CHAPTER 7

OUTPUT SCREENS OF THE PLATFORM
INTEROPERABLE SOLUTION TOOL

7.1 INTRODUCTION

This chapter shows the output screen shots of the tool Platform Interoperable Solution (PIS) for the following two cases:

- .NET Component with Java Client
- EJB component with .NET client

Whatever may be the client’s platform (.NET or Java), the client should first run the configuration server, through which one form of component is configured into another form. Since the configuration server had been created using java, run that server as shown in the screen shot given below.

7.2 .NET COMPONENT WITH JAVA CLIENT

To configure the .NET component form into another form for java client first run the configuration server.
On running the configuration server we get.

From the client machine, run the client program for which the configuration is needed.

Running the configuration client results in:

The screen shot shown below is the login window form. The IP address is the address of the machine in which the component had been deployed.
While clicking on ‘Login’ button the configuration window, which is shown below will be displayed.

In this case, .NET component has to be configured into accessible form to Java client. So choose .Net Service config option button. The respective commands stored in XML files will be loaded in to the textbox, by clicking ‘load’ button. The commands can be stored during run time into the XML file with the help of ‘Save’ button. The steps of configuring the .NET component into IDL are shown in the following screen shots.
By clicking on ‘Perform’ button the IDL form will be created for the given .NET component form. Now, IDL form has to be configured into the accessible form of Java client. This configuration is done by clicking on Client Service Config option button. The necessary configuration commands,
which had been already stored in XML files will be automatically executed one by one as shown in the following screen shot:

Now the .NET component is in accessible form to the Java Client. So run the .NET server and then run the Java client to access the .NET component.
Running the Java Client results in:

```
D:\practice\Latha\OutProcess\DotNet\bin
D:\practice\Latha\OutProcess\DotNet\bin\AdderServer.exe
server running

D:\practice\Latha\OutProcess\DotNet\runClient.bat
D:\practice\Latha\OutProcess\DotNet\java -Djava.naming.factory.initial=con:com.jm
d1.cosnaming.COMctxFactory -Djava.naming.provider.url=iio://localhost:8087 AddCl
tient
input the two commands:
surname 1:
45
surname 2:
6
get initial naming context
cic received, retrieve add
call add method
result: 51.0
D:\practice\Latha\OutProcess\DotNet
```
Similarly for EJB server with .NET client, first run the configuration server.

Now set the .NET environment on by executing the command: setenv (setCsc), which results in:
Instead of .Net service Config select EJB service config. Click on load button in order to load all the commands, which are stored in XML files.

By clicking on ‘load’ button the commands stored in XML files will be executed one by one automatically as shown in the following screenshot:
After getting IDL form for EJB component, to get the executable form for .NET client, click on client service config option button. By clicking on the ‘load’ button, the configuration steps stored in the XML files will be loaded and displayed in the text box of the configuration window, as shown in the following screen shot:

The commands stored in XML files are executed one by one automatically as shown in the following screen shot:
The screen shot of the deployed EJB component is shown below:

7.4 CONCLUSION

This Chapter outlines various steps involved in the implementation of the tool Platform Interoperable Solution and the resulting Screen Shots.