**Feature Spec and Test Procedures**

[Spec User’s Guide](#h.4dnhv6bl2cp4)

[Explore](#h.uqmlrxburf6q)

[Known Bugs](#h.fy9l5djhxus3)

[Failing Tests](#h.g7584dhfd4gv)

[Other Findings](#h.255a5j5c5vh3)

[Requirements](#h.hg7ht1kc8jq2)

[Interface/Functional Spec](#h.dkhm82tb6lds)

[Overview](#h.tj492u8tvx1i)

[Description](#h.y647aplqfkxw)

[Fields](#h.s4uk9hw5pbnt)

[GUI](#h.7h4nj4qlhx86)

[Remarks](#h.7vnqcfgs2slc)

[Behavior When Changing the Coordinate System](#h.vqjm6mo0o4yv)

[Examples](#h.32fsayctuf6m)

[Test Procedures](#h.afp33sm1hbfq)

[Assumptions](#h.evplg86fl6xx)

[Existing Tests](#h.c0clhzqg8we)

[Recommended Additional Tests](#h.3g0z56b7d67s)

# Spec User’s Guide

This section is for spec writers to explain how we write specs. Assume for this release that this document will be used by software testers to test the software, and by technical writers to write end-user documentation. When possible write the specifications so that content can be used with little or no modification in user-documentation and have therefore written the document as if the end-user is the reader. The organization of the functional spec section is modelled after MATLAB’s help. See existing GMAT example for the For command, Spacecraft Epoch, and Propagator for good examples of finished specifications.

**General spec writing guidelines**

* Follow all rules in the [GMAT Style Guide](https://docs.google.com/document/d/1dgoXJNL5XqyGQQXd_RGCxDAmqyCOQDiiPZo9p20Fslk/edit?authkey=CLDDzqoI&hl=en_US&authkey=CLDDzqoI#heading=h.pt9u8o9vvvcj).
* Use clear, simple, active voice
* Assume tech-writer/end-user are the primary audience for all material not contained in message boxes (see below). i.e. (write with end-user quality material in main sections, use whatever is necessary to convey the point in message boxes).
* Read existing specs for example of style before writing
	+ See Force-Model for a rigorous Resource Example
	+ See Target for a rigorous Command example
* If you are a feature lead drafting a spec chapter, emphasize completeness of content. The style does not have to be perfect, it will be reworked by the doc owner.
* If you are the PDL, focus on clarity and consistency of the text.
* Here are special annotation styles

|  |
| --- |
| Open Issue: Use a red table box like this to point out issues in behavior or functionality that are not resolved. THESE ARE NOT BUGS!! Must be resolved before spec is finished. |

|  |
| --- |
| Caution: Use a red table box to include potentially confusing or very important information for users. Examples include when a feature only works in the script and not in the GUI, or if a feature can potentially be misused if the user does not understand something critical. |

|  |
| --- |
| For GUI Tester: Put information intended for a specific document user in a separate grey table box. For example, if a feature may require a unique GUI test type, let the GUI tester know by including the information in a box like this.  |

Put script snippets in a grey table box and use monospace font.

|  |
| --- |
| BeginFiniteBurn aFiniteBurn(aSat)BeginFiniteBurn aFiniteBurn(aSat)BeginFiniteBurn aFiniteBurn(aSat) |

# Explore

This section is to document an initial list all known issues with this feature. Put findings here during feature exploration. This is a temporary holding place, and all of these items MUST be moved to JIRA early in the feature finalization or they will not be tracked or scheduled to be fixed!

## Known Bugs

Known bugs [committed in JIRA](http://li64-187.members.linode.com:8080/secure/Dashboard.jspa) against this feature.

## Failing Tests

Tests failing in the nightly regression reports related to this feature.

## Other Findings

Issues you find when performing exploratory testing.

# Requirements

Move requirements from formal SRS into a table below and delete this sentence.

These are working requirements. They are included here for review and convenience purposes. After review, requirements are maintained in the formal SRS located at SourceForge in /trunk/doc/SystemDocs/Requirements. Send final requirements to CCB lead (SPH).

|  |  |
| --- | --- |
| **ID** | **Requirements** |
| FRR-xyz | The requirement. |
| FRR-xyz. | The requirement |

# Interface/Functional Spec

## Overview

A one-sentence description of the feature

## Description

Describe the object at a high level to give the user an intuitive understanding of what the object does. If there are complex interactions discuss briefly (See ForceModel for example) and point to Remarks sections for further clarification.

*See Also*: Put list of resources and commands that interact with the object here.

## Fields

See the [User Interface Spec](https://docs.google.com/spreadsheet/ccc?key=0Am0YJMLTVzxedGxmYmtnVHVvbG45eWd5aWFkbTY2a1E&hl=en_US#gid=0) spreadsheet for reference information for fields. This section is usually empty other than the hyperlink.

## GUI

Include GUI screenshots for each unique GUI configuration. Describe behavior of GUI for interactive fields when changes in one field can affect the layout or functionality of the GUI.

The Field Spec referenced in the previous section describes fields in gory detail each GUI item so this section is usually sparse. For example, if there are only text boxes on GUI dialog box, there is nothing to say here that is not already said in the field spec and this section only contains a screenshot and a few statements clarifying there is no unique behavior or complex coupling or GUI modes. However, if you can change a combo box and the GUI configuration changes, that information must be described here.

## Remarks

Describe gory details of the resource.

* Complex field interactions
* Items relevant to user but not described in the Fields section above.
* Group relevant information into subsections and begin the heading with “Behavior When”

Here are examples from existing specs:

**Best Practices for Using Numerical Integrators**

We recommend that you study the performance and accuracy analysis documented later in this section to select a numerical integrator for your application. You may need to perform further analysis and comparisons for your application. The comparison data below suggest that the PrinceDormand78 integrator is the best all purpose integrator in GMAT. When in doubt, use the PrinceDormance78 integrator, and set MinStep to zero so that the integrator’s adaptive step algorithm controls the minimum integration step size.

### Behavior When Changing the Coordinate System

When you select a Coordinate System for a spacecraft, you specify the origin and axes set with respect to which the state is defined. When you change the coordinate system via the GUI, the GUI performs the conversion from the initial coordinate system to the new system. Similarly, when you specify a state component via the script, that state component is set in the coordinate system defined in the CoordinateSystem field. For example, the following lines would result in the X-component of the Cartesian state of MySat to be 1000, in the EarthFixed system.

## Examples

Describe the example with a short sentence two and include minimal script for example below:

|  |
| --- |
| Create Spacecraft aSat;Create ForceModel aForceModel;Create Propagator aProp;aProp.FM = aForceModel;aProp.Type = PrinceDormand78;aProp.InitialStepSize = 60;aProp.Accuracy = 1e-011;aProp.MinStep = 0;aProp.MaxStep = 86400;aProp.MaxStepAttempts = 50;aProp.StopIfAccuracyIsViolated = true;BeginMissionSequencePropagate aProp(aSat) {aSat.ElapsedDays = .2}; |

# Test Procedures

## Assumptions

If you are making assumptions about how tests will be performed or that other test areas will cover some of this functionality describe that here.

## Existing Tests

Describe existing test types using a row for each class of test.

|  |  |  |
| --- | --- | --- |
| Priority | Status | Summary |
|  |  |  |
|  |  |  |
|  |  |  |

## Recommended Additional Tests

Nominal Tests

|  |  |  |
| --- | --- | --- |
| Priority | Status | Summary |
|  |  | [for Resources only] Cloning the resource in the Mission Sequence, using all non-default values in all field configurations |
|  |  |  |
|  |  |  |

Edge/Corner/Stres

|  |  |  |
| --- | --- | --- |
| Priority | Status | Summary |
|  |  |  |
|  |  |  |
|  |  |  |

Unique Validation

|  |  |  |
| --- | --- | --- |
| Priority | Status | Summary |
|  |  |  |
|  |  |  |
|  |  |  |

Unique Mode Tests

|  |  |  |
| --- | --- | --- |
| Priority | Status | Summary |
|  |  |  |
|  |  |  |
|  |  |  |

Unique GUI Tests

These are tests that are unique to the GUI interface for this feature that are not covered by the standard GUI test template and procedures.

|  |  |  |
| --- | --- | --- |
| Priority | Status | Summary |
|  |  |  |
|  |  |  |
|  |  |  |