The GMAT C-Interface Architecture and API

Darrel J. Conway
Thinking Systems, Inc.

January 11, 2012

Abstract

This document defines the GMAT C-Interface architecture (in the first major section) and API (in the second).

1 Introduction

2 Architecture

For now, this section is a catch-all for architectural thoughts. Very stream of consciousness and other high woo-woo catch phrases apply.

- One approach to this task is to rework the design of the engine to allow for an interpreted version of GMAT. This approach affects the Moderator and Configuration Manager most directly, the Sandbox a bit less, and also has implications for the Publisher that need to be thought about. To do this:
  - GMAT must be modularized. Modularization allows for different flavors of engine components based on the execution environment. If these components reside in separate libraries, the impact on the code supplying the simulation data is minimized.
- The following development platforms/languages need to be supported to claim success:
  - MATLAB
  - C? *(That seems to be implied by the project name...)*
  - Python

3 The C-Interface API

3.1 Accessing GMAT Classes

General access to GMAT functionality is performed using a 3 step process:

1. Create an object providing the required functionality
2. Access the functionality by calling methods on the new object
3. Discard the object

3.2 Converter APIs

GMAT's Converters are accessed using the Convert() method of a converter class. The Convert method has the signature

```
bool Convert(Integer inType, Real inData[], Integer outType, Real outData[]);
```

This method returns true if the conversion succeeded, and false if it failed.

3.3 Force Model APIs

3.4 Propagation APIs