch. This process is described in detail in Appendix A. Background clutter vehicles are also modeled using JSAF, and the signatures modeled similarly.

To summarize:

- SA-6 – TEL + Targeting Radar + Radio using JSAF; EO/IR sig. using SigSim; RF sig. using Xpatch
- SA-8 – TEL + Targeting Radar + Radio using JSAF; EO/IR sig. using SigSim; RF sig. using Xpatch
- Scud – TEL + Targeting Radar + Radio using JSAF; EO/IR sig. using SigSim; RF sig. using Xpatch

4.1.6 Human and Organizational Behavior Representations (HOBR)

There are several human operators and an organization present in the scenario vignette to be modeled in JOSEF. They are:

- UAV Operators
- F-15E Operators
- JSTARS Operators
- Time Sensitive Targeting Cell staff

The diagrams and design of these behaviors can be found in Appendix B.

Current agent-based approaches to knowledge handling, modeling, and simulation rely heavily on classic representation techniques. These techniques, which include rule-based expert systems, genetic algorithms, neural networks, and similar approaches all suffer from one or more weaknesses that make them unsuitable to meet the needs of modeling and simulation. Human and Organizational Behavior Representation (HOBR) in M&S requires speed, efficiency, the capability to store and retrieve deep levels of knowledge, scalability, adaptability, and low maintenance. Most current HOBRs do not represent humans well enough to adequately train cognitive decision-making skills. Table 4-1 identifies some of the shortcomings of current HOBR models.

Table 4-11. State of HOBRs

Brittle	Hard to modify
Slow	Cannot use deep (human) knowledge
Complex	Expensive to maintain
Rigid	Not scalable or adaptable

The demand for greater realism in simulations forces developers to upgrade the behavior of simulated entities. Consequently, the field of intelligent entities is heavily populated with techniques aimed at providing intelligent behavior in simulated entities. While there are difficult issues associated with using knowledge to produce an efficient, intelligent agent, the more fundamental problem is obtaining knowledge in the first place. Indeed, the volume of knowledge required can become a cost limiting factor unless significant advances are made in knowledge acquisition.

Conceptual Graphs (CGs) are based on the work of John Sowa, who in turn based his work on the graphical logic of Charles Peirce. With a foundation in logic representation, CGs also have first-order and predicate logic operations defined allowing for inference and theorem proving procedures.

Visually, a CG mimics the knowledge representation ability of diagrams used in discussions using whiteboards, slides, and napkins. These drawings are often text snippets (typically enclosed in squares or ovals) and lines (possibly with a label) connecting one snippet to another. Experts often use these visual aids to quickly and effectively communicate complicated, technical details during brainstorming sessions. In CGs, text snippets are called Concepts, and line connections are called Conceptual Relations. A Concept may also contain another CG to provide contextual or nested information.

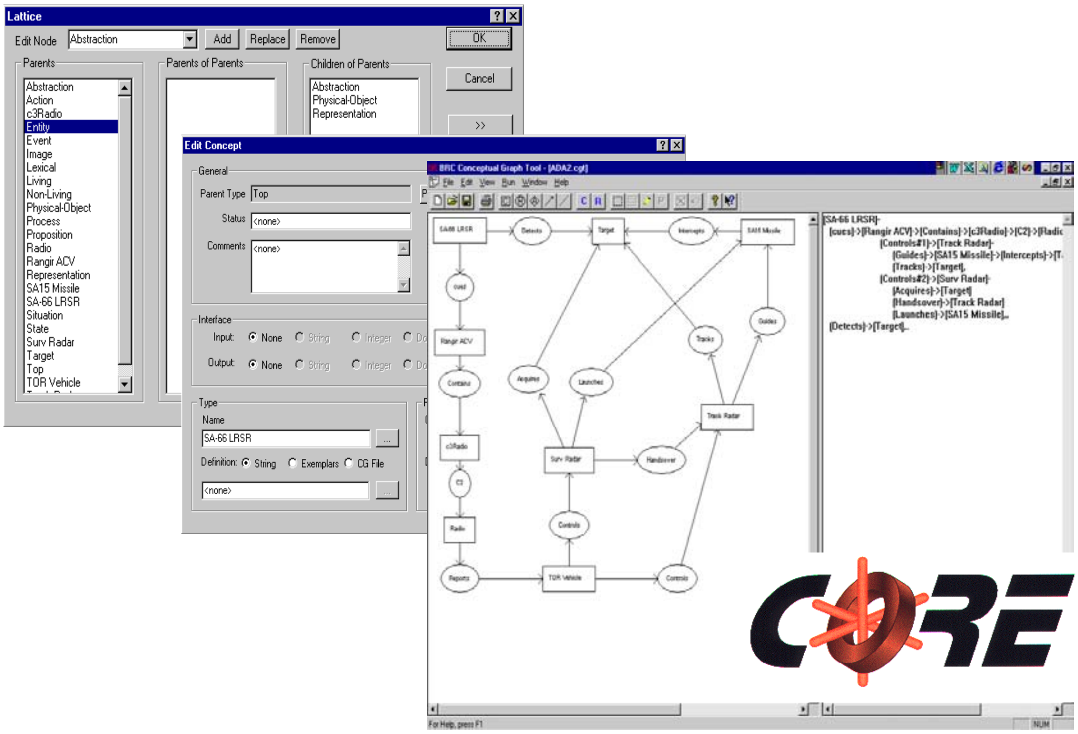


Figure 4-7. CORE User Interface

As the user inserts the knowledge into the system, it is automatically parsed into ANSI Conceptual Graph Interchange Format (CGIF), the emerging national standard for knowledge interchange. This allows knowledge to be easily shared and distributed among users and other applications conforming to the standard.

4.1.7 C2 Modeling and Simulation

The modeling of latencies due to C2 processing and operator workload is done using CORE, as is the modeling of the TST Cell. The TST Cell modeling is illustrated in Appendix B.

4.1.8 Sensor Representations

The following sensors are modeled in JOSEF:

- Low Altitude Navigation and Targeting Infrared for Night (LANTIRN) using PRISM.
- UAV EO Sensor using PRISM.
- APG-70 (SAR mode only) using ACS.
- JSTARS GMTI and SAR using ACS.
- Rivet Joint SIGINT (COMINT and ELINT) using Prophet.

4.1.8.1 EO/IR Modeling

The JOSEF EO/IR Sensor will be provided by PRISM, which will operate as an HLA federate. PRISM will model the F-15E LANTIRN system, and the EO camera on the UAV. The EO/IR sensor takes as input LOS passband transmittance and path radiance, target/background exitances and reflected radiances, position and orientation with respect to the sensor, projected area weights, attenuated target delta-T and background apparent temperature. Details on the data provided by the CSE to PRISM are found in Appendix A. The following diagram illustrates the computational data flows within PRISM.

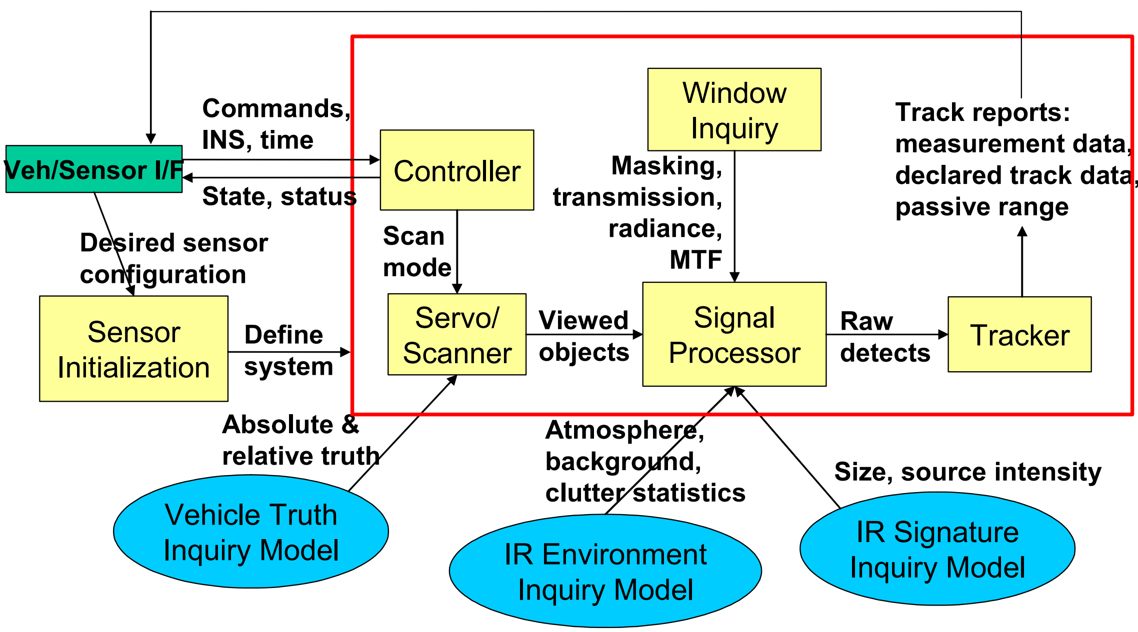


Figure 4-8. EO/IR model data flow in PRISM.

4.1.8.2 GMTI/SAR modeling

The Experiment Federation SAR/GMTI Sensor will be provided by ACS, which will operate as a standalone HLA federate. ACS will be used to model both JSTARS SAR/GMTI functions and the F-15E imaging SAR.

The most important inputs from the CSE to ACS are:

- Target ID's within area of interest
- Via Radtran, X & K –band 2-way refracted path attenuation (dB), Projection of target velocity vector on path direction ("r-dot")
- Apparent sensor view direction vector
- Target RF cross section from X-patch, modeled, or empirical data as function of wavelength, azimuth, and polarization
- Target cross-range and down-range dimensions (projected area), grazing angle

- Average background and object radar returns (γ and mean σ_0) as function of grazing angle and material type
- Precipitation cell RCS, backscatter and range extent

The following diagram illustrates the computational data flows within ACS.

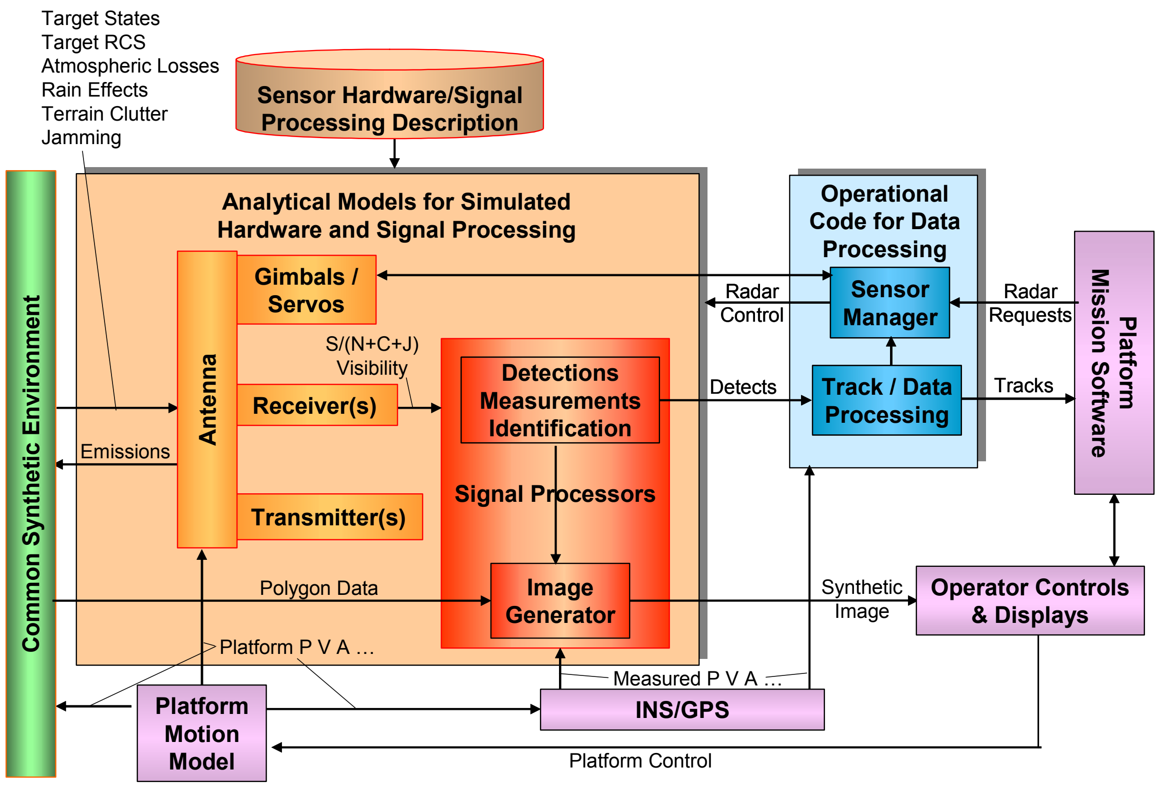


Figure 4-9. ACS data flows.

4.1.8.3 SIGINT Modeling

The SIGINT functions on Rivet Joint are modeled by the Prophet model, which is a standalone HLA federate. The Prophet federate receives signal data from the CSE whenever an emitter is on. This data is periodically updated to account for the platform movement during the on time. The receiver starts at a random point in its scan pattern and cycles through the scan pattern throughout the simulation run. During a particular dwell of the scan patterns, the receiver keeps track of all on emitters in the frequency band of the dwell. It keeps track of all on and off events for the emitter. The dwell frequency band is divided into "N" frequency channels (UPPER_LIMIT_FREQUENCY – LOWER_LIMIT_FREQUENCY) / DELTA_ FREQUENCY) and each frequency channel has the noise (thermal and galactic) computed. Each scan dwell period is divided into "M" time bins (DWELL_TIME/ DELTA_TIME) as shown in Figure 4-10. The ignition noise of the platform is added as a signal to the channels of the dwell. Then, each signal is placed in the channels and sets of intermediate signals are developed. Each of these signals is corrupted in angle and by signal type identification and signal reports are produced.

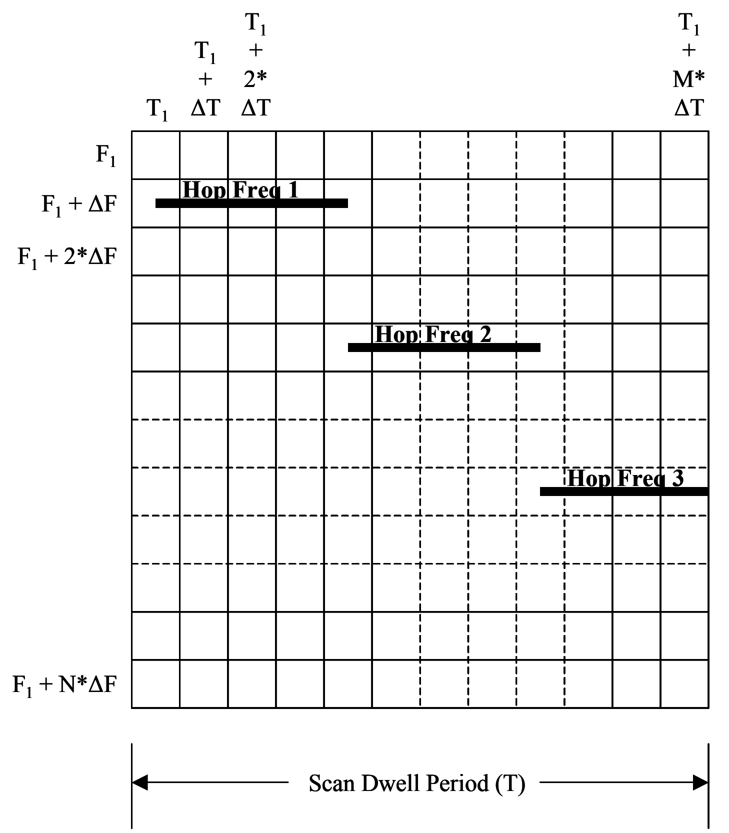
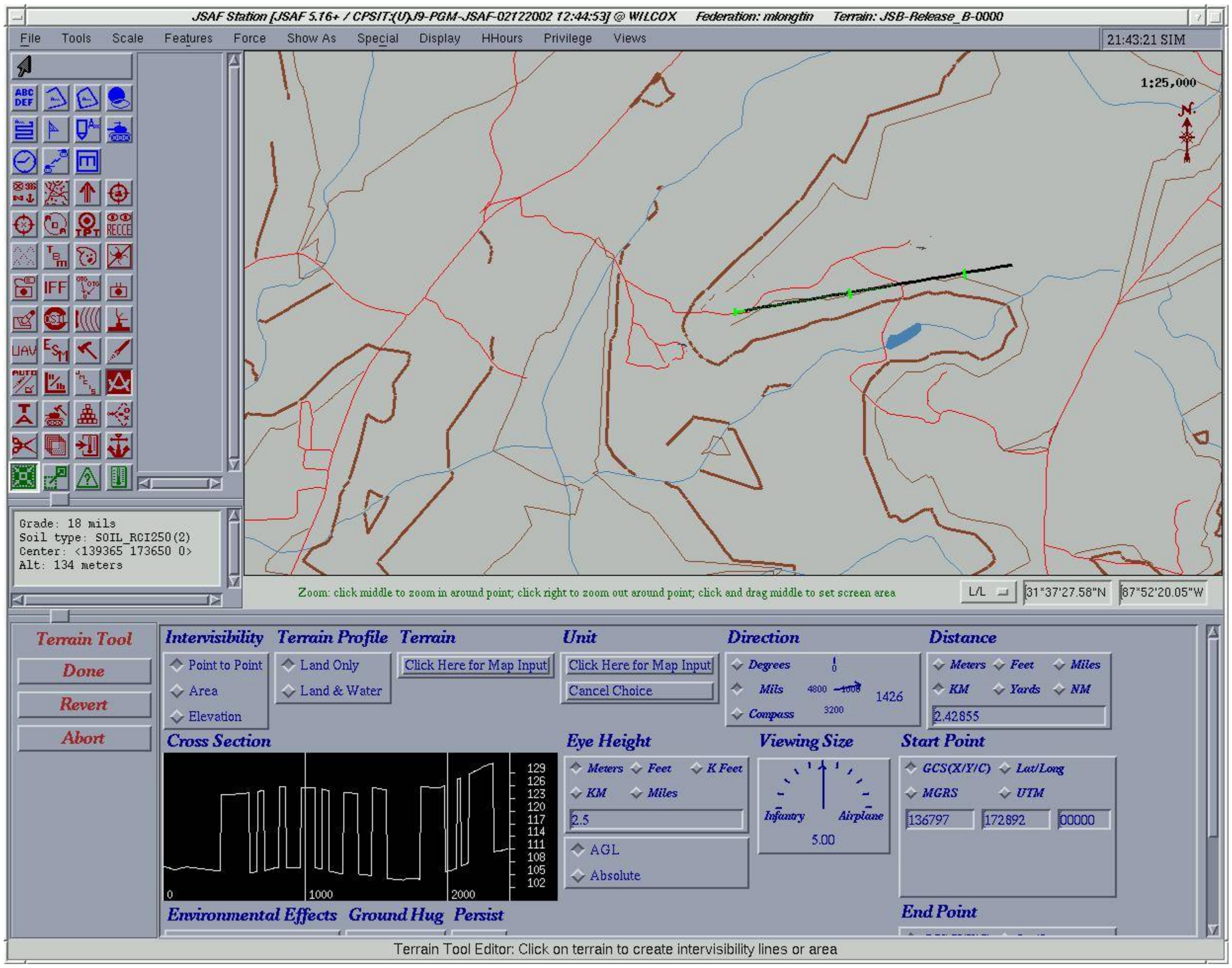


Figure 4-10. Simulated FFT Output Representation

As Figure 4-14 illustrates, the receiver Fast Fourier Transform (FFT) output is simulated by an “N-point” FFT across “M” time divisions (bins) within a single scan dwell period. Figure 4.10 also provides a graphical illustration of the simulated FFT output frequency/time bins and how a non-conventional radio might “hop” across multiple transmit frequencies during a single scan dwell period.

4.1.9 COP Visualization

The COP visualization for JOSEF is provided by the Plan-View Display (PVD) capability in JSAF. The JSAF is shown below.



4.1.10 Common Services /Models

The CSE is a newly created federate that serves the EO/IR/RF sensors using a platform, terrain, environment server federate that computes line-of-sight (LOS) and returns the accumulated attenuation/extinction coefficient following the LOS. It uses a preloaded terrain and feature database. The CSE is described in detail in Appendix A.

4.1.11 Experiment Execution Control

The hlaControl™ tool will be used in JOSEF, and is a powerful agent-based system for managing the planning, execution, and performance analysis of an HLA federation. It has all the functionality of the standard HLA Federation Execution Planners Workbook (FEPW), plus full life- cycle federation management capabilities. This enables cost effective federation management, and enables the ability to determine if performance requirements are satisfied and even identify and correct run time inaccuracies. hlaControl™ has planning tables and visual representation allow the user to create a complete federation plan. Graphical displays of your network topology make life-cycle planning quick and easy.

hlaControl™ starts the federation by launching remote federates and provides full control of the federation execution through the HLA Management Object Model (MOM). These services include create/destroy, join/resign, and publish/subscribe,. Remote agents are used to manage distributed federates from a visual control station. hlaControl™ will auto-discover federates and add them to your federation plan.

hlaControl™'s visual displays of your federation enable performance monitoring and tuning. The tool's ability to collect and playback performance data enables you to compare configurations of the federation execution. Federate, host, and network performance thresholds can be identified, and generate alerts when corrective action is needed.

4.2 CONCEPT OF EXECUTION

Throughout these sections, *Federates* are in *Green italics*, JCIET Systems are in Blue, and Targets are in Red.

4.2.1 Sequence Diagram for Rivet Joint

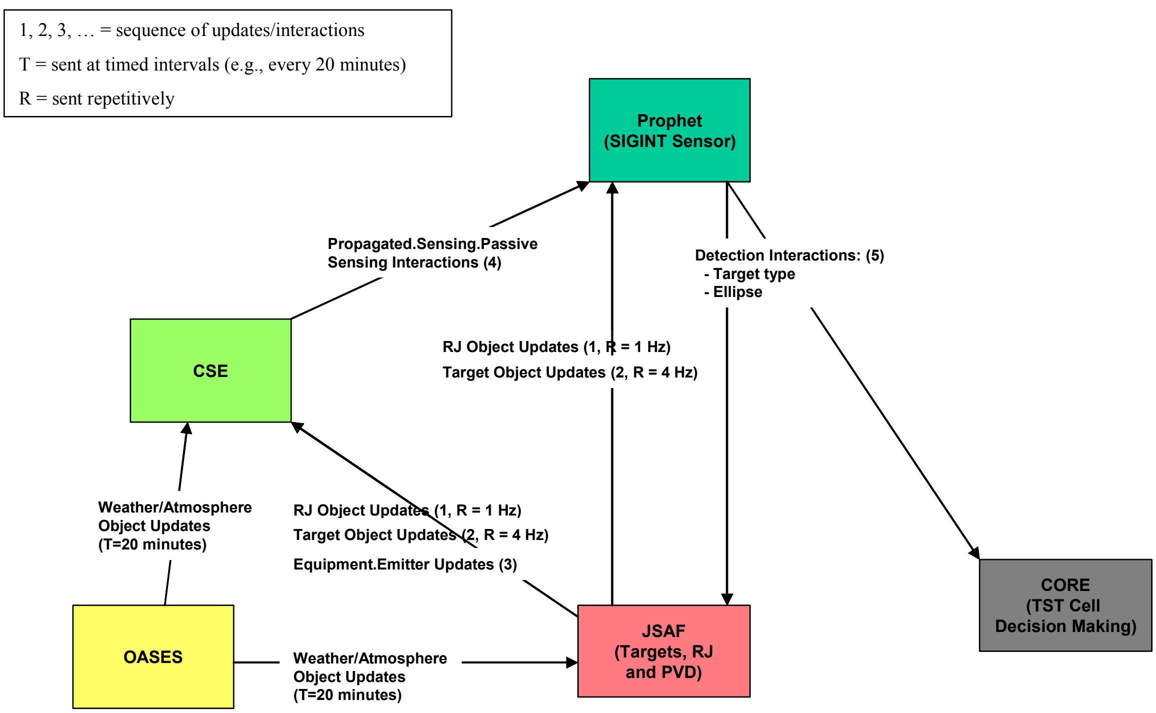


Figure 4-11. Sequence diagram for Rivet Joint.

1. *OASES* sends periodic weather & atmosphere updates to *CSE Services* and *JSAF*
2. *JSAF* sends Rivet Joint platform state updates to *CSE Services* and *Prophet*
3. *JSAF* sends Target position updates to *CSE Services* and *Prophet*
4. *JSAF* sends Target radio / radar emissions to *CSE Services*
5. *CSE Services* sends propagated radio / radar emissions to *Prophet* for the Rivet Joint

6. *Prophet* sends detections of Target radio / radar emissions to *CORE* for the TST Cell
7. *Prophet* sends detections of Target radio / radar emissions to *JSAF* for display purposes

4.2.2 Sequence Diagram for JSTARS

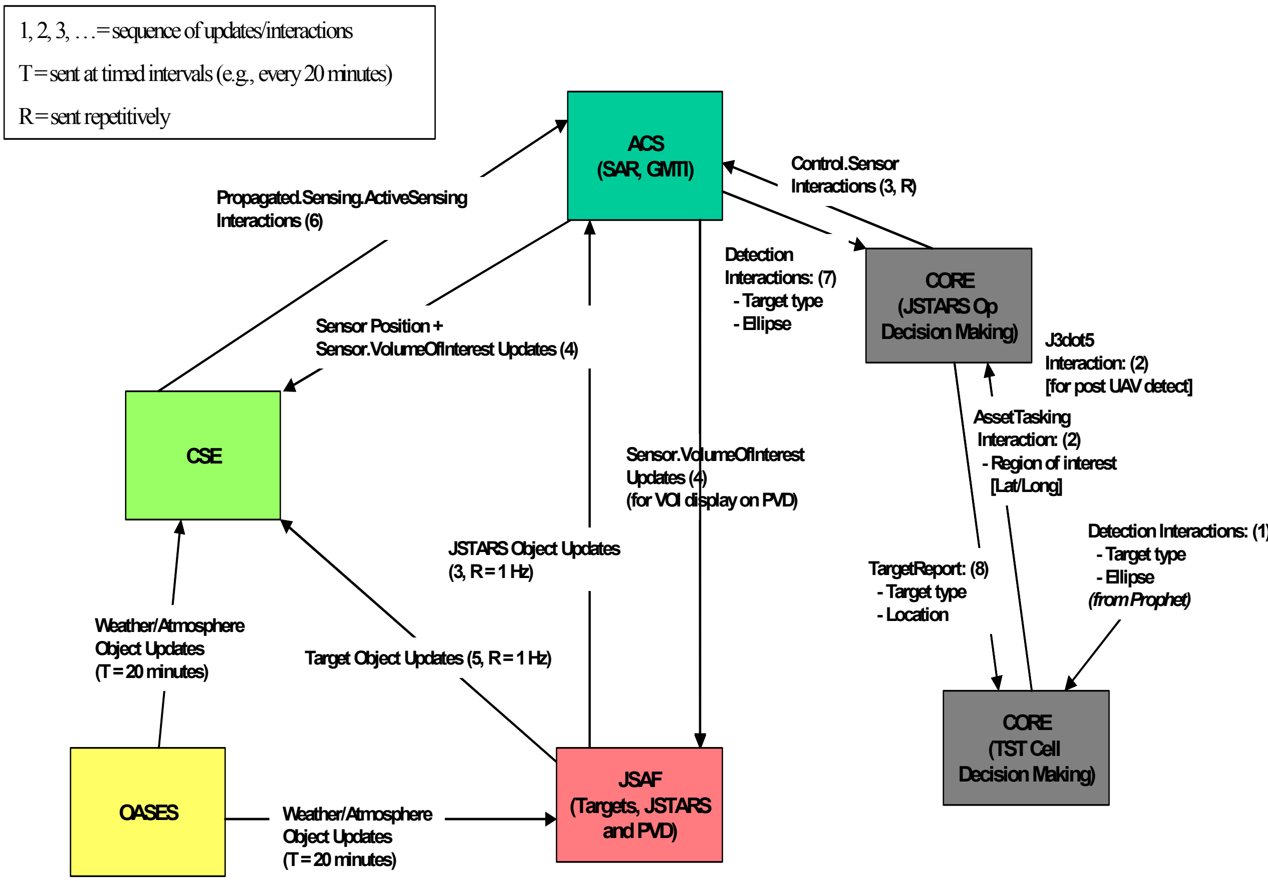


Figure 4-12. Sequence diagram for JSTARS.

1. *Prophet* sends detections of Target radio / radar emissions to *CORE* for the TST Cell
2. *OASES* sends periodic weather & atmosphere updates to *CSE Services* and *JSAF*
3. The TST Cell in *CORE* sends an Asset Tasking or J3.5 message to *CORE* for the JSTARS Operator
4. *CORE* sends Sensor Control interactions to *ACS* for the JSTARS
5. *JSAF* sends JSTARS position updates to *ACS*
6. *ACS* sends JSTARS sensor position and VOI updates to *CSE Services*
7. *ACS* sends JSTARS sensor VOI updates to *JSAF* for display purposes
8. *JSAF* sends Target position updates to *CSE Services*
9. *CSE Services* sends propagated MTI and/or SAR sensing chances to *ACS* for the JSTARS
10. *ACS* sends radar detections of Target objects to *CORE* for the JSTARS Operator
11. *ACS* sends radar detections of Target objects to *JSAF* for display purposes
12. The JSTARS Operator in *CORE* sends a Target Report to *CORE* for the TST Cell

4.2.3 Sequence Diagram for the UAV

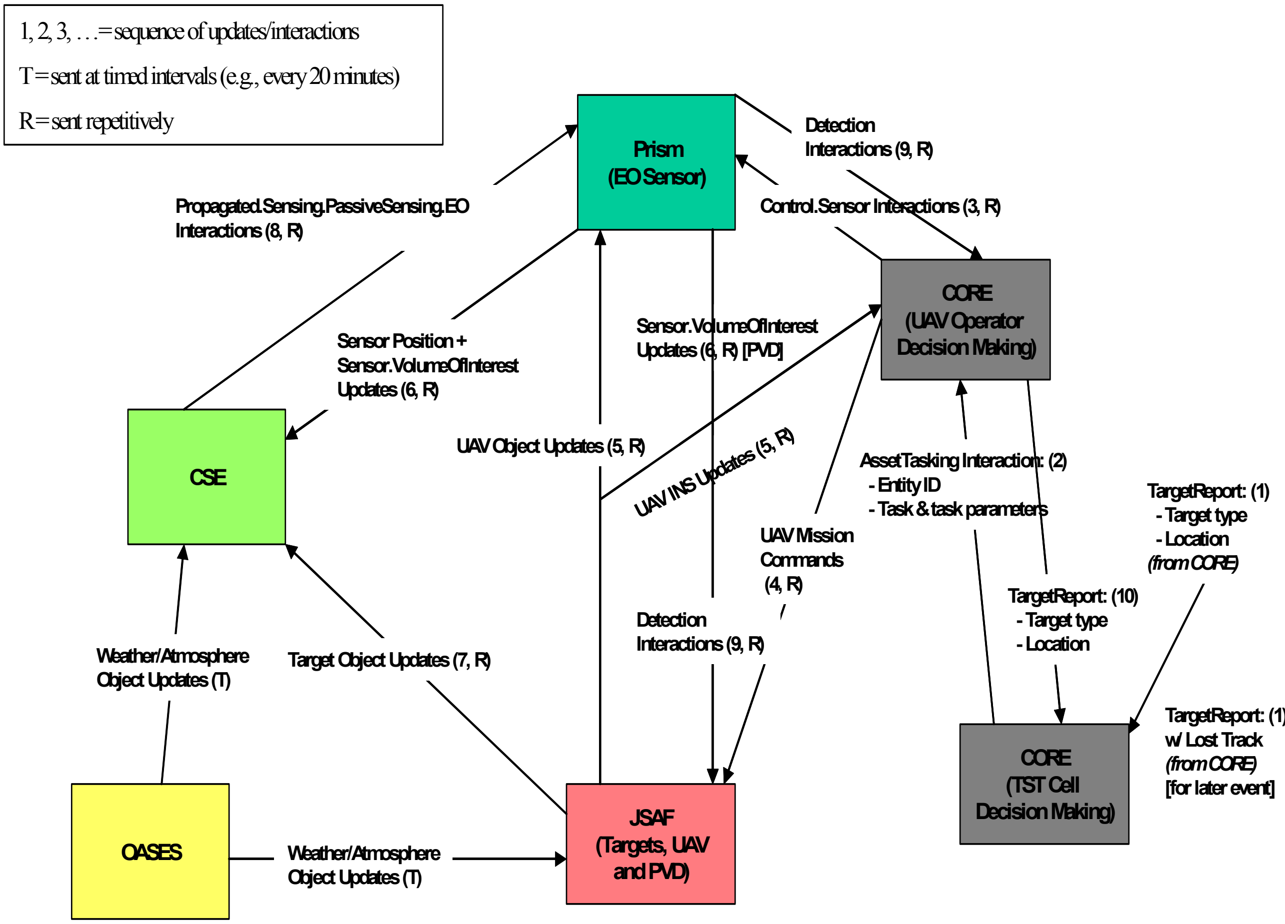


Figure 4-13. Sequence diagram for the UAV.

1. The JSTARS Operator in *CORE* sends Target Reports to *CORE* for the TST Cell (perhaps with a Lost Track indicator)
2. *OASES* sends periodic weather & atmosphere updates to *CSE Services* and *JSAF*
3. The TST Cell in *CORE* sends Asset Tasking to *CORE* for the UAV Operator
4. *CORE* sends Sensor Control interactions to *PRISM* for the UAV
5. *CORE* sends UAV mission updates to *JSAF*
6. *JSAF* sends UAV position updates to *PRISM* and INS updates to *CORE* for the UAV Operator
7. *PRISM* sends UAV sensor position and VOI updates to *CSE Services*
8. *PRISM* sends UAV VOI updates to *JSAF* for display purposes
9. *JSAF* sends Target position updates to *CSE Services*
10. *CSE Services* sends propagated EO sensing chances to *PRISM* for the UAV
11. *PRISM* sends EO detections of Target objects to *CORE* for the UAV Operator
12. *PRISM* sends EO detections of Target objects to *JSAF* for display purposes
13. The UAV Operator in *CORE* sends a Target Report to *CORE* for the TST Cell

4.2.4 Sequence Diagram for F-15E

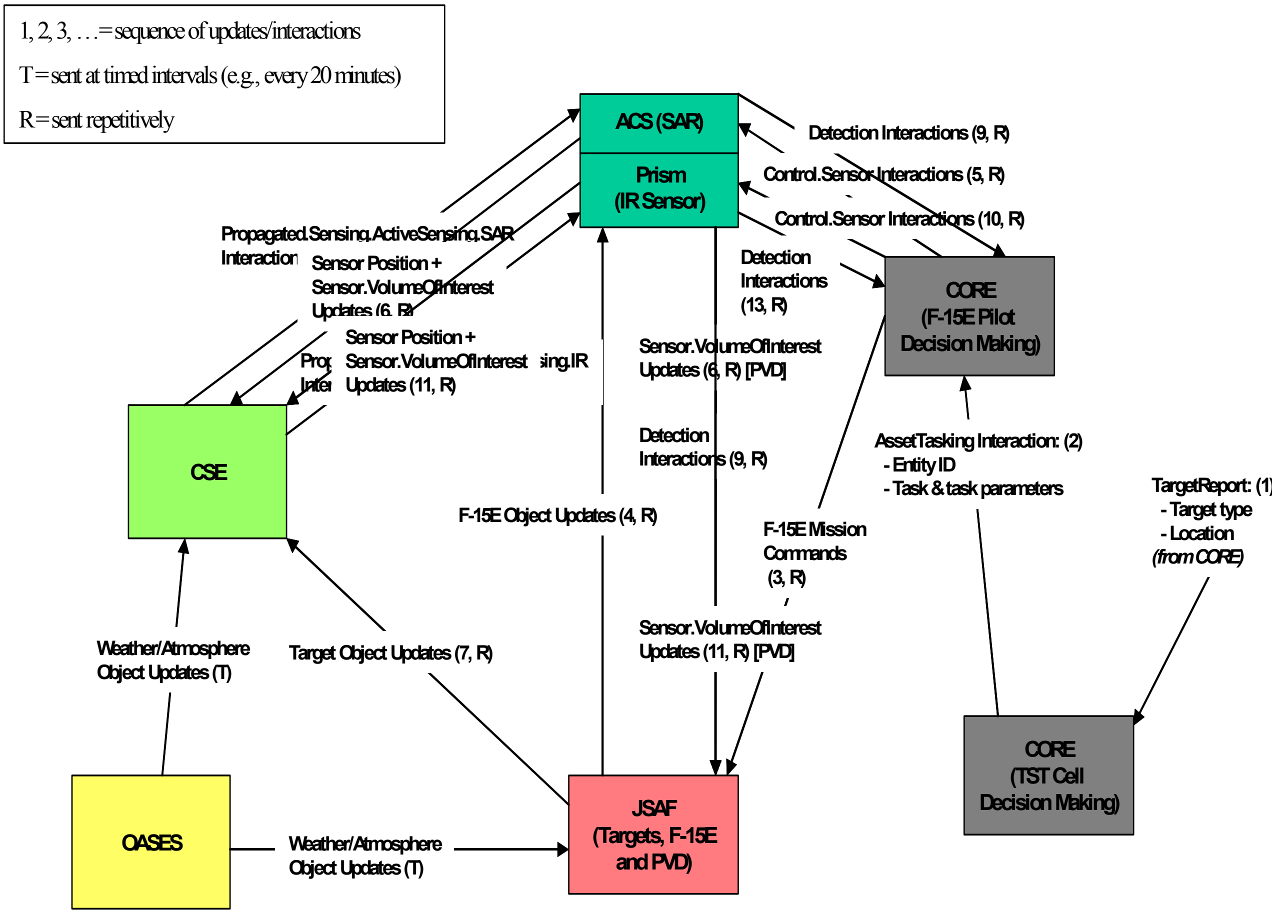


Figure 4-14. Sequence diagram for the F-15E.

1. The UAV Operator in *CORE* sends a Target Report to *CORE* for the TST Cell
2. *OASES* sends periodic weather & atmosphere updates to *CSE Services* and *JSAF*
3. The TST Cell in *CORE* sends Asset Tasking to *CORE* for the F-15E Pilot
4. *CORE* sends F-15E mission updates to *JSAF*
5. *JSAF* sends F-15E position updates to *ACS* and *PRISM*
6. *CORE* sends Sensor Control interactions to *ACS* for the APG-70 Radar
7. *ACS* sends APG-70 Radar position and VOI updates to *CSE Services*
8. *ACS* sends APG-70 Radar VOI updates to *JSAF* for display purposes
9. *JSAF* sends Target position updates to *CSE Services*
10. *CSE Services* sends propagated SAR sensing chances to *ACS* for the APG-70 Radar
11. *ACS* sends SAR detections of Target objects to *CORE* for the F-15E Pilot
12. *ACS* sends SAR detections of Target objects to *JSAF* for display purposes
13. *CORE* sends Sensor Control interactions to *PRISM* for the LANTIRN
14. *PRISM* sends LANTIRN position and VOI updates to *CSE Services*
15. *PRISM* sends LANTIRN VOI updates to *JSAF* for display purposes

16. *CSE Services* sends propagated IR sensing chances to *ACS* for the LANTIRN
17. *PRISM* sends IR detections of Target objects to *CORE* for the F-15E Pilot
18. *PRISM* sends IR detections of Target objects to *JSAF* for display purposes

4.2.5 Start-Up Sequence

This section will describe how JOSEF is brought up, to include:

- the beginning of day computer network checks
- Federation creation
- federate join sequence
- status checks required prior to time advance
- time advance initiation
- data collection integrity checks
- federate resignation process
- Federation destruction
- Annexes with specific run procedures for all Federates and software

4.3 INTERFACE IDENTIFIER MATRIX

The matrix below identifies where interfaces exist between Federates and Simulations. Where the interface is handled via Object Attribute Updates over the HLA RTI, a Green ball (●) is used. Where the interface is handled via Interactions over the HLA RTI, a Magenta diamond (◆) is used. Where a direct API interface is used, a Blue boxed caret (⍓) is used.

Federates							*CORE*	
From: *To:*	*ACS*	*CSE*	*JSAF*	*PRISM*	*OASES*	*Prophet*	*Stand-Alone*	*Embedded*
ACS		●	●◆				◆	
CSE	◆			◆		◆		
JSAF	●◆	●		●◆		●	◆	⍓
PRISM		●	●◆					
OASES		●	●					
Prophet			◆				◆	
CORE (S-A)	◆		◆				◆	
CORE (Emb)			⍓					

ACS to CSE Services

- Equipment.Sensor.MTI_SAR object for APG-8 and APG-70 radar states

ACS to JSAF

- Equipment.Sensor.MTI_SAR object for APG-8 and APG-70 radar states
- Detection.MTI / .SAR interactions for estimated type and position of targets, to include error ellipse

ACS to CORE

- Detection.MTI / .SAR interactions for estimated type and position of targets, to include error ellipse

CSE Services to ACS
- Propagated.Sensing.ActiveSensing interactions for propagated radar returns

CSE Services to PRISM
- Propagated.Sensing.PassiveSensing.EO interactions for propagated EO energy
- Propagated.Sensing.PassiveSensing.IR interactions for propagated IR energy

CSE Services to Prophet
- Propagated.Sensing.PassiveSensing.COMINT / .ELINT interactions for propagated target emissions

JSAF to ACS
- Platform.ACSnode objects for JSTARS and F-15E positions
- Control.Sensor.SAR interactions for the F-15E APG-70

JSAF to CSE Services
- Platform.ACSnode objects for RJ position
- Platform.Target objects for target and ClutterSim PFT position
- Platform.Target objects for Cultural Feature PFT position
- Equipment.Emitter.Radio / .Radar objects for target emissions

JSAF to PRISM
- Control.Sensor interactions to control the UAV EO sensor and F-15E LANTIRN
- Platform.ACSnode objects for UAV position

JSAF to Prophet
- Platform.ACSnode object for RJ position
- Platform.Target objects for target position

JSAF to CORE
- Message.TargetReport interactions for UAV reports on targets

JSAF to Embedded CORE
- *TBD* API calls for communications from the TST Cell to the UAV Operator and F-15E Pilot
- *TBD* API calls for UAV INS updates
- *TBD* API calls for UAV EO and F-15E sensor detections

PRISM to CSE Services
- Equipment.Sensor.EO object for UAV EO sensor state
- Equipment.Sensor.IR object for F-15E LANTIRN sensor state

PRISM to JSAF
- Equipment.Sensor.EO object for UAV EO sensor state
- Detection.EO / .IR interactions for estimated type and position of targets
- Equipment.Sensor.IR object for F-15E LANTIRN sensor state

OASES to CSE Services
- Atmosphere.* objects for gridded weather effects

OASES to JSAF
- Atmosphere.* objects for gridded weather effects

Prophet to JSAF
- Detection.COMINT / .ELINT interactions for estimated positions of emitting targets, to include error ellipse

Prophet to CORE
- Detection.COMINT / .ELINT interactions for estimated positions of emitting targets, to include error ellipse

CORE to ACS
- Control.Sensor.MTI / .SAR interactions from the JSTARS Operator to the APG-8

CORE to JSAF
- Message.AssetTasking interactions for communications from the TST Cell to the UAV Operator and the F-15E Pilot

CORE to CORE
- Message.AssetTasking and Message.J3dot5 interactions for communications from the TST Cell to the JSTARS Operator
- Message.TargetReport interactions for communications from the JSTARS Operator to the TST Cell

Embedded CORE to JSAF
- *TBD* API calls to control the UAV sensor, F-15E APG-70, and F-15E LANTIRN
- *TBD* API calls to maneuver the UAV and F-15E
- *TBD* API calls for target reports from UAV

This page intentionally left blank

5. REQUIREMENTS TRACEABILITY

The following system-level Experiment trace matrix shows the relationship among the goals, objectives, and requirements. Its purpose is as a checklist to ensure the goals and objectives have been addressed by system-level requirements. The matrix is presented in four columns:

- First column contains the Experiment Goals
- Second column contains the Experiment Objectives
- Third column contains the Experiment system-level Requirements
- Fourth column contains the Test Methods associated with each system-level requirement

Each Goal is presented in major row trace path, which contain Objectives sub-rows that contain Requirement sub-rows with identified test methods

Goals	Objectives	Requirements	Test
Come As You Are —Limit investment in time and resources by minimizing new modeling and simulation (M&S) development. Demonstrate reuse and integration of best-of-breed Air Force and DoD simulations and environments.	OT-5 Uses existing / legacy M&S entities	RT-5 be composed of existing components, which require minimal/no modifications to operate within the federation.	Observation
	OT-6 Uses physics-based representations of battlespace entities	RT-6 make reuse of physics-based representations of battlespace entities and phenomenological models.	Analysis
	OR-4 Gain insight into JSB technical challenges and risks to identify implementation and/or technology knees of the curve	RR-3 an opportunity to gain insight into the technical challenges and risk associated with developing high resolution/high fidelity synthetic environment federations and federates. (e.g., what are the high payoff opportunities for improvement in performance prediction for EO/IR sensors attributable to the increased resolution and fidelity of the synthetic target and its environment).	Analysis
Leave Nobody Out —Conduct a collaborative effort that includes government agencies, laboratories, product centers, and industry partners. Demonstrate a cooperative venture.	OP-1 Defines and demonstrates collaborative development	RP-2 be conducted using the Experiment Federation Development and Execution Process (FEDEP), tailored appropriately.	Observation
		RP-1 be conducted as a cooperative effort that includes participants from Government agencies, laboratories, product enters, and industry	Observation
Build on the Joint Combat Identification Evaluation Team (JCIET) 2002 event — Replicate scenario(s)/vignettes from the event; use event data for validating the Experiment Federation; and use the experience to identify the challenges migrating to future (To-Be) operational military architectures.	OT-1 Executes in real-time, as needed	RT-1 in near real-time (e.g., *clock* time) for limited scenario(s) that may involve human operators	Test / Demo
	OT-2 Immerses the operator / warfighter with acquirer / engineer under realistic conditions	RT-8 provide for active warfighter involvement in the federation development, test, and/or operation RP-5 provide for JCIET staff review and absorption of their input into the experiment.	Observation Observation
	OR-5 Perform one significant and representative Use Case for JCIET event rehearsal	RR-4 including a significant and representative scenario for the JCIET event. RR-6 showing an Experiment contribution that would add value to the JCIET event.. RT-3 support operation of a JCIET event scenario/vignette (e.g., may be only one).	Demo Analysis Demo
	OR-7 Demonstrate relevance of the Experiment to Air Force objectives.	RR-2 relevant to Air Force objectives. (e.g., Pds are relevant to the Air Force Attack mission. The targets must be detected before they can be attacked).	Analysis
		RR-6 showing an Experiment contribution that would add value to the JCIET event.	Analysis
Address Fundamental Questions —e.g., "Why JSB (SBA) -- What Are the Discriminators?" "What are the Engineering issues that stress the simulations and challenge the synthetic environment?"	OT-3 Provide repeatable simulations	RT-2 be repeatable (e.g., using the same federation and parameters produce the same results).	Demo
	OR-1 Model technical performance area(s) that occur during JCIET 02 event	RR-1 analysis that yields the MOP values for sensors (in varying environments). RT-4 accommodate variations in environmental conditions (e.g., add/remove fog), modification to the scenario laydown (e.g., relocate targets), etc. — provide for flexible operations.	Analysis Analysis Analysis

Goals	Objectives	Requirements	Test
	OR-2 Demonstrate discriminating value-added of JSB	RR-3 an opportunity to gain insight into the technical challenges and risk associated with developing high resolution/high fidelity synthetic environment federations and federates. (e.g., what are the high payoff opportunities for improvement in performance prediction for EO/IR sensors attributable to the increased resolution and fidelity of the synthetic target and its environment). RR-6 showing an Experiment contribution that would add value to the JCIET event. RR-7 showing how an Experiment rehearsal before the JCIET event could predict/provide better understanding of the performance of the systems in the JCIET event scenario vignette it is applied to. It should identify areas of technical concern or those requiring special attention during the exercise. RT-4 accommodate variations in environmental conditions (e.g., add/remove fog), modification to the scenario laydown (e.g., relocate targets), etc. — provide for flexible operations.	Analysis Analysis Analysis Demo
	OR-5 Perform one significant and representative Use Case for JCIET event rehearsal	RR-4 including a significant and representative scenario for the JCIET, which could be used as a JCIET event rehearsal. (e.g., The tanks under trees scenario will be run through in the JCIET event more than once.).	Analysis
		RR-6 showing an Experiment contribution that would add value to the JCIET event.	Analysis
		RT-3 support operation of a JCIET event scenario/vignette (e.g., may be only one).	Demo
	OR-7 Demonstrate relevance of the Experiment to Air Force objectives.	RR-2 relevant to Air Force objectives. (e.g., Pds are relevant to the Air Force Attack mission. The targets must be detected before they can be attacked).	Analysis
		RR-6 showing an Experiment contribution that would add value to the JCIET event.	Analysis
		RT-4 accommodate variations in environmental conditions (e.g., add/remove fog), modification to the scenario laydown (e.g., relocate targets), etc. — provide for flexible operations.	Demo
Design for Extension —Consider near-term need to support Global Strike Task Force (GSTF) Command and Control (C2) Constellation studies and analysis and Future SBA Requirements.	OP-2 Defines and demonstrates the simulation validation process	RP-2 be conducted using the Federation Development and Execution Process (FEDEP), tailored appropriately. RP-3 be designed to support demonstration of the JSB concept. RP-4 define a verification, validation, and accreditation (VV&A) strategy/approach for the Experiment Federation.	Observation Test Observation
	OP-4 Identifies (and understands) the processes for defining Data for collection.	RT-7 provide for collecting simulation data (e.g., instrumentation) to support the analysis.	Observation
		RP-3 be designed to support validation of the JSB concept	Test

Goals	Objectives	Requirements	Test
	OR-2 Demonstrate discriminating value-added of JSB	RR-3 an opportunity to gain insight into the technical challenges and risk associated with developing high resolution/high fidelity synthetic environment federations and federates. (e.g., what are the high payoff opportunities for improvement in performance prediction for EO/IR sensors attributable to the increased resolution and fidelity of the synthetic target and its environment). RT-12 provide for its synthetic environment to include a high resolution/high fidelity representation of the sub-area in the JCIET event "playbox" where the experiment vignette is exercised.	Analysis Demo
	OR-3 Demonstrate use of JSB to identify JSB (SBA) system requirements and to identify higher fidelity synthetic environment requirements for advanced sensors	RR-3 an opportunity to gain insight into the technical challenges and risk associated with developing high resolution/high fidelity synthetic environment federations and federates. (e.g., what are the high payoff opportunities for improvement in performance prediction for EO/IR sensors attributable to the increased resolution and fidelity of the synthetic target and its environment). RR-8 gain understanding of the M&S components used in the Experiment Federation to represent advanced sensors and dynamic, high fidelity synthetic environments. RR-7 show how an Experiment rehearsal before the JCIET event could predict/provide better understanding of the performance of the systems in the JCIET event scenario vignette it is applied to. It should identify areas of technical concern or those requiring special attention during the exercise. RR-4 include a significant and representative scenario for the JCIET event, which could be used as a JCIET event rehearsal. (e.g., The tanks under trees scenario will be run through in the JCIET event more than once.).	Analysis Analysis Analysis Demo
	OR-7 Demonstrate relevance of the Experiment to Air Force objectives	RR-2 being relevant to Air Force objectives. (e.g., Pds are relevant to the Air Force Attack mission. The targets must be detected before they can be attacked).	Analysis
		RT-4 accommodate variations in environmental conditions (e.g., add/remove fog), modification to the scenario laydown (e.g., relocate targets), etc. — provide for flexible operations.	Demo
Identify JSB (SBA) Requirements—Demonstrate JSB (SBA) concepts. Demonstrate use of JSB (SBA) to identify JSB (SBA) System Requirements and identify higher fidelity synthetic environment requirements for advanced sensors.	OP-1 Defines and demonstrates collaborative development	RP-1 be conducted as a cooperative effort that includes participants from Government agencies, laboratories, product enters, and industry	Observation
	OR-2 Demonstrate discriminating value-added of JSB	RR-3 an opportunity to gain insight into the technical challenges and risk associated with developing high resolution/high fidelity synthetic environment federations and federates. (e.g., what are the high payoff opportunities for improvement in performance prediction for EO/IR sensors attributable to the increased resolution and fidelity of the synthetic target and its environment).	Analysis

Goals	Objectives	Requirements	Test
		RR-7 show how an Experiment rehearsal before the JCIET event could predict/provide better understanding of the performance of the systems in the JCIET event scenario vignette it is applied to. It should identify areas of technical concern or those requiring special attention during the exercise..	Analysis
		RT-12 provide for its synthetic environment to include a high resolution/high fidelity representation of the sub-area in the JCIET event "playbox" where the experiment scenario vignette is exercised.	Demo
	OR-4 Gain insight into JSB technical challenges and risks to identify implementation and/or technology tradeoffs	RT-4 accommodate variations in environmental conditions (e.g., add/remove fog), modification to the scenario laydown (e.g., relocate targets), etc. — provide for flexible operations.	Analysis
		RT-12 provide for its synthetic environment to include a high resolution/high fidelity representation of the sub-area in the JCIET event "playbox" where the experiment scenario vignette is exercised	Demo
		RR-7 showing how an Experiment rehearsal before the JCIET event could predict/provide better understanding of the performance of the systems in the JCIET event scenario vignette it is applied to. It should identify areas of technical concern or those requiring special attention during the exercise.	Demo

This page intentionally left blank

6. NOTES

This section shall contain any general information that aids in understanding this document (e.g., background information, glossary, rationale). This section shall include an alphabetic listing of all acronyms, abbreviations, and their meaning as used in this document and a list of any terms and definitions needed to understand this document.

6.1 ABBREVIATIONS AND ACRONYMS

[Add list here]

6.2 [POINTS OF CONTACT (POCS)]

POCs that will be involved in the transition efforts are identified in the tables that follow:

TABLE 8-1. [PROJECT SPONSOR] POCS

NAME	CODE	PHONE (DSN:nnn)

TABLE 8-2. [SSA] POCS

NAME	CODE	PHONE (DSN:nnn)

TABLE 8-3. [COMMAND TURNING PRODUCT OVER TO SSA - DEVELOPING ACTIVITY] POCS

NAME	CODE	PHONE (DSN:nnn)

7. APPENDIX A. COMMON SYNTHETIC ENVIRONMENT SERVICE SOFTWARE DESIGN

7.1 INTRODUCTION

This document describes the design of JOSEF's Common Synthetic Environment (CSE) Service. The following section will describe the basic setting of the experiment and describes the need for a CSE service in order to meet the experiment's goals. In particular, the difficult part of the experiment – the inclusion of high-fidelity sensor modeling will be described and their use of multispectral environment databases.

7.1.1 BACKGROUND

See Section 1.2.1.

7.2 CSE DESIGN DECISIONS

For the acquisition process to function effectively in a varied and changing world, multifaceted scenarios must be created in predictive, accurate, tailorable, and repeatable synthetic environments. These synthetic environments must provide the authoritative framework to support Program Manager responsibilities established in AFI 16-1002, providing a validated environment for contracting and engineering. Technically, the JSB must focus on model integration, scenario validity and availability, accessibility, interactive real-time adjustment, repeatable results, affordability, adaptability, and realism. Operationally, the capability must focus on being part of an integrated environment that brings together analysis, assessment, training, and the other functions into a coherent whole.

A critical need for a JSB is to provide for the resolution of disparate simulation environments to achieve consistent and accurate environmental representations between and among the simulations within a federation—whereas today almost all simulations possess unique synthetic environment representations, and environment resolution for realistic interoperability present a serious technical challenge. The JSB needs to establish a set of open standards and best practices, along with explicit capabilities, for implementing a common synthetic environment for high fidelity sensors that may be validated against live test results.

7.2.1 Common Synthetic Environment (CSE) Design Overview

The Common Synthetic Environment (CSE) federate will model the environment elements that are used to compose the synthetic battlespace. These elements include the synthetic environment data, that includes the natural, cultural, and material-codes, and terrain data; meteorological, weather, and atmospheric data; and astronomic and stellar data; and the data services necessary to reflect the environment within which the systems and system components interoperate, at the appropriate level of detail for the simulation/federation. The data services include effects models that describe the interactions between the environmental elements and the specific system simulations. Effects models include aerodynamic drag, radar and optical backscatter, radio propagation, ballistics, etc. Accurate physics-based models are necessary for these to be correct.

The CSE architecture will be based upon the Environment Architecture that was developed under the Defense Advanced Research Project Agency's (DARPA) Synthetic Theater of War Initiative and that has been further extended under the Defense Modeling and Simulation Office's (DMSO) Integrated Natural Environment initiative. This existing architecture is based upon a flexible, extensible, easy-to-use Environment Application Programmer's Interface (API) to access the Synthetic Environment features of the virtual world. The top-level design requirements for the existing architecture and its API are enumerated below.

1. The Environment API must *enable maximum reuse* of the software developed in this project, encourage the addition of further environmental effects into Advanced Distributed Simulation, and ease the effort required in adding environmental effects to existing simulators.
2. The Environment API must be flexible enough to *support multiple fidelity levels* in models of environmental phenomena, since no one level of fidelity can satisfy all the needs of the Advanced Distributed Simulation community.
3. The Environment Architecture must enable various environment models to be *added or removed* from the simulation *at exercise configuration time*. This is because different simulations have different requirements, due to the either operational need or system constraints placed by CPU load, network bandwidth, or interoperability with legacy simulators.

To meet these design requirements, the Environment Architecture separates the implementation of the environmental phenomena from the mechanism by which the environmental support is accessed by other applications in the federation. This separation isolates the choice of models from the rest of the federation.

The Virtual World Interface Package, or VWIP, is an Environment API. The VWIP provides both an Application Programmer's Interface (API) to the environment models and architectural support for these models. The separation of the model implementation from the federation's environmental access is implemented in VWIP through the concepts of generic model registration and generic environment parameter access and effect creation. Generic model registration allows the incorporation of alternative or cooperating models for environmental phenomena and effects. Multiple models may be registered to support the simulation of a particular environmental phenomenon. The environmental modeling federate can be configured to enable or disable models for a given federation. As a result, the federation is unaware of which models have been configured to service environment requests.

For the JSB Experiment Federation, this existing architecture will be augmented with additional models (e.g., IR and RF transmittance models) and multi-spectral data to provide the required environmental effects to the JSB sensor models. The resultant architecture is shown below in Figure 7-1.

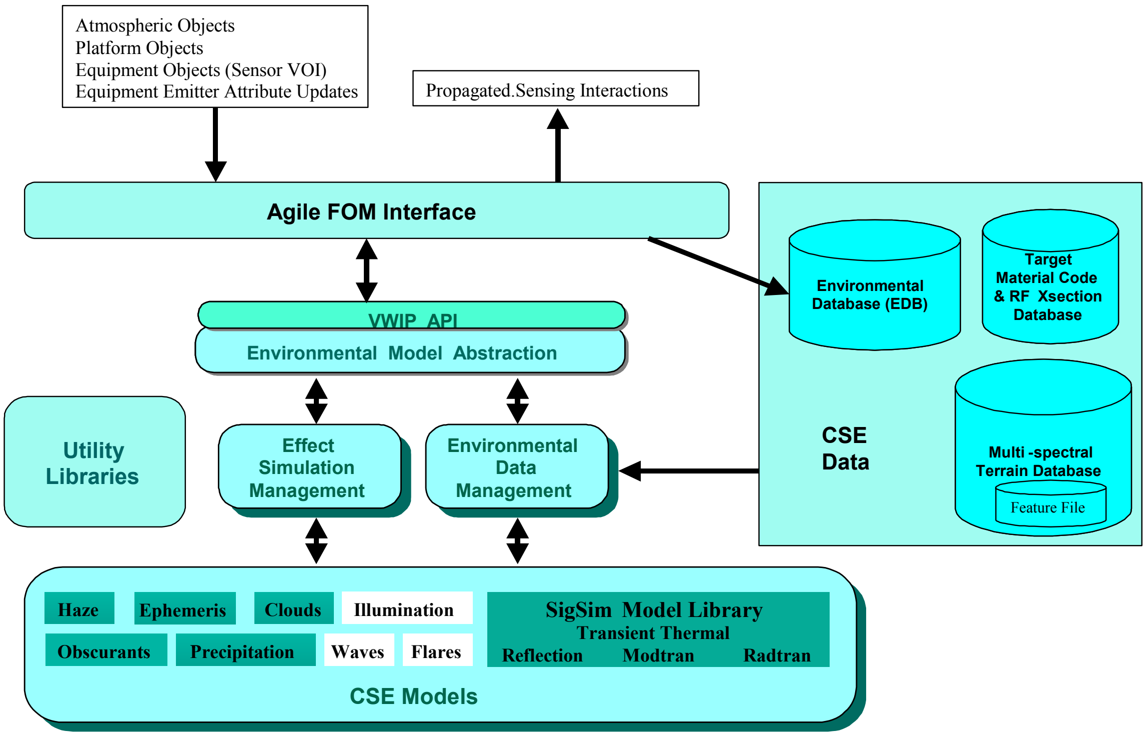


Figure 7-1. Top Level CSE Design (the models highlighted in white are not used in Experiment 1).

7.3 CSE ARCHITECTURAL DESIGN

The following sections provide the architectural design of the CSE including:

- Overviews of the major CSE components:
  - Data
  - Models
  - Utilities
- Descriptions of the primary internal APIs.

7.3.1 CSE Components

The following sections provide a design overview of the major components of the CSE. The components are grouped into three major categories:

- CSE Data – those components that represent state data of various elements of the synthetic battlespace (e.g., terrain, cultural features, atmosphere, targets, etc.).
- CSE Models – those components that utilize CSE Data to model environmental effects required by the JSB sensors.
- CSE Utilities – those components that provide utility functions for external communications and operating upon CSE Data.

7.3.2 CSE Data

The following sections provide an overview of the three CSE Data components.

7.3.2.1 Terrain

The JSB Experiment Federation Terrain Database provides for polygon representation with multi-spectral attribution and abstract features. The JSB Experiment Federation Terrain Database will be pre-generated, and includes:

- Two resolutions of terrain in the database:
    - High Resolution Postage Stamps Triangular Irregular Network (TIN) – elevation data is captured at irregularly spaced intervals (i.e., not using fixed elevation grid spacing) to obtain an extremely high degree of correlation with the physical elevations of the real world terrain.
    - Medium Resolution: The areas surrounding the High Resolution Postage Stamps where the elevation data will generally be represented in a regularly spaced grid, potentially with some TINed areas along linear or areal features.
- Man-made features: static civilian vehicles, farmhouses, utility buildings, storage tanks and farm equipment and roads
- Natural Features: trees, treelines and woodlots

Features are represented with one meter accuracy (DFAD Level 5), and the terrain at DTED Level 1 (30 meter accuracy). Each polygon in the two Postage Stamp areas has been attributed to support generate IR background temp, IR clutter, RF background clutter: sigma zero near and wide. Table 7-1 provides the attribution that has been assigned, and the 48 unique Material Systems that will appear in the terrain database. A Material System is a 1D layered set of contacting materials, with boundary conditions at top and bottom layers. An example is an AsphaltRoad material System: consisting of unpainted black asphalt, 2 inches thick, over a 2 inch gravel base over clay down to an isothermal layer. Material Systems permit the assignment of physical properties to any layer, so that thermal models can be run on them.

The database source code is compiled to produce the CSE's run-time **c**ompact **t**errain **d**ata**b**ase (ctdb) in Global Coordinate System (GCS) coordinates. GCS is a set of local Cartesian coordinate systems which jointly cover the earth's surface. At low latitudes, each local system covers an area of one degree latitude by one degree longitude, with integral latitude and longitude boundaries. At the equator, each local system spans an area approximately 112km square. At greater latitudes, a local system spans multiple degrees of longitude, maintaining a nominal width of 112km. Each local system is tangent to the WGS84 reference ellipsoid at the center of its region, i.e., at the half degree latitude and middle longitude marks. Each of these regions is a *GCS cell*, and the point of tangency to the ellipsoid is the *GCS cell origin*; each cell has a unique *GCS cell id*. See Figure 7-2 for an example.

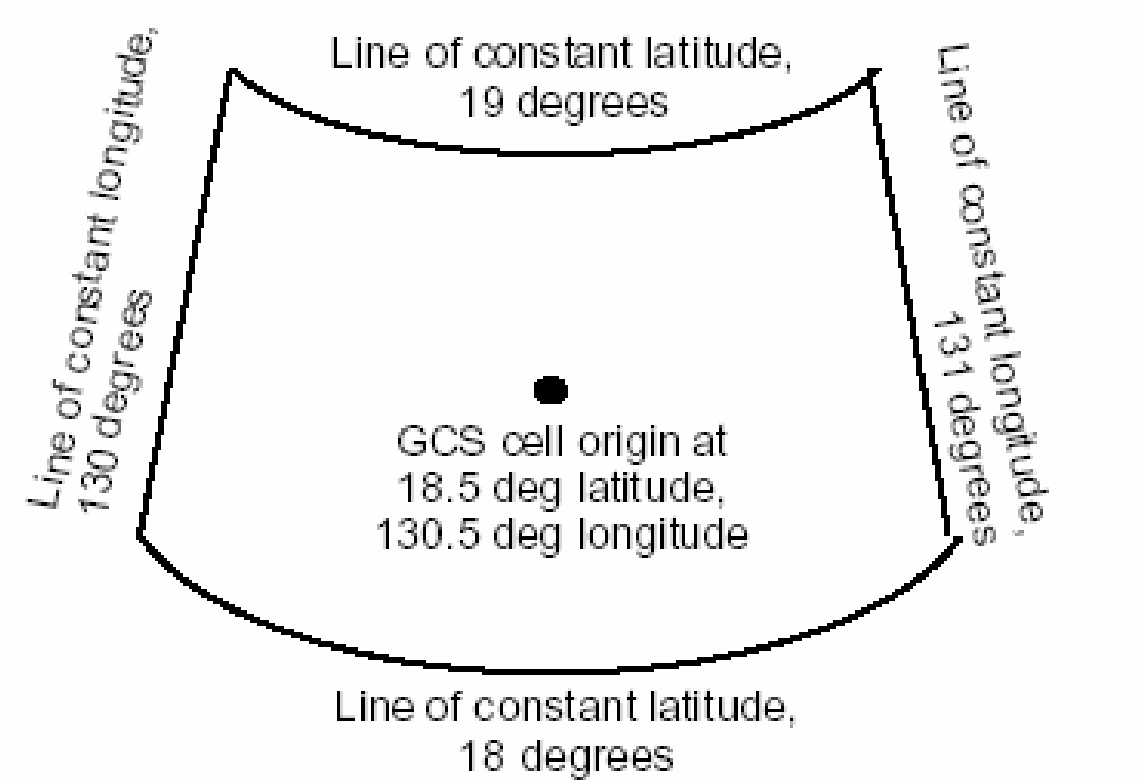


Figure 7-2: A GCS cell is a nominal one-degree latitude by one-degree longitude area of the earth's surface. The GCS cell origin is the center of the cell. Latitude curvature is exaggerated.

Information in the ctdb is represented as terrain skin (the terrain polygons), linear features (roads, rivers, treelines, etc.), point features (trees, power poles, etc.), and volume features (buildings, forests, etc.). Besides the polygonal representation, the terrain skin, linear features, and point features are augmented by a Polygon Attribute Table (PAT) that contains the material code attribution shown in Table 7-1. Currently, the ctdb only stores roofline/canopy top vertices for volume features, meaning that the polygonal representation and attribution of building faces is not stored. As result, the ctdb will be augmented with a flat file (generated by the Synthetic Environment Evaluation – Inspection Tool (SEE-IT)) that will contain the volume model name, surface area and attribution for each face, and a surface normal).

Index#	Noun Description	SMC_	MCC_	MCU_	RST_	RST_	VEG_
1	4 Lane Highway (ConCrete)	21	46	48	998	11	998
2	2 Lane Road (Asphalt)	5	46	48	998	9	998
3	Airfield Runway (Asphalt)	5	46	48	9	9	998
4	Landing Strip (Grass)	255	153	48	18	18	998
5	Country Road (Sand/Gravel Mix)	120	153	48	998	20	998
6	Dirt Road (Soil)	104	153	48	998	18	998
7	Deciduous Canopy (Leaves, Oak)	204	150	48	998	998	24
8	Coniferous Canopy (Leaves, Pine)	204	152	48	998	998	12
9	Coniferous Tree (Pine)	201	150	48	998	998	31
10	Deciduous Tree (Oak)	201	152	48	998	998	30
11	Moderate Deciduous Trees (Oak)	201	150	48	998	998	30
12	Moderate Coniferous Trees (Pine)	201	152	48	998	998	31
13	Scrub Field (Herbaceous/Scrub)	200	154	998	998	19	51
14	Grassy Field (Grass not Graded)	256	104	48	998	5	8
15	Dirt Field (Soil)	104	104	48	998	19	998
16	Sandy Field	88	88	48	998	27	998
17	Wetland	202	116	65	998	5	72
18	House Trailer (w/Vinyl Siding)	130	2	999	998	998	998
19	Roof, Metal	64	143	999	998	998	998
20	Building, Roof, Asphalt	5	98	117	998	998	998
21	Building, Roof, Lumber	56	98	117	998	998	998
22	Building, Roof, Masonry	62	131	117	998	998	998
23	Building, Roof, Gravel	46	5	999	998	998	998
24	Building, Roof, Lumber	56	56	117	998	998	998
25	Building, Walls, Concrete, Painted	205	56	999	998	998	998
26	Building, Walls, Lumber, Painted	205	21	999	998	998	998
27	Building, Walls, Lumber	56	56	999	998	998	998
28	Building, Walls, Vinyl Siding	130	56	999	998	998	998
29	Building, Walls, Concrete	21	56	999	998	998	998
30	Building, Walls, Masonry	62	21	999	998	998	998
31	Building, Walls, Masonry with Lumber Backing	62	62	117	998	998	998
32	Building, Walls, Concrete	21	21	999	998	998	998
33	Building, Walls, Metal	64	64	999	998	998	998
34	**Park Playground Equipment (Painted)**	**205**	**64**	**999**	**998**	998	**998**
35	Lakes, Rivers, Streams	116	99	65	998	998	998
36	Dam (Reinforced Concrete)	83	83	83	998	998	998
37	**Farming Equipment/Trucks (Painted)**	**205**	**64**	**999**	**998**	998	**998**
38	Large Vehicle Tires	85	64	48	998	998	998
39	Water Storage Tank (Painted)	205	64	116	998	998	998
40	Cut Trees (on truck beds)	117	64	999	998	998	998
41	Cut Trees (In piles on ground)	117	104	48	998	998	998
42	Concrete Parking Lot	21	46	48	998	11	998
43	Asphalt Parking Lot	5	46	48	998	9	998
44	Gravel Parking Lot	46	46	48	998	20	998
45	Piled Irrigation Pipe-1	21	999	21	998	998	998
46	Piled Irrigation Pipe-2	64	999	64	998	998	998
47	Big Satellite Dish	72	150	999	998	998	998
48	Dash-8 Airplane	1	2	999	998	998	998

Table 7-1. Material System Attribution in the Terrain Database.

7.3.2.2 Target and Clutter Vehicle Data

The CSE will contain a database of material code attribution and radar cross-sections for the targets and clutter vehicles that will be simulated in the JSB Experiment Federation. The database will include:

- Radar cross-sections as generated by Xpatch®. Xpatch is a set of prediction codes and analysis tools that use the shooting-and-bouncing ray (SBR) method to predict realistic far-field and near-field radar signatures for 3D vehicle models.
- Material Systems attribution, on a per face basis, for 3D models of the vehicles of interest. These Material Systems attributes specify the materials encountered at, and below, the surface of each face, as well as the thermal boundary conditions associated with each layer interface.

7.3.2.3 Atmospheric Data

The Experiment Federation Atmospheric Data will be provided by the OASES federate with extensions for the RF and IR spectrum. An OASES federate serves the dynamic 3D grid atmospheric data which is used as input for many of the CSE's models (e.g., atmospheric models, weather models, etc.). The database of Atmospheric Data that OASES will serve will be generated using both historical data (i.e., one year prior) and Radiosonde data. The resultant database will have a temporal resolution of twenty minutes.

The spatial resolution of the 3D atmospheric grid will vary in resolution. The vertical layers of the grid will appear at the following altitudes (ft):

100, 250, 500, 1000, 2000, 5000, 10k, 20k, 40k

The horizontal resolution of the grid will vary within the game area as follows:

- Resolution in Postage Stamp Areas – 3 Km * 3 Km
- Outside of Postage Stamp Areas – 27KM*27KM (Low Resolution)

The data that will be served by OASES is described in Section 4.1.3.

When the Atmospheric data is received from OASES through the RTI, the CSE stores the data internally in the Environmental DataBase (EDB). In the EDB, the data is stored as a single grid of values which covers the entire terrain database in two dimensions with a configurable number of altitude layers. This grid has fixed pre-configured spacing in the X, Y, and Z dimensions, as served by OASES. The grid of values that can be stored cover a predefined set of environmental parameters, as specified in the FOM. The interface to the EDB allows the grid points for a single environmental parameter to be filled in (or stored) one at a time. Thus, each Atmosphere object, which would contain data for a single environmental parameter, would fill in portions of the EDB grid, and other portions of the EDB grid would remain unset.

7.3.3 CSE Models

The electromagnetic signature appropriate to a given target or terrain object depends, in general, upon a number of sources and physical processes. Extraterrestrial radiation (i.e. solar, lunar, and starlight), as well as terrestrial radiation (from man-made objects such as streetlights, radar transmitters, and thermal sources to natural processes such as fire and lightning), propagate through the atmosphere to illuminate the object of interest. Some of this radiation is absorbed, heating the object and increasing its thermal emission, while the rest is either reflected from the surface or transmitted through the object to other objects or thermal sinks. Some of these processes are illustrated below in Figure 7-3.

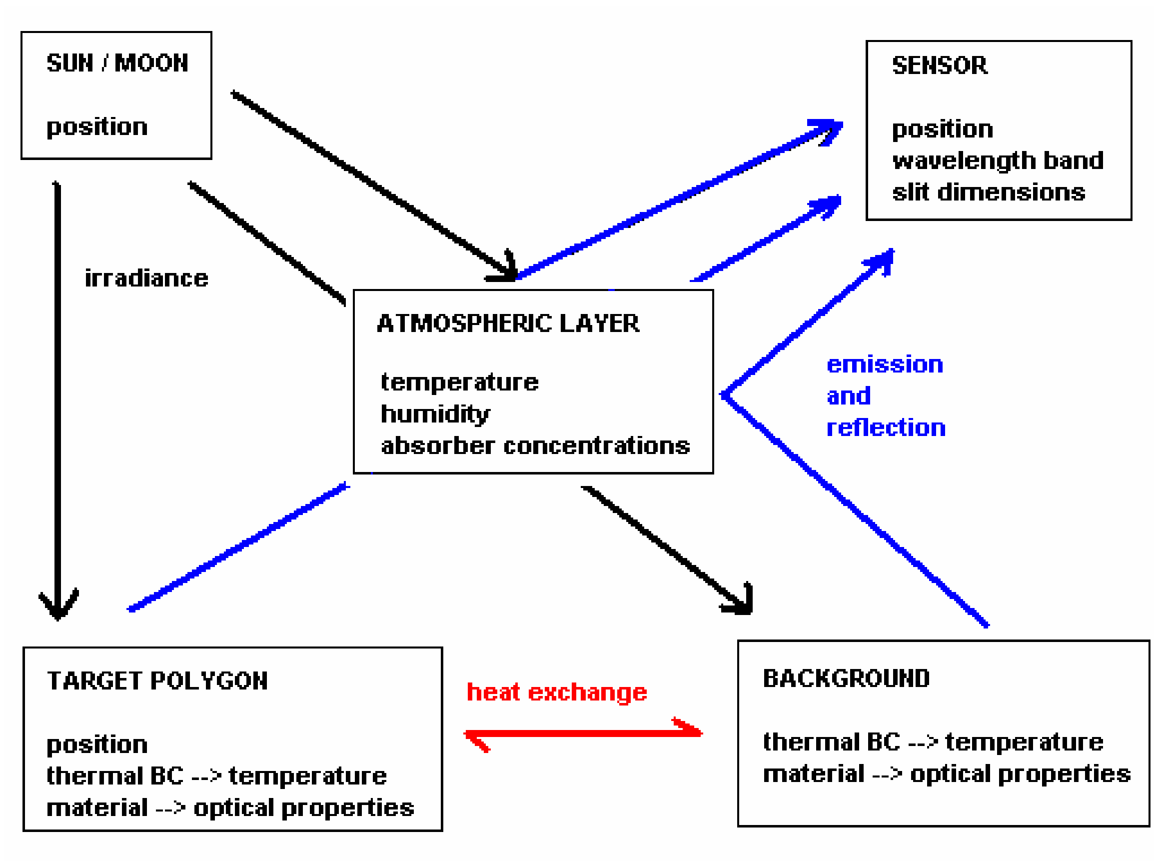


Figure 7-3. Examples of Sources and Physical Properties That Affect Signature.

In practice, solution of such a complex coupled system is facilitated by modeling the component processes separately. In particular, for dynamic, reactive signature prediction within time constraints, simplifying approximations are routinely made within each model which may or may not differ from those applied to other processes. There may be more than one way, with different conceptual constructs and approximations, to model a particular process.

The question of how to break the coupled electromagnetic problem into component processes is separate from that of deciding upon the method of modeling each process. The CSE approach is to answer the first question via a set of API's, each of which supplies both the input parameters which define the process's context, and the output quantities of interest which result from the model's solution of the problem. The latter are useful as inputs to the next stage. As long as an API is sufficiently robust and well-defined, the intra-CSE architecture may remain unchanged as underlying models are compared or replaced as computational methods improve.

Within the CSE, the following component processes have been identified, and will be more thoroughly discussed in this section: Atmospherics, weather, radar cross section determination, radar wave propagation, ephemeris, thermal absorption and emission, and reflection. Each subsection will provide a description on the particular model being employed for Experiment 1 and a description of the API parameters and output.

7.3.3.1 Ephemeris Model

The observer-relative position of the sun and moon is accurately modeled; even to the point of correctly simulating eclipses, as illustrated in Figure 7-4. The Ephemeris Model accepts as input the geographic location, given in Global Coordinate System (GCS) coordinates, as well as date and time, and produces the azimuth and elevation of the sun and moon in the virtual sky, as outlined in Figure 7-5. The location of the sun and moon are calculated based on the well-known orbital paths of the earth and moon.

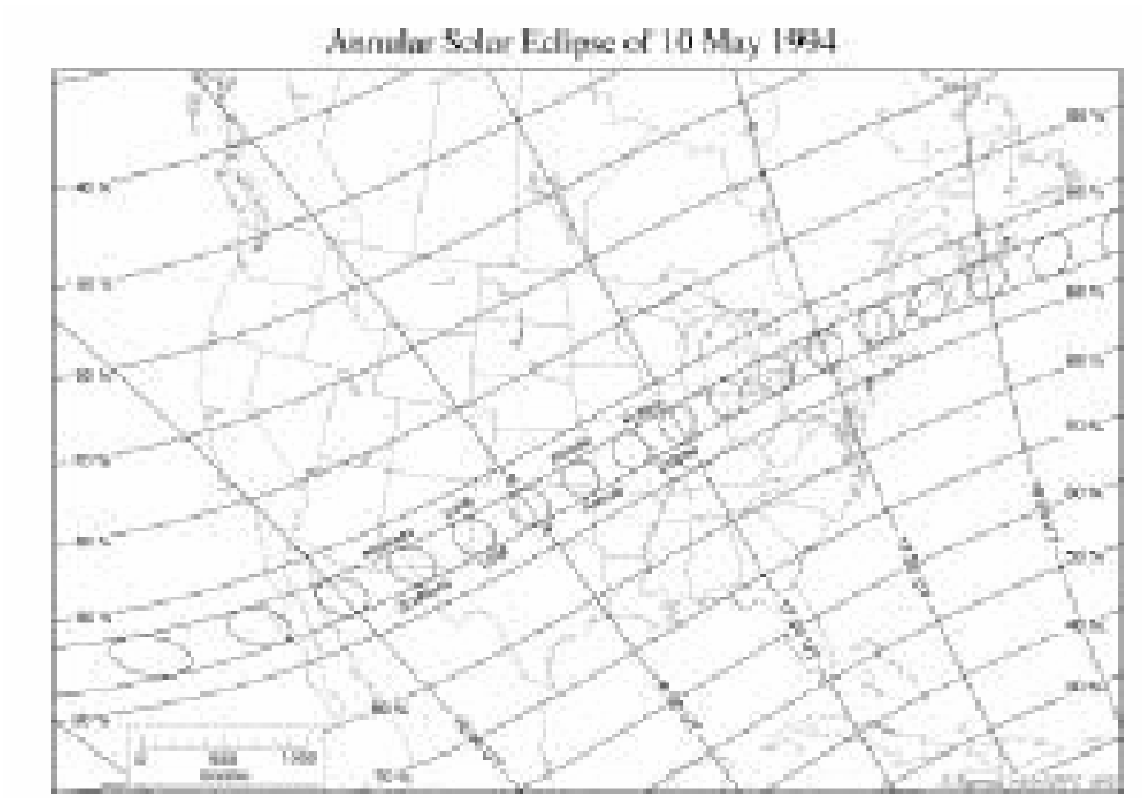


Figure 7-4: The Ephemeris Model accurately models eclipses.

The Ephemeris Model also computes the moon's phase. This is important to modeling nocturnal illumination – vital in night time military operations. The output of the Ephemeris Model directly feeds the Modtran model, described in the next section.

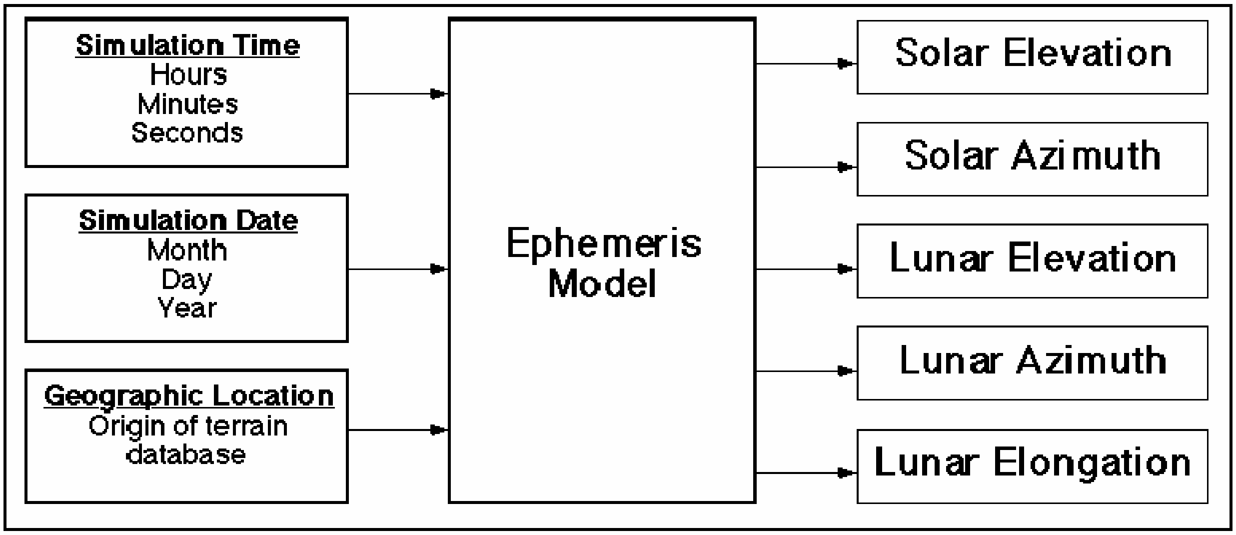


Figure 7-5: Ephemeris Model inputs and outputs.

7.3.3.2 Atmospheric Models

The Earth's atmosphere significantly affects signal propagation via molecular absorption and scattering. Macroscopically, the real part of the atmosphere's index of refraction bends the radiative path out of correspondence with the *in-vacuo* "line-of-sight". The complex part contributes to absorptive attenuation of the signal. Due to gravity, the molecular concentrations, and consequently the atmospheric pressure, temperature, and index of refraction vary with altitude, requiring specification of atmospheric "profiles" for proper representation of each. In addition, the disruptive effects of clouds, rain, and fog may occur, and these phenomena are characterized by localized water vapor or droplet profiles.

Even in the absence of sources, atmospheric molecules also emit thermal radiation, with temperature-dependent spectra. This "downwelling" or "upwelling" IR must be taken into account for proper prediction of the irradiance on an object of interest.

Due to the wide range of "particle size" vs wavelength ratio encountered, there are differences between the traditional modeling approaches to radar as opposed to EO/IR propagation. For this reason, Experiment 1 will employ two atmospheric modeling engines: Modtran (for EO/IR) and Radtran (for RF).

7.3.3.2.1 Modtran

MODTRAN is a state-of-the art atmospheric radiation model developed by the United States Air Force Research Laboratory. With a spectral resolution (FWHM) of 50 to 2 cm-1, this environmental engine calculates atmospheric thermal radiance, molecular absorption and scattering for evaluation of transmittance, and direct solar or lunar irradiance (given the ROI latitude, longitude, day of year and time of day). It is based upon a sophisticated molecular band model, and is equipped with absorption line data from HITRAN (the AFGL line atlas) for the twelve most significant atmospheric molecules. A variety of default atmospheres, which include further aerosols as well as cloud, ice particle, rain, and fog profiles, are provided based upon user-accessible seasonal, visibility, wind speed and rain rate options (among others). Alternatively, and more importantly for the current application, MODTRAN allows full input of user-specified atmospheric profiles, which specify the temperature and air pressure at each layer, as well as layer-specific densities, extinction and absorption coefficients, and asymmetry parameters for each aerosol, water state or droplet size, including CFC's and heavy molecules as well as user-defined aerosols.

Atmospheric and weather data, served by OASES and stored within the CSE's EDB, will be the source for definition of the current atmospheric state. Because this data is based upon homogenous atmospheric "cells" (i.e. its data distinguishes both horizontal and vertical profiles), while Modtran assumes purely vertical stratification, there necessarily needs to be some method of correlating the two if one is going to run the Modtran engine on the data. This problem is alleviated upon realization that any particular point along a radiative path falls within exactly one cell or Modtran "layer". Thus, for a given sensor orientation and position, SigSim can calculate the refracted path, determine which cells this path visits, and define a Modtran layer of corresponding molecular composition, with layer thickness given by the vertical projection of the path (see Figure 7-6 below). The same process is used for RF propagation.

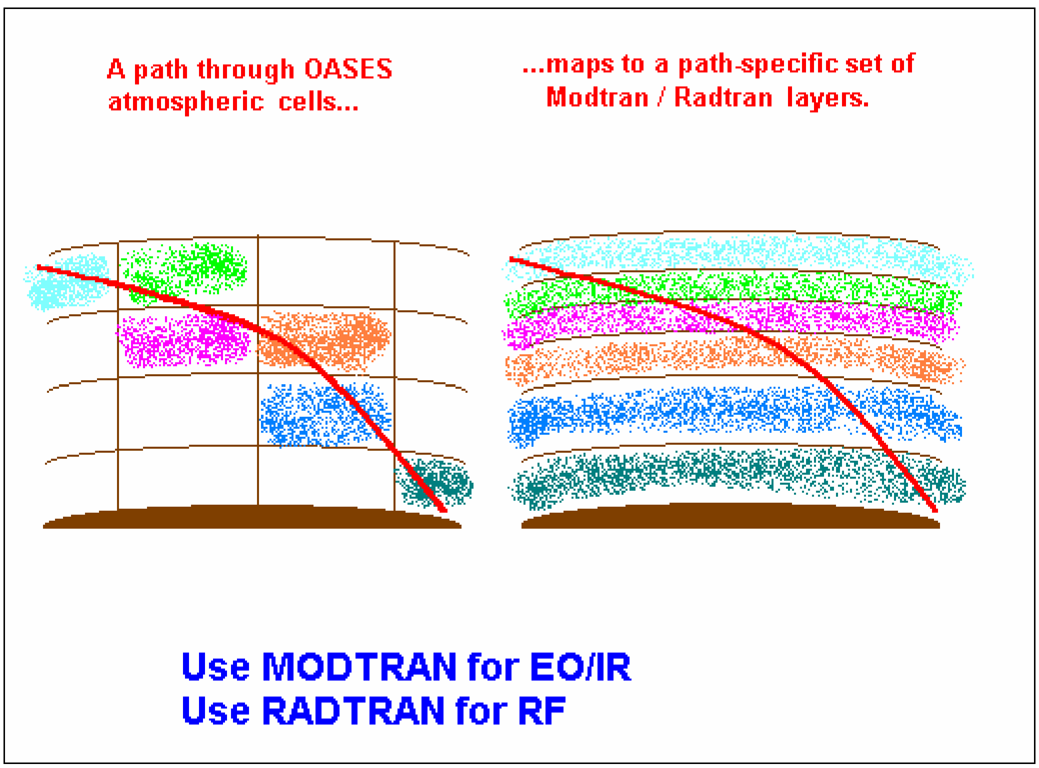


Figure 7-6. Mapping of OASES Atmospheric Cells to Modtran / Radtran Layers.

Once the atmosphere and path are defined, subsequent calls to Modtran return spectral arrays for each of the following: direct source irradiance (properly attenuated), atmospheric thermal radiance, transmittance, and non-direct irradiance that is nevertheless scattered into the path. Both single and multiple-scattering effects may be included, and proper refraction of the radiative path is maintained at all times.

For this mode, three API's are required:

- The Cell Conversion API must, for each cell, supply the types and concentrations of molecular constituents, the pressure, relative humidity, and temperature. The conversion output will supply these quantities for each *layer*, as well as the effective vertical dimension of the layer relative to the path. Sensor orientation information is provided elsewhere, as is information related to clouds and obscurants. Differences between Radtran and Modtran with respect to the API are expected to be minor, if present at all.
- The Atmospherics API inputs this layer-specific information and returns the transmittance and atmospheric thermal path radiance along the path. As above, sensor orientation information is provided elsewhere.
- The Irradiance API outputs direct and diffuse electromagnetic irradiance (power per steradian per square centimeter) at the Earth's surface for each source: solar, lunar, and night sky being the prime examples. Ephemeris data from that API is used to ascertain the flux density at the top atmospheric layer; the Conversion API output above then governs the transmittance to the Earth surface.

If running Modtran multiple times for each path is deemed too slow, there are alternative methods of obtaining these quantities which rely upon knowledge of only the ephemeris and a few pre-calculable "effective coefficients", aggregated over molecular species, at each Atmospheric cell. Specifically, Modtran is pre-run for each cell and two coefficients are generated. The first is an effective "path radiance coefficient", the aggregate isotropic thermal radiance per unit volume per unit solid angle. This coefficient may subsequently be used to predict the thermal radiance along multiple paths within the same static atmospheric profile. The second is an effective extinction coefficient, useful for subsequent Beer's Law calculations of transmission between two points. To get the approximate direct and diffuse extraterrestrial radiation upon a surface, the source irradiance at the top atmospheric altitude (given via ephemeris data) can be attenuated similarly.

In this mode that is to be used to optimize efficiency and accuracy for JOSEF, the goal of a quasi-static atmosphere through which multiple paths are calculated is truly realized. The Cell Conversion API then is modified to output extinction and path radiance coefficients at each cell, while the Atmospherics API is modified to input these coefficients instead of layer-and-molecule-specific raw data.

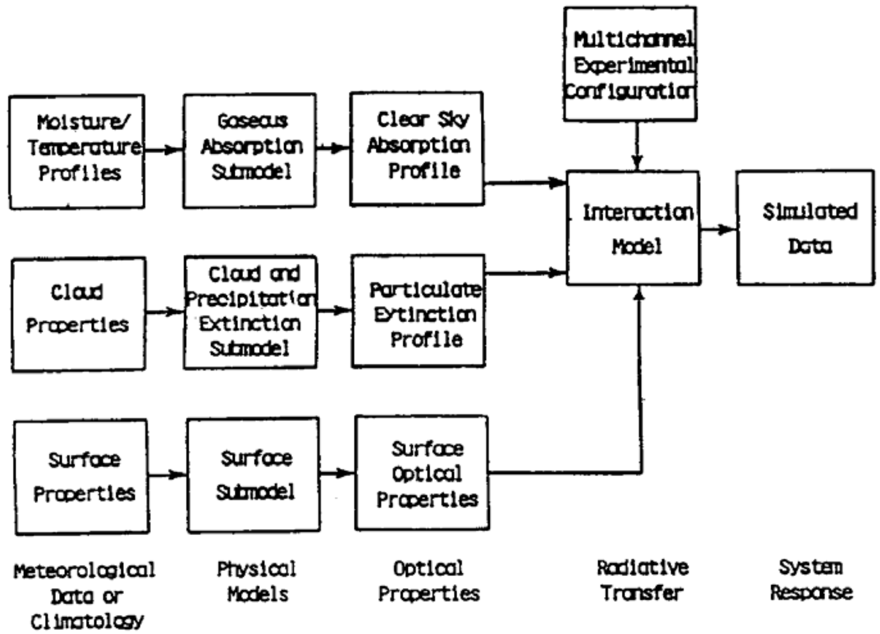
7.3.3.2.2 RADTRAN

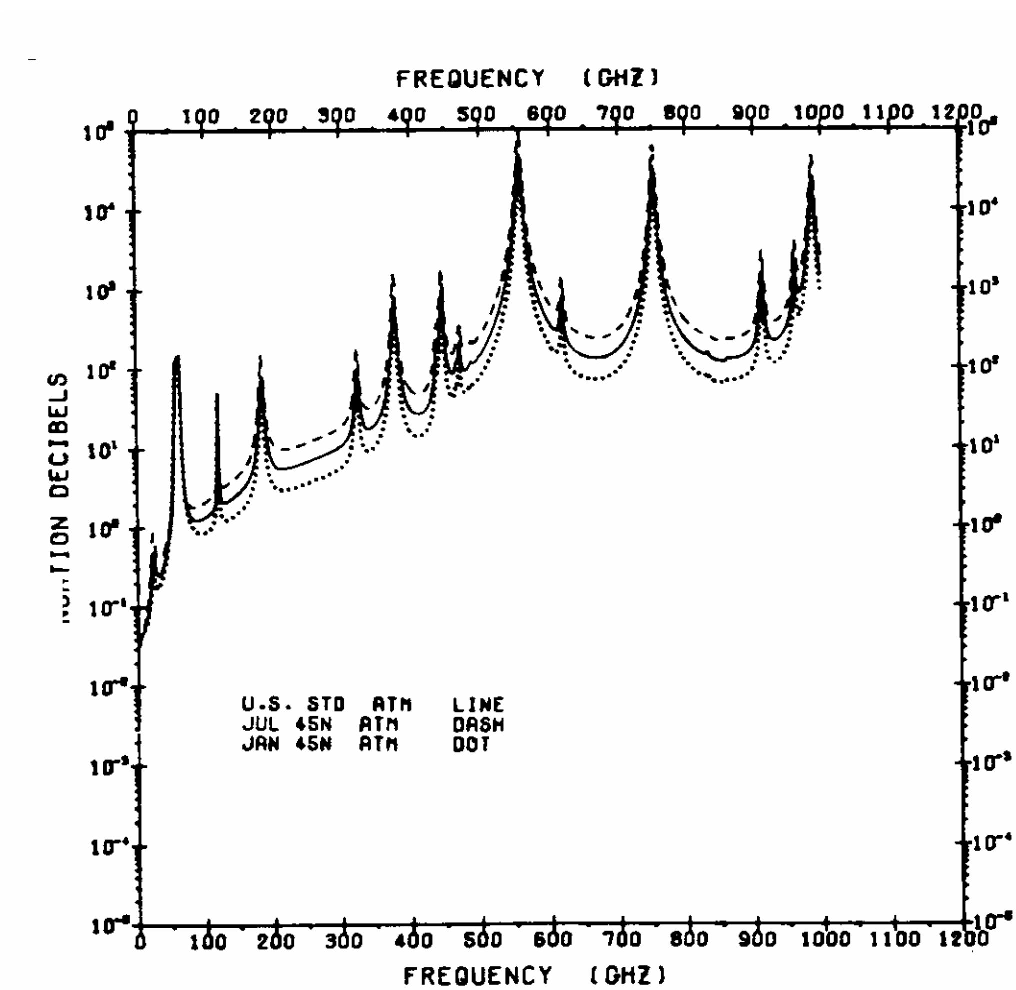
The RADTRAN computer code was developed by the Air Force Geophysics Laboratory (AFGL) to provide atmospheric attenuation and brightness temperature calculations for typical atmospheric paths over the frequency range from 1 to 300 GHz. The calculation is based on evaluation of the radiative transfer equation for thermal emission of microwave frequencies, and has been enhanced to include evaluation of frequency dependent, polarized surface emissivity, precipitation scattering, scalar multiple scattering, and polarization-dependent multiply-scattered brightness temperatures. Absorption of water vapor is computed using the expression of Barret and Chung for frequencies less than 60 GHz. For frequencies between 60 and 300 GHz the absorption is evaluated from the 183 Ghz line plus the non-resonant background, using the model of Gaut and Reifenstein. At higher frequencies, water vapor absorption is modeled using the Van Vleck-Weisskopf line shape, a set of 54 rotational lines and the Gaut-Reifenstein continuum.

Oxygen absorption is evaluated using the parameters of Meeks and Lilley. This continuum model is adequate at moderate relative humidities. The correction for first order coherence effects in overlapping lines is included as an option. Many of these expressions are summarized in Falcone, et. al., *Atmospheric Attenuation of Millimeter and Submillimeter Waves: Models and Computer Code*, AFGL-TR-79-0253 (1979). In calculating atmospheric emission, the upwelling and downwelling radiation is calculated via a recursive relation originally ascribed to E.T. Florance.

The computer speed for clear atmospheres is achieved by fitting empirical data to resonant line absorption and adjusting the non-resonant absorption to account for higher frequency contributions. For atmospheres containing hydrometeors, the attenuation is determined by table interpolation for rain and multiplication for fog/clouds. This procedure eliminates time consuming calculations while retaining the accuracy of the full Mie Theory. Internal to RADTRAN are models of clear atmospheres, clouds and rain which are called by a computer code list similar to LOWTRAN. All models in RADTRAN are typical of mid-latitude atmospheric conditions and all are physically consistent, e.g., cloud model liquid water content is determined by integrating over the model drop-size distribution.

Atmospheric input parameters are height (km), temperature (°K), pressure (mb), relative humidity (%), fog/cloud liquid water content (g/m3), and rain (mm/ hr). Either single frequencies or bands of frequencies (in GHz units) may be considered, for horizontal, vertical or slant paths.


Multichannel Experimental Configuration
Moisture/ Temperature Profiles
Gaseous Absorption Submodel
Clear Sky Absorption Profile
Interaction Model
Simulated Data
Cloud Properties
Cloud and Precipitation Extinction Submodel
Particulate Extinction Profile
Surface Properties
Surface Submodel
Surface Optical Properties
Meteorological Data or Climatology
Physical Models
Optical Properties
Radiative Transfer
System Response


FREQUENCY (GHZ)
DECIBELS
U.S. STD ATM LINE
JUL 45N ATM DASH
JAN 45N ATM DOT

7.3.3.2.3 Weather Model

The Gridded Weather Model represents uniform weather conditions within three-dimensional, rectilinear cells (the same Atmospheric cell structure as served by OASES) placed within the exercise area, as shown in Figure 7-7. This model thus allows the CSE to represent varying weather conditions across the terrain, as well as at various altitudes. The Gridded Weather Model was designed to handle ambient weather conditions within the cell boundaries. By varying environmental conditions from cell to cell, it is possible to create spatially distributed (and time-varying) weather patterns.

The Gridded Weather Model assumes uniform weather conditions within a grid cell and discontinuous changes across cell boundaries. The actual atmospheric parameters are served by the OASES federate. Finally, although the grid resolution can be configured per-exercise, it is fixed for the duration of the exercise.

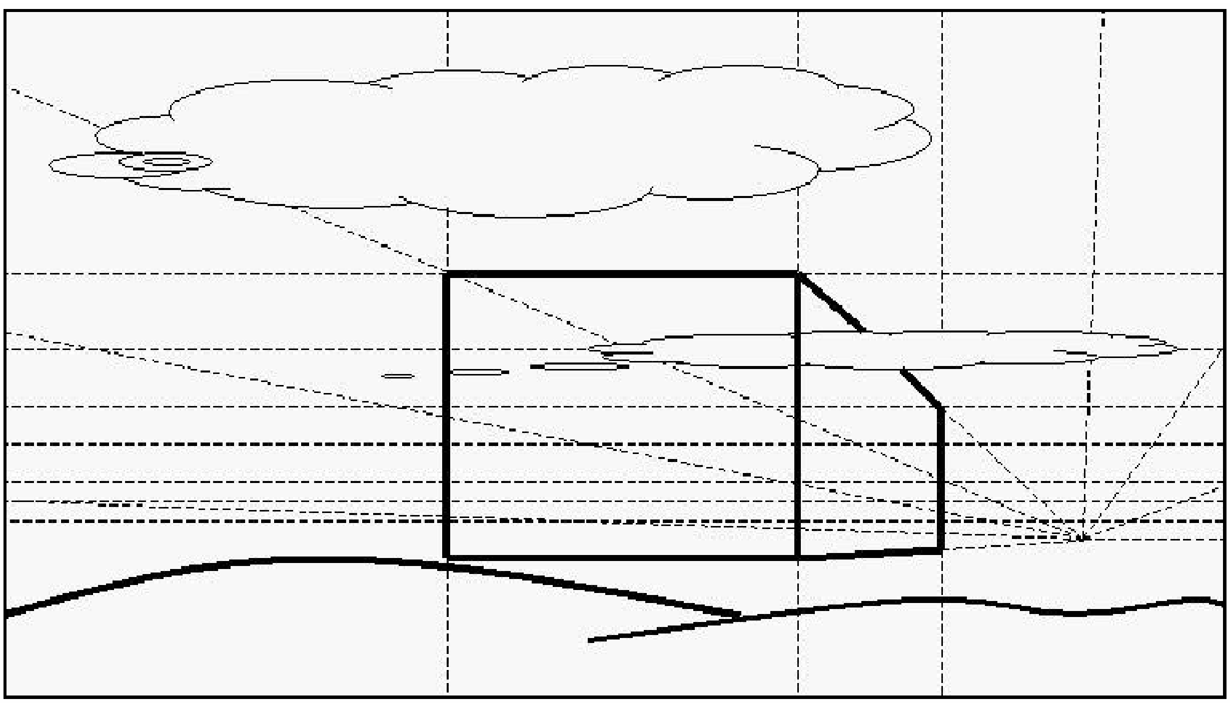


Figure 7-7: Gridded Weather Model supporting full 3-D volumetric values

7.3.3.2.4 Atmospheric Cloud Model

The Atmospheric Cloud Model simulates the natural sky clouds by using coarse descriptions of the clouds to produce continuous, high resolution (approximately 100m grid spacing) clouds that are consistent with the low resolution inputs. The major effect of this model is to simulate the reduction of visibility along any ray passing through the modeled clouds.

The Atmospheric Cloud Model (implemented by *libenvcloud*) uses the same horizontal fractals and fractal time evolution as the Cloud Scene Simulation Model (CSSM), as well as some of the wind effects. The vertical evaporation and convection effects from CSSM are not used in the CSE. Without using a grid of water densities, this model used a function, which executes the fractal function only when needed. This ReScale and Add (RSA) function uses a small grid of random numbers and a few cloud type dependent fractal parameters to produce the density values. The RSA output values are combined with the vertical cloud layer profiles to four-dimensional clouds (i.e. clouds which vary in time and space). Liquid water content value can be produced by this procedure at any point on demand. The key steps in reconstructing a cloud field are shown in Figure 7-8. Liquid water content is used to compute transmissivity along each step of a ray contained within a cloud layer.

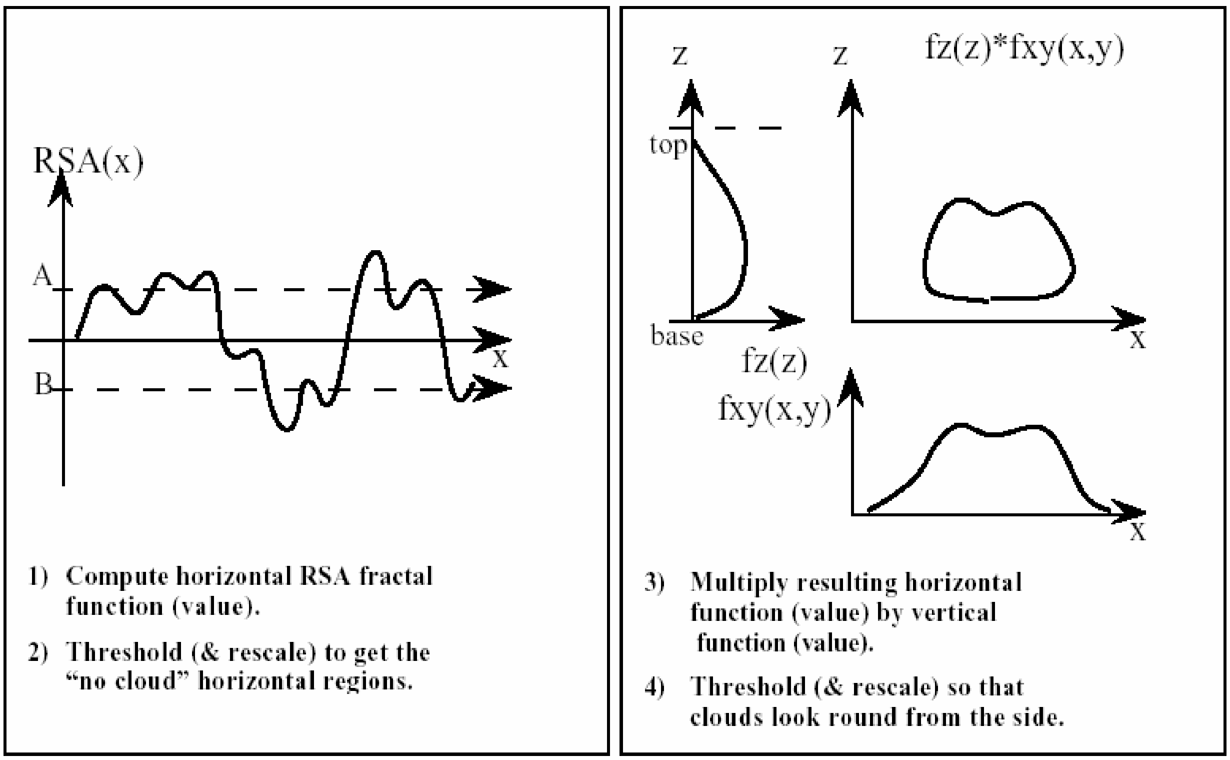


Figure 7-8: CSE Cloud Model algorithm

The atmospheric cloud model deterministically constructs on demand 3-D fractal clouds that vary continuously in time and space given a coarse description of the cloud characteristics, and then calculates the transmissivity of a user specified line-of-sight though the given cloud field.

The inputs to the model are:

- Input location – the location in world coordinates at which to start the ray along which visibility should be measured.
- Target location – the location in world coordinates at which to complete the ray along which visibility should be measured.
- Simulation time – in seconds since 00:00:00 GMT, January 1, 1970.
- Cloud cover – the fraction of the sky is covered in clouds, in the range of 0.0 to 1.0.
- Cloud ceiling – altitude above sea level at which the cloud layer starts (bottom).

- Cloud height – altitude above sea level at which the cloud layer ends (top).
- Cloud type – an enumeration of the kind of cloud formations (e.g., cumulus, cirrus, etc).
- Wind velocity – a world coordinates based vector indicating the speed and direction of the wind.

The first two inputs are explicit. The remaining 6 inputs are environmental queries made internal to the model.

The model has one output:

- Ray visibility – the transmissivity due to atmospheric clouds along the specified line-of-sight, in the range of 0.0 to 1.0.

7.3.3.2.5 Obscurants Model

Munitions, smoke, dust, and muzzle blasts are commonplace on the battlefield and are tactically important phenomena. Generally referred to as obscurants, these airborne particles are represented in simulation using the Combined Obscuration Model for Battlefield Induced Contaminants (COMBIC), which was developed by the U.S. Army Research Laboratory Battlefield Environment Directorate (ARL-BED). COMBIC models the production, transport, diffusion, and shape of battlefield obscurants. The model computes the distribution, density, and transport of airborne obscurants based on airflow, humidity, temperature, and pressure. The COMBIC model uses both semi-empirical data and first-principle physics to determine the movement and distribution of a series of ellipsoidal Gaussian “puffs” and tapered Gaussian “plumes” of smoke and dust. The puffs and plumes are referred to as “Gaussian” because their density, and hence their opacity, falls off by the Gaussian function from their center to their periphery.

The COMBIC model has two phases of execution. In phase one, the full time-history evolution of a given obscurant source under given weather conditions is calculated. In phase two, transmissivity through the obscurant cloud between points in the environment is calculated. COMBIC phase one takes as input two primary types of data, environmental conditions (e.g., temperature, wind velocity) and obscurant source descriptions (e.g., smoke munition type, burning vehicle). Each obscurant source is described as one to five sub-clouds with each sub-cloud being either a Gaussian puff or a Gaussian plume (a three-dimensional tapered geometry). The model outputs a table of subcloud centerline trajectories (downwind distance at time T, centroid height at time T), sub-cloud dimensions (sigma X, Y, and Z at time T), the mean concentration downwind with respect to time, and the extinction (mass produced and mass airborne at time T). COMBIC phase one includes a capability to aggregate a large number of detonations that occur at about the same time into one or two aggregate sub-clouds. This is referred to as a “barrage”. COMBIC phase two allows transmissivity between target - viewer pairings to be calculated at any point in an obscurant clouds history. COMBIC is written in FORTRAN.

COMBIC is used in the CSE to model battlefield obscurants such as howitzer and mortar delivered obscurant smoke, signal smoke, smoke from burning vehicles, and dust kicked up by artillery explosions and mine clearing line charges.

The inputs to the model are:

- Input location – the location in world coordinates at which to start the ray along which visibility should be measured.
- Target location – the location in world coordinates at which to complete the ray along which visibility should be measured.
- Sensor wave band – an enumeration that includes all sensor bands utilized in JSB Experiment Federation.
- Wind velocity – a vector in world coordinates indicating the speed and direction of the wind in meters per second.
- Relative humidity – expressed as a percentage.
- Temperature – temperature in degrees Celsius.
- Barometric pressure – barometric pressure in Newtons per meters squared.

The first three inputs are explicit. The remaining four inputs are environmental queries made internal to the model.

The model has one output:

- Ray visibility – transmissivity through all smoke plumes along the viewing ray, in the range of 0.0 to 1.0. Transmissivity represents the fraction of the incident radiation, in the selected band, that remains at the end of the specified path.

Three separate libraries are involved in the representation of smoke and dust in the CSE. The first library (implemented by libsmokesim) simulates the state of locally generated obscurant clouds and maintains the state (e.g., dead reckons) remotely generated obscurants. The second library (implemented by libsmokerdr) accesses pre-computed COMBIC data and is used by the other two libraries. The data in this library is generated by a large series of runs of the unmodified COMBIC Phase 1 software. COMBIC Phase 1 is run for each wind speed for each obscurant source using default weather conditions (e.g. temperature = 21$_o$C and relative humidity = 50%). Wind speeds of 1, 3, 5, 7, 9, 11, 13, and 17 meters per second are supported in the CSE. Scripts automate the process of creating the data file from COMBIC outputs. The run-time library uses these data files to determine the plume characteristics at time **t**, and interpolates between entries to assure a continuous evolution of cloud geometry.

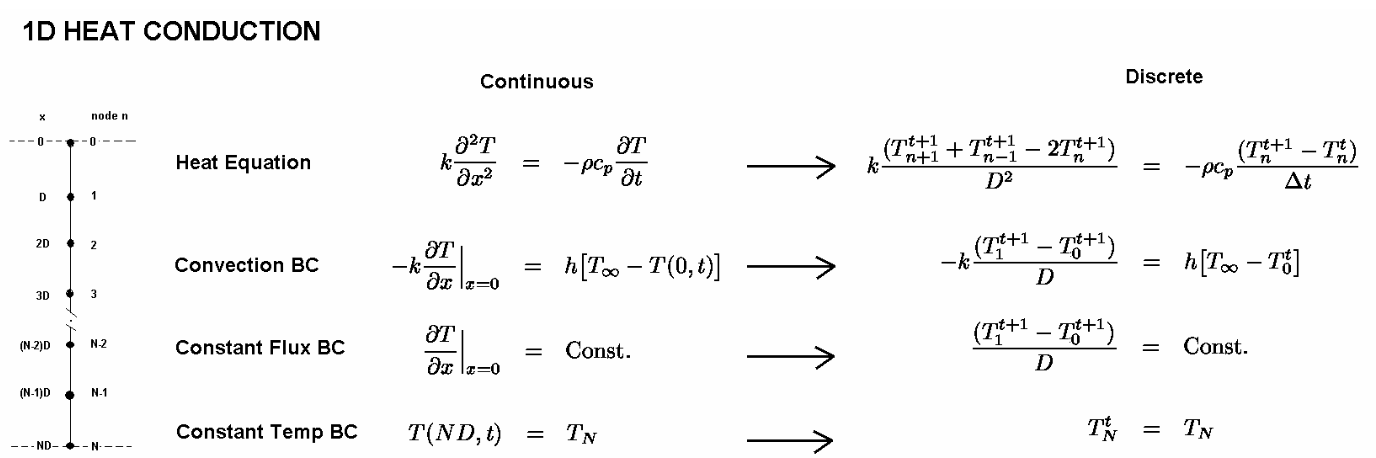
The third library (implemented by libsmkint) calculates the transmissivity through COMBIC generated smoke and dust. This library is the Phase 2 COMBIC software converted from FORTRAN to ANSI C, and adapted to handle real-time data. Transmission through simulated smoke and dust is computed based on the target and observer locations, sensor wavelength and characteristics of obscurant clouds along the line-of-sight. The effects of each obscurant cloud between the target and the observer are computed. To speed this computation, the cloud list is pre-filtered to eliminate clouds outside the field of view. Further, the line of sight is checked against a trapezoidal bounding volume, which encloses each sub-cloud. Only if it intersects is the transmissivity along the line of sight calculated.

In addition to the various forms of high explosive dust and burning vehicle smoke, a number of tactical smoke sources are supported.

7.3.3.2.6 Thermal Model

The Thermal API addresses the passing of appropriate surface geometry dimensions and normals, material system identification, and associated thermo-physical material properties. In addition, the type of boundary condition or dynamic state (including convective heating/cooling, conductive contact, constant heat flux, and insulation) is specified at each surface, along with associated parameters. Both 1D and 3D finite-difference models are supported.

The Spring Experiment will utilize SigSim's 1-dimensional transient thermal model, illustrated below. The model accommodates a variety of boundary conditions (BC) at each surface, some of which are described below. Once these conditions are set up, SigSim uses a Gauss-Seidel iterator to solve the discrete form of the heat diffusion equation for the temperature at each node as a function of time.



The Planck equation is applied to the computed surface temperature, and the result multiplied by the emissivity associated with the surface material, resulting in a prediction for the emitted radiance from the surface. This is added to the radiance reflected from the surface, as described below.

7.3.3.2.7 Reflection Model

The Reflectance API must pass as input the directional and diffuse channel-band irradiances, unit vectors toward the sources (solar, sky, and lunar), the surface normal, and the sensor orientation vector. It must return the channel-band reflected radiance, which is then propagated back to determine the at-the-aperture flux in W/cm2.

For calculation of the reflected radiance due to a particular surface material, the canonical property of interest is the Bi-directional Reflectance Distribution Function, or "BRDF", defined as the differential ratio of reflected electromagnetic flux density per solid angle (radiance) in a given direction to the incoming flux density (irradiance) at a particular incidence angle.

Knowing a particular material's BRDF is valuable for two reasons: not only is one then able to predict the reflected signature given a certain incident light configuration, but the process may be turned around -- as these dependencies are unique to material classes, knowing the BRDF for a variety of materials allows one to predict the identification of unknown materials given their reflected radiance properties, as one might do for large-scale radiometric maps of hostile or inaccessible environments.

As implied above, the BRDF, in general, depends on both incident and reflected angles, with respect to the surface normal. Integration over one or more of these angles is typically required for comparison with optical measurement data, as for most experimental setups it is the total reflection into a hemisphere (the DHR, or Directional-Hemispherical Reflectance) which is measured. In addition, BRDF's also typically depend upon the wavelength of light being reflected, which provides another useful dependency for unique identification.

A number of empirical and semi-empirical BRDF models have been introduced throughout the years (see the table below). They range from the common Phong Model (a variant of which is used to drive the OpenGL lighting environment), to complex, physics based constructions such as the Hapke model. Some even account for polarization dependence, as in the NEFDS design.

For the JSB Experiment 1, the CSE has adopted the SigSim Modified Phong model – the decision being motivated by three main factors. First, through Surface Optics Corporation, we have access to quality, spectral DHR data at given incidence angles. Second, as SigSim is often used in conjunction with OpenGL to drive the rendering process, the chosen BRDF, when properly integrated, should be of such a form as to reproduce the canonical OpenGL relation between incoming and outgoing radiance:

Given a general light source with diffuse and specular components, i.e. of the form:

$$\mathbf{L_i = Edif/\pi + Espec\ \delta(\ \Omega - \Omega_i\ )\ ,}$$

The Phong model (upon which OpenGL is based) predicts an outgoing reflectance of the form

$$\mathbf{Lr = K_{amb}\ L_{dif} + L_{spec}\ [Kdif\ \cos\ (\ \theta i\ ) + Kspec\ \cos\hat{}_n\ (\ \Phi\ )]\ ,}$$

where **Φ** is the angle between the mirror direction and the viewing direction**,** and **n** is a “shininess” parameter related to the specular lobe width.

Third, after a review of both the data actually currently available for synthetic imaging, and a knowledge of the kind of information most users would have on hand, it became clear that there was the need for a more intuitive set of basis variables. These variables are the DHR, the percent of this which is specular, the angle at which the measurement was made, and the width of the specular lobe. From these, the parameters $\mathbf{K_{amb}}$, $\mathbf{K_{dif}}$ and $\mathbf{K_{spec}}$ in the SigSim Modified Phong Model are derived.

In many cases, the set of variables used for any particular BRDF are derivable from those used in other models, and work is being done to calculate as many of these relations as possible, which would allow one BRDF engine to accept input from a wide variety of models. Also in research is the use of specialized models to accommodate such things as large-scale vegetative canopies, bodies of water, and turbulent air masses.

For purposes of generality, the Reflectance API should be constructed to include an index specifying the particular BRDF model in use, followed by at least seven (7) wavelength-dependent generalized parameters in double precision (i.e. *parm1, parm2, parm3..*), some of which may not be used in a particular model. A numbering scheme should be assigned to the parameters of each of the following models so that it is understood implicitly which parameter resides in which API location.

SigSim Modified Phong Model

Equation

$$\mathbf{P(\theta_i,\phi_i,\theta_r,\phi_r) = K_{dif} / \pi + K_{spec} \cos^n(\Phi) / (\pi \cos(\theta_i))}$$

Model Input Parameters	Definition
$\mathbf{K_{dif}}$	Diffuse reflectivity constant
$\mathbf{K}$spec	Specular reflectivity constant
$\mathbf{\Phi}$	Angle between mirror direction and view direction
$\mathbf{\theta_i}$	Angle between light source and surface normal
n	"Shininess" parameter (related to width of spectral lobe)

NEF (Beard-Maxwell) Model

Equation

$$\mathbf{P'_{U,T}(\theta_i,\phi_i,\theta_r,\phi_r) = [R(\beta;n,k)] * [\{P'_{FS}(\theta_N)\cos^2(\theta_N)\} / \{ R(\beta=0;n,k) * \cos(\theta_i)\cos(\theta_r)\}] * [SO(\beta,\theta_N;\tau,\Omega)] + P'_D + [\{2 * P'_V\} / \{\cos(\theta_i)\cos(\theta_r)\}]}$$

Model Input Parameters	Definition
n-ik	Complex index of refraction
P'_D	Diffuse BRDF Parameter (sr^{-1})
P'_V	Volumetric BRDF Scattering Parameter (sr^{-1})

Ω	Shadowing and Obscuration parameter
τ	Shadowing and Obscuration parameter
P'_{FS}	Measurement-based first surface BRDF curve (sr^{-1})
θ_N	Zenith angle of the scattering element relative to the surface normal of the material

Hapke Model

Equation

$$\mathbf{P(\mathit{i,e,g}) = [\mathit{w}/4\pi] * [1/(\mu_0 + \mu)] * [\{\mathit{p}(g)\} + \{\mathit{p}(g)\mathit{B}(g)\} + \{\mathit{H}(\mu_0)\mathit{H}(\mu)\} - 1]}$$

Model Input Parameters	Definition
g	Phase Angle
w	Volume Scattering Albedo
p	Volume Angular Scattering Function
$p(g)$	$p(g) = [\Sigma_j N_j \sigma_j Q_{sj} p_j(g)] / [\Sigma_j N_j \sigma_j Q_{sj}]$
B_0	Amplitude of Opposition Effect
H	Angular-Width Parameter of Opposition Effect

Sandford-Roberston Model

Equation

$$\mathbf{P_{bd}(\theta_r,\phi_r,\theta_i,\phi_i) = [1/\pi] * [\{P_d g(\theta_i) g(\theta_r)\} / \{G(b)^2\}\} + \{.25\} * [\{1-(P_d+\varepsilon)g(\theta_i)\} / \{G(b)\}] * [\{h(\alpha)\} / \{\mathit{H}(\theta_i)\cos\theta_r\}]}$$

Model Input Parameters	Definition
ε	Hemispherical emittance
P_d	Directional-Hemispherical diffuse reflectance
b	Geometric parameter governing reflectance at grazing angles
e	Geometric parameter governing the width of the specular lobe

7.4 CSE UTILITIES

The following sections provide an overview of the two major CSE utilities – its interface to the HLA, and terrain database utilities.

7.4.1 Agile FOM Interface (AFI)

The goal of the AFI is to allow the models within the CSE to be written to use the objects and interactions defined in the CSE SOM independent of the FOM being used for the current federation execution. The AFI will use FOM mapping definitions provided in a data file to map the internal CSE SOM representation to the external FOM representation in a manner that is invisible to the rest of the CSE.

The AFI is part of the RTI Interface Layer (RIL). The RIL is a software layer that connects the application libraries with the RTI. The RIL provides a set of abstractions designed to simplify coding for the RTI, to centralize RTI implementation and policy decisions and to provide FOM agility. The RIL is a set of C++ classes that have been architected to provide a convenient and powerful layer between the application and the RTI. The goal is to provide:

1. FOM agility.
2. Convenient abstractions of RTI concepts.
3. Automatic RTI processing when appropriate.
4. Centralized RTI policy decisions.

The RIL does not attempt to hide the fact that the CSE is using the HLA RTI. The object/attribute and interaction/parameter paradigm is still used, however all calls to the RTI are made through RIL classes.

Although applications utilizing the AFI have traditionally been real-time Human-In-The-Loop (HITL) systems, most of the upcoming uses of the CSE will be in SBA or engineering predictive domains. To better allow for the strictly reproducible results and faster-than-real-time execution such users will desire, compatibility with the HLA time management services is being added to the AFI. Time Management services enable a federation to ensure that the simulation time clocks (which may be completely divorced from the wall-clock time of the real world) remain in sync with each other as necessary, while still taking advantage of distributed computing resources.

Since the CSE is more of a “middleman” federate, computing answers to queries about models running on other federates, rather than executing its own entities, it has no need to advance simulation state on its own. Therefore, it will not regulate the time advance of other federates- it will simply constrain itself to the simulation time that the rest of the system has decided to achieve. All of its outgoing data is in the form of interactions, published in direct response to changes in sensor Volume of Interest (VOI). Since the responses generally represent instantaneous propagation of electromagnetic energy, the messages will be sent as quickly as possible. That is, the outgoing time stamps on interactions will always be set to the current simulation time plus the minimum federation lookahead value. The lookahead value must be carefully selected to balance the tradeoffs between simulation performance (primarily the maximum ratio of simulation time to real time that can be achieved) and repeatability / validity of results.

7.4.2 Terrain Model Utilities

The Compact Terrain Database library, libctdb, is used within the CSE to access elevation, soil type, and feature data of the terrain database. Major libctdb functions include:

- Reading the database into memory or cache
- Maintaining useful information about the database, such as its size, minimum and maximum elevation, and UTM zone, northing and easting or GCS cell id (its location on the planet)
- Point elevation lookup
- Elevation lookup along a line segment (find high ground, find terrain profile, etc.)
- Soil type lookup
- Vehicle placement (rotation matrix generation)
- Intervisibility calculation (including terrain and vehicle blockage)
- IR and Radar clutter calculation
- Generating graphic representation of the terrain such as contour maps and hypsometric maps, in real time

The following are some of the major libctdb functions that will be used:

- ctdb_apparent_size finds the apparent size of the perpendicular parallelepiped described by length, width, height, given the eye point x0,y0,z0, the target location x1,y1,z1 and target rotation. The apparent width and height of the object are returned, as well as a location corrected to be at the bottom center of the apparent location.
- ctdb_lookup_attributes fills out an array of attribute values, one for each attribute specified by attr_list, whose length is num_attributes. ret_list[i] is the value for attribute attr_list[i], and it's interpretation is attribute-specific.
- ctdb_lookup_attr_code converts the three character Feature and Attribute Code to a 16 bit integer value. The digits 0 through 9 are assigned their actual value and the upper case characters A through Z are assigned the value 10 through 36. The FACC is then converted as if it were a base 36 number.
- ctdb_lookup_attr_name converts the 16 bit integer facc value to the three character FACC name.
- ctdb_find_ground_intersection locates the point at which the ray from <x0 y0 z0> to <x1 y1 z1> crosses through the terrain or a specified feature type (qual specifies exactly what should be taken into account). The point of intersection (+/- 1 meter along the ray when testing ground polygons) is returned in pt_ret.

- ctdb_next_linear_ref returns the next linear reference feature in the search space. The return value indicates how many vertices are in the feature. A return value of zero indicates that there are no more linear features. In addition to the X,Y locations of each vertex, the absolute elevations of the tops of the features which correspond to each vertex location are returned. Also, and a description of the feature is returned. NULL pointers may be passed for any undesired return values.
- ctdb_next_volume returns the next volume in the search space. The return value indicates how many vertices are in the volume. A return value of zero indicates that there are no more volumes. The returned volume model name will be used to access the SEE-IT generated file to obtain surface areas and attribution of the model faces.
- ctdb_point_to_point performs an intervisibility check starting at the point x0,y0 and proceeding to the point x1,y1. z0 is the eye point of the viewer and zl and zh are the bottom and top of the target. width is the width of the target, and is used when comparing against individual trees and buildings, otherwise a zero-width target is assumed. All these values are in meters. The visible target area is returned as a floating point number in the range 0.0 to 1.0 (0.0 for complete blockage, 1.0 for complete visibility). Since visibility can only get smaller as more features are tested, knowing the minimum visibility interesting to the application can greatly enhance the speed of calculation. Even very small values (such as 0.05) can greatly increase speed. If the visibility measure drops below this visibility, 0.0 will be returned. The portion of the ray from x0,y0 to x1,y1 within each database crossed is clipped to the boundaries of that database.

In addition to these existing functions, additional functions are being developed to support the interrogation and retrieval of volume feature data from the SEE-IT produced flat file described in Section 7.4.1.1.1.

7.5 INTERFACE DESIGN

The following tables provide the inputs and outputs of the major internal CSE APIs that were discussed in the previous sections.

Atmospheric Model APIs	
Inputs	***Outputs***
Atmospheric Cell Boundaries (lat1, long1, z1) (lat2, long2, z2)	Cell 0.40-25.0 um spectral radiances
Atmospheric Cell Atmosphere-Weather parameters	just-in-time Cell RF-band precipitation clutter cross-sections
Atmospheric Cell 0.40 - 14 um spectral extinction coefficients	Cell RF-band index of refraction
Atmospheric Cell RF-band extinction coefficients	just-in-time Cell Modtran Card 2A-2C Model Atmospheres
	just-in-time Cell 0.40-25.0 um spectral extinction coefficients
	just-in-time Cell RF-band precipitation extinction coefficients
	EO/IR-band LOS path transmittance
	EO/IR-band LOS path radiance

	RF-band 2-way, refracted path attenuation

Terrain API	
Inputs	***Outputs***
Line of Sight (LOS) Request	Solar, Lunar, Sky Irradiances on background materials
Material System SMC, MCC, MCU, etc. (as function of LOS request)	Background Material System Surface Boundary conditions
Material State (burning, hot, explosion, etc)	Tsurface of Background Material System
Material System lat, long (as function of LOS request)	Background RF sigma-zero or gamma (for RCS)
Surface Normal (as function of LOS request)	

Target and Clutter Vehicle API	
Inputs	***Outputs***
Platform/Target List and IDs	Target RCS as a function of Target Aspect Angle and lamda
Target Material Systems	Target Material System Surface Boundary conditions
Target Radar Cross-Section (elevation, azimuth, lamda, horizontal and vertical polarization)	Solar, Lunar, Sky Irradiances on target ID materials
Platform/Target State (gun barrel fired, target moving, engine idle, etc)	Target Material System Surface Boundary conditions
Platform/Target Position (lat, long, z)	Tsurface of Target Material Systems
Platform/Target Orientation (I, j, k)	
Platform/Target Velocity (vi, vj, vk)	

Sensor LOS Query API	
Inputs	*Outputs*
Sensor Orientation relative to Platform (i,j,k)	LOS passband transmittance and path radiance
	Effective RF LOS for Terrain intersections
	Sensor Platform - Target Rdot (velocity radial)

Ephemeris API	
Inputs	***Outputs***
Location (GCS corrdinates)	Sun azimuth and elevation
Julian date	Moon azimuth and elevation
Time	

Reflectance API	
Inputs	*Outputs*
Directional and diffuse channel-band irradiances	Channel-band reflected radiance
Unit vectors toward the sources (solar, sky, lunar)	At-the-aperture flux (W/cm2)
Sensor orientation	
Obscurants API	
Inputs	***Outputs***
LOS path	Transmissivity through all smoke plumes along the viewing ray, in the range of 0.0 to 1.0
Sensor wave band	
Wind velocity *(model internal query)*	
Relative humidity (%) *(model internal query)*	
Temperature (degrees C) *(model internal query)*	
Barometric pressure (Newtons per meter squared) *(model internal query)*	
Atmospheric Cloud API	
Inputs	*Outputs*
LOS path	Ray visibility – the transmissivity due to atmospheric clouds along the specified line-of-sight, in the range of 0.0 to 1.0.
Simulation time – in seconds since 00:00:00 GMT, January 1, 1970. *(model internal query)*	
Cloud cover – the fraction of the sky is covered in clouds, in the range of 0.0 to 1.0. *(model internal query)*	
Cloud ceiling – altitude above sea level at which the cloud layer starts (bottom). *(model internal query)*	
Cloud height – altitude above sea level at which the cloud layer ends (top). *(model internal query)*	
Cloud type – an enumeration of the kind of cloud formations (e.g., cumulus, cirrus, etc). *(model internal query)*	
Wind velocity – a world coordinates based vector indicating the speed and direction of the wind. *(model internal query)*	

7.6 CSE DESIGN

It may be confusing that the JSB Experiment Federation would contain two federates that both provide data on the natural environment. Both OASES (Ocean Atmosphere Space Environment Service) and CSE (Common Synthetic Environment) can be considered "environment servers"- for a certain definition of environment. OASES uses "environment" to mean weather. It is the weather federate for the JSB prototype, and provides the definitive background atmospheric state for the experiment. "Environment" as used by the CSE has a broader meaning of "everything outside the sensor". That includes the earth's surface, structures, vehicles, people, weapons, vegetation, clouds of particles, and the stars above, as well as the weather that OASES provides. CSE will subscribe to many different providers of these effects, and combine them all into unified inputs to drive sensor models.

Sensor federates (like ACS and PRISM) will not communicate with OASES directly- they only listen to CSE. However, some of the CSE messages will contain values derived from OASES data.

There are two reasons why it is desirable to separate the sensor models from direct communication with OASES:

- Logically, the atmosphere is something outside of the sensor, so its simulation should not be controlled by the sensor model code. JSB will try to increase model uniformity by moving the calculations to interpret the weather data into one place (the CSE).
- In future uses of the JSB, the atmospheric and weather data that will be served by OASES will undoubtedly be expanded. By only having the CSE subscribe to OASES data, the modifications required within the entire JSB is limited to a single application.

7.6.1 End-to-End Example

The following subsections provide an end-to-end example of how environmental data would be prepared for, consumed by, and utilized within the JSB Experiment Federation. Included in this example is how the CSE will utilize environmental state data published by OASES, how it will augment target state data published the Platform federate (JSAF), how it will calculate environmental effects information required by the Sensor federates (PRISM, ACS, and Prophet), and the how it will interact with a Sensor federate.

7.6.2 Pre-Experiment Configuration

The experiment designer selects geographical and temporal extents and provides them to terrain designer and weather designer. The terrain designer takes a SEDRIS Transmittal, containing the appropriate data to affect sensors (primarily the material systems for the surface and base of the terrain), and compiles it into a ctdb file (the run-time terrain format for JSAF, the CSE, and Prophet), maintaining the same attributes attached to the terrain polygons.

The weather designer uses the scenario extents to select appropriate portions of weather logs collected by JCIET personnel, and uses them as the inputs to a meteorological model (COAMPS) that outputs a grid in the OASES disk format. The grid resolution will be 3 by 3 kilometers, and a different grid will be available for every 20-minute period. (With a maximum scenario extent of about 175 km, the grid may be about 60x60).

The experiment designer receives the ctdb file and loads it into JSAF, the CSE, and Prophet, then proceed to lay out vehicles and routes (in JSAF) for the experimental scenario, based on data logged during JCIET. (Since not all vehicles had their positions accurately recorded, there will be some guesswork initially). They also must configure the sensor assets and the TST Cell behavior.

7.6.3 Federation Execution

The CSE federate subscribes to Atmosphere, Platform, and Warfighting events (including the operation of sensor controls).

OASES reads disk file and publishes the initial exercise atmospheric grid and state data (subsequent updates will be published at twenty minute intervals). The Atmosphere Objects published by OASES are read in by the AFI and used to instance the CSE's EDB. Information in these Objects dealing with weather are used by the Gridded Weather model, and cloud meteorology is input into the CSSM-lite model, to produce 3D cloud density profile. Attenuation attributes within the Atmosphere Objects are provided to the SigSim Model Library for consumption its Atmospheric models to calculate direct and diffuse source radiance, properly attenuated.

Military platforms and Clutter vehicles will be instanced in their respective federates, begin their movement along planned routes, and publish platform state (vehicle type, position, angle, velocity, etc.). The CSE will read in this platform state data and maintain platform state for future use.

Sensor federates publishing the Volume of Interest (VOI), or a change in the VOI, for their sensor will initiate calculations in the CSE[1]. The general flow of calculations is:

- Calculate refracted line of sight to determine the sensor FOV using indices of refraction calculated by the SigSim Atmospheric models
- Query ctdb to obtain cultural features within the FOV to include the surface area of the visible faces and material systems of those faces
- Query platform state data to obtain targets/clutter vehicles within the FOV to include BRDF (for EO/IR sensors) or radar cross-section (for RF sensors) based upon vehicle position/orientation with respect to the sensor FOV

[1] The sensor federates will also publish other information about the sensor that the CSE will use, including the sensor location and orientation, and some specific sensor characteristics such as passband (IR sensors) and wavelength (RF sensors).

- Perform n Line-Of-Sight queries to ctdb to obtain background terrain material system attributes
- Apply BRDF and/or radar cross-section to the relevant known irradiance to provide a reflected radiance for each of the above.
- For IR sensors, given thermal boundary conditions for each of the above (platforms, cultural features, terrain), apply the SigSim thermal model to determine surface temperature, and thus the thermal emitted radiance at each surface
- For IR sensors, add thermal emitted radiance to reflected radiance
- Calculate attenuation, relative to the return path, using the previously calculated extinction coefficients
- For IR sensors, add path radiance along the path to provide the “at the aperture” flux

The results of these calculations are combined into a Propagated.Sensing interaction and sent to the sensor federate. Contents of these interactions for the sensor types will include:

- IR Sensor:
    - Target/Background passband exitances & reflected radiances, area-weighted totals (with weights)
    - Attenuated, “at-aperture” target delta T and background T (derived from the above)
    - Position/orientation with respect to the sensor
- RF Sensor:
    - Target ID’s within area of interest
    - Via Radtran, X & K –band 2-way refracted path attenuation (dB)
    - Projection of target velocity vector on path direction (“r-dot”)
    - Apparent sensor view direction vector
    - Target RF cross section from X-patch, modeled, or empirical data as function of wavelength, azimuth, and polarization
    - Target cross-range and down-range dimensions (projected area), grazing angle
    - Average background γ, mean δc as function of grazing angle and material type
    - Precipitation cell RCS, backscatter and range extent

In response to the Propagated.Sensing interaction, the sensor federate will execute its own code, and if the target is detectable, send a message to JSAF and/or the TST-Cell federate to create or update tracked target data. Is this accurate? Isn’t a detect message or more precisely a “detect” event actually generated within the core federates representing the sensor operators?

This page intentionally left blank

8. APPENDIX B. HUMAN AND ORGANIZATIONAL BEHAVIOR MODELING

As was stated in Section 4.1.6, there are several human operators and an organization present in the scenario vignette to be modeled in JOSEF. They are:

- UAV Operators
- F-15E Operators
- JSTARS Operators
- Time Sensitive Targeting Cell staff

These are modeled using the CORE Conceptual Graph editor and the CORE graph processor engine. In terms of rapid behavior modeling, the CG notation of concepts (rectangles), relationships (ovals), and actors (diamonds) provides us with a visual programming capability that can not only express rules and facts, like traditional expert system shells such as CLIPS, but also compiled libraries of behavioral code embedded in actors. This practical and efficient mechanism for building hybrid knowledge and behavior agents is illustrated below in detail for each of the four agent types listed above.

8.1 UAV OPERATORS

8.1.1 Maneuvering

The on-station location and pattern consists of two minute legs and a standard turn at each end – a thirty degree bank turn, all done at standard loitering speeds (approximately 70 knots). The wind speed and direction will determine the power setting required to hold a constant speed. The focal point for pattern exists inside the Camden Ridge/Pine Hills MOA.

The behavior governing the transit to and from the on-station location to perform tasks, and re-tasking behaviors is as follows. When the UAV operator gets a coordinate, the operator will first point the camera at the coordinates. The UAV will then be commanded to fly at the coordinate. The UAV will fly at target, and if no detection is made, fly outbound for 2 minutes and then do a 60 degree bank turn and head back toward the coordinate. While the maneuver is being performed, the operator moves the camera in a search pattern.

When given coordinates to search, the following pattern is flown while searching. The sensor is cued to the target coordinates and the UAV is flown towards the coordinates. The coordinate refinement operation – flying right at something is performed until an overflight has occurred.

8.1.2 Controlling the Sensor

When a search area has been determined, the sensor is controlled in the following manner while transiting to the search area. The sensor is cued, and the operator will wait until he has some resolution in the target area. The operator will not search the target area until he is within 10 miles. The figure below illustrates this.

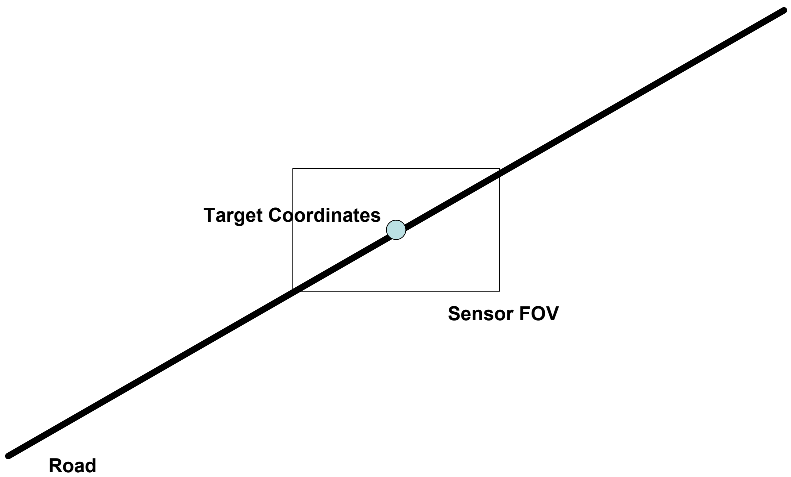


Figure 8-1. Sensor cued to target coordinates.

Roads are searched first if present and near the target coordinates. If the target is not in the Sensor field-of-view (FOV), scan right to left to two times the sensor FOV to each side, as indicated in the figure below.

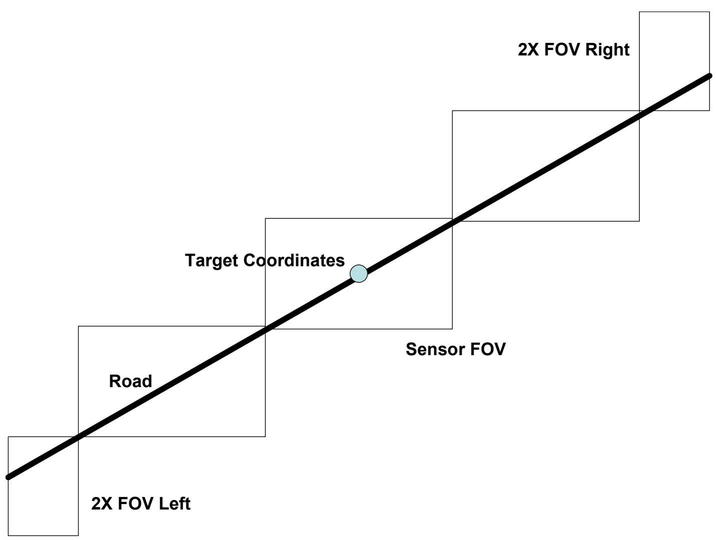


Figure 8-2. Search pattern along a road.

If the target is not found, hit recue button which send the FOV center back to the original target coordinates. When the sensor is covering an area, a raster search is performed. A raster search is an outward expanding rectangular pattern, starting from the target coordinates expanding by 0.5 FOV each time. The figure below shows the raster search pattern.

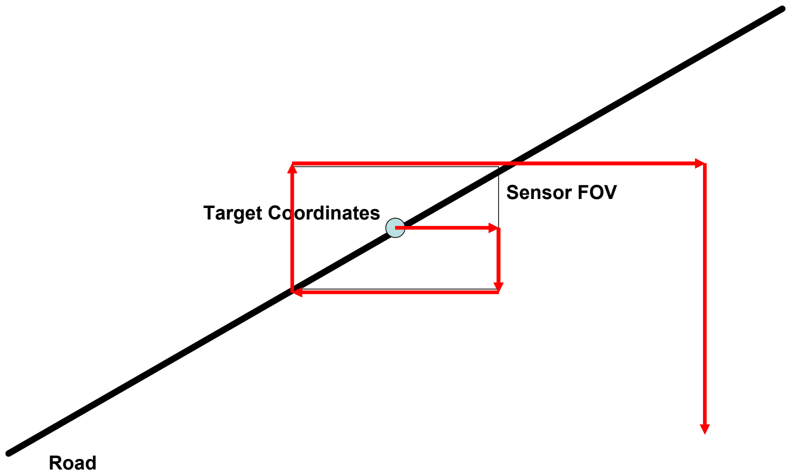


Figure 8-3. Raster search pattern.

8.1.3 Conceptual Graphs

The behavior of the UAV Operators is implemented using conceptual graphs (CGs) built with the CORE CG Editor and executed by the CORE graph processor engine, the heart of the agents. Concepts are represented by rectangles, relationships by ovals, and actors by diamonds. The actors can be as simple as a math operation or as complex as a neural network of existing expert system implementation. In this way we see the CG methodology as being an integrative architecture for artificial intelligence systems or behavior models.

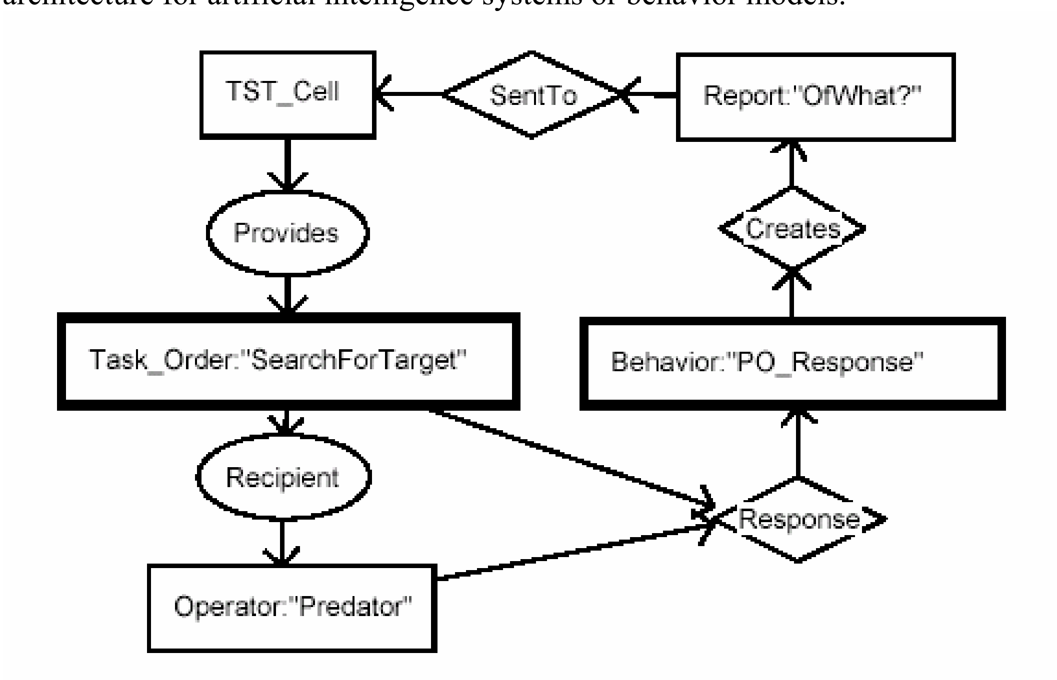


Figure 8-4. Top-level CG for UAV Operator.

In the figure above, the TST Cell concept represents an encapsulation of the "ghost" representation of the TST Cell and hides the HLA calls made to get information from the TST Cell federate. The UAV receives a task order, and the recipient is the UAV Operator.

Synchronization is enforced before the Operator responds using an actor. When both inputs to the response actor are TRUE, namely a task order has arrived and the Operator is ready to receive it, the Response concept is then evaluated. Note that the Response concept appears in bold. This indicates that a concept contains a CG itself. When the Response CG has been executed, a report is constructed and sent to the TST Cell.

We next examine the internals of the Response CG.

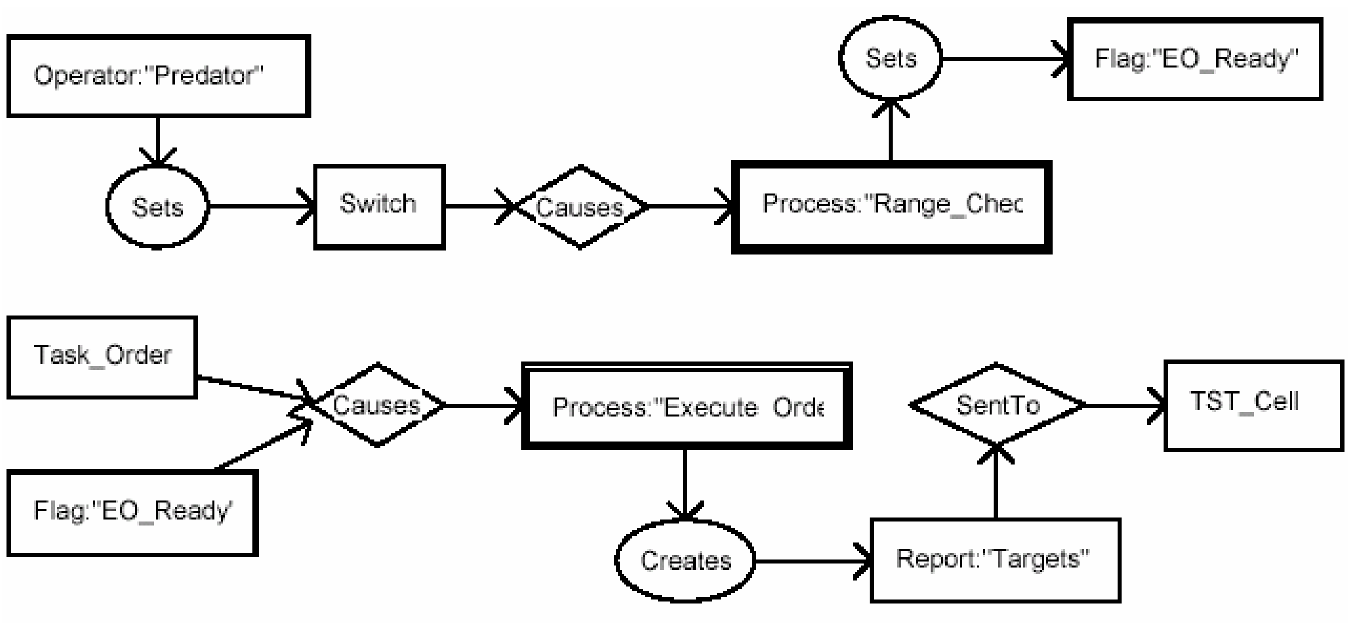


Figure 8-5. Response CG implementation.

The Response CG consists of two CGs, both of which must execute to be true for the Response CG to execute as described above. The top CG controls the behavior of the UAV with regard to maneuvering the UAV so that its EO Sensor can be employed. Note that for the lower CG to execute, the upper CG must have successfully executed so that the "EO_Ready" flag gets set. The Execute Order concept is also a complex implementation that controls the sensor to implement the behavior described above.

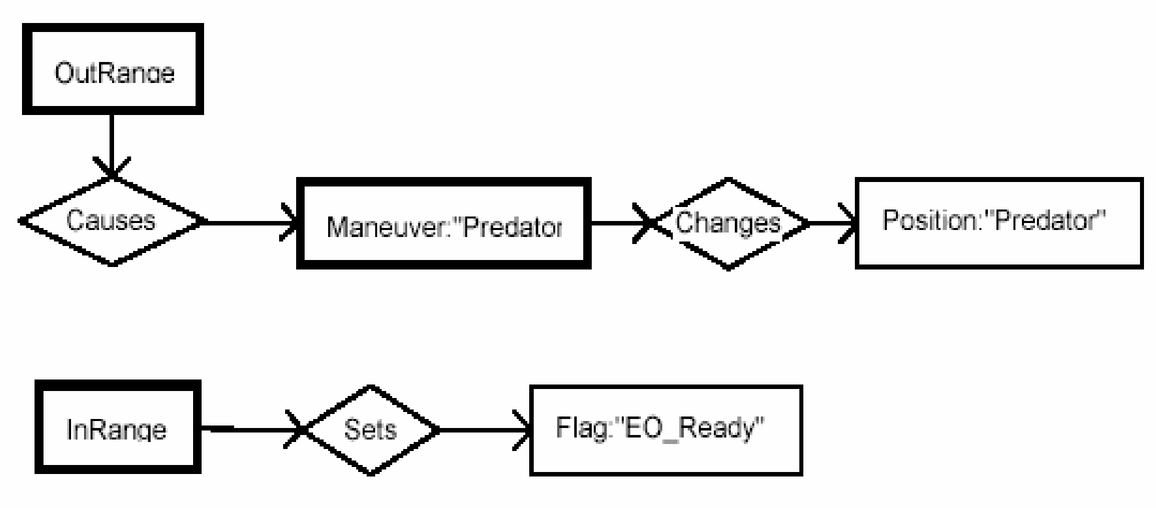


Figure 8-6. Range Check CG.

The Range Check CG is shown in Figure 8-6 above, and is in two parts. The upper graph segment controls and encapsulates the maneuvering of the UAV, and once this is completed, and the UAV is in-range for the sensor to be employed, the EO_Ready flag is set.

We next discuss the Maneuver concept shown in the upper CG.

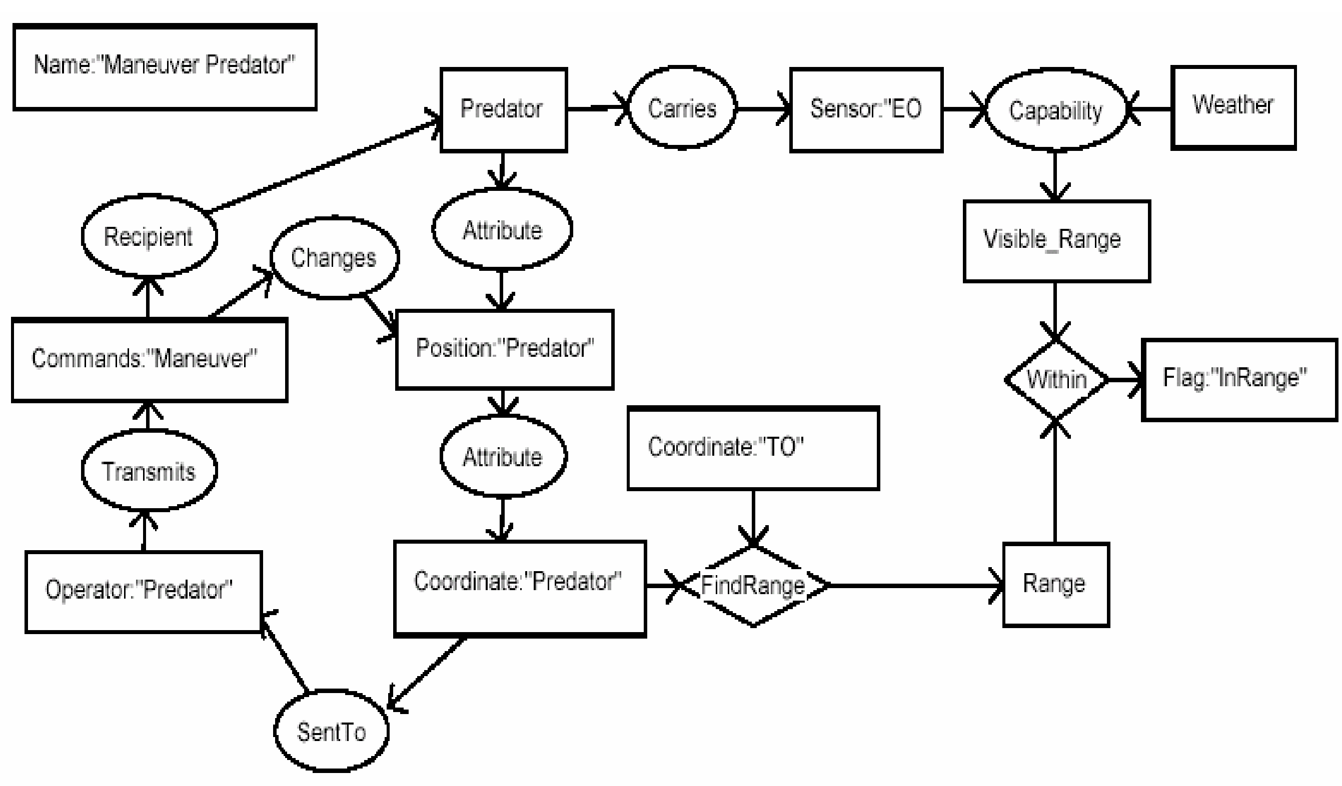


Figure 8-7. Manuever CG.

The Maneuver CG encapsulated the calls made to JSAF to control the platform model. The key computations are the actors that do the simple math of finding the range and determining if the range is within the limits of the EO Sensor. Once the UAV is maneuvered within range, and Execute Order concept from Figure 8-5 must be implemented. The key features of this CG is the implementation of the Scan Area CG, the encapsulation of the Acquire code in an actor to get an initial target ID (or detection), and the Zoom CG. This is illustrated below.

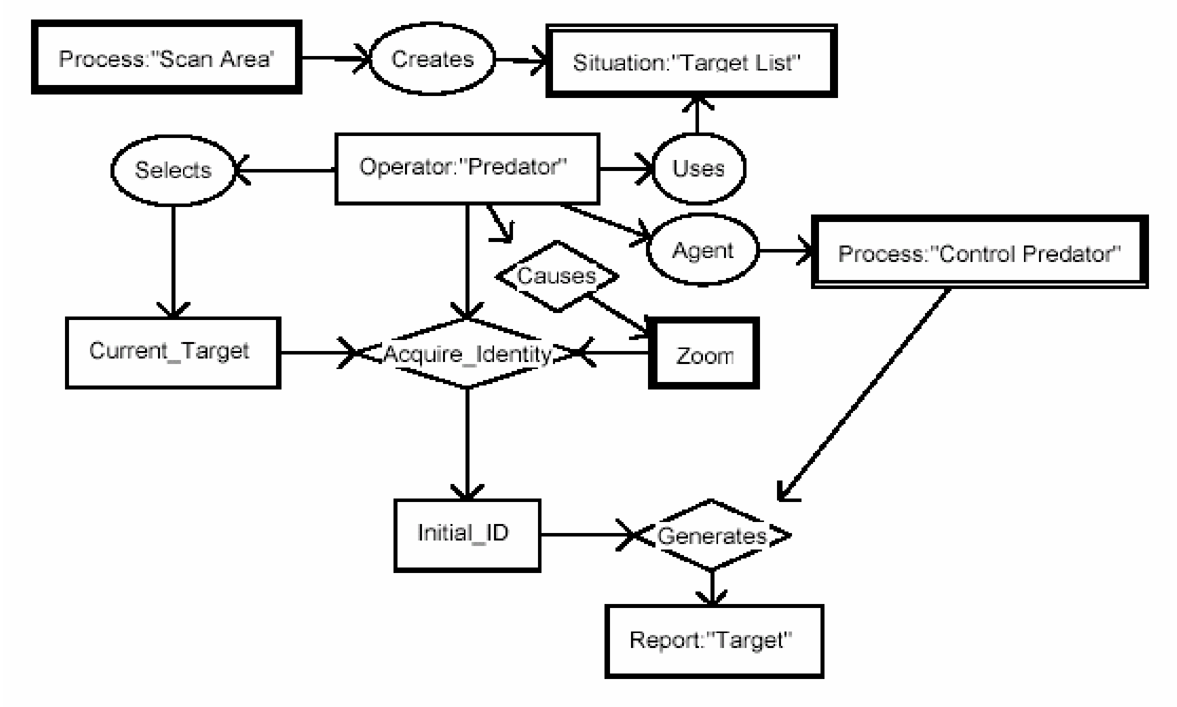


Figure 8-8. Execute Order CG.

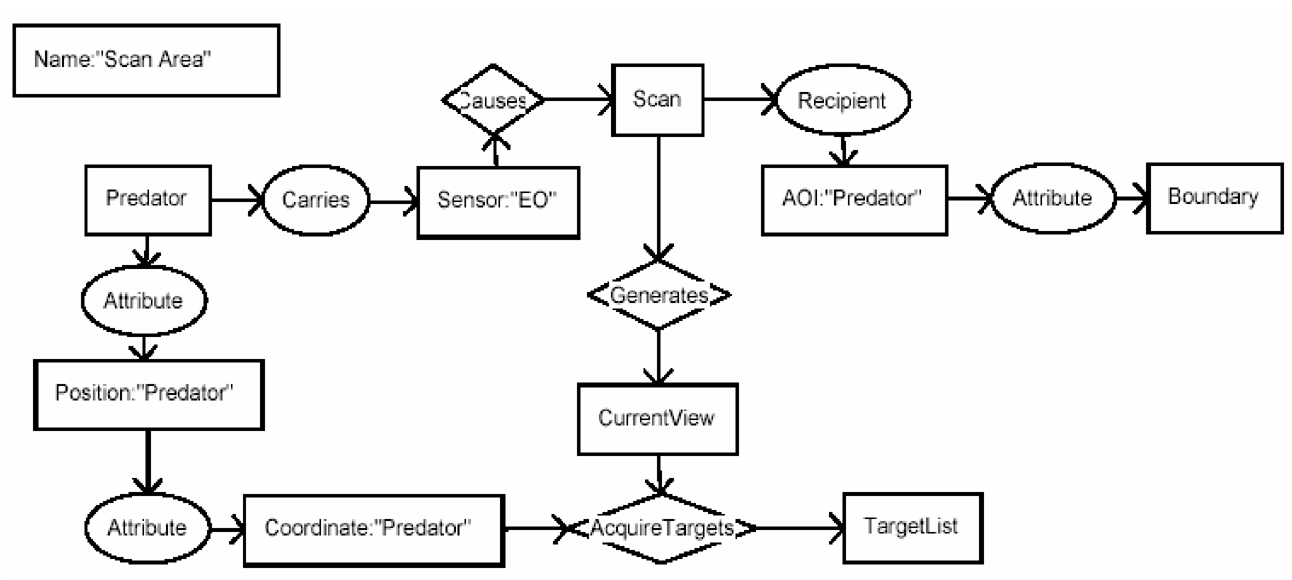


Figure 8-9. Scan Area CG.

The noteworthy features here are the way the HLA calls made to the PRISM EO Sensor federate and the data received from it are nicely hidden using the semantics above – "causing" the scan and "generating" the Current View. Note that Acquire is called again to sort the various screen elements into a list of potential targets.

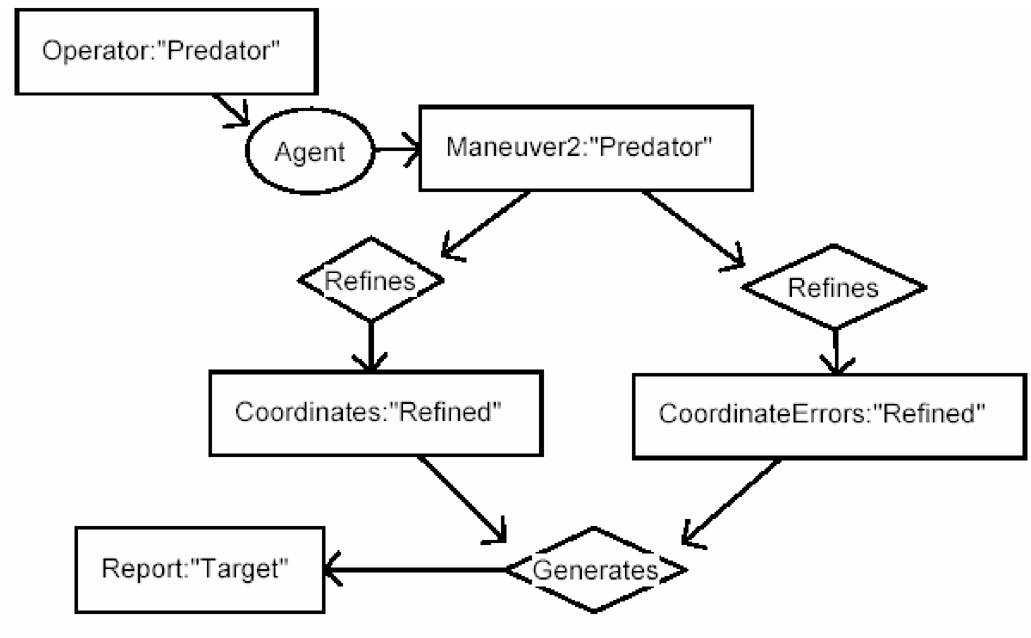


Figure 8-10. CG that determines if the coordinate quality is sufficient.

The above CG controls the behavior that implements the overflight-type behavior described above. If the coordinates have not been sufficiently refined, further maneuvering is performed until the coordinate quality is sufficient.

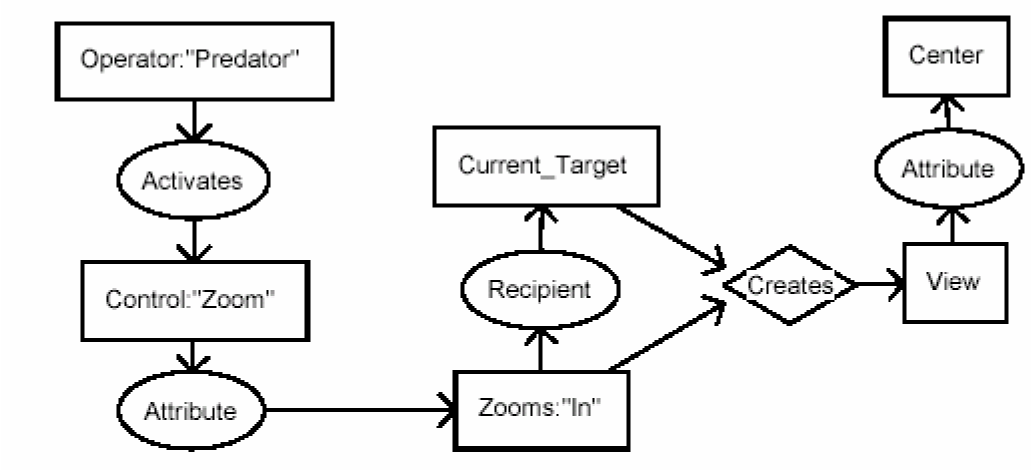


Figure 8-11. Zoom control CG.

The Zoom control CG is used in the process of coordinate refinement. Interestingly, the Acquire model is used to control this behavior. As the sensor is zoomed in, the probabilities computed in Acquire increase, leading to a nice way to control the zooming behavior.

8.2 F-15E OPERATORS

We now provide a discussion of the implementation of the model of the F-15E Operators. As before, a variety of functionality is nicely hidden – HLA calls moderating interactions with the TST Cell, IR sensor, and RF sensor federates; encapsulating calls to Acquire to perform target detection/ID based on the properties of the screen elements, and calls made to JSAF APIs to control the maneuvering of the platform.

8.2.1 Maneuvering

In terms of on-station location and pattern, the F-15E will fly an East-West pattern, traveling two minutes at 420 knot on each leg. At the end of a leg a 30 degree bank turn is made and the F-15E heads back toward the original position (60 degree banks may be made). This pattern will be flown just south of the Camden Ridge/Pine Hill MOA.

The behavior controlling the F-15E's transit to and from on-station location to perform tasks are as follows. Once the F-15E has been tasked, it will not be re-tasked until the current task is complete. When the F-15E is given coordinates to search, the LANTIRN can be used immediately if it is close enough – within ten miles – and the F-15E is flown directly at the target coordinates. Once the F-15E is tasked, there is no re-tasking until the current mission is completed.

If the F-15E is within ten miles when the order is received, first cue the LANTIRN at the target coordinates, and then fly the F-15E toward the target coordinates. If the target is not found after an over-fly, travel two minutes out from the target coordinates, and then maneuver directly back at the target coordinates.

If the APG-70 must first be used, the following behavior must be performed. The target coordinates have to be between 70 deg and 20 degrees off of the F-15E heading. When the F-15E Weapons System Officer (WSO) starts making the SAR map, the target coordinates need to be between 20 and 50 degrees off of heading.

If the target coordinates are too close but not in range of the LANTIRN, fly directly out from the target coordinates until the F-15E is 20 miles out, then turn back in at a 60 degree bank, and offset the nose 30 degree left or right. The SAR map may then be executed. Do a second SAR map 15 seconds after the first one using the refined target coordinates. When finished with the making the SAR map and refining the coordinates, the LANTIRN is cued to the refined target coordinates and the F-15E is flown directly at the target coordinates.

8.2.2 Controlling the Sensor

The SAR is used between 20 and 40 miles out. Put the target 20 to 50 degrees off the nose and at 20 miles out make a 5 mile resolution map. Using these coordinates, after a 15 second delay, a second SAR map is made.

The LANTIRN is activated when the second SAR map is done after an image analysis interval (~15 sec). This can be done anytime within 10 miles.

8.2.3 Conceptual Graphs

The behavior of the F-15E Operators is implemented using a similar philosophy as with the UAV Operators – maneuvering the platform into position before using the sensors. The difference here is that two different sensors must be used in sequence, with the results from the first sensor being fed into the second. The top-level CG is very similar to that of the UAV Operator.

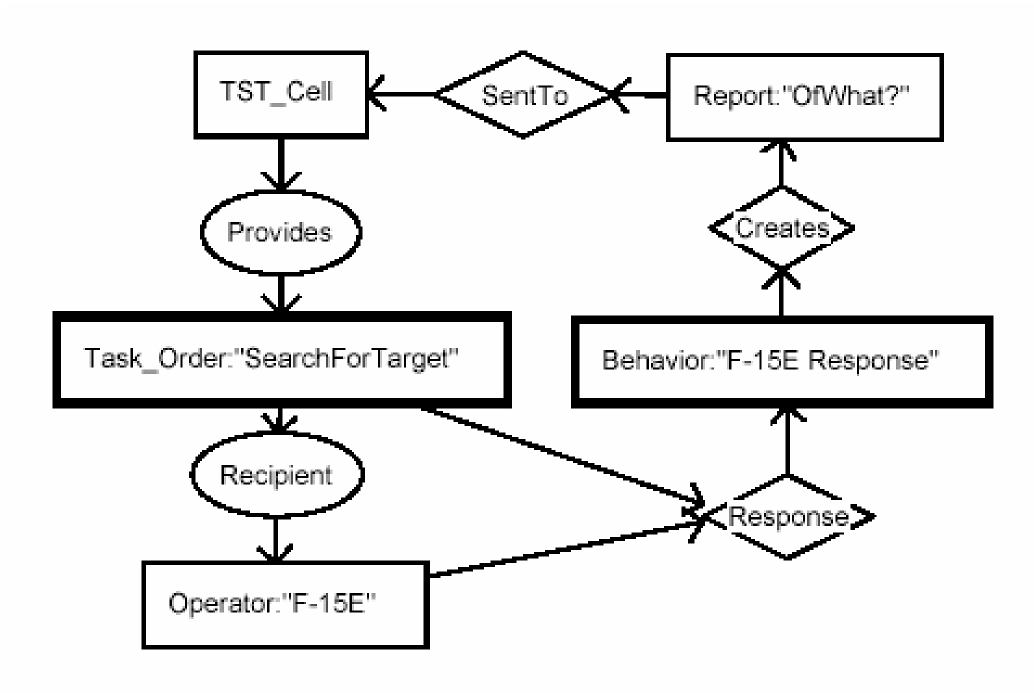


Figure 8-12. Top-level CG for the F-15E Operators.

In Figure 8-12 above, the TST Cell concept represents an encapsulation of the "ghost" representation of the TST Cell and hides the HLA calls made to get information from the TST Cell federate. The F-15E receives a task order, and the recipient is the F-15 Operator.

Synchronization is enforced before the Operator responds using an actor. When both inputs to the response actor are TRUE, namely a task order has arrived and the Operator is ready to receive it, the Response concept is then evaluated. When the Response CG has been executed, a report is constructed and sent to the TST Cell.

The Response CG is more complex than before, but has one fundamental morphological similarity to the UAV Operator CG. The Response CG has two parts, the top CG governing the platform maneuvering before the sensors can be used with the key action being the computations of the values of the flags "SAR Ready" and IR Ready".

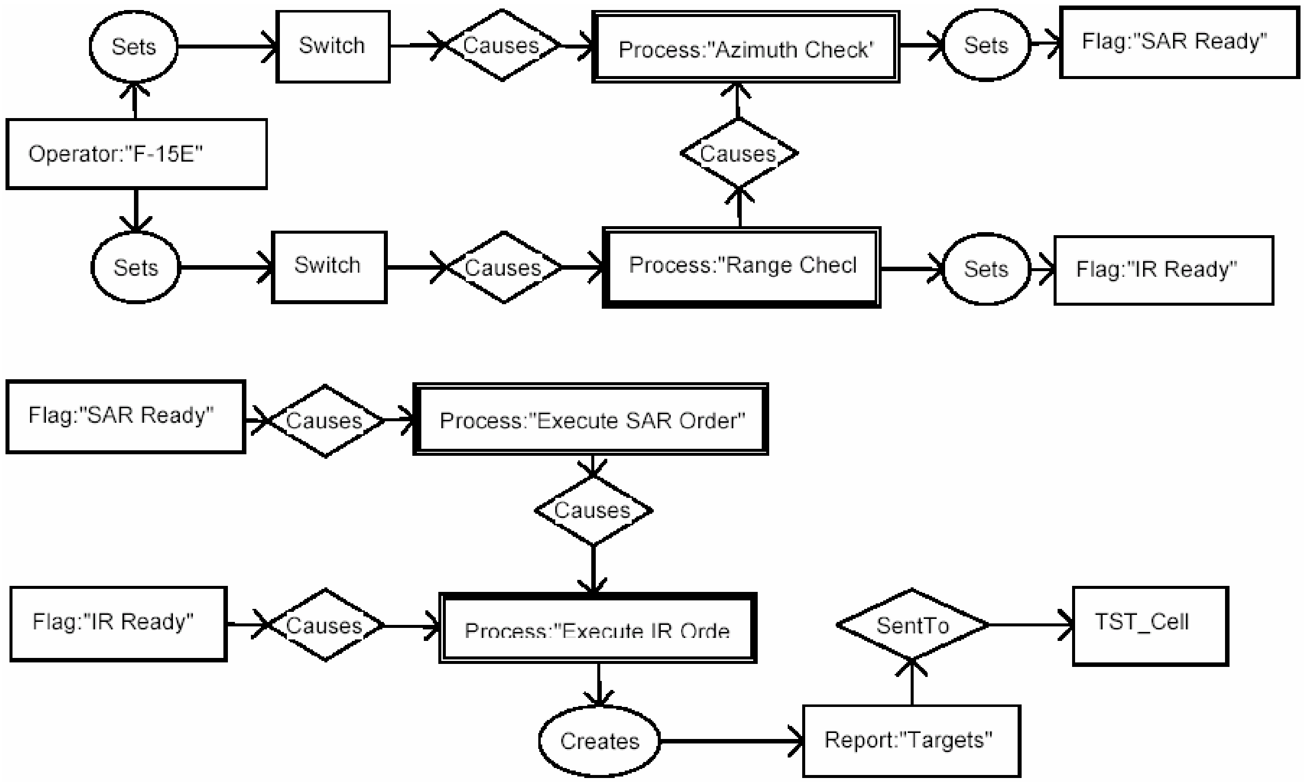


Figure 8-13. F-15E Operator Response CG.

The **Azimuth Check** and **Range Check** concepts encapsulate the calls made to JSAF to maneuver the platform correctly in order to employ the sensors. Not surprisingly, since there are azimuth constraints on the usage of the APG-70, the azimuth must be explicitly controlled.

The **Execute SAR Order** concept implements the interaction with the ACS federate that is used to model the APG-70, and makes the required HLA calls to send the control interactions to the ACS federate and receive the target messages from it.

The **Execute IR Order** concept implements the interaction with the PRISM federate that is used to model the LANTIRN pod, and makes the required HLA calls to send the control interactions to the PRISM federate and receive the list of screen elements from it. This also encapsulates the calls made to Acquire that use the list of screen elements to compute detections and IDs.

We now examine the CG used to control maneuvering with respect to azimuth, to illustrate how maneuver is implemented. A similar approach is taken for range maneuvers.

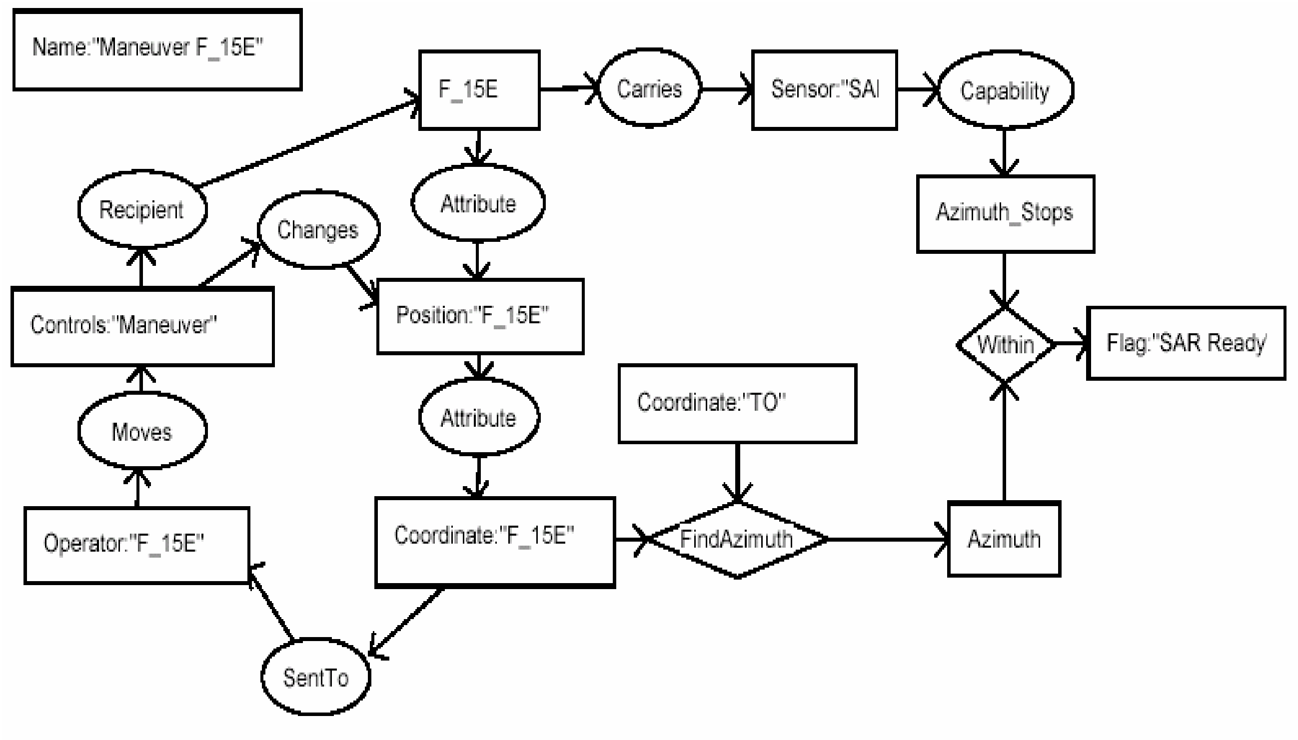


Figure 8-14. Azimuth maneuver control CG.

The Maneuver CG encapsulated the calls made to JSAF to control the platform model. The key computations are the actors that do the simple math of finding the range and determining if the azimuth is within the limits of performance the AGP-70 SAR model. Once the F-15E is maneuvered within the correct azimuth, the SAR Execute Order CG can be executed.

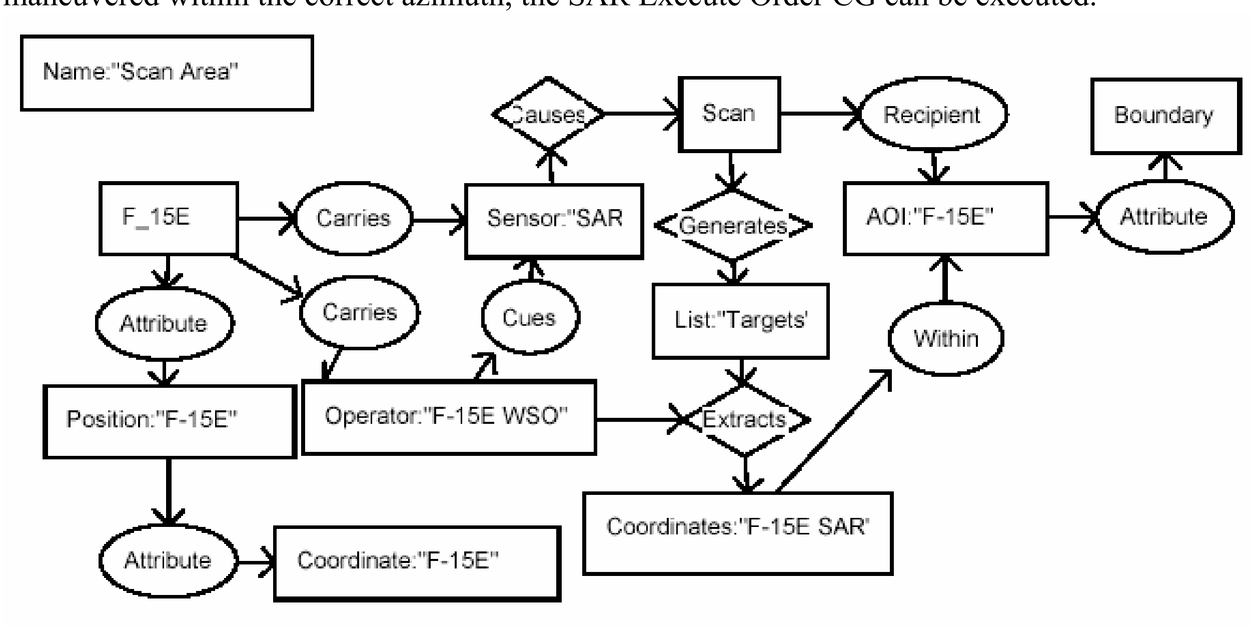


Figure 8-15. SAR Scan Area CG.

The most important features in the **SAR Scan Area** CG are the way the HLA calls made to the ACS RF Sensor federate and the data received from it are nicely hidden using the semantics above – "causing" the scan and "generating" the list of targets. Note that unlike the situation in the previous section on UAV Operator modeling, the code that evaluates the output of the sensor is still bundled into the RF sensor federate, unlike the way the EO federate separated sensor functions and human behavior, in this case operators looking at the SAR image and making detection decisions.

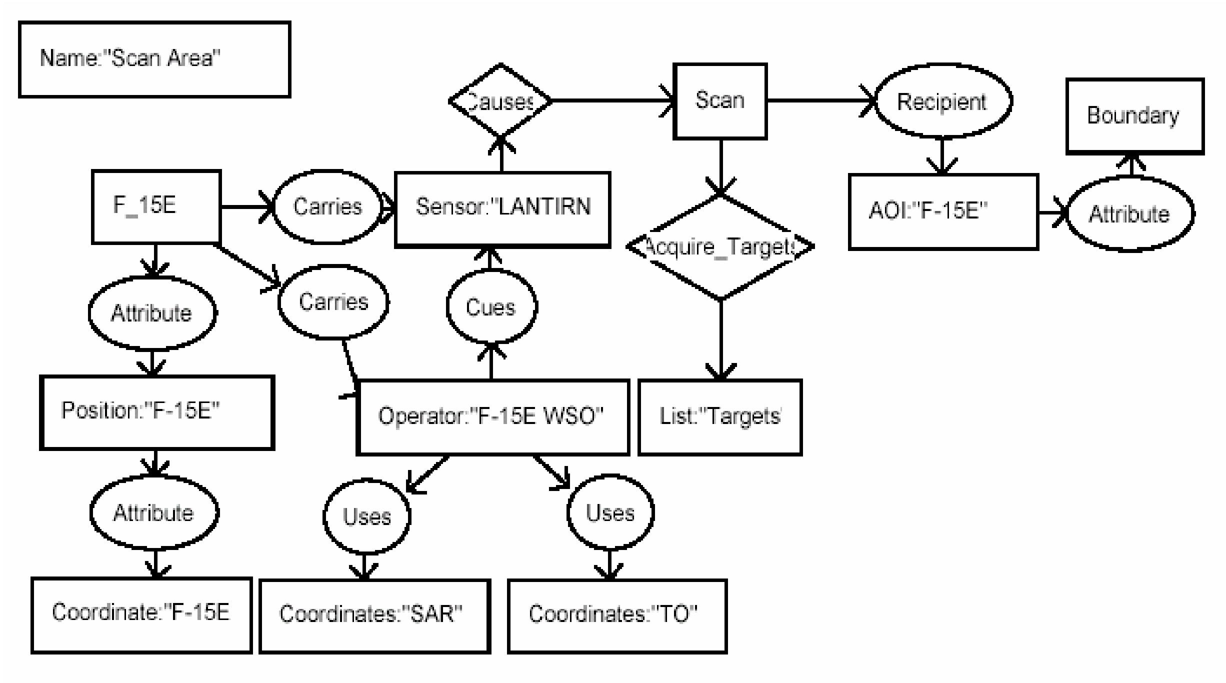


Figure 8-16. Scan Area CG for LANTIRN.

Note the similarity the between the **IR Scan Area** CG and the **SAR Scan Area** CG. The main difference is that as with the UAV Operator, the IR federate sends back screen elements, not detections. Note that like the situation in the previous section on UAV Operator modeling, the code that evaluates the output of the sensor is separated from the IR sensor federate – PRISM, unlike the way the RF federate combined sensor functions and human behavior.

8.3 JOINT STARS (JSTARS) OPERATORS

The implementation of the agent to model the JSTARS Operators is slightly less complex in that no maneuvering needs to be performed, since the JSTARS is flying a fixed figure eight racetrack. This orbit is located so that the Camden Ridge/Pine Hills MOA are within the APG-8 radar system, used in the SAR and GMTI modes. The bulk of the work in this implementation is determining when to activate the sensors based on where the JSTARS is on the racetrack, and doing the HLA coordination with the HLA federate that models the sensor in terms of commands and responses – in this case a list of detections.

The pattern established with the first two agents in terms of a top-level CG are the same. In Figure 8-17 below, the TST Cell concept represents an encapsulation of the "ghost" representation of the TST Cell and hides the HLA calls made to get information from the TST Cell federate. As with the established pattern, the JSTARS receives a task order in order to initiate execution of the agent CGs.

Synchronization is enforced before the Operator responds using an actor. When both inputs to the response actor are TRUE, namely a task order has arrived and the Operator is ready to receive it, the Response concept is then evaluated. When the Response CG has been executed, a report is constructed and sent to the TST Cell.

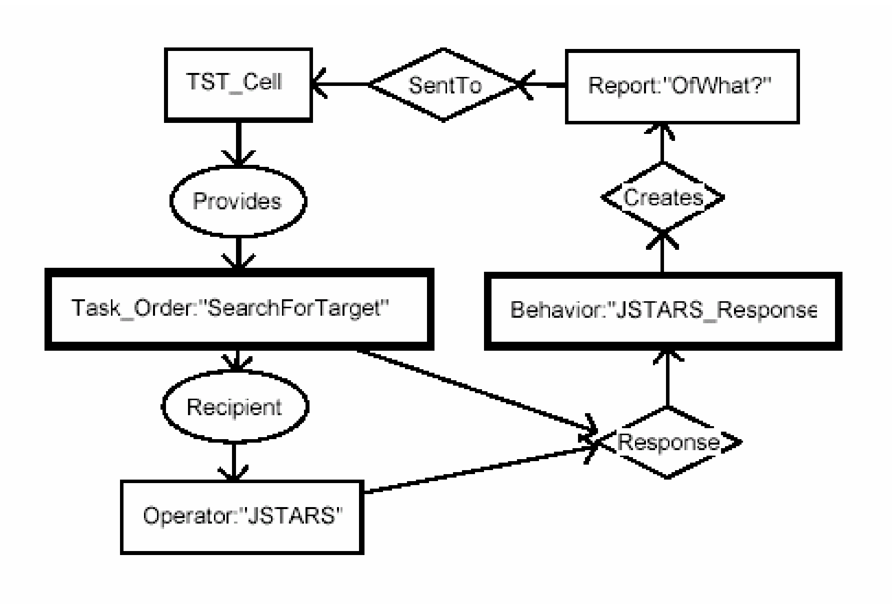


Figure 8-17. Top-level CG for the JSTARS Operators.

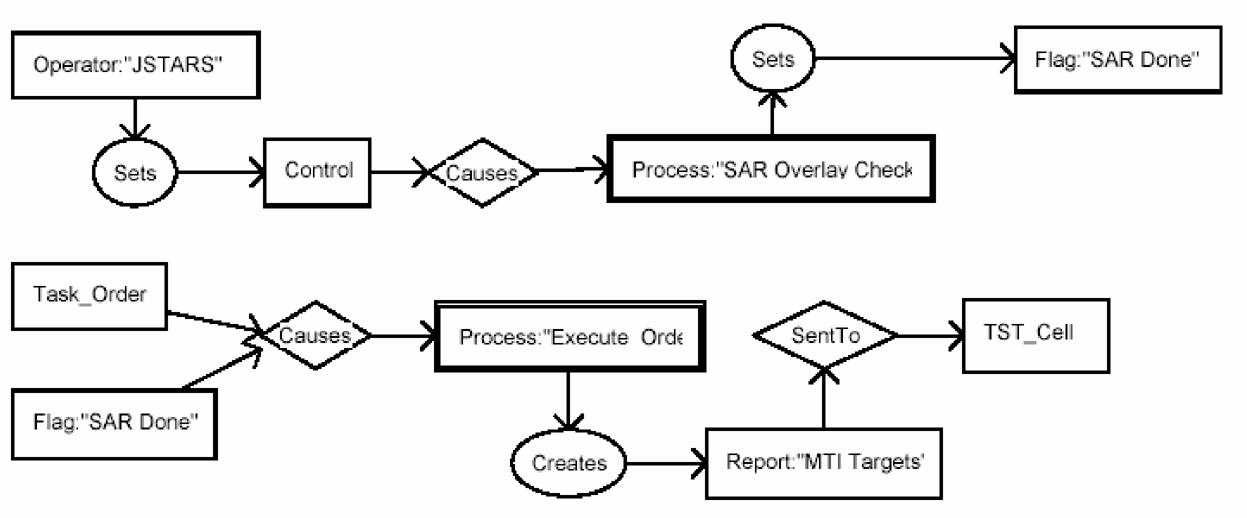


Figure 8-18. JSTARS Response CG.

The Response CG for the JSTARS, illustrated in Figure 8-18 above, differs from the previous two implementations, but has one fundamental structural similarity to the UAV and F-15E Operator Responses. The Response CG has two parts, the top CG governing an action that must be performed before the sensors can be used with the key action being the computation of the values of the flag "SAR Done".

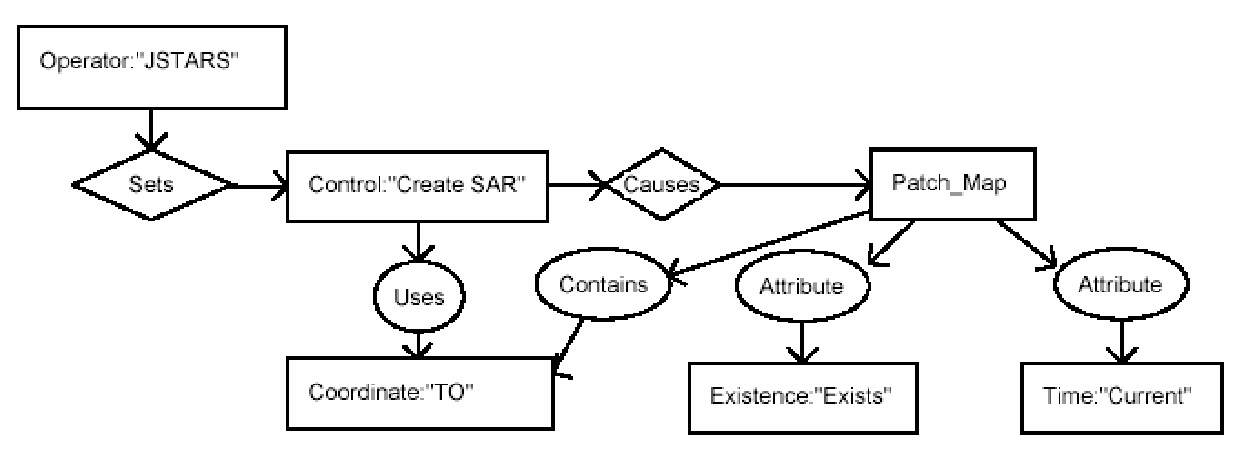


Figure 8-19. The SAR Overlay Check CG.

The **SAR Overlay Check** concept in figure 8-18 is implemented in Figure 8-19 above. The key function performed is the encapsulation of the HLA interaction with the ACS federate that models the APG-8 in order to generate a SAR patch map. This sets the "Exists" and "Time Current" flags to be true and allows the rest of the processes controlling the use of the APG-8 sensor in the GMTI mode.

As is shown in Figure 8-20 below, once these synchronization points in the previous CGs are passed, the **Scan Area** concept, implemented as a CG in Figure 8-21 can be executed.

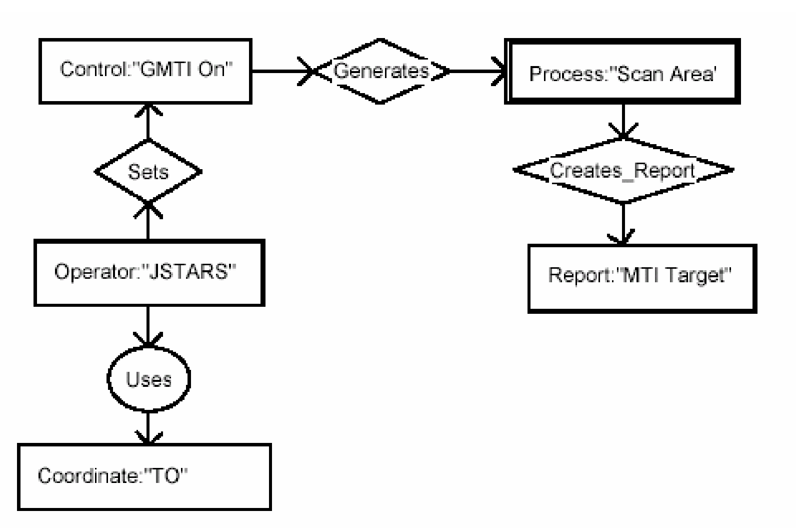


Figure 8-20. GMTI Control CG.

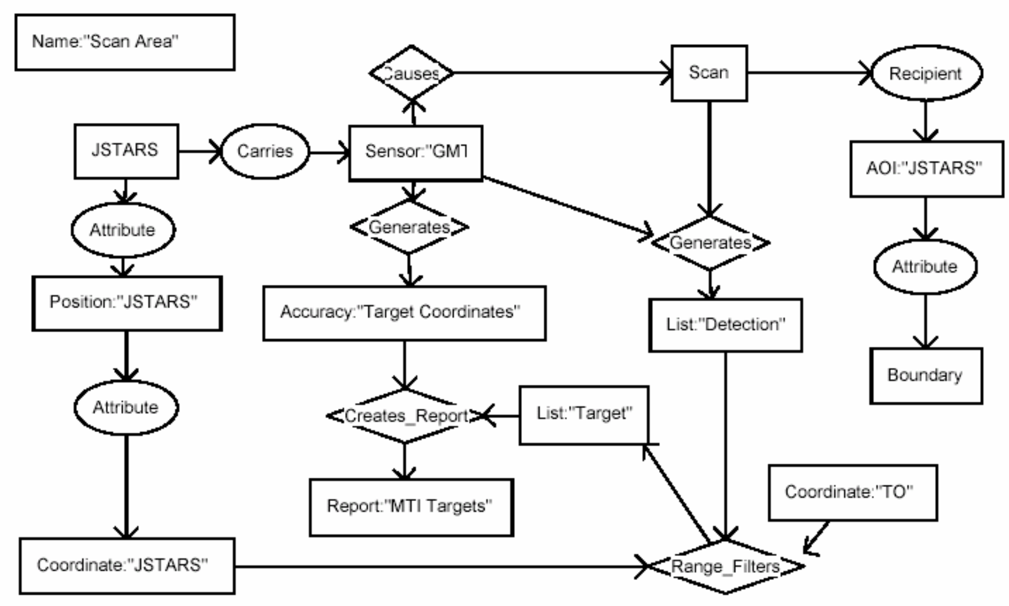


Figure 8-21. JSTARS Scan Area CG for GMTI.

The most important features in the **SAR Scan Area** CG are the way the HLA calls made to the ACS RF Sensor federate and the data received from it are nicely hidden using the semantics above – "causing" the scan and "generating" the list of targets. Note that unlike the situation in the previous section on UAV Operator modeling, the code that evaluates the output of the sensor is still bundled into the RF sensor federate, unlike the way the EO federate separated sensor functions and human behavior, in this case operators looking at the GMTI display and making detection decisions.

8.4 TIME SENSITIVE TARGETING CELL

The Time Sensitive Targeting (TST) Cell implementation is slightly different than the previous three agents, in that the output of an ***organization*** is to be modeled as opposed to ***specific human functions***. While CORE could be used to model the TST Cell in this way, the time and scope of the effort does not permit this. Instead, we focus on modeling the effective outputs of the organization – the allocation of ISR and strike assets to targets based on C2ISR inputs. In general, we model ISR assets from the perspective of their ability to generate target coordinates of various qualities and identify targets as moving or stationary. The goal is to generate a refined target coordinate of a target that is stationary and assign a strike asset, also based on target priority, in the minimum time.

In order to accomplish this, we map error estimates of target coordinates from ISR assets (Rivet Joint, JSTARS, and the UAV) into scale of 1-5, one being the most refined and 5 the least. Each asset has some ability to lower the refinement value. The TST Cell implementation seeks to assign a sequence of assets that minimizes the coordinate quality value in the least amount of time, and then the target is assigned a strike asset when it becomes stationary.

The pattern established with the first three agents in terms of a top-level CG is the same, so we omit it here. Synchronization is enforced before the Operator responds using an actor. When both inputs to the response actor are TRUE, namely a task order has arrived and the Operator is ready to receive it, the Response concept is then evaluated. When the Response CG has been executed, a task order is constructed and sent to the appropriate asset via an HLA call.

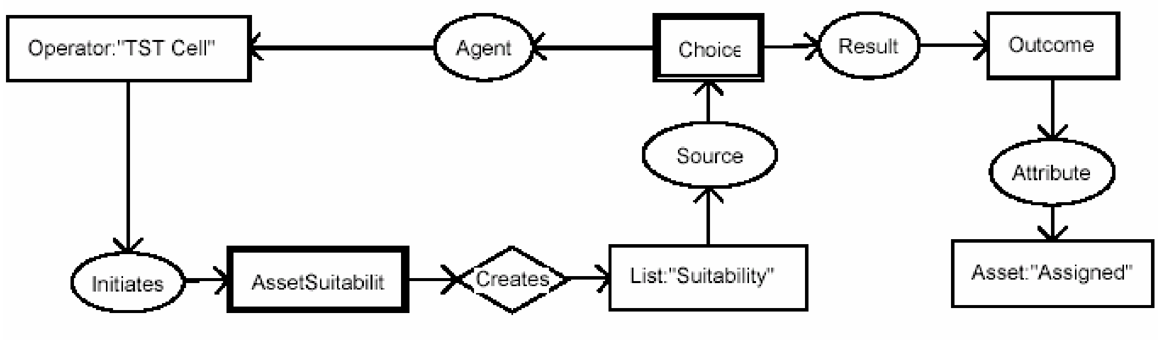


Figure 8-22. TST Cell main control CG.

As is shown in Figure 8-22 above, the main control CG first assesses asset suitability. Once a list of suitable assets has been formulated, the concept Choice has its CG executed. This is where the bulk of the work arises. The optimization problem has been segmented into nine cases. The nature of the CG shown below in Figure 8-23 indicates that each mutually exclusive option is checked. For purposes of illustration a representative selection of the cases will be discussed.

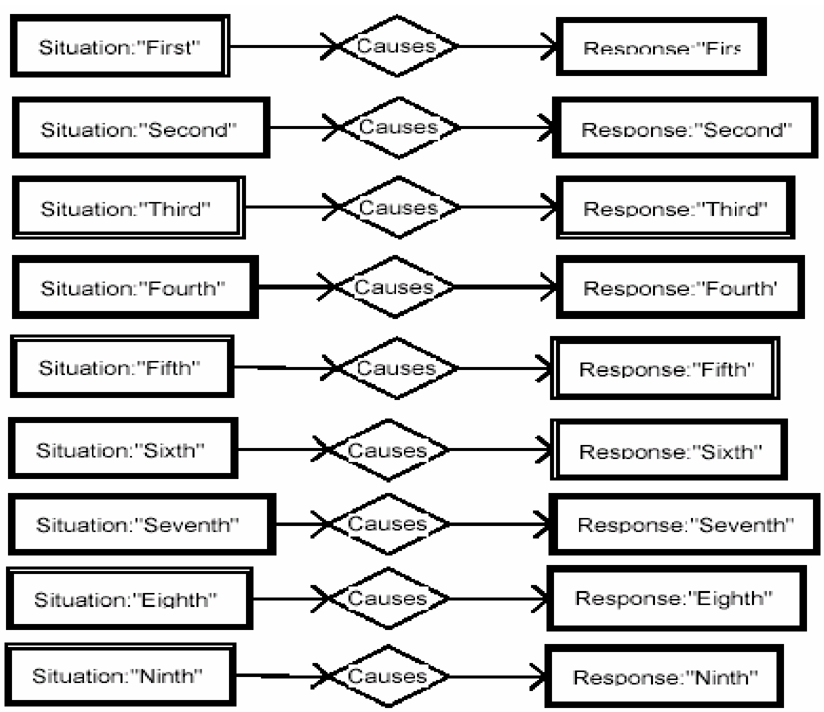


Figure 8-23. The Choice concept CG implementation.

The first case, illustrated in Figure 8-24 below, labeled **Situation: First** covers the easiest case where there is a stationary target with a coordinate quality of 1 and a priority of 1.

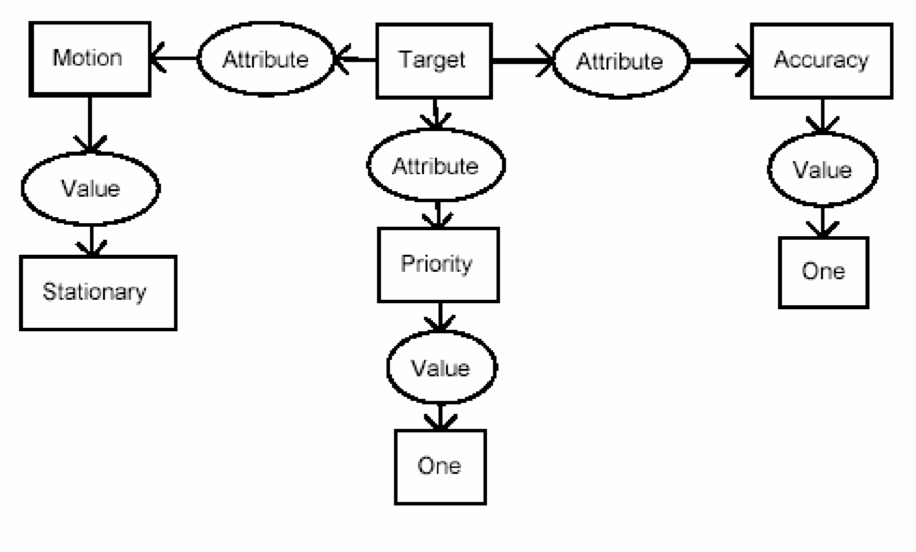


Figure 8-24. The CG governing a stationary target with the highest coordinate quality and priority.

In this case, as Figure 8-25 illustrates, the F-15E is the asset selected.

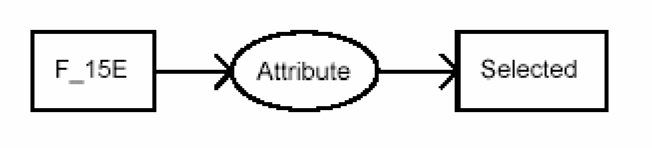


Figure 8-25. CG selecting the F-15E as the asset.

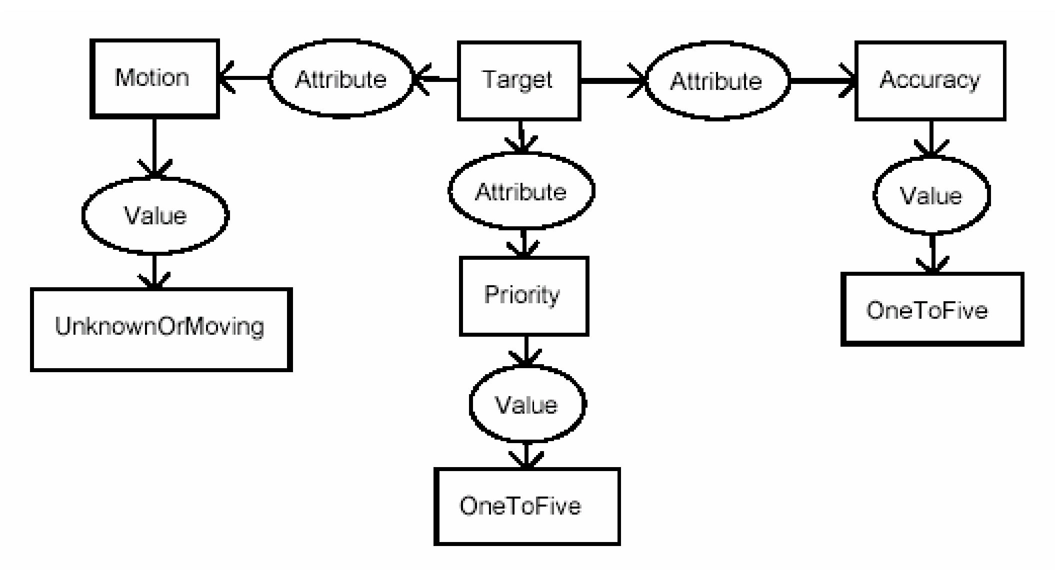


Figure 8-26. CG governing the second situation considered.

In this case, the target motion is unknown or moving, the target coordinate quality can range from 1 to 5, and the priority can be from 1 to 5. Due to the unknown motion of the target, JSTARS is the asset selected, and is implemented as in Figure 8-26, with JSTARS substituted for the F-15E.

In the third case to be considered, shown below in Figure 8-27, there is more than one target, so both the UAV and JSTARS are selected.

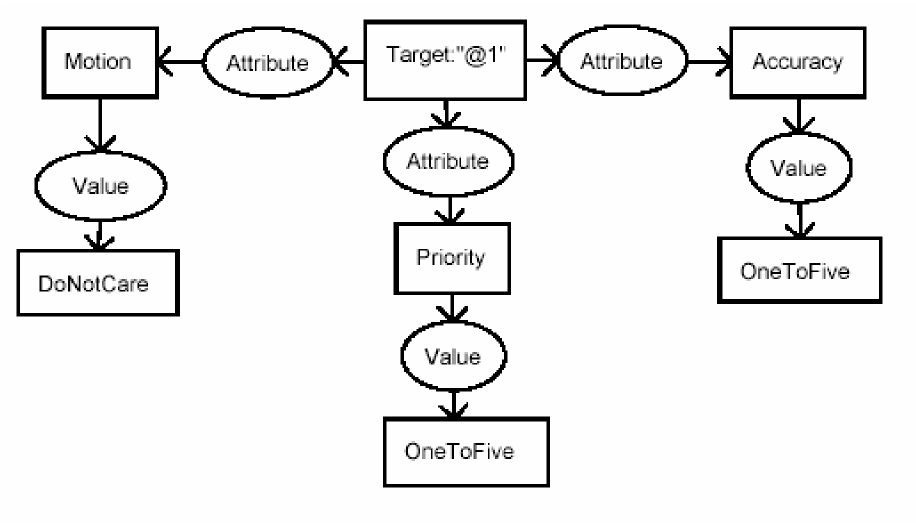


Figure 8-27. CG for more than one target.

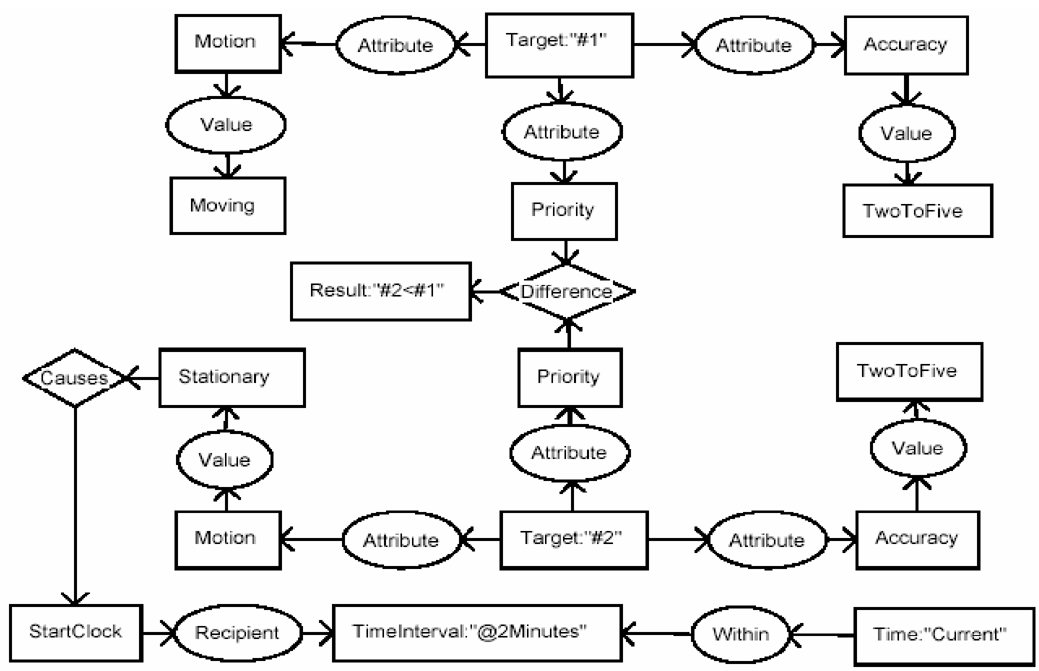


Figure 8-28. CG for the case of two targets, one stationary, one moving, and each has unrefined coordinates.

In the next case, shown above in Figure 8-28, the logic becomes more complicated, where there are two targets with one moving and one stationary, but the coordinate quality has not been refined enough to permit strike asset tasking. A time-based process is begun for the stationary target, with a nominal duration of two minutes that would force a re-evaluation of asset tasking. Initially, both the JSTARS and the UAV are assigned to monitor the moving target, both to determine if it becomes stationary and to further refine its coordinates when it does, as shown in Figure 8-30.

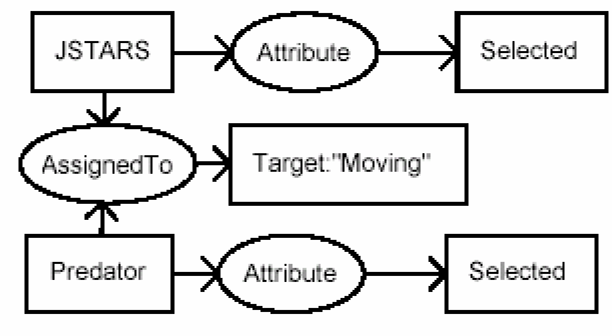


Figure 8-30. CG assigning both the UAV and JSTARS to a moving target.

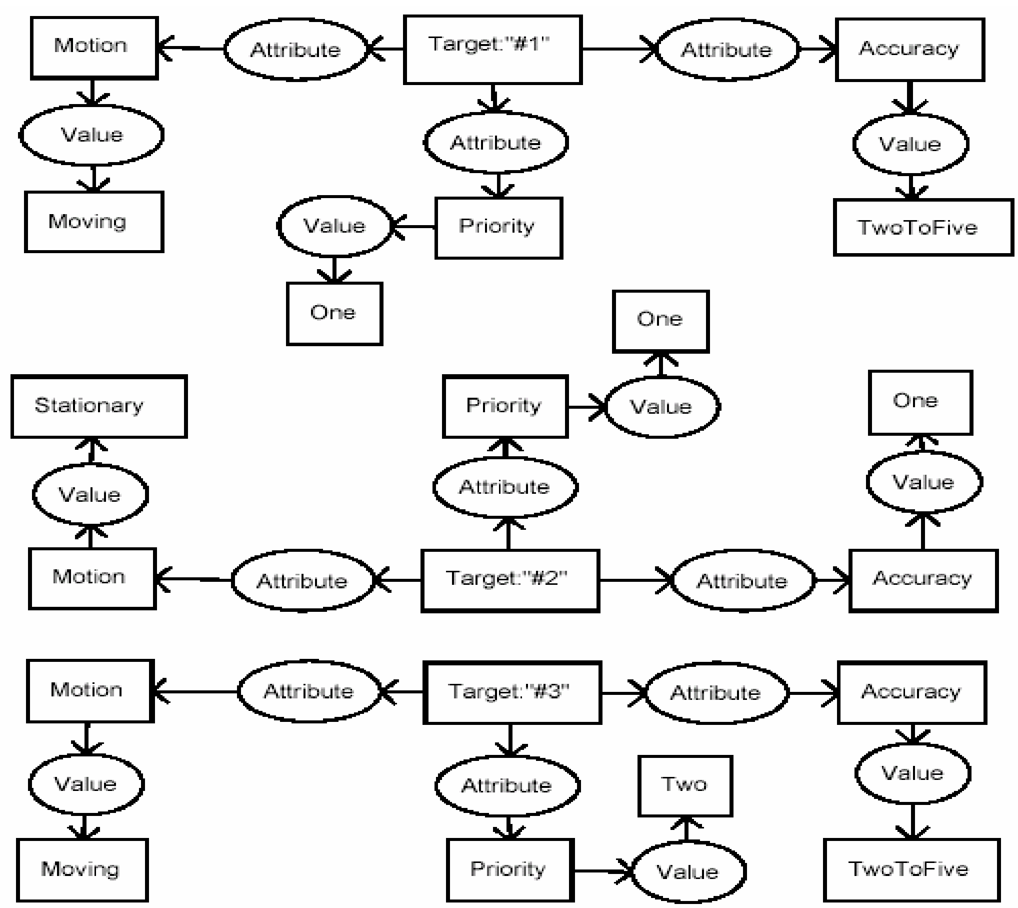


Figure 8-31. CG illustrating a basic three target case.

The three target case illustrated in Figure 8-31 above considers the case where one target is stationary and has refined coordinates, but the other two targets are moving but with unrefined coordinates. In this case, the F-15E is the asset selected for the stationary target, JSTARS is assigned to all the moving targets, and the UAV is assigned to the highest priority moving target. The CG shown below in Figure 8-32 implements this asset assignment.

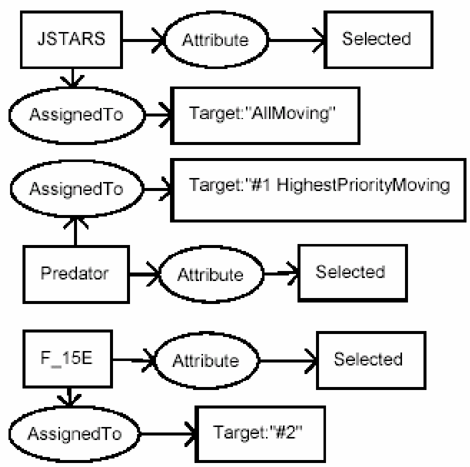


Figure 8-32. CG implementing the asset assignment for the three target case.

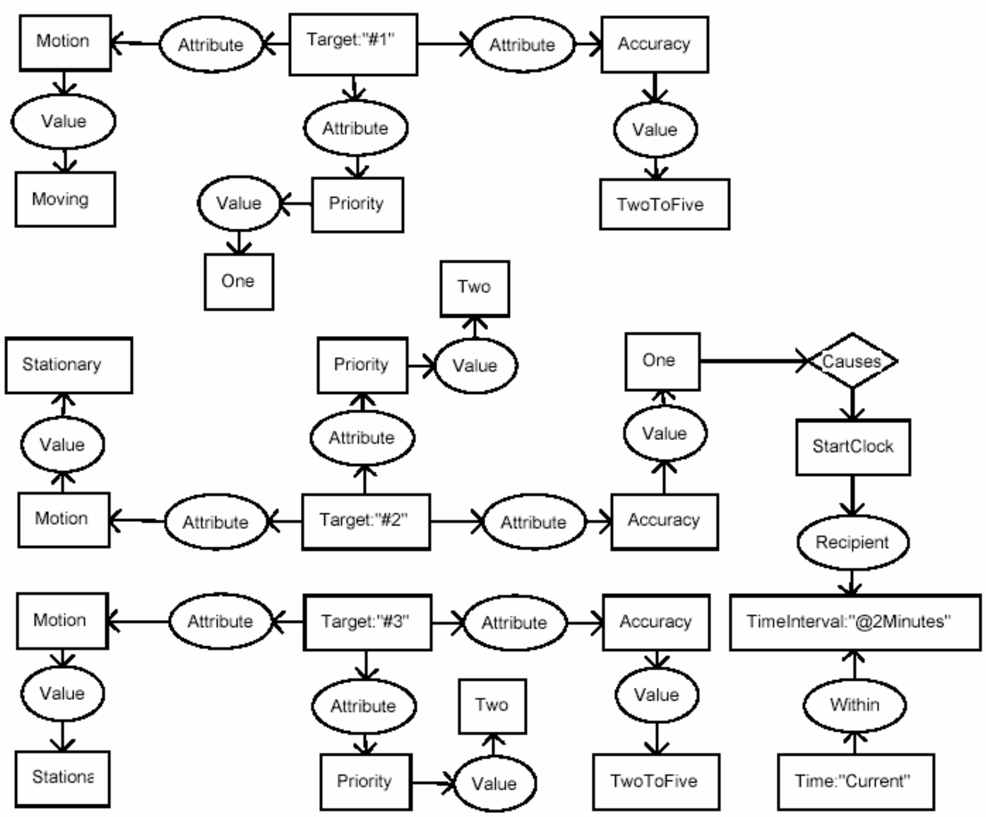


Figure 8-33. Complex three target case.

In the complex three target case shown in Figure 8-33 a high priority moving target requires further coordinate refinement, but two targets that are stationary where one has refined coordinates and one requires further refinement. A timer is set on the stationary target with refined coordinates for re-evaluation. In this case the JSTARS is assigned to the moving targets, the UAV is assigned the highest priority moving target, and the F-15E is assigned the stationary target. The purpose of the timer activity in Figure 8-33 is to cover the case that a higher priority moving target becomes stationary and its coordinates can be refined. Figure 8-34 below illustrates the asset assignment.

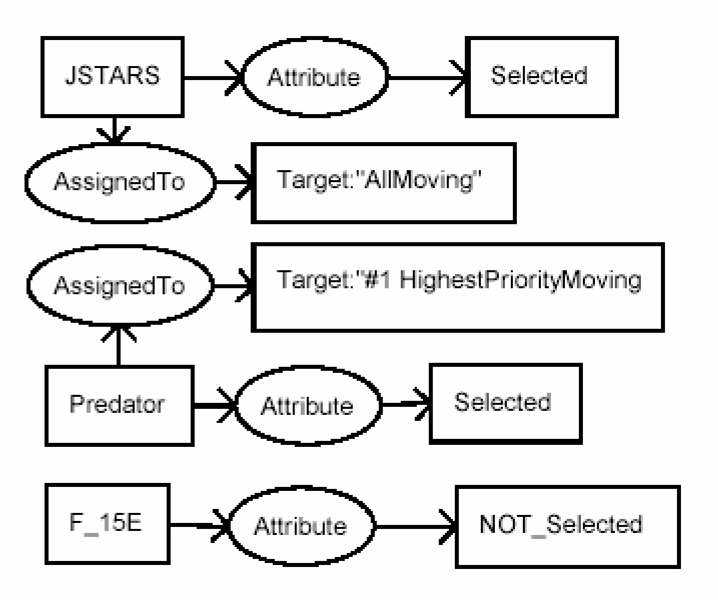


Figure 8-34. CG for asset assignment in the complex three target case.

The remaining cases can be similarly understood using the cases illustrated up to this point.