

WebSphere Application Server Version 6.1

Sales and Technical Enablement Workshop

Lab 07 – SIP Servlet Support

Part 1. Introduction

In this lab we will use the Application Server Toolkit (AST) to construct a simple SIP application that can be used to validate the installation and configuration of WebSphere Application Server v6.1 as a SIP Servlet provider. This is very similar to how we have used the HTTP Servlet snoop in the past to establish successful web container installation, configuration and operation.

We will use the open source SIP client SIPp obtained from http://sourceforge.net/projects/sipp to drive client load and provide an opportunity to test out various server scenarios. SIPp is a performance testing tool for the SIP protocol. It includes a few basic SipStone user agent scenarios and establishes and releases multiple calls with the INVITE and BYE methods. It features the dynamic display of statistics about running tests (call rate, round trip delay, and message statistics), periodic CSV statistics dumps, TCP and UDP over multiple sockets or multiplexed with retransmission management, regular expressions and variables in scenario files, and dynamically adjustable call rates.

We will use SIPp to provide a client load to our SIP application (designated as 'remote' in the following diagram), which is identified by the acronym UAC, representing User Agent Client. The basic signal sequence that this client generates and expects to receive is given in the following diagram. For a definition of what the SIP response codes represent, please see the Appendix.

```
SIPp UAC               Remote
        |(1) INVITE         |
        |------------------>|
        |(2) 100 (optional) |
        |<------------------|
        |(3) 180 (optional) |
        |<------------------|
        |(4) 200            |
        |<------------------|
        |(5) ACK            |
        |------------------>|
        |                   |
        |(6) PAUSE          |
        |                   |
        |(7) BYE            |
        |------------------>|
        |(8) 200            |
        |<------------------|
```


Part 2. Configure the Environment

If you do not have a standalone WebSphere Application Server profile from a previous lab, please create one now. If you do have this defined, please skip to step 8, below.

Note: You can not use a standalone WebSphere Application Server profile that was created using the development template because the SIP Container transport chains have not been defined…

We will use this application server as the deployment target for the SIP Servlet application that we develop, and not use it as a remote application server, like we did in the Application Server Toolkit lab. Rather, we will develop the application and "throw it over the wall" for testing. We will assume for the purposes of this lab that the name of the application server that you create(d) is AppSrv02.

__ 1. Use the Profile Management Tool (PMT) to create a standalone WebSphere Application Server.

__ 1.1. Select the shell icon to start a shell window (see the Appendix), then enter the command

```
/opt/IBM/WAS61/AppServer/bin/ProfileManagement/pmt.sh.
```

__ 1.2. Alternatively, open the Profile Management Tool directly from the K menu (see the Appendix) by selecting the lower "IBM WebSphere" → "Application Server Network Deployment V6.1" → "Profile Management Tool."

__ 1.3. Follow the following steps to create the new standalone application server:

__ 1.3.1. On the panel "Welcome to the Profile Management tool," select "Next."

__ 1.3.2. On the panel "Environment Selection," select "Next."

__ 1.3.3. On the panel "Profile Selection Options," select "Next."

__ 1.3.4. On the panel "Administrative Security," deselect "Enable administrative security," then select "Next."

__ 1.3.5. On the panel "Profile Creation Summary," select "Create."

__ 1.3.6. On the panel "The Profile Management tool created the profile successfully" deselect "Launch the First steps console," then select "Finish."

__ 2. Start the application server and determine the port that the administrative console is listening on.

__ 2.1. Select the shell icon to start a shell window (see the Appendix), then enter on one line the command

```
/opt/IBM/WAS61/AppServer/profiles/AppSrv02/bin/startServer.sh
server1
```

__ 2.2. Alternatively, start the application server directly from the K menu (see the Appendix) by selecting the lower "IBM WebSphere" → "Application Server Network Deployment V6.1" → "Profiles" → "AppSrv02" → "Start the server."

__ 2.3. If you do not already have a shell window open, select the shell icon to start a shell window (see the Appendix), then enter the following command on one line and record this port value (we observed 9062).

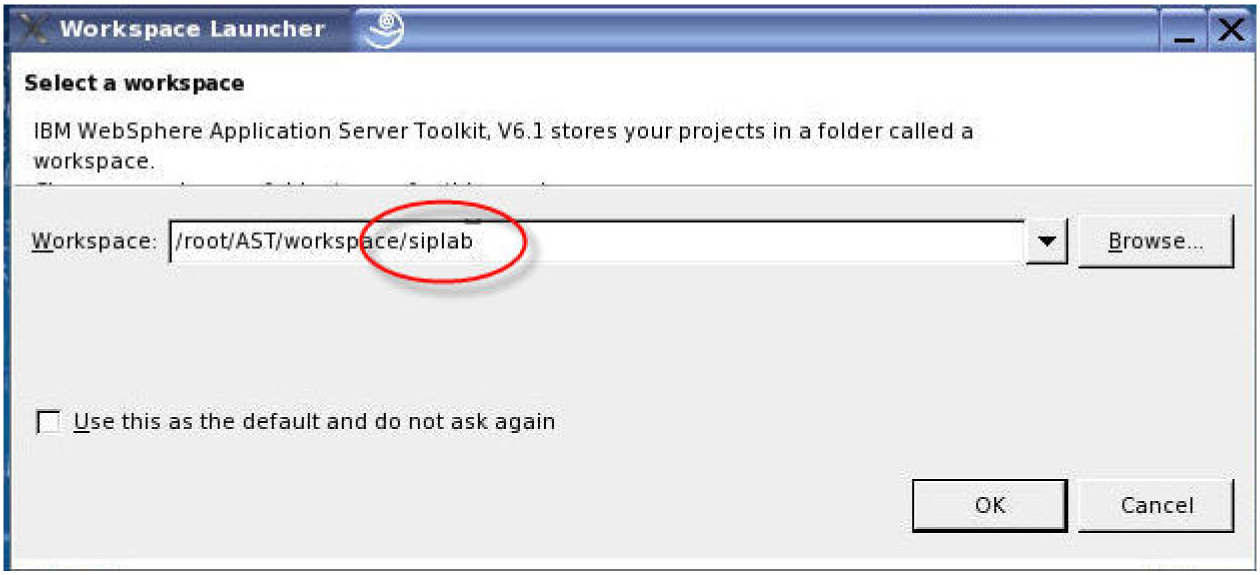

```
grep "Administrative console port"
/opt/IBM/WAS61/AppServer/profiles/AppSrv02/logs/AboutThisProfile
.txt
```

Part 3. Development of a SIP Servlet Application

__ 3. Start the Application Server Toolkit (AST)

__ 3.1. Open the AST directly from the K menu (see the Appendix) by selecting the upper "IBM WebSphere" → "Application Server Toolkit V6.1" → "Application Server Toolkit V6.1."

__ 3.2. On the panel "Select a workspace," specify a workspace of "/root/AST/workspace/siplab," then select "OK."

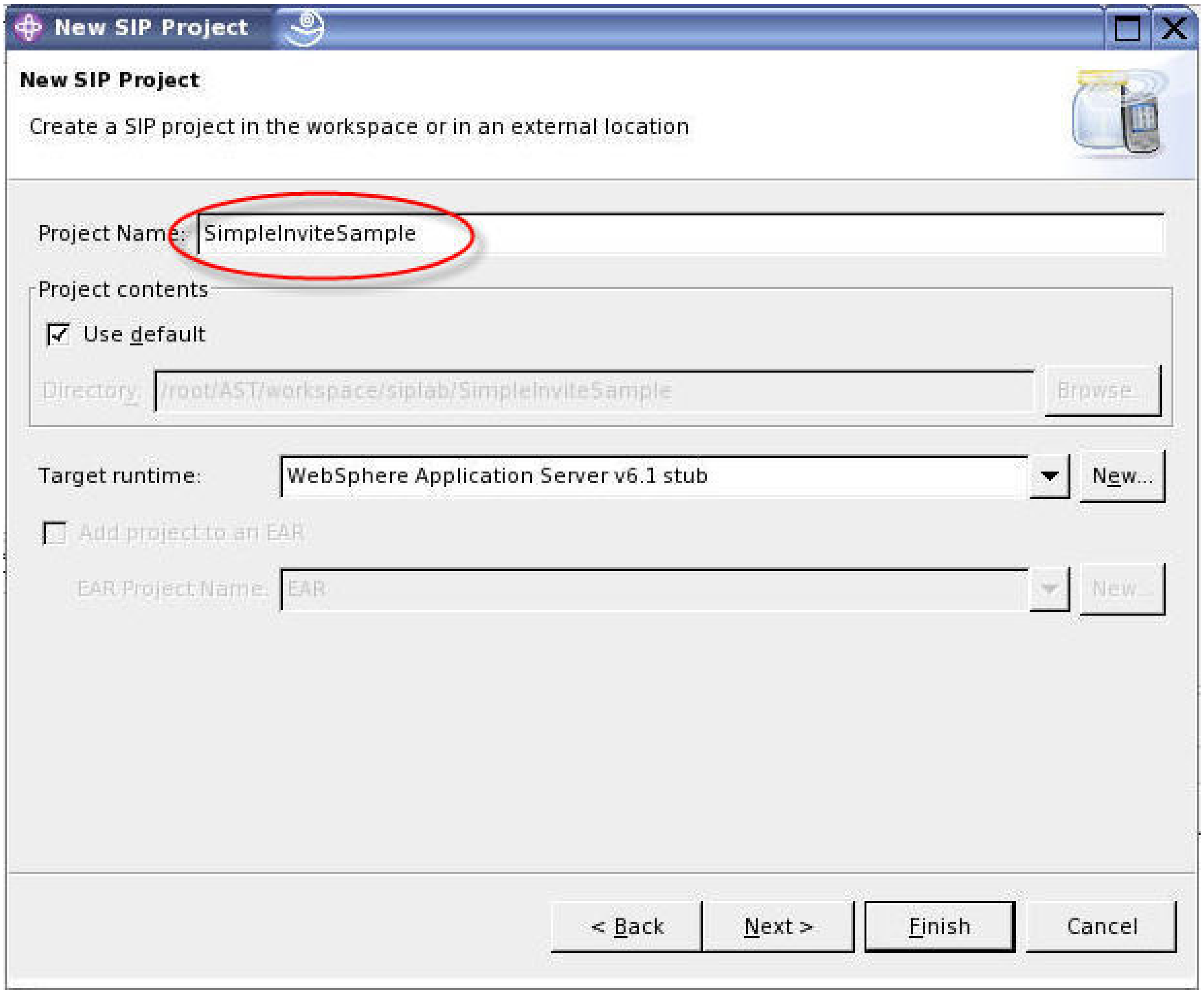


__ 3.3. Select the "Workbench" icon.

__ 4. Create a new SIP Project

__ 4.1. File → New → Other → SIP → SIP Project. On the panel "Select a wizard," select "Next."

__ 4.2. On the panel "New SIP Project," enter a project name of "SimpleInviteSample," then select "Next."



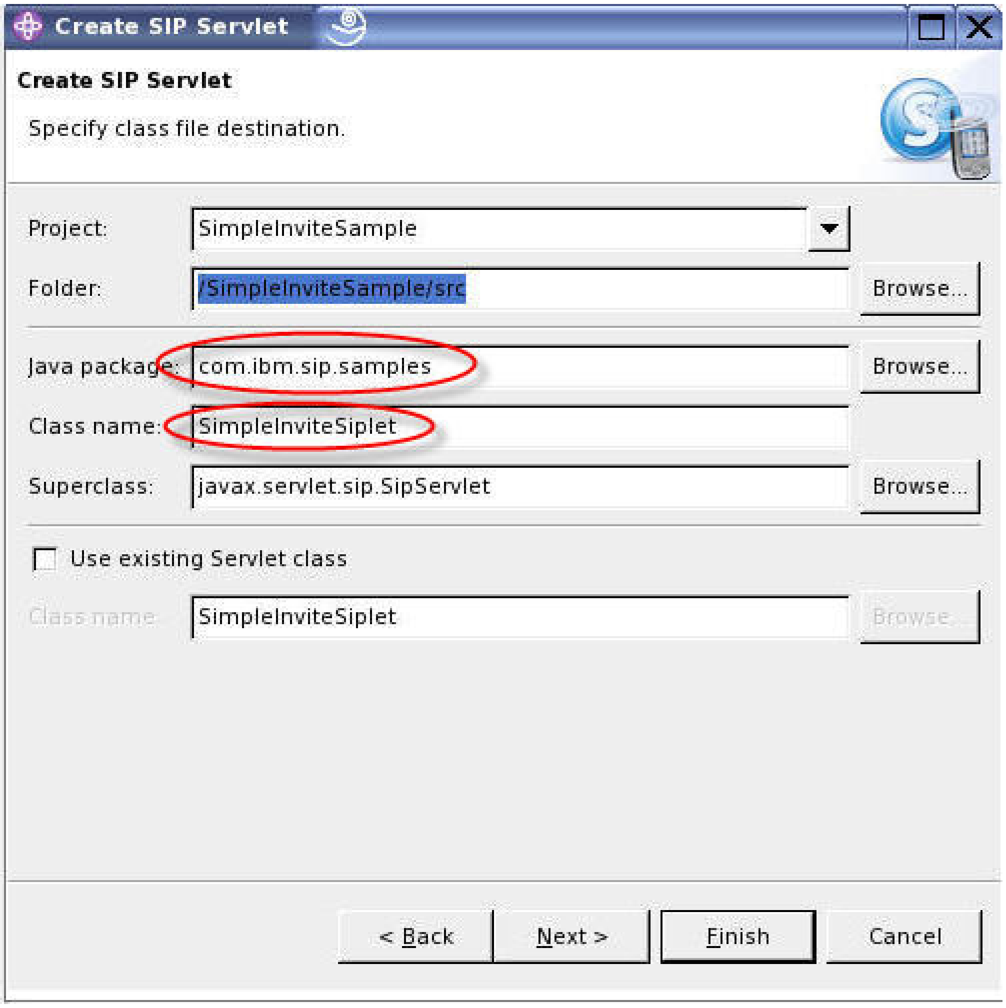
__ 4.3. On the panel "Select Project Facets," select "Finish."

__ 4.4. Select "Yes" to accept opening a J2EE perspective on the "Open Associated Perspective" dialog.

__ 5. Create a new SIP Servlet.

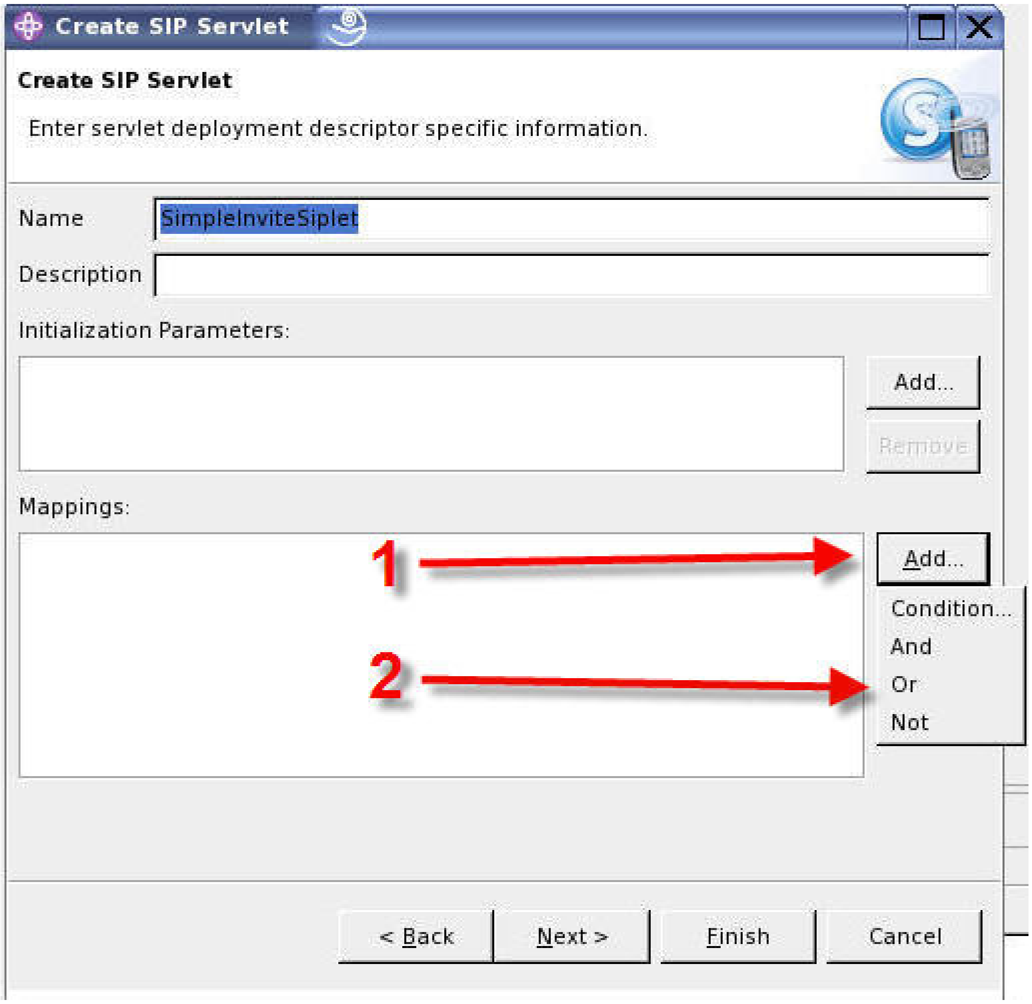
__ 5.1. File → New → Other → SIP → SIP Servlet. On the panel "Select a wizard," select "Next."

__ 5.2. On the panel "Create SIP Servlet," enter a Java package of "com.ibm.sip.samples," a classname of "SimpleInviteSiplet", then select "Next."

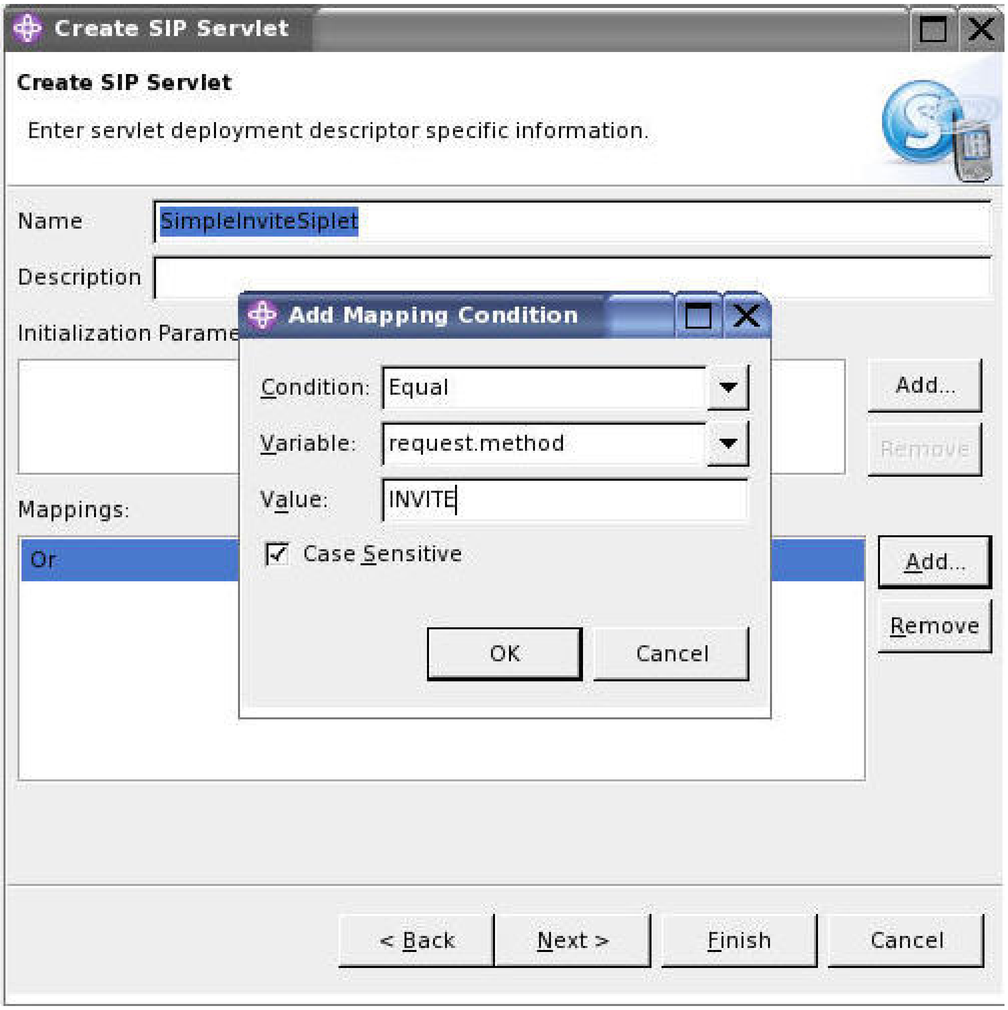


__ 5.3. Since this is a pure SIP application and not a converged application, we do not need to be concerned about application mappings as defined in web.xml. For SIP applications, we do need to define the appropriate mappings when the runtime receives specific SIP signals as we will define in sip.xml. We will need to handle the INVITE and BYE signals for our SIPp UAC test case. Add the following two mappings from the wizard. Please be careful to pay attention to the following sequence of steps:

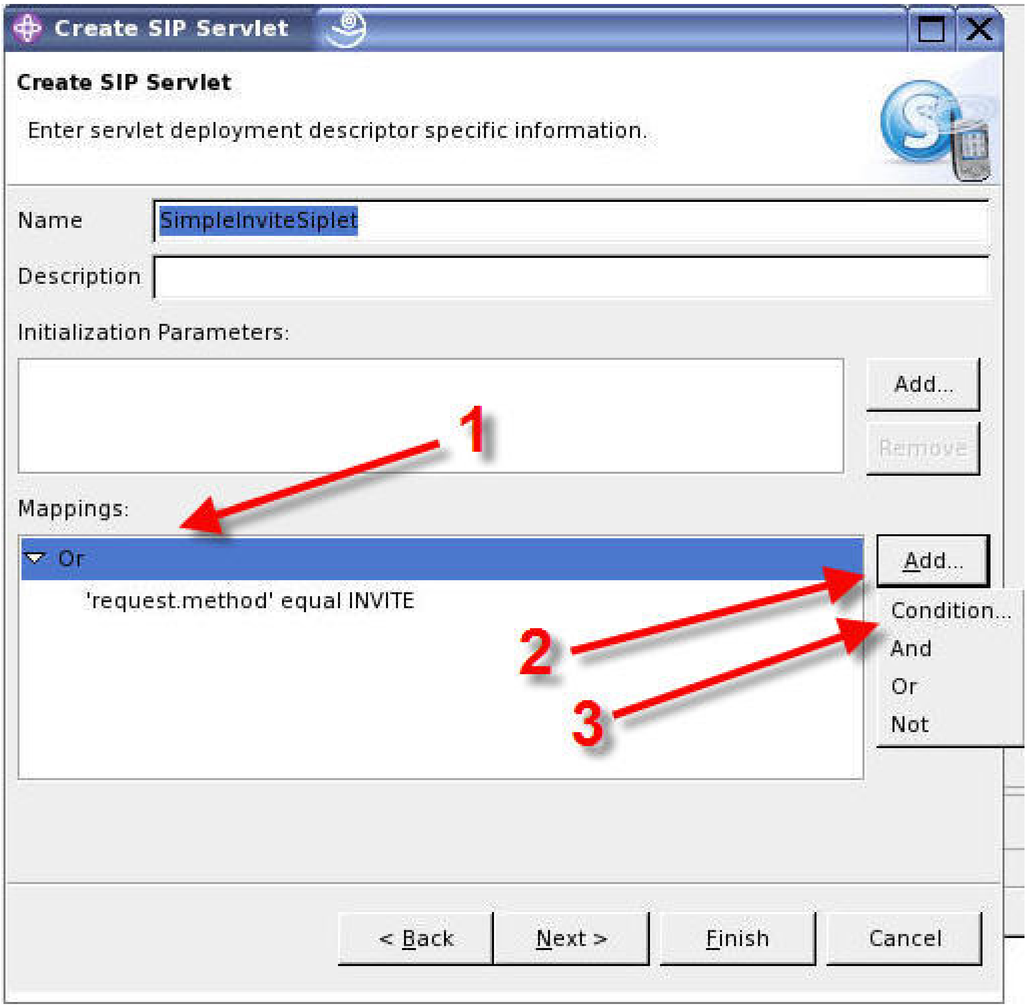
__ 5.3.1. On this second “Create SIP Servlet” panel, select the mappings “Add” button, then select “Or.”

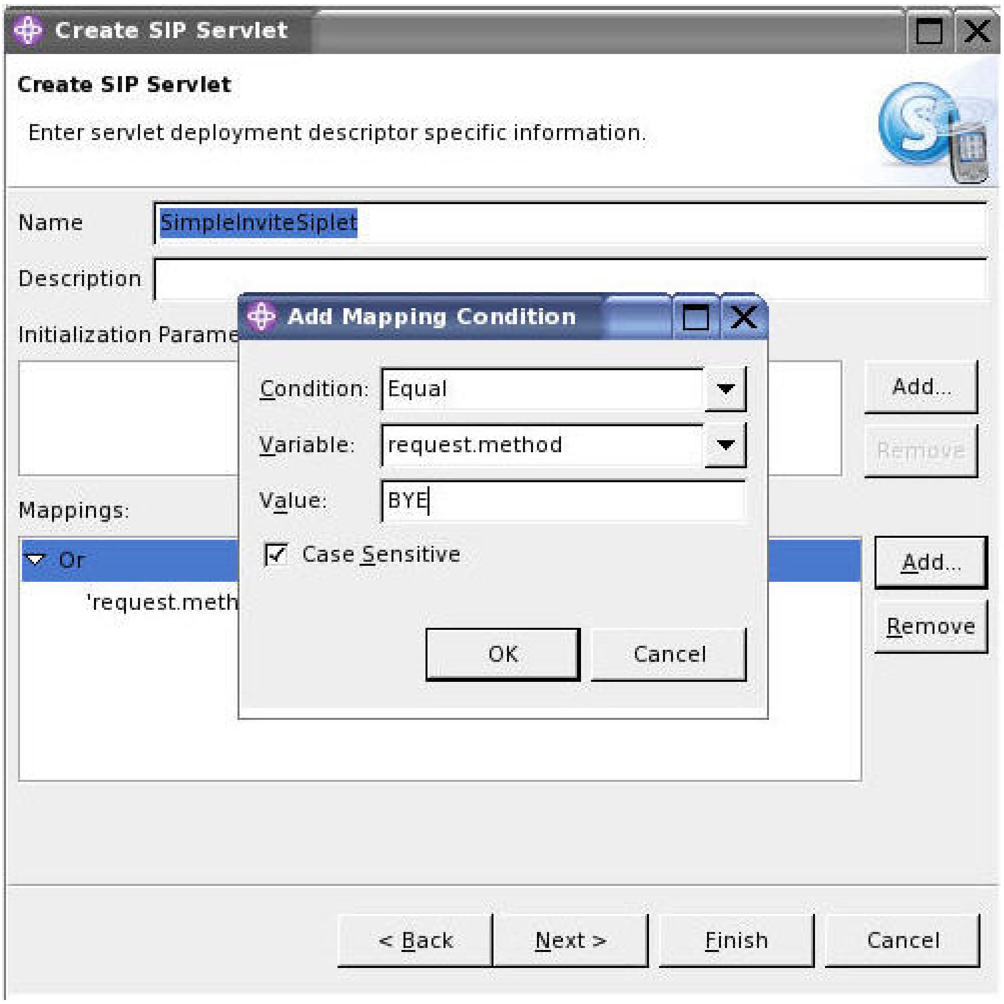


__ 5.3.2. Again, select the mappings "Add" button, then select "Condition..." and enter a value of "INVITE," then select "OK."



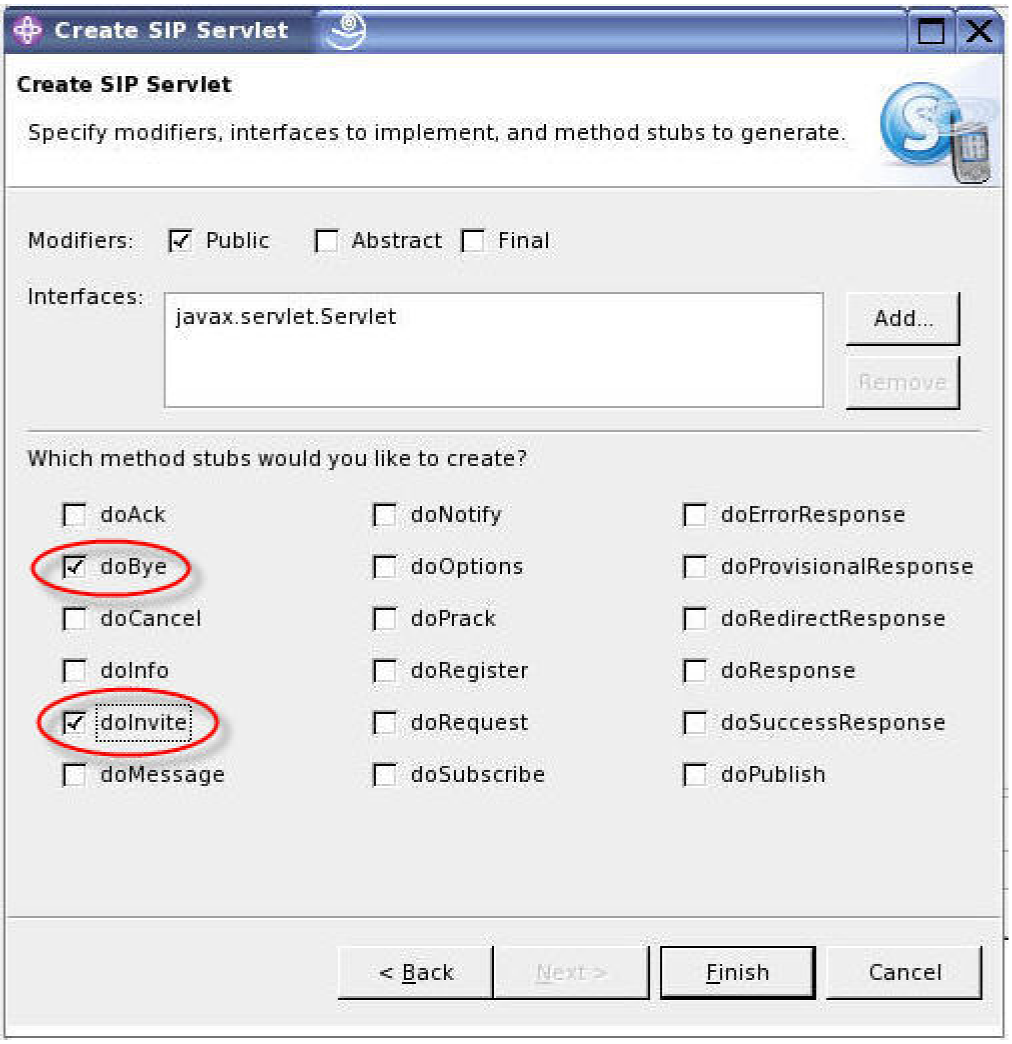
__ 5.3.3. Select “Or” listed under mappings then select the mappings “Add” button. Select “Condition.” Enter a value of “BYE.” Select “OK.”


Create SIP Servlet
Create SIP Servlet
Enter servlet deployment descriptor specific information.
Name
SimpleInviteSiplet
Description
Initialization Parameters:
Add...
Remove
1
Mappings:
Or
'request.method' equal INVITE
Add...
Condition...
And
Or
Not
2
3
< Back
Next >
Finish
Cancel



__ 5.3.4. We are finished identifying mappings, so select "Next."

__ 5.3.5. On this third "Create SIP Servlet" panel, select the "doBye" and "doInvite" methods for which to generate stubs. Finally, select "Finish."

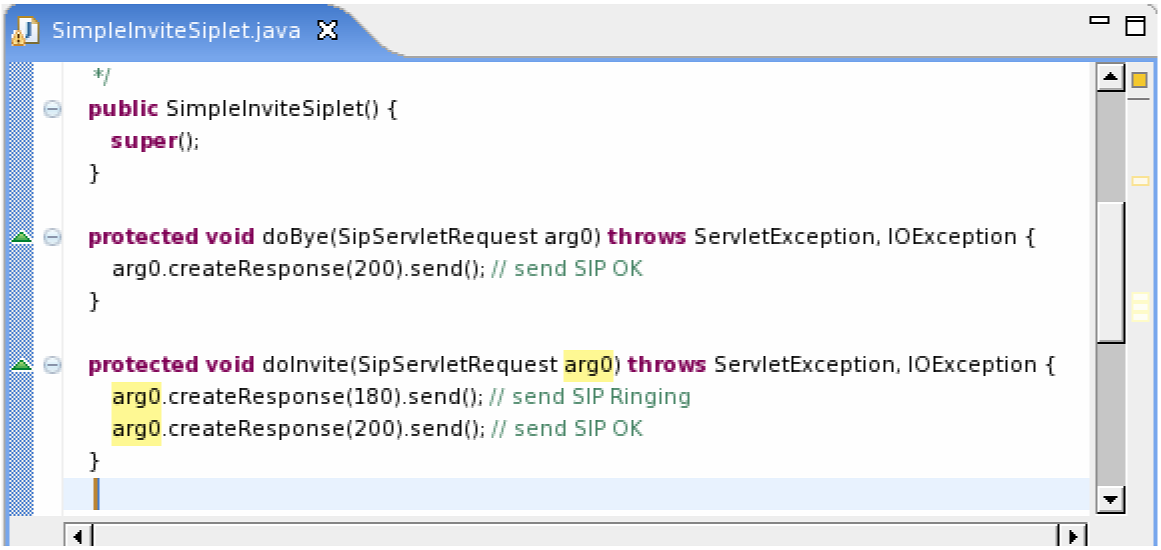


__ 5.3.6. Replace the code in the doBye method in the Java editor with

```
arg0.createResponse(200).send(); // send SIP OK
```

__ 5.3.7. Replace the code in the doInvite method with

```
arg0.createResponse(180).send(); // send SIP Ringing
arg0.createResponse(200).send(); // send SIP OK
```



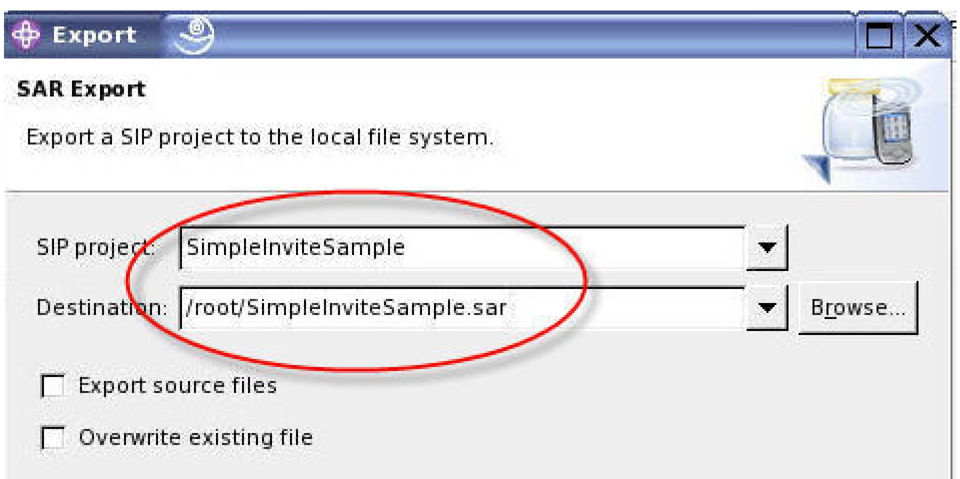
__ 5.3.8. Save your project by entering Control-S (^S).

__ 6. Export the SIP Project as a SIP Archive that can be installed into the runtime.

__ 6.1. File → Export → SAR File.

__ 6.2. On the panel "Select," select "Next."

__ 6.3. On the panel "SAR Export," identify the SIP project "SimpleInviteSample" with a destination of "/root/SimpleInviteSample.sar," then select "Finish."



Part 4. Deployment and Testing

__ 7. Deploy the SIP application

__ 7.1. Launch a browser at the port identified in step Part 2, Step 2.2, in our case, http://localhost:9062/ibm/console.

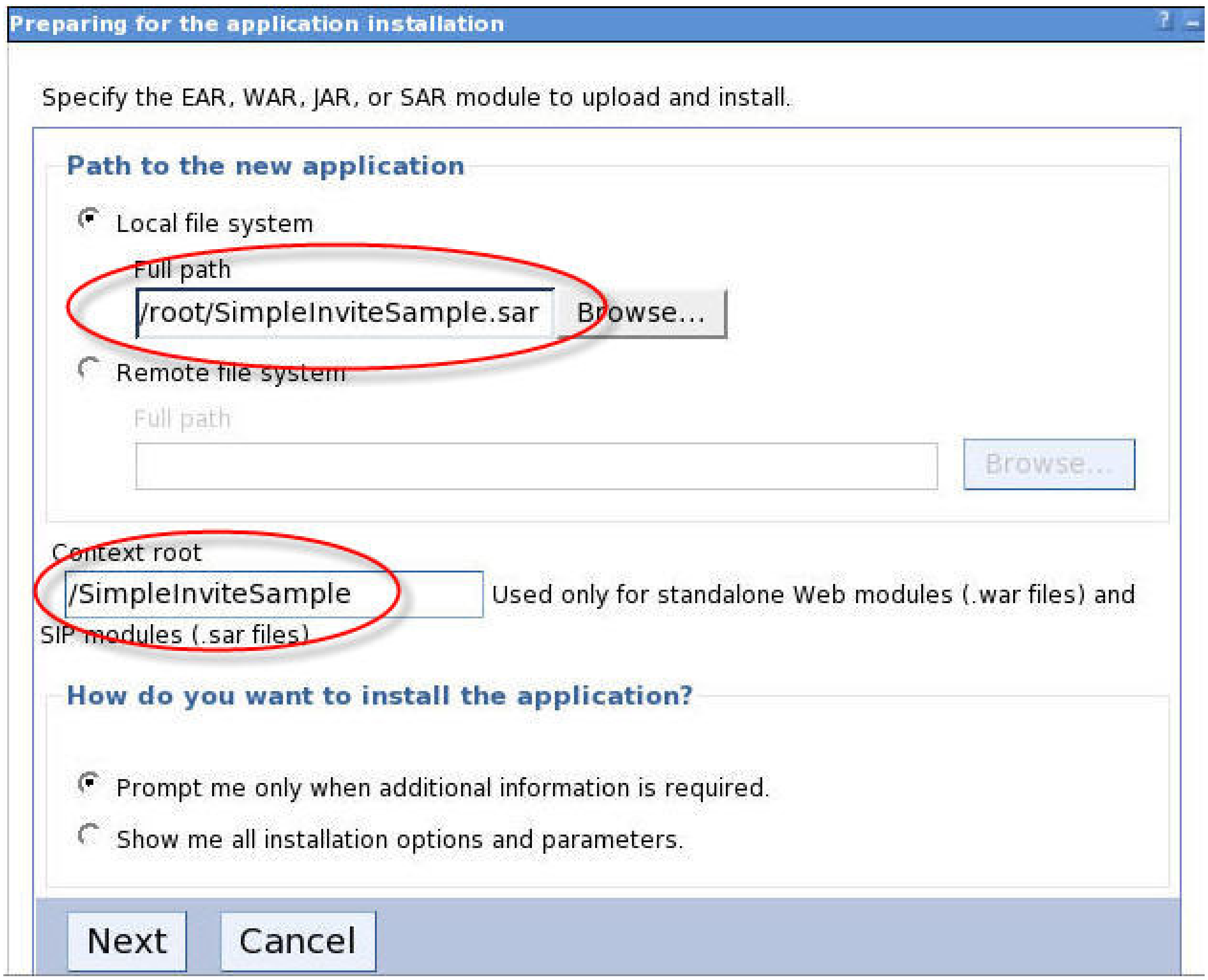
__ 7.2. Login to the Administrative Console.

__ 7.3. Install the SIP Archive (SAR).

__ 7.3.1. Applications → Install New Application.

__ 7.3.2. On the panel "Preparing for the application installation," browse the local file system for "/root/SimpleInviteSample.sar," then select "Open."

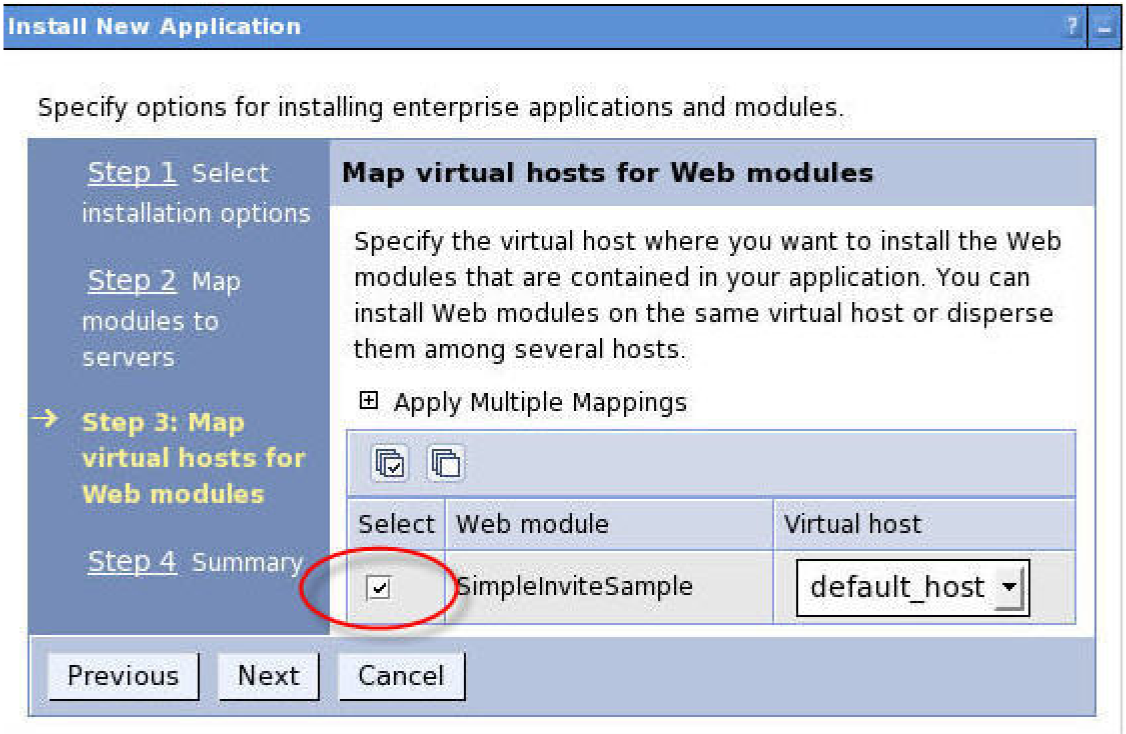
__ 7.3.3. Specify a context root of "/SimpleInviteSample" on this same panel, then select "Next."



__ 7.3.4. For Step 1, "Select installation options," accept the defaults and select "Next."

__ 7.3.5. For Step 2, "Map modules to servers," accept the defaults and select "Next."

__ 7.3.6. For Step 3, "Map virtual hosts for Web modules," select the module "SimpleInviteSample" and map to the virtual host "default_host," then select "Next."



__ 7.3.7. For Step 4, "Summary," select "Finish."

__ 7.3.8. Select "Save."

__ 8. Test the SIP application

__ 8.1. Prepare the SIP application "SimpleInviteSample" for testing.

__ 8.1.1. Start "SimpleInviteSample."

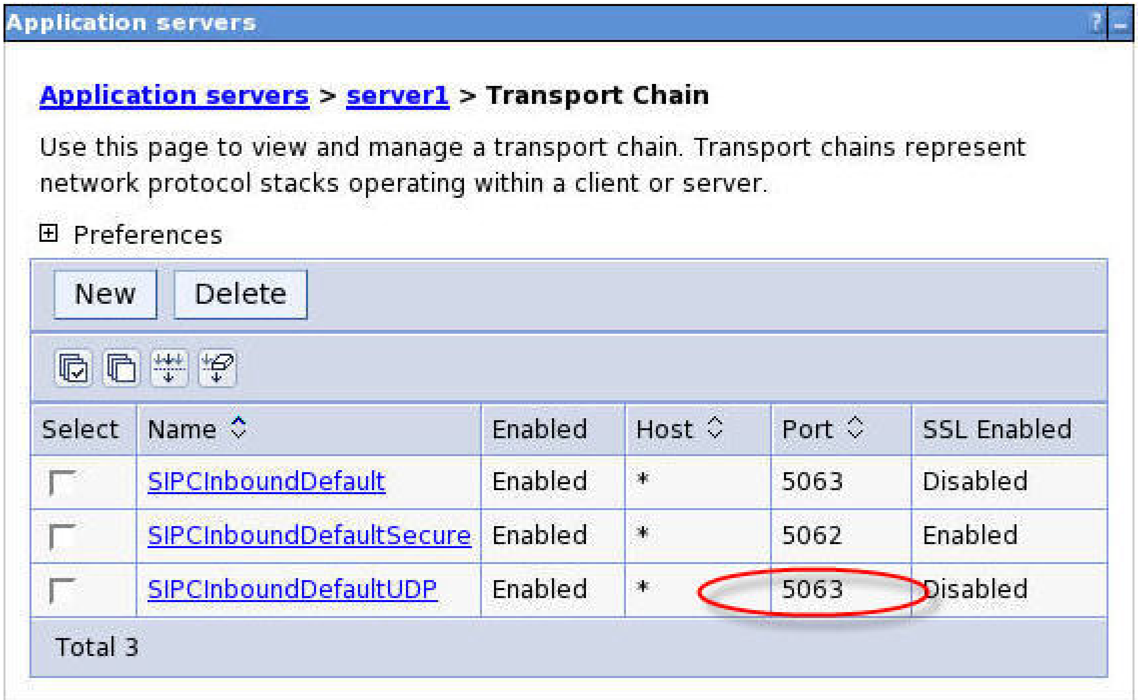
__ 8.1.1.1. Applications → Enterprise Applications.

__ 8.1.1.2. Select the Enterprise Application "SimpleInviteSample_sar," then select the "Start" button. It is possible that the screen will not properly refresh, so even though there is the message that the application started successfully, the icon may not correctly reflect this. Simply select the refresh icon next to Application Status.

__ 8.1.2. Identify the SIP listening port. While the default is 5060, since we potentially have multiple profiles defined we may have a different port assigned.

__ 8.1.2.1. Servers → Application servers → server1 → SIP Container Settings → SIP container transport chains

__ 8.1.2.2. We will be using the default UDP transport for the SIP client. Record the port that the SIPCInboundDefaultUDP transport chain is using, 5063 in our case.



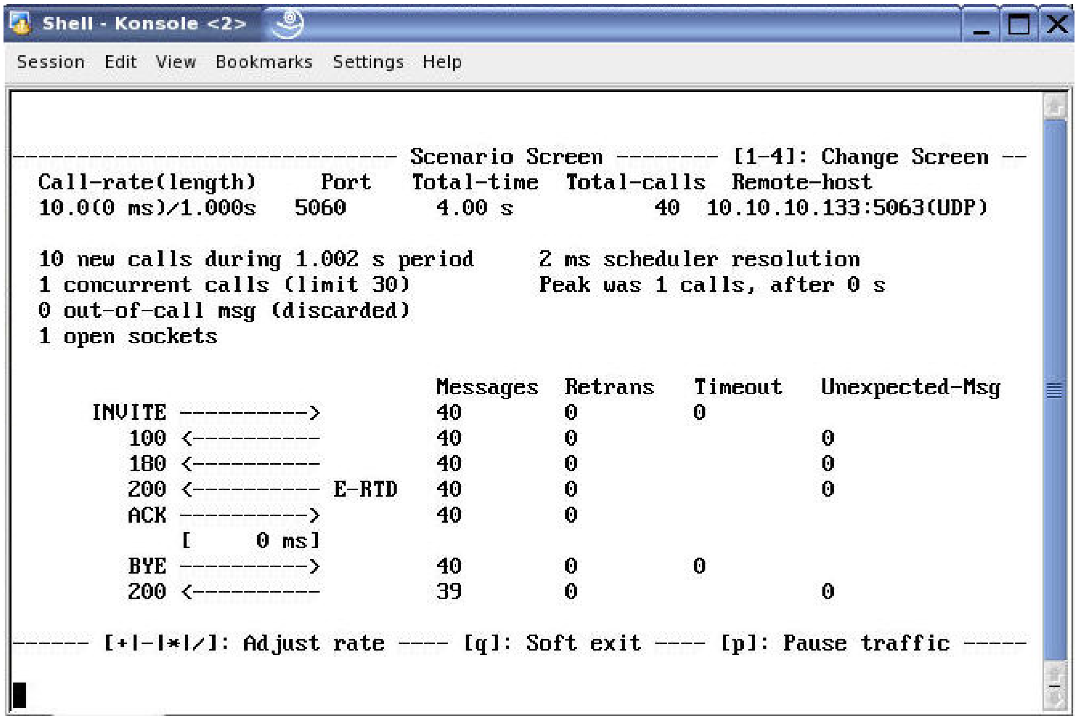
__ 8.2. Execute the SIPp client obtained from sipp.sourceforge.org. This client will generate INVITE and BYE signals at a specified rate to initiate and then tear down sessions. It also provides a dynamic display of the current and summary status of the sessions generated.

__ 8.2.1. If you do not already have a shell window open, select the shell icon to start a shell window (see the Appendix), then enter the subdirectory `/root/sipp`.

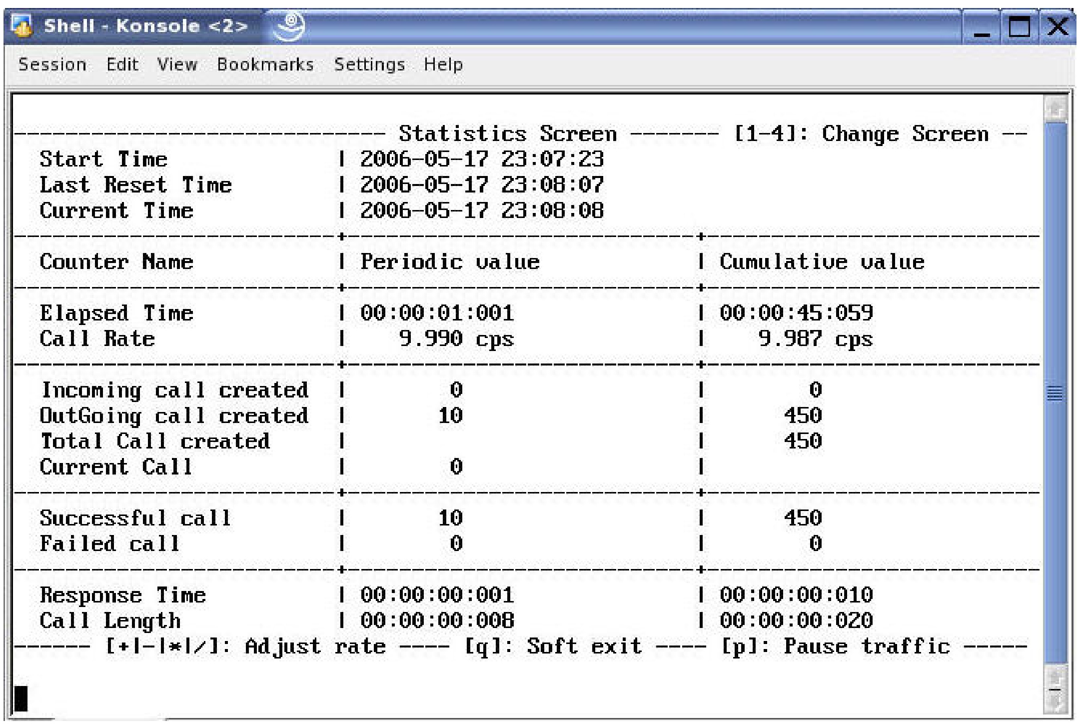
__ 8.2.2. Execute the sip client entering on one line the command, replacing 5063 with the value determined in Part 4, Step 8.1.2.2.

```
./sipp localhost:5063 -sn uac
```

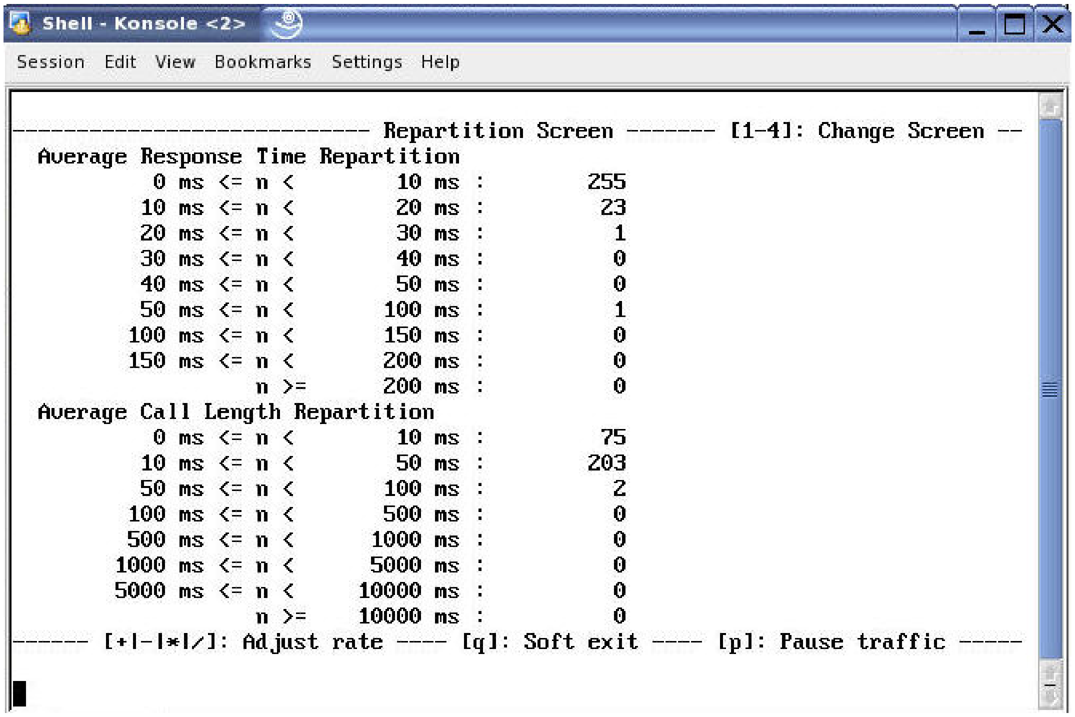
__ 8.2.3. Investigate the 4 screens available in this client by simply entering the number '1' '2' '3' or '4'. To close the client, enter a lowercase 'q', or terminate using control-c. Screen '1' displays the signaling sequence, '2' a summary, and '3' the average response times.



```
Shell - Konsole <2>
Session  Edit  View  Bookmarks  Settings  Help

------------------------------ Scenario Screen -------- [1-4]: Change Screen --
  Call-rate(length)     Port   Total-time  Total-calls  Remote-host
  10.0(0 ms)/1.000s   5060      4.00 s          40  10.10.10.133:5063(UDP)

  10 new calls during 1.002 s period      2 ms scheduler resolution
  1 concurrent calls (limit 30)           Peak was 1 calls, after 0 s
  0 out-of-call msg (discarded)
  1 open sockets

                                 Messages  Retrans   Timeout   Unexpected-Msg
      INVITE ---------->         40        0         0
         100 <----------         40        0                   0
         180 <----------         40        0                   0
         200 <---------- E-RTD   40        0                   0
         ACK ---------->         40        0
             [     0 ms]
         BYE ---------->         40        0         0
         200 <----------         39        0                   0

------ [+|-|*|/]: Adjust rate ---- [q]: Soft exit ---- [p]: Pause traffic -----
```



```
Shell - Konsole <2>
Session  Edit  View  Bookmarks  Settings  Help

----------------------------- Statistics Screen ------- [1-4]: Change Screen --
  Start Time                | 2006-05-17 23:07:23
  Last Reset Time           | 2006-05-17 23:08:07
  Current Time              | 2006-05-17 23:08:08
-------------------------+---------------------------+--------------------------
  Counter Name              | Periodic value            | Cumulative value
-------------------------+---------------------------+--------------------------
  Elapsed Time              | 00:00:01:001              | 00:00:45:059
  Call Rate                 |    9.990 cps              |    9.987 cps
-------------------------+---------------------------+--------------------------
  Incoming call created     |        0                  |        0
  OutGoing call created     |       10                  |      450
  Total Call created        |                           |      450
  Current Call              |        0                  |
-------------------------+---------------------------+--------------------------
  Successful call           |       10                  |      450
  Failed call               |        0                  |        0
-------------------------+---------------------------+--------------------------
  Response Time             | 00:00:00:001              | 00:00:00:010
  Call Length               | 00:00:00:008              | 00:00:00:020
------ [+|-|*|/]: Adjust rate ---- [q]: Soft exit ---- [p]: Pause traffic -----
```



__ 8.2.4. YMMV: Note the sequence of signaling between the client and the server, as indicated by the SIP codes shown on these sample screenshots.

Part 5. Appendix

Open a shell window. Select the icon at the bottom of the desktop that resembles a sea shell.

Select an application from the K menu. The K menu is the list of programs as provided by the KDE desktop. Select the icon at the lower left of the desktop that has the letter ‘N’ inside a circle.

SIP/2.0 Status Codes Subset

```
1xx:  Provisional; request received, continuing to process the request

100   Trying
180   Ringing
181   Call Is Being Forwarded
182   Queued
183   Session Progress

2xx:  Success

200   OK
202   accepted: Used for referrals

3xx:  Redirection
4xx:  Client error
5xx:  Server error
6xx:  Global error
```