Problem

Are there are limitations on how/where IDL events are defined?

Solution

Most definitely!

For those unfamiliar with the history of the notification channel API ACS provides; during one of the first ALMA Software integrations there were many problems with subsystems publishing events with **`event_type`** names unknown to other subsystems, subsystems not subscribing to the correct **`event_type`**, etc. In addition to this, there were complaints from developers that the API required too much CORBA knowledge. To make a long story short, the Integration, Testing, and Support team requested that ACS make it harder for there to be confusion about events between subsystems and HLA requested that CORBA be completely hidden from developers using the NC API. As a result, the API became trivial to use but lost a lot of its power in the process. Below is a list of restrictions that came as a result of these requests:

- events are limited strictly to instances of IDL structs. While you may be able to publish another CORBA type such as a sequence from C++ using templated methods, one cannot necessarily receive this type in another language (e.g., Java).
  - While ACS does not fully support a sequence of event structs as the event data type, it is still possible (with more CORBA exposure) to receive such events in Java. If there should be a need for this, please contact the ACS team, and or have a look at the ACS modules acssamp, acssampGui.
- IDL event structs must be defined within top-level IDL modules and **not** sub-modules, interfaces, etc. This is because of the fact that Java's IDL mapping is quite strict regarding CORBA anys ACS must create to store your event within **and** ACS does not have advance knowledge of every type of IDL struct/event that will be sent within ALMA. A more detailed explanation is that for Java:
  - ACS (*alma.acs.nc.Consumer* to be precise) must dynamically obtain a reference to a method called *extract* to convert the ICD event (i.e., CORBA any) to the correct Java object which is in turn passed to the *receive* method you have registered with a *Consumer* object
  - The *extract* method is defined within the CORBA stubs for the IDL defining your ICD event. ACS has no knowledge of these stubs
  - The *extract* method is defined within a class which is named after the ICD event with "Helper" appended to it. For example, the standard "temperatureDataBlockEvent" would have an *extract* method defined in "temperatureDataBlockEventHelper"
  - Using the CORBA Any/event, ACS takes advantage of certain **`JacORB`** features (i.e., **`JacORB`** provides a nice implementation of the **`toString`** method for CORBA **`TypeCode`** objects) which lets us determine the "Helper" class for your ICD event **if and only if** the ICD event is defined within IDL modules alone. If you define the ICD event within an IDL interface; we cannot automatically determine the helper class containing this *extract* method because ACS cannot tell that your ICD event was defined within an IDL interface:
    - the Java CORBA mapping states that IDL constructs declared inside an IDL interface are mapped into a package with "Package" appended to its name. Confer chapter 1.17 "Mapping for Certain Nested Types" in the OMG Mapping Specification.
    - the **`toString`** method of CORBA **`TypeCode`** objects does **not** append "Package" after the IDL interface name within the return value
    - determining that "Package" is part of the package/class name of the "Helper" object **might** be possible, but would be a very heavy operation requiring numerous networks calls to the CORBA Interface Repository
  - Note that as of ACS 6.0.2, it is possible (but still discouraged) to define an event struct inside an IDL interface. Stronger nesting has not been tested and probably still fails.
- ALMA Software Engineering has a coding standard which states that the names of all IDL structs used as event data must end with *Event*.
- The usual ACS restriction applies that CORBA object references not be passed around between components, see FAQPassComponentReference.

Related articles

- How can more people do development with ACS on the same machine without disturbing each other?
- Which ports are used by ACS?
- Problems connecting to ACS servers on a remote machine: bad /etc/hosts
- Why does the getComponent method of ZLegacy/ACS.ContainerServices return an object of type None?
- Why are some of my print statements not showing up in the container output section of acscommandcenter?