JSC-61777-models/environment/ephemerides

JSC Engineering Orbital Dynamics Ephemerides Model

Simulation and Graphics Branch (ER7)
Software, Robotics, and Simulation Division
Engineering Directorate

Package Release JEOD v5.1

Document Revision 2.1
July 2023

National Aeronautics and Space Administration
Lyndon B. Johnson Space Center
Houston, Texas

JSC Engineering Orbital Dynamics Ephemerides Model

Document Revision 2.1
July 2023

David Hammen
Blair Thompson

Simulation and Graphics Branch (ER7)
Software, Robotics, and Simulation Division
Engineering Directorate

National Aeronautics and Space Administration
Lyndon B. Johnson Space Center
Houston, Texas

Executive Summary

The JEOD Ephemerides Model provides a generic framework for defining ephemeris models. The model comprises five sub-models:

Ephemeris Item Ephemeris models represent as functions of time the position and velocity of planetary bodies and barycenters of multiple planetary bodies. Some models also represent as functions of time the orientations and angular velocities of planetary bodies. The Ephemeris Item sub-model defines the classes needed to represent these planetary states.

Ephemeris Interface Any JEOD-based implementation of an ephemeris model needs to function properly with the rest of JEOD. One key aspect of this is the JEOD concept of a reference frame tree. JEOD uses reference frames to describe the states of objects in space. Reference frames link to other reference frames to form a reference frame tree. JEOD relies upon the ephemeris models to define the base of this reference frame tree. The Ephemeris Interface sub-model prescribes basic behaviors that any ephemeris model must follow in the form of the EphemerisInterface class.

Ephemeris Manager The RefFrameManager class defined in the Reference Frame Model provides basic capabilities regarding the management of a set of reference frames connected to form a reference frame tree. The Ephemeris Manager sub-model extends the RefFrameManager, adding capabilities related to managing planets, ephemeris items, and ephemeris models.

De4xx Ephemerides The Jet Propulsion Laboratory has a long history of modeling the behavior of the solar system in the form of ephemeris models. The most recent members of these models are the 400/403/405/... series of the JPL Development Ephemerides, or the DE4xx models for short. The De4xx Ephemerides sub-model provides the data for the DE405, DE421 and DE440 models and defines the class De4xxEphemeris, which extends the EphemerisInterface class.

Propagated Planet The propagated planet sub-model provides the ability to have planetary state computed via some ephemeris model or propagated by integrating the equations of motion. This can be particularly useful for objects such as asteroids for which an ephemeris model is not readily available.

Contents

List of Tables

Chapter 1

Introduction

1.1 Purpose and Objectives of the Ephemerides Model

This document describes the Ephemerides Model.[1] The Ephemerides Model addresses the problems of building the heavenly body basis of the JEOD reference frame tree and of representing the translational and rotational states of those heavenly bodies.

1.2 Context within JEOD

The following document is parent to this document:

- *JSC Engineering Orbital Dynamics* [8]

The Ephemerides Model forms a component of the environment suite of models within JEOD v5.1. It is located at models/environment/ephemerides.

1.3 Document History

Author	Date	Revision	Description
Thomas Brain	September, 2021	4.0	Updated
David Hammen	February, 2012	2.1	Updated
David Hammen	September, 2010	2.0	Updated
Blair Thompson	November, 2009	1.0	Initial document

[1] "An ephemeris is a tabulation of computed positions and velocities (and/or various derived quantities such as right ascension and declination) of an orbiting body at specific times. The plural form of ephemeris is ephemerides." Reference: JPL Solar System Dynamics web page http://ssd.jpl.nasa.gov.

1.4 Document Organization

This document is formatted in accordance with the NASA Software Engineering Requirements Standard [11].

The document comprises chapters organized as follows:

Chapter 1: Introduction -This introduction describes the objective and purpose of the Ephemerides Model.

Chapter 2: Product Requirements -The requirements chapter describes the requirements on the Ephemerides Model.

Chapter 3: Product Specification -The specification chapter describes the architecture and design of the Ephemerides Model.

Chapter 4: User Guide -The user guide chapter describes how to use the Ephemerides Model.

Chapter 5: Inspections, Tests, and Metrics -The inspections, tests, and metrics describes the procedures and results that demonstrate the satisfaction of the requirements for the Ephemerides Model.

Chapter 2

Product Requirements

2.1 Project Requirements

Requirement Ephemerides_1: Top-level Requirement

Requirement:
The Ephemerides Model shall meet the JEOD project requirements specified in the JEOD v5.1 top-level document.

Rationale:
This model shall, at a minimum, meet all external and internal requirements applied to the JEOD v5.1 release.

Verification:
Inspection

Requirement Ephemerides_2: Ephemeris Items

Requirement:
The Ephemerides Model shall provide the ability to represent and operate on the ephemeris items described by some ephemeris model.

2.1 Basic capabilities. Each ephemeris item shall have a settable and gettable name and timestamp and shall be capable of being enabled/disabled and activated/deactivated.

2.2 Position and velocity of a point in space. An ephemeris point shall extend the ephemeris item capabilities by providing the ability to represent and update the translational state (position and velocity) of some point in space associated with some inertial frame of reference.

2.3 Position and velocity of a planet. An ephemeris planet shall extend the ephemeris point capabilities for points in space associated with the center of mass of a planetary body.

2.4 Orientation and angular velocity of a planet. An ephemeris orientation shall extend the ephemeris item capabilities by providing the ability to represent and update the rotational state (orientation and angular velocity) of some planet-fixed reference frame.

Rationale:

The JPL Developmental Ephemerides provide position and velocity of the Sun, the planets and the Moon and also provide the orientation and angular velocity of the Moon.

Verification:

Inspection, Test

Requirement Ephemerides_3: Ephemeris Interface

Requirement:

An ephemeris model provides state information on one or more planetary object. A framework that characterizes in generic terms what constitutes a JEOD-compliant ephemeris model is needed to enable the use of different ephemerides in JEOD.

3.1 Base class. The Ephemerides Model shall define a framework that prescribes the behavior of any implementation of a JEOD-compliant ephemeris model.

3.2 Derived classes. A JEOD-compliant implementation of an ephemeris model shall be capable of being initialized, activated, updated, and shall build the elements of the reference frame tree described by the ephemeris model consonant with the way in which JEOD calculates gravitational acceleration.

Rationale:

This requirement specifies an object-oriented scheme in which a base class defines but does not implement the behaviors levied on the derived classes.

The requirement to construct the reference frame tree consonant with gravity calculations can be met in at least three ways:

- A flat scheme with the system barycenter at the root of the tree and all other ephemeris-based reference frames descending directly from this root,
- A hierarchical scheme with the system barycenter at the root, subsystem barycenters (planets plus moons) descending from the system barycenter, and a planet and its moons descending from the subsystem barycenter,
- A mixed scheme.

Verification:

Inspection

Requirement Ephemerides_4: Ephemeris Manager

Requirement:

The JEOD reference frame manager provides a base capability for managing a set of reference frames that are connected in the form of a tree. The Ephemerides Model ephemerides manager extends these base capabilities.

4.1 Registration Services. The Ephemerides Model shall provide registration and lookup capabilities for planets, ephemeris items, and integration frames.

4.2 Ephemeris Model Management. The Ephemerides Model shall provide the ability to register ephemeris models with the Ephemerides Model and shall coordinate the initialization, activation, and updating of the registered ephemeris models.

Rationale:
The JEOD 2.0 implementation of the dynamics manager was overly complex. Functionality that pertained to reference frames only was moved to the reference frame manager sub-model while functionality that pertained to items related to ephemerides was moved to the ephemeris manager sub-model. This is an internal requirement that reflects this architectural decision.

Verification:
Inspection, Test

Requirement Ephemerides_5: Solar System Representation

Requirement:
The Ephemerides Model shall provide the ability to represent:

5.1 Planetary position and velocity. The model shall provide the ability to represent the position and velocity of the Sun, the planets, Pluto, and the Moon.

5.2 Lunar orientation and angular velocity. The model shall provide the ability to represent the orientation and angular velocity of the Moon.

5.3 JPL Development Ephemerides. The model shall use the JPL Development Ephemerides as the basis for these state representations.

Rationale:
These specified items are those represented in the JPL Development Ephemerides. The JPL DE4xx series represents the most recent of a very long history of planetary ephemeris models.

Verification:
Inspection, Test

Requirement Ephemerides_6: Development Ephemeris Implementation

Requirement:
The implementation of the DE4xx ephemeris model shall satisfy the constraints levied in requirement Ephemerides_3 on extensions to the base ephemeris interface capabilities.

Rationale:
This requirement is a distinguishing feature of JEOD 2.x versus JEOD 1.x. It changes the model from a set of routines into a set of classes that fits into the overall architecture.

Verification:
Inspection, Test

Requirement Ephemerides_7: Propagated Planet

Requirement:

The Ephemerides Model shall provide the ability to:

7.1 Ephemeris Mode. The model shall provide the ability to compute the position and velocity of a planetary body via some ephemeris model.

7.2 Propagated Mode. The model shall provide the ability to compute the position and velocity of a planetary body by propagating the state over time per the gravitational forces that act on the body.

7.3 Mode Change. The model shall provide the ability to dynamically switch the mechanism used to compute state from ephemeris mode to propagated mode.

Rationale:

This is an external requirement levied upon JEOD by its customers.

Verification:

Inspection, Test

Chapter 3

Product Specification

3.1 Conceptual Design

The Ephemerides Model addresses the problems of building the heavenly body basis of the JEOD reference frame tree and of representing the translational and rotational states of those heavenly bodies. This section describes the conceptual design of the model. The first subsection describes concepts that are critical in understanding the model. The second subsection describes the modules that form the model.

3.1.1 Key Concepts

3.1.1.1 Reference Frames and the Reference Frame Tree

The JEOD concept of a reference frame is the centralizing feature of JEOD. Planets contain reference frames, vehicles contain reference frames, and various derived frames augment these planetary and vehicular frames. Reference frames contain state and connectivity information. The state aspect of a reference frame provides the means to represent the translational and rotational states of the entity associated with the reference frame. The connectivity aspect of a reference frame provides the means to connect one reference frame to another, eventually forming the JEOD reference frame tree. All active reference frames are a part of this reference frame tree.

The Ephemerides Model plays a key role in forming the basis of the reference frame tree and in populating the states of those frames that represent heavenly bodies. The root of the reference frame tree is the one true inertial reference frame of a simulation. (This assumes the existence of a Newtonian inertial reference frame. There is no such thing, but the errors are negligibly small with the appropriate choice.) This root frame is always established by some ephemeris model. The frames representing the Sun and the planets descend from this root frame. The connectivities between these heavenly body frames and the states of these frames are also established by some ephemeris model.

The ephemeris models must construct the heavenly body portion of the JEOD reference frame tree in a way that is consistent with how the Gravity Model computes gravitation. That model calculates gravitational acceleration per Newton's law of gravitation for gravitating bodies that are

at or below a vehicle's integration frame, but calculated it as a third body effect otherwise.

3.1.1.2 Ephemeris Model

As used in JEOD, an ephemeris model is a model that provides a means for determining the translational and/or rotational states of heavenly bodies as a function of time. Multiple JEOD models make use of various ephemerides. Ephemerides are essential in propagating the state of a vehicle that is gravitationally attracted to multiple heavenly bodies. The need for ephemeris models in JEOD goes much deeper than this. An ephemeris model of some sort is needed in almost every JEOD-based simulation. All vehicle-based reference frames are represented with respect to some parent ephemeris-based reference frame.

3.1.1.3 JPL Development Ephemeris Models

The Russian Institute for Applied Astronomy, the Paris Observatory, and the Jet Propulsion Laboratory (JPL) are the three leading institutions that provide high precision ephemerides of solar system bodies. JEOD uses the JPL Development Ephemeris (DE) models. Three such models are provided with JEOD, the DE405, DE421 and DE440 models.

3.1.1.4 Ephemeris Item

A JEOD ephemeris model represents the translational and/or rotational states of one or more heavenly bodies as a function of time. Note the Reference Frame Model already provides the standard mechanism in JEOD for representing such states. An issue can arise with the use of multiple ephemeris models: What if multiple ephemeris models provide potentially conflicting representations of the same quantity? The solution to this problem is to provide a bridge between the ephemeris models and the states they can provide. The ephemeris items are the abstract representation of this bridge.

3.1.1.5 Ephemeris Manager

The ephemeris models collectively form the basis of the JEOD reference frame tree. Something is needed to ride herd over the ephemeris models to make this happen. This 'something' is the ephemeris manager.

3.1.1.6 Subscription

Computing planetary ephemerides would pose an unnecessary computational expense were all possible heavenly bodies represented in every JEOD-based simulation. To avoid this problem, the reference frame tree is constructed and states are computed only for those bodies needed elsewhere in the simulation. This need is assessed based on the reference frame subscription model. Models that need to use some aspect of a reference frame register a subscription to that frame. The Ephemerides Model components build the heavenly body basis of the reference frame tree based

upon these subscriptions. Only those items that are needed are placed in the active reference frame tree.

3.1.2 Model Components

The Ephemerides Model comprises a set of modules that address different aspects of the problems that need to be solved by the model. Each such module is a separate subdirectory of the root model directory, `$JEOD_HOME/models/environment/ephemerides`.

3.1.2.1 Ephemeris Item

The ephemeris item module (model subdirectory ephem_item) implements the concept of an ephemeris item. The class `EphemerisItem` establishes the abstract basis of this concept. The derived class `EphemerisPoint` pertains to translational states while `EphemerisOrientation` pertains to rotational states. A specialization of the `EphemerisOrientation` class, `EphemerisZXZOrientation`, pertains to ephemeris models that represent orientation in terms of a classical (z-x-z) Euler sequence. Figure 3.1 depicts the ephemeris items defined in the Ephemerides Model.

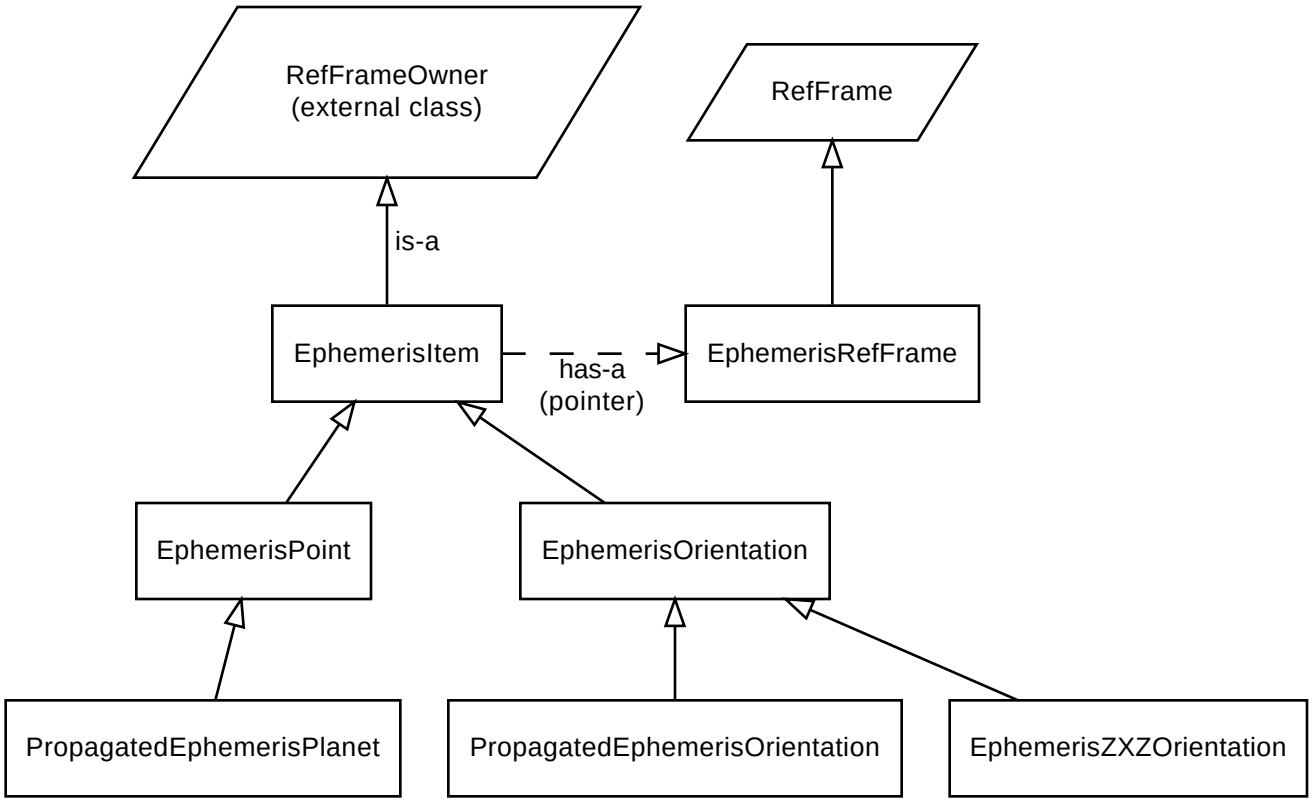

Figure 3.1: Ephemeris Items and Ephemeris Reference Frames

3.1.2.2 Ephemeris Interface

The ephemeris interface module (model subdirectory ephem_interface) defines several classes.

Ephemerides Model Messages The class `EphemeridesMessages` is a standard message identifier class. It defines the message identifiers for all of the messages generated by the Ephemerides Model.

Ephemeris Reference Frame The class `EphemerisRefFrame` (depicted in figure 3.1), derives from the `RefFrame` class.

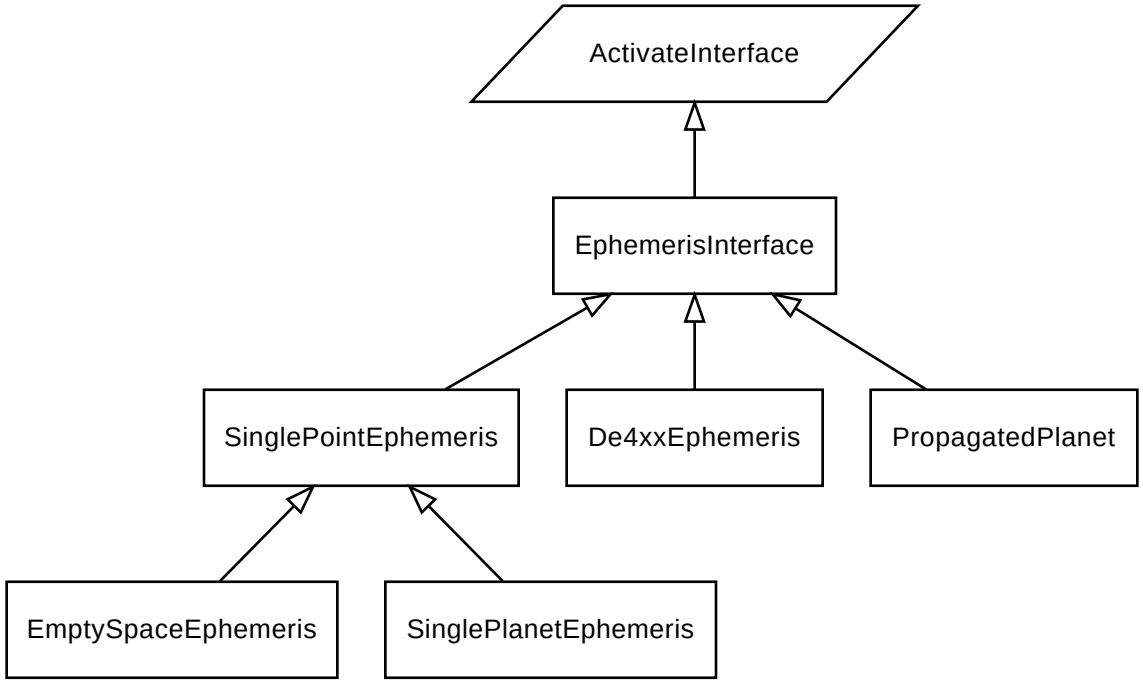

Figure 3.2: Ephemeris Interface

Ephemeris Interface The class `EphemerisInterface` defines, in an abstract sense, the basic characteristics of an ephemeris model. All JEOD ephemeris models are an instances of some class that derives from `EphemerisInterface`. An ephemeris model will be called upon to build its share of the reference frame tree via a prescribed sequence of actions. The model will also be called to update the ephemeris items for which it is responsible. The former happen at initialization time and whenever the reference tree might change due to subscription changes; the updates are performed on a scheduled basis. Figure 3.2 depicts the ephemeris models defined in the Ephemerides Model.

Single Point Ephemeris Very simple simulations only need a very simple ephemeris model, one that describes the sole inertial frame in the simulation. The class `SinglePointEphemeris` provides a simple model where there is but one EphemerisPoint in the entire simulation.

Empty Space Ephemeris The simplest of all simulations involves vehicles operating in otherwise empty space. The class `EmptySpaceEphemeris` specializes the class `SinglePointEphemeris`

for use in simulations that does not involve any gravitating bodies: Empty space. The root node of the reference frame tree is the one and only ephemeris reference frame in the simulation.

Single Planet Ephemeris The next step up in complexity involves simulations about a single planetary object where the gravitational influences of all other planetary bodies can be ignored. The class `SinglePlanetEphemeris` specializes the class `SinglePointEphemeris` for use in simulations that involve but one gravitating body. The root node of the reference frame tree is the planet-centered inertial frame for that planet.

3.1.2.3 Ephemeris Manager

The ephemeris manager module (model subdirectory ephem_manager) defines the class `EphemerisManager`. This class derives from the `RefFrameManager` class, and in turn forms the basis for the `DynManager` class. There is exactly one `DynManager` instance in every JEOD-based simulation, and hence exactly one instance of `EphemerisManager`.

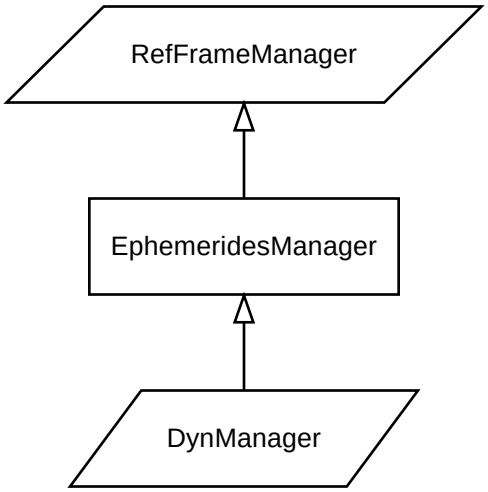

Figure 3.3: Manager Classes

3.1.2.4 JPL Development Ephemeris

The Development Ephemeris module (model subdirectory de4xx_ephem) defines a number of classes pertaining to reading and using a JPL Development Ephemeris (DE) model.

These JPL ephemerides have been made publicly available in the form of Chebyshev polynomial coefficients. The DE405, DE421 and DE440 models, used by the Ephemerides Model, are well accepted in the aerospace industry. It covers roughly 600 years from 1600 to 2170 [9] with DE440 covering 1550 to 2550.

These models are based on least-squares fits of observation data followed by a rigorous numerical integration of the equations of motion of the solar system bodies [7]. The observation data consist

of optical transits of the Sun and planets since 1911, radar ranging to Mercury and Venus since 1964, tracking of deep space probes, planetary orbiters and landers since 1971, and lunar laser ranging since 1970 [9]. The force models include point mass gravity of the Moon, planets, Sun, and selected asteroids; a relativistic post-Newtonian correction; luni-solar torques on the figure of the Earth; and Earth and Sun torques on the figure of the Moon [9, 13]. Numerical integration of equations of motion used a variable step-size, variable order Adams method. The maximum allowable integration order was 14 [13].

Additional information regarding the Development Ephemeris models (and other information) can be found on the JPL Solar System Dynamics web page *http://ssd.jpl.nasa.gov*. The web page has an online ephemeris application tool called HORIZONS.

The implementation of the Development Ephemeris module defines several classes.

Development Ephemeris The class `De4xxEphemeris` is an ephemeris model (the class derives from `EphemerisInterface`) and provides the external interface to the Development Ephemeris module. This class contains several `EphemerisItem` objects (described above) and a single `De4xxFile` object (described below).

Ephemeris File The class `De4xxFile` encapsulates the processes of reading and interpreting the contents of a JPL ephemeris file. This class contains instances of several classes and structs intended for use in this class only.

`De4xxFileHeader` contains the header of a Development Ephemeris file.

`De4xxFileCoef` contains the Chebyshev polynomial coefficients that form the bulk of a Development Ephemeris file.

`De4xxFileIO` provides the library and symbol loading mechanism to obtain data from the Development Ephemeris source data

`De4xxFileItem` contains a description of a single item (a planet position, barycenter position, or lunar orientation) in a Development Ephemeris file.

`De4xxFileRefTime` converts time as represented in JEOD to time as represented in the a Development Ephemeris file.

`De4xxFileRestart` reopens the relevant Development Ephemeris file on restart, making a `De4xxFile` object (and by extension, the containing `De4xxEphem` object) checkpointable and restartable.

`De4xxFileSpec` specifies the Development Ephemeris file that is to be used in a simulation.

`EphemerisDataSetMeta` contains properties pertaining to a Development Ephemeris version retrieved from its shared library

`EphemerisDataItemMeta` contains properties for an ephemeris item modeled within a DE version and retrieved from its shared library

`EphemerisDataSegmentMeta` contains properties for a portion or segment of a DE model usually split up by a span of 20 or 50 years retrieved from its shared library

Enumerations The class `De4xxBase` encapsulates two enumerations used in the module. One enumeration, `De4xxFileBodies`, names the planetary items in the order implemented in a JPL ephemeris file. The other, `De4xxEphemBodies`, names the modeled items in the order they are implemented in the `De4xxEphemeris` class.

3.1.2.5 Propagated Planet

The propagated planet module (model subdirectory propagated_planet) provides planetary state via a DynBody object whose state can be propagated using the JEOD state integration techniques. Scenarios in which a simulation will use a PropagatedPlanet object include:

- An object such as an asteroid for which an ephemeris model is not readily available.
- An object such as a planet that is represented in some other ephemeris model but the simulation developer wants the planet to be propagated to ensure that the planet and the vehicles operating in the vicinity of the planet obey the same laws of physics.

The propagated planet module provides mechanisms that accommodate these scenarios. The class PropagatedPlanet defines multiple modes in which a propagated planet planet object operates. In all modes, the model ensures consistency between the translational states of the dynamic body's composite frame and the planet's planet-centered frame and between the rotational states of the dynamic body's composite frame and the planet's planet-fixed frame.

3.2 Mathematical Formulations

3.2.1 Chebyshev Approximation

The Development Ephemeris models released for public use contain Chebyshev polynomial coefficients that describe the positions of the planets as functions of time. This polynomial representation requires significantly fewer data points than the complete set of position and velocity vectors for all solar system bodies at each time step in the numerical integration of the equations of motion. The polynomial form also allows users to readily interpolate between the fixed time steps of the ephemeris as required.

The full time span of the DE405 ephemeris of each solar system body was segmented into contiguous intervals (i.e. granules) of fixed length. Different bodies were given different granule lengths as required to ensure the desired level of accuracy. Nine pairs of position and velocity values were selected for each granule at equally spaced times, one pair at each endpoint and seven pairs in between. A constrained least-squares technique was used to fit Chebyshev coefficients to the position and velocity pairs such that the resulting Chebyshev polynomials were an exact fit at the endpoints. This ensured continuity between adjacent granules [9, 13].

Chebyshev polynomials are defined as

$$T_n(\tau) = \cos(n \cdot \arccos \tau) \tag{3.1}$$

where $|\tau| \leq 1$. The polynomials have the following recursive properties:

$$\begin{aligned} T_0(\tau) &= 1 \\ T_1(\tau) &= \tau \\ T_{n+1}(\tau) &= 2\tau T_n(\tau) - T_{n-1}(\tau) \text{ for } n \geq 1 \end{aligned} \tag{3.2}$$

A function $f(t)$ can be represented with a Chebyshev polynomial from the following approximation

$$f(t) \approx \sum_{i=0}^{n-1} a_i T_i(\tau) \tag{3.3}$$

where a_i are the coefficients determined by least-squares adjustment.

The function input time (t) must fall within the interval of the appropriate granule of data, $t_1 \leq t \leq t_2$. Because the Chebyshev polynomials are only valid in the interval $-1 \leq \tau \leq 1$, the reference time t is mapped to the associated time τ in the allowable polynomial interval using the equation

$$\frac{t - t_1}{t_2 - t_1} = \frac{\tau - (-1)}{1 - (-1)} \tag{3.4}$$

Solving for τ yields

$$\tau = 2\frac{t - t_1}{t_2 - t_1} - 1 \tag{3.5}$$

The time derivatives of the Chebyshev polynomials exist as analytical expressions. These expressions could be used to compute velocities from position polynomials. In the case of the DE405 ephemeris, Chebyshev polynomials were fit directly to the velocity components resulting from numerical integration. No differentiation is required in order to approximate velocity.

3.3 Interactions

3.3.1 JEOD Models Used by the Ephemerides Model

The Ephemerides Model uses the following JEOD models:

- *Container Model* [1].
  The Ephemerides Model uses registry as the basis for checkpointing the contents of the checkpointable objects at checkpoint time and for restoring the contents of the checkpointable objects at restart time.
- *Dynamic Body Model* [2].
  The propagated planet module models planets as dynamic bodies so that they can be propagated if desired.
- *Mathematical Functions Model* [19].
  The `EphemerisZXZOrientation` class uses the vector capabilities defined in the Mathematical Functions Model.
- *Memory Management Model* [3].
  As do almost all other models, the Ephemerides Model uses the Message Handler Model to manage dynamically-allocated memory.
- *Message Handler Model* [14].
  As do almost all other models, the Ephemerides Model uses the Message Handler Model to generate messages.

- *Named Item Model* [15].
  Various modules use the Named Item Model to construct and validate names.

- *Planet Model* [10].
  Ephemerides that pertain to planetary bodies (which includes the Sun) are stored in one of the reference frames contained in a `BasicPlanet` object.

- *Quaternion Model* [4].
  The `EphemerisZXZOrientation` class uses quaternions to propagate the rotational state between updates.

- *Reference Frame Model* [17].
  External users of the Ephemerides Model access ephemerides via the reference frames in which those ephemerides are stored.

- *Simulation Engine Interface Model* [16].
  As do almost all other models, the Ephemerides Model uses the Simulation Engine Interface Model to make model data and functions visible to the simulation engine.

- *Time Model* [21].
  The DE4xx ephemeris and propagated planet modules use the the Time Model to timestamp reference frames. The DE4xx ephemeris module also uses the Time Model to obtain the current Terrestrial Time. The JPL ephemerides are Chebyshev polynomials in time.

3.3.2 JEOD Models That Use the Ephemerides Model

- Multiple models.
  The *Body Action Model* [5], *Dynamic Body Model* [2], and *Gravity Model* [20] use pointers and references to instances of the `EphemerisRefFrame` class and use the `EphemeridesManager` to find reference frames and planets.

- *Dynamics Manager Model* [6].
  The `DynManager` class defined in the Dynamics Manager Model derives from the `EphemeridesManager` class. The `DynamicsIntegrationGroup` class uses the `EphemeridesManager` (via the `DynManager`) to update ephemerides.

- *Planet Model* [10].
  The `BasePlanet` class defined in the Planet Model contains instances of the `EphemerisRefFrame` class, one for the planet-centered inertial frame and the other for the planet-centered, planet-fixed frame. `BasePlanet` instances register themselves and the frames they contain with the `EphemeridesManager`.

- *Rotation, Nutation, and Precession Model* [18].
  The `PlanetOrientation` class defined in the Rotation, Nutation, and Precession Model derives from the `EphemerisInterface` class and contains an `EphemerisOrientation` object. `PlanetOrientation` instances register themselves as ephemeris models with the `EphemeridesManager`.

3.4 Detailed Design

Details of the design of the JEOD Ephemerides Model can be found in JEOD Ephemerides Model Reference Manual [12].

3.5 Inventory

All Ephemerides Model files are located in `${JEOD_HOME}/models/environment/ephemerides`. Relative to this directory,

- Model header and source files are located in model `include` and `src` subdirectories. See table **??** for a list of these configuration-managed files.
- Model documentation files are located in the model `docs` subdirectory. See table **??** for a list of the configuration-managed files in this directory.

Chapter 4

User Guide

4.1 Instructions for Simulation Users

This section describes how to set up and use the Ephemerides Model in an existing Trick 10 simulation.

4.1.1 Controls

The simulation developer typically establishes which ephemeris models are employed in a given simulation. In most simulations, the simulation user should leave this as-is.

The user may want to specify the JPL Development Ephemeris version which can be done via input file. The default configuration is set to DE405. To change this, simply add the following line in the input file.

```
env.de4xx.set_model_number(<integer value of model>)
## Ex.
env.de4xx.set_model_number(440)
```

JEOD currently provides versions 405, 421, and 440.

The JEOD dynamics manager and dynamics integration provides the ability to control whether ephemerides are updated as derivative class jobs. This can be set from the input file.

```
dynamics.dyn_manager.deriv_ephem_update = <True or False>
```

Note however that overriding this from the default for your simulation may invalidate the simulation results.

4.1.2 Accessing Ephemerides

In general, you do not access the ephemerides directly from the ephemeris models. You instead those data from those planet reference frames. The ephemeris models store the translational state

of a planet in the planet's inertial reference frame, the rotational state of a planet in the planet's planet-fixed reference frame state.

The one exception to this are the translational states of various barycentric frames. These frames are contained in the ephemeris models. Assuming the environment simulation object is established as described in section 4.2, the Earth-Moon barycenter frame is the simulation variable `env.de4xx.earth_moon_barycenter_frame`. The solar system barycenter is also present in the JPL ephemeris model, but its translational state is identically zero and its rotational state is the identity rotation.

The recommended practice for logging these ephemerides data in a Trick 10 simulation is to use the JEOD logger Python module. The following assumes each planet is represented as a Trick10 simulation object that contains a data member named `planet` of type `Planet` and whose name is the name of the planet. (This is the recommended practice for defining JEOD planetary objects in Trick10).

```
# Load the JEOD logger module.
import sys
import os
sys.path.append ('/'.join([os.getenv("JEOD_HOME"), "lib/jeod/python"]))
import jeod_log

def log_trans_states (data_record, interval, file_suffix, \
                      planets, other_frames=()) :
  """
  Log the states of the specified reference frames.

  Arguments:
  data_record -- Trick data recorder sim object.
  interval    -- Simulation time between successive data blocks.
  file_suffix -- Suffix for the log file.
  planets     -- Comma separated list of names of sim objects that
  contain planets; the planet's translational states are logged.
  other_frames -- Comma separated list of names of additional reference
  frames whose translational states are to be logged.
  """

  # Create and initialize the logger.
  logger = jeod_log.Logger (data_record.drd)
  logger.open_group (interval, file_suffix)

  # Log the translational states for the specified planets
  for planet in planets :
     logger.log_ref_frame_trans_state (planet + ".planet.inertial")

  # Log the additional translational states
  for frame in other_frames :
    logger.log_ref_frame_trans_state (frame)
```

```
    # Close the logger.
    logger.close_group ()

    return
```

Using the above Python function, the translational states of the Sun, Earth, Moon, and Earth-Moon barycenter can be logged via

```
log_trans_states (data_record, interval, "state", \
                  ["sun", "earth", "moon"], \
                  ["env.de4xx.earth_moon_barycenter_frame"])
```

Refer to the User Guide chapters of *Planet Model* [10] and *Reference Frame Model* [17] for details on using the Planet and RefFrame classes.

4.1.3 Accessing Additional Ephemerides

Simulation users have on occasion express a desire to access ephemerides for planets that are not represented in the simulation. One approach is to add Trick10 simulation objects to the simulation that represents those additional planetary bodies. Subscriptions can be created for these extraneous bodies in the input file.

An alternative approach is to create those extra planets in the input file, thereby avoiding altering the S_define file just to access these auxiliary data. The following illustrates this with the planets Uranus and Neptune.

```
# Create a planet and name it Uranus (note use of lower and upper case).
uranus = trick.Planet()
uranus.set_name("Uranus")

# Register the planet with JEOD and issue a subscription.
uranus.register_planet(dynamics.dyn_manager)
uranus.inertial.subscribe()

# Register the object with Trick.
trick.TMM_declare_ext_var_s(uranus, "Planet uranus")

# Now do the same for Neptune.
neptune = trick.Planet()
neptune.set_name("Neptune")
neptune.register_planet(dynamics.dyn_manager)
neptune.inertial.subscribe()
trick.TMM_declare_ext_var_s(neptune, "Planet neptune")
```

The states of these additional parameters can be logged by changing the call to the Python function `log_trans_states` described in section 4.1.2 above.

```
log_trans_states (data_record, interval, "state", \
                  ["sun", "earth", "moon"], \
                  ["env.de4xx.earth_moon_barycenter_frame", \
                   "uranus.inertial", "neptune.inertial"])
```

4.2 Instructions for Simulation Developers

This section describes how to create a Trick 10 simulation that uses the Ephemerides Model.

4.2.1 The `env` Simulation Object

JEOD provides a Trick10 `S_module` file that defines the standard JEOD `env` simulation object. This Trick10 simulation object contains an instance of the De4xxEphemeris class. Place the following in your `S_define` file, just after the dynamics manager simulation object:

```
#include "JEOD_S_modules/environment.sm"
```

where `JEOD_S_modules` is a symbolic link to `$JEOD_HOME/sims/shared/Trick10/S_modules`.

4.2.2 The `S_overrides.mk` File

The Development Ephemeris data provided with JEOD is in the form of C++ source files that are compiled into shared libraries. The DE4XX ephemeris model loads the symbols from the shared library as needed per DE4XX version and by segments of time. These shared libraries are compiled outside of the standard model source code Trick build process. However, Trick provides a mechanism in the `S_overrides.mk` file by which simulation developers can augment the build process. JEOD provides the make rules and cmake commands necessary to build the shared libraries and link them to a convenient location within the standard Trick build directory. Developers need only include `$JEOD_HOME/bin/jeod/generic_S_overrides.mk` in their simulations' `S_overrides.mk` file.

```
include $(JEOD_HOME)/bin/jeod/generic_S_overrides.mk
```

In this version of JEOD, JPL Development Ephermis models DE405, DE421 and DE440 are all built for each versions' full range of dates. Only symbols associated with the requested date are loaded at runtime which keeps the memory usage low despite having the entire date range available.

4.2.3 Using other DE4XX Models

The standard JEOD `env` De4xxEphemeris model will default to DE405 version inside its constructor. To use any other version (currently DE421 and DE440), simply set the model number in the input file for the simulation run.

```
env.de4xx.set_model_number(421)
env.de4xx.set_model_number(440)
```

4.2.4 Lunar Orientation

The JPL Development Ephemeris module automatically updates lunar orientation as a part of the ephemerides update process if the orientation of the Moon is needed somewhere in the simulation. Nothing special needs to be done with regard to lunar orientation if ephemerides are updated at the derivative rate or if lunar orientation only needs to be updated along with all other ephemerides.

If ephemerides are updated as scheduled jobs but you need lunar orientation at the derivative rate (or at a faster scheduled rate) you will need to add a derivative class or scheduled job to your S_define file that calls the DE4xx ephemeris model's `propagate_lunar_rnp` function. For example, adding the following job to the `env` simulation object will force lunar orientation to be propagated between updates.

```
P0 ("derivative") env.de4xx.propagate_lunar_rnp ();
```

4.3 Instructions for Model Developers

This section describes how to use the Ephemerides Model in C++ code.

4.3.1 Using the Ephemerides Model

The programmatic use of the Ephemerides Model is for the most part limited to those developers who are also extending some aspect of the model or of some model that already uses the Ephemerides Model. The former usage is covered in the following section. For the latter, refer the the instructions for model developers in the model that is being extended.

4.3.2 Extending the Ephemerides Model

The `EphemerisInterface` class is the primary class that external users may need to extend. Key methods that need to be defined are described below.

4.3.2.1 `initialize_model`

The `initialize_model` function should be called early in the initialization process. The name `initialize_model` is a suggested name rather than a mandatory one; this method is typically called directly from the S_define file. This method should

- Perform model-specific initializations such as opening input files,
- Register the model itself with the ephemeris manager,
- Register with the ephemeris manager any reference frames that are contained directly in the model, and
- Register with the ephemeris manager all ephemeris items that describe the data the model is capable of computing.

The order in which the various ephemeris models' `initialize_model` method can be significant. Multiple models can register ephemeris items with the same name. The first model to register such an item is the winner. For example, suppose you are developing an ephemeris model of the Jovian moons, with Jupiter's center of mass as the central frame for this model. Your Jovian moon model should be registered after the JPL DE4xx model is registered. If the DE4xx model is active, that model will be responsible for updating Jupiter's planet-centered inertial frame. Your model will add the Jovian moons to the overall ephemerides, and they will be correctly connected to the rest of the solar system.

4.3.2.2 `ephem_initialize`

The `ephem_initialize` function is called by the ephemeris manager's `initialize_ephemerides` function, presumably after all of the ephemeris models in the simulation have completed their basic initialization and after all of the planets have registered themselves and their reference frames with the ephemeris manager. The subscription status of the ephemeris items is not yet known, but the existence (or lack thereof) of the underlying reference frames is known. For example, if your new ephemeris model can provide ephemerides for Planet X, but there is not Planet X in the simulation, your `initialize_ephemerides` function should disable the those capabilities related to modeling Planet X.

4.3.2.3 `ephem_activate`

The `ephem_activate` function is called by the ephemeris manager's `activate_ephemerides` function, in reverse ephemeris model registration order. This function in turn is called at initialization time and when changes in subscribed frames indicate that the reference frame tree needs to be rebuilt. The subscription status of the ephemeris reference frames is known at this time. The primary purpose of this function is to coordinate that known subscription status with the activities that will occur later on in your model.

4.3.2.4 `ephem_build_tree`

The `ephem_build_tree` function is called by the ephemeris manager's `activate_ephemerides` function, in normal ephemeris model registration order. The function is called after all ephemeris models have been activated. This function should set the root frame of the reference frame tree if the ephemeris model is responsible for that root node. All non-root nodes should be then connected to the correct parent node.

4.3.2.5 `ephem_update`

The `ephem_update` function is called by the ephemeris manager's `update_ephemerides` function. The ephemeris model should respond to this call by updating each reference frame state for which it is responsible. Each updated reference frame should be timestamped with the current dynamic time.

4.3.3 Adding a DE4XX model

To add a new DE4XX version to JEOD, use the following steps:

1. Set the JEOD_HOME environment variable

2. Download the DE4XX version from the JPL FTP server replacing XX with the desired version number.
   ftp://ssd.jpl.nasa.gov/pub/eph/planets/ascii/de4XX

   ```
   ## When complete, the downloaded files will be
   ## located at pub/eph/planets/ascii/de4XX
   wget -m -nH ftp://ssd.jpl.nasa.gov//pub/eph/planets/ascii/de4XX
   ```

3. Move the ascii files to the jeod de4xx ascii data directory
   `$JEOD_HOME/models/environment/ephemerides/de4xx_ephem/ascii_full_set`

   ```
   mv pub/eph/planets/ascii/de4XX/* \
       $JEOD_HOME/models/environment/ephemerides/de4xx_ephem/ascii_full_set
   ```

4. Convert the ascii to C++ source files

   ```
   cd $JEOD_HOME
   make -f bin/jeod/makefile REGEN_DE4XX_DATA=1
   ```

This will parse the ascii files and generate the resulting
`$JEOD_HOME/models/environment/ephemerides/de4xx_ephem/data/data_src/de4XX.cc` file.

Once the `de4XX.cc` file is generated, the shared library (`libde4XX.so`) will be built as part of the simulation build process.

4.4 Frequently Asked Questions

1. *How do I access ephemeris model data?*
   In general, you don't. The primary job of any ephemeris model is to store the requested ephemeris data as states of the appropriate reference frame objects. As a user, you should access states from the reference frames in which those data are stored rather than accessing them from some ephemeris model.

2. *How do I control which ephemerides are computed?*
   You do this by subscribing to the reference frame that contains the desired data. An ephemeris model updates the states of the active reference frames that the model "owns". Reference frame activity is controlled by subscription.

3. *How do I access ephemeris model data for objects that aren't in the simulation?*
   The easiest way to accomplish this is to via a Trick 10 simulation input file. No modifications to the simulation are needed. See section 4.1.3 for details.

4. *How do I use the DE421 or DE440 model?*
   JEOD v5.1 defaults to the DE405 model. Simply follow the instructions in section 4.2.3 for details.

5. *How do I make the JPL ephemeris model cover a longer time span?*
   If you are using the DE421 model, you can't. It covers July 29, 1899 to October 9, 2053. If you are using the DE405 model, override the default value of the `EPHEM_YEARS` make variable in your `S_overrides.mk` file. See section 4.2.2 for details.

6. *How do I update the lunar orientation?*
   Lunar orientation is automatically updated by the JPL Development Ephemeris module as a part of the ephemerides update process. See section 4.2.4 for those cases where this automated update is insufficient.

Chapter 5

Inspections, Tests, and Metrics

5.1 Inspection

This section describes the inspections of the Ephemerides Model.

Inspection Ephemerides_1: Top-level Inspection

This document structure, the code, and associated files have been inspected, and together satisfy requirement Ephemerides_1.

Inspection Ephemerides_2: Design Inspection

Table 5.1 summarizes the key elements of the implementation of the Ephemerides Model that satisfy requirements levied on the model. By inspection, the Ephemerides Model satisfies requirements Ephemerides_2 to Ephemerides_7.

Table 5.1: Design Inspection

Requirement	Satisfaction
Ephemerides_2 Ephemeris Items	The `Ephemerides Model/ephem_item` directory defines the classes and methods that implement this requirement.
Ephemerides_3 Ephemeris Interface	The `Ephemerides Model/ephem_interface` directory defines the classes and methods that implement this requirement.
Ephemerides_4 Ephemeris Manager	The `Ephemerides Model/ephem_manager` directory defines the classes and methods that implement this requirement.
Ephemerides_5 Solar System	The `Ephemerides Model/de4xx` directory defines the classes and methods that implement this requirement.

Continued on next page

Table 5.1: Design Inspection (continued from previous page)

Requirement	Satisfaction
Ephemerides_6 Ephemeris Interface Extension	The class De4xxEphemeris defined in the `Ephemerides Model/de4xx` directory inherits from the EphemerisInterface class and constructs the planetary aspects of the reference frame tree in the prescribed manner.
Ephemerides_7 Propagated Planet	The `Ephemerides Model/propagated_planet` directory defines the classes and methods that implement this requirement.

5.2 Tests

This section describes various tests conducted to verify and validate that the Ephemerides Model satisfies the requirements levied against it. The tests described in this section are archived in various `verif` directories in the JEOD directory `models/environment/ephemerides`.

Test Ephemerides_1: DE4xx Unit Test

Purpose:

The purpose of this test is to verify that the Ephemerides Model correctly compute requested ephemerides for various solar system bodies.

Requirements:

By passing this test, the Ephemerides Model satisfies requirements Ephemerides_2, Ephemerides_3, Ephemerides_5, and Ephemerides_6.

Procedure:

The Ephemerides Model include a test program for verifying proper routine function. To run the test program, go to the `verif/unit_tests/de4xx_ephem` directory and type `make`. This will create the test article, the required binary ephemeris files, and conduct a series of tests based on the 2004 Venus transit of the Sun (08:19:44 UTC, June 8, 2004) and on the validation data supplied by JPL. The test is run four times, covering the set product (big endian, little endian)$\times$(DE405, DE421).

Results:

The test program runs two tests for each ephemeris file, a Venus transit test and a verification test.

The test program uses the ephemeris model to compute the position and velocity of the Sun, Venus, and the Earth at the time of the 2004 transit of Venus. These are compared to values obtained from the JPL HORIZONS system. The angular separation between the position of Venus as seen from the Earth as opposed to the position of the Sun as seen from the Earth is compared to the known value at the time of this transit. All comparisons must pass defined thresholds for the transit test to pass as a whole.

After the Venus transit tests, the test program then reads the appropriate verification file (`testpo.405` or `testpo.421`). This file, supplied by JPL, specifies various states for various bodies at various times. The test program uses the model to compute the corresponding state. The computed and provided values must agree to within a JPL-prescribed threshold for the comparison to pass. All comparisons must pass for this test to pass as a whole.

The results of one such test run are shown below.

```
./test_program -model 405 -endian little
================================================================================

JEOD epehemeris model test
  Ephemeris file : JPL-DE405.little_endian.bin
  DE model       : 405
```

```
  File type      : little endian
  Machine type   : little endian
  Byteswapping   : off

-----------------Venus transit test - 08:19:44 UTC June 8, 2004-----------------

Earth-Moon barycenter position and velocity check:
  Position vector magnitude at 08:19:44 UTC June 8, 2004
    value from JPL Horizons             = 1.520e+11 m
    value from JEOD ephemeris model     = 1.520e+11 m
    relative error                      = -2.2e-13

  Velocity vector magnitude at 08:19:44 UTC June 8, 2004
    value from JPL Horizons             = 2.934e+04 m/s
    value from JEOD ephemeris model     = 2.934e+04 m/s
    relative error                      = 1.97e-13


Venus-Earth-Sun angular separation check:
  Time = 08:19:44 UTC June 8, 2004
   Expected value                     = 626.9 arc sec
   Computed value                     = 626.9 arc sec
   error                              = 0.010 arc sec

Venus transit test status: Passed

-------------------------------------------------------------------------------


----------------------------Ephemeris dataset test-----------------------------

Relative position tests: Count of tested pairs, by planet indices
   |    1   2   3   4   5   6   7   8   9  10  11  12  13
---+-----------------------------------------------------
 1 |    -   3   5   8   4   7   8   4   7   2   1   9   6
 2 |    6   -   4   4   2   1   9   4   9   5   1   4   7
 3 |    4   5   -   2  10   2   4   5   7   6   3   5   8
 4 |    5   3   4   -   2   5   6   1   5  10   6   6   4
 5 |    5   6   5   4   -   6   3   5   5   8   5   8   5
 6 |    4   3   5   3   3   -  11   5   8   7   4   5   6
 7 |    2   5   9   5   7   0   -  10   4   3   6   7   8
 8 |    7   5   6   4   3   5   7   -   1   4   8   3   3
 9 |    2  10   4   2   1   4   5   5   -   8   6   1   4
10 |    3   2   4   4   4   4  10   3   2   -   3   2   4
11 |    5   3   3   0   3   5   5   2   5   3   -   7   4
12 |    6   6   6   2   8   2   5   4   3   3   6   -   4
```

```
13 |    2   2   4   5   7   7   2   6   5   5   4   3   -

Relative velocity tests: Count of tested pairs, by planet indices
   |    1   2   3   4   5   6   7   8   9  10  11  12  13
---+-----------------------------------------------------
 1 |    -   4   5   5   3   2   8   3   5   3   3   7   5
 2 |    4   -   6   5  10   4   7   8   3   3   6   8   5
 3 |    6   8   -   3   0   4   5   8   5   6   6   5   4
 4 |    3   5   2   -   4   4   4   7   4   4   7   2   3
 5 |    5   3   2   4   -   3   5   4   5   3   4   5   4
 6 |    6   5   8   7   2   -   8   4   3   4   4   0   4
 7 |    8   4   4   9   9   8   -   7   4   3   2   4   2
 8 |    3   5   9   4   2   6   5   -   4   3   5   4   5
 9 |    6   6   8   3   3   3   1   3   -   5   8   0   6
10 |    3   3   7   3   8   6   8   4   6   -   6   4   1
11 |    6   8   5   4   2   7   4   8   3   3   -   6   5
12 |    2   4   5   6   7   4   5   5   6   4   4   -   3
13 |    4   6   8   1   6   6   6   7   3   3   4   9   -

Planet indices:
   1 = Mercury           8 = Neptune
   2 = Venus             9 = Pluto
   3 = Earth            10 = Moon
   4 = Mars             11 = Sun
   5 = Jupiter          12 = Solar system barycenter
   6 = Saturn           13 = Earth-Moon barycenter
   7 = Uranus

Nutation, libration tests: Count of tested items
Item      | Angle  Rate
----------|------------
Nutation  |     0     0
Libration |    60    57


Ephemeris test dataset test results:
  # Passed: 1597
  # Failed: 0
  Status  : Passed

--------------------------------------------------------------------------------

JPL-DE405.little_endian.bin test summary: Passed

================================================================================
```

Test Ephemerides_2: DE4xx Comparison Simulation

Purpose:

The purposes of this test are to verify that the Ephemerides Model can be employed with the Trick simulation environment and correctly computes requested ephemerides for various solar system bodies.

Requirements:

By passing this test, the Ephemerides Model satisfies requirements Ephemerides_2 to Ephemerides_6.

Procedure:

Compare the ephemerides generated by the simulation with the results of the JPL online tool HORIZONS for various solar system bodies and various times. The HORIZONS tool can be found at URL *http://ssd.jpl.nasa.gov/?horizons*.

Results:

Each comparison of ephemerides computed by the simulation with ephemerides computed by the HORIZONS tool showed a match in position and velocity vectors to at least 15 digits. The results of a sample of the test cases used for comparison are shown in table 5.2. NOTE: For some solar system bodies, the HORIZONS online tool does not utilize the DE405 ephemerides. This seemed to be the usual case for bodies farther from the Sun than Mars. In cases where the DE405 was not used by the HORIZONS tool the comparison between the model and the tool results showed an acceptable (but not exact) match. The ephemeris used by the HORIZONS tool in each case is listed in the header of ephemeris generated by the tool.

Table 5.2: De4xx Simulation Results

Ephemerides computed by the Ephemerides Model test simulation that were compared with results of the JPL HORIZONS tool. In all cases, the model results matched the JPL results to at least 15 digits. The date in the table is the Julian Ephemeris Date (JED), which is the Julian Date expressed in the Terrestrial Time (TT) scale. This is the same as Coordinate Time (CT) used by the JPL HORIZONS tool.

Date (JED)	**Reference Body**	**Target Body**	**Position** (km)	**Velocity** (km/s)
2453147.5	Earth	Moon	-1.144514189946983E+04	-9.671675601865520E-01
			3.601512411881938E+05	-5.528033762838737E-02
			1.874270860935389E+05	3.693039024383601E-02
2453147.5	Earth	Sun	7.298448725554791E+07	-2.562749354399740E+01
			1.217548698456285E+08	1.327280438377108E+01
			5.278605001076507E+07	5.755037832421791E+00
2454000.5	Earth	Venus	-2.484673958478145E+08	-1.437902149252885E+01
			4.003218279094194E+07	-5.695930452876355E+01
			2.418466066464748E+07	-2.428170990877956E+01
2451234.0	Sun	Earth	-1.346561079177961E+08	-1.285186994778481E+01
			5.644644430198597E+07	-2.496341304044990E+01
			2.447321438897178E+07	-1.082367226693504E+01
2455013.5	Sun	Mars	1.987606576116480E+08	-7.414247633852530E+00
			6.823251638747539E+07	2.248902545222841E+01
			2.592782174112418E+07	1.051537808581921E+01

Test Ephemerides_3: Propagated Planet Simulation

Purpose:

The purposes of this test are to:

- Verify the ability to propagate a planet via JEOD integration,
- Verify the ability to switch the source of a planet's state from an ephemeris model to a propagated planet, and
- Serve as a testbed for the Simulation Engine Interface Model multiple integration group capability.

Requirements:

By passing this test, the Ephemerides Model satisfies requirements Ephemerides_2, Ephemerides_3, Ephemerides_4, and Ephemerides_7.

Procedure:

Compare the planetary states as generated by the simulation when run in ephemeris model versus those generate when run in propagated mode.

Results:

The comparisons are favorable but far from perfect. The propagated states degrades from those provided by the ephemeris model, with differences between propagated and computed growing as time progresses. The errors after 150 years of propagation are about the same for RK4 versus Gauss-Jackson integration. This suggests that the source of the error is using the ephemeris model as a point of departure. The DE ephemerides are optimized for position, not velocity. They were not optimized for taking some randomly chosen point as the basis for propagation.

5.3 Requirements Traceability

Table 5.3 summarizes the inspections and tests that demonstrate the satisfaction of the requirements levied on the model.

Table 5.3: Requirements Traceability

Requirement	**Traces to**
Ephemerides_1 Top-level Requirement	Insp. Ephemerides_1 Top-level Inspection
Ephemerides_2 Ephemeris Items	Insp. Ephemerides_2 Design Inspection Test Ephemerides_1 DE4xx Unit Test Test Ephemerides_2 DE4xx Comparision Test Ephemerides_3 Propagated Planet
Ephemerides_3 Ephemeris Interface	Insp. Ephemerides_2 Design Inspection Test Ephemerides_1 DE4xx Unit Test Test Ephemerides_2 DE4xx Comparision Test Ephemerides_3 Propagated Planet
Ephemerides_4 Ephemeris Manager	Insp. Ephemerides_2 Design Inspection Test Ephemerides_2 DE4xx Comparision Test Ephemerides_3 Propagated Planet
Ephemerides_5 Solar System	Insp. Ephemerides_2 Design Inspection Test Ephemerides_1 DE4xx Unit Test Test Ephemerides_2 DE4xx Comparision
Ephemerides_6 DE4xx Module	Insp. Ephemerides_2 Design Inspection Test Ephemerides_1 DE4xx Unit Test Test Ephemerides_2 DE4xx Comparision
Ephemerides_7 Propagated Planet	Insp. Ephemerides_2 Design Inspection Test Ephemerides_3 Propagated Planet

5.4 Metrics

Table 5.4 presents coarse metrics on the source files that comprise the model.

Table 5.4: Coarse Metrics

File Name	Number of Lines			
	Blank	**Comment**	**Code**	**Total**
Total	0	0	0	0

Table 5.5 presents the extended cyclomatic complexity (ECC) of the methods defined in the model.

Table 5.5: Cyclomatic Complexity

Method	File	Line	ECC
jeod::De4xxBase::__attribute__ ((unused)	de4xx_ephem/include/de4xx_base.hh	174	1
jeod::De4xxBase::number_jeod_items (int de_version_num __attribute__ ((unused)	de4xx_ephem/include/de4xx_base.hh	192	1
jeod::De4xxBase::number_trans_points (int de_version_num __attribute__ ((unused)	de4xx_ephem/include/de4xx_base.hh	202	1
jeod::De4xxBase::number_grav_models (int de_version_num __attribute__ ((unused)	de4xx_ephem/include/de4xx_base.hh	212	1
jeod::De4xxBase::number_physical_bodies (int de_version_num __attribute__ ((unused)	de4xx_ephem/include/de4xx_base.hh	222	1
jeod::De4xxEphemeris::set_model_number (int denum_in)	de4xx_ephem/include/de4xx_ephem.hh	250	1
jeod::De4xxEphemeris::get_model_number ()	de4xx_ephem/include/de4xx_ephem.hh	259	1
jeod::De4xxEphemeris::get_header_data ()	de4xx_ephem/include/de4xx_ephem.hh	268	1
jeod::De4xxFileSpec::get_model_number ()	de4xx_ephem/include/de4xx_file.hh	213	1
jeod::De4xxEphemItem::De4xxEphemItem (void)	de4xx_ephem/src/de4xx_ephem.cc	89	1
jeod::De4xxEphemItem::~De4xxEphemItem (void)	de4xx_ephem/src/de4xx_ephem.cc	106	1
jeod::De4xxEphemeris::De4xxEphemeris (void)	de4xx_ephem/src/de4xx_ephem.cc	117	3
jeod::De4xxEphemeris::~De4xxEphemeris (void)	de4xx_ephem/src/de4xx_ephem.cc	216	1
jeod::De4xxEphemeris::shutdown (void)	de4xx_ephem/src/de4xx_ephem.cc	230	2
jeod::De4xxEphemeris::activate (void)	de4xx_ephem/src/de4xx_ephem.cc	246	2

Continued on next page

Table 5.5: Cyclomatic Complexity (continued)

Method	File	Line	ECC
jeod::De4xxEphemeris:: deactivate (void)	de4xx_ephem/src/de4xx_ephem.cc	265	1
jeod::De4xxEphemeris:: timestamp (void)	de4xx_ephem/src/de4xx_ephem.cc	278	1
jeod::De4xxEphemeris::get_name (void)	de4xx_ephem/src/de4xx_ephem.cc	291	1
jeod::De4xxEphemeris:: initialize_model (const TimeManager & time_manager, EphemeridesManager & ephem_manager, const std::string & time_type)	de4xx_ephem/src/de4xx_ephem.cc	304	2
jeod::De4xxEphemeris:: initialize_time (const TimeManager & time_manager, const std::string & time_type)	de4xx_ephem/src/de4xx_ephem.cc	339	5
jeod::De4xxEphemeris:: initialize_file (void)	de4xx_ephem/src/de4xx_ephem.cc	380	1
jeod::De4xxEphemeris:: initialize_items (EphemeridesManager & ephem_manager)	de4xx_ephem/src/de4xx_ephem.cc	414	13
jeod::De4xxEphemeris:: ephem_initialize (EphemeridesManager & ephem_manager JEOD_UNUSED)	de4xx_ephem/src/de4xx_ephem.cc	486	5
jeod::De4xxEphemeris:: activate_nodes (void)	de4xx_ephem/src/de4xx_ephem.cc	529	8
jeod::De4xxEphemeris:: activate_em_nodes (unsigned int tot_active)	de4xx_ephem/src/de4xx_ephem.cc	572	12
jeod::De4xxEphemeris:: determine_root_node (void)	de4xx_ephem/src/de4xx_ephem.cc	642	11
jeod::De4xxEphemeris:: ephem_activate (EphemeridesManager & ephem_manager JEOD_UNUSED)	de4xx_ephem/src/de4xx_ephem.cc	707	6

Continued on next page

Table 5.5: Cyclomatic Complexity (continued)

Method	File	Line	ECC
jeod::De4xxEphemeris::ephem_build_tree (EphemeridesManager & ephem_manager)	de4xx_ephem/src/de4xx_ephem.cc	766	12
jeod::De4xxEphemeris::ephem_update (void)	de4xx_ephem/src/de4xx_ephem.cc	827	14
jeod::De4xxEphemeris::time_is_in_range (void)	de4xx_ephem/src/de4xx_ephem.cc	912	1
jeod::De4xxEphemeris::propagate_lunar_rnp (void)	de4xx_ephem/src/de4xx_ephem.cc	929	3
jeod::De4xxEphemeris::initialize_model (const TimeManager & time_manager, DynManager & dyn_manager, const std::string & time_type)	de4xx_ephem/src/de4xx_ephem_dynmanager.cc	43	1
jeod::De4xxFileSpec::De4xxFileSpec (void)	de4xx_ephem/src/de4xx_file.cc	84	1
jeod::De4xxFileSpec::set_model_number (int denum_in)	de4xx_ephem/src/de4xx_file.cc	97	1
jeod::De4xxFileIO::De4xxFileIO (void)	de4xx_ephem/src/de4xx_file.cc	105	1
jeod::De4xxFileHeader::De4xxFileHeader (void)	de4xx_ephem/src/de4xx_file.cc	127	2
jeod::De4xxFileHeader::~De4xxFileHeader (void)	de4xx_ephem/src/de4xx_file.cc	148	1
jeod::De4xxFileItem::De4xxFileItem (void)	de4xx_ephem/src/de4xx_file.cc	158	2
jeod::De4xxFileRefTime::De4xxFileRefTime (void)	de4xx_ephem/src/de4xx_file.cc	182	1
jeod::De4xxFileCoef::De4xxFileCoef (void)	de4xx_ephem/src/de4xx_file.cc	198	1
jeod::De4xxFileRestart::De4xxFileRestart (De4xxFile & in)	de4xx_ephem/src/de4xx_file.cc	215	1
jeod::De4xxFileRestart::~De4xxFileRestart (void)	de4xx_ephem/src/de4xx_file.cc	228	1

Continued on next page

Table 5.5: Cyclomatic Complexity (continued)

Method	File	Line	ECC
jeod::De4xxFileRestart:: simple_restore (void)	de4xx_ephem/src/de4xx_ file.cc	238	1
jeod::De4xxFile::De4xxFile (void)	de4xx_ephem/src/de4xx_ file.cc	249	1
jeod::De4xxFile::~De4xxFile (void)	de4xx_ephem/src/de4xx_ file.cc	289	1
jeod::De4xxFile::shutdown (void)	de4xx_ephem/src/de4xx_ file.cc	302	1
jeod::De4xxFile::open (void)	de4xx_ephem/src/de4xx_ file.cc	313	6
jeod::De4xxFile::reopen (void)	de4xx_ephem/src/de4xx_ file.cc	402	2
jeod::De4xxFile::close (void)	de4xx_ephem/src/de4xx_ file.cc	425	3
jeod::De4xxFile::time_is_in_ range (double time)	de4xx_ephem/src/de4xx_ file.cc	466	2
jeod::std::process_mem_usage (double& vm_usage, double& resident_set)	de4xx_ephem/src/de4xx_ file.cc	495	1
jeod::De4xxFile::capture_ mem_stats ()	de4xx_ephem/src/de4xx_ file.cc	532	2
jeod::De4xxFile::pre_initialize (void)	de4xx_ephem/src/de4xx_file_ init.cc	67	11
jeod::De4xxFile::initialize (double epoch_time, double del_day, double time_offset, double init_time)	de4xx_ephem/src/de4xx_file_ init.cc	170	4
jeod::l1_point (double b1b2_ mass_ratio)	de4xx_ephem/src/de4xx_file_ init.cc	270	3
jeod::De4xxFile::update (double time)	de4xx_ephem/src/de4xx_file_ update.cc	50	16
jeod::De4xxFile::interpolate (double time, double fblk)	de4xx_ephem/src/de4xx_file_ update.cc	254	8
jeod::EphemerisInterface::~ EphemerisInterface (void)	ephem_interface/include/ ephem_interface.hh	159	1
jeod::SinglePointEphemeris:: timestamp (void)	ephem_interface/include/ simple_ephemerides.hh	301	1

Continued on next page

Table 5.5: Cyclomatic Complexity (continued)

Method	File	Line	ECC
jeod::SinglePointEphemeris:: get_name (void)	ephem_interface/include/ simple_ephemerides.hh	314	1
jeod::SinglePointEphemeris:: ephem_update (void)	ephem_interface/include/ simple_ephemerides.hh	327	1
jeod::EphemerisRefFrame:: EphemerisRefFrame (void)	ephem_interface/src/ephem_ ref_frame.cc	47	1
jeod::EphemerisRefFrame::~ EphemerisRefFrame (void)	ephem_interface/src/ephem_ ref_frame.cc	57	1
jeod::EphemerisRefFrame:: set_ephem_manager (Base EphemeridesManager * manager)	ephem_interface/src/ephem_ ref_frame.cc	65	1
jeod::EphemerisRefFrame:: set_active_status (bool new_ status)	ephem_interface/src/ephem_ ref_frame.cc	77	2
jeod::SinglePointEphemeris:: SinglePointEphemeris (void)	ephem_interface/src/simple_ ephemerides.cc	59	1
jeod::SinglePointEphemeris::~ SinglePointEphemeris (void)	ephem_interface/src/simple_ ephemerides.cc	74	3
jeod::SinglePointEphemeris:: activate (void)	ephem_interface/src/simple_ ephemerides.cc	86	2
jeod::SinglePointEphemeris:: deactivate (void)	ephem_interface/src/simple_ ephemerides.cc	101	1
jeod::SinglePointEphemeris:: set_name (const char * new_name)	ephem_interface/src/simple_ ephemerides.cc	113	2
jeod::EmptySpaceEphemeris:: EmptySpaceEphemeris (void)	ephem_interface/src/simple_ ephemerides.cc	148	1
jeod::EmptySpaceEphemeris:: ~EmptySpaceEphemeris (void)	ephem_interface/src/simple_ ephemerides.cc	160	1
jeod::EmptySpaceEphemeris:: set_name (const char * new_name)	ephem_interface/src/simple_ ephemerides.cc	170	3

Continued on next page

Table 5.5: Cyclomatic Complexity (continued)

Method	File	Line	ECC
jeod::EmptySpaceEphemeris:: initialize_model (EphemeridesManager & ephem_manager)	ephem_interface/src/simple_ ephemerides.cc	199	2
jeod::EmptySpaceEphemeris:: ephem_initialize (EphemeridesManager & ephem_manager JEOD_UN USED)	ephem_interface/src/simple_ ephemerides.cc	219	2
jeod::EmptySpaceEphemeris:: ephem_activate (EphemeridesManager & ephem_manager JEOD_UN USED)	ephem_interface/src/simple_ ephemerides.cc	242	1
jeod::EmptySpaceEphemeris:: ephem_build_tree (EphemeridesManager & ephem_manager)	ephem_interface/src/simple_ ephemerides.cc	254	2
jeod::SinglePlanetEphemeris:: SinglePlanetEphemeris (void)	ephem_interface/src/simple_ ephemerides.cc	282	1
jeod::SinglePlanetEphemeris:: ~SinglePlanetEphemeris (void)	ephem_interface/src/simple_ ephemerides.cc	294	1
jeod::SinglePlanetEphemeris:: set_name (const char * new_name)	ephem_interface/src/simple_ ephemerides.cc	304	3
jeod::SinglePlanetEphemeris:: initialize_model (EphemeridesManager & ephem_manager)	ephem_interface/src/simple_ ephemerides.cc	331	2
jeod::SinglePlanetEphemeris:: ephem_initialize (EphemeridesManager & ephem_manager)	ephem_interface/src/simple_ ephemerides.cc	350	6
jeod::SinglePlanetEphemeris:: ephem_activate (EphemeridesManager & ephem_manager JEOD_UN USED)	ephem_interface/src/simple_ ephemerides.cc	395	1

Continued on next page

Table 5.5: Cyclomatic Complexity (continued)

Method	File	Line	ECC
jeod::SinglePlanetEphemeris:: ephem_build_tree (EphemeridesManager & ephem_manager)	ephem_interface/src/simple_ ephemerides.cc	407	2
jeod::EphemerisItem::get_ name (void)	ephem_item/include/ephem_ item_inline.hh	74	1
jeod::EphemerisItem::set_ timestamp (double time)	ephem_item/include/ephem_ item_inline.hh	87	1
jeod::EphemerisItem:: timestamp (void)	ephem_item/include/ephem_ item_inline.hh	100	1
jeod::EphemerisItem::is_ enabled (void)	ephem_item/include/ephem_ item_inline.hh	113	1
jeod::EphemerisItem:: deactivate (void)	ephem_item/include/ephem_ item_inline.hh	126	1
jeod::EphemerisItem::is_active (void)	ephem_item/include/ephem_ item_inline.hh	138	1
jeod::EphemerisItem::set_ owner (EphemerisInterface * new_owner)	ephem_item/include/ephem_ item_inline.hh	151	1
jeod::EphemerisItem::get_ owner (void)	ephem_item/include/ephem_ item_inline.hh	163	1
jeod::EphemerisItem::set_ manager (BaseEphemerides Manager * new_manager)	ephem_item/include/ephem_ item_inline.hh	176	1
jeod::EphemerisItem::get_ manager (void)	ephem_item/include/ephem_ item_inline.hh	189	1
jeod::EphemerisItem::set_head (EphemerisItem * head_ item)	ephem_item/include/ephem_ item_inline.hh	202	1
jeod::EphemerisItem::get_ head (void)	ephem_item/include/ephem_ item_inline.hh	215	1
jeod::EphemerisItem::set_next (EphemerisItem * next_ item)	ephem_item/include/ephem_ item_inline.hh	228	1
jeod::EphemerisItem::get_next (void)	ephem_item/include/ephem_ item_inline.hh	241	1
jeod::EphemerisItem::get_ target_frame (void)	ephem_item/include/ephem_ item_inline.hh	254	1

Continued on next page

Table 5.5: Cyclomatic Complexity (continued)

Method	File	Line	ECC
jeod::EphemerisItem::get_enabled_item (void)	ephem_item/include/ephem_item_inline.hh	267	3
jeod::EphemerisItem::EphemerisItem (void)	ephem_item/src/ephem_item.cc	59	1
jeod::EphemerisItem::~EphemerisItem (void)	ephem_item/src/ephem_item.cc	80	3
jeod::EphemerisItem::validate_name (const char * file, unsigned int line, const char * new_value, const char * old_value, const char * variable_name)	ephem_item/src/ephem_item.cc	93	4
jeod::EphemerisItem::set_name (const char * pname, const char * fname)	ephem_item/src/ephem_item.cc	126	1
jeod::EphemerisItem::set_name (const char * new_name)	ephem_item/src/ephem_item.cc	146	2
jeod::EphemerisItem::set_name_internal (char * new_name)	ephem_item/src/ephem_item.cc	178	3
jeod::EphemerisItem::set_target_frame (EphemerisRefFrame & frame)	ephem_item/src/ephem_item.cc	197	11
jeod::EphemerisItem::enable (void)	ephem_item/src/ephem_item.cc	275	5
jeod::EphemerisItem::disable (void)	ephem_item/src/ephem_item.cc	312	4
jeod::EphemerisItem::activate (void)	ephem_item/src/ephem_item.cc	341	2
jeod::EphemerisItem::is_activatable (void)	ephem_item/src/ephem_item.cc	360	5
jeod::EphemerisOrientation::EphemerisOrientation (void)	ephem_item/src/ephem_orient.cc	56	1
jeod::EphemerisOrientation::~EphemerisOrientation (void)	ephem_item/src/ephem_orient.cc	68	1

Continued on next page

Table 5.5: Cyclomatic Complexity (continued)

Method	File	Line	ECC
jeod::EphemerisItem:: updates_what (void)	ephem_item/src/ephem_ orient.cc	78	1
jeod::EphemerisOrientation:: enable (void)	ephem_item/src/ephem_ orient.cc	92	4
jeod::EphemerisOrientation:: note_frame_status_change (RefFrame * frame)	ephem_item/src/ephem_ orient.cc	117	5
jeod::EphemerisOrientation:: default_suffix (void)	ephem_item/src/ephem_ orient.cc	161	1
jeod::EphemerisOrientation:: disconnect_from_tree (void)	ephem_item/src/ephem_ orient.cc	174	1
jeod::EphemerisZXZ Orientation::EphemerisZXZ Orientation (void)	ephem_item/src/ephem_ orient_zxz.cc	77	1
jeod::EphemerisZXZ Orientation::~EphemerisZX ZOrientation (void)	ephem_item/src/ephem_ orient_zxz.cc	90	1
jeod::EphemerisZXZ Orientation::get_euler_ angles (void)	ephem_item/src/ephem_ orient_zxz.cc	100	1
jeod::EphemerisZXZ Orientation::get_euler_ angles (double * angles)	ephem_item/src/ephem_ orient_zxz.cc	113	1
jeod::EphemerisZXZ Orientation::get_euler_rates (void)	ephem_item/src/ephem_ orient_zxz.cc	130	1
jeod::EphemerisZXZ Orientation::get_euler_rates (double * rates)	ephem_item/src/ephem_ orient_zxz.cc	143	1
jeod::EphemerisZXZ Orientation::propagate (double to_time)	ephem_item/src/ephem_ orient_zxz.cc	158	4
jeod::EphemerisZXZ Orientation::update (const double * angles, const double * derivs, double time)	ephem_item/src/ephem_ orient_zxz.cc	222	1
jeod::EphemerisPoint:: EphemerisPoint (void)	ephem_item/src/ephem_ point.cc	56	1

Continued on next page

Table 5.5: Cyclomatic Complexity (continued)

Method	File	Line	ECC
jeod::EphemerisPoint::~ EphemerisPoint (void)	ephem_item/src/ephem_ point.cc	68	1
jeod::EphemerisPoint::note_ frame_status_change (Ref Frame * frame)	ephem_item/src/ephem_ point.cc	78	3
jeod::EphemerisPoint:: default_suffix (void)	ephem_item/src/ephem_ point.cc	105	1
jeod::EphemerisPoint:: disconnect_from_tree (void)	ephem_item/src/ephem_ point.cc	118	3
jeod::EphemerisPoint:: initialize_state (void)	ephem_item/src/ephem_ point.cc	133	1
jeod::EphemerisPoint::update (const double * position, const double * velocity, double time)	ephem_item/src/ephem_ point.cc	147	1
jeod::EphemerisPoint:: update_scaled (const double * position, const double * velocity, double scale, double time)	ephem_item/src/ephem_ point.cc	168	1
jeod::EphemerisItem:: updates_what (void)	ephem_item/src/ephem_ point.cc	191	1
jeod::EphemeridesManager:: ref_frame_tree_needs_rebuild ()	ephem_manager/include/ ephem_manager.hh	112	1
jeod::EphemeridesManager:: EphemeridesManager (void)	ephem_manager/src/ephem_ manager.cc	56	1
jeod::EphemeridesManager::~ EphemeridesManager (void)	ephem_manager/src/ephem_ manager.cc	80	1
jeod::EphemeridesManager:: ephem_note_tree_status_ change (void)	ephem_manager/src/ephem_ manager.cc	98	1
jeod::EphemeridesManager:: add_planet (BasePlanet & planet)	ephem_manager/src/ephem_ manager.cc	115	4

Continued on next page

Table 5.5: Cyclomatic Complexity (continued)

Method	File	Line	ECC
jeod::EphemeridesManager:: find_base_planet (const char * name)	ephem_manager/src/ephem_ manager.cc	153	3
jeod::EphemeridesManager:: get_num_planets (void)	ephem_manager/src/ephem_ manager.cc	181	1
jeod::EphemeridesManager:: disable_add_ephemeris (void)	ephem_manager/src/ephem_ manager.cc	199	1
jeod::EphemeridesManager:: add_ephemeris (Ephemeris Interface & ephem_if)	ephem_manager/src/ephem_ manager.cc	221	1
jeod::EphemeridesManager:: clear_added_ephemerides (void)	ephem_manager/src/ephem_ manager.cc	241	4
jeod::EphemeridesManager:: add_ephem_item (EphemerisItem & ephem_ item)	ephem_manager/src/ephem_ manager.cc	276	11
jeod::EphemeridesManager:: find_ephem_item (const char * name)	ephem_manager/src/ephem_ manager.cc	382	3
jeod::EphemeridesManager:: find_ephem_angle (const char * name)	ephem_manager/src/ephem_ manager.cc	407	3
jeod::EphemeridesManager:: find_ephem_point (const char * name)	ephem_manager/src/ephem_ manager.cc	432	3
jeod::EphemeridesManager:: add_integ_frame (Ephemeris RefFrame & ref_frame)	ephem_manager/src/ephem_ manager.cc	462	1
jeod::EphemeridesManager:: find_integ_frame (const char * name)	ephem_manager/src/ephem_ manager.cc	480	3
jeod::std::get_integ_frames (void)	ephem_manager/src/ephem_ manager.cc	506	1
jeod::EphemeridesManager:: is_integ_frame (const Ref Frame & ref_frame)	ephem_manager/src/ephem_ manager.cc	519	3

Continued on next page

Table 5.5: Cyclomatic Complexity (continued)

Method	File	Line	ECC
jeod::EphemeridesManager::find_integ_frame_index (const EphemerisRefFrame & ref_frame)	ephem_manager/src/ephem_manager.cc	545	4
jeod::EphemeridesManager::add_ref_frame (RefFrame & ref_frame)	ephem_manager/src/ephem_manager.cc	589	1
jeod::EphemeridesManager::set_target_frame (RefFrame & ref_frame)	ephem_manager/src/ephem_manager.cc	603	4
jeod::EphemeridesManager::initialize_ephemerides (void)	ephem_manager/src/ephem_manager.cc	656	2
jeod::EphemeridesManager::update_ephemerides (void)	ephem_manager/src/ephem_manager.cc	677	3
jeod::EphemeridesManager::activate_ephemerides (void)	ephem_manager/src/ephem_manager.cc	702	5
jeod::EphemeridesManager::find_planet (const char * name)	ephem_manager/src/find_planet.cc	45	3
jeod::EphemeridesManager::add_planet (Planet & planet)	ephem_manager/src/find_planet.cc	82	1
jeod::PropagatedEphemerisPlanet::~PropagatedEphemerisPlanet (void)	propagated_planet/src/propagated_planet.cc	78	1
jeod::PropagatedEphemerisPlanet::update (double dyn_time)	propagated_planet/src/propagated_planet.cc	88	2
jeod::PropagatedEphemerisOrientation::PropagatedEphemerisOrientation (DynBody & dyn_body, BodyRefFrame & frame)	propagated_planet/src/propagated_planet.cc	122	1
jeod::PropagatedEphemerisOrientation::~PropagatedEphemerisOrientation (void)	propagated_planet/src/propagated_planet.cc	139	1

Continued on next page

Table 5.5: Cyclomatic Complexity (continued)

Method	File	Line	ECC
jeod::PropagatedEphemeris Orientation::update (double dyn_time)	propagated_planet/src/ propagated_planet.cc	149	2
jeod::PropagatedEphemeris Planet::Propagated EphemerisPlanet (Dyn Body & dyn_body, BodyRef Frame & frame)	propagated_planet/src/ propagated_planet.cc	183	1
jeod::PropagatedPlanet:: PropagatedPlanet (void)	propagated_planet/src/ propagated_planet.cc	200	1
jeod::PropagatedPlanet::~ PropagatedPlanet (void)	propagated_planet/src/ propagated_planet.cc	227	1
jeod::PropagatedPlanet:: shutdown (void)	propagated_planet/src/ propagated_planet.cc	237	2
jeod::PropagatedPlanet:: activate (void)	propagated_planet/src/ propagated_planet.cc	252	2
jeod::PropagatedPlanet:: deactivate (void)	propagated_planet/src/ propagated_planet.cc	271	1
jeod::PropagatedPlanet:: timestamp (void)	propagated_planet/src/ propagated_planet.cc	284	1
jeod::PropagatedPlanet::get_ name (void)	propagated_planet/src/ propagated_planet.cc	297	1
jeod::PropagatedPlanet:: initialize_model (const TimeManager & time_ manager, DynManager & dyn_manager_ref)	propagated_planet/src/ propagated_planet.cc	310	6
jeod::PropagatedPlanet:: ephem_initialize (EphemeridesManager & ephem_manager)	propagated_planet/src/ propagated_planet.cc	391	5
jeod::PropagatedPlanet::set_ commanded_mode (PropagatedPlanet::Mode new_mode)	propagated_planet/src/ propagated_planet.cc	448	1
jeod::PropagatedPlanet::set_ mode (void)	propagated_planet/src/ propagated_planet.cc	460	13

Continued on next page

Table 5.5: Cyclomatic Complexity (continued)

Method	File	Line	ECC
jeod::PropagatedPlanet:: ephem_activate (EphemeridesManager & ephem_manager JEOD_UNUSED)	propagated_planet/src/ propagated_planet.cc	507	3
jeod::PropagatedPlanet:: ephem_build_tree (EphemeridesManager & ephem_manager)	propagated_planet/src/ propagated_planet.cc	529	3
jeod::PropagatedPlanet:: ephem_update (void)	propagated_planet/src/ propagated_planet.cc	551	5

Bibliography

[1] Hammen, D. Container Model. Technical Report JSC-61777-utils/container, NASA, Johnson Space Center, Houston, Texas, July 2023.

[2] Hammen, D. Dynamic Body Model. Technical Report JSC-61777-dynamics/dyn_body, NASA, Johnson Space Center, Houston, Texas, July 2023.

[3] Hammen, D. Memory Management Model. Technical Report JSC-61777-utils/memory, NASA, Johnson Space Center, Houston, Texas, July 2023.

[4] Hammen, D. Quaternion Model. Technical Report JSC-61777-utils/quaternion, NASA, Johnson Space Center, Houston, Texas, July 2023.

[5] Hammen, D. Body Action Model. Technical Report JSC-61777-dynamics/body_action, NASA, Johnson Space Center, Houston, Texas, July 2023.

[6] Hammen, D. Dynamics Manager Model. Technical Report JSC-61777-dynamics/dyn_manager, NASA, Johnson Space Center, Houston, Texas, July 2023.

[7] Heafner, P.J. *Fundamental Ephemeris Computations For use with JPL data.* Willmann-Bell, Richmond, Virginia, 1999.

[8] Jackson, A., Thebeau, C. JSC Engineering Orbital Dynamics. Technical Report JSC-61777-docs, NASA, Johnson Space Center, Houston, Texas, July 2023.

[9] Montenbruck, O. and Gill, E. *Satellite Orbits.* Springer, Netherlands, 2005.

[10] Morris, J. Planet Model. Technical Report JSC-61777-environment/planet, NASA, Johnson Space Center, Houston, Texas, July 2023.

[11] NASA. NASA Software Engineering Requirements. Technical Report NPR-7150.2, NASA, NASA Headquarters, Washington, D.C., September 2004.

[12] National Aeronautics and Space Administration, Johnson Space Center, Software, Robotics & Simulation Division, Simulation and Graphics Branch, 2101 NASA Parkway, Houston, Texas, 77058. *JEOD Ephemerides Model Reference Manual*, July 2023.

[13] Seidelmann, P.K. *Explanatory Supplement to the Astronomical Almanac.* University Science Books, Sausalito, California, 2006.

[14] Shelton, R. Message Handler Model. Technical Report JSC-61777-utils/message, NASA, Johnson Space Center, Houston, Texas, July 2023.

[15] Shelton, R. Named Item Model. Technical Report JSC-61777-utils/named_item, NASA, Johnson Space Center, Houston, Texas, July 2023.

[16] Shelton, R. Simulation Engine Interface Model. Technical Report JSC-61777-utils/sim_interface, NASA, Johnson Space Center, Houston, Texas, July 2023.

[17] Spencer, A. Reference Frame Model. Technical Report JSC-61777-utils/ref_frames, NASA, Johnson Space Center, Houston, Texas, July 2023.

[18] Spencer, A. Rotation, Nutation, and Precession Model. Technical Report JSC-61777-environment/RNP, NASA, Johnson Space Center, Houston, Texas, July 2023.

[19] Thompson, B. Mathematical Functions Model. Technical Report JSC-61777-utils/math, NASA, Johnson Space Center, Houston, Texas, July 2023.

[20] Thompson, B. and Morris., J. Gravity Model. Technical Report JSC-61777-environment/gravity, NASA, Johnson Space Center, Houston, Texas, July 2023.

[21] Turner, G. Time Model. Technical Report JSC-61777-environment/time, NASA, Johnson Space Center, Houston, Texas, July 2023.