SpaceNet 2.5r2 User's Guide

Jet Propulsion Laboratory
California Institute of Technology

Massachusetts Institute of Technology

TABLE OF CONTENTS

INTRODUCTION

SpaceNet is a software tool that models space exploration from a supply chain and logistics architecture perspective. At the start of this project, SpaceNet was a fully functional model that was the result of two years of development and application in the NASA community. The peak of its analytical capability was demonstrated in the "Integrated Cx Mission Modeling" study for the Constellation Program, where it was used to quantify performance drivers for the lunar campaign [1]. That version of SpaceNet was primarily a MATLAB application and graphical user interface (GUI) supported by an Excel database.

Over the course of SpaceNet development and customer interaction, buzz words such as "commonality" and "repairability" floated around as potential analysis topics. The first area of such an analysis would be in modeling these "-ilities" at the subsystem level; the second, in propagating their impacts back to the system and mission level, the current domain of SpaceNet. The following upgrades were implemented in SpaceNet to handle this new domain of analysis:

- Model the *reconfigurability* of systems such that duty cycles, functional capabilities, and operational states can be chosen dynamically depending on the mission plan.
- Analyze the *reuse* of systems beyond the utilization of static, pre-deployed assets, specifically surface mobility systems.
- Analyze the impact of *commonality* and cannibalization among systems such as surface rovers, habitation modules, and propulsive stages.
- Model surface *repair* activities and how they impact system availability, spare parts and tool demands, and the time available for exploration and maintenance.

Due to the fundamental change in modeling triggered by the proposed upgrades (subsystem level vs. system level), a decision was made to re-structure SpaceNet from the ground up, rather than modify the existing version of the model. The re-structuring was pursued with the goal of creating a flexible, modular framework that would be extensible to future upgrades and use cases beyond the scope of this project.

One of the first actions was to transition from a procedural, "bus" architecture to an object-oriented environment. To achieve this, data models, object and domain hierarchies, and simulation events were created with the intent of being able to define any space logistics exploration mission. A product of the object-oriented conceptualization was the addition of element-centric demand models. In previous versions of SpaceNet, cargo demands were only generated at the mission level and driven by parameters such as crew size and surface duration. With element-centric demand models, cargo demands can be generated on a per-element basis and then aggregated up to the mission level. This allows reconfigurable elements to have unique, customizable demand models as a function of their operational states.

Parallel to the efforts in software architecting was the development of logic flow diagrams and modular prototypes from a user's perspective. While data objects and component interfaces formed the basis of the modeling framework, it was not clear how a user would interact with them when constructing a scenario for analysis and trade study purposes. A conceptual user interface was generated by breaking down an exploration campaign into the high-level modules a user would define. These modules include the network definition, mission creation (including the transportation architecture), demand modeling, cargo manifesting, and simulation and visualization. Modular prototypes were designed not only with the use cases and proposed upgrades in mind, but with user accessibility and flexibility as well.

Major Revisions in SpaceNet 2.5

SpaceNet 2.5 includes many major revisions from previous versions. Most of the code was heavily restructured to provide a modular architecture for flexibility for future additions. Some of the major additions include:

Operational Additions:

- Arbitrary-Burn Space Transports
- Surface Vehicles and Surface Transports
- Flight Transports
- Multi-destination Scenarios
- Element-centric Demand Models
- Modular Demand Models
- Reconfigurable Element Operational States
- Common Spares and Spares Scavenging
- Repairable Spares, Auto-Repair
- Transport-level Manifesting
- Data Source Independence (Online SQL Database and Excel File Database)

Usability Improvements:

- Completely Redesigned User Interface
- Element-based Bat Chart
- Scenario Feasibility Chart (Cumulative Delivery Capability and Demands)
- Time-Expanded Supply Network Visualization
- Repairability Effectiveness Chart
- Continuous Pre-Simulation for Real-time Error Feedback
- Robust Simulation Error Handling

SpaceNet History

Designation	Comments
SpaceNet 1.1	Prototype (2005)
SpaceNet 1.2	Established visualizations and database (2006-2007)
SpaceNet 1.3	Public release, scenarios focused on lunar sorties (2007)
SpaceNet 1.4	Scenarios focused on lunar campaigns, demand modeling (2007-2008)
SpaceNet 2.0	Code migration to Java (platform independent), advanced visualizations, SQL database (2007-2008)
SpaceNet 2.5	Diverse scenarios (ISS, Moon, Mars), user-centric design, flexible data structures, element-centric demand models, "-ilities" analysis (2008-2009)
SpaceNet 2.5r2	Second release of SpaceNet 2.5 – includes usability improvements, demand export options compatible with matrix-based manifesting methods, native data editing capabilities (including online databases), a revised look-and-feel that provides a more similar environment across platforms, and other general additions

SpaceNet User Interface

The SpaceNet User Interface (UI) was completely redesigned for SpaceNet 2.5. It is distributed as a .jar (Java Archive) file, and should run by simply double-clicking the icon. It was developed using **Java Version 6 Update 14**, earlier versions of Java are not guaranteed to operate correctly. You can check your version of Java at http://www.java.com/en/download/index.jsp.

Although Java itself is platform-independent, it uses the native window manager for the operating system on which it is run. Furthermore, the Java Virtual Machine currently available or the Macintosh Operating System is version 1.5, whereas version 1.6 is available for Windows. SpaceNet was developed on Windows XP SP3, and further tested with Windows Vista and Macintosh OSX. If you notice problems, try running SpaceNet from the command line, and notify the development team with any error traces.

Note: To run JAR files from the command line in Windows, do the following:

1. Click the "Start Button" and choose "Run…" or hold the Windows button down and press R
2. Enter "cmd" and press enter
3. Navigate to the SpaceNet directory using `cd` to change directories, and `dir` to list the contents of a directory. For example, `cd My Documents` will change to the "My Documents" folder, while `cd ..` will change to the directory one level higher.
4. Issue the command `java -jar spacenet_2-5r2.jar` to launch SpaceNet

Scenario Panel

When opening an existing scenario or creating a new scenario, the first inputs are within the scenario panel, displayed in the red bar at the top of the screen.

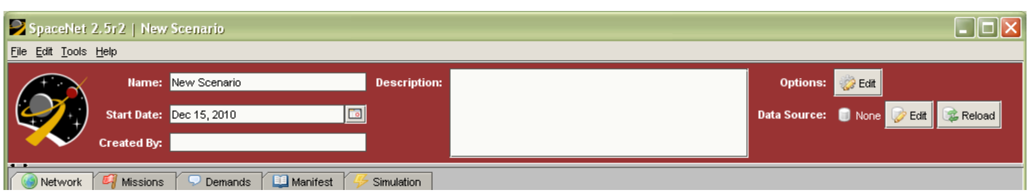


Figure 1: The scenario options panel at the top of the SpaceNet screen.

- **Name:** A name to reference the scenario.
- **Start Date:** The starting date for the scenario, used internally as an epoch to measure simulation time. Although the first mission need not start at the scenario start date, there should be no events that start before the scenario start date.
- **Created By:** A reference to the person who created the scenario.
- **Description:** A short description of the scenario for reference.
- **Scenario Options:** Sets global parameters within the scenario.
- **Data Source:** Displays the type of data source to use. Allows spreadsheet-based and online databases.

Note: You can collapse the scenario panel by clicking the upward-pointing arrow at the left-hand side of the border, directly under the SpaceNet logo.

Scenario Options

The scenario options, accessible via the red scenario panel as well as via the "Edit > Options" menu. This dialog provides one window to manage several categories of options useful for customizing scenarios.

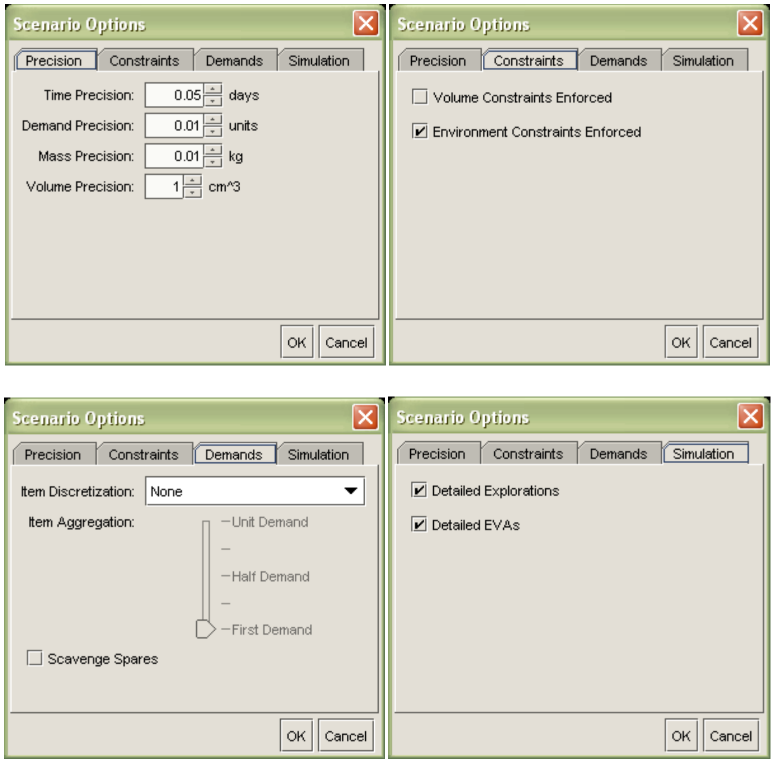


Figure 2: Scenario Options Dialog

Precision

The precision tab sets the minimum-resolution for four different quantities. The time precision sets the smallest quantity allowed for scheduling event executions or transport durations. The demand precision sets the smallest number of units of resources allowed to be demanded between events, or in some cases. The mass precision sets the precision of all computed masses, including cargo capacities and element dry masses. Finally, the volume precision sets the precision of all computed volumes, including cargo capacities and element volumes.

Constraints

When adding elements to carriers, or resources to containers, there are three constraints that set capacity limits: mass, volume, and environment. Mass constraints are always enabled, however sometimes it may be desired to turn volume constraints off to simplify modeling. For the same reason, environmental constraints can also be disabled.

Demands

There are a few options as to how demands are generate during the simulation. The discretization scheme identifies whether non-continuous resources (e.g. spare parts) are treated as continuous or not. If discretization is turned on, items can either be aggregated by elements or by location. Additionally, if discretization is enabled, the items can be aggregated either at the first partial demand, whole unit demand, or somewhere between. The other demand option – scavenging – defines whether spare parts can be utilized from previously-decommissioned elements.

Simulation

Two simulation options help to speed up long-running simulations. In long-duration explorations with a large number of EVAs, if mission-level demand models are used the simulation execution can be simplified to ignore the generated EVA events. By ignoring these events, there are fewer simulation points to evaluate requiring substantially less time and computational resources.

Data Source

The data source provides SpaceNet with the customizable objects used during simulation. The data that is loaded includes nodes, edges, resources and elements. SpaceNet interacts with data sources through a common interface, allowing the use of several types, including local files (Excel) and online databases (SQL).

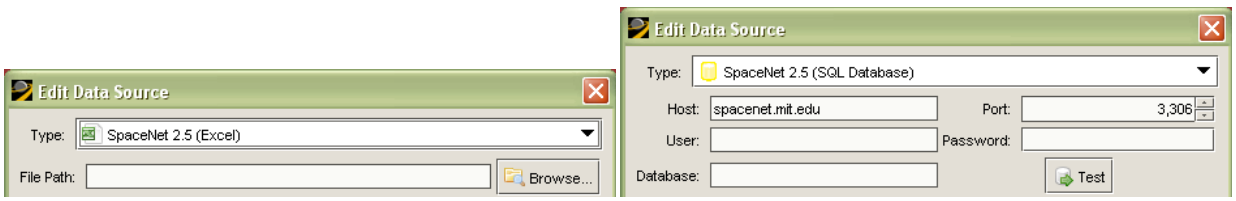


Figure 3: Data source dialog inputs – spreadsheet (left) and online (right) databases

The SpaceNet 2.5 (Excel) data source uses a local Excel spreadsheet to draw information. When specifying this type of data source, browse to find the correct Excel file. The SpaceNet 2.5 (SQL Database) data source accesses an online MySQL database to draw information. It requires the host name (domain or IP address) of the server, the port number to access, the MySQL username and password, and the database name.

When loading the data source the following occurs:

1. The database is error-checked (spreadsheet only)
2. Each node is copied into an Node Library in memory, existing nodes are updated if selected
3. Each edge is copied into an Edge Library in memory, existing edges are updated if selected
4. Each resource is copied into a Resource Library in memory, existing resources are updated if selected
5. The name and type of each element is saved into an Element Library in memory

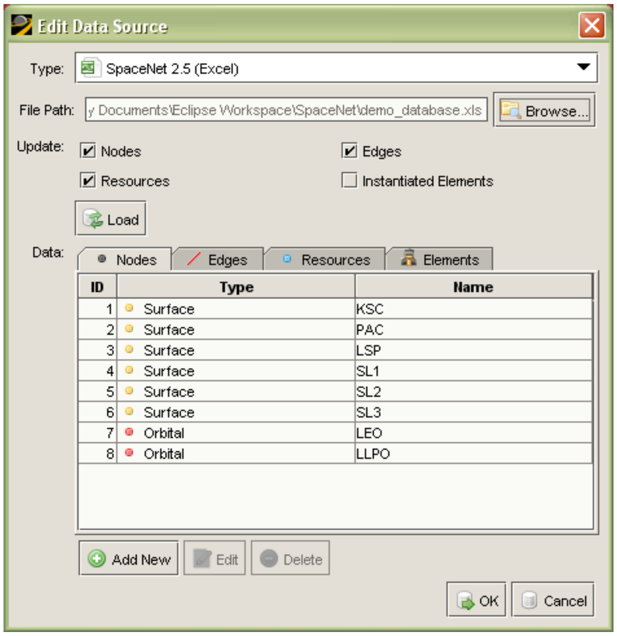


Figure 4: Data source dialog after loading a database

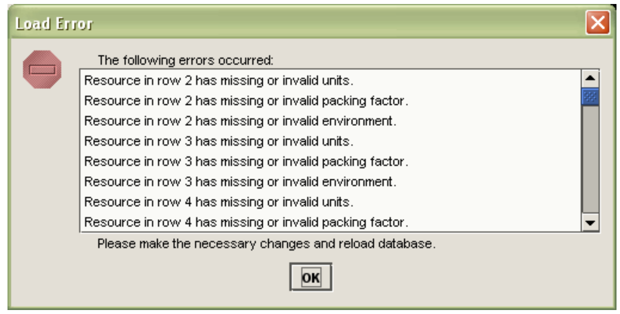


Figure 5: Loading error dialog

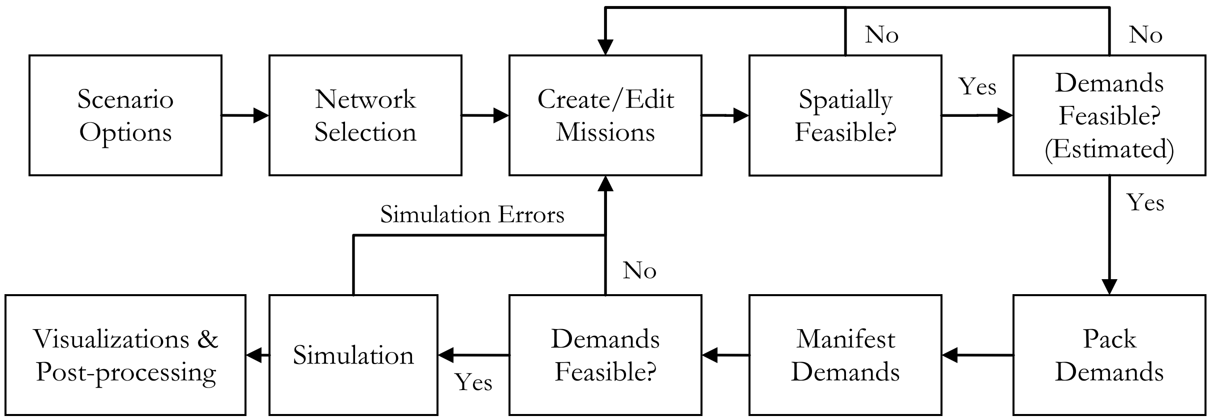
All of the objects loaded into memory are saved with the scenario. Since only the name and type of each element is actually saved in memory (to reduce scenario size), the data source must be accessed each time a new element is instantiated in a Create Elements event. If you are not adding any new elements to your scenario, you do not need to distribute your database with the saved scenario file.

When reloading a database, there are two options:

- **Update existing nodes/edges/resources:** Overwrites existing objects in the library with new copies from the data source. This only works with nodes, edges, and resources, as these objects are continually referenced from the data source.
- **Update instantiated elements:** Updates existing elements in the scenario with new data from the data source. Although this method works to make changes, the element type cannot be changed once it is instantiated without removing all existing references to the element. Also, any customization of elements (from the Edit button in the Create Elements event) will be overridden.

Scenario Creation Flow Chart

The process of scenario creation is often an iterative process in order to achieve logistics closure. The following flow chart describes the typical path of scenario creation.


Scenario Options
Network Selection
Create/Edit Missions
Spatially Feasible?
Demands Feasible? (Estimated)
No
No
Yes
Yes
Simulation Errors
No
Visualizations & Post-processing
Simulation
Yes
Demands Feasible?
Manifest Demands
Pack Demands

NETWORK TAB

The Network Tab serves as a visualization of the static supply network. On the left side of the screen, the scenario type can be changed to filter the visible nodes and edges. Below, the visibility of the nodes and edges can be toggled manually. When a node or edge is selected from the list, detailed information is displayed below the network visualization and superposed on the network.

- **Scenario Type:** Sets the default visibility of nodes and edges.
  - ISS scenario uses Earth surface and orbital nodes
  - Lunar scenario uses Earth and Moon surface and orbital nodes
  - Moon-only scenario uses Moon surface and orbital nodes
  - Martian scenario uses Earth, Moon, and Mars surface and orbital nodes
  - Mars-only scenario uses only Mars surface and orbital nodes
  - Solar System scenario uses all available nodes

On the right side of the screen, an image of the network is displayed, superposed with positions of surface nodes and representations of the orbital nodes. Surface node locations are calculated based on the Lambert-Azimuthal projection. Orbital nodes and various celestial bodies are relatively sized based on arctangent filters for minimum and maximum sizes and distances. Edges represent abstracted trajectories and are drawn as arcs between the origin and destination nodes but do not represent actual physics.

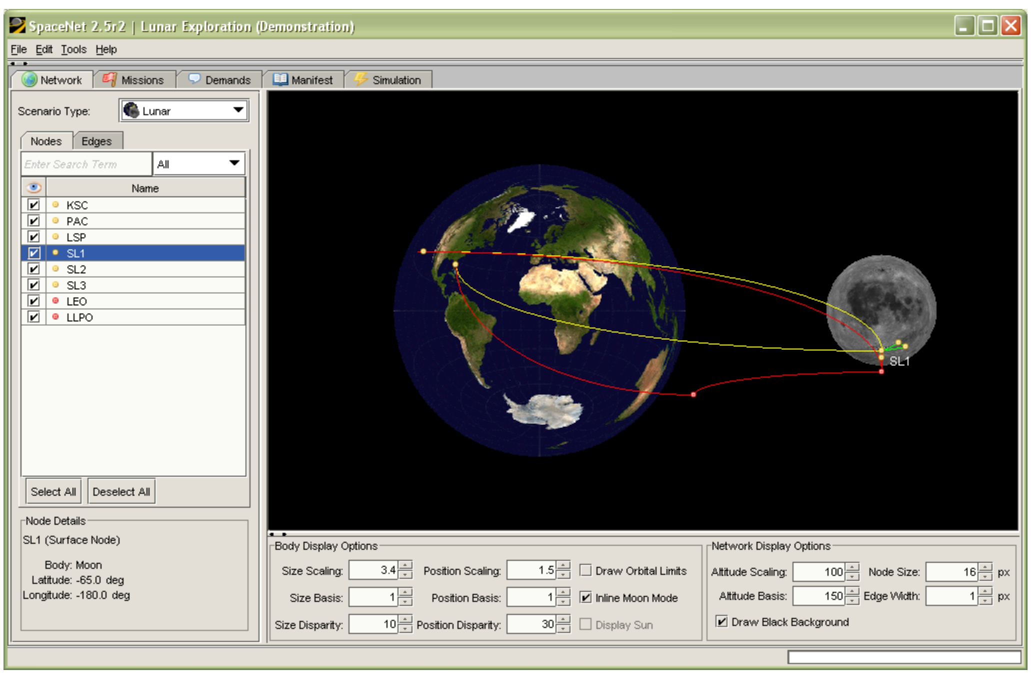


Figure 6: Network visualization for a lunar scenario with surface (yellow) and orbital (red) nodes, and surface (green), space (red), and flight (yellow) edges

There are several options below the network display to customize the visualization. The scaling of planetary bodies and their moons is calculated using an arctangent filter; the three inputs – scaling, basis, and disparity – change the filter parameters. There are also options to switch between inline mode (moons to the side or planets) and standard mode (moons below planets), switch the background color between black and white, and display or hide orbital limits.

If you attempt to uncheck a node or edge that is use in the scenario, you will receive a warning message indicating which events or missions reference the object. You must remove all references to a node or edge before it can be unchecked.

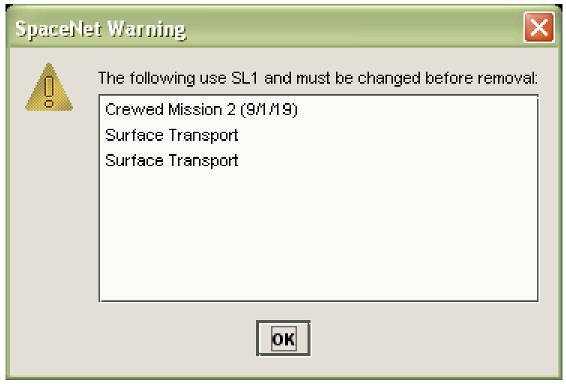


Figure 7: SpaceNet warning of a node or edge in use

MISSIONS TAB

The Missions Tab is used to create missions that operate over the network. As missions are created, they are listed on the left-hand side of the screen. This list can be directly edited to change both the start date and the mission name. The buttons at the bottom are used to add a new mission, edit an existing mission, copy a mission, and permanently remove a mission.

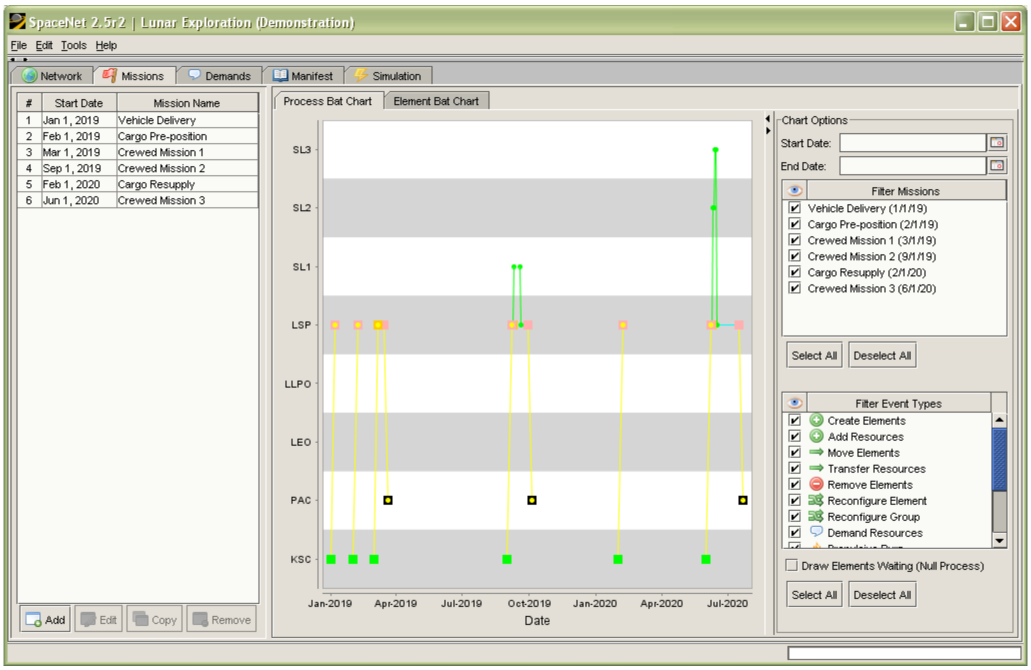


Figure 8: Missions tab showing an example scenario's process bat chart with six missions

On the right-hand side, there are two visualizations:

- **Process Bat Chart:** This chart shows how events operate on the time expanded network. The visible processes can be filtered by date, by mission, or by event type. Each event type has a representation on the chart, including the "null process" which was notated as a Wait Process in past versions of SpaceNet. Instantaneous events are represented with a square while finite-duration processes are represented with a line.
  - Create Elements, Add Resources: Green Square
  - Move Elements, Transfer Resources: Orange Square
  - Remove Elements: Black Square
  - Demand Resources: Blue Square
  - Reconfigure Element, Reconfigure Group: Pink Square
  - Propulsive Burn: Red Square
  - EVA: Cyan Square
  - Space Transport: Red Line
  - Surface Transport: Green Line
  - Flight Transport: Yellow Line
  - Crewed Exploration: Blue Line

- **Element Bat Chart:** This chart also shows the time-expanded network, but instead of viewing the events, the individual elements' locations are tracked over time. The chart can be filtered by date, and the visible elements can be selected. Elements locations are only recorded when entering or exiting nodes with straight lines connecting transportation between nodes independent of transport process.

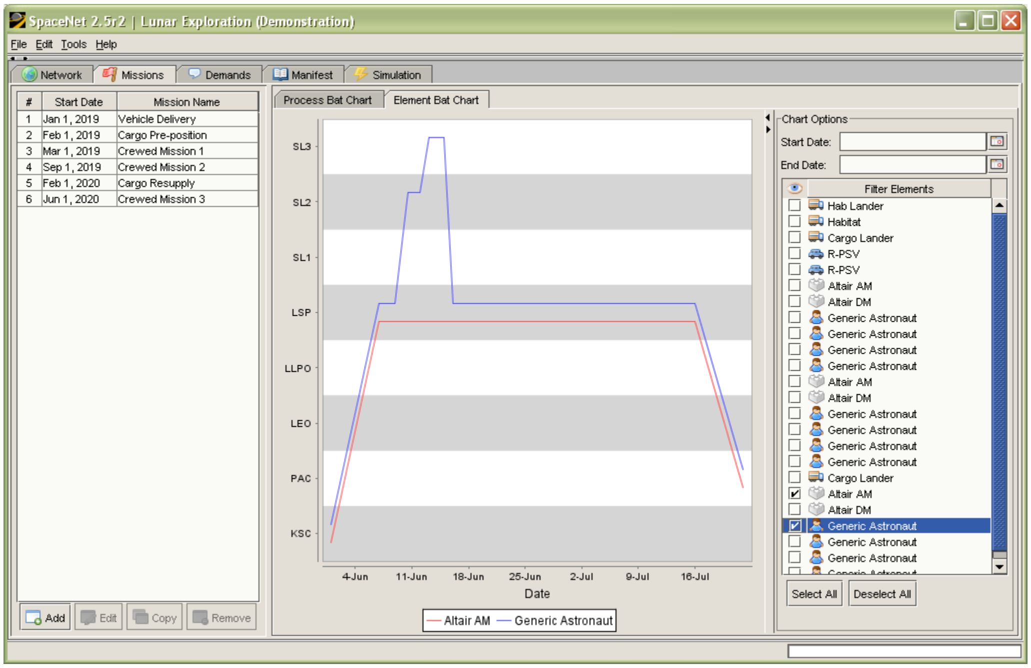


Figure 9: Element bat chart showing the location history of two elements

Mission Options

When adding a new mission or editing an existing mission, the left-hand side of the screen displays the mission options, while the right-hand side of the screen displays the process bat chart for the mission.

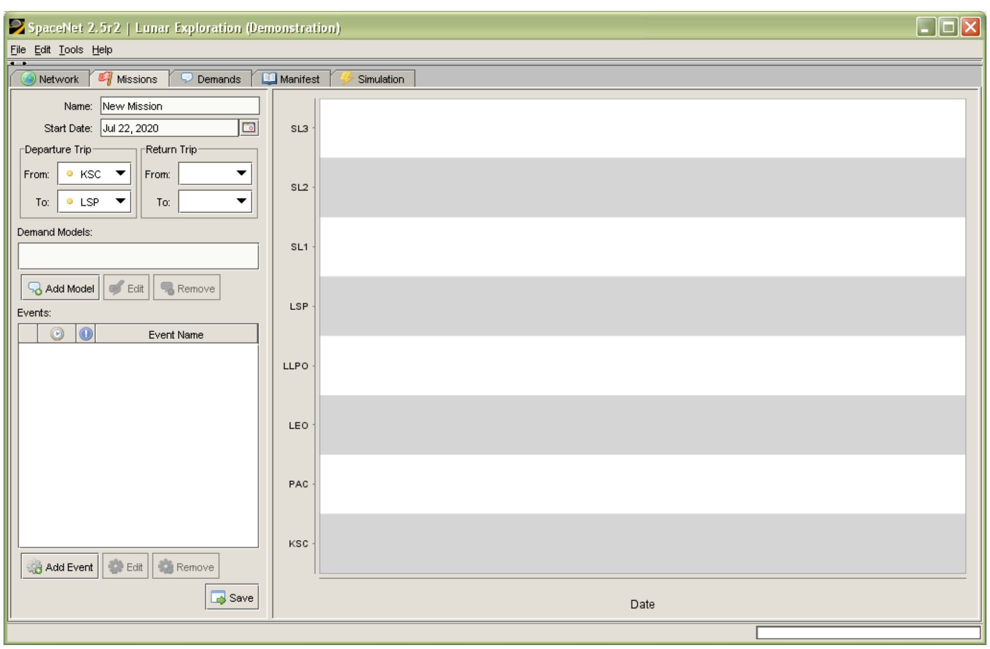


Figure 10: Screen while creating a new mission

- **Name:** Text used to identify the mission in the scenario. The first five letters will be used as a default preface for any events in this mission as well as any new elements instantiated in a Create Elements event. Elements used across several missions will be assigned a default prefix of the mission in which they were instantiated.
- **Start Date:** Date used as an epoch for events in the mission.
- **Departure Trip:** Origin and destination nodes used for mission-level demand models to identify the time of arrival and surface duration.
- **Return Trip:** Origin and destination nodes for crewed return trips used for mission-level demand models to identify the time of departure and dormant duration (time until next mission, if any exist). If a mission does not include a return trip (e.g. cargo delivery or resupply), set the origin and destination to blank.
- **Demands:** A list of mission-level demand models in effect.
- **Events:** List of mission events associated with this mission. The clock icon illustrates the relative execution time (in days after the mission start date), and the blue circle illustrates the priority of concurrent events. The event time (days relative to mission start), priority, and name can be directly edited by double-clicking.

Mission Demands

Mission-level demand models used to generate non-element specific demands during the mission. These demands are defined to be aggregated at the mission destination node at the time of arrival. There are four types of demand models currently available:

- **(Timed) Impulse Demand Model:** Creates a one-time demand for a set of resources scheduled for the first transportation arrival at the destination node. Add and remove rows using the “Add” and “Remove” buttons, change the resource type (generic, continuous, or discrete), and choose the resource from the appropriate columns, then type in the desired amount.

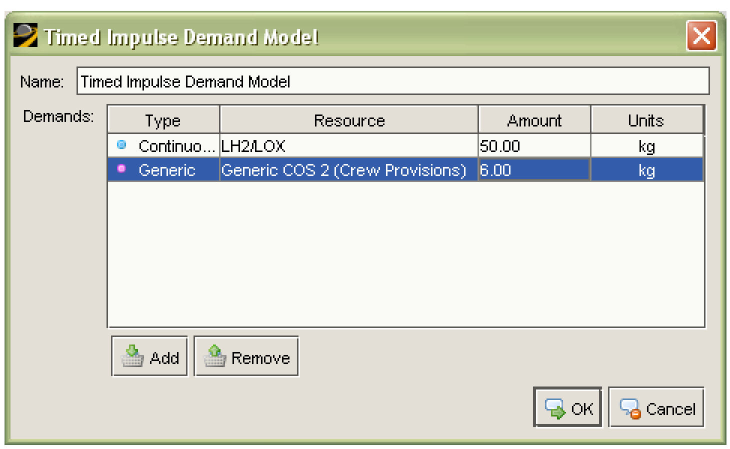


Figure 11: Example of an impulse demand model with two resources

- **Rated Demand Model:** Creates a demand for a set of resources based on daily rates and the mission duration. Add and remove rows using the “Add” and “Remove” buttons, change the resource type (generic, continuous, or discrete), and choose the resource from the appropriate columns, then type in the desired daily rate.

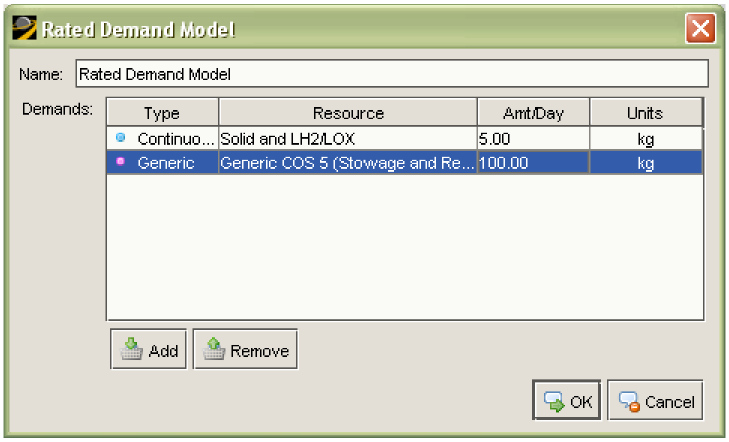


Figure 12: Example of a rated demand model with two resources

- **Consumables Model (Open ECLSS):** Creates a demand for resources based on the NASA Space Logistics Consumables Model with parameters adjusted for a typical open ECLSS system. The following fields are inferred from the mission events:
  - Crew Size: Sum of crew members instantiated in this mission with create events
  - Exploration Duration: Sum of crewed exploration durations
  - Dormant Duration: Mission duration minus exploration and transit durations
  - Transit Duration: Sum of all transport durations
  - Transit EVAs: Sum of all EVAs while in transit
  - EVAs per Week: Sum of all EVAs divided by mission duration
  - Crew per EVA: Sum of crew members in first EVA or Exploration
  - EVA Duration: Duration of first EVA or Exploration
  - Sabatier Reaction: Disabled due to irresolvable circular reference in Excel model

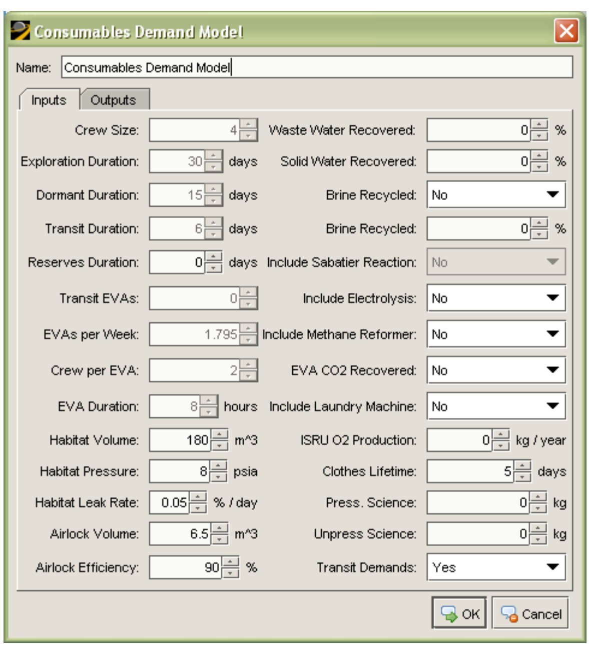


Figure 13: Example consumables model inputs

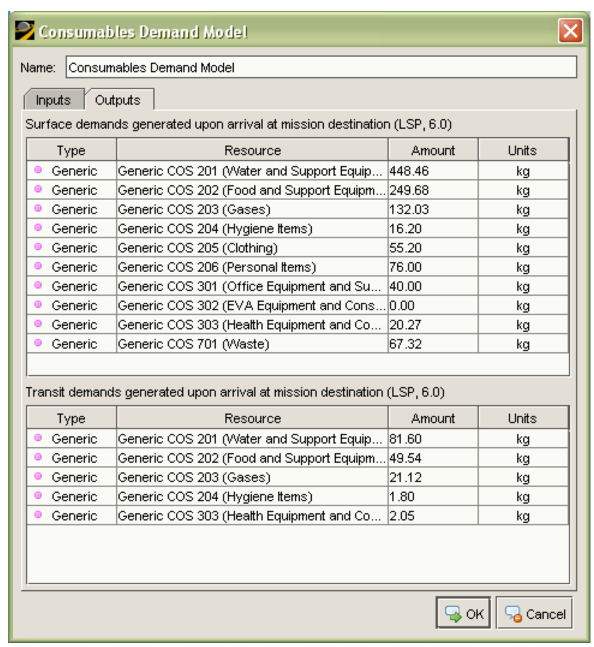


Figure 14: Example consumables model outputs

- **Consumables Model (Closed ECLSS):** Creates a demand for resources based on the NASA Space Logistics Consumables Model with parameters adjusted for a typical closed ECLSS system. All fields are inferred in the same manner as the open ECLSS consumables model.

Mission Events

The mission events form the basis of the simulation. There are five "core" events (with a few variations) of which all others are comprised of (Create, Move, Reconfigure, Demand, and Remove), two derived events approximated as instantaneous (Propulsive Burn, Crewed EVA), one process of finite duration (Crewed Exploration), and three transports of finite duration with an origin and destination (Surface, Space, Flight).

All events have a few items of common information:

- **Name:** A customizable name to reference the event. Note that if the mission-prefix is removed (text before vertical bar), it will no longer be updated when the mission name changes.
- **Node:** The location of the event, used to filter selectable elements. User-generated events can only occur at nodes, but auto-generated events can occur at either nodes or edges.
- **Time:** The execution time of the event, relative to the start of the mission.
- **Priority:** The priority of this event over other events (lowest number first). User-generated events have priorities between 1 and 5. Events auto-generated from composite events, processes, and transports have priority less than or equal to 0. Execution order for events with the same priority cannot be predicted.

When creating new events, they are pre-filled with data as from the currently-selected event, or the last event in the mission if none are selected:

- **Time:** Same time as selected event, or after selected process or transport
- **Priority:** Priority of selected event + 1 or 1 if selected process or transport
- **Node:** Same location as selected event, or destination of selected transport

Also, when the edit event dialog is opened or the node, time, or priority is changed, SpaceNet will use pre-simulation to display the current state of the network to attempt to prevent errors. If an error condition does happen, an error icon is displayed next to the event, and the error message is visible via a roll-over message and at the top of the event dialog.

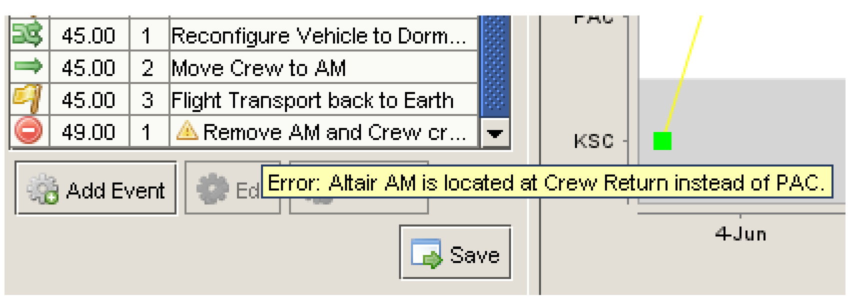


Figure 15: Example of an error condition displayed in the events list

Create Elements

The Create Elements event is used to instantiate elements. The element library is displayed on the right-hand side of the dialog, and is filterable by element type and search bar. When an element is added from the element library, the data source is queried to retrieve the full element information.

Elements can either be instantiated at a node or in an existing carrier. Information is displayed to track the capacities of carriers (mass, volume, and crew capacities). For nodes, these bars measure the total mass, volume, and number of crew at the node.

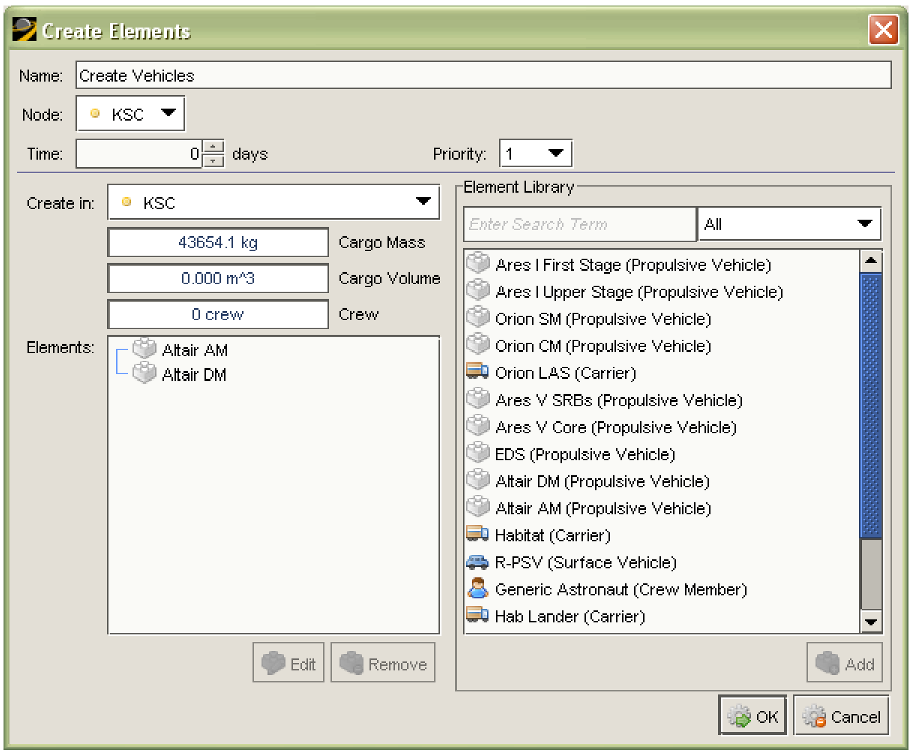


Figure 16: Example of an event instantiating several elements at a node

Elements can also be edited from the Create Elements dialog by clicking the "Edit" button below the list of elements. The preferred way to edit elements, however, is to edit and reload the data source (See Data Source).

Move Elements

The Move Elements event is used to instantaneously move a set of elements to the current node or existing carrier. To select elements to move, check their entry. As in the Create Elements event, the cargo mass, cargo volume, and crew capacities are displayed and updated.

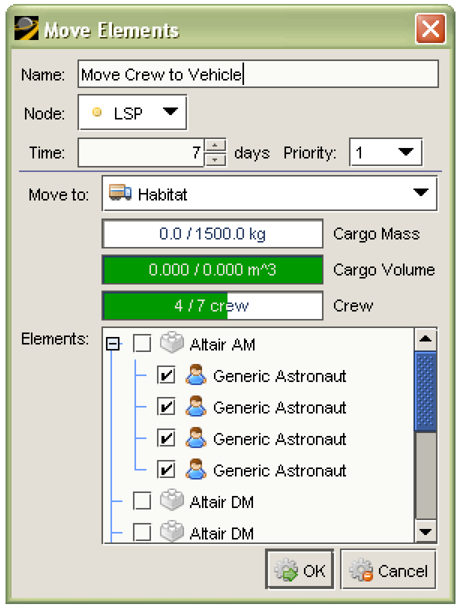


Figure 17: Example event moving elements to a node

Add Resources

The Add Resources event is used to create resources inside an existing container. The container cargo environment, and mass and volume capacities are displayed and updated as more resources are added.

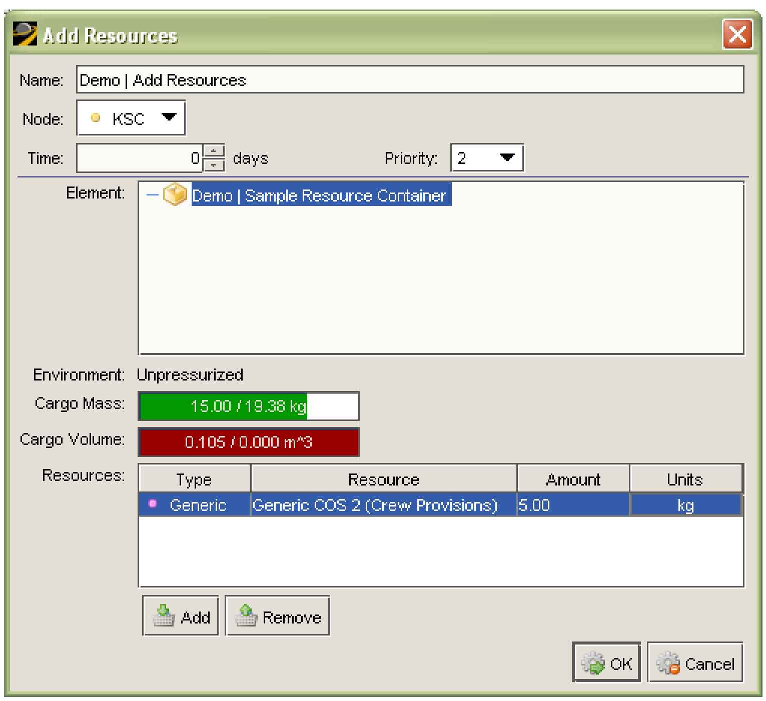


Figure 18: Example event adding resources to a container

Transfer Resources

The Transfer Resources event is used to transfer resources from one container to another. A list of the current resources and amounts is displayed as the "Contents." To transfer an amount of a resource, type in the amount desired in the "Transferred" column. As in the Add Resources event, mass and volume capacities are displayed and updated as resources are transferred.

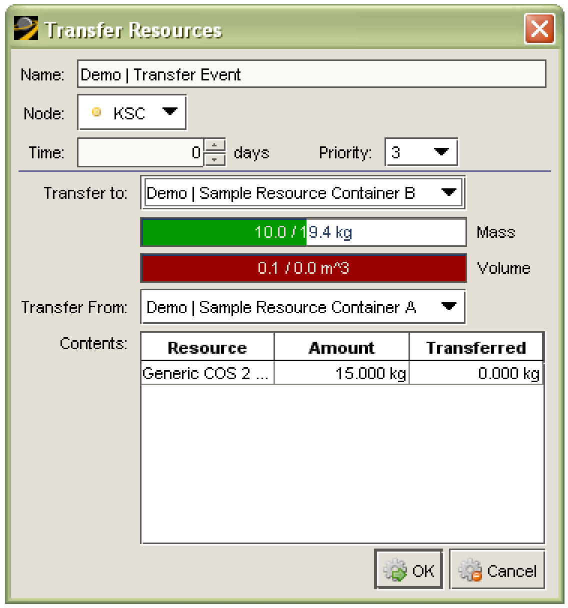


Figure 19: Example event transferring resources from one container to another

Remove Elements

The Remove Elements event removes elements from the simulation to prevent unintended demands from being aggregated (crew at the return site, for example). If the removed element contains any nested elements at execution time, they also will be removed from the simulation.

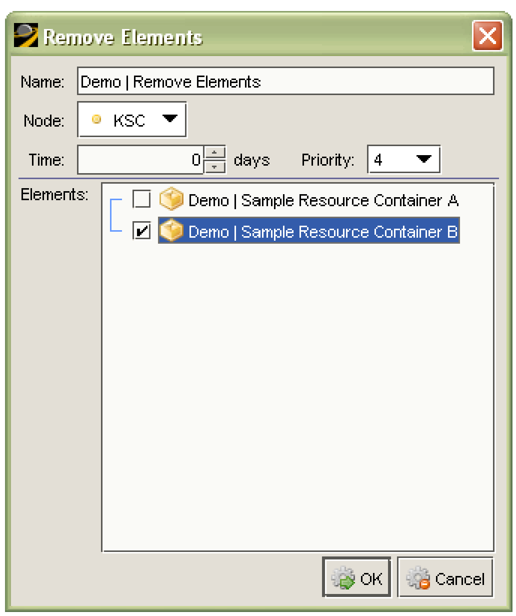


Figure 20: Example event removing an element from the simulation

Demand Resources

The Demand Resources event generates a demand for resources by a selected element. A demand event differs from a demand model in that if there are insufficient resources to satisfy a demand event, no errors will be raised until simulation (i.e. it omits the demand aggregation cycle). For this reason, demand events are not often used in scenarios.

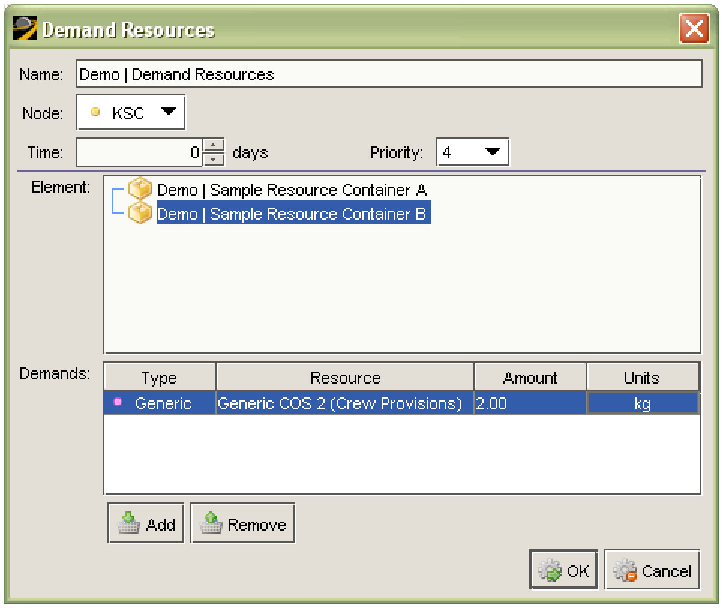


Figure 21: Example event generating demands from an element

Reconfigure Element

A Reconfigure Element event is used to change the operational state of an element to one of its specified states. Only one element may be changed at a time with this event since elements may have more than one state of the same type (e.g. a crew member may have a nominal "Active" state and an EVA "Active" state).

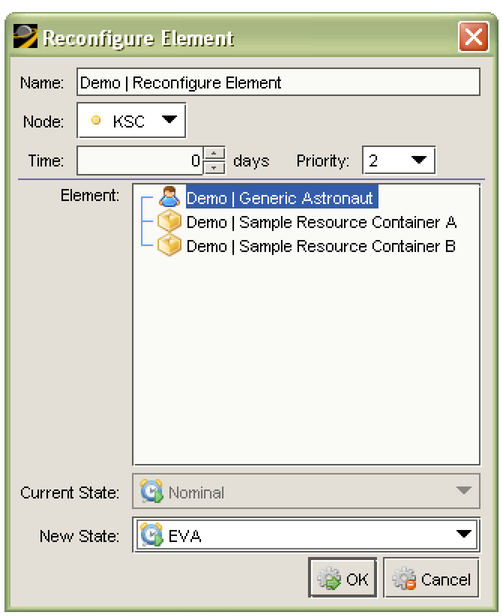


Figure 22: Example event reconfiguring an element to a different state

Reconfigure Group

A Reconfigure Group event is used to reconfigure a group of elements to a specific type of state. There are five types of states: Active, Special, Quiescent, Dormant, and Decommissioned. If an element has more than one state of the same type, the specific one that is chosen is unknown. If an element does not have a state of the specified type, nothing happens.

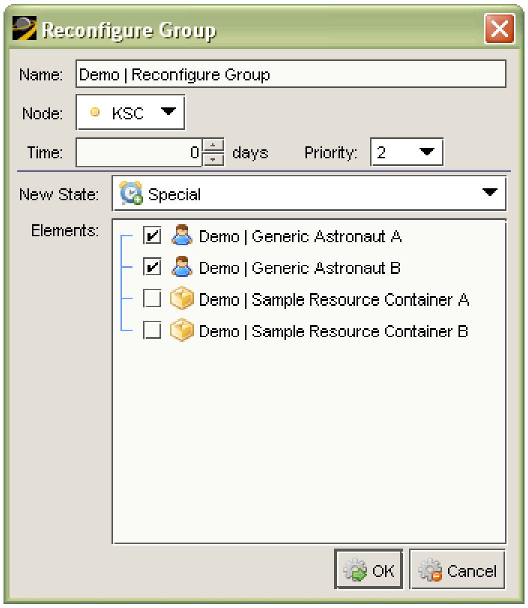


Figure 23: Example event reconfiguring a group of elements

Propulsive Burn

A Propulsive Burn event is used to perform an impulsive burn by one or more elements to achieve a target delta-v (change in velocity, meters per second). Propellant is expended according to the rocket equation.

There are two types of burns: OMS (Orbital Maneuvering System), and RCS (Reaction Control System), and only propulsive vehicles with the appropriate capabilities can perform the burns. Some propulsive vehicles may only have one or the other capability, while others have both.

Equation 1: Ideal rocket equation solved for delta-v achievable from fuel mass

$$\Delta V = I_{sp} \cdot g_0 \cdot \log\left(\frac{m_0}{m_0 - m_{fuel}}\right)$$

m_0 : Initial stack mass

m_{fuel} : Fuel mass

g_0 : Acceleration of gravity

I_{sp} : Vehicle specific impulse

ΔV : Change in velocity

Equation 2: Ideal rocket equation solved for fuel mass required to achieve delta-v

$$m_{fuel} = m_0\left(1 - \exp\left(\frac{-\Delta V}{I_{sp} \cdot g_0}\right)\right)$$

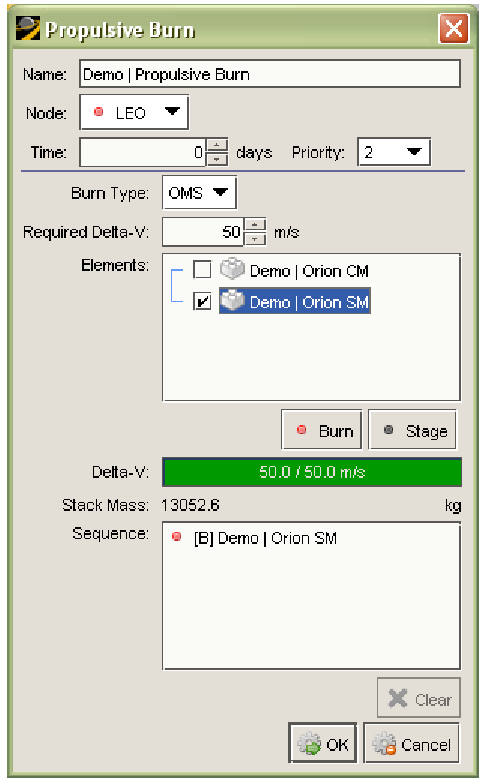


Figure 24: Example event with one burn to achieve a specified delta-v

The event allows the selection of a set of elements (the stack), from which the total mass is calculated for the rocket equations. Next, a burn-stage sequence is specified to set the order of burns by different elements (if required), and the intermittent staging, or removal, of spent elements. After each burn, the appropriate amount of propellant mass is removed from the target propulsive vehicle up to its current remaining amount.

Crewed EVA

A Crewed EVA event schedules one extra-vehicular activity. During the EVA, the selected crew members are moved external to the habitat, and reconfigured to an EVA state, if specified. After the EVA duration, the crew members are returned to the crew habitat and reconfigured to their previous states. Additional demands can also be specified to capture airlock losses or additional demands not handled by the crew's EVA states.

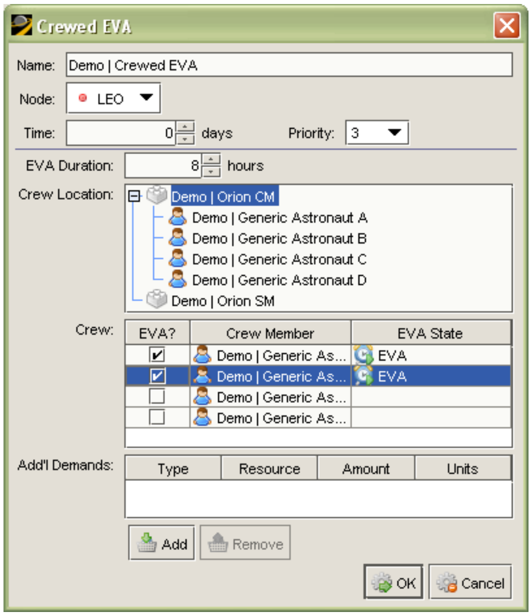


Figure 25: Example crewed EVA with two crew members participating and no additional demands

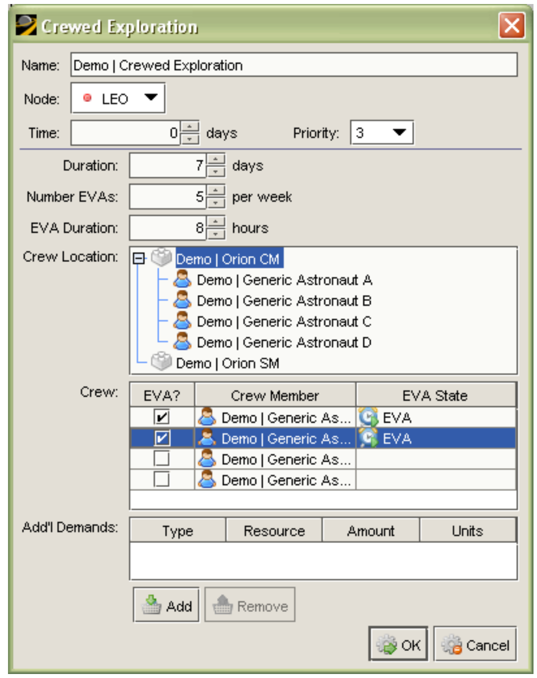


Figure 26: Example crewed exploration that will schedule five EVAs over the course of a week

Crewed Exploration

A Crewed Exploration process schedules a number of Crewed EVA events. The inputs are identical to the Crewed EVA event, except that there are parameters to set the duration of the exploration, and the number of EVAs per week. Each scheduled Crewed EVA uses the same EVA duration, crew, and additional demands. The EVAs are scheduled with equal time separation from each other and from the start and end of the process. For example, if five EVAs are to be scheduled over seven days, the first will be at time 1.17, the second at 2.33, and so on up until the last at time 5.83.

Surface Transport

A Surface Transport event moves a surface vehicle (and its nested elements) across a surface edge at a constant speed. When the event executes, the surface vehicle is reconfigured to its given transport state and moved to the edge. The arrival event at the destination node is scheduled according to the edge's distance, the transport speed, and the duty cycle (percentage of the time the vehicle is in motion).

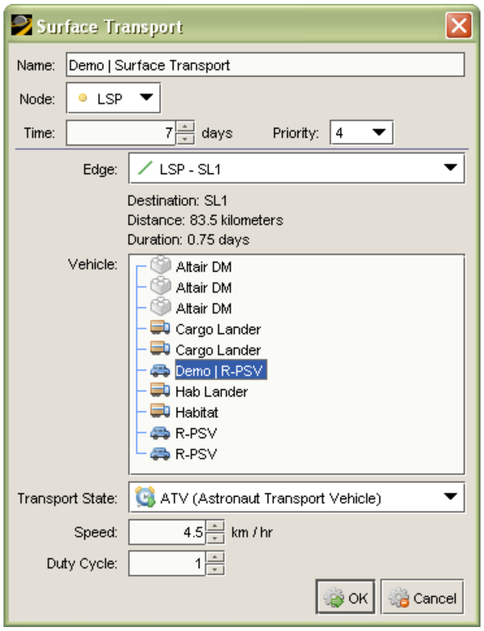


Figure 27: Example surface transport moving a surface vehicle to a surface location

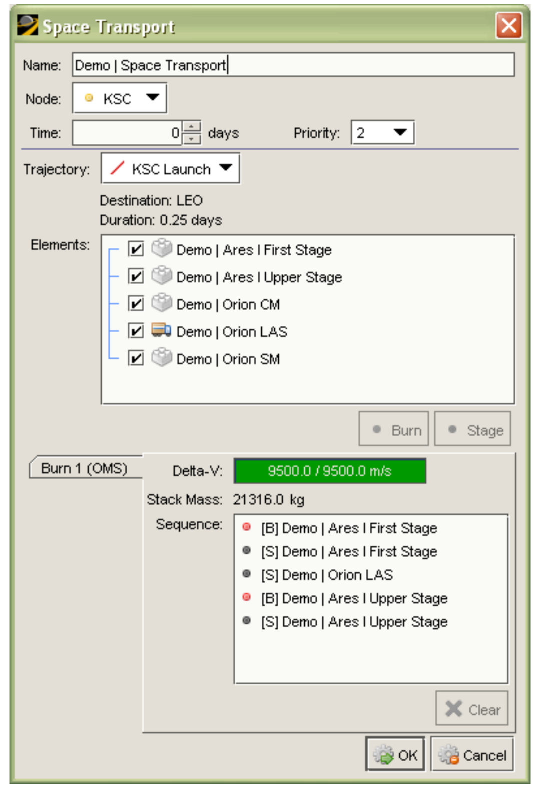


Figure 28: Example space transport using a first and upper stage to achieve the target burn

Space Transport

A Space Transport event moves a set of elements across a space edge using a series of propulsive burns. When the event executes, the selected elements are moved to the edge, and the burns and arrival at the destination node are scheduled.

Flight Transport

A Flight Transport moves a set of elements up to a mass capacity and crew limit across a flight edge. Only nested elements are counted against the constraints, so the specific carrier element used is unimportant. When the event executes, the selected elements are moved to the edge, and the arrival at the destination node is scheduled.

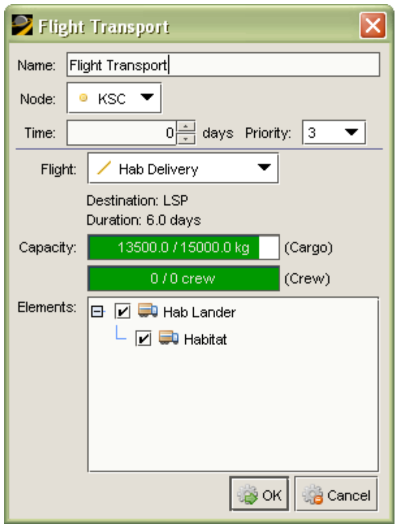


Figure 29: Example event delivering a surface vehicle

ELEMENT EDITOR

After creating elements in a Create Event, it is often useful to inspect their data, and potentially edit it. Ideally, elements should be edited in the database rather than when instantiated, but there are some occasions where it may be quicker.

All elements have the following fields:

- **Name:** A name to reference the element. Note that if the mission-prefix is removed (text before vertical bar), it will no longer be updated when the mission name changes. The element's name is also the only field that is *not* updated when updating elements from the data source.
- **Icon:** A visual representation of the element used in many menus throughout SpaceNet, including the simulation animation. Each element type has a default icon but it can be changed to a different icon available in the drop-down menu.
- **Class Of Supply:** The class of supply for this element. Most elements created in events will likely be derivatives of COS 5 (Stowage and Restraint), COS 6 (Exploration and Research), COS 8 (Habitation and Infrastructure), or COS 9 (Transportation and Carriers).
- **Environment:** The environment in which the element must be nested. Unpressurized elements may be positioned directly at nodes or within unpressurized cargo carriers, while pressurized elements may only be positioned within pressurized cargo carriers.
- **Accommodation Mass:** [Not currently enabled] The amount of additional class of generic COS 5 required to nest this element inside a carrier (kilograms).
- **Mass:** The element's dry mass (kilograms).
- **Volume:** The element's volume (cubic meters). Most elements are assumed to be rigid, so the volume is not affected by contents.
- **States:** A list of the possible states for the element. State types include: Active (nominal operation), Quiescent (reduced operation), Dormant (not operational), and Decommissioned (permanently disabled and available for spares scavenging).
- **Initial State:** The state of the element when it is created.
- **Parts:** A list of spares (including quantity) for the element. Elements with the same resource type for a part are said to have "common spares."

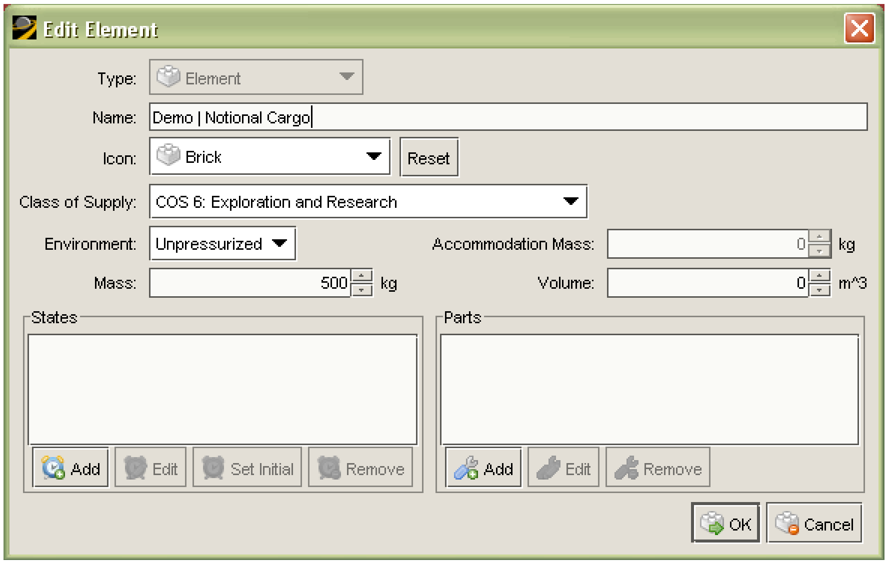


Figure 30: Example element

Resource container elements have a few additional fields associated with the resources they can hold:

- **Max Cargo Mass:** The maximum mass (kilograms) of cargo including nested elements.
- **Max Cargo Volume:** The maximum volume (cubic meters) of cargo.
- **Cargo Environment:** The environment of the cargo area. Unpressurized container environments can carry only unpressurized resources, while pressurized container environments can carry both unpressurized and pressurized resources.
- **Resources:** A listing of all the initial resources inside the container.

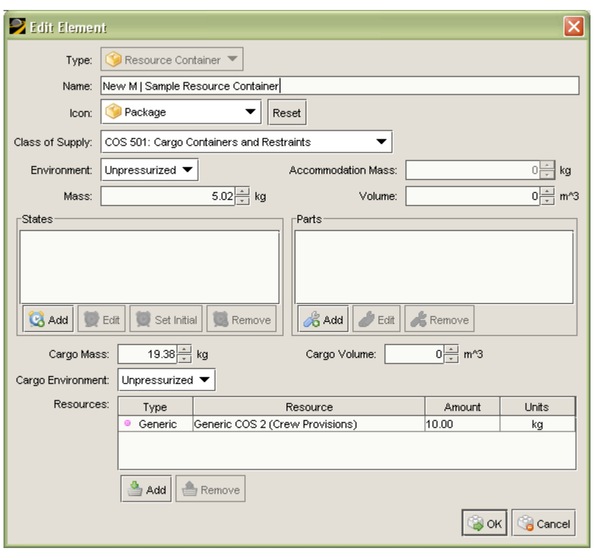


Figure 31: Example container element

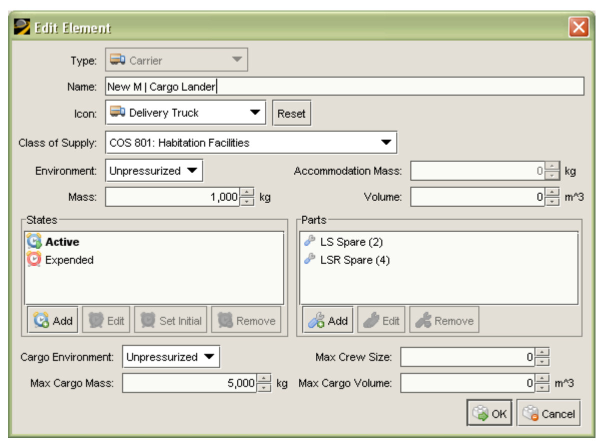


Figure 32: Example carrier element

Carrier elements have a few additional fields associated with the cargo they can carry:

- **Cargo Environment:** The environment of the cargo area. Unpressurized cargo environments can carry only unpressurized elements, while pressurized cargo environments can carry both unpressurized and pressurized elements.
- **Max Crew Size:** The maximum number of crew members directly inside the carrier.
- **Max Cargo Mass:** The maximum mass (kilograms) of cargo including nested elements.
- **Max Cargo Volume:** The maximum volume (cubic meters) of cargo.

Propulsive vehicles have a few additional fields associated with their propulsive capabilities:

- **OMS Capabilities:** Checked if the element can perform OMS burns.
- **RCS Capabilities:** Checked if the element can perform RCS burns.
- **Isp:** The element's engine specific impulse (seconds).
- **Shared Fuel Tank:** Checked if the element has one fuel tank to supply both OMS and RCS burns.
- **Fuel Type:** The resource to supply the engine.
- **Max Fuel:** The maximum amount of fuel stored in an onboard tank.
- **Initial Fuel:** The initial amount of fuel stored in an onboard tank.

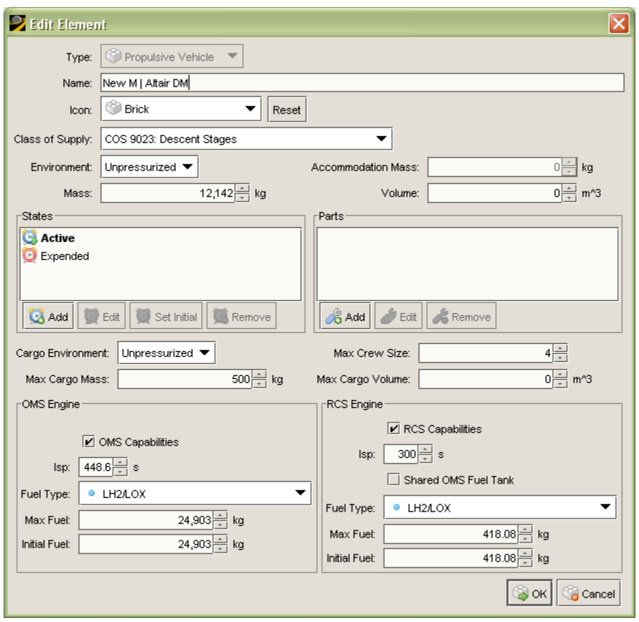


Figure 33: Example propulsive vehicle

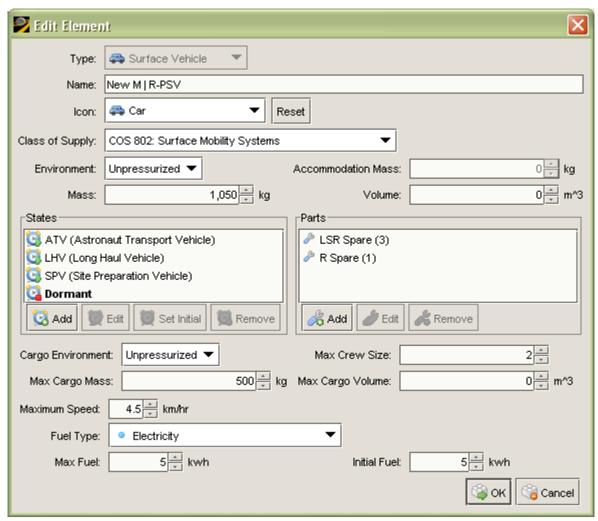


Figure 34: Example surface vehicle

Surface vehicles have a few additional fields associated with their surface mobility:

- **Maximum Speed:** The largest speed the vehicle can travel at (kilometers per hour).
- **Fuel Type:** The resource to supply the engine.
- **Max Fuel Amount:** The maximum amount of fuel stored in an onboard tank.
- **Initial Fuel Amount:** The initial amount of fuel stored in an onboard tank.

Parts

When adding parts to elements, the discrete resources must be turned into a part application. This requires adding a few extra fields:

- **Part**: The discrete resource to use as the part type.
- **Quantity**: The quantity of the part that the element uses.
- **Mean Time to Failure**: The mean time (hours) to failure for this part for advanced demand modeling. If the part never fails, input 0.
- **Mean Time to Repair**: The mean time (hours) to repair this part, used in the repairability analysis. If the part is not repairable, input 0.
- **Mean Repair Mass**: The mass (kilograms) of generic COS 4 needed to repair this part instead of using a spare, used in the repairability analysis.
- **Duty Cycle**: The fraction of time this part is in use, used to scale the mean time to failure.

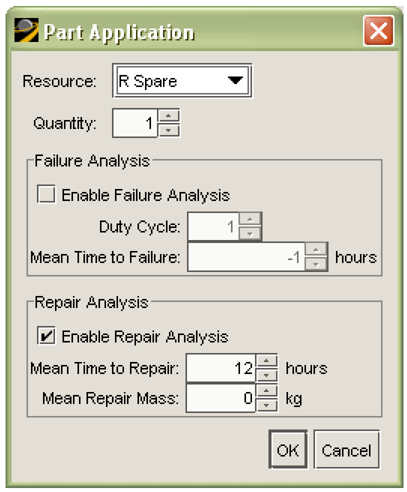


Figure 35: Example part application

States and Demand Models

Element's states also incorporate demand models, so that when an element is in a particular state, only those demand models are in effect for generating demands.

- **Name:** A name to reference the state.
- **Type:** There are four types of states, but three (Active, Quiescent, and Dormant) are operationally identical and only serve as different labels. The fourth state type, Decommissioned, serves a special purpose to signify that availability to scavenge for spares. When an element is decommissioned, all of its parts become accessible to demands at that node.
- **Duty Cycle:** The fraction of the time the demand models are effective (scales the demands).
- **Demand Models:** A list of element-centric demand models.

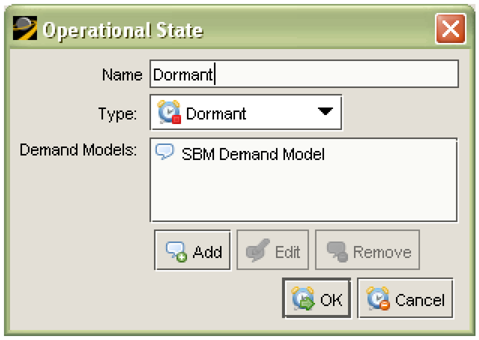


Figure 36: An example active state with one demand model

There is a wide variety of element-centric demand models, and several are very similar to mission-centric demands. Element demand models, however, produce demands over every non-zero time step. For example, if 2.5 days have passed since the last event was executed; all the elements are requested to generate demands for the 2.5 days.

- **(Timed) Impulse Demand Model:** Generates a one-time demand for fixed set of resources. It is executed the first time its state is requested to generate demands and never again.

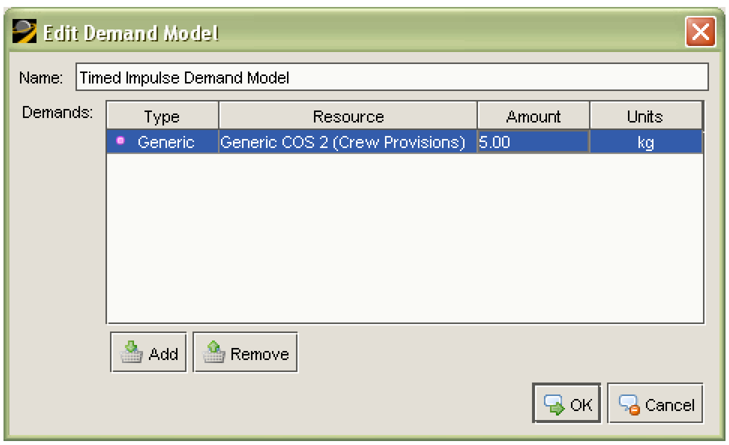


Figure 37: Example (timed) impulse demand model with two resources

- **Rated Demand Model:** Generates demands at a constant rate (amount per day).

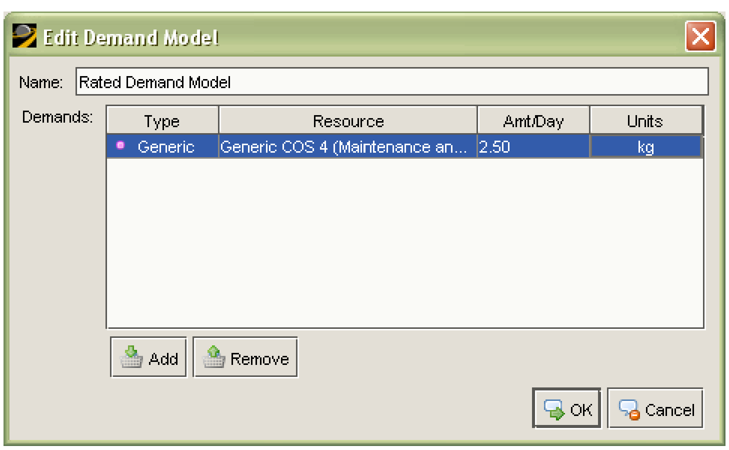


Figure 38: Example rated demand model with two resources

- **Sparing by Mass Model:** Generates demands for pressurized and unpressurized spares based on a percentage of the element's dry mass per year. If any parts are defined for the element, a fraction of the demands proportional to the part's fraction of the element's dry mass will be generated of the part's resource type. All other spares are generated with generic COS 4 resources.

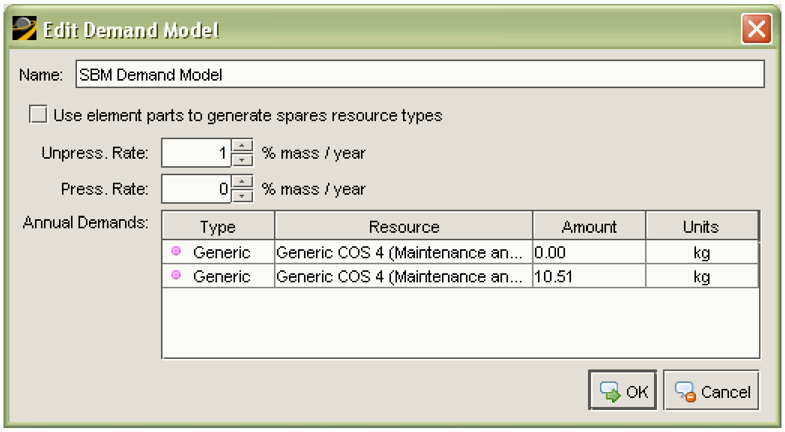


Figure 39: Example sparing by mass model for an element

DEMANDS TAB

The demands tab is used to inspect and analyze the demands generated throughout a scenario, as constructed in the Missions Tab. Several visualizations are available for viewing aggregated demands as well as demands at the element, mission, and location levels. Also, there are tools to aid in commonality and repairability analyses. There are no directional dependencies between the Missions Tab and the Demands Tab, so the user can switch between them at will.

Demand Options

There are a couple of options that can be used to tweak the outputs of the demands, which are available at the left-hand side of the Demands Tab. The first set deal with discretization, aggregation, and scavenging (also available in the scenario options dialog).

- **Item Discretization:** Method used to treat discrete resources.
  - None: No attempt is made to keep discrete resources in unit amounts (they are treated as continuous resources).
  - By Element: Discrete resources are pooled by each element using an aggregation option before unit demands are created.
  - By Location: Discrete resources are pooled at each location using an aggregation option before unit demands are created.
  - By Scenario: Discrete resources are pooled at the scenario-level (for all elements) using an aggregation option before unit demands are created.
- **Aggregation Option:** Method used to aggregate discrete items during item discretization.
  - Ahead of Demands: Unit demands are created before demands sum to a unit amount, always over-estimating the demands.
  - Behind Demands: Unit demands are created after demands sum to a unit amount, always under-estimating the demands.
- **Scavenge Spares:** If checked, decommissioned elements are used as a source of spares. More detailed information can be found on the "Commonality Analysis" tab.

The next two options manage the visualizations appearing within the demands tab, but do not alter the execution of the simulation for later analysis.

- **Estimate Logistics Container Masses:** If checked, an amount of additional generic COS 5 demands based on the demanded resource's packing factor is added to the charts to represent an estimated mass of logistics containers yet to be created in the manifesting process.
- **Consume Existing Resource:** If checked, any pre-existing resources (e.g. initial resources in containers) will be accessed for demands. By providing these initial resources, the resulting "unsatisfied" demands will drop accordingly in all visualizations.

The last option provides the ability to export demands to file for additional analysis or post-processing. Either the raw or aggregated demands can be written to file, the raw demands being each request by an element or mission, aggregated demands being similar to the supply network visualization discussed below.

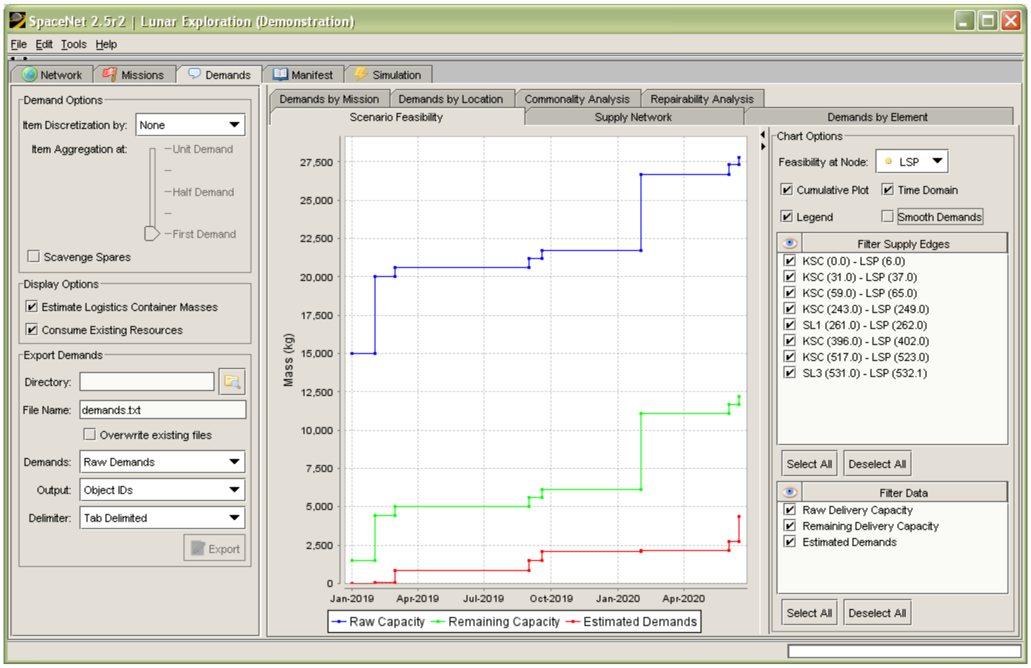


Figure 40: The demands tab, with the demand options on the left-hand side

Demand Visualizations

A wide variety of demand visualizations provide the user feedback that the correct demands are being generated and that the scenario is feasible from an estimated demands perspective.

Scenario Feasibility

The Scenario Feasibility chart displays the raw delivery capacity, remaining delivery capacity (after existing events), and estimated demands at a node. Demands are estimated because they do not take into account packing and manifesting inefficiencies. In the cumulative view, if the red estimated demands line crosses the green remaining delivery capacity line, the scenario is infeasible at that location.

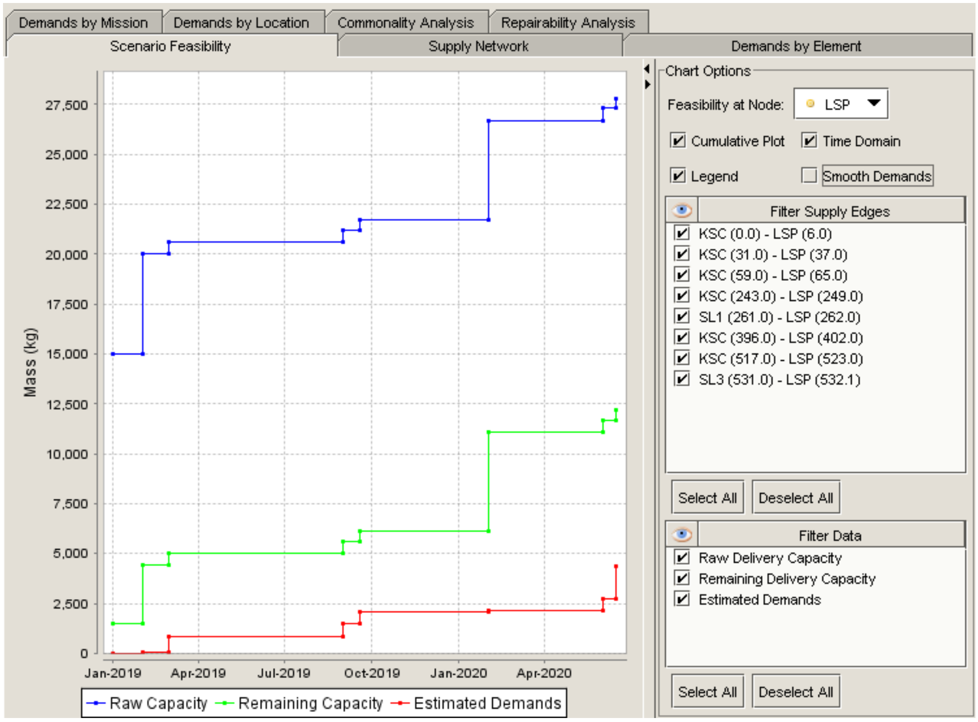


Figure 41: Scenario feasibility chart with cumulative data over time

Options:

- **Feasibility at Node:** Change the node to inspect feasibility. The displayed demands are those that are generated by elements located at that node, or missions with a destination of that node.
- **Cumulative Plot:** If checked, capability and demands will increase cumulatively for each transport, if unchecked, capability and demands will be shown for each transport.
- **Time Domain:** If checked, cumulative capability and demands are plotted over time instead of over each transport.
- **Legend:** Display the legend below the chart of the selected categories.
- **Smooth Demands:** As the simulation is the product of a discrete-event simulation, there are discrete jumps in capacity and demand levels. This option smooths the demands line (assuming linear demand generation).
- **Filter Supply Edges:** Filter the display of the supply edges (transports) identified in the scenario. Cumulative totals are not affected by deselecting items.
- **Filter Data:** Filter the display of the Raw Delivery Capacity (total cargo capacity of all carriers involved in the transport), Remaining Delivery Capacity (total cargo capacity less any nested elements), and Estimated Demands (estimate of demand mass during the time).

Supply Network

The supply network visualization displays the transports and aggregated demands in the time-expanded network. The lines are sized proportionally to the amount of remaining capacity (green for positive margins, red for negative, infeasible margins). Each line has a blue dot at its destination, indicating the potential to supply resources at that point. The blue dots are sized proportionally to the total mass of aggregated, estimated demands at a location. Demands can be satisfied by any resources delivered before it is aggregated (lines to the left).

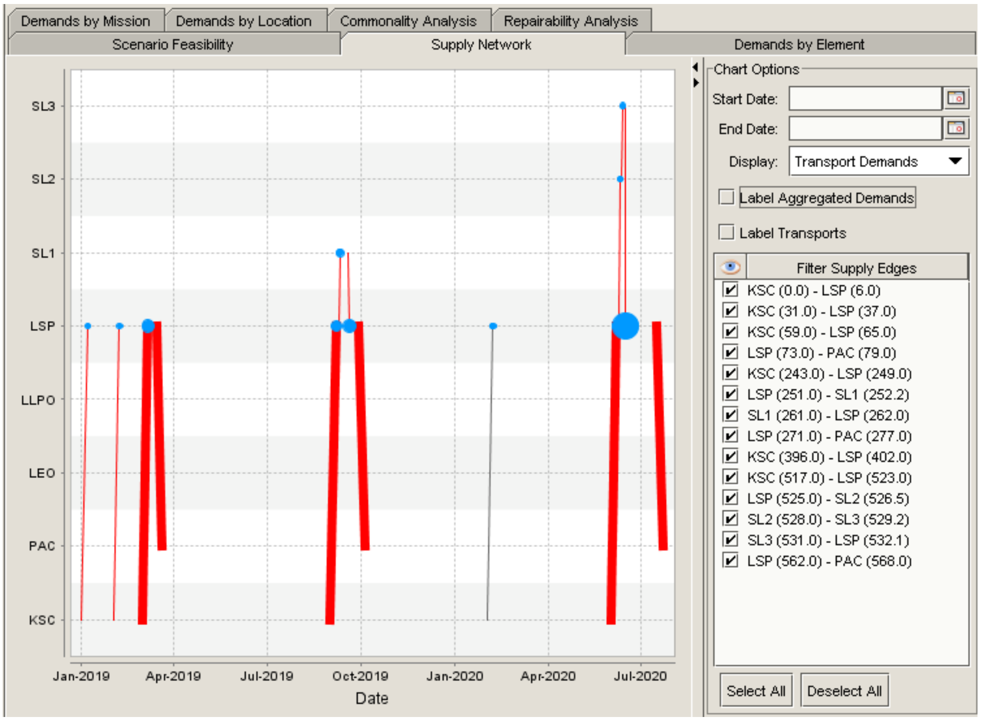


Figure 42: Example supply network with infeasible (red) transports

Options:

- **Start Date:** Sets a lower-bound for the display.
- **End Date:** Sets an upper-bound for the display.
- **Display:** Choose whether to display the Transport Capacity (remaining cargo capacity of all carriers in the transport), Transport Demands (estimated demands during the transport event), or Net Transport Capacity (capacity less estimated demands).
- **Label Aggregated Demands:** If checked, a text label is placed next to each dot indicating the total mass of aggregated demands.
- **Label Transports:** If checked, a text label is placed next to each line indicating the selected display option value.

Demands by Element

The Demands by Element chart allows a closer inspection at the origin of demands on a per-element basis. If only one element is selected from the list, the demands are plotted over time. If more than one element is selected, the demands are aggregated over the entire scenario.

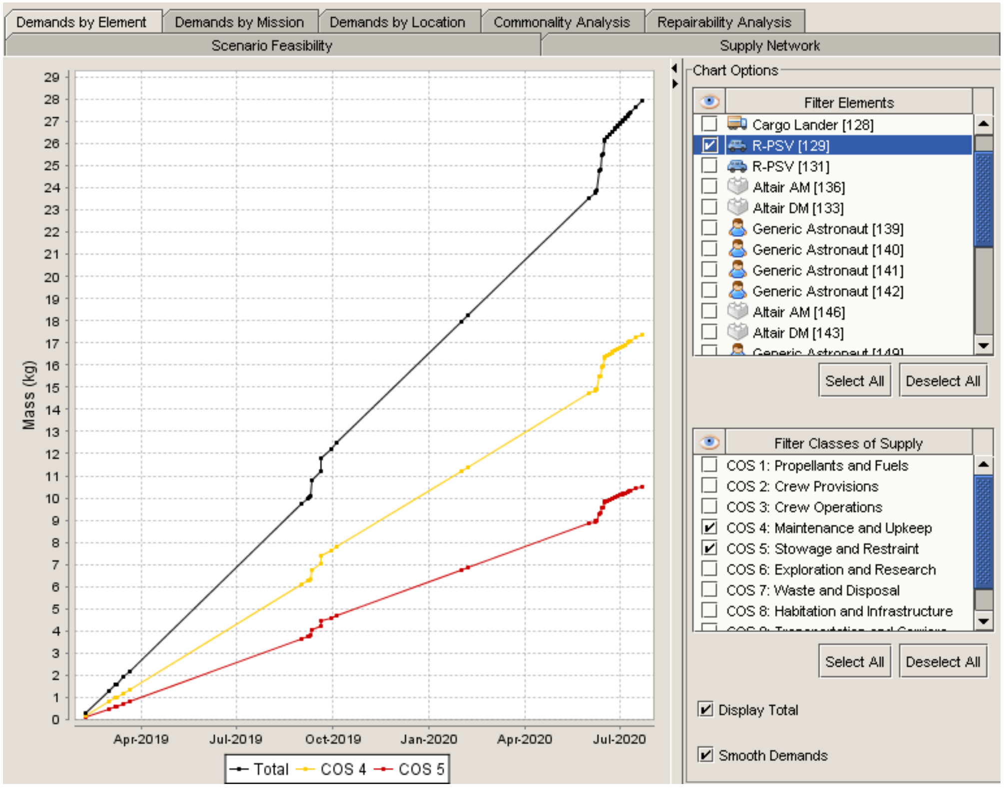


Figure 43: Demands for a single element are plotted over time

Options:

- **Filter Elements:** Filter the display of elements created in the scenario. If one element is selected, the demands are plotted over time, otherwise the demands are aggregated over the entire scenario.
- **Filter Classes of Supply:** Filters the display of the 10 base classes of supply. Demands for more specific classes of supply (e.g. COS 401) are lumped into their parent class of supply.
- **Display Total:** Display a black line representing the total of all classes of supply. (Only enabled for single-element analysis)
- **Smooth Demands:** As the simulation is the product of a discrete-event simulation, there are discrete jumps in demand quantities. This option smooths the demands line assuming linear demand generation. (Only enabled for single-element analysis)

Demands by Mission

The Demands by Mission chart breaks the demands into approximated periods by mission. Each mission's duration is defined as the time from its start date until the start of the next mission. If missions are concurrent, the demands are associated with the most recently-starting mission.

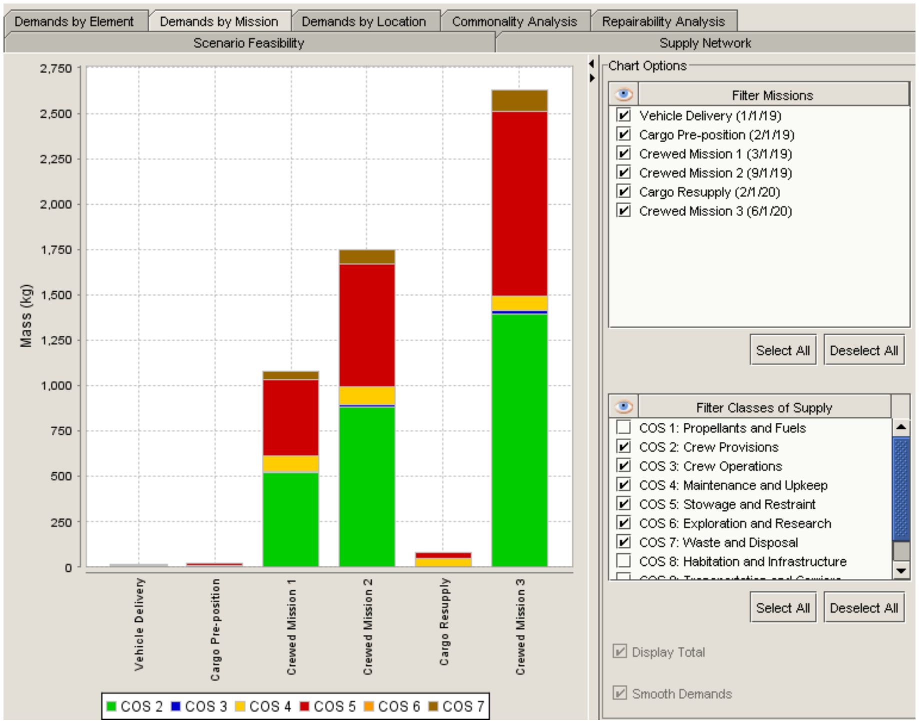


Figure 44: Demands for six example missions

Options:

- **Filter Missions:** Filter the display of missions. If one mission is selected, the demands are plotted over time, otherwise the demands are aggregated.
- **Filter Classes of Supply:** Filters the display of the 10 base classes of supply. Demands for more specific classes of supply (e.g. COS 401) are lumped into their parent class of supply.
- **Display Total:** Display a black line representing the total of all classes of supply. (Only enabled for single-mission analysis)
- **Smooth Demands:** As the simulation is the product of a discrete-event simulation, there are discrete jumps in demand quantities. This option smooths the demands line assuming linear demand generation. (Only enabled for single-mission analysis)

Demands by Location

The Demands by Location chart breaks the demands into the location of origin. The origin of demands is defined as either the location of the element that makes the demand, or the destination of a mission.

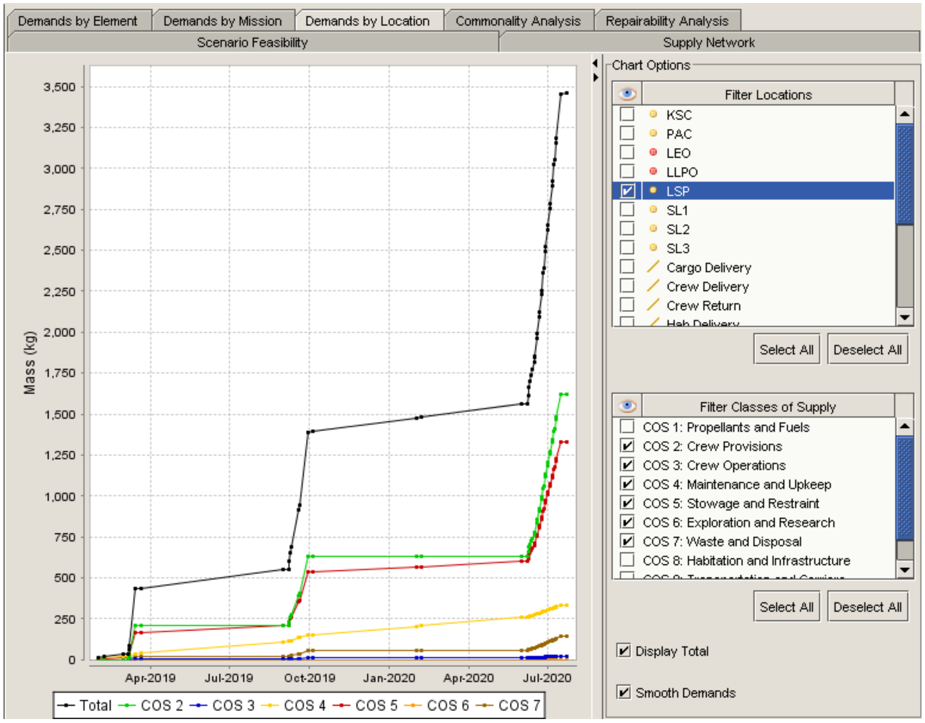


Figure 45: Demands at an example location

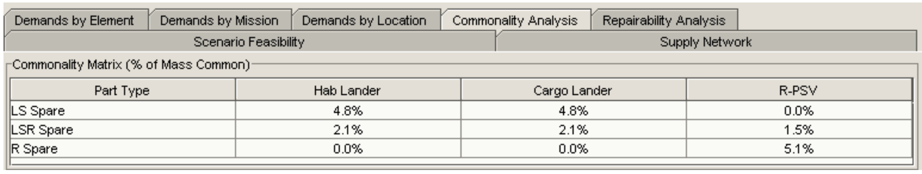
Options:

- **Filter Locations:** Filter the display of locations. If one location is selected, the demands are plotted over time, otherwise the demands are aggregated over the entire scenario.
- **Filter Classes of Supply:** Filters the display of the 10 base classes of supply. Demands for more specific classes of supply (e.g. COS 401) are lumped into their parent class of supply.
- **Display Total:** Display a black line representing the total of all classes of supply. (Only enabled for single-location analysis)
- **Smooth Demands:** As the simulation is the product of a discrete-event simulation, there are discrete jumps in demand quantities. This option smooths the demands line assuming linear demand generation. (Only enabled for single-location analysis)

Commonality Analysis

The Commonality Analysis tab is intended to provide basic analysis of the impacts of common spares between elements, specifically through scavenging spares. There are two main portions to this tab: the Commonality Matrix, and the Scavenging History chart.

The Commonality Matrix displays all elements that are assigned parts in columns, and each part type in rows. In each cell, the value displayed is the fraction of the element's mass that is comprised of the particular part. If a part has non-zero entries for more than one element, it is considered a common spare.



Part Type	Hab Lander	Cargo Lander	R-PSV
LS Spare	4.8%	4.8%	0.0%
LSR Spare	2.1%	2.1%	1.5%
R Spare	0.0%	0.0%	5.1%

Figure 46: Example commonality matrix for a few parts and elements

If scavenging is enabled in the demand options, the Scavenging History chart displays a history of which parts have been scavenged from where, and at what time.

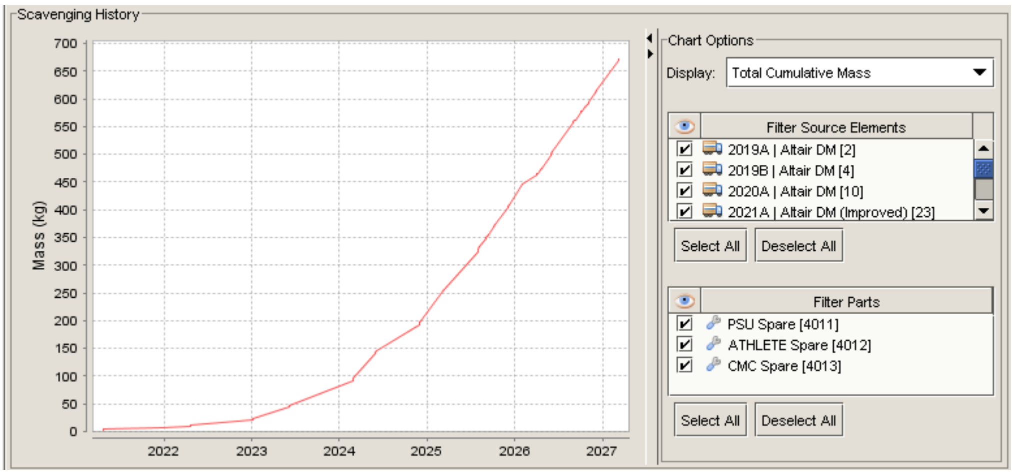


Figure 47: Displaying the cumulative scavenging over the scenario

Options:

- **Display:** Changes the form of data displayed
  - Total Cumulative Mass: Lumps all source elements and part types together and plots total scavenged mass over time.
  - Cumulative Units: Plots total units scavenged for each part type over time.
  - Mass by Source: Plots total mass of all spares scavenged from each source element over time.
- **Filter Source Elements:** Filters the display of the various source elements, which are elements that have been scavenged during the scenario.
- **Filter Parts:** Filters the display of the parts that have been scavenged during the scenario.

Repairability Analysis

The Repairability Analysis tab is intended to provide basic analysis of the repairable parts, and the crew time / mass savings trade that can be executed by repairing specific parts during the scenario. The goal of the repairability analysis is to assign a set of parts that are to be repaired rather than spared during crewed missions. Any spares that are demanded during cargo (unmanned) missions are pushed to the next crewed mission since spares are unlikely to be replaced without the presence of crew.

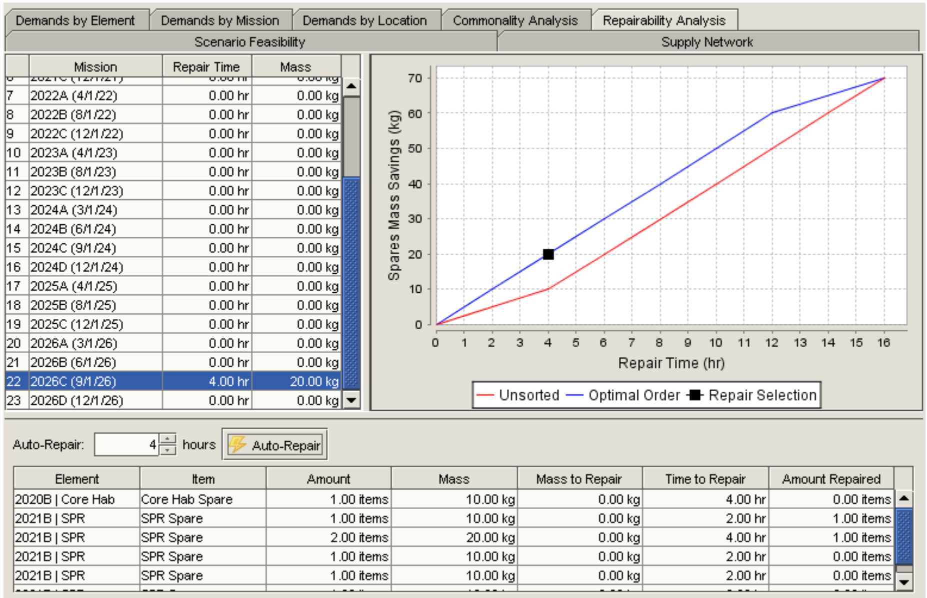


Figure 48: Repairability analysis for a crewed mission with discretized spares

For each crewed mission, a list of demanded spares is displayed at the bottom of the tab, indicating the element demanding the spare, the resource type, and amount. Also displayed is the mass of generic COS 4 and crew time required to perform the repair. The amount to be repaired can be modified in the last column. The chart above the table gives a visual representation of the repairable spares, both in the unsorted order of the table (red line), and sorted (blue line) to give the best mass-time trade. The black dot shows the current mass-time tradeoff based on the items that have been selected to be repaired.

A note on item discretization: if item discretization is turned off, demands are displayed as continuous amounts and repairs can fall anywhere on the continuum. If item discretization is on, however, demands are lumped to unit amounts, and repair of partial units is not allowed.

Finally, the auto-repair option will select the best combination of spares to repair given a set time to spend on repair activities. Note that if item discretization is turned on, there selected spares to repair tend to jump around as the auto-repair time passes discrete values.

MANIFESTING TAB

Once the scenario has been determined to be feasible from the mission perspective and from the estimated demands perspective, the Manifesting Tab is used to pack the demands into individual logistics containers, and manifest the containers onto carriers for transport. The packed, manifested demands are used as supplies for the scenario during simulation.

Demand Aggregation

The first step to manifesting is the aggregation of demands into lumped sums. Demands are aggregated on a per-location basis, backwards in time to each point where a manifesting decision must be made (whether to pack and manifest demands now, or earlier in time). These points are called *supply points*, and are the result of *supply edges*, which correspond to transport events in the scenario. If demands originate on the edge (during a transport), they are notated with an asterisk. These "edge demands" *must* be manifested on the specific transport where they originated to properly satisfy the demand.

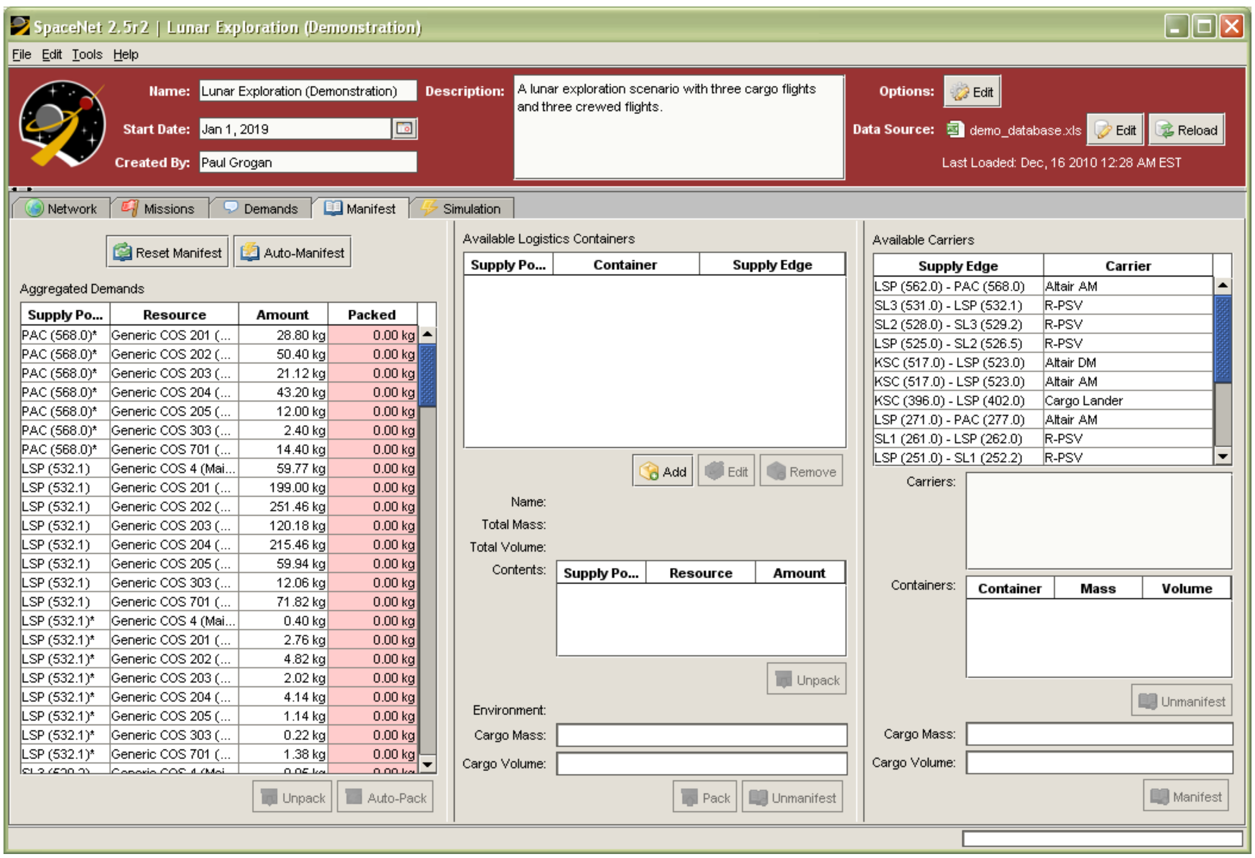


Figure 49: Aggregated demands list

Demand aggregation happens automatically when the "Reset Manifest" button is clicked, and the resulting demands can be viewed in the left-most table on the Manifesting Tab. The supply point at which the demand has been aggregated to, as well as the resource and amount demanded are also visible. The last column displays how much of the demand has been packed, when an entire aggregated demand line item has been packed, the last column turns green.

Demand Packing

Once demands have been aggregated, the next step is to pack them into logistics containers. Containers are specialized elements that can hold a set amount of resources. There are several hard-coded containers that are currently enabled for use in packing:

- **Crew Transfer Bag (CTB):** Container generally used to carry crew consumables, crew provisions, and waste equipment (COS 2, COS 3, and COS 7).
  - Mass: 1.81 kilograms
  - Volume: 0.0529 cubic meters
  - Maximum cargo mass: 45.36 kilograms
  - Maximum cargo volume: 0.0529 cubic meters
- **Double CTB:** Same as a CTB, but larger in size.
  - Mass: 3.62 kilograms
  - Volume: 0.1058 cubic meters
  - Maximum cargo mass: 90.72 kilograms
  - Maximum cargo volume: 0. 1058 cubic meters
  -
- **Half CTB:** Same as a CTB, but smaller in size.
  - Mass: 1.00 kilograms
  - Volume: 0.0248 cubic meters
  - Maximum cargo mass: 27.22 kilograms
  - Maximum cargo volume: 0.0248 cubic meters
- **Liquid Tank:** Container generally used to carry liquid resources (COS 201)
  - Mass: 34.37 kilograms
  - Volume: 0.0748 cubic meters
  - Maximum cargo mass: 74.8 kilograms
  - Maximum cargo volume: 0.0748 cubic meters
- **Liquid Tank Derivative:** Same as a liquid tank, but smaller in size.
  - Mass: 11.4567 kilograms
  - Volume: 0.0249 cubic meters
  - Maximum cargo mass: 24.9333 kilograms
  - Maximum cargo volume: 0.0249 cubic meters
- **Gas Tank:** Container generally used to carry gaseous resources (COS 203)
  - Mass: 108 kilograms
  - Volume: 2.75 cubic meters
  - Maximum cargo mass: 100 kilograms
  - Maximum cargo volume: 2.75 cubic meters
- **Gas Tank Derivative:** Same as a gas tank, but smaller in size.
  - Mass: 10.8 kilograms
  - Volume: 0.275 cubic meters
  - Maximum cargo mass: 10 kilograms
  - Maximum cargo volume: 0.275 cubic meters
- **SHOSS Box:** Container generally used to carry unpressurized spares (COS 4).
  - Mass: 120 kilograms

- Volume: 0.444 cubic meters
- Maximum cargo mass: 200 kilograms
- Maximum cargo volume: 0.444 cubic meters

- **Pressurized SHOSS Box:** Container generally used to carry pressurized spares (COS 4).
  - Mass: 0 kilograms
  - Volume: 0.8 cubic meters
  - Maximum cargo mass: 200 kilograms
  - Maximum cargo volume: 0.8 cubic meters

To manually pack a demand, add a new logistics container by clicking the "Add" button and choosing the desired container. Next, select the container, select the demand you wish to pack, and click the "Pack" button to add the demand (up to the capacity limits) to the container. The list of demands (and original aggregation supply point), and capacity constraints can be viewed by selecting a container.

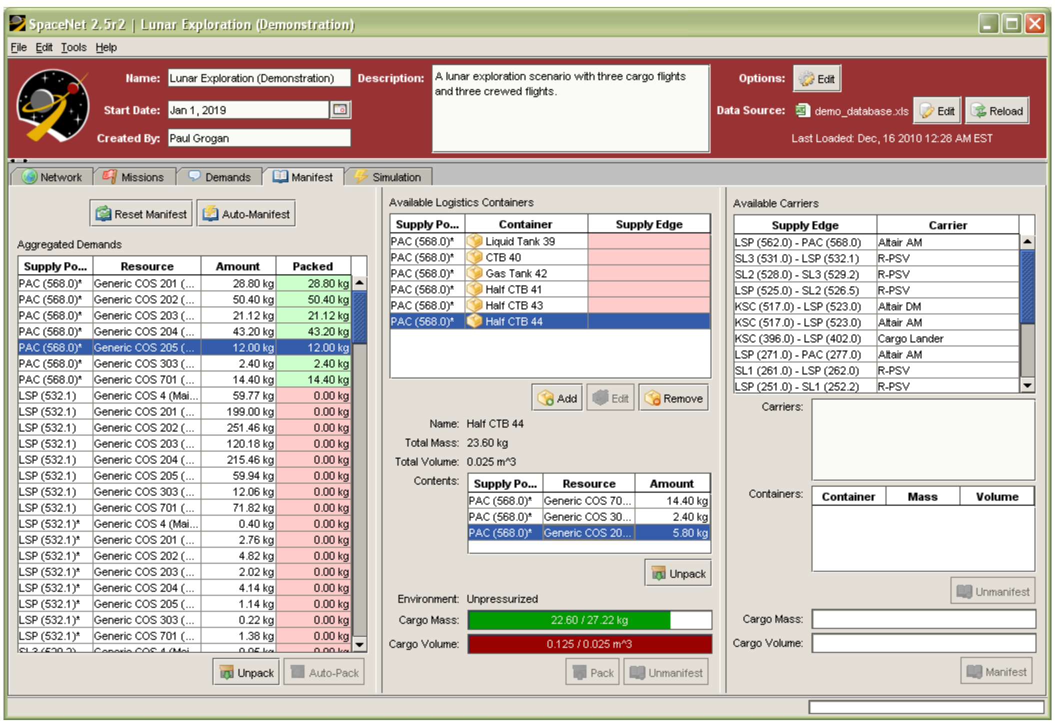


Figure 50: Aggregated demands partially packed into logistics containers

Manual packing is very tedious and can be easily automated with the Auto-Pack tool. If auto-packing is used (by clicking the "Auto-Pack" button when a demand is selected), the algorithm uses the following logic:

1. Determine proper container type bases on class of supply:
   a. COS 1 → Liquid Tank
   b. COS 203 → Gas Tank / Gas Tank Derivative
   c. COS 201 → Liquid Tank / Liquid Tank Derivative
   d. COS 2, COS 3, or COS 7 → CTB / Half CTB
   e. Pressurized COS 4 → Pressurized SHOSS Box
   f. Unpressurized COS 4 → SHOSS Box
   g. COS 6 → Customized Science Container (uses resource packing factor to set mass)

2. If not COS 6, look for existing containers of the correct type manifested to exist at the demand's location at or after the demand aggregation time. Attempt to pack demands in to remaining space.
3. Pack remaining demands into new containers of the correct type.

Demand Manifesting

Once containers are packed, they can then be manifested onto carriers for transport. On the right-hand side of the manifesting tab, the list of available carriers is visible. Each row corresponds to a top-level element in a transport event, and nested elements can be selected from the expanded view below. The list of manifested containers and current capacity constraints can be visible below the list of carriers.

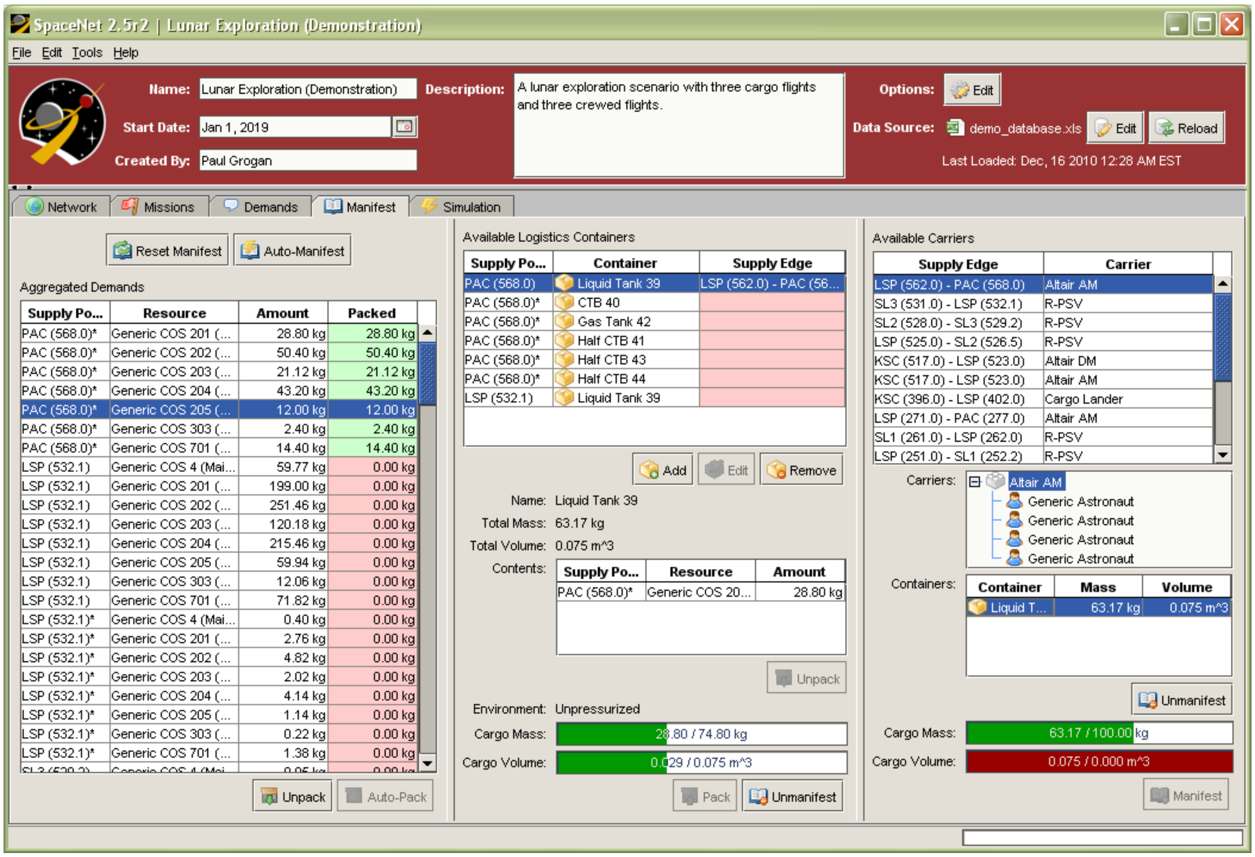


Figure 51: Packed containers partially manifested onto carriers during transports

To manually manifest a container, select the container, and choose a compatible supply edge, and finally choose a carrier with excess capacity. Compatible supply edges are those with a destination of the same node as the container's supply point, but arrive at the same or earlier time.

If a container is manifested onto a supply edge that *does not* originate from an Earth surface node, a secondary container is created to be manifested again (to reach an Earth surface node). In this fashion, containers can be transferred between several transports en route to the final destination.

Auto-Manifesting

Since demand packing and manifesting are very tedious for even short-duration scenarios, an automatic manifesting option has been included. It can be used at any time by clicking the "Auto-Manifest" button above the aggregated demands.

Although the auto-manifesting tool can be helpful to reduce the workload on the user, it will often not be optimal, and will often require user interaction to correct its logic. There are a few limitations to the auto-manifesting tool:

- While manifesting, elements don't take into account previously-manifested containers (e.g. from different legs of the trip). This can cause simulation errors as the auto-manifest overloads carriers which have existing containers.
- There is not a good way to move containers from one carrier to another, except immediately before a transport. This can cause the unintended effect of containers inadvertently being moved by carriers after delivery to a location (i.e. there is no way to off-load containers after delivery to a location).

The logic for the auto-manifesting tool puts emphasis on carry-along supplies:

For each supply point P, in reverse-chronological order:

1. For each demand aggregated to supply point P that originates from an edge:
    a. Auto-pack the demand
2. For each demand aggregated to supply point P that originates from a node:
    a. Auto-pack the demand
3. For each container C containing an edge demand from supply point P
    a. For each supply edge E supplying point P, in reverse-chronological order:
        i. For each top-level carrier R:
            1. If possible, manifest container C onto carrier R
            2. If manifested, go to next container
4. For each container C containing a demand from supply point P
    a. For each supply edge E supplying point P, in reverse-chronological order:
        i. For each top-level carrier R:
            1. If possible, manifest container C onto carrier R
            2. If manifested, go to next container

SIMULATION TAB

The final step of evaluating a scenario is simulation. The Simulation Tab contains the options to run a full simulation, taking into account manifested containers, repairing, and scavenging, and tracks the evolution of several Measures of Effectiveness (MOEs).

Simulation Overview

The simulation engine uses the following outline:

1. For each mission in the scenario, push each event onto the event stack
2. For each manifest action, push a manifest event onto the event stack (creates or moves the container to the proper carrier at the right time)
3. While the event stack has more events, get next event:
    a. Tabulate MOEs changed by this event
    b. If this event extends a finite duration from the previous event:
        i. For each element registered in the simulation:
            1. Generate demands
            2. Discretize demands (if enabled)
            3. Repair demands (if enabled)
            4. Attempt to satisfy demands from co-located elements
    c. Execute the event
    d. If the next event extends a finite duration from this event, save the network history

Simulation Options

- **Detailed Explorations:** Since crewed exploration processes generate many crewed EVA events, and each crewed EVA event corresponds to several other events (move in/out of habitat, reconfigure to EVA state), the internal process of logging the network history can run out of memory for long-duration scenarios. By unchecking detailed explorations, the individual EVA events are not actually generated, which greatly decreases the amount of memory used, but any EVA-related demands must be handled at the mission-level.
- **Detailed EVAs:** Similar to the detailed explorations option, deselecting this option will not generate the individual move/reconfigure events for the EVAs, cutting down on the amount of memory used, but again any EVA-related demands must be handled at the mission-level.
- **Simulation Errors:** If any errors occur during simulation, they will be displayed in the error log with a simple message. The simulation will not halt, though, so if more errors are generated, they may be attributed to one single error.

Simulation Visualizations

- **Network History:** The network history tab shows a visualization of the supply chain network and the flow of elements and supplies over time. Press the "play" button to play a simple animation. While the animation is running, the various elements and locations can be seen on the right-hand side, along with a representation of their contents.
- **Measures History:** The measures history tab shows the evolution of the Measures of Effectiveness (MOEs) over the course of the simulation.

- **Location History:** The location history tab shows the total mass of each of the ten base classes of supply at a specific location over the course of the simulation.
- **Element History:** The element history tab shows the total mass of each of the ten base classes of supply contained within an element over the simulation.

USE CASE SCENARIOS

During the development of SpaceNet 2.5, several use case scenarios were identified to target a wide range of analysis capabilities and target features.

Lunar Hub-Spoke (Central Outpost)

The Lunar Hub-Spoke scenario consists of 24 flights (fourteen crewed, ten cargo) over the span of seven years (June 2019 – December 2026), corresponding to the build-up and steady-state of an outpost located on the rim of Shackleton Crater. The scenario models infrastructure delivery and mission durations as detailed in Scenario 4.0.0 (developed by the Constellation Program Strategic Analysis Team). Several of the later missions include crewed surface transportation to nearby locations of scientific interest.

Some of the challenges of this scenario include:

- Faster scenario development than SpaceNet 1.3/1.4 to support twenty or more missions
- High level of reuse of elements from mission to mission
- Increasing dependence on ISRU technology

This scenario uses the Space Logistics Consumables Model (developed by JPL and JSC) to generate demands for the crew, EVAs, and habitat functions including the Environmental Control and Life Support System (ECLSS) at the outpost. Spares demands are handled with element-centric models that use percentages of the element's dry mass per year to estimate spare parts. The operational state of each element is reconfigured to account for the arrival and departure of crews.

The total cargo delivery capability over the campaign is approximately 146 metrics tons, of which 60 metric tons are reserved for infrastructure, 58 metric tons are used for crew and infrastructure demands, and 20 metric tons are dedicated to science payloads. The remaining margin, approximately 8 metric tons, is available for emergency reserves, or safety stocks. Additional infrastructure or science equipment could be manifested as well.

Three elements (Power & Support Unit – PSU, Crew Mobility Chassis – CMC, and Tri-ATHLETE) were analyzed for commonality with expended descent modules (DM) with respect to spare parts cannibalization. The percentage of an element's dry mass comprised of a particular common spare (as measured by % common by mass) was used to compare different levels of commonality.

Two additional elements (Core Habitat and Small Pressurized Rover – SPR) were evaluated for the tradeoff between mass savings and repair time during crewed missions. Existing data for maintenance tasks on the core habitat and the ascent module estimate an average repair time of 0.2 hours per kilogram of spare parts. Similar to commonality, the percentage of an element's dry mass compromised of a particular repairable spare (% repairable by mass) was used to measure different levels of variability.

Lunar Spoke-Hub (Mobile Outpost)

The objective of the Lunar Spoke-Hub scenario is to achieve global exploration via reusable surface mobility elements. The campaign consists of 16 flights (ten crewed, six cargo) beginning in July 2019 and ending in January 2025. This scenario begins with a few short duration sorties to the South Pole. A long-term traverse across the surface begins in 2020, making extensive use of the added capability to model surface transportation.

Surface transportation is enabled by the ATHLETE robot/rover, which can navigate both rugged and smooth terrain while carrying a habitat. This campaign is structured so that one crew meets another at a particular location. The two crews switch vehicles (habitat and ascent module) so that the first crew can return to Earth, while the next crew can continue on the surface transport. Periodic cargo flights to intermediate locations are used to re-supply the crew. Crew exchanges are chosen at scientifically interesting sites where focused exploration can take place. The flaw in this transit strategy is that the crew is not able to return from any location on the surface, thus adding risk to the scenario. This could be mitigated by towing an additional ascent module at all times at the expense of surface exploration.

International Space Station (ISS) Resupply (Post Shuttle Retirement)

The ISS Resupply scenario was designed to inspect a currently operational system rather than analyze a future scenario. This use case will model the International Space Station re-supply logistics in the post-shuttle retirement era of 2010-2015. NASA's baseline re-supply plan, consisting of utilization of cargo space through pre-negotiated agreements on International Partner (IP) flights, and US commercial cargo transportation systems (COTS) will be evaluated for feasibility.

A few challenges with this approach include:

- Non-zero initial state of system
- Complex models may be required for realistic resource and spares demands
- Analysis emphasis on finding optimal schedule using subset of known vehicle capabilities

Mars Exploration

The Mars scenario will model the Human Martian Exploration. Robotic precursors will be sent to Mars two launch window opportunity before the crew and scout three different locations on the surface and select the site for subsequent human missions. Mars Ascent Vehicle (MAV) and Earth Return Vehicle (ERV) will be sent to Mars one launch opportunity before the crew for pre-deployment. The MAV will be pre-deployed at the selected site. Then Transfer and Surface Habitat (TSH) and 5 crew members will be sent to the selected site on the surface. For the subsequent human missions, a pre-deployment of the MAV and ERV will be repeated one launch window opportunity before each crewed mission.

Some of the challenges with this scenario include:

- High variance of launch windows and flight duration (astrodynamics)
- Increasing consideration of alternative propulsion methods
- Increasing health concerns for long-term human space habitation
- Analysis emphasis on refining high-level campaign architecture

REFERENCES

1. Gralla E., Shull S., de Weck O., "A Modeling Framework for Interplanetary Supply Chains", AIAA-2006-7229, *AIAA Space 2006 Conference and Exposition*, San Jose, California, September 19-21, 2006.

2. Shull S., Gralla E., Siddiqi A., de Weck O., Shishko R., "The Future of Asset Management for Human Space Exploration: Supply Classification and an Integrated Database", AIAA-2006-7232, *AIAA Space 2006 Conference and Exposition*, San Jose, California, September 19-21, 2006.

3. Lee G., Benzin K., de Weck O., Armar N., Siddiqi A., Jordan E., Whiting J., "Integrated Cx Mission Modeling," *ATA-01-1006,* 2007.

4. Shull, S. "Integrated Modeling and Simulation of Lunar Exploration Campaign Logistics", M.S. Thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, Massachusetts. 2007.

5. Shull, S., Gralla E., Silver M., Li X., de Weck O., "Modeling and Simulation of Lunar Campaign Logistics", AIAA-2007-6244, *AIAA Space 2007 Conference and Exposition*, Long Beach, California, September 18-20, 2007.

6. de Weck O.L., Simchi-Levi D., Shishko R., Ahn J., Gralla E., Klabjan D., Mellein J., Shull A., Siddiqi A., Bairstow B, Lee G., "SpaceNet v1.3 User's Guide", NASA/TP-2007-214725, January 2007.

7. Siddiqi A., de Weck O., "Spare Parts Requirements for Space Missions with Reconfigurability and Commonality", Journal of Spacecraft and Rockets, 44 (1), 147-155, January-February 2007.

8. Siddiqi A., de Weck O., Lee., G, "Matrix Modeling Methods for Space Exploration Campaign Logistics Analysis", AIAA-2008-7749, *AIAA Space 2008 Conference and Exposition*, San Diego, California, September 9-11, 2008.

9. Siddiqi A., Shull S., de Weck O., "Matrix Methods Analysis of International Space Station Logistics", AIAA-2008-7605, *AIAA Space 2008 Conference and Exposition*, San Diego, California, September 9-11, 2008.

10. Lee G., de Weck O., Armar N., Jordan E., Shishko R., Siddiqi A., and Whiting J., "SpaceNet: Modeling and Simulating Space Logistics", AIAA-2008-7747, *AIAA Space 2008 Conference and Exposition*, San Diego, California, September 9-11, 2008.

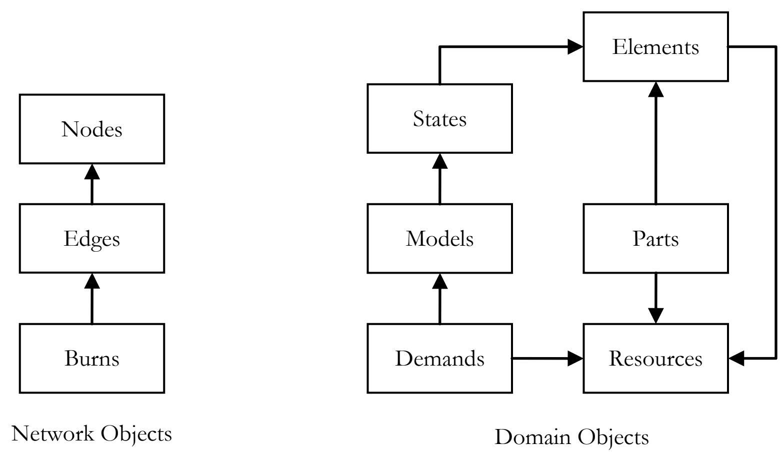
11. Armar, N. "Cargo Revenue Management for Space Logistics", AIAA-2009-6723, *AIAA Space 2009 Conference and Exposition*, Pasadena, California, September 14-17, 2009.

12. Grogan P., Armar N., Siddiqi A., de Weck O., Lee G., Jordan E., Shishko R., "A Flexible Architecture and Object-oriented Model for Space Logistics Simulation", AIAA-2009-6548, *AIAA Space 2009 Conference and Exposition*, Pasadena, California, September 14-17, 2009.

APPENDIX A: DATA STRUCTURES GUIDE

The data used to construct SpaceNet scenarios can be described from two perspectives: the database and data structures. In essence, the database acts as a user interface for the data structures, which are used in the core programming. A solid understanding of the data structures used in SpaceNet will greatly aid your further understanding of the database.

The SpaceNet data structures consist of nine classes of objects, forming two groups of data (network and domain). The network group includes nodes, edges, and burns, and describes the spatial locations and transports in the scenario. The three objects are related in that each edge has an origin node and a destination node, while each burn has an associated edge.

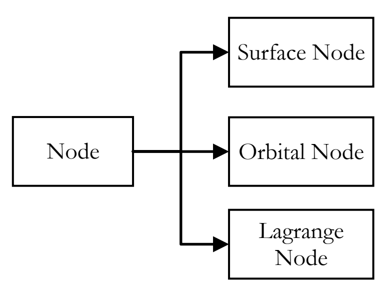


The domain group includes elements, parts, states, models, demands, and resources, and describes the many properties of the various elements that can be used in the scenario. The six objects are related in that elements can contain zero or more parts and states, states can contain zero or more models, models can contain zero or more demands, and each demand is associated with a resource. Also, elements can be associated with resources for fuel tanks, and parts each correspond to one resource.

Node

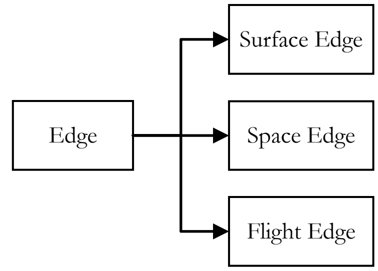
There are three types of nodes:

- **Surface nodes** represent static locations on a planetary body, given by the latitude and longitude.
- **Orbital nodes** represent time-invariant stable orbits about a planetary body, given by an apoapsis, periapsis, and orbital inclination.
- **Lagrange nodes** represent the five points of zero net acceleration in a three-body system.



Edges

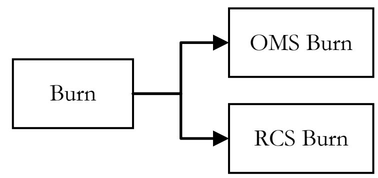
There are three types of edges:



- **Surface edges** represent transitions between two surface nodes.
- **Space edges** represent transitions between two nodes using a specified list of propulsive burns.
- **Flight edges** represent transitions between two nodes using flight architectures that are known to close with a given cargo and crew capacity.

Burns

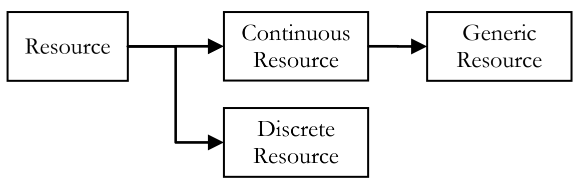
There are two types of burns:



- **OMS burns** (Orbital Maneuvering System)
- **RCS burns** (Reaction Control System)

Each can only be satisfied by propulsive vehicles with the correct capabilities. Each burn is measured by the change in velocity (delta-v) that is required of the elements.

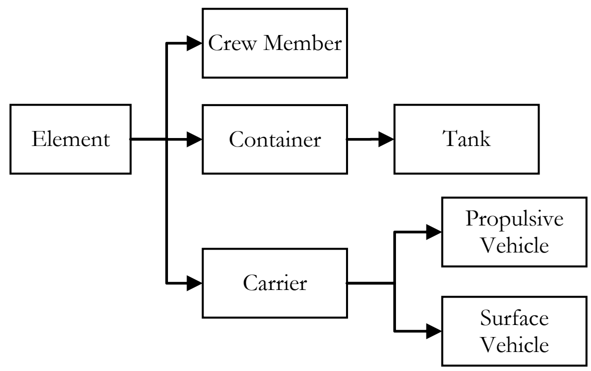
Resources



There are three types of resources:

- **Continuous resources** represent resources that are continuous in nature, such as propellants, liquids, and gases. Continuous resources have a customizable unit of measurement as well as a unit mass, unit density, and environment (pressurized or unpressurized).
- **Generic resources** represent instances of a specific class of supply (COS) as a continuous resource. References to generic resource type ID's (TID) are made by the opposite of the class of supply number, for example, generic COS 4 is TID -4, generic COS 203 is TID -203. Generic resources always have units of kilograms.
- **Discrete resources** represent resources that are discrete in nature, such as individual supply items and spare parts. Discrete resources are always measured in units, but allow customized unit masses and unit volumes.

Elements



There are many types of elements, arranged in a hierarchy to leverage on the inheritance of common properties.

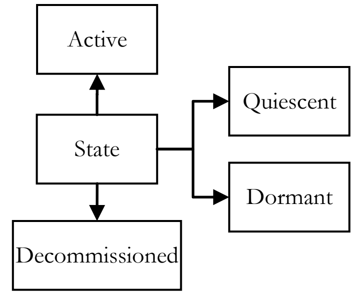
- **Elements** represent the basic building block of scenarios and have a specified mass, volume, and environment (either pressurized or unpressurized). Elements may have a set of parts used in scenario repair activities and may have a set of states used during scenario demand generation. When elements are instantiated in a "Create Elements" event, they are assigned a unique identifier (UID).
- **Crew Members** are specialized elements that represent humans in scenarios that have an active time fraction that dictates the portion of each day that can be used for exploration and maintenance.
- **Resource Containers** are specialized elements that can contain resources up to a mass and volume capacity. Containers also have a cargo environment (either pressurized or unpressurized) that dictate what type of resources can be carried (pressurized resources can only be carried in a pressurized cargo environments, unpressurized resources can be carried in either).
- **Resource Tanks** are a specialized type of resource container that can only contain one type of resource at a time. Due to this simplification, their capacity constraint can be described by a maximum amount instead of a maximum mass and volume.
- **Carriers** are specialized elements that can carry other elements up to a mass and volume capacity. Carriers also have a cargo environment that works in the same way as the container's cargo environment.
- **Propulsive Vehicles** are a specialized type of Carrier that may perform OMS and/or RCS propulsive burns. Propulsive vehicles may have separate or shared OMS and RCS fuel tanks of a specific resource type (fuel).
- **Surface Vehicles** are a specialized type of Carrier that may perform surface transports. Surface vehicles have maximum speeds and a fuel tank to store a specific resource type.

Part Applications

Part applications match resources (specifically discrete resources) with elements, representing parts as applied in a particular element. Part applications provide failure and repair data such as mean time to failure (MTTF), mean repair time (MRT), and mass to repair, which is the amount of generic resources to perform a repair.

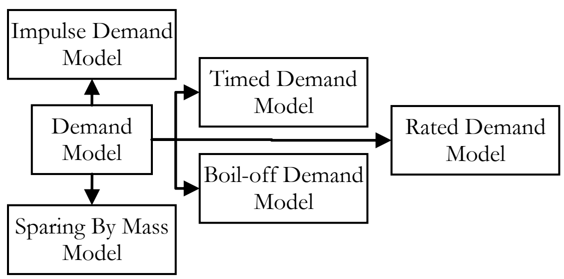
States

States match demand models with elements, with each state representing an operational configuration of the element. There are four types of states:



- **Active states** represent the normal operation of an element.
- **Special states** represent special operational states of elements such as crew members on EVA.
- **Quiescent states** represent less-than-normal operation of an element.
- **Dormant states** represent minimal operation of an element, typically used to represent elements before activation (e.g. during transit).
- **Decommissioned states** represent an element that is not operational, and identifies that the parts can be scavenged by other elements. Elements may not transition from a decommissioned state.

Demand Models



Demand models capture resource consumption or production by elements in a particular state during the scenario. There are many types of demand models including:

- **(Timed) Impulse demand models** generate a one-time demand for a set of resources.
- **Rated demand models** generate demands for a set of resources at a constant rate (linear in time).
- **Sparing by mass models** generate a specified percentage of an element's mass each year in generic COS 4 pressurized and unpressurized spares.

Demands

Demands simply represent combinations of resources and amounts. Demands are used in conjunction with the impulse, timed impulse, and rated demand models and also for initial states of resource containers.

APPENDIX B: DATABASE GUIDE

Nodes Table

Field	Description	Values	Required For*		
			Sur	Orb	Lag
id	Unique identifier	Unique Integer	X	X	X
type	Type of node	{surface, orbital, lagrange}	X	X	X
name	Name of node, typically abbreviated to 3-4 letters.	String	X	X	X
body_1	Body of surface location, body of orbit, or major body of Lagrange point	{sun, earth, moon, mars}	X	X	X
latitude	Latitude of surface location	Double [-90.0, 90.0] (degrees)	X		
longitude	Longitude of surface location	Double [-180.0, 180.0] (degrees)	X		
apoapsis	Major radius of orbit	Double [0, inf] (kilometers)		X	
periapsis	Minor radius of orbit	Double [0, inf] (kilometers)		X	
inclination	Inclination of orbit	Double [0, inf] (degrees)		X	
body_2	Minor body of Lagrange node	{sun, earth, moon, mars}			X
lp_number	Number of Lagrange point	{1,2,3,4,5}			X
description	Description of node	String			

*Abbreviations: Sur – Surface Node, Orb – Orbital Node, Lag – Lagrange Node

Edges Table

Field	Description	Values	Required For*		
			Sur	Spa	Flt
id	Unique identifier	Unique Integer	X	X	X
type	Type of edge	{surface, space, flight}	X	X	X
name	Name of edge	String	X	X	X
origin_id	Origin node ID	Node ID	X	X	X
destination_id	Destination node ID	Node ID	X	X	X
duration	Duration of space edge or flight	Double [0, inf] (days)		X	X
distance	Distance of surface edge	Double [0, inf] (kilometers)	X		
max_crew	Crew capacity for flight	Double [0, inf]			X
max_cargo	Cargo capacity for flight	Double [0, inf] (kilograms)			X
description	Description of edge	String			

*Abbreviations Sur – Surface Edge, Spa – Space Edge, Flt – Flight Edge

Burns Table

Field	Description	Values	Req'd For*	
			OMS	RCS
id	Unique identifier	Unique Integer	X	X
edge_id	Associated space edge	Space Edge ID	X	X
time	Time of burn (after start)	Double [0, inf] (days)	X	X
order	Order of the burn in sequence	Integer	X	X
type	Type of burn	{oms, rcs}	X	X
delta_v	Required change in velocity	Double [0, inf] (meters/second)	X	X

* Abbreviations: OMS – orbital maneuvering system, RCS – reaction control system

Resource Types Table

Field	Description	Values	Req'd For*	
			Cont	Disc
id	Unique identifier	Unique Integer	X	X
type	Type of resource	{continuous, discrete}	X	X
name	Name of resource	String	X	X
cos	Class of supply	COS ID	X	X
units	Units of measurement	String	X	X
unit_mass	Unit mass	Double [0, inf] (kilograms)	X	X
unit_volume	Unit volume	Double [0, inf] (cubic meters)	X	X
packing_factor	Amount of additional COS 5 required to pack a unit of this resource in a container	Double [0, inf] (kilograms)	X	X
environment	Resource environment – pressurized resources cannot be placed in unpressurized containers	{pressurized, unpressurized}	X	X
description	Description of the resource	String		

*Abbreviations: Cont – continuous, Disc – discrete

Elements Table

Field	Description	Values	Req'd For*						
			E	CM	RC	RT	CA	PV	SV
id	Unique identifier	Unique Integer	X	X	X	X	X	X	X
type	Type of element	{element, crew member, container, tank, carrier, propulsive, surface}	X	X	X	X	X	X	X
name	Name of element	String	X	X	X	X	X	X	X
cos	Class of supply	COS ID	X	X	X	X	X	X	X
environment	Element environment – pressurized elements cannot be placed in unpressurized carriers	{pressurized, unpressurized}	X	X	X	X	X	X	X
accommodation_mass	Amount of additional COS 5 required to pack element in a carrier	Double [0, inf] (kilograms)	X	X	X	X	X	X	X
mass	Mass of element	Double [0, inf] (kilograms)	X	X	X	X	X	X	X
volume	Volume of element	Double [0, inf] (cubic meters)	X	X	X	X	X	X	X
max_crew	Crew capacity limit	Integer					X	X	X
cargo_mass	Cargo mass capacity	Double [0, inf] (kilograms)			X	X	X	X	X
cargo_volume	Cargo volume capacity	Double [0, inf] (cubic meters)			X		X	X	X
cargo_environment	Cargo environment constraint	{pressurized, unpressurized}			X	X	X	X	X
active_fraction	Fraction of the day that a crew member is available for exploration or maintenance	Double [0, 1]		X					
oms_isp	OMS specific impulse	Double [0, inf] (seconds)						X	
max_oms	Maximum OMS fuel amount	Double [0, inf] (units)						X	
oms_id	OMS fuel resource ID	Resource ID							
rcs_isp	RCS specific impulse	Double [0, inf] (seconds)						X	
max_rcs	Maximum RCS fuel amount	Double [0, inf] (units)						X	
rcs_id	RCS fuel resource ID	Resource ID							
max_speed	Maximum speed	Double [0, inf] (km/hr)							X
max_fuel	Maximum fuel amount	Double [0, inf] (units)							X
fuel_id	Fuel resource ID	Resource ID							X
description	Description of the resource	String							

*Abbreviations: E – element, CM – crew member, RC – resource container, RT – resource tank, CA – carrier, PV – propulsive vehicle, SV – surface vehicle

Part Applications Table

Field	Description	Values	Required
id	Unique identifier	Unique Integer	X
part_id	Discrete resource ID	Resource ID	X
element_id	Element ID	Element ID	X
qpa	Quantity per application	Integer	X
duty_cycle	Fraction of the time the part is in use	Double [0, 1]	X
time_to_failure	Mean time to failure of this part	Double [0, inf] (hours)	X
mean_repair_time	Mean time to repair this part	Double [0, inf] (hours)	X
mass_to_repair	Amount of additional COS 4 required to repair this part	Double [0, inf] (kilograms)	X

States Table

Field	Description	Values	Required For*			
			A	Q	DM	DC
id	Unique identifier	Unique Integer	X	X	X	X
element_id	Element ID	Element ID	X	X	X	X
name	Name of state	String	X	X	X	X
state_type	Type of state	{active, quiescent, dormant, decommissioned}	X	X	X	X
initial_state	Boolean value for the initial state	{TRUE,FALSE}	X	X	X	X

* Abbreviations: A – active, Q – quiescent, DM – dormant, DC - decommissioned

Models Table

Field	Description	Values	Required For			
			R	I	T	SBM
id	Unique identifier	Unique Integer	X	X	X	X
type	Type of demand model	{rated, timed, sparing by mass}	X	X	X	X
state_id	State ID	State ID				
name	Name of demand model	String	X	X	X	X
parts_list	Whether to use the part application list to drive sparing-by-mass demands	{TRUE,FALSE}				X
unpress_rate	Mass of unpressurized spares (as a percentage of element's dry mass) per year.	Double [0, inf]				X
press_rate	Mass of pressurized spares (as a percentage of element's dry mass) per year.	Double [0, inf]				X

* Abbreviations: R – rated, T – timed impulse, SBM – sparing by mass

Demands Table

Field	Description	Values	Required
id	Unique identifier	Unique Integer	X
model_id	ID of the model to set demand parameters	Rated, Impulse, or Timed Model ID	
container_id	ID of the container to set initial contents	Container ID	
resource_id	ID of the resource (opposite of COS ID if using generic)	Resource ID	X
amount	Boolean value if this is the element's initial state	Double [0, inf] (units)	X