MARTe2-core - Design Improvements #191

Objects

04.08.2015 11:10 - André Neto

Status:	Closed	**Spent time:**	0.00 hour
Priority:	Normal		
Assignee:			
Category:			

Description

Main definitions

Object

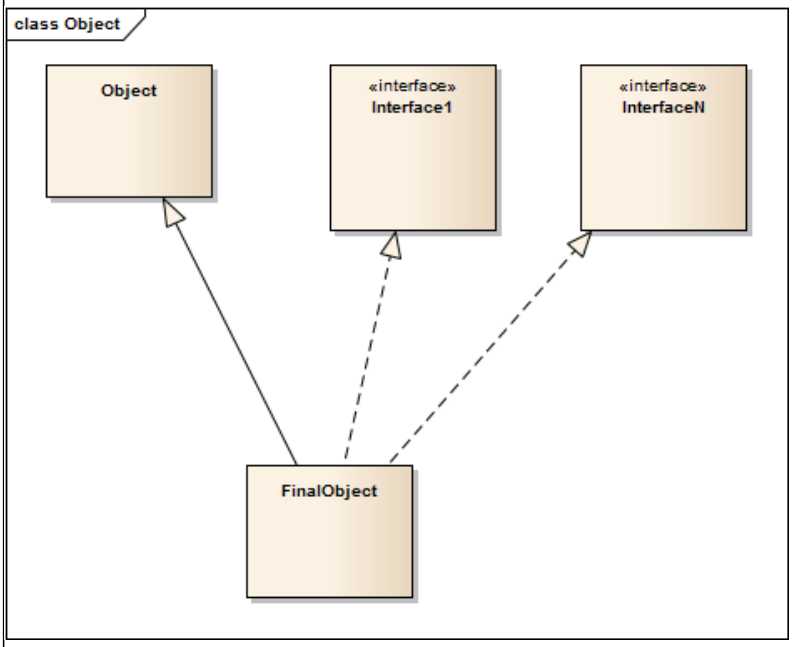
The MARTe Object is a class which offers the following functionality:

1. Can be managed through a shared/smart pointer mechanism (MARTe-EX-F-2.1.1);
2. Can be automatically constructed given a class name (MARTe-EX-F-2.1.2 and MARTe-EX-F-2.1.4);
3. Provides a standard initialisation/configuration entry function;
4. It has a name (MARTe-EX-F-2.1.4);
5. Is introspectable (MARTe-EX-F-2.1.4).
    1. As a minimum the following information shall be available:
        1. The name of the class;
        2. The name of the object;
        3. The version of the class from which the code was compiled (i.e. revision control version)
    2. Shall enable reflection of derived classes that wish to expose this information.

As in the previous version of the framework any Object class instance shall automatically register itself in a class database. This database shall expose:

1. All the classes that were instantiated by the framework for any given application;
2. The number of objects that were instantiated for any given class.

It was agreed that as a general rule only the final classes (i.e. the ones to be used by the end-user) will inherit from Object. This implies that the final classes are constructed as a collection of direct inheritance from Object and from all the interfaces. It is likely that there will be some documented exceptions to this rule.



It was also decided to forbid the usage of the new operator in objects. This will allow to allocate objects (and associated memory) to specialised heaps which is highly beneficial to embedded systems.

Smart/shared pointers

As in the previous version of MARTe (i.e. BaseLib2) the framework shall support, promote and enforce the usage of shared pointers. This was implemented using GCRTemplate and GCRefences. Only minor refactoring is expected.

Naming of classes to be agreed.

Reference containers

As per previous framework, i.e.:

1. Find by name
    1. Relative;
    2. Absolute;
2. Find by index

With the following additions:

1. Find(name) shall return a vector with all the references found for a given object name (and not the first that was found);
2. Support reverse find, i.e. allow for a contained object to know who is its father.
    1. This includes being able of getting the full history of the object path (i.e. to have a reference to each object that is in the path from this object to the root of the search)

Naming of classes to be agreed.

Global object database

As per previous framework. We have discussed the possibility of having multiple global object database on top of an anonymous root, but the idea was not mature yet.

History

#1 - 04.08.2015 11:18 - André Neto

- Description updated

#2 - 04.08.2015 11:33 - André Neto

- Description updated

#3 - 04.08.2015 14:45 - André Neto

- Description updated

#4 - 04.08.2015 14:59 - André Neto

- File Object.png added

- Description updated

#5 - 04.08.2015 17:42 - André Neto

- Description updated

#6 - 11.04.2019 16:00 - André Neto

- Status changed from New to Closed

Files

Object.png	9.42 KB	04.08.2015	André Neto