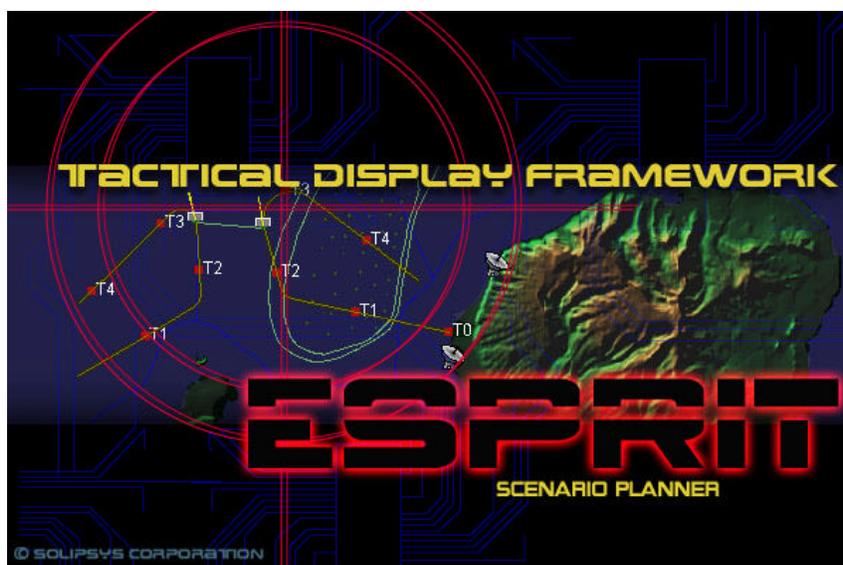


User's Manual

Exercise Scenario Planning Real-time Integrated Test (ESPRIT) Tool Version 4.0



UM Version 1.4
Document No.: SOL-01-032

March 1, 2002

Prepared by:

SOLIPSYS



SOLIPSYS CORPORATION
6100 Chevy Chase Drive Suite 200 Laurel, MD 20707-2929
301/568-8900 voice 301/483-8901 fax
www.solipsys.com

ABSTRACT

The Exercise Scenario Planning Real-time Integrated Test (ESPRIT) was developed to support a variety of applications associated with the planning, rehearsing, and execution of complex missions, test scenarios and system stimulation/simulation events. This User's Manual describes the user's interaction with the system to build, preview, rehearse and execute a scripted scenario, and to define both sensor and vehicle models and instantiate these models into the context of an overall scenario or test event. The manual describes the following functions:

Pre-planning – The creation of vehicle and sensor models and entities; the creation of overlays. These are the basic building blocks required to construct a scenario. A library of models is provided that have been created by others. The fidelity of the model is dependent on the parameters of the scenario being defined.

Scenario-planning – The use of previously created models for the assignment of sensors and entities to a scenario, and description of the behavior of these entities throughout the course of the scenario.

Scenario-execution – Occurs in one of three modes: preview mode, local execution mode, and event monitoring mode.

Stimulation mode – Provides outputs that are specific to the event being executed in exercise mode.

Revision History

Revision	Date	Description
1.0	01/23/1999	Version 1.0 released.
1.1	02/15/1999	Version 1.1 released.
1.2	05/31/2001	Version 1.2 released.
1.3	06/04/01	Version 1.3 released.
1.4	03/01/02	Version 1.4 released, RMT v4.0

*Comments or questions about ESPRIT or this User's Manual
should be directed to espritsupport@solipsys.com.*

TABLE OF CONTENTS

1	SCOPE.....	1
2	CONCEPT OF OPERATION	1
2.1	GUI/Server Architecture	2
2.2	Scenario Planning.....	3
2.2.1	Scenario Pre-Planning.....	3
2.2.2	Scenario Creation.....	3
2.3	Scenario Execution	3
2.3.1	Scenario Preview	3
2.3.2	Local Execution (Rehearsal)	4
2.3.3	Event Monitoring.....	4
3	ESPRIT DISPLAY	4
4	CAPABILITIES.....	5
5	PRE-PLANNING	6
5.1	Model Creation.....	7
5.1.1	Sensor Models	7
5.1.1.1	Sensor Model Chooser.....	7
5.1.1.2	Sensor Model Editor	8
5.1.1.3	Saving the Sensor Model.....	12
5.1.2	Vehicle Models	12
5.1.2.1	Vehicle Model Chooser	12
5.1.2.2	Vehicle Model Editor	14
5.1.2.2.1	Entry	15
5.1.2.2.2	Description.....	15
5.1.2.3	Saving the Vehicle Model	16
5.1.3	TBM and Interceptor Models	16
5.1.3.1	TBM and Interceptor Model Creation.....	16
5.1.4	Sensor Entities.....	17
5.1.4.1	Sensor Entity Creation.....	17
5.1.4.2	Sensor Entity Chooser.....	18
5.1.4.3	Sensor Entity Editor	19
5.1.4.4	Saving the Sensor Entity.....	21

6	PLANNING.....	22
6.1	Scenario Creation	22
6.1.1	Scenario Selection	22
6.1.2	Scenario Classification	25
6.1.3	Scenario Editor	25
6.1.3.1	Vehicle Entities and Trajectory Definitions	26
6.1.3.2	Adding a TBM or Interceptor Vehicle Entity.....	32
6.1.3.3	Synchronizing Vehicle Trajectories	33
6.1.3.4	Sensor Placement and Assignment	35
6.1.3.5	Constraint Definition.....	38
6.1.4	Save Scenario.....	39
6.1.5	Certify Scenario	39
6.1.6	Close Scenario	40
7	SCENARIO EXECUTION	40
7.1	Scenario Open	40
7.2	Scenario Chooser	41
7.3	Scenario Execution	43
7.3.1	Scenario Preview Mode	43
7.3.2	Local Execution	44
7.3.2.1	Track Simulation.....	45
7.3.2.2	Detection Simulation.....	45
7.3.3	Constraint Monitoring.....	46
7.3.4	Event Monitoring.....	47
8	INSTALLATION AND CONFIGURATION.....	55
8.1	Server Installation	55
8.1.1.1	Directory Structure	55
8.2	Server Configuration.....	57
8.2.1	Environment Variables	58
8.2.1.1	envset.csh	58
8.2.2	Automatic Startup	60
8.3	Client Installation	61
8.3.1	ESPRIT Client Installation.....	61
8.3.2	Client Configuration	62
8.3.2.1	Property Files	63
8.3.2.2	Environment Variables	63
9	NOTES.....	64

9.1 List of Acronyms and Abbreviations 64

LIST OF FIGURES

Figure 1. ESPRIT Iterative Planning Process	1
Figure 3. ESPRIT Client/Server Architecture	3
Figure 5.. ESPRIT Main Display.....	5
Figure 7. Sensor Model Menu Selection.....	7
Figure 9. Sensor Model Chooser Panel.....	8
Figure 11. Sensor Model Editor Panel.....	9
Figure 13. Vehicle Model Menu Selection.....	12
Figure 15. Vehicle Model Chooser Panel.....	13
Figure 17. Vehicle Model Editor Panel	14
Figure 19. Saving the Vehicle Model	16
Figure 21. Sensor Entity Creation.....	18
Figure 23. Sensor Entity Chooser	19
Figure 25. Sensor Entity Editor	20
Figure 27. Open Scenario Menu Selection	22
Figure 29. Scenario Chooser Panel.....	23
Figure 31. Scenario Editor Panel	24
Figure 33. Set ScenarioClassification Panel	25
Figure 35. Information Panel	27
Figure 37. Trajectory with Waypoint Menus.....	29
Figure 39. Set Orbit Menu Option.....	31
Figure 41. Orbital Trajectory in Clockwise Direction.....	32
Figure 43. Synchronizing Vehicle Trajectories	34
Figure 45. Scenario Editor Panel	35
Figure 47. Sensor Information Panel	36
Figure 49. Sensor Selector Application	36
Figure 51. Sensor Assignment Panel	37
Figure 53. Sensor Assignment: Add Sensor Assignment Panel	37
Figure 55. Constraint Editor	38
Figure 57. Certify Scenario Menu Selection.....	39
Figure 59. Open Scenario Menu Selection	40
Figure 61. Scenario Chooser Panel.....	41
Figure 63. Opened Scenario.....	42
Figure 65. Scenario Playback Control Panel	43
Figure 67. Scenario Preview Mode.....	44
Figure 69. ESPRIT Simulation Preference Panel	45
Figure 71: Sample RADARS.cfg file with DTEPORT Added.....	46
Figure 73. Constraint Monitor	47
Figure 75. ESPRIT Altitude Graph.....	48
Figure 77: ESPRIT Home Directory Structure	56
Figure 79: ESPRIT ServerSupport Directories	57
Figure 81: ESPRIT Client Configuration Area.....	62

LIST OF TABLES

Table 1. Sensor Model Chooser Field Description.....	8
Table 3. Sensor Model Editor Field Description	10
Table 5. Vehicle Model Chooser Field Description	13
Table 7. Vehicle Model Editor Field Description.....	15
Table 9. TBM Data Set Format.....	17
Table 11. Sensor Entity Chooser Field Description.....	19
Table 13. Sensor Entity Field Descriptions	21
Table 15. Scenario Chooser Field Description	24
Table 17. Scenario Editor Field Description.....	26
Table 19. Information Panel Field Descriptions	27
Table 21. Menu for First and Last Waypoints	29
Table 23. TBM Pop-up Menu Options	33
Table 13. Server Startup Scripts	58
Table 26:ESPRIT Server Environment Variables	59
Table 16: ESPRIT Server File Location Environment Variables	60
Table 18: General Environment Variables.....	60
Table 19: ESPRIT Client Properties	63

ESPRIT OVERVIEW

EO.1 INTRODUCTION

The Exercise Scenario Planning and Real-time Integrated Test (ESPRIT) concept was developed to support the requirements of planning, rehearsing, and executing complex exercises in a variety of situations including hardware in the loop and simulated environments.

The ESPRIT system supports the planning, training, rehearsal, and real-time execution of events in both large scale test range operations as well as in laboratory and system integration and test environments. Additionally, ESPRIT provides an extensive data extraction and post-event analysis capability for event playback and reconstruction.

EO.2 ESPRIT FUNCTIONAL REQUIREMENTS

The ESPRIT system comprises five major components: Visualization, Scenario Planning, Scenario Execution, Data Synthesis, and the Scenario/Event Monitor. The following sections of the Executive Summary highlight the functional requirements of each of the five components.

EO.2.1 Visualization

The visualization tool is built using the Tactical Display Framework (TDF) and provides the user interface to the ESPRIT system for scenario planning, rehearsal, execution, event monitoring, and post-event analysis.

The top-level elements of the visualization tool are:

- Intuitive graphical capability provided to:
 - Enter hazard regions, target and interceptor profiles and characteristics, and mission objectives.
 - View simulated execution of mission plan.
 - View exercise in real-time.
 - Specify and view analysis output during and post-event.
- Integrated display environment for planning and real-time execution that includes the following features:
 - Enter hazard regions, target and interceptor profiles and characteristics, and mission objectives.
 - Capability to enter vehicle track data graphically or via latitude, longitude, and altitude-specific input.
 - Ability to specify/identify a sufficient number of entities, including targets and other support resources, to adequately script and model any likely range exercise.
 - Embedded instrumentation, target, vehicle, debris, and sensor models.
 - Embedded geographic (land/terrain) models.
 - Capability to manually enter and display alphanumeric data over scenarios associated with moving or static elements of the scenario.

- Ability to replay entire and/or partial elements of a mission at variable speeds both slower than and faster than real-time.
- Capability to define tolerance windows for contingency planning.
- Capability to develop single and multiple entity associated timelines to include scenario events and TSPI data outputs.
- Capability to automatically integrate (canned) standard target “pre-launch” scenario data for each available target.
- Ability to manually modify single and associated timelines.
- Provide decision aids to Operations (OPS) Conductor or Test Director during events
- Provide safety alerts as defined by input safety constraints.

EO.2.2 Scenario Planning

The Scenario Planning component is used in conjunction with the visualization tool to interpret the planner’s inputs, to format the scenario plan, and to store the plan to disk. Additionally, this function performs scenario validity checks and provides various planning reports. Its primary functions are to:

- Process visualization tool data input
- Validate scenario script versus equipment performance model
- Compute derived components of the scenario plan such as radar/target Probability of Detection (Pd) tables
- Generate pre-event plans based on input constraints and Go/No-Go criteria
 - Timelines
 - Decision Aid Matrices
 - Profiles
 - Analysis

EO.2.3 Scenario Execution

The Scenario Execution function, under the control of commands issued from the visualization tool, downloads and executes the stored simulation plan. This function generates target detection data, track data, platform motion data, and truth data needed to drive the Data Synthesis and Scenario/Event Monitor functions and/or the external sensor tracking systems and displays. It also ensures the timing constraints are met for defined entity descriptions.

The top-level capabilities of this component are:

- Generation of dynamic output of event plan as provided by plan generator.
- Stand-alone visualization of exercise or test plan.
- Generation of target objects for transmission and synchronized stimulation of shipboard and airborne ESPRIT servers and platform sensors for event training and plan refinement (Future Growth).
- Generation of stand-alone and integrated system simulation/stimulation events.
- Control and synchronization of event and scripted time to the defined timing standard.
- Selects specific output formats for entity and sensor data required by ancillary systems.

EO.2.4 Data Synthesis

The Data Synthesis function correlates track state information from diverse sources and, to the extent possible provides a single, unduplicated track picture to the visualization tool. Track data sources can include sensor reports, tactical data links, simulated tracks from the Scenario Execution function, and simulated tracks from external Distributed Interactive Simulation (DIS) sources.

EO.2.5 Scenario/Event Monitor

The Scenario Event/Monitor Function monitors the state of the scenario tracks against a scripted set of constraints. If a target is evaluated as being in violation of one or more of its constraints, a notification is sent to the visualization tool to alert the operator of the constraint violation.

The Scenario/Event Monitor function:

- Evaluates all Fused Track Data against the scenario constraints and alerts the operator if a violation is detected.
- Maintains all of the constraints that are defined for the active scenario and maintains the current Timeline Mark, used to determine the specific set of constraints that are to be applied.
- Maintains the state of each scenario Target of Interest (TOI) based on the Sim Truth Data that are received from the Scenario Execution Function.
- Uses the simulated data to constrain key scenario targets, whose position may be used to further constrain other scenario targets.

1 Scope

This document describes the concept of operation of the Scenario Planning and Rehearsal Tool (ESPRIT) as well as the scenario planner's interactions with the system for the purposes of scenario planning, scenario preview, event rehearsal, and event monitoring. The key functionality provided by ESPRIT4.0 is scenario pre-planning, scenario scripting, scenario preview, local execution, and real-time event monitoring. This document describes only the procedures used for invoking these functions. These procedures are subject to change as the system matures. Additional user interaction with the system will be provided as additional system features are added.

2 Concept of Operation

ESPRIT was built to support the iterative design and execution of complex scenarios with primary emphasis on support of both simple and high fidelity models, multiple modes of input, user customization, rapid buildup of user defined tools, real time operation, and visual feedback to the user during both planning and execution. Figure 1 presents the ESPRIT planning and execution process. A new user spends time developing and refining models and overlays.

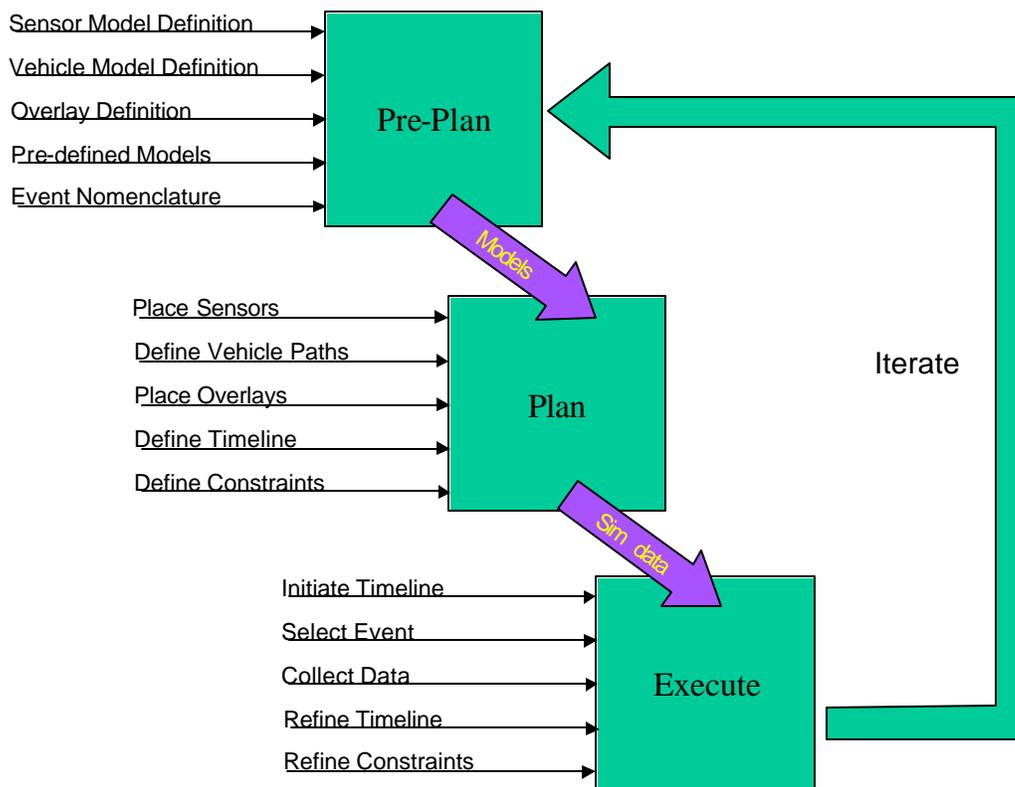


Figure 1. ESPRIT Iterative Planning Process

These models become the tool set for future planning activities. Subsequent activities include scenario planning and rehearsal, followed by model and plan refinement. Finally, ESPRIT supports the execution of the real event by providing a backdrop against which the event can be monitored and compared to assure that event objectives are met and that safety is not compromised.

2.1 GUI/Server Architecture

Figure 2 illustrates the ESPRIT client/server Graphical User Interface (GUI) architecture. The ESPRIT server hosts the ESPRIT software and performs the bulk of the processing. It also stores certified models and entities, and basic scenario information. The clients are connected to servers by assigned port numbers. Multiple RMT spirited/solid pairs can be running simultaneously in the server machine with clients connecting to each, via their unique port assignments. Conversely, multiple clients may be connected to the same RMT spirited/solid pairs by sharing the same port assignment. Each one of these clients must connect to a server with spirited and solid running in order to load and run a scenario.

Spirited and solid are server-side executables and are hosted on the RMT server in the bin directory. TDF can be hosted on either a server/master computer and/or individual PCs.

During planning, the server is commanded by the clients to store or download models, scenarios, etc. During scenario execution, the server interprets the scenario and provides simulated radar detection and tracking messages to the network. This distributed network architecture promotes stand-alone planning, collaborative planning, isolated scenario preview, as well as full participant scenario rehearsal and event monitoring.

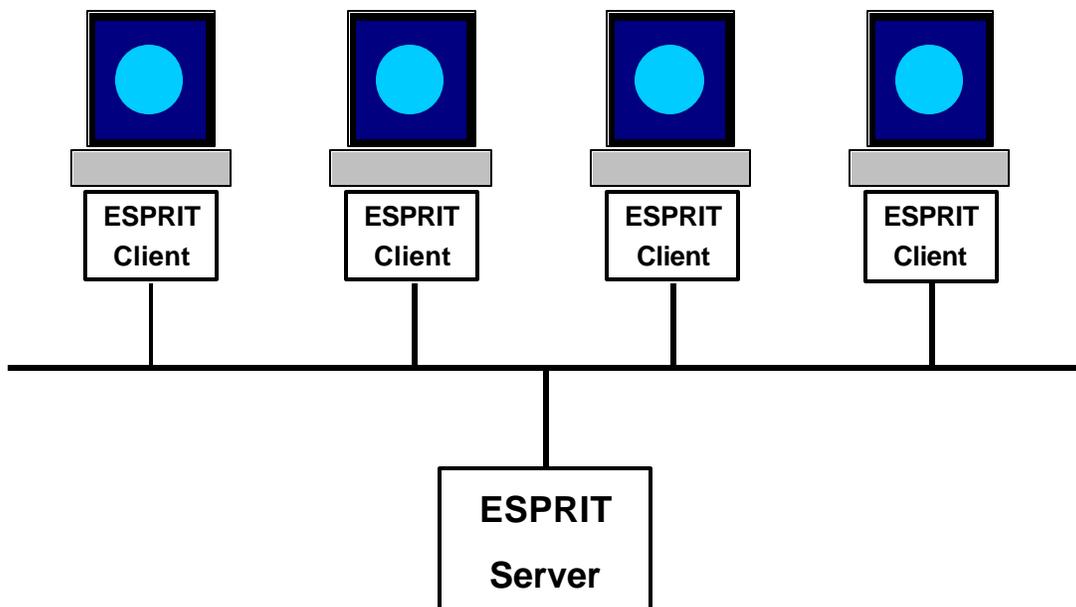


Figure 2. ESPRIT Client/Server Architecture

2.2 Scenario Planning

Scenario planning consists of two distinct phases. Pre-planning is the creation of models and entities. Scenario creation is the selection and placement of previously created entities, and the scripting of their behaviors to create a scenario.

2.2.1 Scenario Pre-Planning

The pre-planning phase of scenario creation involves creating the models and entities from which a scenario can be constructed. Models are first built to define the performance and limitations of different types of sensors and vehicles. Model data is used primarily to constrain sensor and vehicle performance to realistic parameters.

Entities are created as specific instances of appropriate model types and therefore represent the objects with which a scenario can be created. For example, Queen 1 is a sensor entity created from the MPS 25 radar model. It extends the model definition by applying the model to an entity with a specific name and a position. Similarly, CG 53 is a vehicle entity created from the Aegis Cruiser model. Thus, the vehicle entity is a named instance of the model whose position and behavior will be defined during scenario creation.

The operator controls and inputs for the creation and maintenance of vehicle and sensor models are described in detail in Section 5 Pre-Planning.

2.2.2 Scenario Creation

Scenarios are created by opening a new or previously-saved scenario file, adding entities to the file, describing the behavior of the entities, time-synchronizing the entity behaviors to properly depict the event objectives, and saving the work as a named file.

2.3 Scenario Execution

Scenario Execution is the action of playing back a previously scripted scenario. Various modes of playback are provided to meet the different requirements of scenario review, event rehearsal and event monitoring.

2.3.1 Scenario Preview

Scenario preview allows the user to review the scenario plan, including vehicle trajectories and synchronization.

Scenario preview can be started from any selected scenario time, and can be executed at up to 8 times real-time speed. Only planned vehicle positions versus time are generated during scenario preview. No target or radar detection messages are generated.

2.3.2 Local Execution (Rehearsal)

Local execution is the rehearsal mode of ESPRIT. In addition to the planned vehicle positions, simulated radar tracking or radar detection messages are generated.

2.3.3 Event Monitoring

Event monitoring is typically accomplished by executing ESPRIT in the Preview mode while observing actual events. Graphs, reports, and real-time analysis plots allow for qualitative and quantitative comparison of the plan to the actual event.

3 *ESPRIT Display*

The top-level ESPRIT display, shown in Figure 3, consists of several areas:

- The **menu bar**, at the top, lists the available Tactical Display Framework and ESPRIT-specific options
- The **main display** (also known as the **TacSit**), in the middle, displays scenario information along with user-selectable vehicular dynamic symbols overlaid on high-resolution, digitized map products.
- The **classification banner**, framing the TacSit, displays the current environment classification. This is defined as the higher of the current track database classification and the classification of any opened scenario.
- The **embedded track list**, on the right side, provides a filterable inventory of all the tracks in the system.
- The **control bar** along the bottom provides quick access to a variety of mini-displays and commands.

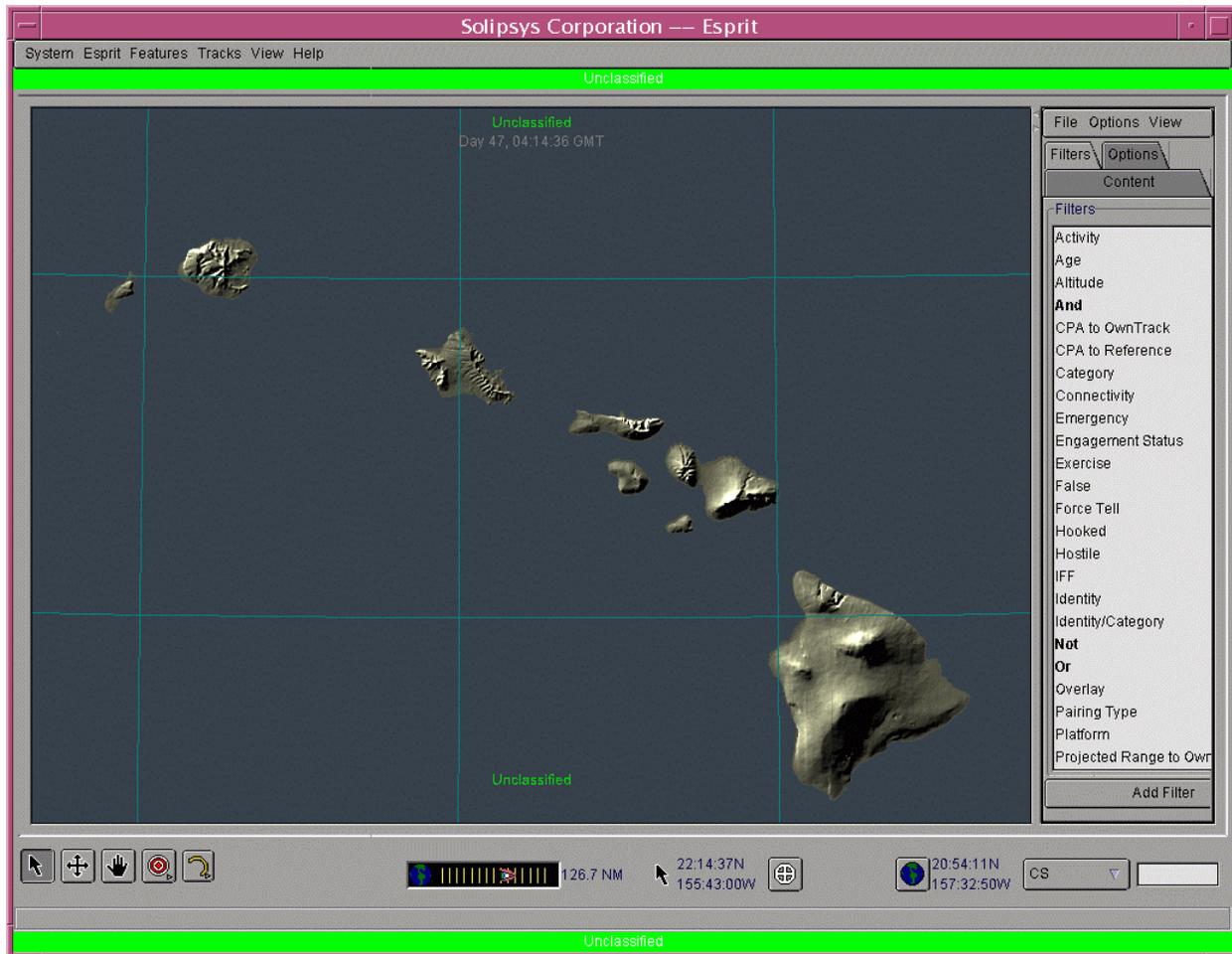


Figure 3.. ESPRIT Main Display

4 Capabilities

Scenario planning tools include capabilities for:

- Building sensor and vehicle models
- Instantiating models to create vehicle and sensor entities
- Defining sensor and vehicle behavior within a scenario
- Synchronizing vehicle trajectories within a scenario
- Scenario evaluation
- Executing the scenario for review and rehearsal
- Monitoring the plan versus the actual event

For purposes of this manual, scenario planning is divided into the pre-planning and planning phases.

5 *Pre-Planning*

Scenario pre-planning consists of those user actions required to create sensor and vehicle models and entities. Models and entities are the specific user assets that can be used repeatedly in subsequent scenario generation. Models are defined in order to constrain the performance of a sensor or vehicle. Entities are specific applications of a model to be used subsequently in scenario definition. For example, “Queen 8” is constructed from the MPS 25 radar model by applying a call sign and location to the model.

5.1 Model Creation

To create a model, select **Esprit** → **Models** → **Sensor Model** or **Esprit** → **Models** → **Vehicle Model** from the TDF menu bar. Figure 4 shows the menu selection.

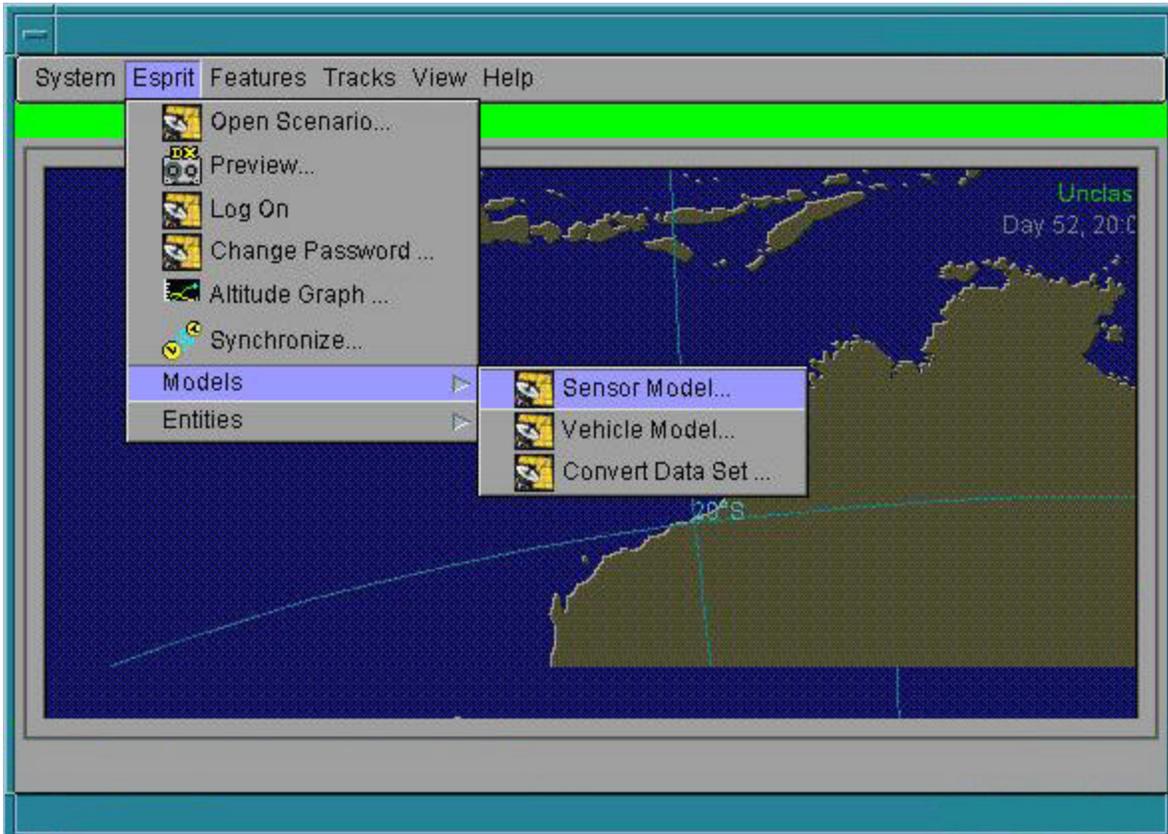


Figure 4. Sensor Model Menu Selection

5.1.1 Sensor Models

Sensor models are used to characterize sensor performance parameters. These parameters are primarily used to schedule sensor detections and to determine probability of detection.

5.1.1.1 Sensor Model Chooser

Selecting **Esprit** → **Models** → **Sensor Model** will present the user with the Sensor Model Chooser panel (Figure 5). To enter a new model, enter the new model name in the text box and select **New**. To modify a current model, highlight the desired model name and select **Open**. Table 1 describes each entry field of the Sensor Model Chooser.

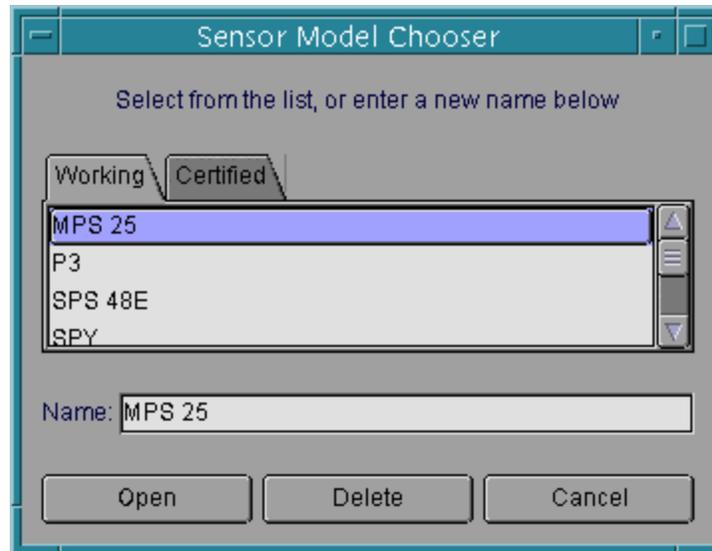


Figure 5. Sensor Model Chooser Panel

Table 1. Sensor Model Chooser Field Description

Field	Entry	Description
Certified/Working Tabs	Select appropriate tab	Displays sensor models from either the Certified or Working directories.
Name	Text	Enter sensor model name either by selecting a sensor model name from the provided list or by typing in a new sensor model name.
<i>Action Buttons</i>		
New/Open		Use either to enter a new model or to open an existing model for viewing or modification. If a current model name is selected, this button reads Open ; if a new model name is entered, it reads New .
Delete		Deletes the currently selected model.
Cancel		Closes the Sensor Model Chooser panel.

5.1.1.2 Sensor Model Editor

The Sensor Model Editor Panel, illustrated in Figure 6, allows the user to enter all pertinent sensor model parameters. The Sensor Model Editor is used to enter sensor performance characteristics, which are used by ESPRIT to simulate periodic sensor/target detections and to calculate sensor to target signal-to-noise ratio.

Hint: The RF loop gain entry allows a shortcut to the entry of the multitude of transmitter and receiver values used to calculate signal-to-noise. If the RF loop gain box is checked, entry of the transmitter/receiver information is unnecessary and will be denied.

Many types of radar have multiple modes of operation, which significantly alter the radar's waveform and performance characteristics. ESPRIT supports the entry of multiple radar modes of operation via the Add Op Mode selection.

Table 2 describes the entries for each of the fields of the Sensor Model Editor panel (depicted in Figure 6).

The image shows a software window titled "Sensor Model Editor". At the top, there is a "File" menu. Below it, the "Model Name" is set to "MPS 25" and the "Type" is set to "Tracking". An "Add Op Mode" button is visible. The main area is divided into several sections:

- Mode Tag:** "Mode 1"
- RF Loop Gain:** A checkbox labeled "Use RF Loop Gain" is unchecked. Below it, the "RF Loop Gain" is set to "5000" dB.
- Accuracy:**
 - Range: 0.002 DM
 - Azimuth: 0.0001 rad
 - Doppler: 0 m/s
 - Elevation: 0.0001 rad
- Period:**
 - Update: 500 ms
 - Off Track: 2 s
 - Drop Track: 4 s
- General:**
 - Max Targets: 1
 - Power: 10000 W
 - Transmit Gain: 1000 dB
 - Receive Gain: 1000 dB
 - Noise Figure: 10 dB
 - System Loss: 5 dB
 - Processing Gain: 100 dB
 - Pulsewidth: 0 ms
 - Frequency: C Band: 4 - 8 GHz
 - Max Range: 2000 m (a dropdown menu is open showing options: m, km, DM, NM, mi)
 - Max Bearing Rate: 3.1400 rad/s
 - Max Bearing Accel: 3.1400 rad/s²
 - Max Elevation Rate: 179.909 deg/s
 - Max Elevation Accel: 179.909 deg/s²

Figure 6. Sensor Model Editor Panel

Table 2. Sensor Model Editor Field Description

Field	Entry	Description
Model Name	None	This field is automatically forwarded from the Sensor Chooser panel.
Type	Tracking, Surveillance, Beacon Transmitter, Beacon Receiver, Telemetry Transmitter, Telemetry Receiver, ITCS Transmitter, ITCS Receiver	Type is selected from the provided menu.
Add Op Mode	Control Button	Permits entry of a new radar operational mode
Mode Tag	Text	Unique sensor mode name.
RF Loop Gain	Decimal Integer	RF loop gain value (dB).
Use RF Loop Gain	Check Box	When checked, indicates user preference to enter RF Loop Gain value instead of Radar power, Transmit Gain, Receiver Gain, Noise Figure, System Loss, and Processing Gain.
<i>Accuracy</i>		
Range	m, km, radar mi, nautical miles, statute miles	Sensor measurement accuracy in range.
Azimuth	deg or rad	Sensor Measurement accuracy in azimuth
Doppler	ft/s, m/s, km/hr, radar mi/hr, knots, statute miles/hr, mach	Sensor Measurement accuracy in range rate.
Elevation	deg or rad	Sensor Measurement accuracy in elevation.

Field	Entry	Description
<i>Period</i>		
Update Period	sec, ms	Time period between radar detects of a stationary target
Off-Track Period *	sec, ms	Time period without detection to declare the sensor is off-track
Drop-Track *	sec, ms	Time period without detection to declare the sensor has lost-track
<i>General</i>		
Max Targets	Decimal integer	Maximum number of targets that can be simultaneously tracked by the sensor.
Power	W or kW	Transmitter power
Transmit Gain	dB	Gain figure of merit for the transmitter
Receive Gain	dB	Gain figure of merit for the receiver
Noise Figure	dB	Loss due to noise
System Loss	dB	Loss due to system noise
Processing Gain	dB	Gain figure of merit for processing
Frequency	L, S, C, X, Ku, K, Ka band	Select type from the provided menu.
Max Range	m, km, radar mi, nautical mi, statute miles	Maximum sensor range coverage
Max Bearing Rate	deg/sec or rad/sec	Maximum allowed rotation in bearing
Max Bearing Accel	deg/sec ² or rad/sec ²	Maximum allowed acceleration in bearing
Max Elevation Rate	deg/sec or rad/sec	Maximum allowed rotation in elevation
Max Elevation Accel	deg/sec ² or rad/sec ²	Maximum allowed acceleration in elevation

* The marked entries are application-specific; their values will not affect ESPRIT's performance.

Hint: If signal-to-noise statistics are unimportant, or if all the various Transmitter/Receiver parameters are not known, enter a high loop gain value (250 dB) to assure radar detections.

5.1.1.3 Saving the Sensor Model

To save the sensor model, select **File** → **Save** on the Sensor Model Editor panel. The model will be stored in the sensor model working directory, and will be accessible for subsequent sensor model modification or sensor entity creation.

5.1.2 Vehicle Models

Vehicle models are used to characterize vehicle performance parameters. These parameters are primarily used to provide default trajectory parameters and to constrain vehicle performance during trajectory definition.

5.1.2.1 Vehicle Model Chooser

The **Model** → **Vehicle** selection (Figure 7) presents the user with the Vehicle Model Chooser panel (Figure 8). To enter a new vehicle model, enter the new model name in the text box and select **New**. To modify a current model, highlight the desired model name and select **Open**.

Table 3 describes the entries for each of the fields of the Vehicle Model Chooser.

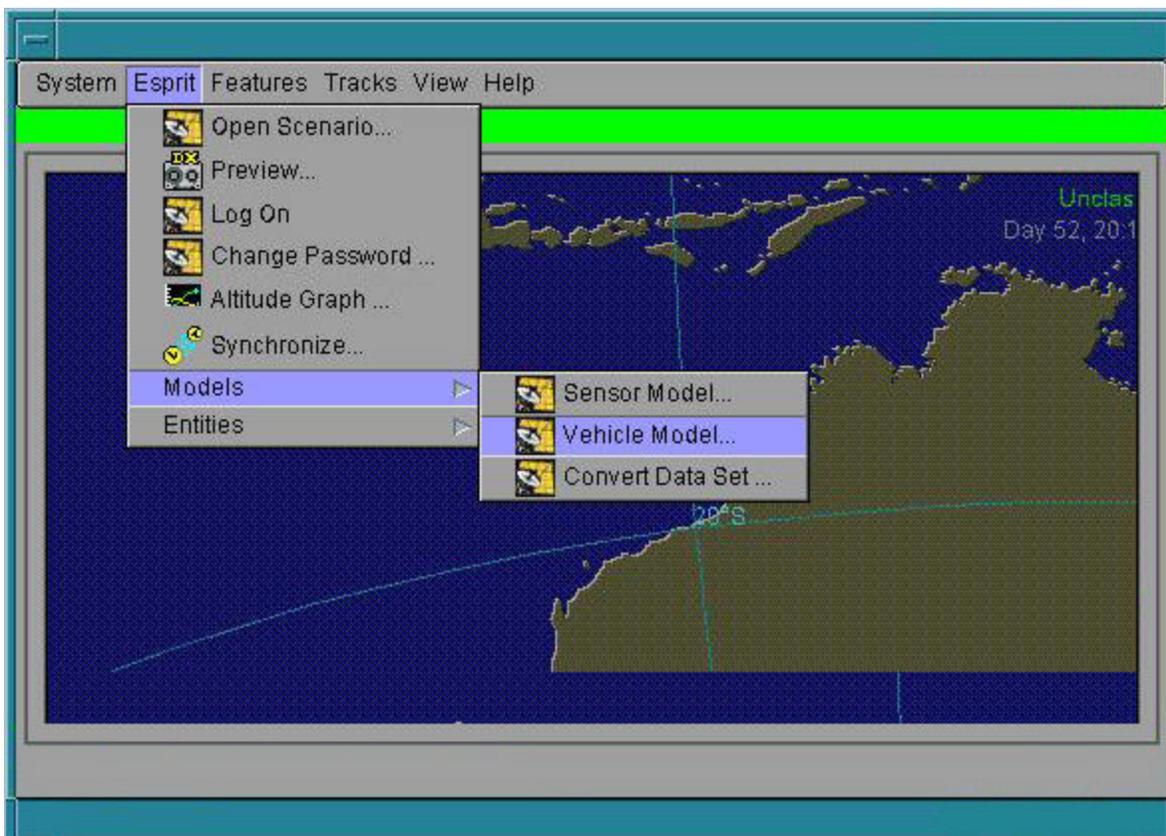


Figure 7. Vehicle Model Menu Selection

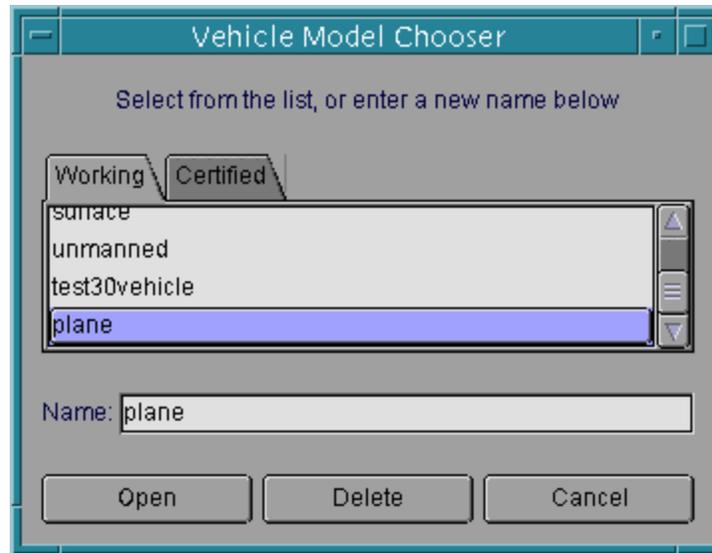


Figure 8. Vehicle Model Chooser Panel

Table 3. Vehicle Model Chooser Field Description

Field	Entry	Description
Certified/Working Tabs	Select appropriate tab	Displays vehicle models from either the Certified or Working directories.
Name	Text	Enter the vehicle model name either by selecting a vehicle model name from the provided list or by typing in a new vehicle name.
<i>Action Buttons</i>		
New/Open		Used either to enter a new model or to open an existing model for viewing or modification. If a current model name is selected, this button reads Open ; if a new model name is entered, it reads New .
Delete		Deletes the currently selected model.
Cancel		Closes the Vehicle Model Chooser panel.

5.1.2.2 Vehicle Model Editor

The Vehicle Model Editor panel (Figure 9) allows the user to enter all pertinent vehicle model parameters.

The screenshot shows the 'Vehicle Model Editor' window with the following parameters:

- File:** Model Name: plane, Type: ABT (Manned), PMRF Veh Type: 0
- Cross Section:** Head-On: 1 m², Side Aspect: 100 m²
- Performance:**
 - Minimum:** Speed: 100 knot, Dive Rate: 0 ft/s, Climb Rate: 0 ft/s, Curve Radius: 0.05 NM, Fuel Rate: 0 lbs/m
 - Nominal:** Speed: 400 knot, Dive Rate: 0 ft/s, Climb Rate: 0 ft/s, Maneuver: 2.332 g's, Bank Angle: 1.166 rad, Curve Radius: 1 NM, Fuel Rate: 10 lbs/m
 - Maximum:** Speed: 1000 knot, Dive Rate: 3280.84 ft/s, Climb Rate: 0 ft/s, Maneuver: 5 g's, Fuel Rate: 100 lbs/m
- Other:** Fuel Capacity: 10000 lbs, Dry Weight: 30000 lbs

A unit selection dropdown menu is open over the 'Nominal Speed' field, showing options: ft/s (selected), m/s, mph, km/h, DM/h, knot, and mach.

Figure 9. Vehicle Model Editor Panel

Table 4 describes the entries for each of the fields of the Vehicle Model Editor.

Hint: The entries in the Performance Minimum/Nominal/Maximum columns are not exactly the same. For example, dive and climb rates are considered opposites, so a maximum dive rate is equivalent to the minimum climb rate. Nominal climb and dive rates are considered to be zero. If this presentation does not appear flexible enough to support your application, you will find that climb and dive rates can be expressed exactly when placing waypoints, within the constraints implied by the vehicle model maximums.

Table 4. Vehicle Model Editor Field Description

Field	5.1.2.2.1 Entry	5.1.2.2.2 Description
Model Name	None	Automatically forwarded from Vehicle Model Chooser panel
Type	User Selection	The pull-down menu allows the user to select the following vehicle types: Surface, Air Breathing Target (ABT) (Manned), ABT (Unmanned).
Cross Section	m ²	Head-on and side aspects of the vehicle
<i>Performance</i>	<i>Minimum</i> <i>Nominal</i> <i>Maximum</i>	Performance entries are used to specify Minimum, Nominal, and Maximum for the vehicle performance parameters.
Speed	ft/s, m/s, km/hr, mi/hr, radar mi/hr, knots, mach	Minimum vehicle speed
Dive Rate	ft/sec, m/s, km/hr, mi/hr, radar mi/hr, knots, mach; <i>Maximum</i> only	Minimum Dive Rate is fixed to 0 mph
Climb Rate	ft/sec, m/s, km/hr, mi/hr, radar mi/hr, knots, mach; <i>Maximum</i> only	Minimum Climb Rate is fixed to 0 mph
Maneuver	g's	Acceleration
Bank Angle	deg, rad	
Curve Radius	m, km, radar mi, nautical mi, statute mi	Minimum radius of curvature during a maneuver
Fuel Rate	lbs/m, kg/m, gal/hr, kg/hr	Minimum fuel consumption rate

Hint: Model performance values do not have to reflect vehicle performance extremes, but can be used, instead, to reflect planning constraints dictated by safety concerns.

5.1.2.3 Saving the Vehicle Model

To save the vehicle model, select **File** → **Save** (Figure 10) from the Vehicle Model Editor menu. The model will be stored in the vehicle model working directory, and will be accessible for subsequent vehicle model modification or vehicle entity creation.

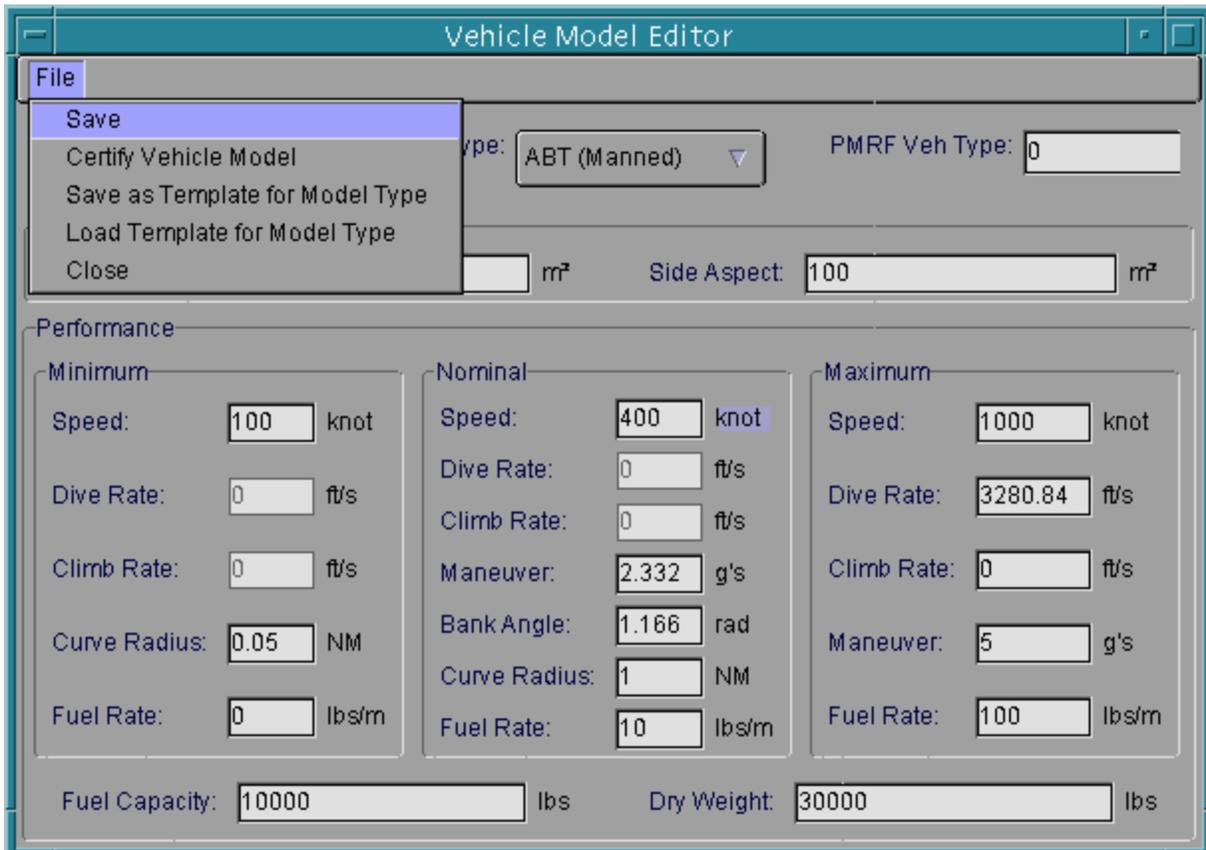


Figure 10. Saving the Vehicle Model

5.1.3 TBM and Interceptor Models

ESPRIT allows the user to enter Tactical Ballistic Missile (TBM) and Interceptor models in addition to the standard vehicle models. These models are imported into a scenario from predetermined data sets that describe their trajectory. Unlike the standard vehicle models, which are entered directly into the scenario and can be easily altered by the user, TBM and Interceptor models can only be modified in limited ways by the user once they have been imported.

5.1.3.1 TBM and Interceptor Model Creation

To create a new TBM model, first locate or create a data set that conforms to the conventions shown in Table 5. Then open the **Esprit** → **Models** → **Convert Data Set** menu; this will bring the Dataset Conversion panel forward. In the Dataset Files menu, select the data set to be converted. Set the *Vehicle Type* to TBM or Interceptor as appropriate and then select **Convert**.

The text “Conversion Completed” will appear at the bottom of the panel when the conversion has finished successfully.

Table 5. TBM Data Set Format

Header	Data Format
Time	Time (s)
Xecfc	X position (m; geocentric coordinates)
Yecfc	Y position (m; geocentric coordinates)
Zecfc	Z position (m; geocentric coordinates)
Xecfcdt	X velocity (m/s)
Yecfcdt	Y velocity (m/s)
Zecfcdt	Z velocity (m/s)
Xecfcdt2	X acceleration (m/s ²)
Yecfcdt2	Y acceleration (m/s ²)
Zecfcdt2	Z acceleration (m/s ²)
Pitch	Pitch (degrees)
Yaw	Yaw (degrees)
Roll	Roll (degrees)

5.1.4 Sensor Entities

A sensor entity is a specific instance of a sensor model. Once a sensor entity is created, it can be used repeatedly in planning operations.

5.1.4.1 Sensor Entity Creation

To create a sensor entity, select **Esprit** → **Entities** → **Sensor Entity** from the menu bar, as shown in Figure 11.

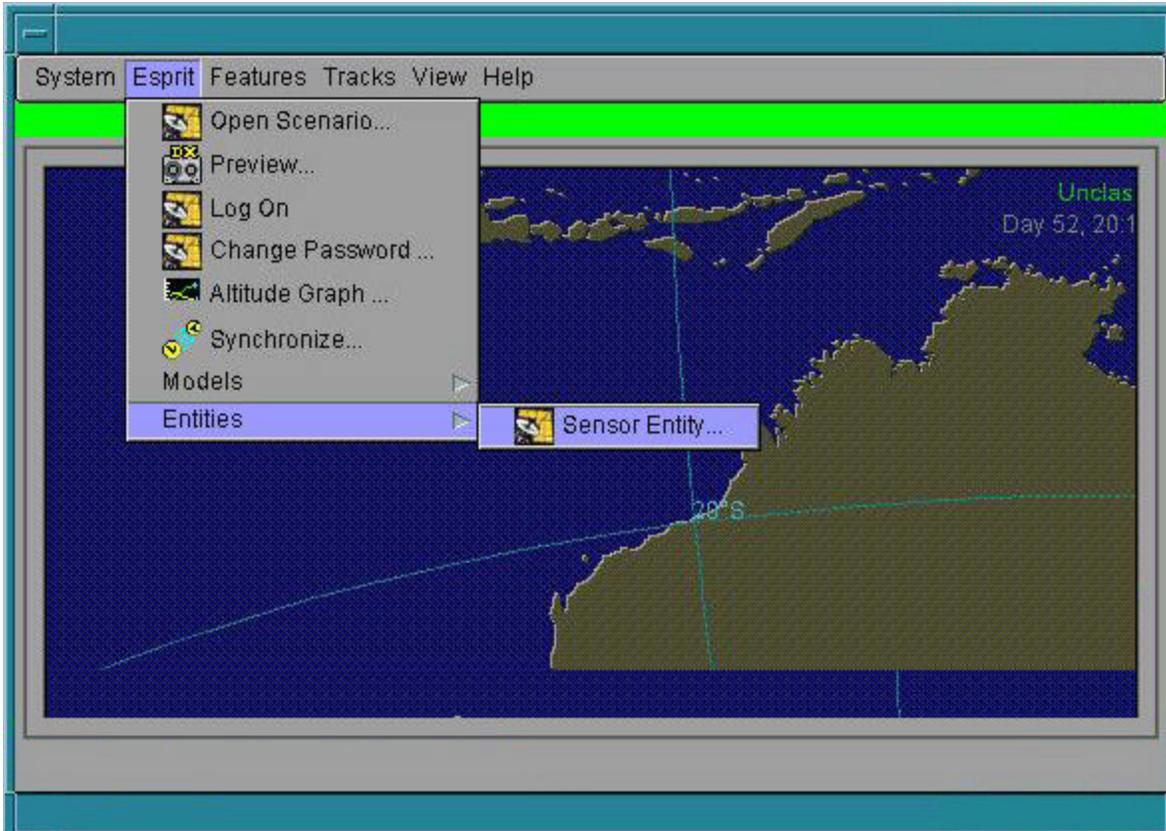


Figure 11. Sensor Entity Creation

5.1.4.2 Sensor Entity Chooser

The Esprit → Entity → Sensor Entity selection presents the user with the Sensor Entity Chooser panel Figure 12. To enter a new entity, enter the new sensor entity name in the text box and select **New**. To make modifications to a current sensor entity, highlight the desired entity name and select **Open**.

Table 6 describes the entries for each of the fields of the Sensor Entity Chooser.

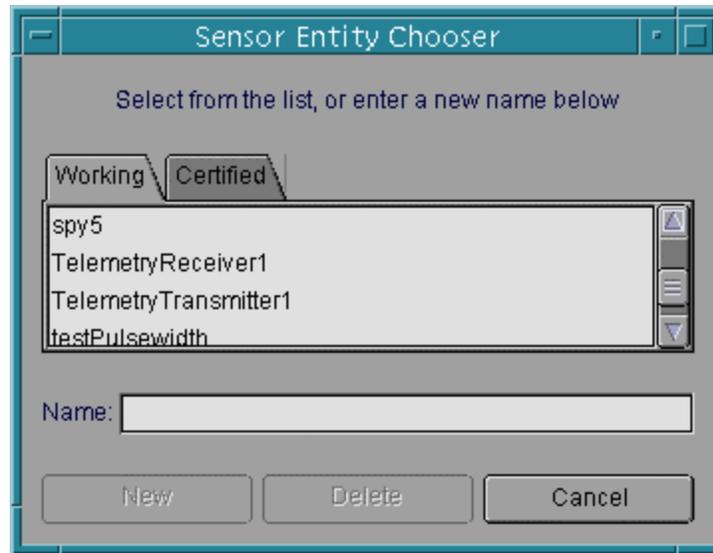


Figure 12. Sensor Entity Chooser

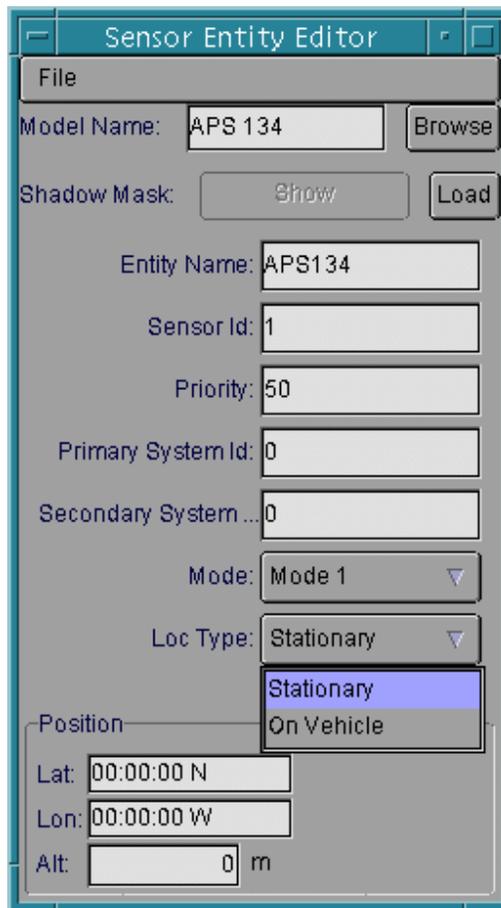
Table 6. Sensor Entity Chooser Field Description

Field	Entry	Description
Certified/Working Tabs	Select appropriate tab	The user selects a tab to view sensor entities from either the Certified or Working directories.
Name	Text	The sensor entity name is entered either by selecting a sensor entity name from the provided list or by typing in a new sensor entity name.
<i>Action Buttons</i>		
New/Open		Used either to enter a new entity or to open an existing entity for viewing or modification. If a current entity name is selected, this button reads Open ; if a new entity name is entered, it reads New .
Delete		Deletes the currently selected entity.
Cancel		Closes the Sensor Entity Chooser panel.

5.1.4.3 Sensor Entity Editor

The Sensor Entity Editor, shown in Figure 13, allows the user to place and name a sensor whose behavior is governed by a previously entered sensor model.

Table 7 describes the entries for each of the fields of the Sensor Entity Editor.



The image shows a screenshot of the "Sensor Entity Editor" dialog box. The dialog has a title bar with the text "Sensor Entity Editor" and standard window controls. Below the title bar is a "File" menu. The main area contains several input fields and buttons:

- Model Name:** A text box containing "APS 134" and a "Browse" button.
- Shadow Mask:** A text box containing "Show" and a "Load" button.
- Entity Name:** A text box containing "APS134".
- Sensor Id:** A text box containing "1".
- Priority:** A text box containing "50".
- Primary System Id:** A text box containing "0".
- Secondary System ...:** A text box containing "0".
- Mode:** A dropdown menu showing "Mode 1".
- Loc Type:** A dropdown menu showing "Stationary". A sub-menu is open, showing "Stationary" (highlighted) and "On Vehicle".
- Position:** A section containing three text boxes: "Lat: 00:00:00 N", "Lon: 00:00:00 W", and "Alt: 0 m".

Figure 13. Sensor Entity Editor

Hint: The location type in the Figure 13 was chosen to be stationary. If a sensor entity is to be placed on a moving platform, change the location type to “On Vehicle” and enter the Vehicle Identifier (VID) in the Position area.

Table 7. Sensor Entity Field Descriptions

Field	Entry	Description
Model Name	None	This field is automatically forwarded from the Sensor Chooser panel.
Entity Name	Alpha-Numeric	Entity Call Sign, i.e. Q1
Sensor Id	Integer	Sensor Identification Number
Priority*	Integer	Assigned Sensor Priority
Primary System Id*	Integer	Identification number of primary reporting system
Secondary System Id*	Integer	Identification number of secondary reporting system
Mode	Pull-down menu entry	Select operating mode from a menu of the modes defined for this sensor
Location Type	Pull-down menu: Stationary; On Vehicle	Select “Stationary” if the sensor has a fixed location. Select “On Vehicle” if the sensor is located on a vehicle.
<i>Stationary</i>	<i>Position</i>	Geodetic position of the sensor.
Latitude	DD:MM:SS	Geodetic latitude.
Longitude	DD:MM:SS	Geodetic longitude.
Altitude	m, kft*	Altitude above mean sea level.
<i>On Vehicle</i>		
VID	Integer	Vehicle Identification

* The marked entries are application-specific; their values will not affect ESPRIT’s performance.

5.1.4.4 Saving the Sensor Entity

To save the sensor entity, select the **Save** button on the Sensor Entity Editor panel.

6 Planning

6.1 Scenario Creation

Scenario creation is the action of using previously defined sensor entities and vehicle models to script the objectives of an event. To accomplish this, the user must:

1. Create vehicle entities by selecting the appropriate vehicle models and describing the vehicles' motion by the placement and editing of waypoints.
2. Add sensor entities to the scenario and assign them, as appropriate, to track the correct VID. This instruction pertains only to tracking sensors. Surveillance sensors will detect all targets within their surveillance volume.
3. Describe the event objectives by synchronizing the vehicle trajectories.

6.1.1 Scenario Selection

Select Esprit → Open Scenario from the menu bar Figure 14 to access the Scenario Chooser.

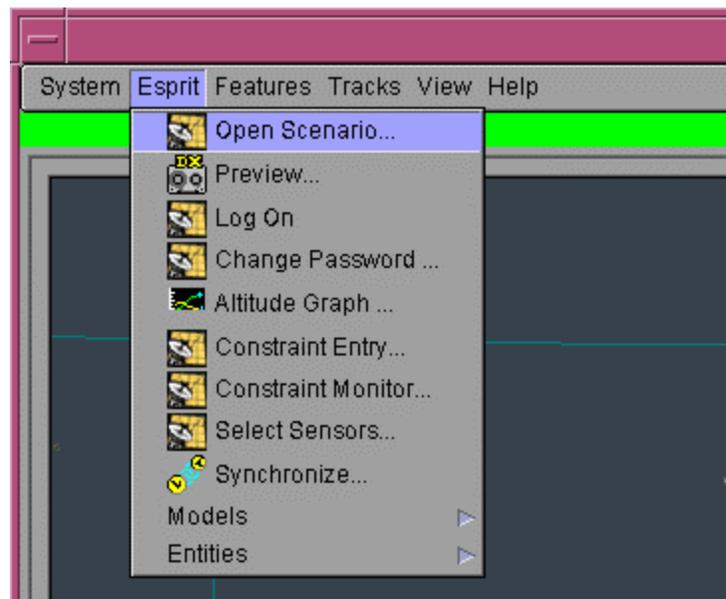


Figure 14. Open Scenario Menu Selection

The Scenario Chooser panel, shown in Figure 15, permits the user either to select a previously entered scenario for modification or to name and enter a new scenario. Upon selection of a scenario for entry or modification, the user is presented with the Scenario Editor panel, Figure 16.

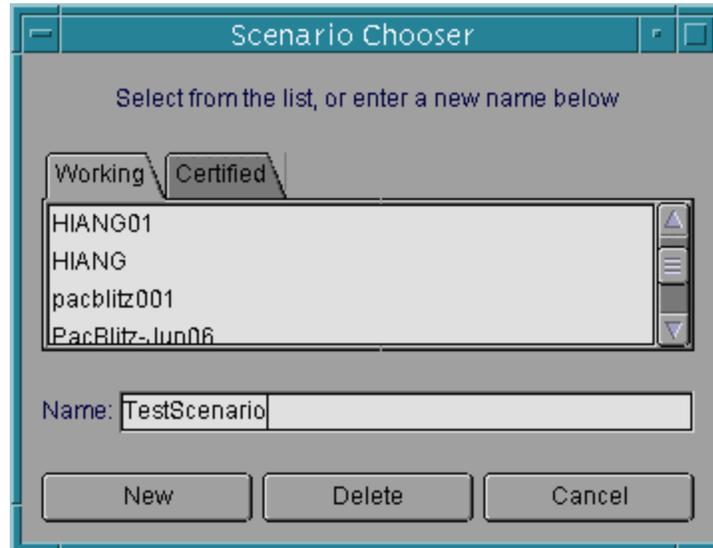


Figure 15. Scenario Chooser Panel

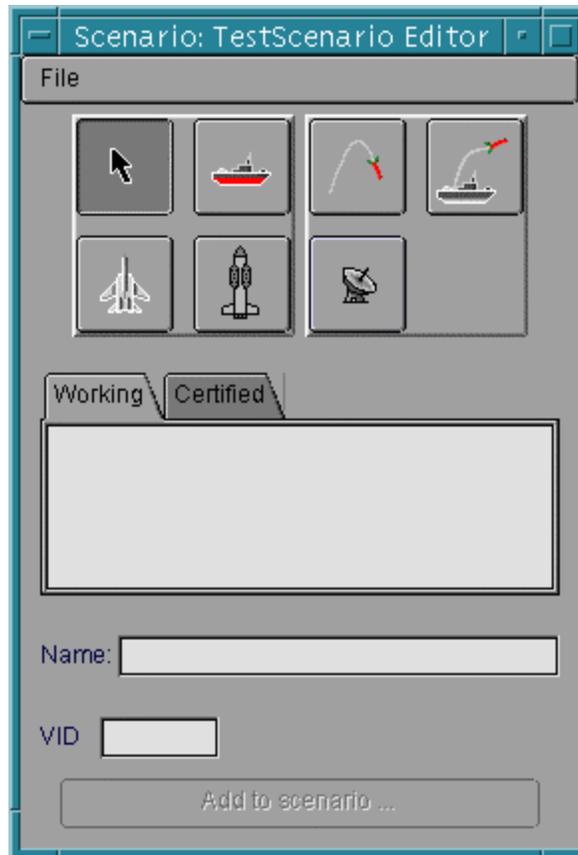


Figure 16. Scenario Editor Panel

Table 8 describes the entries for each of the fields of the Scenario Chooser.

Table 8. Scenario Chooser Field Description

Field	Entry	Description
Certified/Working Tabs	Select appropriate tab	The user selects a tab to view a scenario from either the Certified or Working directories. Note that Certified scenarios cannot be modified.
Name	Text	The scenario name is entered either by selecting a scenario name from the provided list or by typing in a new scenario name.

Field	Entry	Description
New/Open		Used either to enter a new scenario or to open an existing scenario for viewing or modification. If a current scenario name is selected, this button reads Open ; if a new scenario name is entered, it reads New .
Delete		Deletes the currently selected scenario.
Cancel		Closes the Scenario Chooser panel.

6.1.2 Scenario Classification

Each scenario has an associated classification level. During real-time event monitoring, when ESPRIT is processing live tracks, the Classification Banner may not necessarily display the scenario's classification because it reflects the higher of the scenario classification and the classification of the Track Database. To view or change a scenario's classification, right click on the Classification Banner. Selecting the **Set Scenario Classification** menu option will display the Set Scenario Classification panel shown in Figure 17. Alternatively, select the **File** → **Set Scenario Classification** menu option from the Scenario Editor. When the dialog is displayed, the Classification combo box will show the scenario's current classification. The user can choose from the following classification levels: *Unclassified*, *Official Use Only*, *Confidential*, *Secret*, and *Top Secret*. The default classification of a new scenario is *Confidential*.

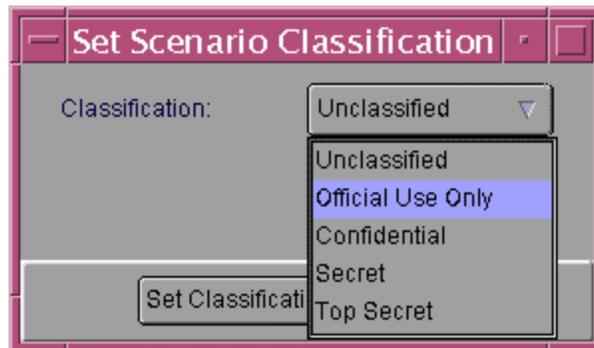


Figure 17. Set Scenario Classification Panel

6.1.3 Scenario Editor

The Scenario Editor allows the user to select sensor entities for assignment to the scenario plan and to use vehicle models to enter vehicle trajectories. Vehicle trajectories can be synchronized to express scenario objectives.

Table 9 describes the entries for each of the fields of the Scenario Editor.

Table 9. Scenario Editor Field Description

Field	Entry	Description
Certified/Working Tabs	Select appropriate icon	The user selects a tab to view a model from either the Certified or Working directories. After selecting the appropriate icon, the vehicle name appears in the text box. Click on text.
Vehicle Identity (VID)	Integer	This unique vehicle identification number is attached to each waypoint.
Add to Scenario	Select Add to Scenario button	Creates initial waypoint for the vehicle, with subsequent clicks creating following waypoints.

6.1.3.1 Vehicle Entities and Trajectory Definitions

Select a vehicle icon from among the icon buttons. Enter the name associated with this particular vehicle entity, assign it a unique VID and select **Add to scenario**. Move the cursor to the TDF display panel. The cursor will become a crossbar, which will enable the placement of waypoints when clicking on the TDF display panel. When a waypoint appears, the Information panel (Figure 18) will appear simultaneously. If the information panel doesn't appear, ensure that the "Bring Forward on Selection" option is selected in the **System** → **Preferences** → **Info** panel. When the vehicle waypoint entry is completed, press **F2** to exit waypoint definition and return to the default mouse pointer. Table 10 describes the entries for each of the fields of the Vehicle Trajectory Information Panel.

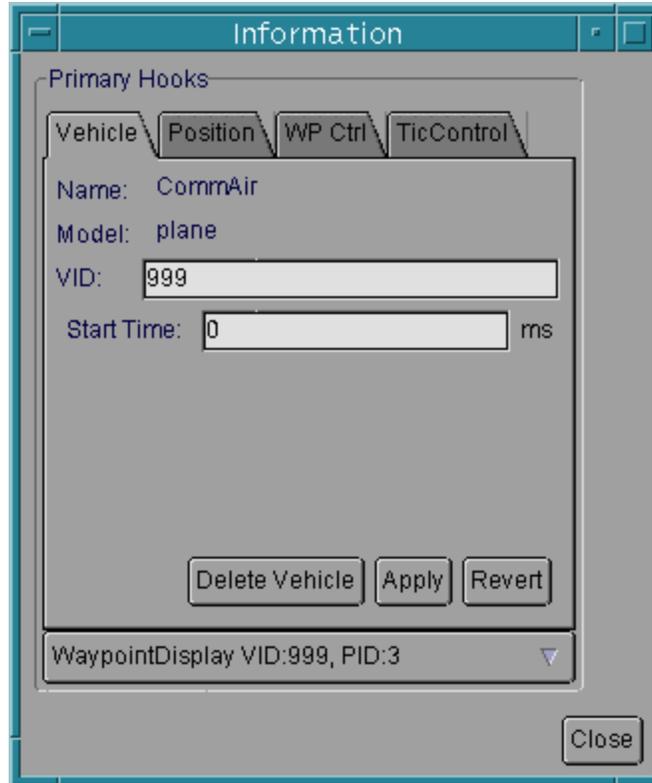


Figure 18. Information Panel

Table 10. Information Panel Field Descriptions

Field	Entry	Description
<i>Vehicle</i>		
VID	Integer	Vehicle identification number.
Start Time	Integer	Explicit time at which vehicle enters the scenario. This will cause any previously set synchronizations to be lost.
<i>Position</i>	<i>Select</i>	Geodetic position of the waypoint.
Lat	N, S	Geodetic latitude (north/south).
Lon	E, W	Geodetic longitude (east/west).
Alt	m	Altitude above mean sea level.

Field	Entry	Description
<i>WPControl</i>		
Annotation	Text	Allows user to enter notes about selected waypoint, which can later be displayed for that waypoint.
Speed	m/s	Speed of the vehicle at the waypoint.
Rad Curvature	rad	Indicates radius of curvature of a turning maneuver initiated at the waypoint.
Climb Rate	m/s	Indicates the rate of climb at the waypoint.
Angle of Ascent	rad	Indicates the angle between the vehicle's trajectory and horizontal.
>>/<<	button	Switches pointer to waypoint placement mode for inserting new successor/predecessor waypoints. (See also pop-up menus, below.)
<i>Tic Control</i>		
Tic Interval	sec	Interval between time tics along the vehicle's trajectory
Tic Annotation Interval	minutes	Interval between time tic annotations
Make T0	Select Make T0	Synchronizes the waypoint to the start of the scenario
T0=0	Select T0=0	Used to set T0 to be the time at which a vehicle first enters the scenario, and adjust all other start times correspondingly. Applies to the entire scenario, not just the selected waypoint.

In planning mode, right-clicking on the waypoint produces another menu. There will be a slight variation in the menus for the first and the last waypoints of a target's trajectory, as shown in Figure 19. The "first waypoint" menu panel offers four options unique to this panel: "Set Orbit", "Stationary Waypoint", "Extend Backward" and "Insert Successor." The "last waypoint" menu panel offers two options unique to this panel: "Extend Forward" and "Insert Predecessor."

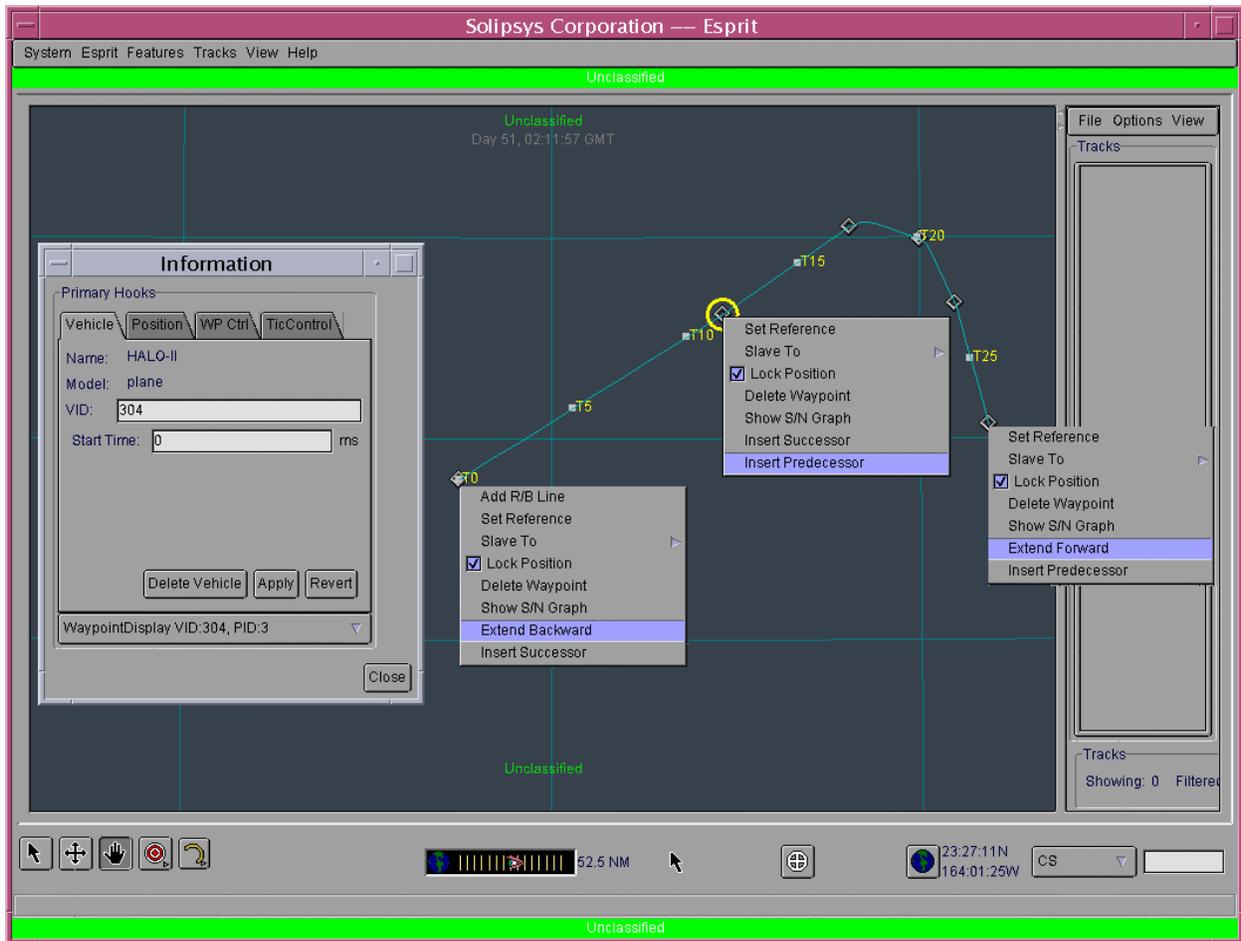


Figure 19. Trajectory with Waypoint Menus

Table 11 describes the entries for each of the selections for the first and last waypoint menus.

Table 11. Menu for First and Last Waypoints

Option	Description
Set Orbit	Enables the user to set an orbital path, in either a counterclockwise or clockwise direction, for a vehicle with a single waypoint. Once an orbital path has been set for the vehicle, the user cannot enter additional waypoints. Additionally, the user cannot extend forward, extend backward, insert predecessor, or successor waypoints while in a non planning mode.
Lock Position	Locks the waypoint position to protect against accidental

Option	Description
	movement during subsequent planning actions.
Stationary Waypoint	Enables the user to specify that a vehicle is stationary. Once a waypoint is set as stationary, the user cannot add additional waypoints. Also, the user cannot extend forward, extend backward, insert predecessor or successor waypoints while in non planning mode.
Delete Waypoint	Deletes selected waypoint.
Show S/N Graph	Produces a signal-to-noise graph for the selected target versus all tracking sensors assigned to track the vehicle and all surveillance sensors.
Extend Backward	Enables waypoint placement immediately after the selected waypoint (applies to first waypoint only).
Insert Successor	Enables waypoint placement immediately after the selected waypoint.
Insert Predecessor	Enables waypoint placement immediately before the selected waypoint.
Extend Forward	Continues placing waypoints in the forward direction of vehicle motion (applies to last waypoint only). This is the default means of placing consecutive waypoints unless an alternative selection has been made.

Hint: Users may choose whether to script waypoints from the beginning forward or from the end backward.

The “Set Orbit” and “Stationary Waypoint” menu options are available only when the user is in the scenario planning mode and has already placed a single waypoint. The “Set Orbit” option permits the user to plot an orbital path for the vehicle. Selecting “Counterclockwise” or “Clockwise” from the “Set Orbit” sub menu sets the direction of the vehicle as shown in Figure 20. The user is presented with an orbital path that can be contracted or expanded as shown in Figure 21. Once the user has set the orbital path for the vehicle, no additional waypoints for that particular vehicle can be inserted. Right-clicking on any of the waypoints in the orbital path presents the user with the option to delete the orbital path and thus the vehicle from the scenario.

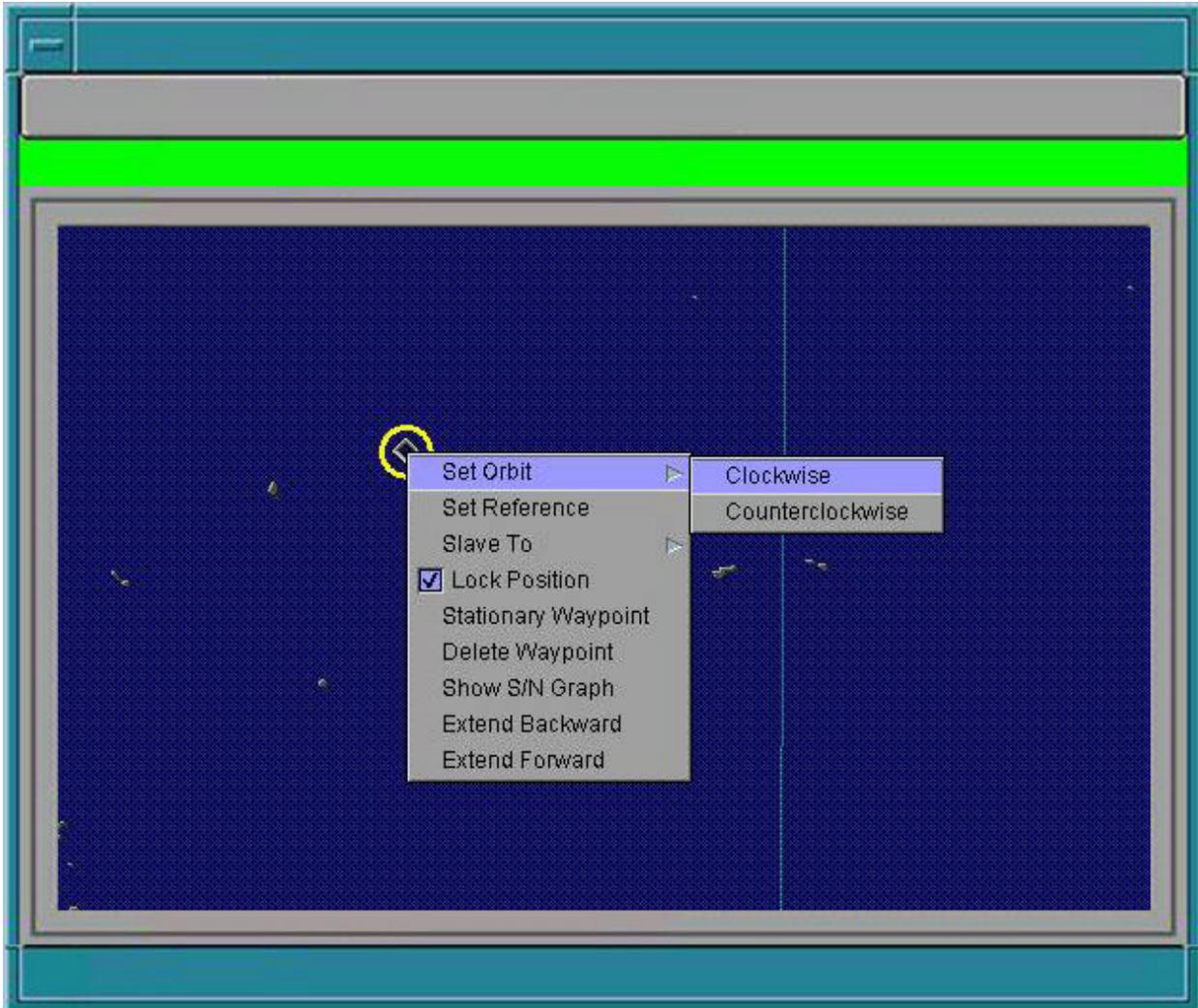


Figure 20. Set Orbit Menu Option

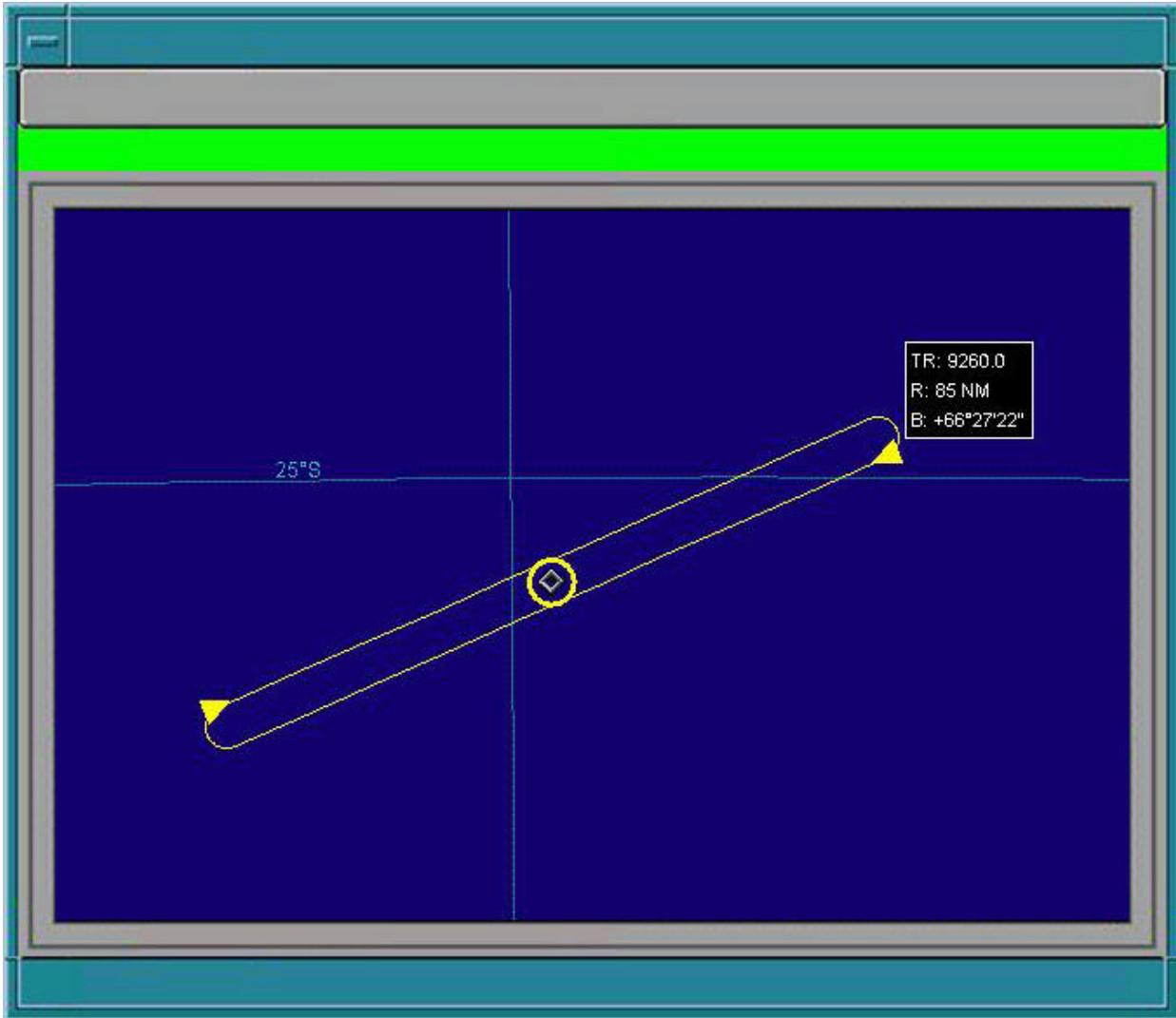


Figure 21. Orbital Trajectory in Clockwise Direction

The “Stationary Waypoint” option allows the user to place a stationary waypoint for a vehicle. Right-clicking on the first waypoint displays a menu with the “Stationary Waypoint” option. Selecting this option delineates the waypoint as stationary and prevents the user from inserting additional waypoints for the vehicle.

Once a waypoint has been delineated as stationary, the user cannot extend forward, extend backward, or insert a successor waypoint.

6.1.3.2 Adding a TBM or Interceptor Vehicle Entity

To add a TBM to a scenario, open the Scenario Editor panel and select the TBM button. To add an Interceptor to a scenario, select the Interceptor button. After selecting the model to add, enter

a vehicle ID, choose **Add to scenario**, and the TBM or Interceptor's trajectory will appear on the scenario. Right-clicking on the trajectory will bring up a pop-up menu, as described in Table 12.

Table 12. TBM Pop-up Menu Options

Option	Description
Add R/B Line	Adds line indicating TBM's current heading and the range from the radar that is tracking it.
Add to Altitude Graph	Adds the TBM's altitude data to a graph of altitude vs. time.
Add Synch Point	Adds a synchronization point.
Modify Model	Allows modification of TBM model.

6.1.3.3 Synchronizing Vehicle Trajectories

Scripting the scenario is accomplished by synchronizing the vehicle trajectories to the scenario timeline. The start of events of interest in the scenario is referred to as **T0**. By default, the first waypoint on each vehicle's trajectory is assigned to T0. To define the start of the scenario, identify the vehicle whose trajectory is to drive the scenario timeline. Typically, T0 is defined by the launch of a TBM or an Interceptor. To add a synchronization point at time of launch, right-click on the first red dot of the trajectory and select **Add Synch Point**. If multiple vehicle trajectories are overlapping, the Add SynchPoint menu will feature options with the VID for each vehicle in the vicinity. Once the new synch point has been added for the desired vehicle, select it, then from the Information Panel select the Tic Control tab, press **Make T0** and then click the **Apply** button. Alternatively, a waypoint on a vehicle's trajectory can be defined as T0 following the same procedure.

The next step is to synchronize the various vehicle trajectories to T0. Hook the T0 Synch Point or Waypoint as a primary track, hook the vehicle's Synch Point or Waypoint as a secondary track, then select **Esprit** → **Synchronize**. From the Synchronization dialog depicted in Figure 22, press **Synchronize** to synchronize the vehicle's trajectory to T0. Checking the **Show Synchronized Lines** checkbox provides visual confirmation that the two points have been synchronized.

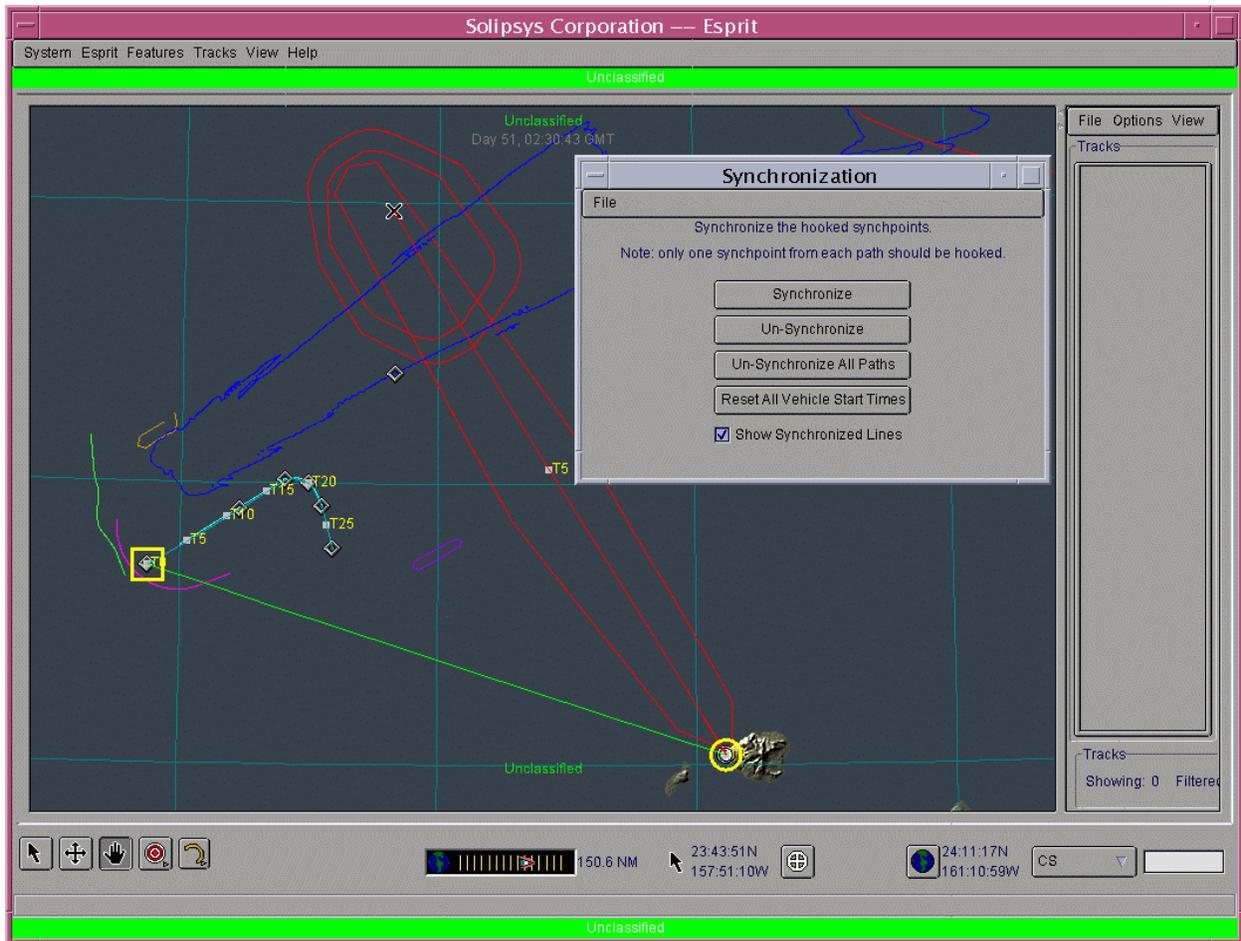


Figure 22. Synchronizing Vehicle Trajectories

6.1.3.4 Sensor Placement and Assignment

To enter a sensor into the scenario, select the sensor icon on the Scenario Editor panel, as shown in Figure 23. Defined sensor entities are displayed in the Working/Certified window. Select the sensor call sign of the sensor to be inserted; the sensor name and sensor identification number (SID) will be displayed in the appropriate windows. Select the **Add To Scenario** button to add the sensor as a participant, and a sensor icon will appear at the chosen sensor entity's location.

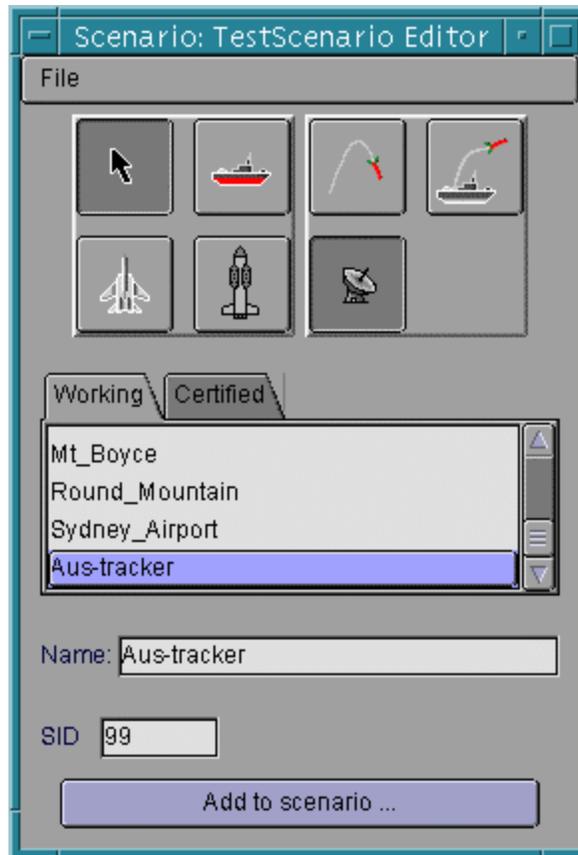


Figure 23. Scenario Editor Panel

There are two ways to assign sensor-tracking responsibility for a tracking sensor. In the first approach, bring up the sensor's information panel either by pressing control-I or by going to the Tracks → Information menu option. An information panel will appear as shown in Figure 24. The second way to display sensor information is by selecting Esprit → Select Sensors. This brings up a **Sensor Selector Application** panel as shown in Figure 25. Selecting the **Assignment** button from the information panel or from the **Sensor Selector Application** panel will bring the **Sensor Assignment** panel (Figure 26) into view. The **Sensor Assignment** panel allows the planner to assign tracking responsibilities to a tracking radar for either discrete segments of the scenario or one assignment for the whole scenario.

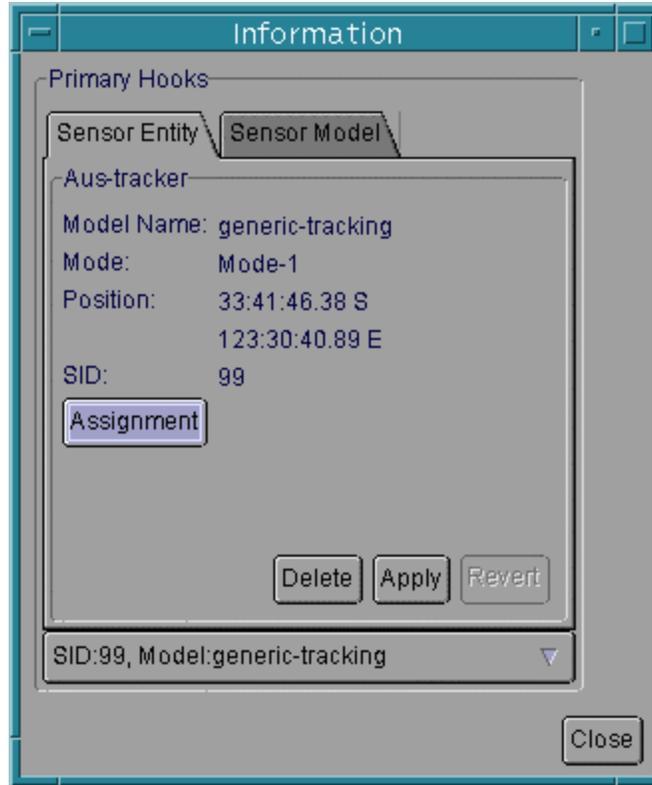


Figure 24. Sensor Information Panel

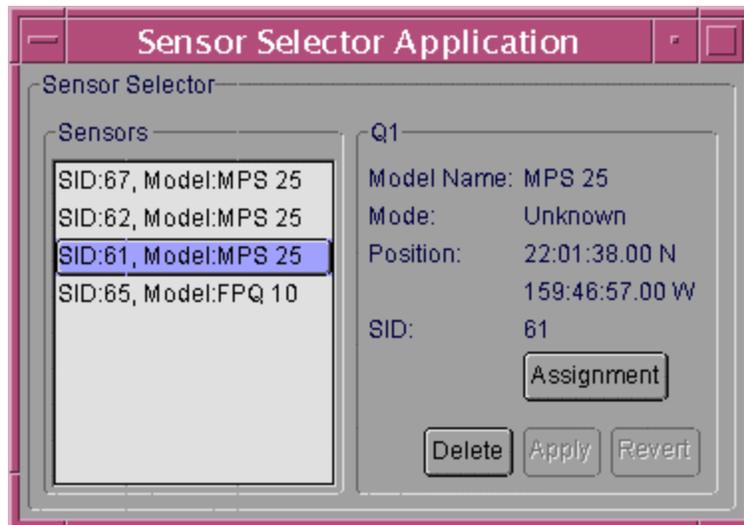


Figure 25. Sensor Selector Application

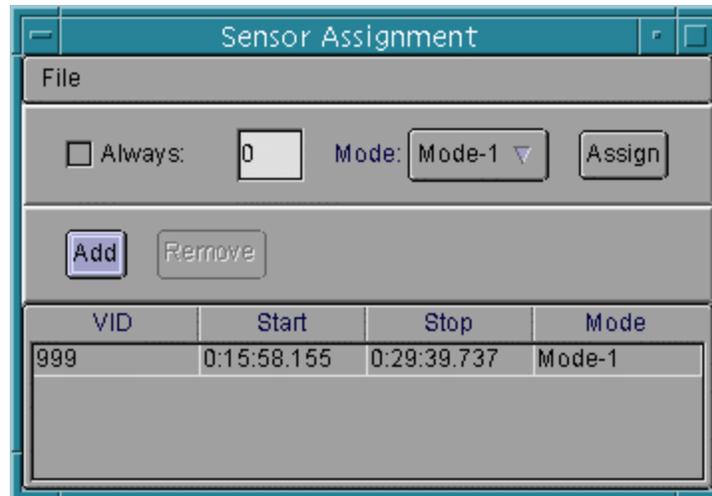


Figure 26. Sensor Assignment Panel

To assign tracking responsibilities for the entire course of a single target, select the **Always:** checkbox and enter the VID of the vehicle to track.

To assign tracking responsibilities for a segment of a track, select the **Add** button; this will bring the Add Sensor Assignment panel (Figure 27) into view. Select the waypoint that represents the starting point of the track assignment segment and select **Set Start**. Then select the waypoint that represents the ending point of the track assignment segment and select **Set Stop**.

After the tracking assignment has been made, save the assignment by selecting the File → Save Assignment and Close menu option on the Sensor Assignment panel.

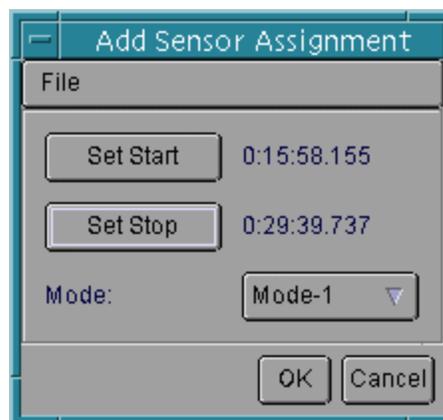


Figure 27. Sensor Assignment: Add Sensor Assignment Panel

6.1.3.5 Constraint Definition

To define constraints (the parameters to which the scenario is restricted), open the Constraint Editor panel (Figure 28) by selecting **Esprit** → **Constraint Entry**. Upon starting the Constraint Editor, any constraints that have been previously defined during this current scenario planning session will automatically load. A panel will appear presenting each constraint type on a separate tab. Each constraint tab contains the fields necessary for defining that particular constraint type. Once the constraint-defining data has been entered into the fields, select *Enter* to save the constraint. The constraint is now saved and displayed in the constraint table, which is also on the active tab. By right clicking on a constraint in the table, that constraint may be deleted. Version 4.0 does not yet support constraint modification in the table.

Constraints are not saved as part of the scenario, rather they are saved in a separate file. To save the constraints defined during a planning session, select **File** → **Save As**, and specify a filename. Selecting **File** → **Open** and choosing a file, will load previously defined constraints into the Constraint Editor.

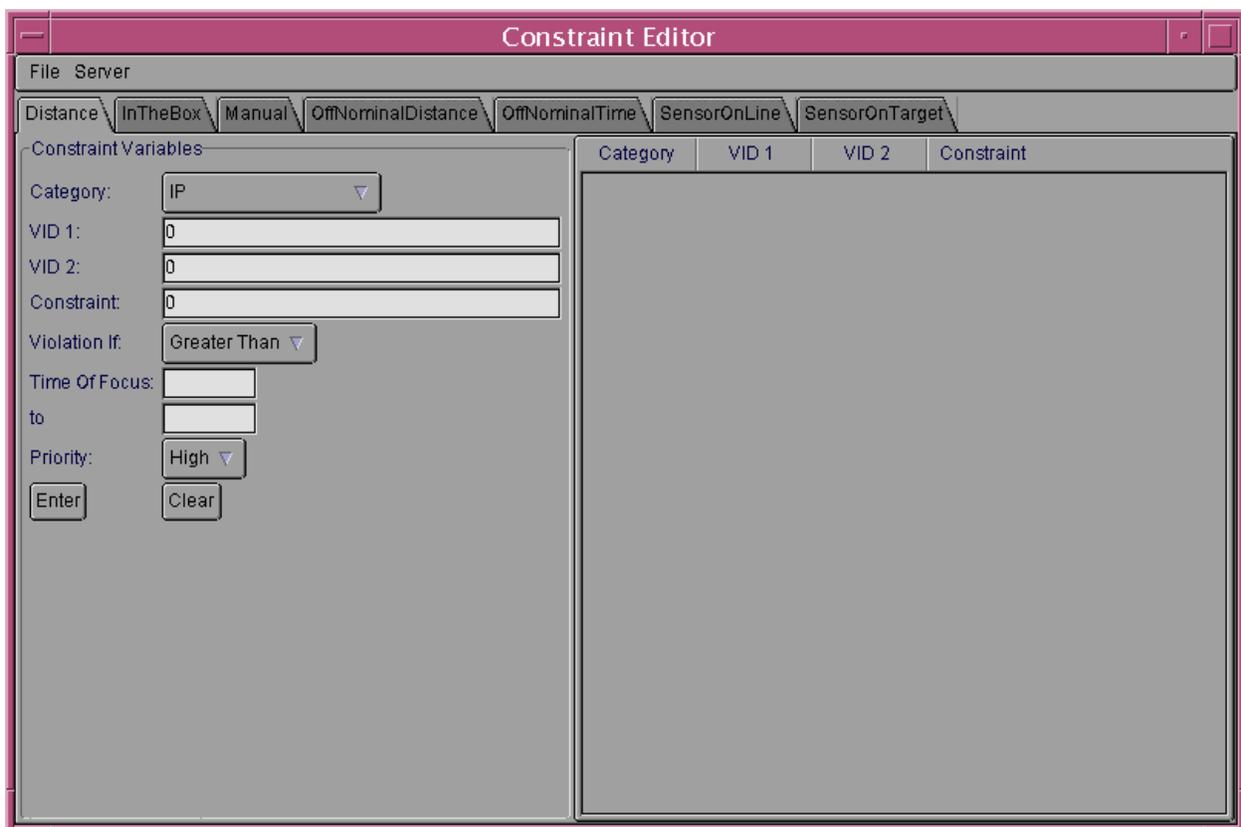


Figure 28. Constraint Editor

6.1.4 Save Scenario

To save a scenario, open the Scenario Editor panel and select **File** → **Save Scenario**. To save a scenario under a new name, select **File** → **Save As...**

6.1.5 Certify Scenario

Certified scenarios are those that have undergone a degree of verification and validation, so that they are protected from modifications. To certify a scenario, open the Scenario Editor panel and select **File** → **Certify Scenario** (Figure 29). If you have not yet logged onto the certification server, you will be prompted to enter the password.

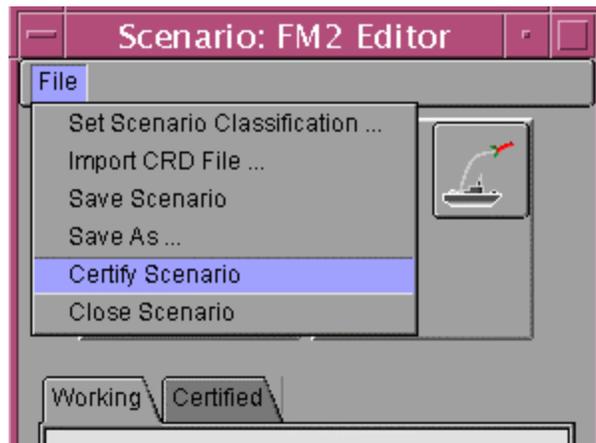


Figure 29. Certify Scenario Menu Selection

To make changes to a certified scenario, first save it under a new name, using **File** → **Save As** option, which will save it in the working directory. After making the desired modifications, re-certify the scenario by selecting **File** → **Certify Scenario**.

6.1.6 Close Scenario

To close the scenario, select **File** → **Close Scenario**. A pop-up box will ask whether to save the scenario. Selecting **Yes** will save the scenario before closing. Selecting **No** will close the scenario without saving changes made since the most recent save.

7 Scenario Execution

Scenario execution encompasses the actions of scenario selection, selecting the mode of execution, selecting the speed of execution, and starting the scenario for viewing.

7.1 Scenario Open

Scenario execution is initiated by selecting **Esprit** → **Open Scenario**, as shown in Figure 30.

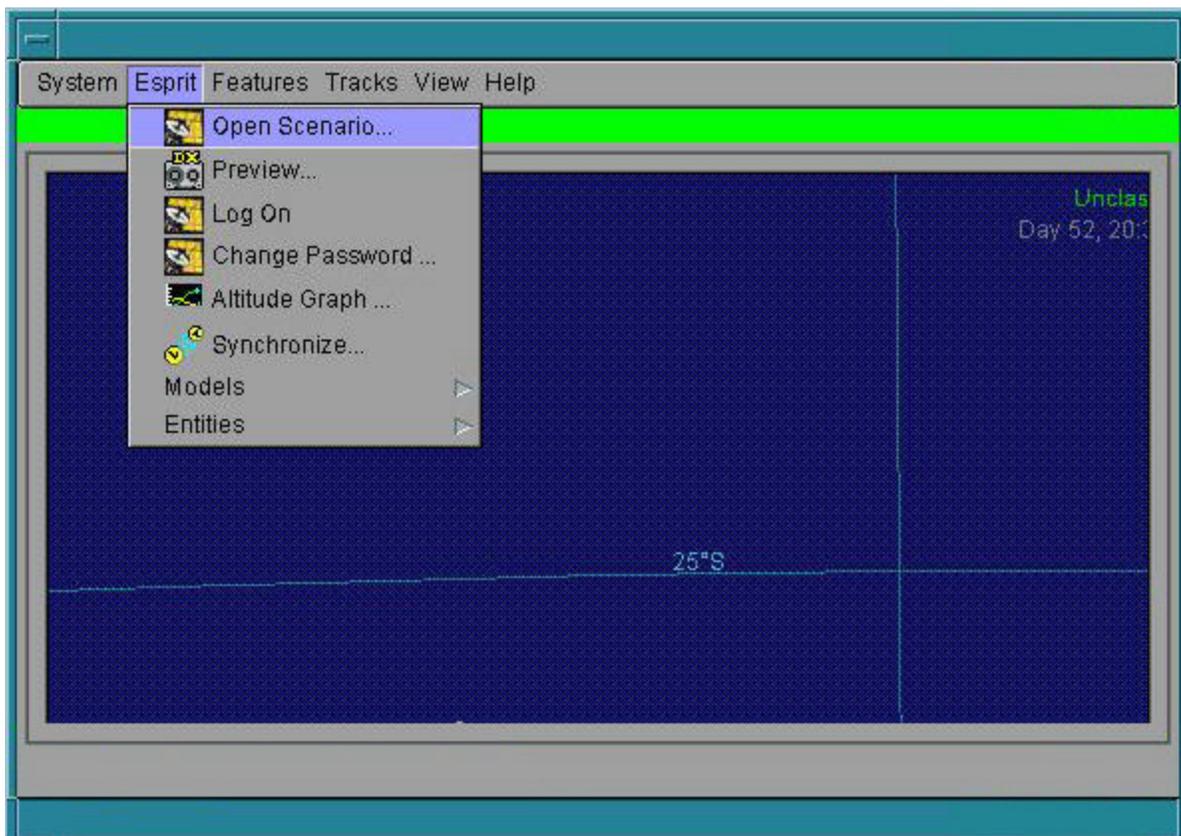


Figure 30. Open Scenario Menu Selection

7.2 Scenario Chooser

The Scenario Chooser panel (Figure 31) permits the user either to select a previously entered scenario for execution or modification, or to name and enter a new scenario. Refer to Table 9 for a description of the Scenario Chooser fields.

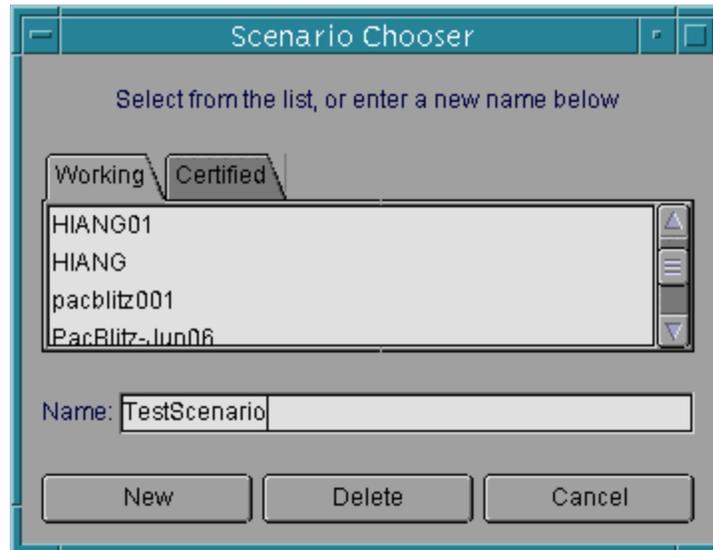


Figure 31. Scenario Chooser Panel

When the selected scenario is opened, the sensor locations, vehicle planned paths, and vehicle waypoints are displayed on the screen, as shown in Figure 32. At this point, the scenario may be either modified (if it is a working scenario) or executed.

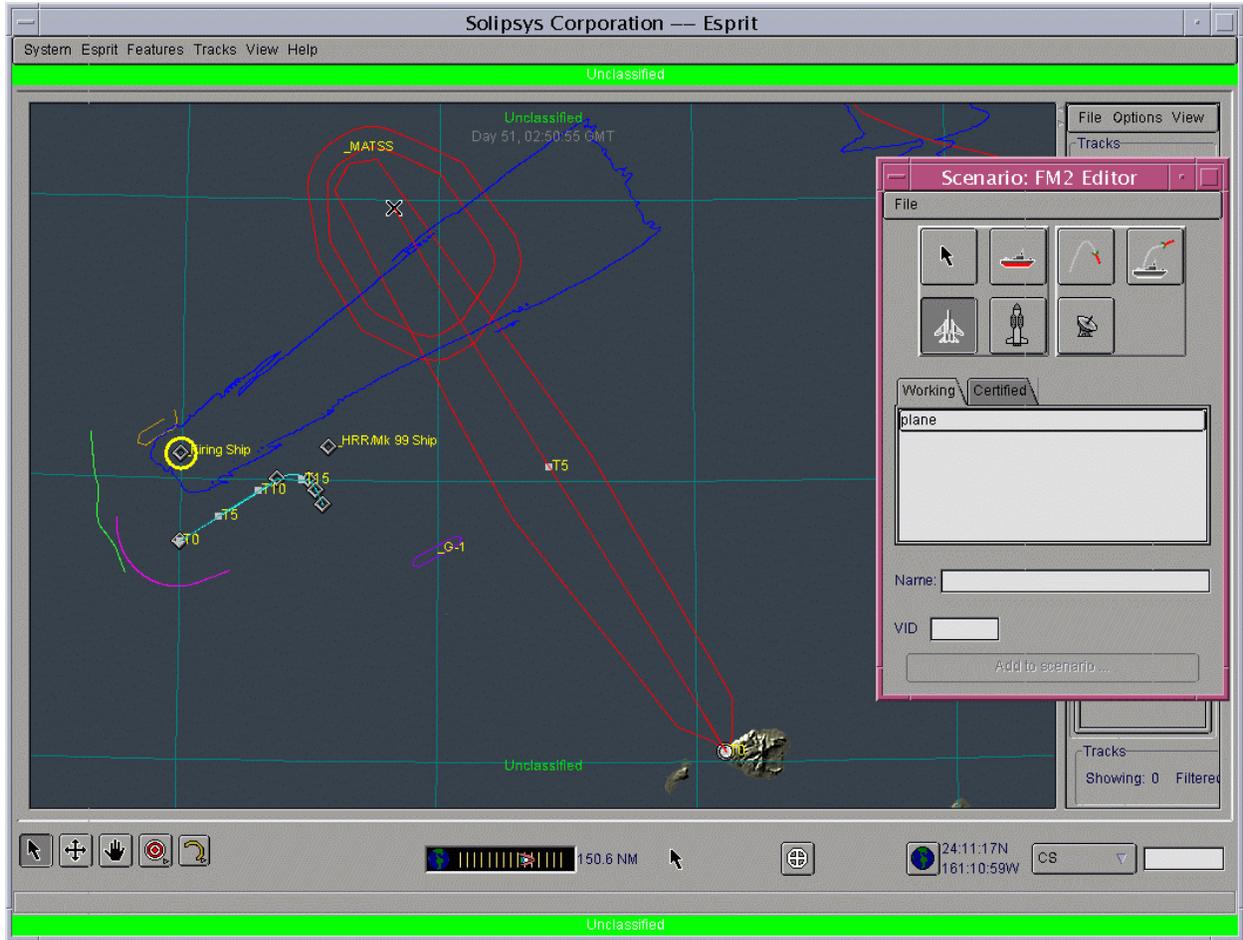


Figure 32. Opened Scenario

7.3 Scenario Execution

Selecting the Preview item from the Esprit pull-down menu gives the user access to the Scenario Playback Control panel. The controls provided by this panel are illustrated in Figure 33. The currently opened scenario is selected for playback.

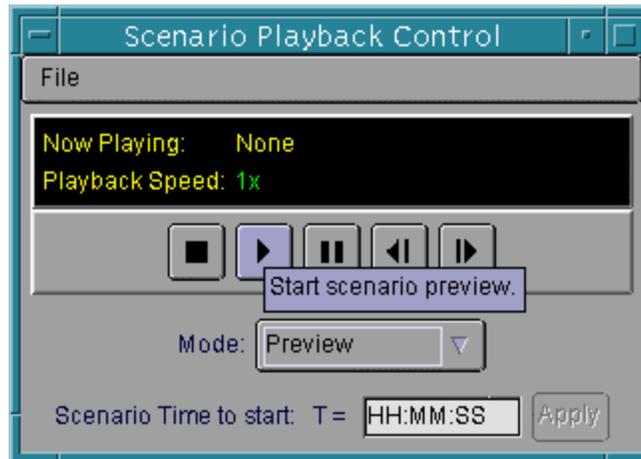


Figure 33. Scenario Playback Control Panel

This panel includes buttons and entries to select:

- Playback stop
- Playback start
- Playback pause
- Playback speed – either slow down or speed up
- Playback Mode – either “Preview” or “Local Execution”
- Time to start

7.3.1 Scenario Preview Mode

Scenario preview allows the user to review the scenario plan, including vehicle trajectories and synchronization. Scenario preview can be started from any selected scenario time, and can be executed at up to 8 times real-time speed. Entering 00:00:00 indicates that preview should start at T0, while a negative time allows a countdown to be performed. Only planned vehicle positions versus time are generated during scenario preview. No target or radar detection messages are generated.

Figure 34 depicts ESPRIT in the scenario preview mode.

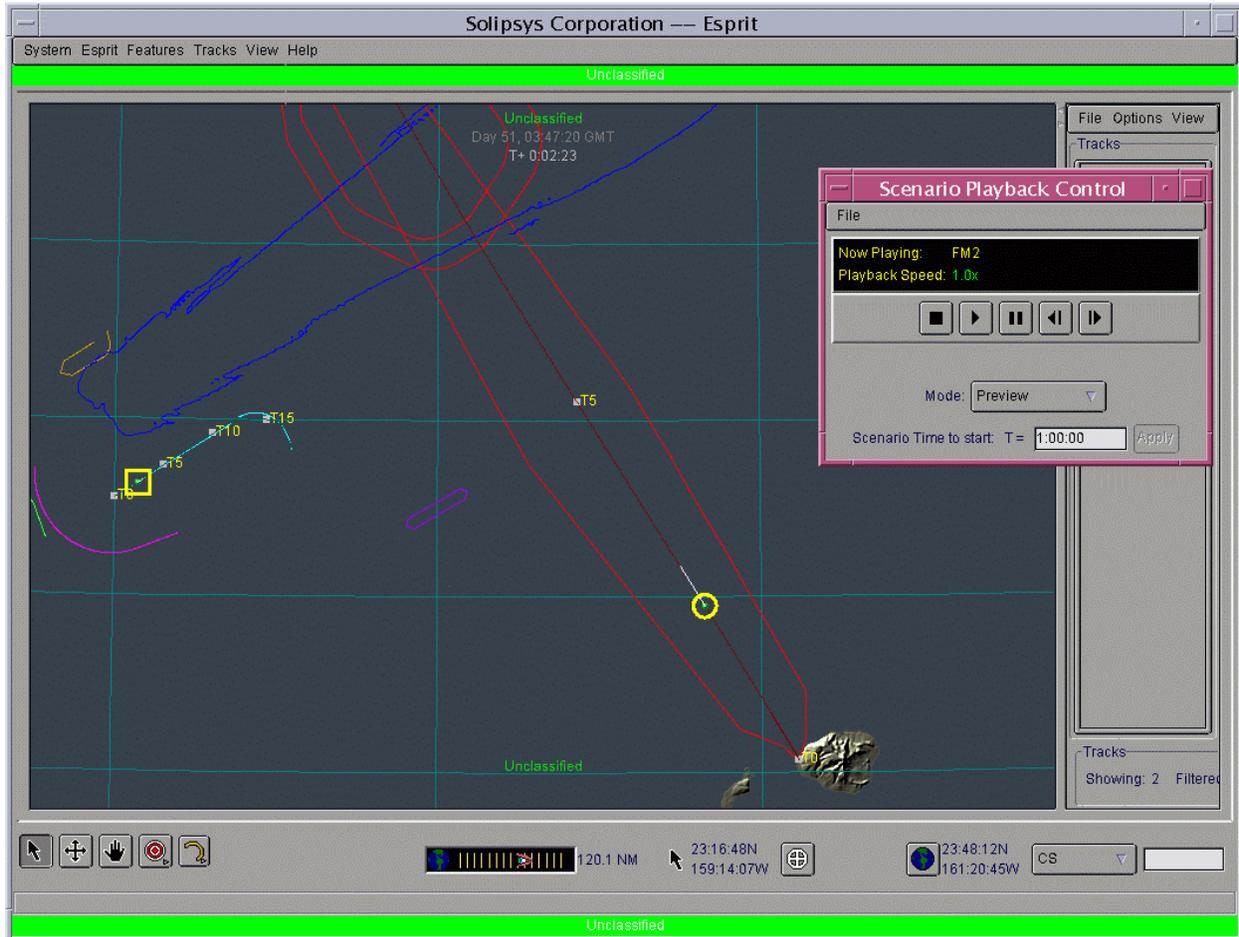


Figure 34. Scenario Preview Mode

7.3.2 Local Execution

Local execution is the rehearsal mode of ESPRIT. In addition to the planned vehicle positions, local execution mode also generates display tracks based upon the ability of simulated sensors in the scenario to "track" the vehicles.

In addition, simulated radar tracking messages or radar detection messages may be generated. To activate the track message simulation, bring up the Esprit preference panel. This is accessed via the menu **System** → **Preferences**, then selecting the Esprit icon under **Options**, then selecting the **Simulation** tab. This is shown in Figure 35.

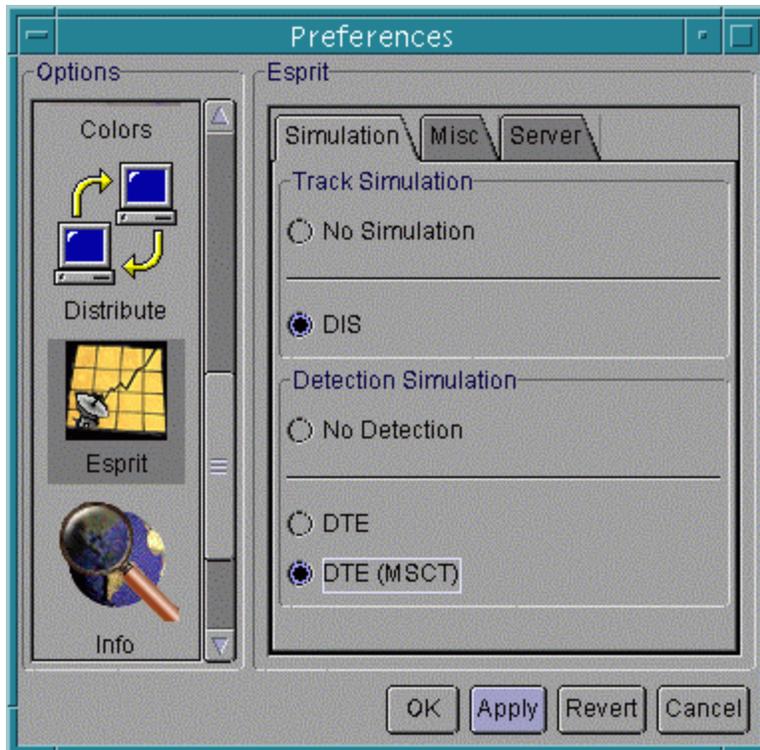


Figure 35. ESPRIT Simulation Preference Panel

7.3.2.1 Track Simulation

By default, when a scenario is loaded, simulation is turned off. Select the desired simulation type and press the Apply button to enable simulation. To disable simulation, select No Simulation and press Apply.

Currently, track simulation has been implemented for Distributed Interactive Simulation (DIS). Enabling DIS simulation will cause ESPRIT to broadcast DIS EntityStatePDU's on the broadcast network defined by the environment variable DIS_NET (defaults to 10.0.1.255 or the primary interface) and the UDP port defined by the environment variable DIS_PORT (defaults to 3000). See Table 14:ESPRIT Server Environment Variables.

7.3.2.2 Detection Simulation

To enable radar detections with the DTE format, select the flavor of message desired and press Apply. (DTE: the standard DTE data format with idle and sync patterns followed by the data block and ending with a parity byte; DTE (MSCT): this is a slight variation of the DTE message with an ID byte, a subID byte, and the radar ID byte followed by the standard data block.)

The DTE messages will be broadcast on the network defined by the environment variable DTE_NET (defaults to 10.0.1.255 or the primary interface, see Table 14:ESPRIT Server Environment Variables). To specify the UDP ports that the DTE messages are broadcast on, a text file is read which maps the sensor identification (SID) to the port assignment. This text file

is the MSCT RADARS.cfg file which is stored in the cfg directory under the directory specified by the environment variable, DBDIR. On each line specifying the parameters for a radar, add the additional parameter DTEPORT <port number>. For example, see Figure 36. (Also, refer to MSCT manual.)

```

PRIMARY_RADAR_ID 3
RADAR 1 LAT 32:48:00S LON 151:49:58E ALT 0000.0 AZERR 0.25 RNGERR 820.0 AZOFFSET 0.0 DTEPORT 4000
RADAR 2 LAT 33:57:05S LON 151:10:48E ALT 0074.0 AZERR 0.25 RNGERR 820.0 AZOFFSET 0.0 DTEPORT 4001
RADAR 3 LAT 33:36:54S LON 150:16:06E ALT 3670.0 AZERR 0.25 RNGERR 820.0 AZOFFSET 0.0 GRU DTEPORT 4002
RADAR 4 LAT 30:26:21S LON 152:14:21E ALT 5224.0 AZERR 0.25 RNGERR 820.0 AZOFFSET 0.0 DTEPORT 4003
RADAR 5 LAT 14:30:27S LON 132:26:27E ALT 0000.0 AZERR 0.25 RNGERR 820.0 AZOFFSET 3.99375 DTEPORT 4004
RADAR 6 LAT 32:46:37S LON 151:49:13E ALT 0000.0 AZERR 0.50 RNGERR 820.0 AZOFFSET 12.37333 DTEPORT 4005
PLOT LAT 33:57:05S LON 151:10:48E RNG 75.0

Radar 1: RAAF Williamtown ATC (ADATS)
Radar 2: Airservices - Sydney Airport
Radar 3: Airservices - Mount Boyce
Radar 4: Airservices - Round Mountain
Radar 5: Tindal - Alenia Radar
Radar 6: Williamtown - TPS 43

```

Figure 36: Sample RADARS.cfg file with DTEPORT Added

In future releases of ESPRIT, the detection type, including port numbers, will be specified as part of the sensor model. Additional detection types will be added as they are identified (e.g., CD2).

7.3.3 Constraint Monitoring

To monitor constraints, open the Constraint Monitor by selecting **Esprit** → **Constraint Monitor**. Upon starting the Constraint Monitor, shown in Figure 37, any constraints that have been defined during this current scenario planning session will be automatically loaded. If constraints are to be loaded from a file, select **File** → **Open**, and choose the file from which the constraints should be loaded. The loaded constraints are sorted by Category and displayed in a tree-table format. To view the constraints for a particular Category, left-click the folder icon next to that Category. This will open that Category folder and display its constraints. To begin constraint monitoring, select **Server** → **Start Monitor**. This will start the constraint monitoring and periodic status messages will be sent to show the violation status of each constraint.

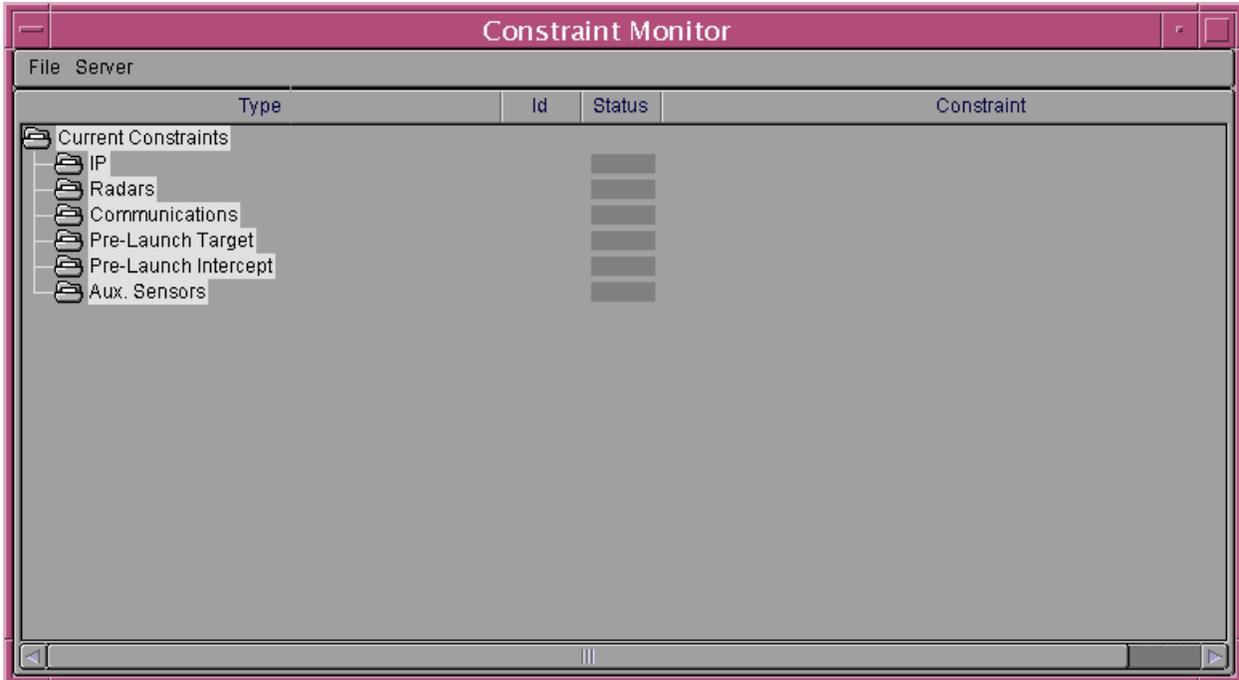


Figure 37. Constraint Monitor

7.3.4 Event Monitoring

Event monitoring is typically accomplished by executing ESPRIT in Preview mode at 1 times the scenario speed while observing real-time events. This allows the user to compare the plan versus the actual event.

To compare the trajectory of a TBM or interceptor to its nominal, add the vehicle to the ESPRIT Altitude Graph. This can be done by right-clicking on a point on the TBM or interceptor trajectory and selecting the **Add to AltitudeGraph** option. Alternatively, bring up the Altitude Graph by choosing **Esprit → Altitude Graph** option. From the Altitude Graph panel shown in Figure 38, select the **File → Add vehicle** menu option and enter the VID of the vehicle in question. By default, the graph displays altitude versus scenario time for the planned path along the nominal trajectory and the path as detected by any sensors tracking the specified VID. To display altitude versus downrange, select **View → X Axis → Downrange**.

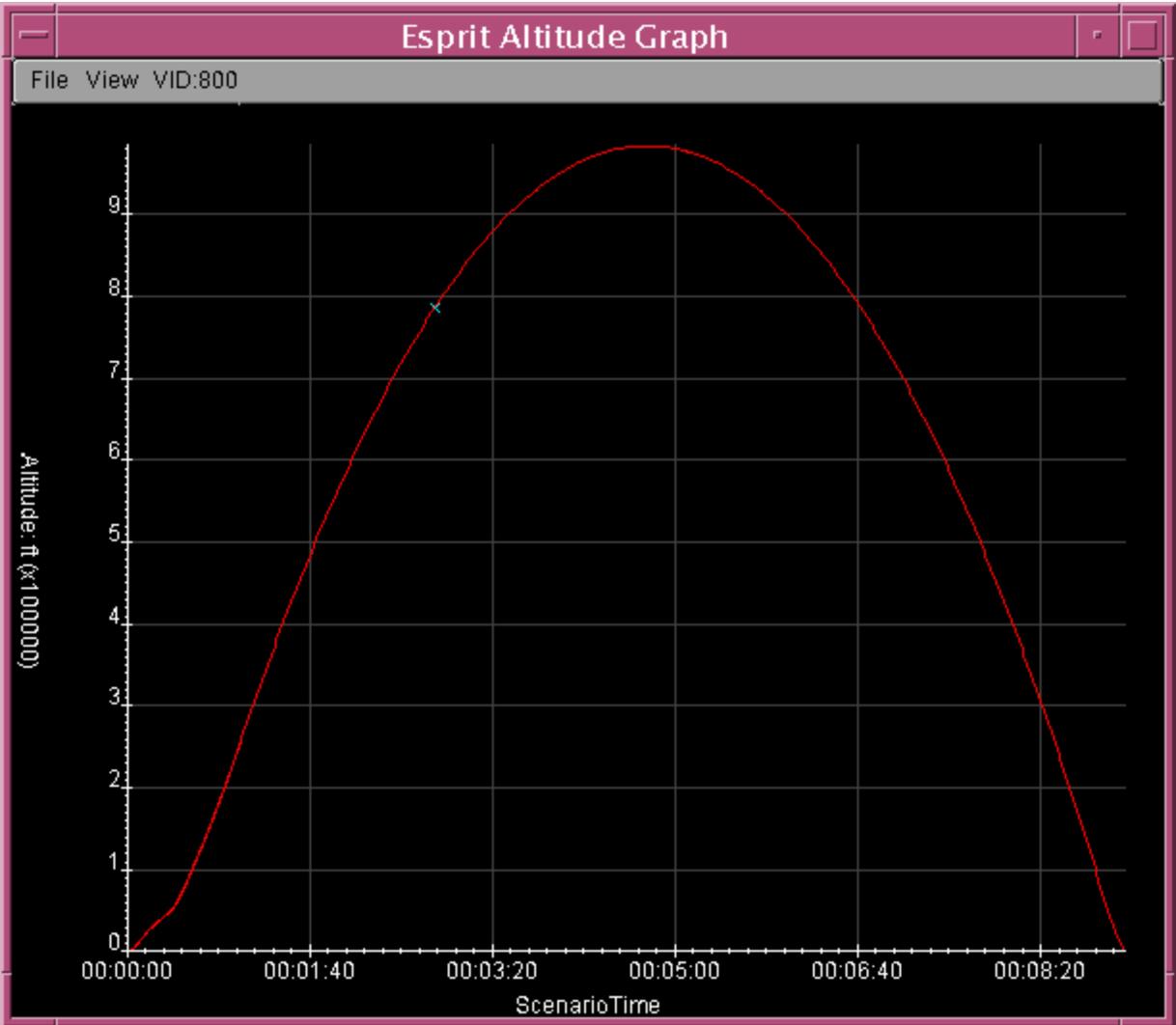


Figure 38. ESPRIT Altitude Graph

Real time events may be recorded using the Data Extract mechanism described in the TDF User's Manual. Please note that live data is not displayed while recorded data is being replayed.

8 Installation and Configuration

The ESPRIT client and server communicate with each other via SoliNet™ messages passed by a daemon process named **solid**. The server daemon itself is called **spirited**. The TDF/ESPRIT client and **spirited** server pass messages back and forth through the **solid** daemon. This architecture allows multiple clients to connect to a **solid** and act as observers of a scenario running in the **spirited**. The **spirited** daemon is responsible for storage of the scenario and generation of the simulated tracks/contacts. Certified models and entities are also stored on the server.

The following sections describe the directory structure and configuration of the ESPRIT scenario planning and rehearsal tool as typically installed on a Solaris™ system. They also discuss how to customize and update your installation and how to properly configure a client to run on Windows NT™.

8.1 Server Installation

The ESPRIT server is installed on a Solaris™ system via an InstallShield™ JavaEdition installer that automatically detects the architecture and OS version to install the appropriate binaries and libraries.

NOTE: Prior to installation, Java v1.3.x should be installed on the system and set to be the default version of Java (via the symbolic link `/usr/java`). The Java JRE is included on the CD in the `jvm` directory. Java 1.3.x is used for both the installation and the ESPRIT client.

To install ESPRIT, insert the CD into the CD-ROM drive. This will automatically start the installation. If for any reason the installation does not start automatically, the installation may be started manually by running the `setup.bin` (on SPARC systems) or `setup_x86.bin` (on Intel systems). Alternatively, the setup may be run by simply running the executable jar file:

```
$ java -jar setup.jar [-console]
```

Adding the console argument will run the installer in console mode, i.e., without the graphical user interface. This allows installation from a remote location over the network.

TIP: If the operating system mounts the CD-ROM with a pound sign ('#') in its path, the installation may fail. In this case, turn off volume management and mount the CD-ROM manually.

Once the installation has been completed, make any necessary modifications to the `envset.csh` script as described in Section 8.2.1.1. This may include modifying `ESPRITROOT` if installed to other than the home directory (Figure 39). If the server is to be started using one of the server scripts described in 8.2, no further action is necessary. If the ESPRIT Server executable is to be started manually, one must 'source' the `envset.csh` script manually.

8.1.1.1 Directory Structure

The default layout for the ESPRIT directory hierarchy is as follows:

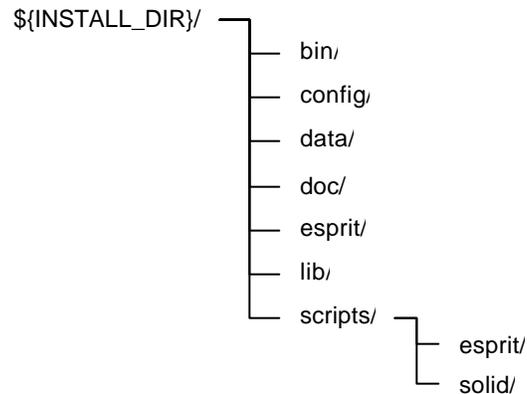


Figure 39: ESPRIT Home Directory Structure

The **bin** directory contains all binary files and executable shell scripts for both the server and the client. The **lib** directory contains dynamically linked libraries and java jar files used by the system. The **data** directory contains many support files, including the geographic databases, plug-in lists, sounds, user preferences, and ESPRIT specific data. The **doc** directory contains the User's Manual, the TDF User's Manual and any other documentation.

The **scripts** directory contains a collection of startup scripts that may be used to run the ESPRIT server. Of these scripts, the only one that the ESPRIT administrator should ever need to edit is **esprit/envset.csh** (see section 8.2.1.1 envset.csh for more information).

The **esprit** subdirectory is where all of the supporting files for the ESPRIT server are found (see Figure 40). At the heart of these files is the **scenarios** directory, where all scripted scenarios are stored. Scenarios are saved to a file that is created in either the **certified** or **working** subdirectories under **scenarios**. For example, if a working version of a scenario named MyTestScenario is saved, it will be saved to a file named **\${INSTALL_DIR}/esprit/scenarios/working/MyTestScenario**.

The **certifiedEntities** and **certifiedModels** directories store the data files for any vehicle/sensor models/entities that have been certified by an authorized user via the certification interface in the client GUI. For more information on how models and entities are certified, see section 5.1 Model Creation.

The **nominalModels** directories hold the converted nominal data sets (e.g., for TBMs), while the **nominal TextModels** hold pre-conversion nominal data sets.

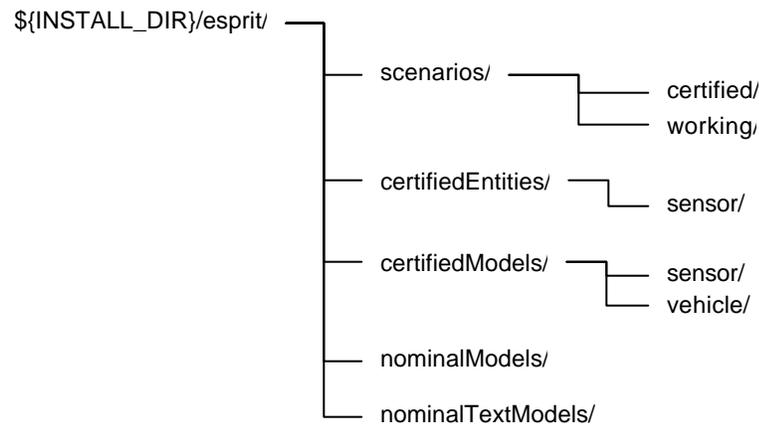


Figure 40: ESPRIT ServerSupport Directories

8.2 Server Configuration

Two processes make up the server,

- **solid** – the process that handles message processing between the client and server, and
- **spirited** – the esprit server daemon that is responsible for scenario planning and simulation functions.

The **spirited** process connects to the **solid** based upon a network port number (37000 by default), thus the **solid** must be started prior to the **spirited**. When an ESPRIT client connects to the same **solid**, it is able to pass messages to the **spirited** server.

These processes may be started, by hand, on the command line, provided that the user's environment is correctly set up. To make this easier, we provide a collection of scripts to set the environment, and start the processes.

To start the server, one simply calls the script **scripts/server**, which starts up a **solid** daemon, then runs the script **scripts/server-`hostname`**, which will start any desired daemons based such as **spirited**. The **server-`hostname`** strategy allows one collection of scripts to be configurable based upon the hostname of the system the **server** script is run on. This in turn invokes the script **`\${ESPRITSCRIPTS}/server_<host name>**, where host name is the name of the server; this file can be created by copying any of the **server_xxx** files in the directory to the required name.

The three main types of server scripts are shown in Table 13. The local server is typically used for planning where resources from other systems are not needed. The solid-only version does not run a scenario planner, but simply maintains a **solid** daemon. Finally, the ESPRIT-only version is used if the planner is going to connect to a remote **solid**. If you do not wish to use the **server-`hostname`** strategy, you may copy the **scripts/esprit/server.local** script over **scripts/server** and all esprit server processes will be started.

NOTE: Before using the **server** script, edit it to set the default **ESPRIT_INST_DIR** variable to correspond with your installation directory. Then rename **server-esprithost** to **server-`hostname`**, where hostname is the name of the system on which the ESPRIT server has been installed.

The **solid** daemon is actually started via the **scripts/solid/solid.csh** script. This script sources the **envset.csh** (see §8.2.1.1envset.csh) script, then runs **solid** in a loop that will restart the **solid** in case it is terminated for any reason.

Like the **solid – solid.csh** script pairing, the **spirited** daemon is actually started via the **scripts/esprit/esprit.csh** script. This script sources the **envset.csh** (see §8.2.1.1) script, then runs **spirited** in a loop that will restart the **spirited** in case it is terminated for any reason.

Table 13. Server Startup Scripts

Server Script	solid	spirited	comments
server	X		Also calls server-`hostname`
server-`hostname`		X	Rename this script replacing the name of your host machine.
esprit/server.local	X	X	Alternative to standard server script
esprit/esprit.csh			Automatically restarts spirited if it is terminated.
solid/solid.csh			Automatically restarts solid if it is terminated.

8.2.1 Environment Variables

The environment variables that the server uses are listed in Table 14 and Table 15. The good news is that these are set in the **envset.csh** script, so once that script is configured properly, most of these can be forgotten (see § 8.2.1.1envset.csh).

8.2.1.1 envset.csh

When the server is run via one of the server scripts, environment variables are set via the script **envset.csh**. In general, the only server variables that need to be set upon installation are the

following: **ESPRIT_INST_DIR** and **SOLINET_PORT**. Additionally, the following may be set to change the behavior of the system: **SOLINET_PORT_UDP** (if solids are to communicate with each other), **DIS_NET**, **DIS_PORT** and **DTE_NET**.

NOTE: Before using the **envset.csh** script, edit it to set the default **ESPRIT_INST_DIR** variable to correspond with your installation directory.

Table 14:ESPRIT Server Environment Variables

Variable	Default Value	Purpose
ESPRIT_INST_DIR	\${HOME}	This variable defines the base directory for the ESPRIT installation.
ESPRITBIN	\${ESPRITROOT}/bin	binary executables
ESPRITLIB	\${ESPRITROOT}/lib	binary libraries
ESPRITJARS	\${ESPRITROOT}/lib	Java jar files
ESPRITLOGS	\${ESPRITROOT}/log	Log files
ESPRITSCRIPTS	\${ESPRITROOT}/scripts	scripts
SOLINET_PORT	37000	port of the solid that ESPRIT should communicate through
SOLINET_PORT_UDP	<not set>	if set, uses solidarity to connect solids
SOLINET_REMOTE	<not set>	hostname of the solid that ESPRIT should communicate through
DIS_NET	10.0.1.255	broadcast address for DIS messages
DIS_PORT	3000	UDP port for broadcast of DIS messages.
DTE_NET	10.0.1.255	broadcast address for DTE contact messages.
DBDIR	\${INSTALL_DIR}/db	gives the location of the MSCT configuration files used for the DTE detection simulation.

Table 15: ESPRIT Server File Location Environment Variables

Variable	Default Value
ESPRIT_ROOT	\${ESPRITROOT}/esprit
ESPRIT_WORKING_SCENARIOS	\${ESPRIT_ROOT}/scenarios/working
ESPRIT_CERTIFIED_SCENARIOS	\${ESPRIT_ROOT}/scenarios/certified
ESPRIT_SENSOR_MODELS	\${ESPRIT_ROOT}/certifiedModels/sensor
ESPRIT_VEHICLE_MODELS	\${ESPRIT_ROOT}/certifiedModels/vehicle
ESPRIT_SENSOR_ENTITIES	\${ESPRIT_ROOT}/certifiedEntities/sensor
ESPRIT_NOMINAL_MODELS	\${ESPRIT_ROOT}/nominalModels
ESPRIT_NOMINAL__TEXT_MODELS	\${ESPRIT_ROOT}/nominalTextModels

Table 16: General Environment Variables

Variable	Default Value	Purpose
PATH	\${ESPRITBIN}:\${PATH}	
LD_LIBRARY_PATH	\${ESPRITLIB}:/usr/local/lib	
CLASSPATH	\${ESPRITJARS}/tdfEsprit.jar: \${ESPRITJARS}/tdf.jar: \${ESPRITJARS}/geo.jar: \${ESPRITJARS}/sol.jar: \${ESPRITJARS}/ms2525a.jar: \${ESPRITJARS}/vecmath.jar: \${ESPRITJARS}/plaf.jar: \${ESPRITJARS}/solinet.jar: \${ESPRITJARS}/solimath.jar: \${ESPRITJARS}/dataextract.jar: \${ESPRITJARS}/granite.jar: \${ESPRITJARS}/slate.jar: \${ESPRITJARS}/datafusion.jar: \${ESPRITJARS}/prophet.jar: \${ESPRITJARS}/esprit.jar: \${ESPRITJARS}/xml.jar: \${ESPRITJARS}/jh.jar: \${ESPRITJARS}/jmf.jar: \${ESPRITJARS}/servlet.jar	ESPRIT plugins TDF standard jars ESPRIT Server jars Other Java APIs

8.2.2 Automatic Startup

Often, the ESPRIT system is configured such that the server is automatically started whenever the system is rebooted. As a convenience to system administrators, a copy of a startup script,

S99esprit, is included in the **scripts/esprit** directory. On Solaris systems, the **S99esprit** script may be copied into **/etc/rc2.d**, then a hard link to that file can be made as **/etc/init.d/esprit**. To start/stop the server, one would then simply run the command (as root): **/etc/init.d/esprit { start | stop }**. If this is used, the environment variable **ESPRITUSER** should be set to the username under which the server will be run. We typically create an esprit account on systems we deploy, thus we set the default **ESPRITUSER** to **esprit**. This in turn sets the default **ESPRIT_INST_DIR** to **/home/esprit**. Please make the appropriate edits to the **S99esprit** script for your installation.

NOTE: Before using the **S99esprit** script, edit it to set the default **ESPRIT_INST_DIR** variable to correspond with your installation directory.

Please refer to your system administrator, or contact us at espritsupport@solipsys.com for assistance.

8.3 Client Installation

8.3.1 ESPRIT Client Installation

The ESPRIT client is installed via an InstallShield™ JavaEdition installer.

NOTE: Prior to installation, Java v1.3.x should be installed on the system and set to be the default version of Java (via the symbolic link **/usr/java**). The Java JRE is included on the CD in the **jvm** directory. Java 1.3.x is used for both the installation and the ESPRIT client.

To install ESPRIT, insert the CD into the CD-ROM drive. This will automatically start the installation guide. If for any reason the installation does not start automatically, the installation may be started manually by running the **setup.exe** (on Win32 systems), **setup.bin** (on SPARC systems) or **setup_x86.bin** (on Intel systems). Alternatively, the setup may be run by simply running the executable jar file:

```
$ java -jar setup.jar [-console]
```

Adding the **-console** argument will run the installer in console mode, i.e., without the graphical user interface. This allows installation from a remote location over the network.

TIP: If the operating system mounts the CD-ROM with a pound sign ('#') in its path, the installation may fail. In this case, turn off volume management and mount the CD-ROM manually.

Once the installation begins you will be presented with and asked to accept the license agreement. Once the license is accepted, accept the installation directory or change it. Next, you are given a choice of minimal, typical or custom installations. A typical installation will automatically include the server files when installed on a Solaris™ system (7 or 8, SPARC or x86). Note that when the installation first begins, several minutes will pass without apparent progress.

8.3.2 Client Configuration

Figure 41 shows the directory structure for the ESPRIT client.

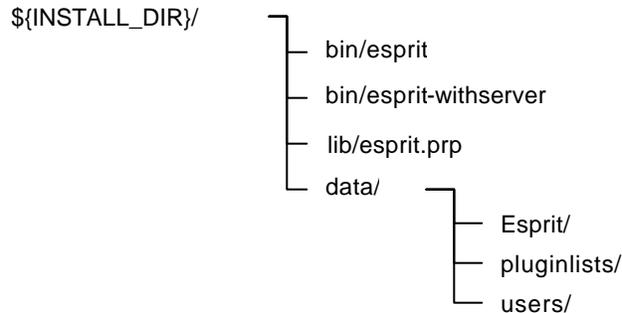


Figure 41: ESPRIT Client Configuration Area

On Solaris systems, the **esprit** script is used to start the client. On Win32 systems, the **esprit.bat** script is used. There are some basic parameters that are set at the top of the **esprit** script that typically would not ever need to be changed and are used to set up the **CLASSPATH** used when running the ESPRIT client. The **esprit.prp** file is the client property file (see § Property Files 8.3.2.1 for more information).

The **esprit-withserver** script is included as a quick and easy way of running both the client and server together on a Solaris system. This script starts both the server and the client, then terminates the server when the client exits. When using this script, make certain that the **SOLINET_PORT** set in the script matches the **EspritClientModule.port** property set in the **esprit.prp** file (see § 8.3.2.1). Also make sure that you are not already running a **solid/spirited** pair on that same port.

Under **data/** are the **Esprit**, **pluginlists** and **users** directories. The **Esprit** directory under **data/** is used to store working versions of models and entities, as well as overlays. Any variant overlay definition specifications will also have data files stored under **Esprit**.

The **pluginlists** directory holds all of the plug-in lists used for manually selecting which plug-in features TDF will load. For the ESPRIT client, this includes the following files: **tdf.plg**, **sol.plg**, **geo.plg** and **esprit.plg**. These plugin lists are used in conjunction with the **esprit.prp** file that selects which lists to use.

The **users** directory stores persistent information for TDF features that store preferences. An example of the preferences would be the choice of which maps and which projections to display. Preferences are stored on a per user basis.

8.3.2.1 Property Files

The **esprit.prp** file contains many properties that affect the behavior of the ESPRIT client. The properties that may need to be modified for ESPRIT are described in Table 17.

The **EspritClientModule.host** property needs to be set if the client is connecting to a remote server. The **EspritClientModule.port** property needs to be set to the port number of the **solid** daemon that is handling the messaging for the ESPRIT server. This port setting should correspond to the value of the environment variable **SOLINET_PORT** set on the server (see §Table 14 for details).

Finally, the **Environment.plugins** property lists the pluginlists needed for the ESPRIT client.

Table 17: ESPRIT Client Properties

Property	Default Value	Purpose
EspritClientModule.host	localhost	Provides the hostname of the system running the ESPRIT server.
EspritClientModule.port	37000	Provides the port number of the solid daemon on the server.
Environment.plugins	FILE \ data/pluginlists/esprit.plg\ data/pluginlists/sol.plg\ data/pluginlists/geo.plg\ data/pluginlists/tdf.plg\ END	These files list the plug-ins that are loaded into TDF.

8.3.2.2 Environment Variables

The scenario planner client utilizes the **PATH** and **CLASSPATH** environment variables listed in Table 16 to find the appropriate executable scripts and jar files. Note that the **CLASSPATH** variable is set explicitly in the startup scripts, **esprit**, **esprit.bat** and, **esprit-withserver**.

9 Notes

9.1 List of Acronyms and Abbreviations

ABT	Air Breathing Target
dB	Decibels
deg	Degrees
ESM	Electronic Surveillance Measures
ESPRIT	Exercise Scenario Planning and Real-time Integrated Test
G	Acceleration of Gravity
gal	Gallons
GOG	Graphical Overlay Group
GUI	Graphical User Interface
hr	Hours
ID	Identification
IR	Infrared
kft	Kilofeet (1000 feet)
km/hr	Kilometers per hour
kts	Knots
kW	Kilowatts
m	Meters
mi	Miles
ms	Milliseconds
OP	Operational
OPS	Operations
Pd	Probability of Detection
RF	Radio Frequency
sec	Seconds
SID	Sensor Identification
TBM	Theater Ballistic Missile
TBMD	Theater Ballistic Missile Defense
TDF	Tactical Display Framework
TOI	Target of Interest

TSPI	Time-Space Position Information
VID	Vehicle Identification
W	Watts

Index

CLASSPATH.....	60
climb rate.....	14, 15, 28
close scenario	40
DIS	4, 45, 59
Distributed Interactive Simulation.....	<i>See</i> DIS
dive rate.....	14, 15
DTE.....	45, 59
environment variables.....	58–60 , 63
GUI architecture.....	2–3
information panel.....	26, 35
installation	
server	55
local execution	ii, 1, 4 , 43, 44
pre-planning	ii, 1, 3, 6
RF loop gain.....	9, 10
save	
scenario	39
sensor entity	21
sensor model	12
vehicle model.....	16
scenario editor.....	23, 25–26 , 35
scenario preview mode.....	ii, 3 , 4, 43
scripting.....	1, 3, 22, 30
sensor	3, 6, 41
assignment.....	35–37
model.....	ii, 1, 6
reports.....	4
sensor entity	17–22
sensor model	7–12
surveillance sensors.....	22
tracking sensors.....	22
simulation	
enabling.....	45
TBM model.....	16–17
waypoints	14, 22, 26, 27, 28, 29–30, 41