

ALUMNO: David Martínez Campos

Asignatura: Programación de Sistemas Distribuidos

Curso: 2021/2022
Semestre: 2º

Fecha: 10-04-2022

PRÁCTICA 5: Para la práctica 5 quiero ofreceros varias opciones pero por supuesto si alguien quiere hacer alguna propuesta de algo relacionado con sistemas distribuidos, adelante. Las propuestas son las siguientes:

- a) ¿Te atreves a hacer la práctica 2 de CORBA en otro lenguaje como Python o Ruby? Muestra evidencias del trabajo con capturas. Punto extra la grabación de un vídeo (no hace falta que la calidad sea buena), simplemente el resultado

El código debería de funcionar pero no se porque me sigue dando errores de que omniidl no encuentra Python habiéndomelo desinstalado tanto omniidl y Python y volviéndolo a instalar en otro orden.

Independientemente, en el github están subidos los códigos .py tanto de los ficheros que se han de crear como de los generados por idl (estos los he inferido de ver un par de tutoriales y de los archivos de la práctica 2 con java); no puedo adjuntar capturas porque omniidl no me reconoce Python así que describiré los pasos.

Primero lo que tendremos que hacer será crear el fichero idl ,en este caso se llama Hello.idl

```
hello.idl
1  module HelloApp {
2      interface Hello {
3          string sayHello();
4      };
5  };
```

Tiene esencialmente la misma forma que el hello.idl de java.

Y los ficheros de Cliente, Servidor y Servant respectivamente



```
HelloClient.py
1  import sys
2  from omniORB import CORBA
3  import CosNaming
4  import HelloApp
5
6
7  orb = CORBA.ORB_init()
8
9  obj = orb.string_to_object("corbaname::localhost:1050/NameService#hello")
10 hello = obj._narrow(HelloApp.Hello)
11
12 if hello is None:
13     print("Can't narrow reference")
14     sys.exit(1)
15
16 print(hello.sayHello())
```



```
◆ HelloServer.py
1  import sys
2  from omniORB import CORBA
3  import CosNaming, PortableServer
4  from HelloServant import HelloServant
5
6  sys.argv.extend(("ORBInitRef", "NameService=corbaname::localhost:1050"))
7
8  # Initialize the ORB and the POA
9  orb = CORBA.ORB_init(sys.argv, CORBA.ORB_ID)
10 poa = orb.resolve_initial_references("RootPOA")
11
12 # create servant and register it with the POA
13 hello = HelloServant()
14 servantId = poa.activate_object(hello);
15
16 # Publish the hello object reference to the Naming Service
17 ref = poa.id_to_reference(servantId)
18
19 # Get the Naming Service's root naming context
20 obj = orb.resolve_initial_references("NameService")
21 rootContext = obj._narrow(CosNaming.NamingContext)
22
23
24 if rootContext is None:
25     print("Failed to narrow the root naming context")
26     sys.exit(1)
27
28 # Create a new context naming
29 # path = [CosNaming.NameComponent("some context", "")]
30
31 # Bind the the new context to the root context
32 # try:
33 #     context = rootContext.bind_new_context(path)
34 #     print("New hello context bound to the Naming Service")
35 #
36 # except CosNaming.NamingContext.AlreadyBound, ex:
37 #     obj = rootContext.resolve(path)
38 #     context = obj._narrow(CosNaming.NamingContext)
39 #
40 #     if context is None:
41 #         print("Context exists but is not a NamingContext")
42 #         sys.exit(1)
43
44 text = "hello"
45 path = [CosNaming.NameComponent("hello", "")]
46
47 try:
48     # context.bind(path, ref)
49     rootContext.bind(path, ref)
50     print("Bound the hello object to the naming service")
51
52 except CosNaming.NamingContext.AlreadyBound, ex:
53     print("Hello object already bound, rebinding new object")
54     # context.rebind(path, ref)
55     rootContext.rebind(path, ref)
56
57 # Activate the POA Manager and run
58 poa._get_the_POAManager().activate()
59 print("Python Server active and waiting...")
60 orb.run()
```



```
HelloServant.py
1  import HelloApp, HelloApp__POA
2
3  class HelloServant (HelloApp__POA.Hello):
4  |      def sayHello(self):
5  |          return "Hello from Python Server!!!"
```

Lo que hacemos ahora es compilar con la siguiente instrucción:

omniidl -bpython hello.idl


Generandonos los stubs o el esqueleto de la aplicación que en este caso esta organizada así:

```
▼ HelloApp
  ● __init__.py
  ▼ HelloApp__POA
    ● __init__.py
    ● Hello_idl.py
```

Ficheros que respectivamente son:

```
● __init__.py X
HelloApp > ● __init__.py
1  # DO NOT EDIT THIS FILE!
2  #
3  # Python module HelloApp generated by omniidl
4
5  import omniORB
6  omniORB.updateModule("HelloApp")
7
8  # ** 1. Stub files contributing to this module
9  import Hello_idl
10
11 # ** 2. Sub-modules
12
13 # ** 3. End
```



```
HelloApp_POA >  _init_.py
1  # DO NOT EDIT THIS FILE!
2  #
3  # Python module HelloApp_POA generated by omniidl
4
5  import omniORB
6  omniORB.updateModule("HelloApp_POA")
7
8  # ** 1. Stub files contributing to this module
9  import Hello_idl
10
11 # ** 2. Sub-modules
12
13 # ** 3. End
```



```
◆ Hello_idl.py
1  # Python stubs generated by omniidl from Hello.idl
2  # DO NOT EDIT THIS FILE!
3
4  import omniORB, _omniipy
5  from omniORB import CORBA, PortableServer
6  _o_CORBA = CORBA
7
8
9  _omniipy.checkVersion(4,2, __file__, 1)
10
11  try:
12      property
13  except NameError:
14      def property(*args):
15          return None
16
17
18  #
19  # Start of module "HelloApp"
20  #
21  __name__ = "HelloApp"
22  _o_HelloApp = omniORB.openModule("HelloApp", r"Hello.idl")
23  _o_HelloApp_POA = omniORB.openModule("HelloApp_POA", r"Hello.idl")
24
25
26  # interface Hello
27  _o_HelloApp_d_Hello = (omniORB.tcInternal.tv_objref, "IDL:HelloApp/Hello:1.0", "Hello")
28  omniORB.typeMapping["IDL:HelloApp/Hello:1.0"] = _o_HelloApp_d_Hello
29  _o_HelloApp.Hello = omniORB.newEmptyClass()
30  class Hello :
31      _NP_RepositoryId = _o_HelloApp_d_Hello[1]
32
33      def __init__(self, *args, **kw):
34          raise RuntimeError("Cannot construct objects of this type.")
35
36      _nil = CORBA.Object._nil
37
38
39  _o_HelloApp.Hello = Hello
40  _o_HelloApp_tc_Hello = omniORB.tcInternal.createTypeCode(_o_HelloApp_d_Hello)
41  omniORB.registerType(Hello._NP_RepositoryId, _o_HelloApp_d_Hello, _o_HelloApp_tc_Hello)
42
43  # Hello operations and attributes
44  Hello_d_sayHello = (((), ((omniORB.tcInternal.tv_string,0), ), None)
45
46  # Hello object reference
47  class _objref_Hello (CORBA.Object):
48      _NP_RepositoryId = Hello._NP_RepositoryId
49
50      def __init__(self, obj):
51          CORBA.Object.__init__(self, obj)
52
53      def sayHello(self, *args):
54          return self._obj.invoke("sayHello", _o_HelloApp.Hello_d_sayHello, args)
55
56  omniORB.registerObjref(Hello._NP_RepositoryId, _objref_Hello)
57  _o_HelloApp._objref_Hello = _objref_Hello
58  del Hello, _objref_Hello
59
60  # Hello skeleton
61  __name__ = "HelloApp_POA"
62  class Hello (PortableServer.Servant):
63      _NP_RepositoryId = _o_HelloApp.Hello._NP_RepositoryId
64
65
66      _omni_op_d = {"sayHello": _o_HelloApp.Hello_d_sayHello}
67
68  Hello._omni_skeleton = Hello
69  _o_HelloApp_POA.Hello = Hello
70  omniORB.registerSkeleton(Hello._NP_RepositoryId, Hello)
71  del Hello
72  __name__ = "HelloApp"
73
74  #
75  # End of module "HelloApp"
76  #
77  __name__ = "Hello_idl"
78
79  _exported_modules = ( "HelloApp", )
80
81  # The end.
```

Link del repositorio github: <https://github.com/dmartinezc4/Practica-5-distribuidos>

Webgrafía/Enlaces usados

<https://omniorb.sourceforge.io/omnipy3/omniORBpy/>

https://www.youtube.com/watch?v=nJYp3_X_p6c

<https://github.com/rebecasouza/hello-CORBA>

<https://github.com/scottsextton/python-corba>

<https://www.oreilly.com/library/view/python-cookbook/0596001673/ch13s06.html>

<https://github.com/troegeer/corba-example>