Space Logistics Exploration Campaign Scenario Specification for SpaceNet

AIAA SciTech Forum

January 10, 2024

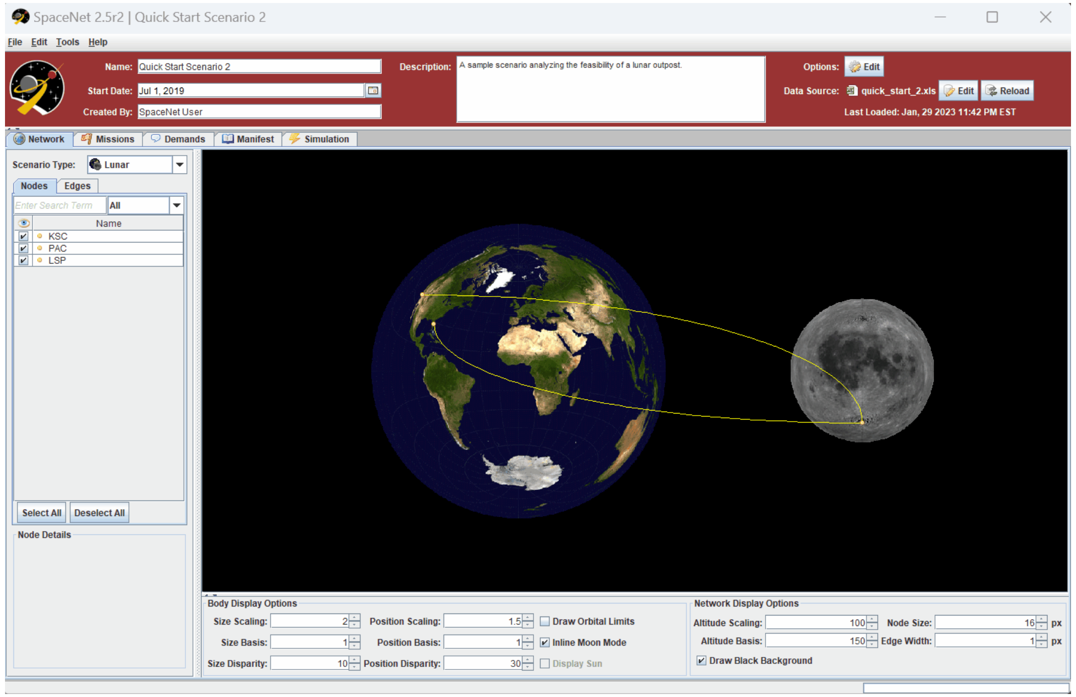
Paul T. Grogan, Sarah Bentley, George Lordos,
Kir Latyshev, Ireland Brown, and Olivier L. de Weck

Overview

- SpaceNet Overview
- Artemis III Quick Start Scenario
- Analysis Capabilities
- Software Interfaces

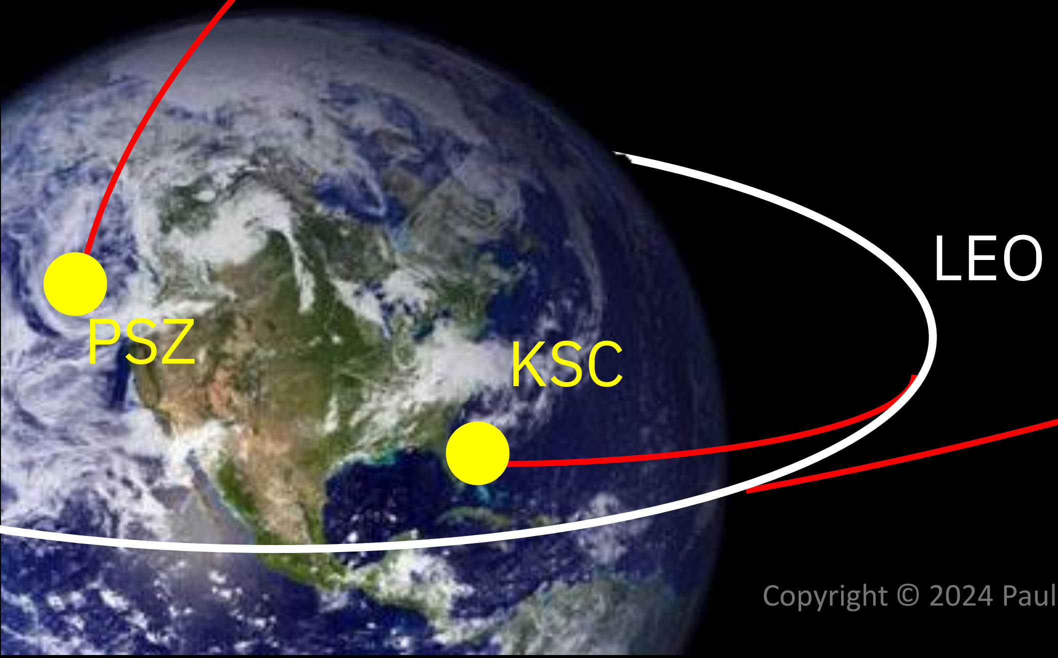
SpaceNet Campaign Logistics Tool

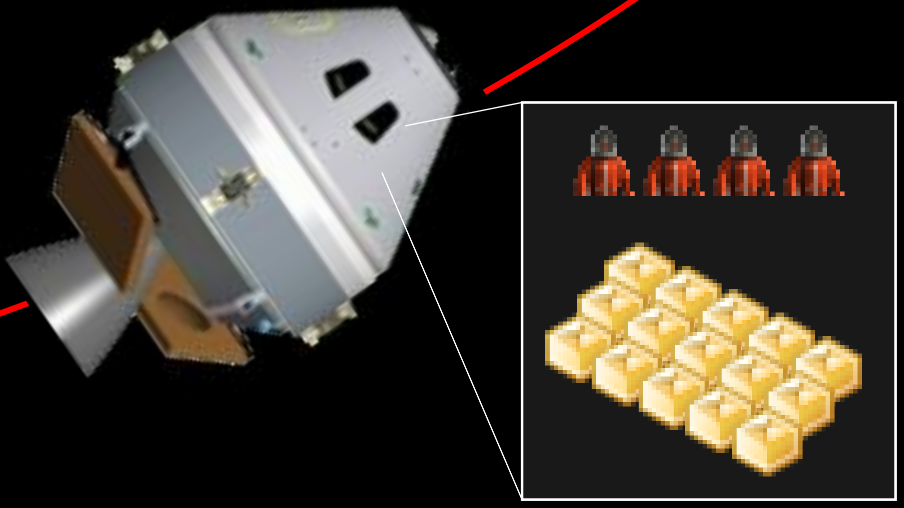
- Originally developed by MIT’s Space Logistics Project under NASA’s Constellation Program
  - 2004-2010 (Java); 2020-Present (Python)
- Model & simulate campaign logistics:
  - Compose integrated scenario
  - Evaluate propulsive feasibility
  - Evaluate logistical feasibility
  - Measure key performance indicators
- Open-source repositories:
  - SpaceNet Java: github.com/space-logistics-org/spacenet-java
  - SpaceNet “Cloud”: github.com/space-logistics-org/spacenet



SpaceNet Simulation State

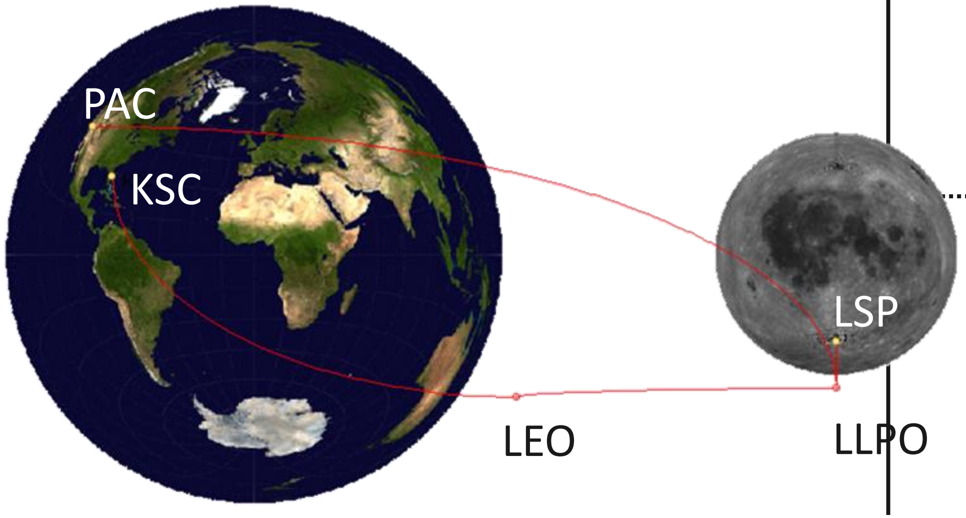
- Exploration Network
  - Nodes: Orbit, Surface, Lagrange
  - Edges: Space, Surface, Flight
- Elements
  - Crew Members
  - Element Carriers
    - Propulsive Vehicle
    - Surface Vehicle
  - Resource Containers



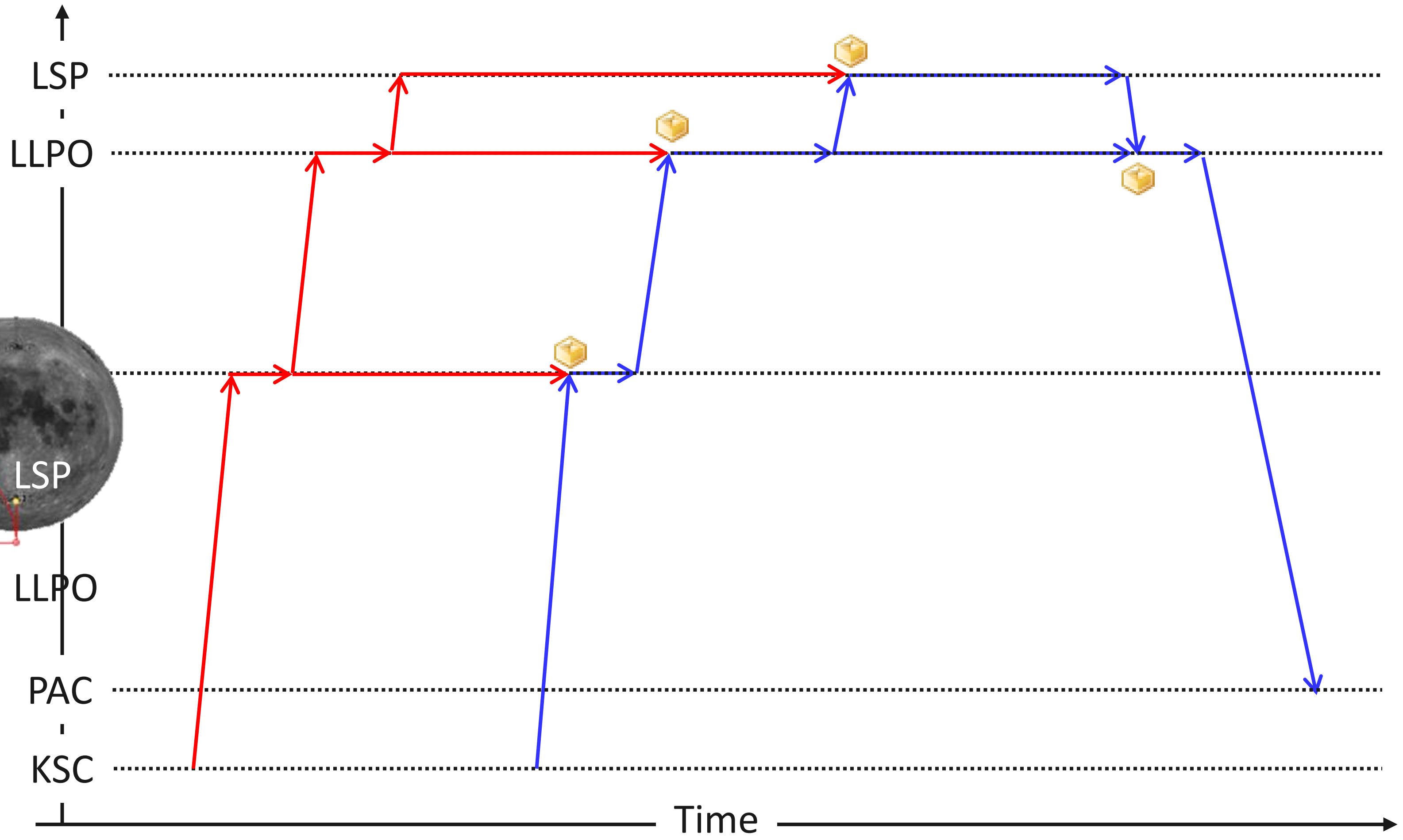


Interplanetary Supply Chain Network

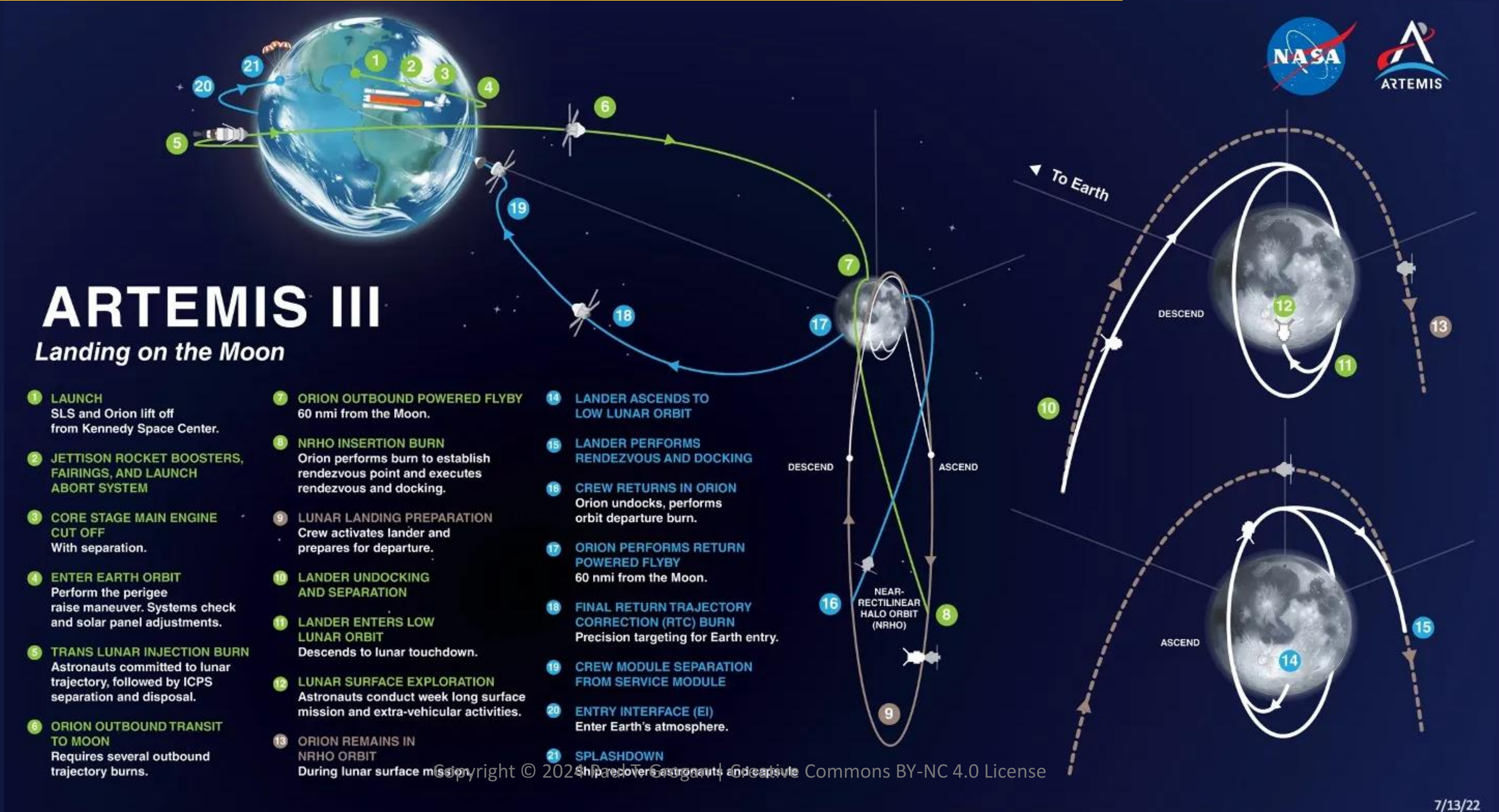
Static Network



Time-expanded Network



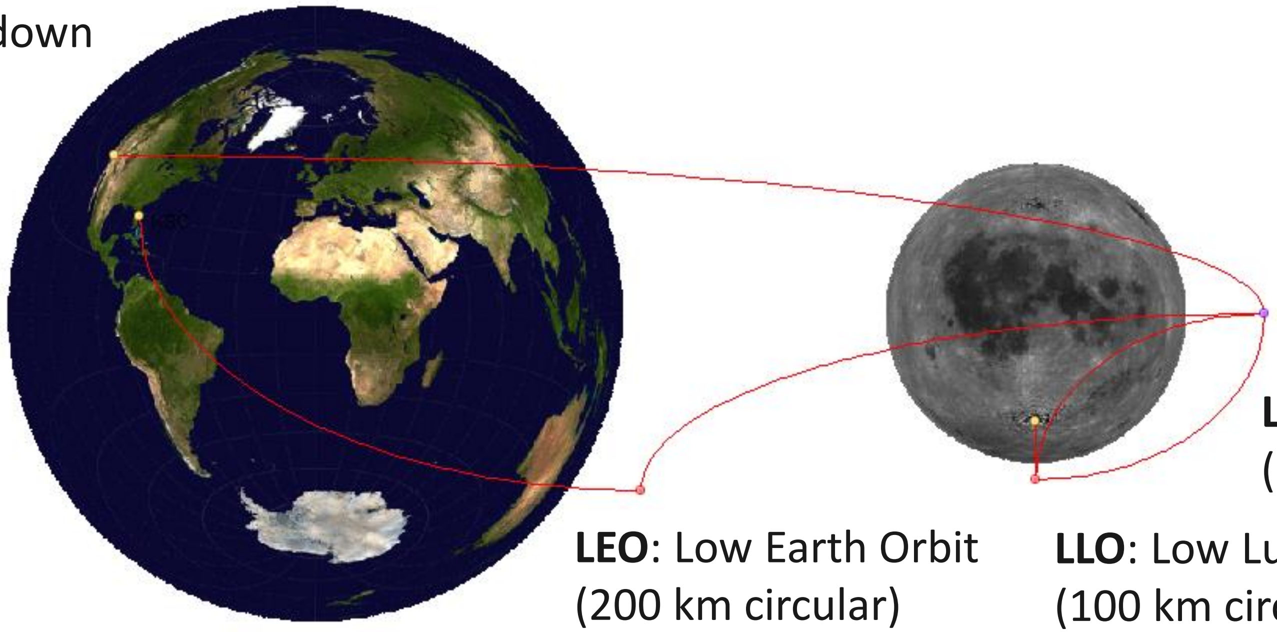
Quick Start Scenario 3: Artemis III



Artemis III Exploration Network

PAC: Pacific Ocean Splashdown
(35.0° N, 117.9° W)

KSC: Kennedy Space Center
(28.6° N, 80.6° W)



NRO: Near Rectilinear Halo Orbit (Earth-Moon L2)

LSP: Lunar South Pole
(89° S, 180° W)

LEO: Low Earth Orbit
(200 km circular)

LLO: Low Lunar Orbit
(100 km circular)

KSC-LEO: Earth Ascent

1. 0.0: 9500 m/s

LEO-NRO: Trans-Lunar Injection

1. 0.0: 3124 m/s
2. 0.3: 30 m/s
3. 3.1: 242 m/s
4. 4.4: 1 m/s
5. 6.4: 126 m/s

NRO-LLO: Low Lunar Orbit Transfer

1. 0.0: 20 m/s
2. 0.5: 648.6 m/s

LLO-LSP: Lunar Descent

1. 0.0: 19.4 m/s
2. 0.2: 1692.5 m/s

LSP-LLO: Lunar Ascent

1. 0.0: 1692.4 m/s
2. 0.2: 19.4 m/s

LLO-NRO: Halo Orbit Transfer

1. 0.0: 649.2 m/s
2. 0.5: 125.1 m/s

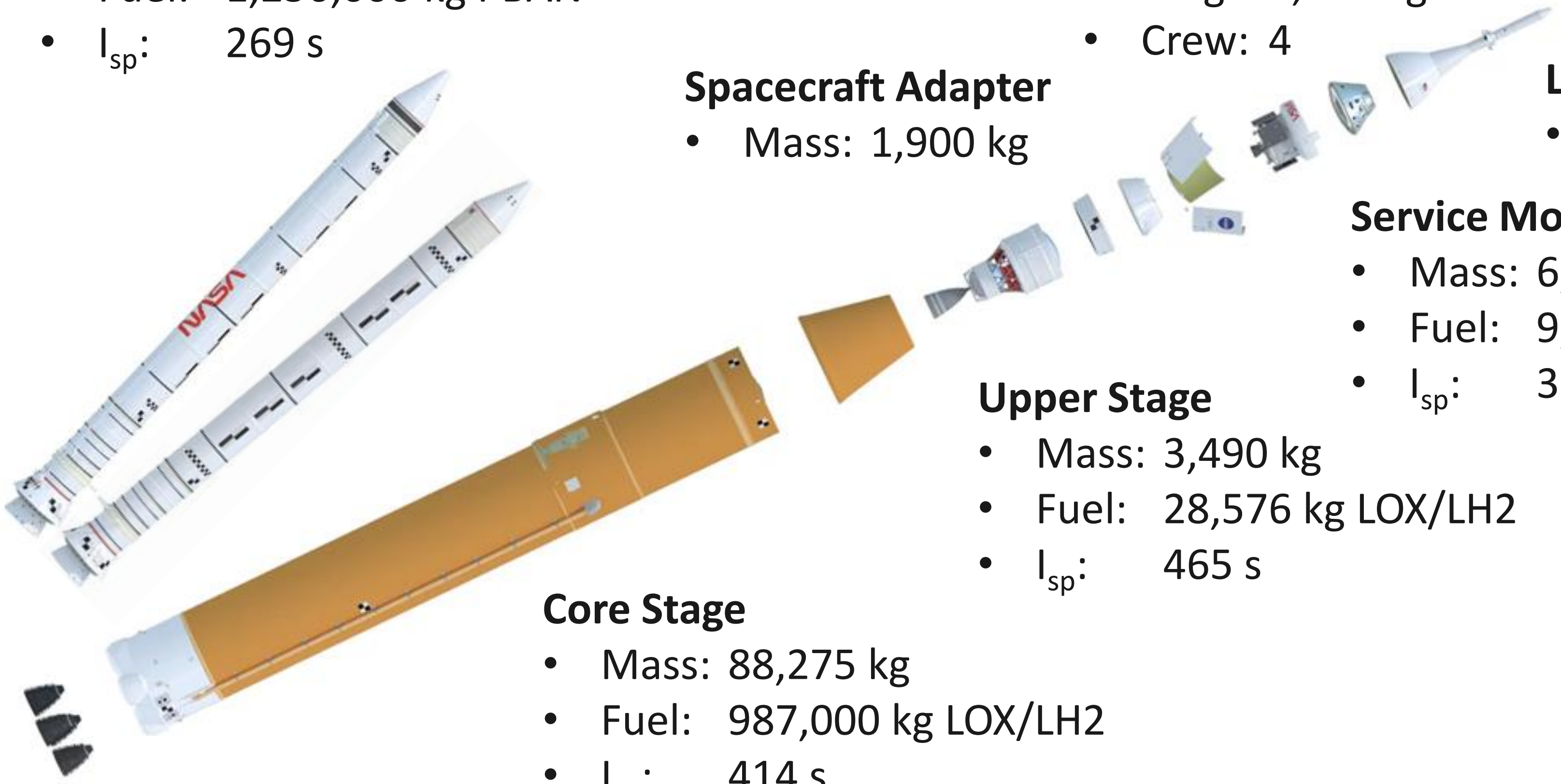
NRO-PAC: Earth Return

1. 0.0: 147 m/s
2. 1.6: 2 m/s
3. 2.8: 263 m/s

Artemis III Element Templates

Crew Member
- Mass: 100 kg

Lunar Samples
- Mass: 100 kg



Solid Rocket Boosters (2)
- Mass: 195,000 kg
- Fuel: 1,256,000 kg PBAN
- I_{sp}: 269 s

Crew Module
- Mass: 9,300 kg
- Cargo: 1,100 kg
- Crew: 4

Spacecraft Adapter
- Mass: 1,900 kg

Launch Abort System
- Mass: 7,250 kg

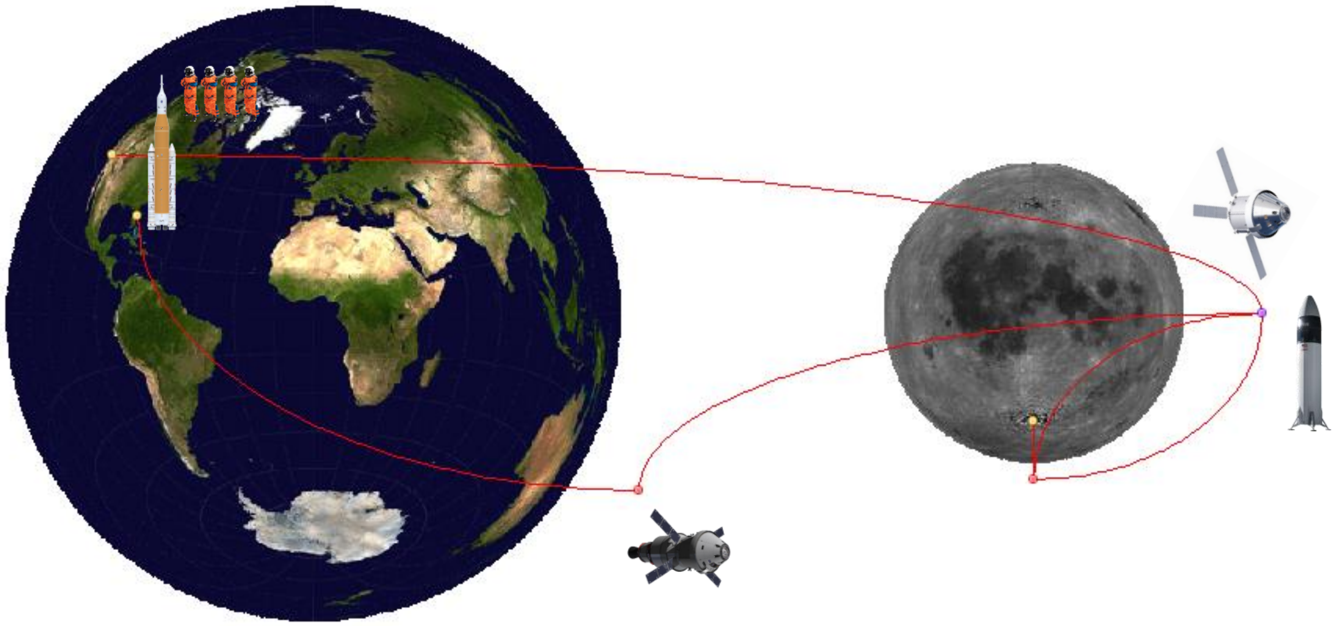
Service Module
- Mass: 6,185 kg
- Fuel: 9,276 kg N2O4/MMH
- I_{sp}: 316 s

Upper Stage
- Mass: 3,490 kg
- Fuel: 28,576 kg LOX/LH2
- I_{sp}: 465 s

Core Stage
- Mass: 88,275 kg
- Fuel: 987,000 kg LOX/LH2
- I_{sp}: 414 s

Human Landing System
- Mass: 8,149 kg
- Cargo: 900 kg
- Crew: 2
- Fuel: 32,285 kg LOX/LCH4
- I_{sp}: 363 s

Artemis III Mission Events



0.0 **Create HLS.1 at NRO**

0.0 **Create SRB.1, CS.1, US.1, SA.1, SM.1, CM.1, LAS.1 at KSC**

0.0 **Create C.1, C.2, C.3, C.4 in CM.1**

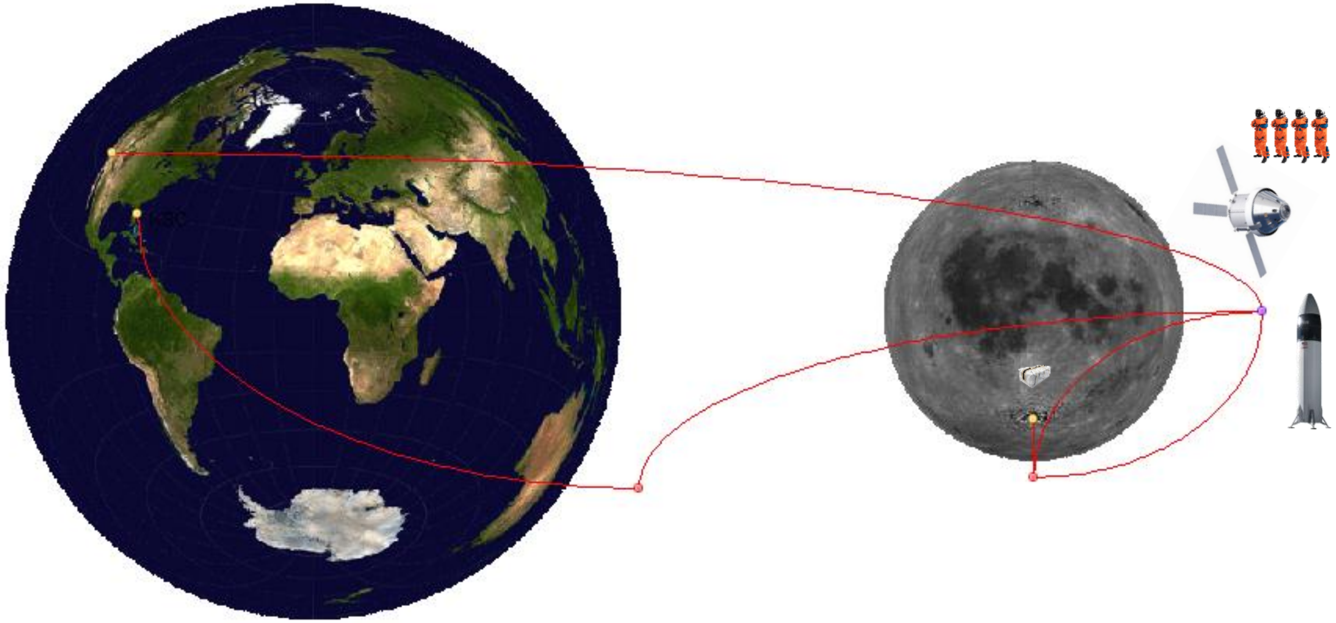
0.0 **Transport SRB.1, CS.1, US.1, SA.1, SM.1, CM.1, LAS.1 on KSC-LEO**

(1) Burn SRB.1, Stage SRB.1, Stage LAS.1, Burn CS.1, Stage CS.1, Stage SA.1

1.0 **Transport US.1, CM.1, SM.1 on LEO-NRO**

(1) Burn US.1, Stage US.1
(2) Burn SM.1
(3) Burn SM.1
(4) Burn SM.1
(5) Burn SM.1

Artemis III Mission Events



8.0 **Move C.1, C.2 to HLS.1**

9.0 **Transport HLS.1 on NRO-LLO**
- (1) Burn HLS.1
- (2) Burn HLS.1

10.0 **Transport HLS.1 on LLO-LSP**
- (1) Burn HLS.1
- (2) Burn HLS.1

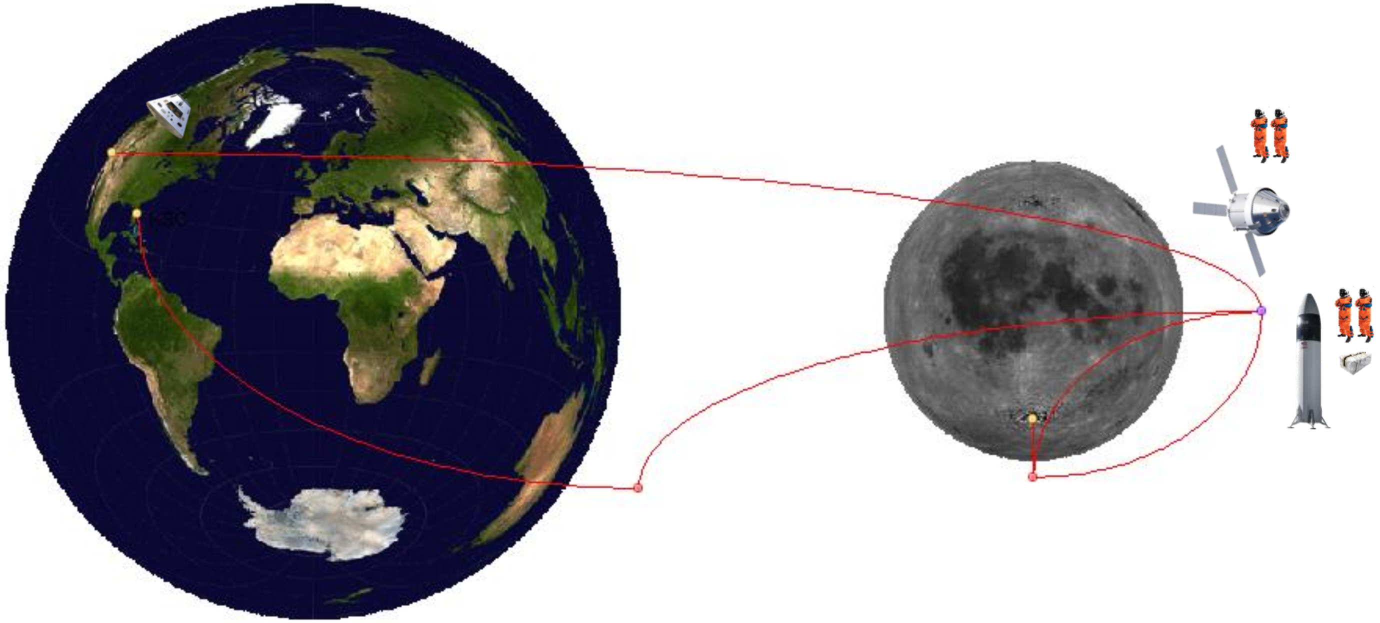
11.0 **Surface Exploration**

18.0 **Create S.1 in HLS.1**

18.0 **Transport HLS.1 on LSP-LLO**
- (1) Burn HLS.1
- (2) Burn HLS.1

19.0 **Transport HLS.1 on LLO-NRO**
- (1) Burn HLS.1
- (2) Burn HLS.1

Artemis III Mission Events



20.0 **Move C.1, C.2, S.1 to CM.1**

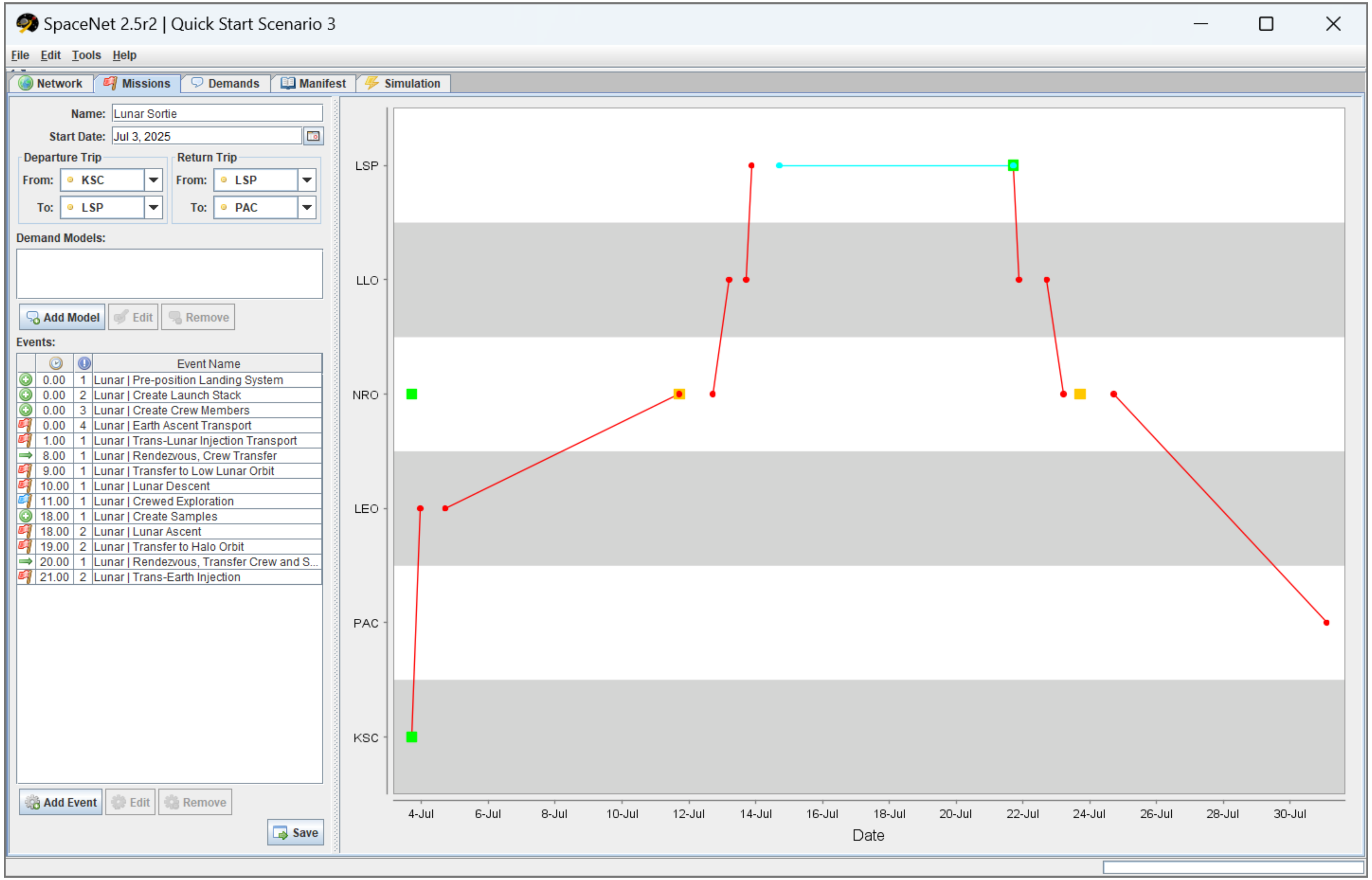
21.0 **Transport CM.1, SM.1 on NRO-PAC**

(1) Burn SM.1

(2) Burn SM.1

(3) Burn SM.1, Stage SM.1

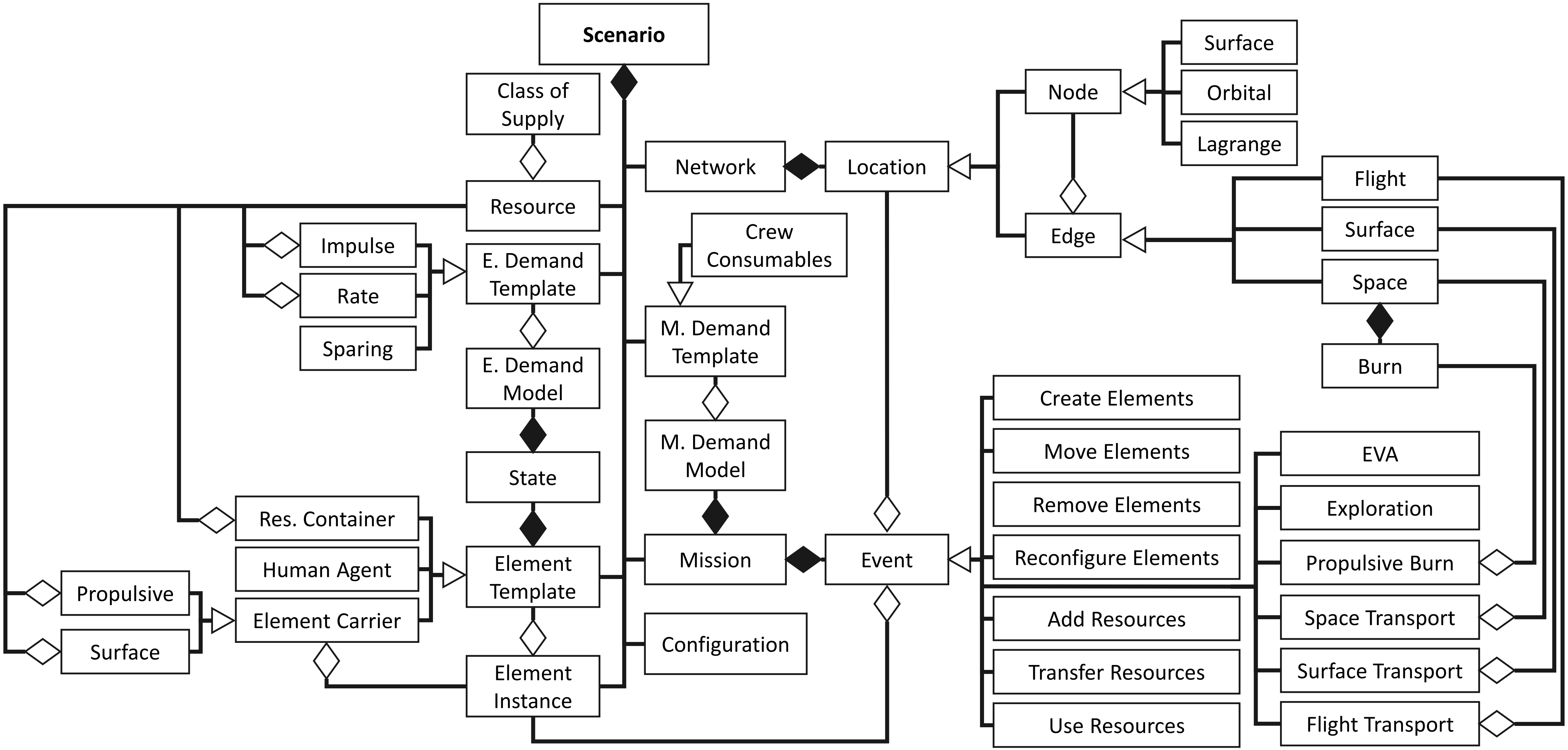
Artemis III Time-Expanded Network



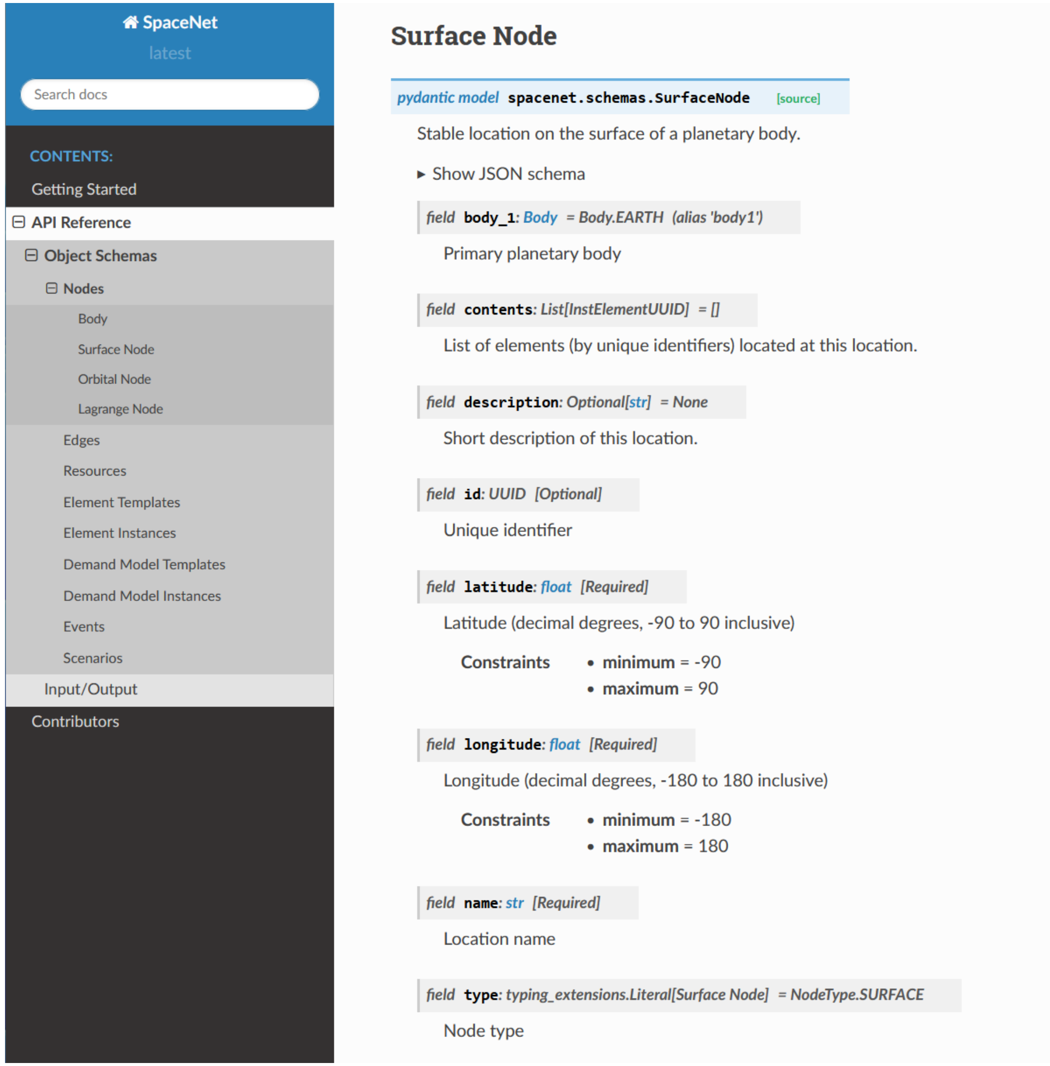
Propulsive Feasibility Analysis

Element	Initial Fuel (kg)	Final Fuel (kg)	Margin (%)
SRBs	1,256,000	0	0.0
Core Stage	987,000	20,807	2.1
Upper Stage	28,576	201	0.7
Landing System	32,285	1,656	5.1
Service Module	9,276	3,472	37.4

Scenario Object-Class Diagram (Simplified)



SpaceNet Object Schemas API



- SpaceNet “Cloud” code base:
  - https://github.com/space-logistics-org/spacenet
- Sphinx-based source code auto-documentation
- Published to ReadTheDocs:
  - https://spacenet.readthedocs.io/

Pydantic Models and JSON Schema

- Transition to JSON serialization over the past 2 years:
  - Pydanic models for each class
  - Exportable to JSON Schema

```
class SurfaceNode(Node):
    type: Literal[NodeType.SURFACE] = Field(
        NodeType.SURFACE,
        title="Type",
        description="Node type",
    )
    latitude: float = Field(
        ...,
        title="Latitude",
        description="Latitude (decimal degrees, -90 to 90 inclusive)",
        ge=-90,
        le=90,
    )
    longitude: float = Field(
        ...,
        title="Longitude",
        description="Longitude (decimal degrees, -180 to 180 inclusive)",
        ge=-180,
        le=180,
    )
```

- Programmatic scenario construction:

```
from spacenet import schemas

ksc = schemas.SurfaceNode(
    name="KSC",
    description="Kennedy Space Center",
    body1="Earth",
    latitude=28.6,
    longitude=-80.6,
)
leo = schemas.OrbitalNode(
    name="LEO",
    description="Low Earth Orbit",
    body1="Earth",
    inclination=28.5,
    periapsis=296.0,
    apoapsis=296.0,
)
ksc_leo = schemas.SpaceEdge(
    name="KSC-LEO",
    description="Earth Ascent",
    origin=ksc.id,
    destination=leo.id,
    duration=timedelta(hours=6),
    burns=[schemas.Burn(time=timedelta(0), delta_v=9500.0)],
)
```

JSON-serialized Scenario


```json
{
  "name": "Quick Start Scenario 3",
  "startDate": "2025-07-04T00:00:00+00:00",
  "scenarioType": "Lunar",
  "network": {
    "nodes": [
      {
        "id": "8dcb20f6-2d80-40c4-8a0b-571b768d7b7f",
        "name": "KSC",
        "description": "Kennedy Space Center",
        "contents": [],
        "type": "Surface Node",
        "body1": "Earth",
        "latitude": 28.6,
        "longitude": -80.6
      },
      {
        "id": "5117e973-31ca-4b6c-955e-62012b8f4b59",
        "name": "LEO",
        "description": "Low Earth Orbit",
        "contents": [],
        "type": "Orbital Node",
        "body1": "Earth",
        "apoapsis": 296.0,
        "periapsis": 296.0,
        "inclination": 28.5
      },
```

```json
    "edges": [
      {
        "id": "cfe856f2-c6a7-494b-aac8-7da95b920b0f",
        "name": "KSC-LEO",
        "description": "Earth Ascent",
        "contents": [],
        "type": "Space Edge",
        "origin": "8dcb20f6-2d80-40c4-8a0b-571b768d7b7f",
        "destination": "5117e973-31ca-4b6c-955e-62012b8f4b59",
        "duration": 21600.0,
        "burns": [
          {
            "id": "4db34826-c49a-4308-a2c7-d6410a743414",
            "time": 0.0,
            "deltaV": 9500.0
          }
        ]
      },
```

Current Development Work

- Transition SpaceNet Java analysis functions to SpaceNet Cloud:
  - Decompose monolithic application into discrete analysis services
  - Create well-defined input and output interfaces (JSON)
  - Host services on cloud/distributed computation platforms
- Create scenario for baseline Artemis Program campaign
  - Probe inconsistencies across multiple missions over 10+ year campaign
  - Evaluate propulsive and logistical feasibility
  - Evaluate logistics strategies such as pre-positioning, ISRU, depots, etc.

Questions and Answers

- Paul Grogan, paul.grogan@asu.edu

- Open-source software repositories:
  - SpaceNet Java: github.com/space-logistics-org/spacenet-java
  - SpaceNet “Cloud”: github.com/space-logistics-org/spacenet