OpenMI version 2 Development plan

GOTO OATC Wiki Home

1: Reengineering and minor refinements (no conceptual changes)

- 1.1 : Renaming
  Renaming methods and properties where these only make sense for models. Examples for this are properties like ModelID and Model description. The reason for this change is that when people are migrating data bases, or other non model component the current naming is misleading.

- 1.2 : Changing to collections
  Version 1.4 has many examples of Fortran like methods, such as GetOutputExchangeItem(index) and OutputExchangeItemCount. These should be changed to generic lists. This will make the standard simpler and make programming easier and safer. See discussion

2: The Link, Trigger, and Getvalues (minor conceptual change)

- 2.1 : The link should be provided by the linkable component.
  With version 1.4 the link is provided by the GUI, which makes it impossible for the developers to take advantage of making their own smart implementation. The link should therefore be provided by the providing component. This will enable the component provider to make more simple and efficient wrapper code. One specific problem we had with dataOperation arguments that were over-written, is also solved when the own implementation of the link is possible.

- 2.2 : It should be possible to invoke the GetValues method from other places than other LC's
  With version 1.4 only LinkableComponent can invoke the GetValues methods. This makes it difficult to use LinkableComponent in contexts other than model to model linking. Moreover, testing (e.g. Unit testing) of LinkableComponents is a tedious process, because all kinds of dummy LinkableComponet must be created in order to test the invocation of GetValues. We have some idea of how to do this, but there are still some problems with clearing buffers, because the callback for getting the earliest input time requires some handle to the calling component. As I see it, moving the GetValue to the Link does not solve this problem, anyway this might still be a fine idea.

- 2.3 : Getting rid of the trigger
  The concept of the trigger confuses many users. We can get rid of the trigger by adding an Update method to the ILinkableComponent. When this method is invoked this component will progress until a time specified in the method arguments and pull the remaining components in the configuration.

3 : GetValues returning other types than arrays of doubles (medium or major conceptual change)

Specifically Alterra has asked for the ability of returning categorized data, which calls for extending the allowed return types from the GetValues method. Currently only ScalarSets (arrays of doubles) and vector sets are allowed. However, instead of extending the list of allowed data types the generic type Object could be the one to return.

4 : Persistent state, set parameters, etc (minor conceptual change)

Peter Gisjbers has for a very long time asked us to solve a very specific problem, which arises e.g. in connection with using OpenMI compliant models in real-time systems like Delft Fews.

5 : Time Issues (Major conceptual change)

For LinkableComponent which does not work with time, many methods in the current standard is confusing, because time is used. E.g. time is an argument in the GetValues method. So, the task is to go through all the standard interfaces and if possible change things to the standard works in logical way both for our time stepping models and for timeless components.

6 : Put values (Major conceptual change)

For people using LinkableComponents for simple things the pull driven pattern it is difficult to understand. We should consider allowing a push driven approach also, which means adding PutValues methods to the standard.

7 : DSS's

One on the problems with the current standard is that people running OpenMI configuration based on components from different providers need access the model input and output data through the individual (often very different) user interfaces. So, can we change the standard so it allows for building complete user friendly systems which are underneath running OpenMI components?

8 : Merge ILinkableComponent and IEngine

Currently most of users in reality implement IEngine and not ILinkableComponent and as result it makes wrappers dependent a lot on SDK. The suggestions is to include functionality of IEngine into new ILinkableComponent.