

Bootstrap Process in the Bionet Platform

Jun Suzuki, Michael Le and Tatsuya Suda

{jsuzuki, suda}@ics.uci.edu

mvle@uci.edu

<http://netresearch.ics.uci.edu/bionet/>

Dept. of Information and Computer Science
University of California, Irvine

1

Bootstrapping

- When starting to execute the bionet platform, you should perform the bootstrap process first.
- Bootstrapping is the process to
 - assign a set of configuration properties to the bionet platform
 - initialize the bionet message transport
 - initialize the bionet container
 - initialize the platform representative
 - initialize a set of bionet services
 - Initialize a data collector object (optional)
 - initialize a CE factory object (optional)
 - start accepting incoming messages

Bootstrapping Interface

- Java interface is defined for bootstrapping.
- Interface definition:

2 Classes for Bootstrapping

- The current bionet platform provides 2 different classes that implements the bootstrapping interface.
 - `edu.uci.ics.bionet.util.BootptrapWithMuCORBA`
 - bootstraps the bionet platform with the default bionet message transport
 - our own CORBA implementation called `muCORBA`.
 - `edu.uci.ics.bionet.util.BootptrapWithJacORB`
 - bootstraps the bionet platform with `JacORB`, another (third party) CORBA implementation, as a bionet message transport.
 - Note that `JacORB` version 1.4.1 or higher should be used with this class.
 - » If you want to use any lower version of `JacORB`, use

Bootstrap Process

Bootstrap process is same, independently from which message transport (muCORBA or JacORB) to be used.

Action sequence

– Assign a set of configuration properties to the bionet platform

- locate the bionet.properties file to obtain a set of configuration properties
 - Configuration properties are described in the file named “bionet.properties”.
 - » placed at the user's home directory.
 - » formatted as key-value pairs.
 - Each user needs to configure the file before bootstrapping the bionet platform.
 - c.f. See also the documentation about the format of configuration properties
- register them into the Java's system property.

5

– Initialize the bionet message transport

- instantiate ORB through calling `init()` on `org.omg.CORBA.ORB`.
 - c.f. See the documentation about command-line arguments for the parameters assigned to `init()`.

– Initialize the bionet container

- create a default (root) POA through calling `resolve_initial_references("RootPOA")` on the initialized ORB (bionet message transport).
- create a bionet-specific (child) POA
- assign the bionet container policies to the created (child) POA
 - See the last slide for a set of standard policies in the bionet container
- create the CEActivator (a servant manager) and assign it to the bionet container
 - CEActivator is used to activate and deactivate CEs.
- activate the bionet container.

– Initialize the platform representative

- instantiate `PlatformRepresentative` and register it in the bionet container
 - c.f. see the documentation about platform representative for more details.
- calls `setPlatformRepresentativeRef()` on the created platform representative to register its own reference.
- calls `setBionetContainer()` on the created platform representative to register the reference to the bionet container.

– Initialize a set of bionet services

- lookup Java's system property to see which bionet services are initialized.
 - That should be described in `bionet.properties`.
 - » c.f. See also the documentation about the format of configuration properties
- initialize the specified bionet services and register their references to the platform representative.
 - c.f. see the documentation about platform representative for more details.

– Initialize a data collector object (optional)

- lookup Java's system property to see if a data collector object needs to be created.
 - That should be described in `bionet.properties`.
 - » c.f. See also the documentation about the format of configuration properties
 - A data collector object is used to collect various measurement data in experiments (e.g. CE's energy level, CE's location, etc.)
 - » c.f. see the documentation about data collector for more details.
- create a data collector object and register its reference to the platform representative.
 - c.f. see the documentation about platform representative for more details.

Standard Choices in the Bionet Container

- Initialize a CE factory object (optional)
 - lookup Java's system property to see if a CE factory object needs to be created.
 - That should be described in `bionet.properties`.
 - » c.f. See also the documentation about the format of configuration properties.
 - A CE factory is used to install CEs to the bionet platform.
 - » c.f. see the documentation about CE factory for more details.
 - create a CE Factory and register its reference to the platform representative.
- Start accepting incoming messages
 - calls `run()` on the bionet message transport (ORB).

- A set of POA policies used in the bionet container
 - TRANSIENT
 - NON_IMPLICIT_ACTIVATION
 - USER_ID
 - USE_SERVANT_MANAGER
 - RETAIN
 - UNIQUE_ID
 - ORB_CTRL_MODEL