Problem

How can I inherit from two different IDL interfaces?

Solution

Suppose there is an idl file:

```
module Example
{
   interface Simple1 : ACS::ACSComponent {
      void Routine1();
   };


  interface Simple2 : ACS::ACSComponent {
    void Routine2();
  };
};
```

And I want to write a component which implements both of these interfaces. Should I:

1) Create an interface which inherits from both of these, and then do the implementation as usual?

OR

2) Create an implementation which inherits from both of the POA classes for the two interfaces (generated from IDL) of both of these?

The answer is 1 - create an interface which inherits from both of the interfaces, then proceed as usual. Your IDL file, then, would end up looking something like this:

```
module Example
{
   interface Simple1 : ACS::ACSComponent {
      void Routine1();
   };


  interface Simple2 : ACS::ACSComponent {
    void Routine2();
  };

  interface Derived: Simple1, Simple2 {};

};
```

and you would implement the interface *Derived*.

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?