Installation and Configuration Guide

iPlanet Trustbase Payment Services

Version 1.0

October 2001

Contents

List of Figures

Introduction

The following chapter discusses all related documents to this guide.

Overall Layout

The manual Covers:

- Introduction to Payment Initiation
- Installing iPlanet Trustbase Payment Services
- Installing The Reference Tools: Bank in a Box, Tooled Up and Buyers Bank
- Configuring Four Corner Certificates and Payment Schemes
- Running the Payment Processing Models
- Developing your Back End Systems using the Payment Initiation Library (CPI) API

Related Documents

The following documents are considered pre-requisites to installing iPlanet Trustbase Payment Services (iTPS)

- Eleanor . iTPS is based on the Eleanor Technical Specification and as such you need to have familiarised yourself with this document.

  `https://www.trustbase.net/era`

  Eleanor Scheme Technical Specification Version 1.0b

  Eleanor Scheme Operating Rules

  Eleanor Scheme Product Guide

  Note This Website requires a Username and password that should have been given to you when you joined the Identrus Scheme

- Identrus Message Specifications. iTPS is based on the IDentrus four corner model and as such four servers configures as identrus Transaction Cordinators (TC) using iPlanet Trustbase Transaction manager (iTTM) are assumed to be up and running

  `http://www.identrus.com`

  Identrus PKI Compliance (IT-PKI)

  Transaction Coordinator requirements (IT-TCFUNC)

  Core messaging specification (IT-TCMPD)

  Certificate Status Check Messaging specification (IT-TCCSC)

  Identrus Digital Signature Messaging System Specification (IT-DSMSSP, ver 2.0).

  Transaction Coordinator Certificate Status Check (CSC) Protocol Definition (IT-TCCSC, ver 2.0b)

  Note In order to access the documents within this website you need a Username and password that should have been given to you when you joined the Identrus Scheme.

- iPlanet Trustbase Transaction Manager (iTTM) documentation itself can be found below:

  http:/ / docs.iplanet.com / docs / manuals / trustbase.html

Chapter 1

Introduction to Payment Initiation

This chapter provides an overview of iPlanet Trustbase Payment Services. It discusses a payment initiation model that is based upon the Identrus framework. It is a set of open specifications that allow the Banks and customer vendors to support implementation of an open network based Payment initiation solution. The chapter looks at some of the key features that go into making payments over an open network such as the Internet. It provides an overview of:

- Payment Initiation Products
- Payment Initiation Schemes
- Payment Initiation Processing Models
- Payment Initiation Reference Components of iPlanet Trustbase Payment Services

Payment Products

Every Payment Scheme includes a number of payment products. Each product has a particular set of features that are made up of a sequence of messages sent between the Buyer, Seller, Seller's Financial Institution and Buyer's Financial Institution.

Kinds of Payment products

Examples of payment products include:

- Payment Order
- Certified Payment Obligation

You need to consult your Payment Scheme specification for more details about which kinds of Payment products are supported by iPlanet Trustbase Payment Services (iTPS).

Payment Order

A Payment order is a revocable, unconditional electronic instruction from the Buyer requesting the Buyer's Bank to execute a credit payment to the Seller on a specific date for a specified amount.

Certified Payment Obligation

Where an Assured Payment is used, the Buyer requests the Buyer's Bank to underwrite (or assure) the payment to the Seller.

Payment Schemes

With the use of a payment scheme, a payment is secured by digitally signing payment instructions, providing authentication, message integrity, non-repudiation and confidentiality. The payment is efficient because parties have pre-established instructions with their banks for payment authorisation, routing and settlement that enables the Buyer to initiate a payment on-line, simultaneously with the purchase transaction, instead of through a separate, off-line step. Additional efficiencies are created through standardised payment processing procedures at the banks. In the case where the Seller requires assurance of payment, the Buyer can electronically request its bank, when initiating payment, to assume the responsibility to pay the Seller. This model does not create a new interbank payment system. It is a new channel or front end to initiate payments on existing, back office payment system.

In the early stages iPlanet Trustbase Payment Services will be supporting one kind of scheme:

Eleanor

You need to consult your Eleanor Documentation as to what payment products will be supported by this scheme.

Payment Processing Models

Depending on the particular e-commerce application different Payment Initiation models may apply. iPlanet Trustbase Payment Services supports the following Payment Initiation models.

Buyer to Buyer's Financial Institution

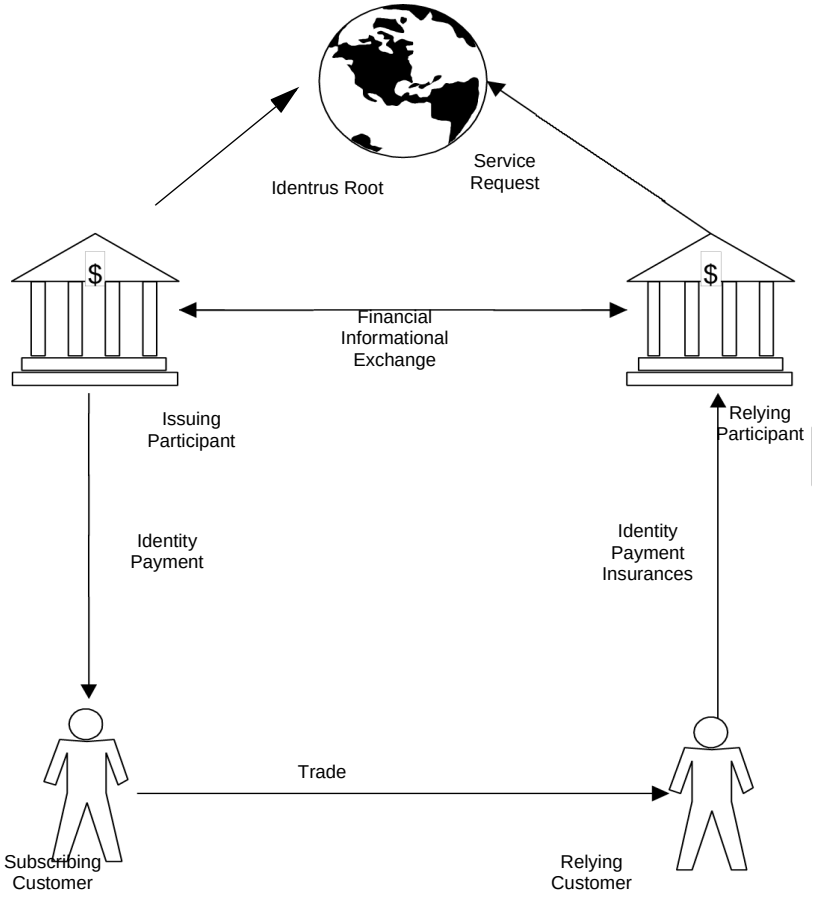
This is where the Buyer initiates a payment directly via the buyers bank. This is also referred to, within The Eleanor Payment Initiation Scheme as the Buyer's Financial Institution Model (BFIM)

Identrus Four Corner Model

The Identrus Four Corner model is utilised to provide enhanced payment initiation services to buyers and sellers. Two trading parties with no previous trading relationship can complete an online purchase or trade and simultaneously arrange for a secure, efficient and, optionally, bank certified payment because there exists the Identrus trust model, which contains pre-established banking relationships between businesses and their respective banks. This is also referred to, within The Eleanor Payment Initiation Scheme as the Sellers Financial Institution Model (SFIM)

The "Four Corner" model, as depicted below, forms the basis of the Identrus PKI network.

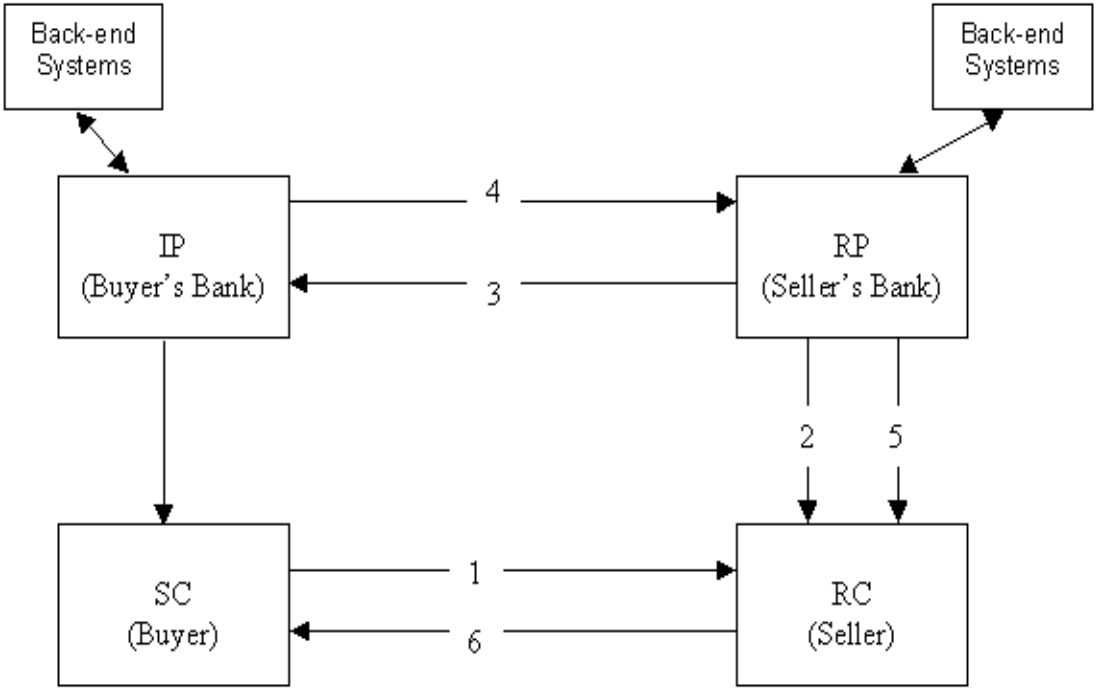
Figure 1-1 Identrus Four-Corner Model



Four Corner Payment Processing

Both the Buyer's Bank and the Seller's Bank needs to be Identrus scheme members. Both the Buyer and the Seller also need to be Identrus enabled by their banks. Within the Eleanor Scheme this is referred to as SFIM

Figure 1-2 Four Corner Payment Model (SFIM)



The message flow is as follows:

1. Buyer sends signed payment information to Seller
   a. The Buyer contacts the seller, and places an order.
   b. The Seller provides some payment information for the Buyer to sign with its certificate, which has been supplied by its Identrus bank (Buyer's Bank)
   c. Buyer signs payment information and sends to the Seller.
2. Seller sends message to Seller's Bank

   a. Seller verifies buyer signature.

   b. Seller creates the product (i.e. Payment Order) selected by the communication between itself and the buyer.

   c. Seller signs the Payment Order message with its certificate, supplied by its Identrus bank.

   d. Sends the message to its bank.

3. Seller's Bank sends message to the Buyer's Bank

   a. Seller's Bank verifies certificate and signature of seller's message.

   b. Seller's Bank informs its legacy systems of the received message.

   c. Seller's Bank finds the location of Buyer's Bank from the Buyer's certificate.

   d. Sends message to the Buyer's Bank.

4. Buyer's Bank sends message to the Seller's Bank

   a. Buyer's Bank verifies the Seller's Bank signature and certificate.

   b. Buyer's Bank verifies the Buyer's signature and certificate and also its authority.

   c. Buyer's Bank informs its legacy systems of the received message.

   d. Sends the appropriate response back to the Seller's Bank

5. Seller's Bank sends response to the Seller

   a. Seller's Bank verifies the Buyer's Bank signature and certificate

   b. Seller's Bank informs its legacy systems of the received response.

   c. Re-signs response

   d. Sends response back to the Seller.

6. Seller informs Buyer of result.

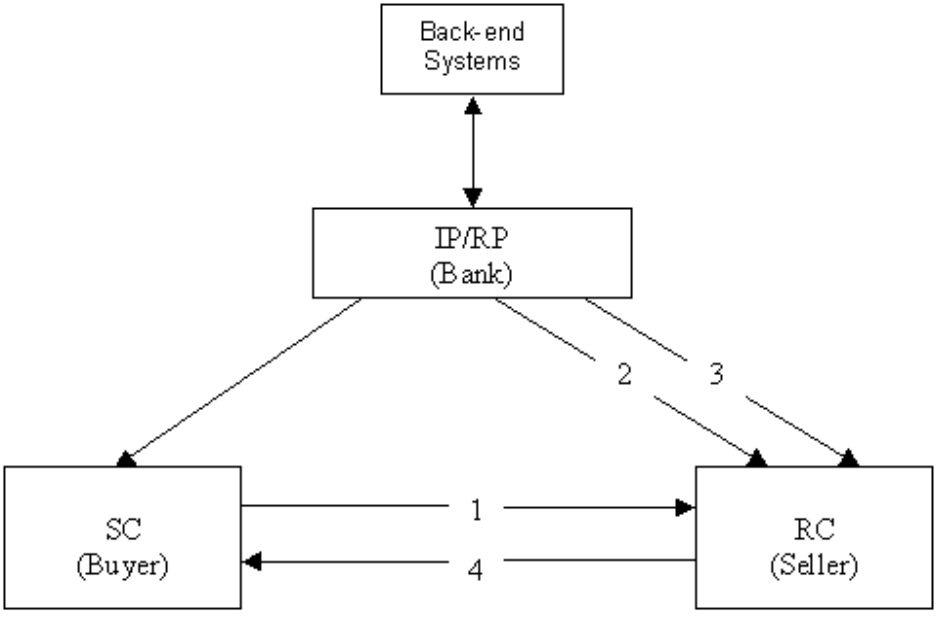
   a. Seller verifies the signature and certificate of its bank

   b. Sends the results of the response message back to the Buyer.

Three Corner Payment Processing

The Three-Corner Model (3CM) is a special case of the SFIM where the Buyer and Seller accounts are held at the same bank. The Buyer's Bank needs to be an Identrus scheme member. Both the Buyer and the Seller needs to be Identrus enabled by their bank.

Figure 1-3 Three Corner Payment Overview



The message flow is as follows:

1. Buyer sends signed payment information to Seller
   a. The Buyer contacts the Seller, and places an order.
   b. The Seller provides some payment information for the buyer to sign with his / her certificate, which has been supplied by its Identrus bank (Buyer's Bank)

   c. Buyer signs payment information and sends to the Seller.

2. Seller sends message to the Bank

   a. Seller verifies buyer signature.

   b. Seller creates the product (i.e. Payment Order) selected by the communication between itself and the buyer.

   c. Seller signs the Payment Order message with its certificate, supplied by its Identrus bank.

   d. Sends the message to its bank.

3. Bank sends response to the Seller

   a. Bank verifies certificate and signature of seller's message.

   b. Bank verifies the Buyer's signature and certificate and also its authority.

   c. Bank informs its legacy systems of the received message.

   d. Sends response back to the Seller.

4. Seller informs Buyer of result.

   a. Seller verifies the signature and certificate of its bank

   b. Sends the results of the response message back to the Buyer.

NOTE More Information about how each payment scheme defines its Models and Payment products can be found at http://www.identrus.com

Examples of supported Schemes include:

Eleanor Payment Reference Specification

Payment Reference Components

In order to initiate a payment around a banking system a number of components are required. The iPlanet Trustbase Payment Services are made up of the following components:

iPlanet Trustbase Payment Services

This comprises of a set of services that are configured within iPlanet Trustbase Transaction Manager and acts as the main banking server to route Payment Messages.

Bank in a Box Back End

Bank in a Box package allows you to test interfacing with legacy systems.

Bank in a Box Admin Tool

In addition to the Back End there is an Admin tool that provides a web interface to the Bank in a Box Back End.

The Seller's Website: Tooledup (SFIM)

This is an example application, in conjunction with the supplied Corporate Payment Initiation Library (CPI) API, that demonstrates how a customer can purchase goods and services through a vendor website that initiates a payment instruction, coupled with the ability to view and cancel a history of payment instructions

Buyers Website (BFIM)

This example, in conjunction with the supplied Corporate Payment Initiation Library (CPI) API, enables a buyer to initiate and cancel payment instructions directly with its bank. This is used when the buyer runs procurement systems or the seller is not a member of the Payment scheme.

Corporate Payment Initiation Library (CPI) API

The Corporate Payment Initiator (CPI) Library API is a Java library providing Eleanor messaging and associated services.

Chapter 2

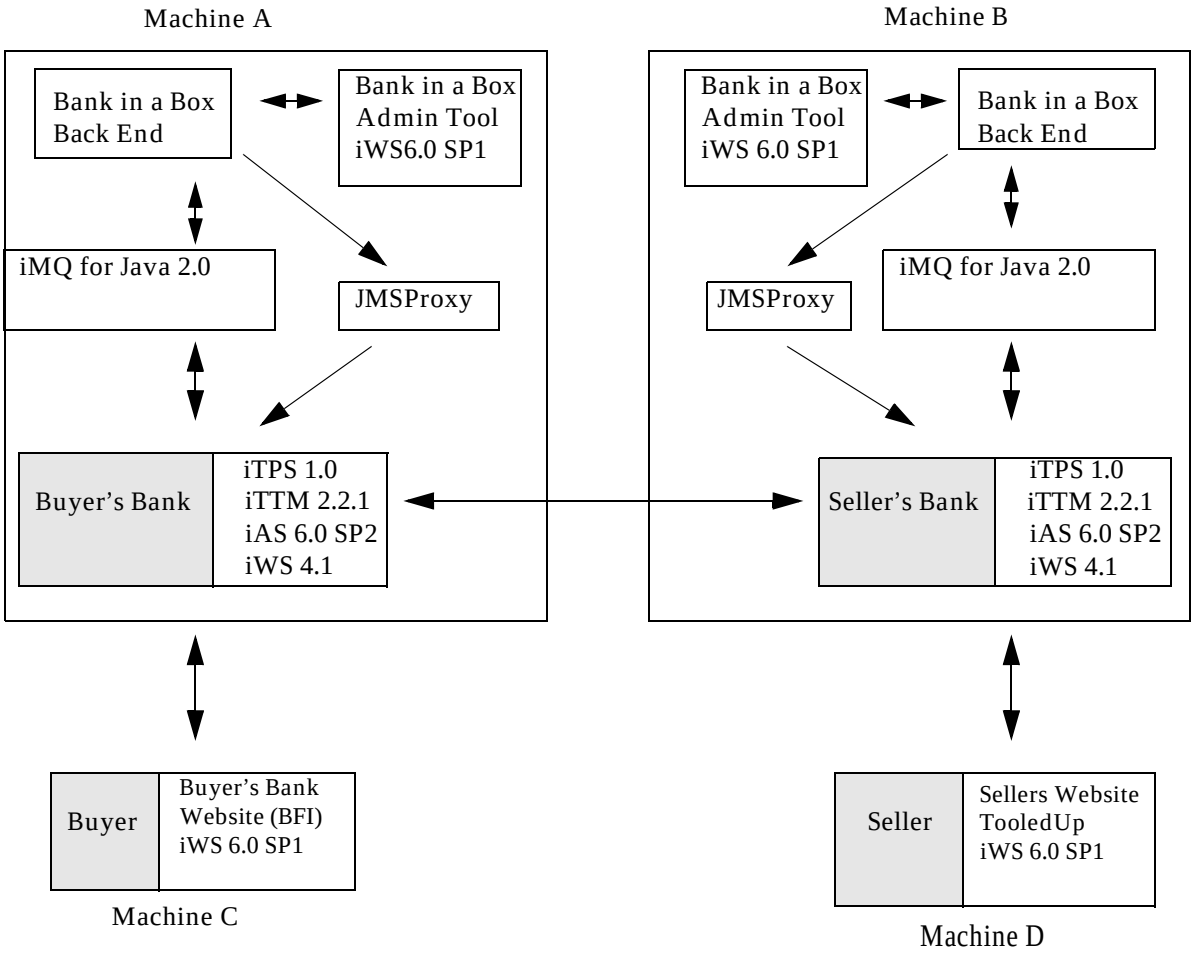
Installation

The following chapter outlines the installation procedures for the various components.

Installation Overview

The diagram below illustrates how the various components are related to each other, and the message paths between each component. In order to have a fully functional system all of these components require installation and configuration.

Figure 2-1 Installation Overview



Although it is not necessary to install the components on individual machines the figure above shows the recommended configuration to avoid unnecessary confusion.

There are a number of main steps that need to be applied appropriately to the four machines labeled Machine A - Machine D in the figure.

1. Install the pre-requisite third party software
    a. An Oracle database must be installed and available for use by all of the machines running in the iTPS installation. An Oracle database may be installed on each node in the system, a single node in the system, or an independent node that is accessed by each of the machines.
    b. Install an Identrus compliant PKI. This must include an appropriate Validation Authority component and be capable of supporting the Identrus Certificate Status Check protocol.
    c. Install an nCipher HSM on each machine in order to perform cryptographic operations
2. Install the base components for the Buyer and Seller's banks
    a. Install the iTTM 2.2.1 on both Machines.
    b. Install the iMQ and its patch on both Machines.
    c. Install the iWS 6.0 for the Bank in a Box administration tools on both machines.
3. Install the components that make up the Payments Services product
    a. Install the iTPS 1.0
    b. Install the JMS Proxy
    c. Install the Bank in a Box (BiaB) back office simulator
    d. Install the Bank in a Box (BiaB) administrator tool
4. Install the Buyer and seller web site components
    a. Install the iWS 6.0 on both machines
    b. Install the Buyers Bank Website (BFI)
    c. Install the Sellers Bank Website (Tooledup demonstrator)
5. Optionally install the CPI library for use in developing applications

Third Party Pre-requisites

Availability

The CD supplied with the product contains all of the required components to install the system EXCEPT:

1. Oracle 8i
2. An appropriate Certificate Authority
3. An appropriate OCSP responder
4. nCipher software

These will need to be acquired from the appropriate vendor, installed and configured, prior to installing any of the iPlanet Payments Services components.

Oracle requirements

Your Oracle installation must be configured with a user capable of :

1. Creating tables
2. Updating tables
3. Dropping tables
4. Running SQL scripts to populate the database

When installing Oracle you will need to allocate sufficient space to the user. We would recommend the following:

- For every 1000 expected messages you will need a minimum of 20Mb of table space.
- The default block size should set to a minimum of 8k

You will be required to provide the details of the Oracle installation at various points during the installation. The information required will be:

1. Hostname - As appropriate
2. Port number - Generally 1521
3. SID - Generally ORCL

The Oracle instance must be available during the installation of the product as most components require the capability to log into the database using SqlPlus and populating tables from information supplied in SQL scripts.

PKI Requirements

Your software must be configured as PKI compliant with Identrus (See Identrus Document IT-PKI http://www.identrus.com) including all Transaction Coordinator profiles.

It is expected that the RA, CA, and VA components are running during the installation as certain components require certificates to be issued.

nCipher requirements

The nCipher components are generally stand alone and little information is required about the nCipher components. It is however useful to know the port that the nCipher Hardserver is running on (Default is 9000) as this is required at some points during installation.

Buyer and Seller Bank base components

iTTM 2.2.1

Each Bank machine will need to have an iTTM installed and configured.

In order to install these components you will need to follow the instructions in the iTTM 2.2.1 installation guide. See, for instance

http://docs.iplanet.com/docs/manuals/trustbase/221/install/contents.htm

or

cdrom/cdrom0/iTTM

The instructions in chapter 1 Pages 13-62 provide information on how to install the following:

1. iWS 4.1
2. iAS 6.0
3. iTTM 2.2.1

It also provides information on how to configure and check that the components are operational.

NOTE: All of the software for the above installation is included on the iTPS CD.

iPlanet Message Queue for Java 2.0

The iPlanet message Queue (iMQ) component provides a means for the iTPS and the Bank in a Box components to communicate with each other. This means that an iMQ installation must be performed on both the Buyers and Sellers bank machines.

iPlanet Message Queue for Java is shipped with iTPS and may be found in the iMQ2.0 sub directory on the CD.

```
cdrom/cdrom0/iTPS/iMQ2.0
```

Installation

The iMQ installation uses the Solaris package mechanisms to install the software on the machine. Assuming that the supplied CD has been mounted on /cdrom then the following commands will install the software:

```
cd cdrom/cdrom0/iTPS/iMQ2.0/imq2_0-pkgs

pkgadd -d ./
```

You will be asked a question during the installation. Unless you have specific installation requirements then by using the defaults provided you will install all of the iMQ packages. These settings will fulfill the iTPS iMQ requirements.

If you require further information then details of how to install iMQ 2.0 can be found in point 7 within the following document that requires vi or Adobe acroreader to read:

```
http://docs.iplanet.com/docs/manuals/javamq/20/install.pdf
```

Example installation and Configuration

```
bash-2.03# uncompress imq2_0-dev-solsparc.tar.Z

bash-2.03# tar  -xvf imq2_0-dev-solsparc.tar

bash-2.03# pkgadd -d imq2_0-pkgs

Select package(s) you wish to process (or 'all' to process all
packages). (default: all) [?,??,q]:
```

Once the iMQ is installed, install the SP1 patch. This process is documented in the file:

```
cdrom/cdrom0/iTPS/iMQ2.0/SP1/111858-01/install.pdf
```

NOTE: The file although containing a .pdf extension is a test file and may be read using the vi editor. Once the software has been installed on either the buyer or seller machine, perform the second installation before progressing to patch the iAS installation.

Configuring with iAS

The next step is to configure the iAS installed as part of the iTTM 2.2.1 installation to use the appropriate iMQ installation. This operation will need to be performed on both of the Buyer and Seller machines. Before performing this operation it is important to ensure that the iAS has been shut down. This can be performed by executing the following scripts:

```
<iTTM install directory>/TTM/Scripts/stoptbase
```

```
<iTTM install directory>/TTM/Scripts/stoptias
```

If the iTTM had been installed in '/opt/TTM' the commands would be:

```
/opt/TTM/Scripts/stoptbase
```

```
/opt/TTM/Scripts/stoptbase
```

To configure iAS for use with iMQ, execute jmssetup. This must be performed as the root user. You will be asked several questions, now illustrated below:

```
bash-2.03# cd /opt/iplanet/ias6/ias/jms/bin

bash-2.03# ./jmssetup

iAS install directory is /opt/iplanet/ias6/ias

Are you using IBM MQ v5.1 as message provider [Y] :n

Enter the dynamic library run path (LD_LIBRARY_PATH) for your JMS
message provider. When finished, hit return only) :

Will append to LD_LIBRARY_PATH? Is this correct? [Y] :

Enter the elmements (absolute path) for the JMS provider
CLASSPATH

When finished, hit return only. :/opt/SUNWjmq/lib/jmq.jar

Enter the elmements (absolute path) for the JMS provider
CLASSPATH

When finished, hit return only. :/opt/SUNWjmq/lib/jmqadmin.jar

Enter the elmements (absolute path) for the JMS provider
CLASSPATH

When finished, hit return only. :

Will append
:/opt/SUNWjmq/lib/jmq.jar:/opt/SUNWjmq/lib/jmqadmin.jar to
CLASSPATH?

Is this correct? [Y] :y
```

Once configured on one machine, configure the second machine before progressing to installing the iTPS components.

At this point there is no need to start the iMQ services. Instructions for starting the iMQ service are shown in Chapter 4.

Installing the iWS 6.0 for BiaB administration

In order to be able to install the Bank in a Box administrator component, a web Server needs to be available. The iTPS CD contains a iWS 6.0 package that is shipped for this use.

Run the iWS6.0 setup tool located in

```
cdrom/cdrom0/iTPS/iWS6.0
```

Selecting the default values for the installation may cause the iWS 6.0 installation to clash with the iWS 4.1 installed for the iTTM 2.2.1. In order to avoid this ensure that the Administration server port and the Web server port are set to values other than 8888 and 80 respectively.

When installing the iWS 6.0 make sure that you select the option that specifies an external JDK 1.2 i.e. /usr/java as the JDK included does not support the BiaB administration tools.

Ensure that a web server is installed on both the Buyer and Seller bank machines prior to moving on to the installation of the iTPS components.

Installing iTPS Components

The iTPS components reside on both the Buyer and Seller bank machines. The following sections describe the installation of these components.

Payments Services installation

Make sure you have installed and configured iPlanet Trustbase Transaction Manager 2.2.1 and iPlanet Message Queue for Java 2.0

1. Make a security back up of your Trustbase directory structure:

```
cp -R <Trustbase_install_directory>/Trustbase \
<Trustbase_install_directory>trustbase.bak
```

This is required because the iTPS install cannot be un-installed, and installing the iTPS more than once on a iTTM installation will not work. If an installation of the iTPS fails for any reason you are advised to restore the backup and start again.

2. Remove the configuration database already installed during the iTTM installation:

   a. At this point the iAS and iTTM components should not be running. Unless they have been started since configuring the iAS for use with iMQ then they will not current be running.

   b. Empty the contents of the configuration table CONFIG from your database. Type the following commands on the machine on which Oracle is installed:

      I. su - oracle

      II. sqlplus

      III. Enter password and User name at the appropriate prompts

      IV. delete from config;

      V. commit; exit;

      When iTTM is recreated again the CONFIG table will be recreated automatically.

3. Run the iPlanet Trustbase Payment Services Installation java class

```
# cd /cdrom/cdrom0/iTPS
# java -classpath . EleanorPaymentsInstaller
```

Figure 2-2 iPlanet Trustbase Payment Services Installation Welcome Screen

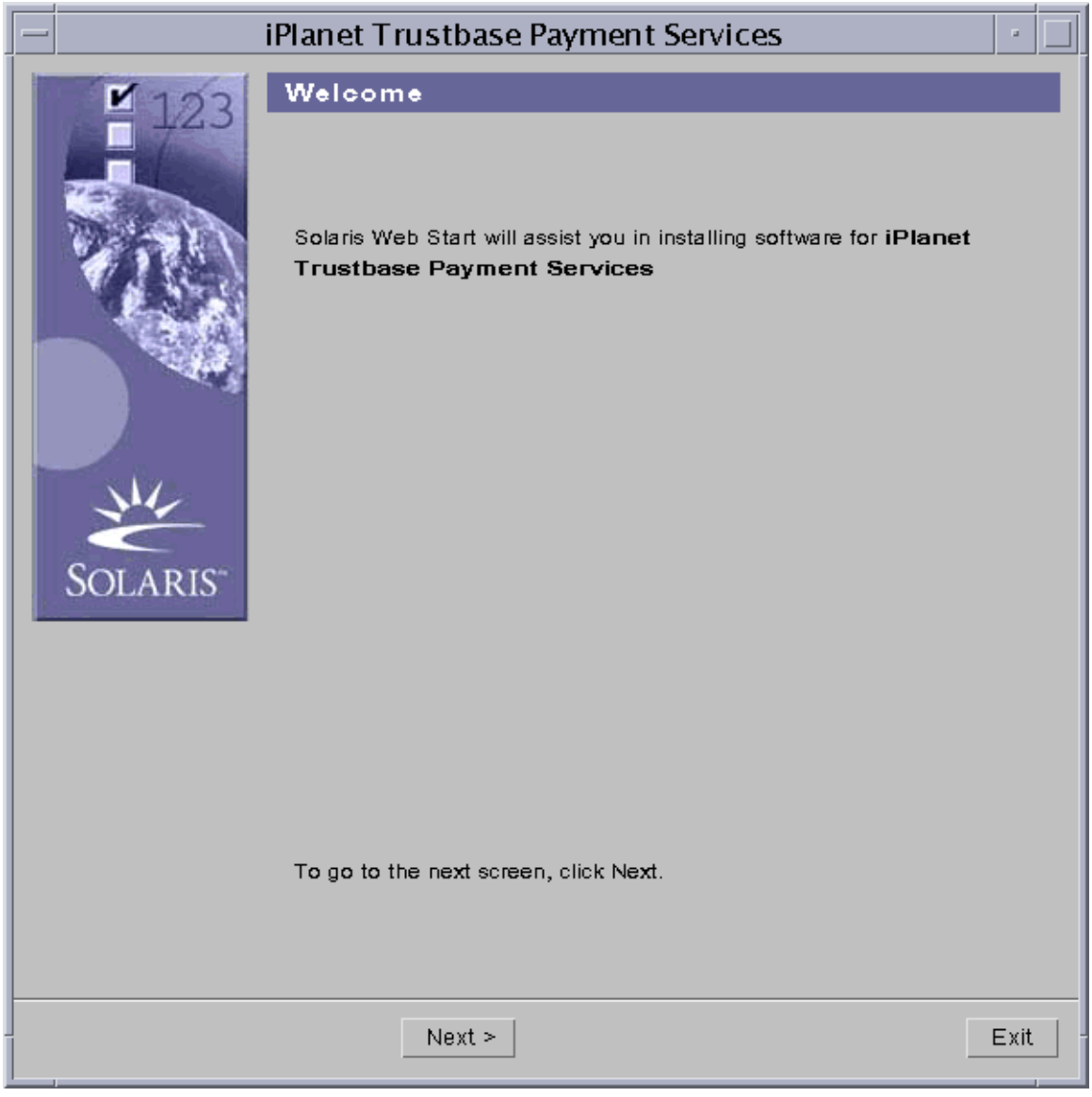


Figure 2-3 Locale Selection

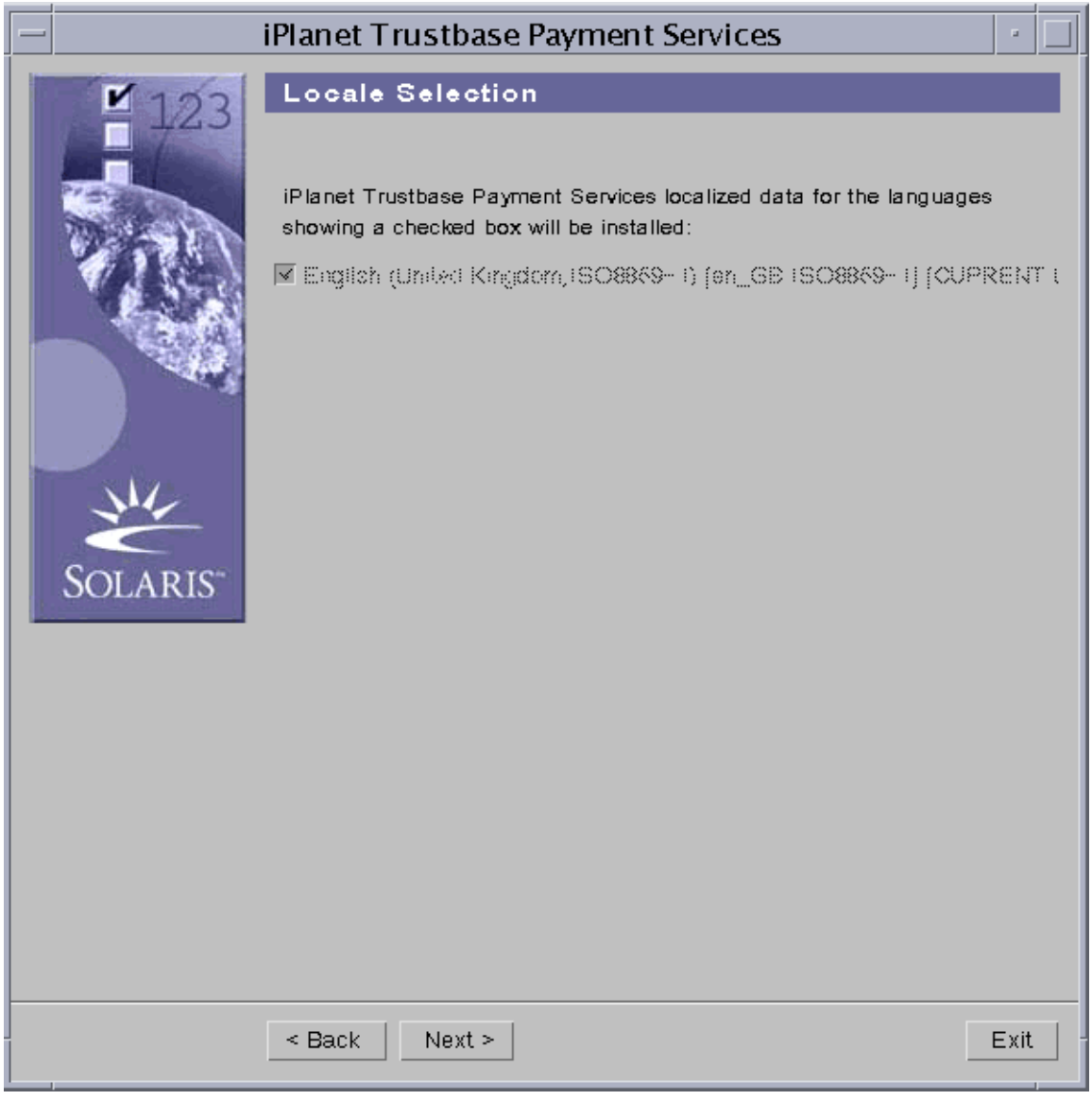


Figure 2-4 iPlanet Trustbase Transaction Manager Installation Directory

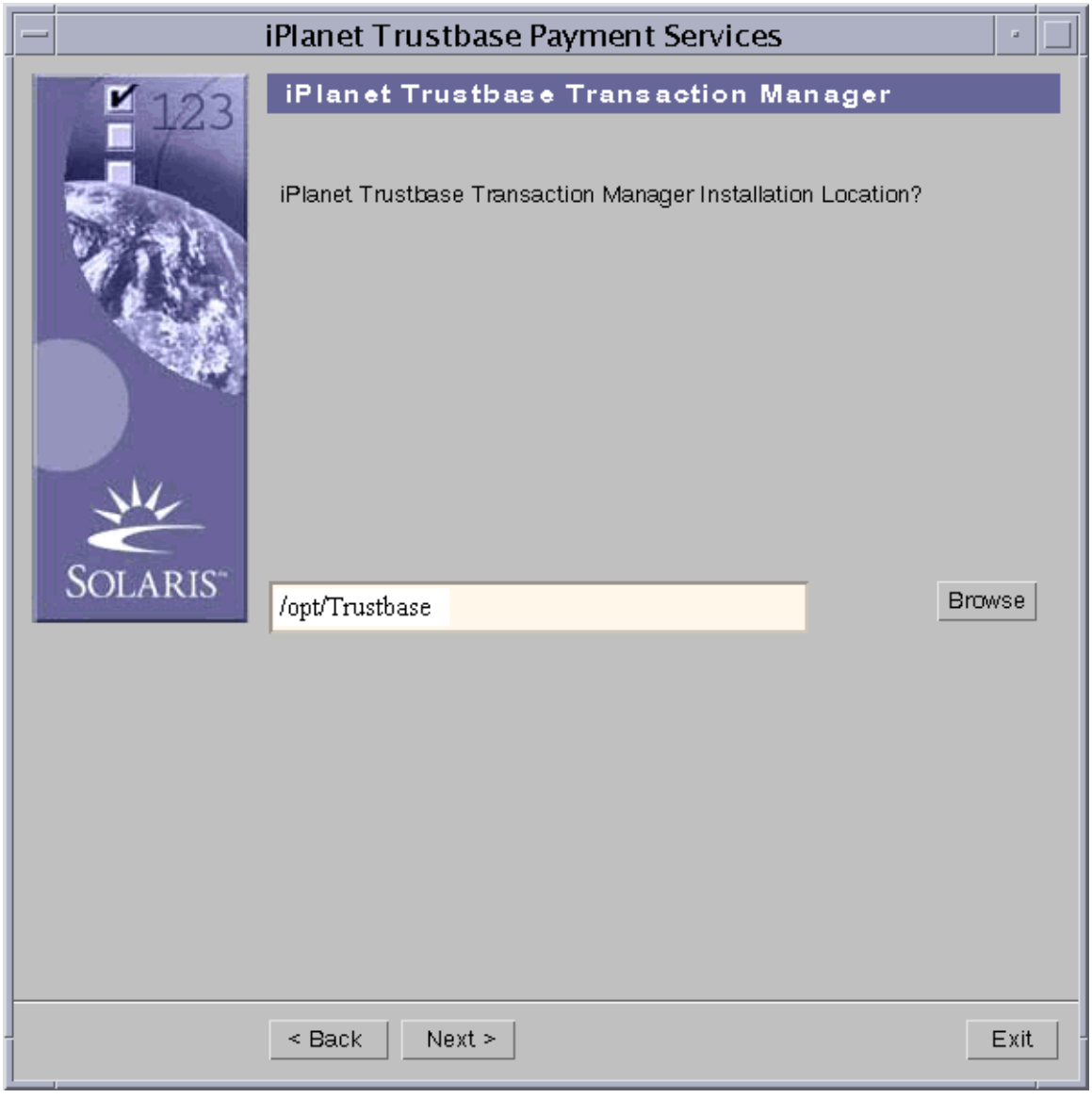
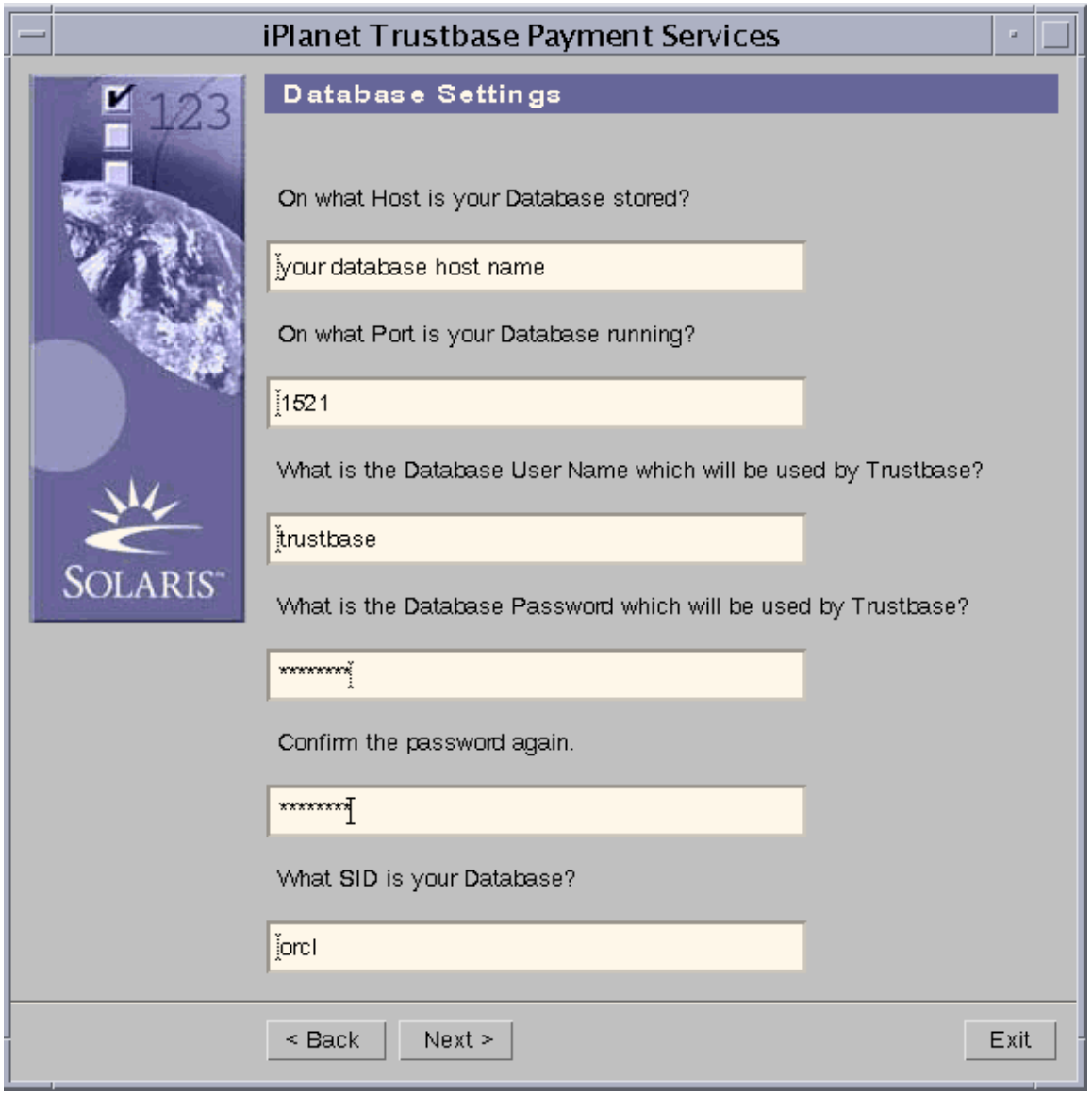


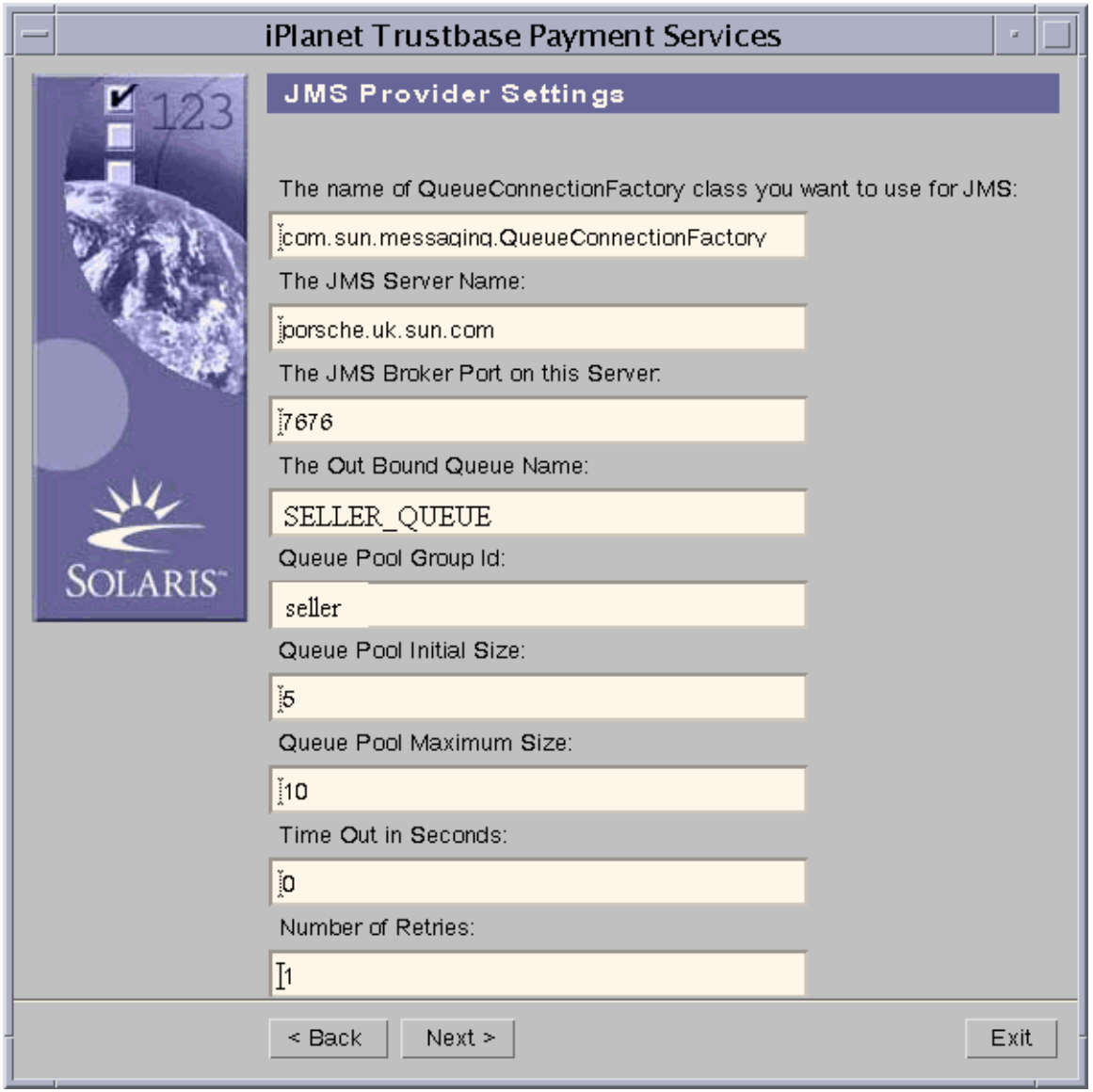
Figure 2-5 Database Settings



The Oracle database being supplied needs to be the database used by the iPlanet Trustbase Transaction Manager software on which iPlanet Trustbase Payment Services plug-in is being installed. The following information is required:

- Oracle login name,
- Oracle login password
- Oracle hostname
- Oracle port number
- Oracle SID.

Figure 2-6 iPlanet Message Queue For Java Settings



Notes: The JMS Broker port default is 7676 unless a non-default installation of iMQ was performed.

The Outbound Queue name is the queue going from the iTPS to BiaB and will need to be recorded for later use. SELLER_QUEUE is a suitable name for this.

The Queue pool group id will need to be recorded for later use. seller is a suitable id for this.

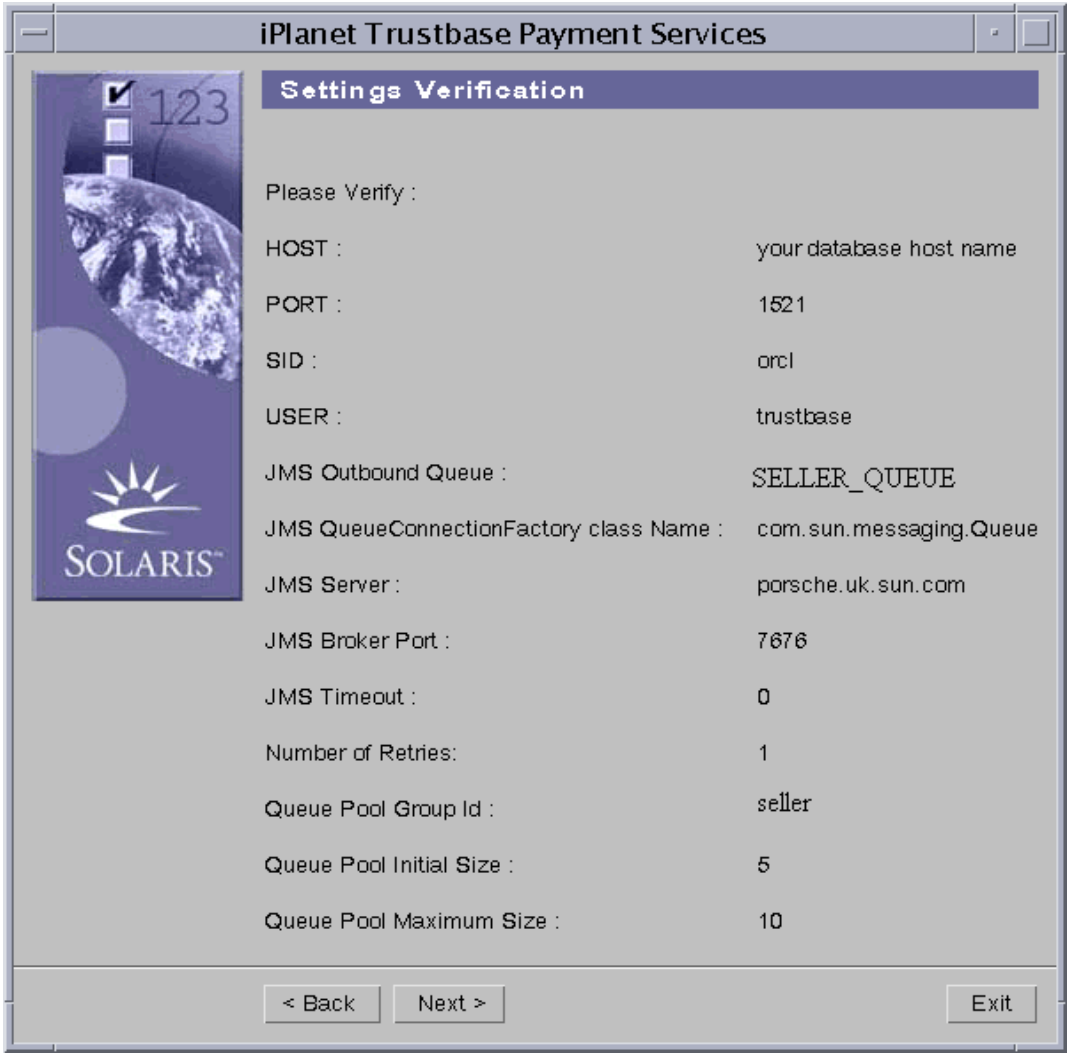
The other defaults provided should be suitable for a standard installation.

Figure 2-7 Payments Mail Settings

Next enter the following as illustrated above.

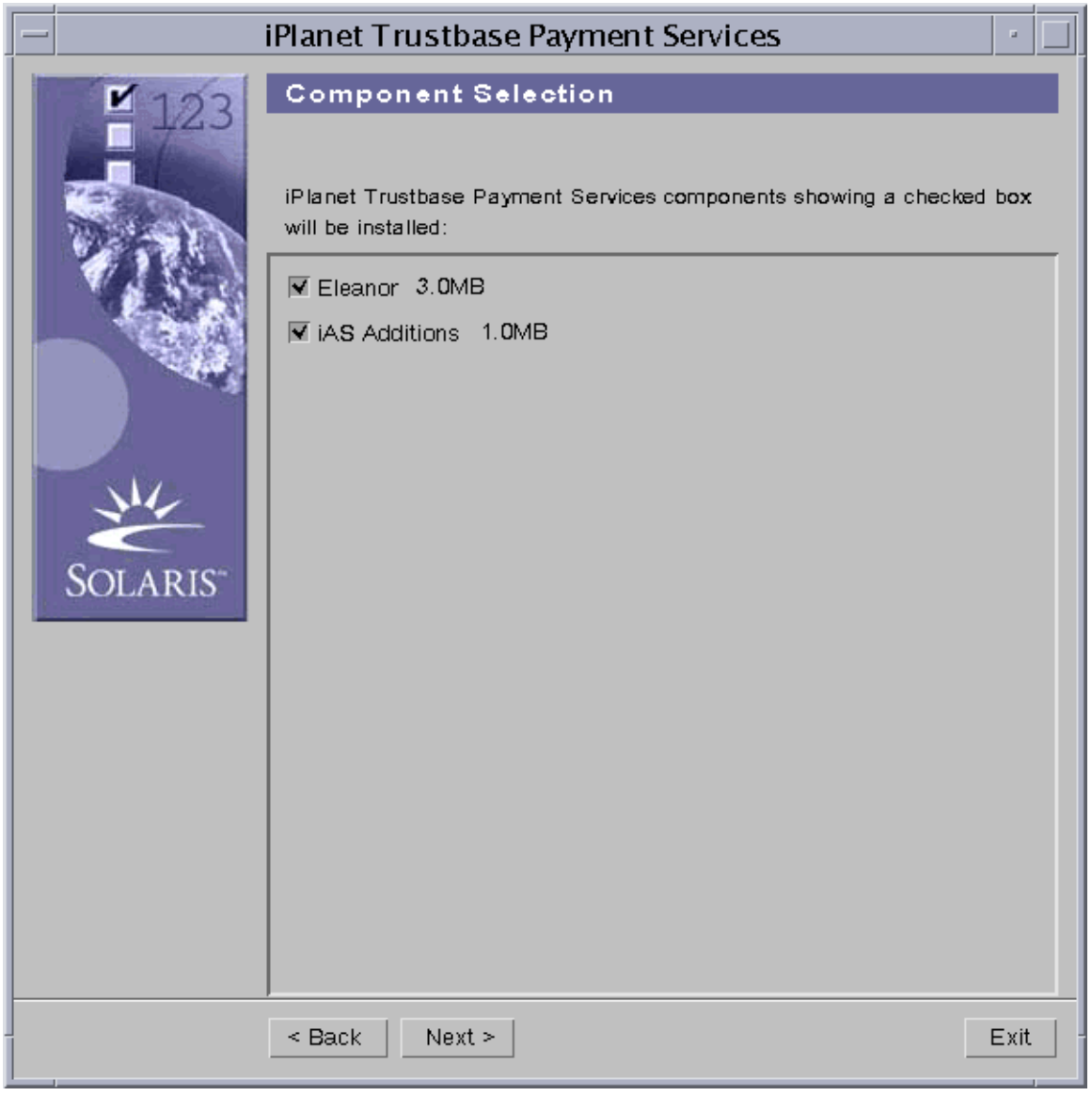
- SMTP host. This is the host where customer email acknowledgements are sent.
- From field. This is the From field of the customer acknowledgement email

Figure 2-8 iPlanet Trustbase Payment Server Verification Panel



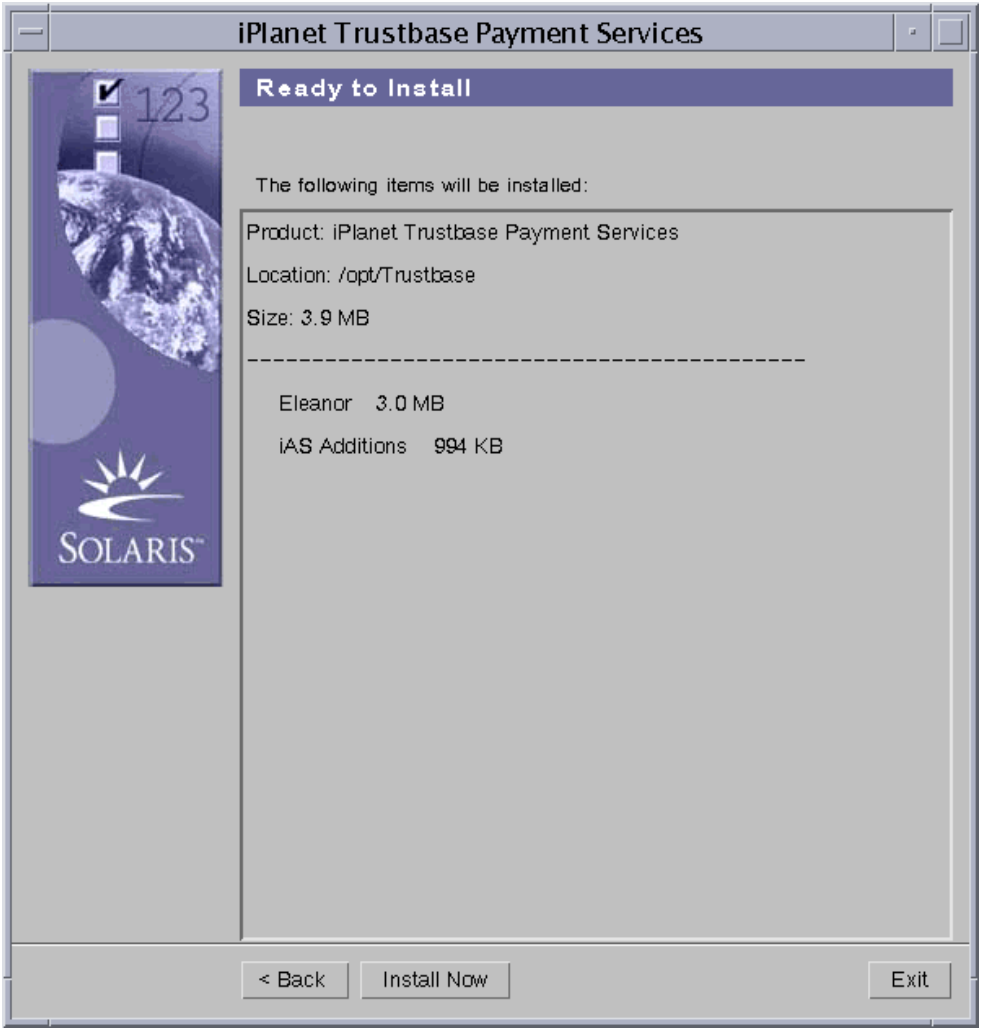
The screen displays the user's choices in order to aid the correct installation. You will need to make a note of the information in this screen as the information is required to install other components later in the process.

Figure 2-9 Component Selection



On entering the screen the size of iPlanet Trustbase Payment Services software application is displayed. In order to install this software the user needs to select the checkbox.

Figure 2-10 Ready to Install



This screen indicates the amount of space that is required to install iPlanet Trustbase Payment Services software. It also indicates the location of the iPlanet Trustbase Transaction Manager system that the iPlanet Trustbase Payment Services plug-in will be installed into.

You should make a note of these locations as they will be required later in the installation process.

Figure 2-11 Updating iPlanet Trustbase Transaction Manager

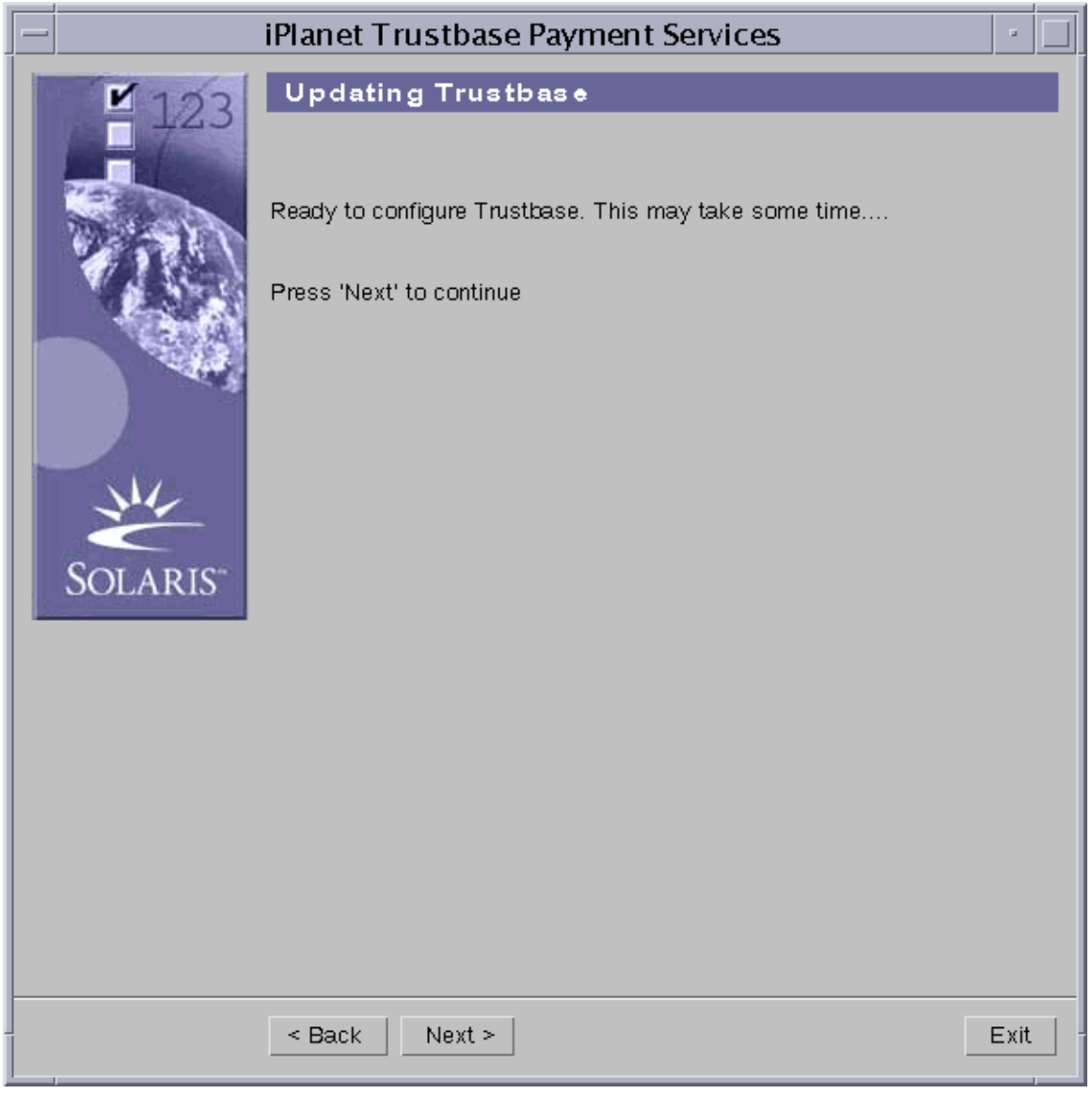
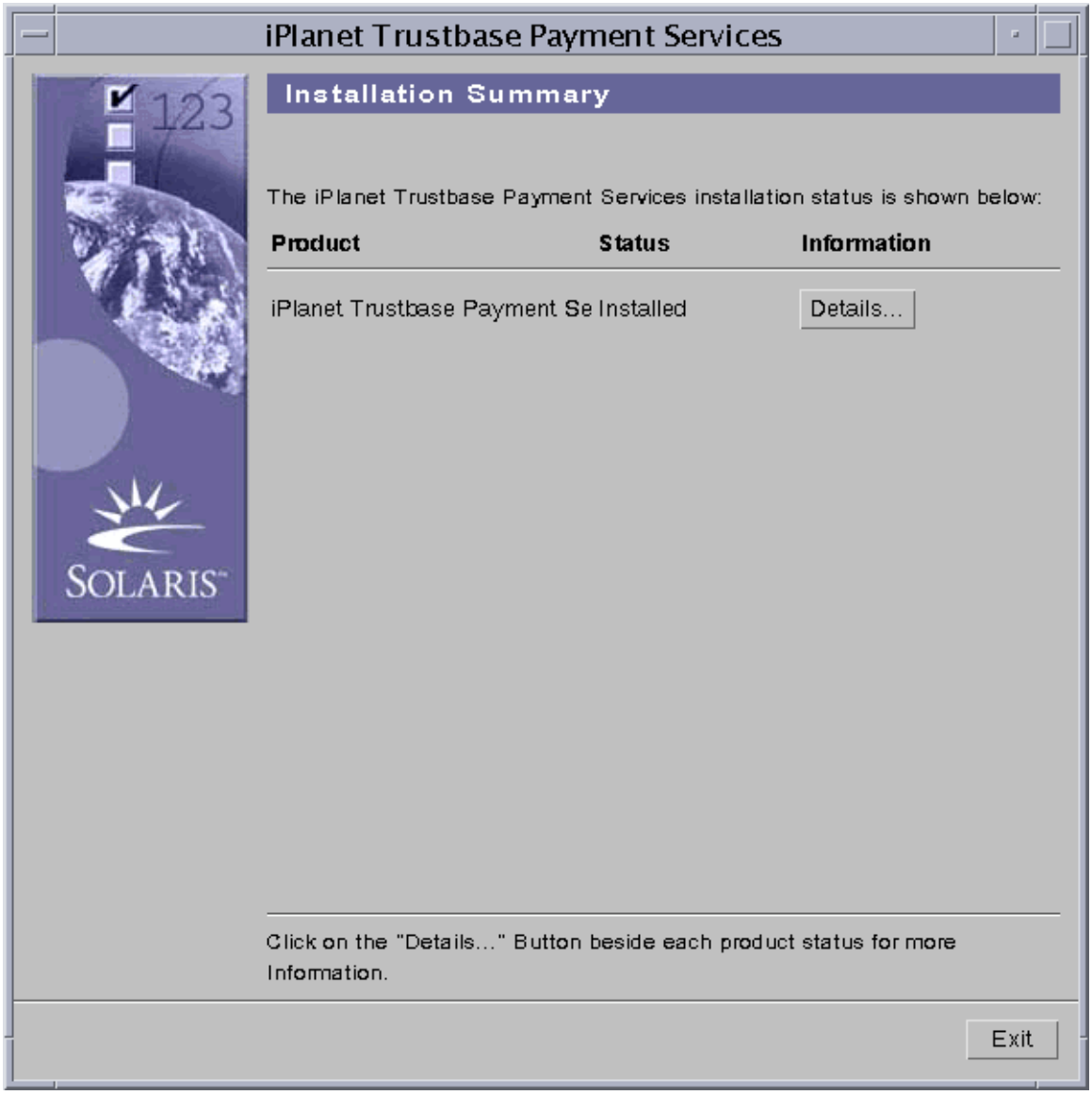


Figure 2-12 Installation Summary



Pressing the details button will display the software installed on the system and alterations to the existing configurations of iPlanet Trustbase Transaction Manager.

Configuring the iTPS database tables

The iTPS Transaction Recovery Process needs to access the subjectDN field of the cert_data table during certificate chain retrieval. The standard install of iTTM 2.2.1 does not store the subjectDN information. A update script is provided with the iTPS that converts the iTTM cert_data table into the necessary format while retaining all the stored certificate information.

This is implemented in the shell script is located in:

```
<iTTM_install_directory>/TTM/Scripts/updateCertDataTable
```

Following the installation of the iTPS.

This script needs to be run once before iTPS is run. It creates a backup of the original cert_data table as cert_data_backup_<timestamp>, adds the subjectDN to the cert_data table and populates it.

Prior to running the script you will need the following information:

- Oracle database username and password
- Database driver class (Usually `oracle.jdbc.driver.OracleDriver`)

The following command runs the script:

```
./updateCertDataTable
```

An example of this is shown below:

```
# ./updateCertDataTable

Enter database connection string (e.g.
jdbc:oracle:thin:user/user@host:1521:orcl):

jdbc:oracle:thin:rainstorm/rainstorm@k9:1521:k9utf8

Enter database driver class (e.g.
oracle.jdbc.driver.OracleDriver):

oracle.jdbc.driver.OracleDriver

Cert count: 1

-----------------------------------------------------------------

Creating backup of cert_data --> cert_data_backup_997350025767

Cert: C=GB,O=Identrus,OU=Identrus Root,CN=Identrus Root CA,
serial: 1, subject: C=GB,O=Identrus,OU=Identrus Root,CN=Identrus
Root CA

Done
```

Note: If this is a new installation and the iTTM has not been used as a Transaction coordinator then there will be a cert count of 0 and the operation will complete almost instantly. The operation will have been successful as the database table columns will have been updated.

This operation needs to be performed on both the Buyer and Seller banks iTTM installation.

Set up iTPS database tables

1. You will now need to run oracle scripts. If Oracle is not installed on the same machine as the iTPS installation then you either have to copy the ./TTM/V2.2/Config/sql directory to the database server or install the Oracle client on the machine.
2. Assuming the sql directory has been copied to the DB server, log on to the database server, su - oracle
3. Change to directory

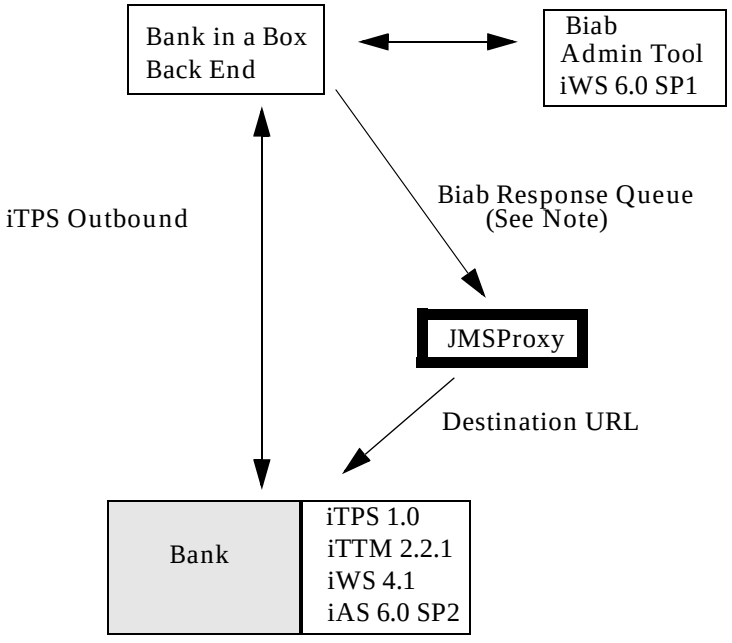
   `<iTTM_install_directory>/TTM/V2.2/Config/sql`
4. Run SQLPlus and enter the username and password
5. Execute the script payments.sql e.g. sqlplus>@payments
6. Exit SQLPlus & the Oracle user.

This will need to be executed on the database(s) used by both the Buyer and Seller banks iTPS installations.

JMS Proxy Installation

The JMS Proxy provides a mechanism for the iTTM to receive inbound messages from an iMQ queue. Messages are taken from the queue and forwarded to iTTM over HTTP. You will need to install a JMS Proxy on both the Buyer and Seller bank machines.

Figure 2-13 Configuring JMS Proxy



Note: This queue is used to send asynchronous response messages from the Bank in a Box to iTPS via the JMS Proxy. The queue name is set as TCQueue/sendName in biabconf.xml and as queue.name in jmsproxy.properties. In order for the JMS Proxy to receive messages on this queue, the queue names used here needs to match.

The JMS proxy is supplied as a compressed archive

```
/cdrom/cdrom0/iTPS/jmsproxy/jmsproxy.tar
```

Extract this file in a suitable location e.g.

```
cp /cdrom/cdrom0/iTPS/jmsproxy/jmsproxy.tar /opt/iplanet
tar -xvpf jmsproxy.tar
```

JMS Proxy Configuration

To configure the server you will need to modify a number of files using the settings mentioned in the previous section.

1. If you have iMQ on your system in the standard location (/opt/SUNWjmq) you will not need to modify the JMQ_DRIVER setting. If the iMQ is not located in the standard location then:

   Modify the script jmsproxy/scripts/jmsproxy such that the JMQ_DRIVER environment variable is pointing to the correct location for the JMQ driver. e.g. /apps/SUNWjmq

2. Modify the following lines in the jmsproxy /config/jmsproxy.properties:

- Destination is the URL to which message content will be forwarded (See figure Figure 2-13)

  ```
  destination=http://hostname/NASApp/NASAdapter/TbaseNASAdapter?Forwarded-by:JMSProxy
  ```

  You will need to change just the hostname component as to an appropriate value e.g.

  http://porsche.UK.Sun.COM/`NASApp/NASAdapter/TbaseNASAdapter?Forwarded-by:JMSProxy`

- queue.host is the hostname of the machine where the JMS broker is listening.

  ```
  queue.host=queue_hostname
  ```

  e.g. queue.host=porsche.UK.Sun.COM

- queue.port The port on which the JMS broker is running by default this will be 7676 unless it was changed during the iMQ installation.

  ```
  queue.port=queue_port
  ```

  ```
  e.g. queue.port=7676
  ```

- queue.name The name of the queue on which to receive messages. This is the asynchronous send queue as specified in the Bank in a Box configuration

  ```
  queue.name=BiabOut
  ```

Note: Make sure the destination URL is the server host name of the appropriate Buyer or Seller bank iTTM installation. Make a note of this URL as you will need this it again when configuring the Bank in a Box components.

Installing Bank in a Box back office simulator

The Bank in a Box (BiaB) back office simulator is designed to create responses to messages received by the iTPS from the buyer and seller web sites. The BiaB must be installed on both the Buyer and Seller Banks servers.

It is not imperative that the iTTM and iTPS are running during installation, and if they have been started following the iMQ proxy installation it is preferable that they are shut down.

In order to install the BiaB on each machine follow the instructions below:

1. Extract a copy the BiaB files from your cdrom to a suitable location e.g.

   ```
   cp cdrom/cdrom0/biab/biab.tar /iplanet
   ```

2. Unpack the tar file
3. To configure the server you will need to modify two files to set certain parameters and run the SQL on the appropriate Oracle database. In order to configure the BiaB follow the instructions below.
4. Run the biab.sql SQL script on the payments database server. This may involve copying the SQL script to the appropriate machine if Oracle is remotely located.

   cd /opt/iplanet/biab/config/sql/

   ```
   biab.sql

   sqlplus username/password

   SQLPlus>@biab

   SQLPlus>exit
   ```

5. Edit the BiaB script so that the environment variables are correct

   ```
   vi biab/scripts/biab
   ```

   a. Modify the script such that the ORACLE_DRIVER and JMQ_DRIVER environment variables are pointing to the correct locations for the oracle driver and JMQ driver respectively.

      Note: You will already have a copy of the ORACLE_DRIVER in the ittm sub-directory e.g.
      <iTTM_install_directory>/TTM/V2.2/Lib3p/10/classes12_01.zip

      Pointing the ORACLE_DRIVER environment variable to this location is an acceptable solution.

b. If you have iMQ on your system in the standard location (/opt/SUNWjmq) you will not need to modify the JMQ_DRIVER setting.

6. The Biabconf.xml file now needs to be modified. The table below identifies the parameters that require modification. The following text is an example illustrating the configuration settings

```
<BiabConfig
responseProcessor="com.iplanet.trustbase.payments.biab.test.Test
ResponseGenerator" threads="10">
    <TCQueue
        host="porsche.UK.Sun.COM"
        port="7676"
        receiveName="SEND_QUEUE"
        sendName="AsyncResponseQueue"
connectionFactory="com.sun.messaging.QueueConnectionFactory"/>
    <AdminQueue
        host="porsche.UK.Sun.COM"
        port="7676"
        receiveName="BiabAdmin"
connectionFactory="com.sun.messaging.QueueConnectionFactory"/>
    <Database
        connectURL="jdbc:oracle:thin:jon/jon@k9:1521:k9"
        driverClass="oracle.jdbc.driver.OracleDriver"
        enableUserTablePrefix="false"/>
</BiabConfig>
```

The actual configuration settings and their use are described in the table below:

Element	Attribute	Description	Requires change?
BiabConfig			
	responseProcessor	The name of a class implementing the ResponseGenerator interface. This object will be used to return a synchronous response to each BackEndMessage received from the iTPS. If this attribute is absent, no synchronous responses will be sent.	See Note
	threads	The number of threads in the thread pool used for servicing both admin and BackEndMessages entering the system	No
TCQueue			
	host	The name of the host where the message queue broker is located	yes
	port	The port on which the message queue broker is listening	yes
	receiveName	The name of the queue on which BackEndMessages will be received. This must be the same as the Send queue that you entered during the iTPS installation.	yes
	sendName	The name of the queue on which **asynchronous** responses will be sent to the iTPS. This is the same queue as specified in the JMS proxy setup in the queue_name parameter.	yes
	connectionFactory	The class name of the queue connection factory	no
AdminQueue			
	host	The name of the host where the message queue broker is located	yes
	port	The port on which the message queue broker is listening	yes
	receiveName	The name of the queue on which Admin messages will be received. This is a unique new queue name that will be used later in the configuration of the BiaB admin tool	yes
	connectionFactory	The class name of the queue connection factory	No
Database			
	connectURL	The URL used to connect to the database	yes
	driverClass	The name of the database driver class	yes
	enableUserTablePrefix	Whether to enable user name mapping in table access. If this is enabled, the database queries will be to tables prefixed with the name of the current user. This is disabled by default.	No

Having installed the BiaB on either the Buyer or Seller Bank machines, install the BiaB on the other machine before moving on to the BiaB administration tool.

Installing Bank in a Box Admin Tool

The BiaB administration tool is a Web application designed to run on the iWS 6.0 Web server set up earlier. A BiaB administrator tool should be installed on both the Buyer and Seller Bank machines that host the iTPS and BiaB components. The BiaB Admin tool web application is located on the BiaB directory.

In order to deploy the Web application you must perform the following:

1. Make sure the IWS_SERVER_HOME environment variable is set to your <server_root> directory. A typical example of this might be

   ```
   IWS_SERVER_HOME=/opt/iws6;export IWS_SERVER_HOME
   ```

2. Make sure that the <server_root>/bin/https/httpadmin/bin directory is in your path.

   ```
   PATH = $PATH:$IWS_SERVER_HOME/bin/https/httpadmin/bin;export
   PATH
   ```

3. Deploy Bank in a Box using the iWS 6.0 web application deployment tool wdeploy. The deployment tool takes a number of parameters:

   `<uri_path>` The URI prefix for the web application. This must be a unique name for the web application for the server it is being deployed to e.g. BiaBAdmin

   `<instance>` The server instance name e.g. porsche.UK.Sun.COM.

   `<vs_id>` The virtual server ID e.g. https-porsche.UK.Sun.COM.

   `<biab_install_directory>` The directory to which the application is deployed. If it doesn't already exist it will be automatically created during deployment. If the directory does exist it needs to be empty.

   ```
   cd cdrom/cdrom0/biab
   ```

   ```
   wdeploy deploy -u <uri_path> -i <instance> -v <vs_id>
   ```

   ```
   -d <biab_install_directory> biab-servlet.war
   ```

   For example,

   ```
   wdeploy deploy -u /BiaBAdmin -i porsche.UK.Sun.COM -v
   https-porsche.UK.Sun.COM -d /web/biab biab-servlet.war
   ```

will deploy the servlet on the porsche.UK.Sun.COM server instance, and will unpack the war file under the directory /web/biab.

4. Once the application is deployed, modify

   `<biab_install_directory>/WEB_INF/classes/queue.properties`

   such that it points to the correct JMQ broker.

5. Copy /opt/SUNWjmq/lib/jmq.jar of the JMS provider into <biab_install_directory>/WEB-INF/lib directory

   in the case of iMQ these files can be found in the host iTPS machine under the following directory

   `<iMQ_install_path>/SUNWjmq/lib`

6. Once the classpath is correct and the queue properties are set, restart the server instance.

7. Once deployed successfully, the Web Site can be accessed from the browser with the following url.

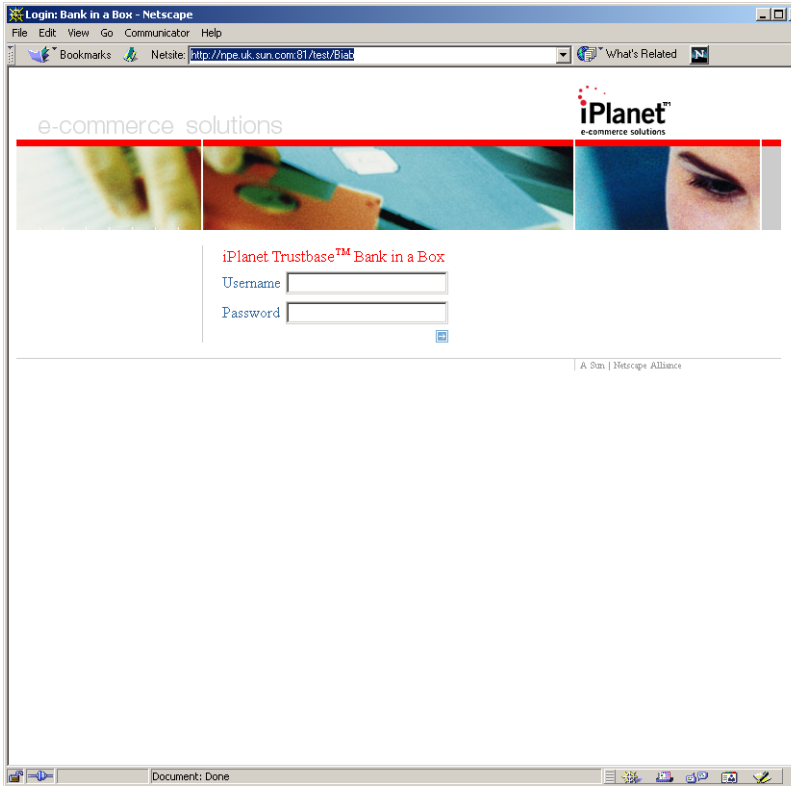
   `http://<hostname>:<port>/<uri_path>/Biab`

   The BiaB admin tool deployed using the previous wdeploy example would be accessed using:

   http://porsche.UK.Sun.COM/`BiaBAdmin`/Biab.

If the server is running and the Web application has deployed successfully the following page will be displayed:

Figure 2-14 Bank in a Box Admin Tool Welcome Screen



Installing the Buyer and Seller websites

The following sections describe how to install the components required to run the Buyer and Seller web sites. These web sites will be used to interact with the Buyer and Seller iTPS components installed previously.

Installing the iWS 6.0

In order to run the web applications that make up the buyer and sellers web sites, a web Server needs to be available on each machine. The iTPS CD contains a iWS 6.0 package that is shipped for this use.

Run the iWS6.0 setup tool located in

```
cdrom/cdrom0/iTPS/iWS6.0
```

Selecting the default values for the installation of the iWS 6.0 should be sufficient for most installations. The only non-standard option you will need to specify is the option that specifies an external JDK 1.2 i.e. /usr/java. This is because the JDK included does not support the buyer and seller web site functionality tools.

Ensure that a web server is installed on both the Buyer and Seller machines prior to moving on to the installation of the Buyer and Seller web applications.

Installing Buyers Bank Website

The bank's web site is archived in to a war file. To install the web site, this war file needs to be deployed on the web server. It can be found on your cdrom as illustrated below

```
cdrom/cdrom0/bfi/bfi.war
```

It does not matter whether iTTM and iTPS are running during installation. However they, and all their associated components such as iAS and iWS, should be running if you need to run this component

1. Make sure the IWS_SERVER_HOME environment variable is set to your <server_root> directory. A typical example of this might be

   ```
   IWS_SERVER_HOME=/opt/iws6;export IWS_SERVER_HOME
   ```

2. Before you can deploy a web application manually, make sure that the <server_root>/bin/https/httpadmin/bin directory is in your path.

   ```
   PATH = $PATH:$IWS_SERVER_HOME/bin/https/httpadmin/bin;export
   PATH
   ```

3. Deploy the war file using following command wdeploy command where:

   <uri_path> is the path name specified while deploying the application.

   `<uri_path>` The URI prefix for the web application.

   `<instance>` The server instance name.

   `<vs_id>` The virtual server ID.

   `<bfi_install_directory>` The directory to which the application is deployed. This directory will be automatically created during deployment, if it doesnt already exist. After deployment, the application will get extracted in this directory. If the directory does exist it needs to be empty.

   ```
   wdeploy deploy -u /<uri_path> -i <instance> -v <vs_id>
   -d <bfi_install_directory> cdrom/cdrom0/bfi/bfi.war
   ```

4. An Oracle JDBC driver needs to be installed in the WEB-INF/lib directory. This will be the same Oracle Driver installed in the Buyer and Seller banks iTTM installations in the lib3p/10 directory. The filename used might be oracle-jdbc-815.zip or classes12_01.zip depending on the version of Oracle you are using. Copy this driver into the WEB-INF/lib directory on the Buyers website machine.

5. Go to the directory

   <bfi_install_directory>/WEB-INF/classes.

   Where <bfi_install_directory> is the directory where the web application is deployed. Open the file bfi.properties and edit the details of the Oracle connect string and the config adapter location to reflect the current installation details.

```
##bfi.properties
driver=oracle.jdbc.driver.OracleDriver
connection=jdbc:oracle:thin:tbase_dbase_user/ \
tbase_dbase_password@tbase_dbase_host:tbase_dbase_port \
:tbase_dbase_sid
ConfigAdapterProperties=
<bfi_install_directory>/WEB-INF/classes/config.properties
```

   The connection string represents the database, where buyer bank's "Bank In a Box" is writing its log. Change the string <bfi_install_directory> with the actual directory name.

6. The Buyers Website needs to communicate with the Buyers Bank. Edit the config.properties file to change the URL to the Buyers Bank iTPS installation.

```
destinationURL=
http://<Buyer_Bank_HostName>/NASApp/NASAdapter/TbaseNASAdapter
```

7. This Buyers Bank application needs a signing certificate chain. This chain must be issued by buyer's bank Certificate Authority in IE5 format.

   The easiest way to create these certificates is to use the Certificate Manager utility supplied with the iTTM 2.2.1 product and described in the iTTM 2.2.1 installation guide. You will need to create a PKCS#10 request for an Identrus compliant End Entity Signing Certificate (Relying Customer Certificate), submit this to the CA that acts for the Buyers Bank, and import the resultant Base64 encoded result. Once you have the certificate, follow the instructions in the utility guide to export the certificate chain in IE5 format.

   Now change <Your_certificate.pfx> with the certificate name.

   ```
   dummySellerCertFileName=
   <bfi_install_directory>/WEB-INF/classes/<Your_certificate.pfx>
   dummySellerCertPassword=password
   ```

After you have finished your changes, you will need to re-start the web server for those changes to take effect.

Installing the Seller's Website TooledUp

The Sellers Website (Tooledup demonstration) is delivered in the form of a tar file called merchant.tar.

Before you can begin to install TooledUp you will need to create a local Certificate Database inside the Webserver for it to use. This certificate database will contain from 3-5 certificates depending on how many roles you assign the certificates to perform, the roles are as follows.

a. Root Certificate or Trust Anchor Certificate (e.g. Identrus Root).

b. Level One Certificate Authority Certificate. (e.g. RP Bank CA)

c. End Entity Signing Certificate (e.g. Signing Certificate e.g. SC from IP Cert) The AIA field within this certificate is used to determine the destination for the payments message)

d. SSL Client Transaction Certificate (e.g. SSL Client Signing Certificate)

e. SSL Server Certificate (e.g. Server-Cert)

To create the certificate databases and import the certificate complete the following steps:

1. Create The Webserver Database

- Access the iWS6 admin server e.g.:

```
./<iws6_install_directory>/startconsole
```

  This will start a browser and allow you to log into the admin server.

- Choose the server to manage and click manage.
- Click on the security tab (it defaults to 'Initialise Trust Database' screen)
- Type in a new password for database and click <ok>. This will create a new database that can only be accessed using the password you have just given so ensure that you do not forget the password!

2. Import The Root Certificate.

- Click the <Install Certificate> Tab.
- Select <Trusted Certificate Authority>, select <message text> and paste in the Base 64 cert from your Root CA
- Click <ok>
- Click <Add Certificate>

3. Import The CA Certificate – Use the same process as Import The Root Certificate (above)
4. Create and import an End Entity Signing Certificate
   - Click the <request certificate> tab.
   - Select <CA URL>
   - Enter “None”
   - Enter “password”
   - Fill in the address details part of the form and press ok.
   - Copy and paste the BASE 64 Request into your Seller Banks CA certificate request form.
   - Retrieve reply from CA and copy the Base 64 cert into the webserver form.
   - Click <Install Certificate.>
   - Select <This Server>, input a name for the cert (e.g. EE Signing Certificate), make a note of the name as you will need it later, Select Message Text and paste in the base 64 cert from the CA.
   - Click <ok>
   - Click <Add Cert>
5. Request, Generate and Import SSL Client Transaction Certificate – Same as for End Entity Signing Cert, but make sure that the name for the certificate is different (e.g. SSL Client Transaction Certificate), and keep a note of the name as you will need it later.
6. Request, Generate and Import SSL Server Certificate – Same as for End Entity Signing Cert except – do not give this certificate a name as the webserver will assign it ‘Server-Cert’.

Now you are ready to install tooledup. You will need several pieces of information which the install script will ask you:

a. The Webserver’s install directory – this is by default */usr/netscape/servers*.

b. The instance name of the webserver you want to install tooledup into. e.g. porsche.UK.Sun.Com

c. The virtual server name of the virtual server you want to install into e.g porsche.UK.Sun.Com

d. The certificate database password.

e. The directory you want to install to.

f. The name of the Signing certificate (the end entity signing certificate - View from the Manage Certificates option in the iws6 Admin Server screen).

g. The name of the SSL Client certificate (view as for Signing Cert).

h. The name of the trust anchor (view as for Signing Cert).

i. The Oracle Database Username (For account where tooledup customer/order details will be stored).

j. The Oracle Database Password.

k. The Oracle Database Machine.

l. The Oracle Database Port.

m. The Oracle Database SID.

Once you have prepared this information you are ready to perform the installation.

Follow the steps below and answer the questions to install the tooledup Seller's Application.

1. umpack the following

```
tar xvf merchant.tar
```

2. cd into the directory

```
<tooledup_install_directory>/merchant/scripts.
```

3. Type ./install to run the install script

4. Answer the questions that are asked by the install script.

5. If the webserver is not running you will get an error saying "Reconfigure Failed" this can be ignored at this stage.

6. Copy the oracle drivers into the directory deployment_dir/WEB-INF/lib

7. Log onto your oracle account and run the script install_merchant_ora.sql

This script can be found in:

```
<tooledup_install_directory>/SQLscripts
```

8. An Oracle JDBC driver needs to be installed in the WEB-INF/lib directory. This will be the same Oracle Driver installed in the Buyer and Seller banks iTTM installations in the lib3p/10 directory. The filename used might be oracle-jdbc-815.zip or classes12_01.zip depending on the version of Oracle you are using. Copy this driver into the WEB-INF/lib directory on the Buyers website machine.

The following is an example transcript console of installing Tooledup

```
# tar xvf merchant.tar

----Truncated text output from the tar command----

# cd merchant/scripts

# ls

acquireparams  install

# ./install

Where is your iPlanet WebServer installation located?

/usr/iplanet/servers/iws6

What is the name of the instance your WebServer instance ?

goblin.uk.sun.com

What is the instance's virtual server called ? [ default ]

https-goblin.uk.sun.com

What is the full path to the directory you wish to deploy the
application to ? [ /usr/iplanet/servers/iws6/deploy ]


What is your keystore password ?

password

What is the nick name of the certificate you wish to sign requests
with? [ Server-Cert ]

End Entity Signing Cert

What is the nick name of the certificate you wish to use in SSL
Client transactions ? [ Server-Cert ]

SSL Client Cert

What is the nick name of the certificate you wish to verify responses
with ?
```

```
Identrus Root CA - Identrus

What is the username of your oracle instance ? [ tooledup ]

gadgets

What is the password for that user of your oracle instance ? [
tooledup ]

{password}

What is the hostname of your oracle instance ? [ goblin ]

windstorm

What is the network port of your oracle instance ? [ 1521 ]


What is the SID of your oracle instance ? [ ORCL ]



These are the parameters that you input

[1] The server location is [ /usr/iplanet/servers/iws6 ]

[2] The server instance is [ goblin.uk.sun.com ]

[3] The virtual server id is [ https-goblin.uk.sun.com ]

[4] The deployment directory [ /usr/iplanet/servers/iws6/deploy ]

[5] The keystore password is [ password ]

[6] The signing certificate nick name is [ End Entity Signing Cert ]

[7] The SSL signing certificate nick name is [ SSL Client Cert ]

[8] The verification certificate nick name is [ Identrus Root CA -
Identrus ]

[9] The oracle user is [ gadgets ]

[10] The oracle password is [ ****** ]

[11] The oracle host is [ windstorm ]

[12] The oracle port is [ 1521 ]

[13] The oracle sid is [ ORCL ]


if these are acceptable hit [0] otherwise hit the number of the
parameter you wish to change or hit [e] to leave the installation
```

```
0
-----------------------------------------------
The directory /usr/iplanet/servers/iws6/deploy does not exist
Do you want to create it ?
-----------------------------------------------
y
-------------------------------
Creating directory
/usr/iplanet/servers/iws6/deploy
-------------------------------
What is your domain name ?
uk.sun.com
domain name - uk.sun.com
host name - goblin
Deploying  web application
Loading  new configuration
Reconfigure failure: server not running

Web application deploy successful
#
```

This installation area now contains several directories and files that are detailed below:

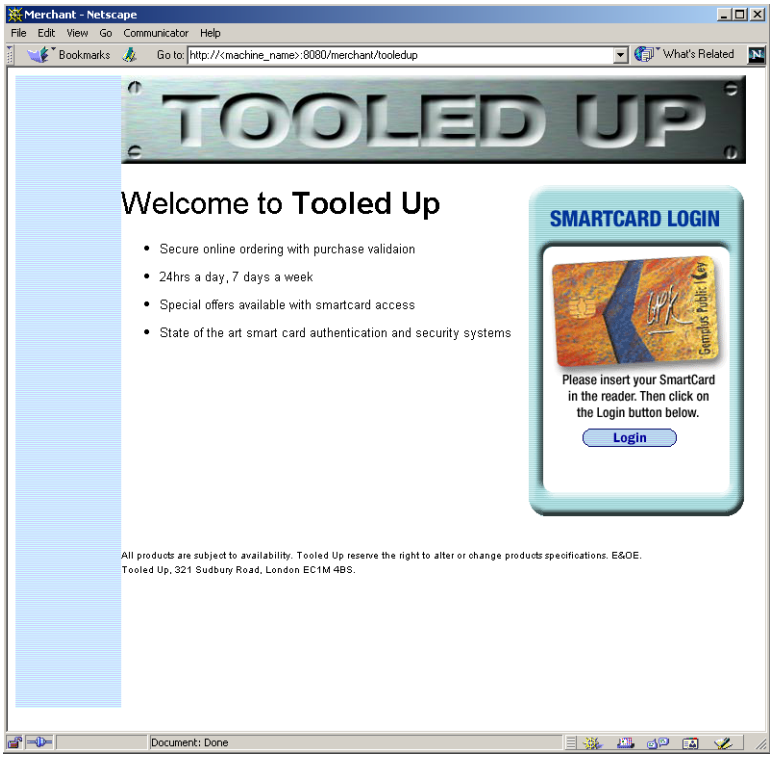
- scripts : This directory contains the install scripts and any data they need.
- SQLscripts : This directory contains the SQL database creation scripts that will create the tables that tooledup needs to run.
- bin : This directory contains the binaries (shared-objects) that tooledup needs to run.
- merchant.war : This is the WAR file that contains the jarfiles and configuration that represent tooledup as an application. This WAR will automatically be deployed by the install script.

In order to use the Tooled up sellers application you will need a SmartCard that will be issued to you by a third party vendor that contains an end entity signing certificate that has been issued by the Sellers Bank CA.

9. Restart the iws6 to be able to access the newly installed web application.You are now ready to run tooledup, access the url tooledup url e.g. http://porsche.UK.Sun.COM/merchant/tooledup

The following screen appears:

Figure 2-15 Sellers Website Tooled Up Welcome Screen



Installing the CPI API

1. The CPI API is delivered in the form of a tar file commonly called

   `cdrom/cdrom0/cpi.tar`

2. This contains several directories and files that are detailed below:

   - <cpi_install_dir>/bin : contains scripts that will set your classpath and help you run the tools you will need. The scripts are all written for use with bourne shell.
   - <cpi_install_dir>/lib : contains all the binaries that the CPI will need to run – this includes shared objects and jarfiles.
   - <cpi_install_dir>/store : This directory will be used to store your TokenKeyStore.
   - <cpi_install_dir>/doc : API documentation and TokenKeyTool detailed documentation.

3. It does not matter whether iTTM and iTPS are running during installation. However they, and all their associated components such as iAS and iWS, should be running if you need to run this component

4. Java 2 Enterpise Edition 1.2 or higher needs to be installed

5. Unpack the file

   `cdrom/cdrom0/cpi.tar`

6. You are now required to use TokenKeyTool. A description of this can be found in

   <cpi_install_dir>`/docs/TokenKeyTool.html`

   By typing help when running TokenKeyTool you can obtain details of how this should be used. To run this script type:

   <cpi_install_directory>/bin/tok.sh

7. Before you can proceed you will need some trusted certificates. These certificates are in files that you have access to and each of the certificate files contain a single PEM format certificate. The certificates that you need will be.

   - C1 : The Identrus Root certificate (In the example below this is called PaymentsRootDevelopment.crt) This is referred to as the verification certificate.
   - C2 : The Buyer CA Certificate.(In the example below this is called StanTheMan.crt)

If you want to cause 4 corner activity you will also need.

- C3 : The Seller CA Certificate

Finally you will need to issue a request for a signing certificate and import the appropriate response into your CertStore. In the example provided the Buyer and Seller signing Certificates are the same

- C4 : The Signing Certificate

8. In order to create your store the following steps need to be performed:

   a. Run the tok.sh script that starts the tokenkeytool.

   b. Type help to obtain details of useage

   c. Create A Trust Domain using openstoremanager command eg openstoremanager –domainspace "file:///install_dir/store" –manager local.

   d. Create a TokenKeyStore using the createstore command eg createstore –store identrus (you will be prompted to give a password - please remember this password).

   e. Import your trusted CA Certificate file using the command importtrustedcerts eg importtrustedcerts –file "filename" (Note the quoting).

   f. Generate a holding key pair for your SellerCertificate using the command genkey eg genkey –dname "CN=CPI Test Cert" (Note the quoting).

   g. View the key to acquire the generated alias for it using the command listkeys eg listkeys.

   h. Request a certificate from your Seller CA using the command certreq eg certreq –alias <generated_key_alias> –dname "CN=CPI Test Cert" –file "/tmp/certrequest" (Note the quoting).

   i. paste the generated Certreq into your CA and get the CA generated Base64 Certificate chain. Store it in a file called "certresponse"

   j. Import the certificate into the database using the command importkeychain –file "/tmp/certresponse" (Note The quoting).

   k. Quit the TokenKeyTool using the command quit.

9. We now illustrate this with an example

```
Script started on Mon 24 Sep 2001 17:01:34 BST
ragnarok# ./tok.sh
```

```
TokenKeyTool> openstoremanager -domainspace
"file:///iplanet/CPITest/store" -manager local

TokenKeyTool> createstore -store identrus

Login to JSS token Internal Key Storage Token: password

TokenKeyTool> importtrustedcerts -file
"/iplanet/CPITest/store/PaymentsRootDevelopment.crt"TokenKeyTool
> importtrustedcerts -file
"/iplanet/CPITest/store/StanTheManCA.crt"

TokenKeyTool> genkey -dname "CN=CPI Test Cert"TokenKeyTool>
listkeys

+KeyEntrys

  +KeyEntry

    subject name: CN=CPI Test Cert

    issuer name: CN=CPI Test Cert

    serial #: 0x7733ad362cc3ecce

    +aliases

      alias: 7733ad362cc3ecce#CN=CPI Test Cert

      +certificate chain

        +certificate [0]

        subjectName: CN=CPI Test Cert

        issuerName: CN=CPI Test Cert

        serial#: 0x7733ad362cc3ecce

        not before: 24-Sep-01 16:03:20

        not after: 24-Sep-02 16:03:20

TokenKeyTool> certreq -alias "7733ad362cc3ecce#CN=CPI Test Cert"
-dname "CN=CPI Test Cert" -file
"/iplanet/CPITest/store/requestfile"TokenKeyTool> importkeychain
-file "/iplanet/CPITest/store/responsefile"

+KeyEntry

  subject name: CN=CPI Test Cert

  issuer name: CN=StanTheMan L1CA,OU=Trustbase,O=iPlanet,C=GB

  serial #: 0x10a

  +aliases

    alias: 10a#CN=StanTheMan L1CA,OU=Trustbase,O=iPlanet,C=GB
```

```
    +certificate chain
      +certificate [0]
        subjectName: CN=CPI Test Cert
        issuerName: CN=StanTheMan
L1CA,OU=Trustbase,O=iPlanet,C=GB
        serial#: 0x10a
        not before: 24-Sep-01 16:09:23
        not after: 19-Sep-02 08:23:24
      +certificate [1]
        subjectName: CN=StanTheMan
L1CA,OU=Trustbase,O=iPlanet,C=GB
        issuerName: CN=Payments Root,OU=Payments
Services,O=iPlanet,C=GB
        serial#: 0x18
        not before: 19-Sep-01 08:23:24
        not after: 19-Sep-02 08:23:24
      +certificate [2]
        subjectName: CN=Payments Root,OU=Payments
Services,O=iPlanet,C=GB
        issuerName: CN=Payments Root,OU=Payments
Services,O=iPlanet,C=GB
        serial#: 0x1
        not before: 29-Aug-01 00:00:00
        not after: 29-Aug-03 00:00:00

TokenKeyTool> listkeys
+KeyEntrys
  +KeyEntry
    subject name: CN=CPI Test Cert
    issuer name: CN=StanTheMan L1CA,OU=Trustbase,O=iPlanet,C=GB
    serial #: 0x10a
    +aliases
```

```
      alias: 10a#CN=StanTheMan L1CA,OU=Trustbase,O=iPlanet,C=GB

      +certificate chain

        +certificate [0]

        subjectName: CN=CPI Test Cert

        issuerName: CN=StanTheMan
L1CA,OU=Trustbase,O=iPlanet,C=GB

        serial#: 0x10a

        not before: 24-Sep-01 16:09:23

        not after: 19-Sep-02 08:23:24

        +certificate [1]

        subjectName: CN=StanTheMan
L1CA,OU=Trustbase,O=iPlanet,C=GB

        issuerName: CN=Payments Root,OU=Payments
Services,O=iPlanet,C=GB

        serial#: 0x18

        not before: 19-Sep-01 08:23:24

        not after: 19-Sep-02 08:23:24

        +certificate [2]

        subjectName: CN=Payments Root,OU=Payments
Services,O=iPlanet,C=GB

        issuerName: CN=Payments Root,OU=Payments
Services,O=iPlanet,C=GB

        serial#: 0x1

        not before: 29-Aug-01 00:00:00

        not after: 29-Aug-03 00:00:00

TokenKeyTool> listcerts

+TrustedCertificateEntrys

  +TrustedCertificateEntry

    +aliases

      alias: 1#CN=Payments Root,OU=Payments
Services,O=iPlanet,C=GB

    +certificate
```

```
      subjectName: CN=Payments Root,OU=Payments
Services,O=iPlanet,C=GB

      issuerName: CN=Payments Root,OU=Payments
Services,O=iPlanet,C=GB

      serial#: 0x1

      not before: 29-Aug-01 00:00:00

      not after: 29-Aug-03 00:00:00

  +TrustedCertificateEntry

    +aliases

      alias: 18#CN=Payments Root,OU=Payments
Services,O=iPlanet,C=GB

    +certificate

      subjectName: CN=StanTheMan L1CA,OU=Trustbase,O=iPlanet,C=GB

      issuerName: CN=Payments Root,OU=Payments
Services,O=iPlanet,C=GB

      serial#: 0x18

      not before: 19-Sep-01 08:23:24

      not after: 19-Sep-02 08:23:24

TokenKeyTool> quit

ragnarok# exit

ragnarok#

script done on Mon 24 Sep 2001 17:12:28 BST
```

10. Now you are ready to run the test harness – you can alter the script called test.sh in the same directory as tok.sh. These can be found in the directory:

    <cpi_install_directory>/cpi/scripts

    The test.sh script has parameters for what certificates need to be used. The parameters it expects are as follows.

    a. Payment amount.

    b. Payment currency

    c. Payment date.

    d. Payment account

    e. Payment reference

    f. Keystore domainspace+store eg file:/ / /<cpi_install_dir>/store#identrus

    g. Keystore password

    h. Verification certificate alias (i.e. The Identrus Root)

    i. Seller signing certificate alias (i.e. The signing certificate)

    j. Buyer signing certificate alias (i.e. The signing certificate)

11. You will need to change the settings for parameters g, h, i and j.

12. Once you have completed that you need to run the test program and receive a response from your TC. It looks something like the example below.

```
Script started on Mon 24 Sep 2001 17:30:38 BST

ragnarok# ./test.sh

Init Seller [ password ] [ file:///iplanet/CPITest/store#identrus ] [ 10a#CN=StanTheMan
L1CA,OU=Trustbase,O=iPlanet,C=GB ] [ 1#CN=Payments Root,OU=Payments
Services,O=iPlanet,C=GB ]

Init Buyer [ password ] [ file:///iplanet/CPITest/store#identrus ] [ 10a#CN=StanTheMan
L1CA,OU=Trustbase,O=iPlanet,C=GB ] [ 1#CN=Payments Root,OU=Payments
Services,O=iPlanet,C=GB ]

**********

CN=StanTheMan L1CA;

OU=Trustbase;

O=iPlanet;

C=GB

**********

CN=StanTheMan L1CA;

OU=Trustbase;

O=iPlanet;

C=GB

**********

CN=StanTheMan L1CA;

OU=Trustbase;

O=iPlanet;

C=GB

**********

CN=StanTheMan L1CA;

OU=Trustbase;

O=iPlanet;

C=GB

**********

CN=StanTheMan L1CA;

OU=Trustbase;

O=iPlanet;

C=GB

*** Hostname: stantheman.uk.sun.com

    ----------------------------------------------------------------
```

```
RESPONSE BEGIN

<?xml version="1.0" encoding="UTF-8"?><!DOCTYPE Acknowledgement
PUBLIC "-//IDENTRUS//ELEANOR_ACKNOWLEDGEMENT_DTD//en"
"file:///bankInterface.dtd"><Acknowledgement><NIB
id="NIB_88A06FA2E96D7490EF266A99F2EAE093A22E788E_1"
version="2.0"><ContextInfo
msggrpid="0C23BFB09A79CBB61E40E33806AAA787AA8D697A"
msgid="SFI01"></ContextInfo><StartTime><LocalTime
id="LocalTime_88A06FA2E96D7490EF266A99F2EAE093A22E788E_1"
time="20010924163046Z"/></StartTime><MsgTime><LocalTime
id="LocalTime_88A06FA2E96D7490EF266A99F2EAE093A22E788E_2"
time="20010924162955Z"/></MsgTime></NIB><Signature
xmlns="http://www.w3.org/2000/02/xmldsig#"><SignedInfo><Canonicaliz
ationMethod
Algorithm="http://search.ietf.org/internet-drafts/draft-ietf-trade-
hiroshi-dom-hash-03.txt"></CanonicalizationMethod><SignatureMethod
Algorithm="http://www.w3.org/2000/02/xmldsig#rsa-sha1"></SignatureM
ethod><Reference
URI="#NIB_88A06FA2E96D7490EF266A99F2EAE093A22E788E_1"><Transforms><
Transform
Algorithm="http://search.ietf.org/internet-drafts/draft-ietf-trade-
hiroshi-dom-hash-03.txt"></Transform></Transforms><DigestMethod
Algorithm="http://www.w3.org/2000/02/xmldsig#sha1"></DigestMethod><
DigestValue>D/BnXyA+JgY60Nq3hn7lxNNJlKE=</DigestValue></Reference><
Reference
URI="#ContentAcknowledgement_E9019A7CF47FD5037FC6D43EDE1E08FD202981
D8_1"><Transforms><Transform
Algorithm="http://search.ietf.org/internet-drafts/draft-ietf-trade-
hiroshi-dom-hash-03.txt"></Transform></Transforms><DigestMethod
Algorithm="http://www.w3.org/2000/02/xmldsig#sha1"></DigestMethod><
DigestValue>CPCKfLiW7xtPWVJxDTsTm8n0/GI=</DigestValue></Reference><
Reference
URI="#Response_E9019A7CF47FD5037FC6D43EDE1E08FD202981D8_1"><Transfo
rms><Transform
Algorithm="http://search.ietf.org/internet-drafts/draft-ietf-trade-
hiroshi-dom-hash-03.txt"></Transform></Transforms><DigestMethod
Algorithm="http://www.w3.org/2000/02/xmldsig#sha1"></DigestMethod><
DigestValue>lqvTPizMdDfehbLpiHYvgi+KZZg=</DigestValue></Reference><
/SignedInfo><SignatureValue>B9UFdLMEBSBnamK4eq1NZHiG2bUNVTLN0nm6Yw4
h6uMFWRVWp76sIw0QJQcrwegyJZD2SLvmKz3uDaBy+sx+wdieq/UTEIuvOrd4TELph7
355i8hOhV3agWdpstxuqupS2PxqpkjTFGCdu1x0SGyxwvRcOXqFudxxiKDt9xYGGk=<
/SignatureValue><KeyInfo><X509Data><X509IssuerSerial><X509IssuerNam
e>C=GB,O=iPlanet,OU=Payments Services,CN=Payments
Root</X509IssuerName><X509SerialNumber>14</X509SerialNumber></X509I
ssuerSerial></X509Data></KeyInfo></Signature><CertBundle><X509Data>
<X509IssuerSerial><X509IssuerName>C=GB,O=iPlanet,OU=Payments
```

```
Services,CN=Payments
Root</X509IssuerName><X509SerialNumber>14</X509SerialNumber></X509I
ssuerSerial><X509Certificate>MIIDQzCCAqygAwIBAgIBDjANBgkqhkiG9w0BAQ
UFADBTMQswCQYDVQQGEwJHQjEQMA4GA1U

.....................

3NoQTXAnM/tQSes7vANiPFskDCg1nxDW0m0dlHBTAYlGeDMOU77wxYAxwD7kn8zMrlB
/uUwOEqsc=</X509Certificate></X509Data></CertBundle><ContentAcknowl
edgement
id="ContentAcknowledgement_E9019A7CF47FD5037FC6D43EDE1E08FD202981D8
_1"><Header
xml:lang="en"><Product>xPx</Product><DocumentType>Acknowledgement</
DocumentType><Version>1.0</Version></Header><References><EleanorTra
nsactionReference>39240ee9250ddcb580002120448471</EleanorTransactio
nReference><SFIReference>Unknown</SFIReference></References><Acknow
ledgementData><AcknowledgementType>PayInst</AcknowledgementType><St
atus>SUCCESS</Status><ReasonCode>00PR00</ReasonCode><ReasonText>Req
uest
Received</ReasonText></AcknowledgementData></ContentAcknowledgement
><Response
id="Response_E9019A7CF47FD5037FC6D43EDE1E08FD202981D8_1"><ResponseD
ata>MIIE/QoBAKCCBPYwggTyBgkrBgEFBQcwAQEEggTjMIIE3zCCAQ+hgZUwgZIxCzA
JBgNVBAYTAnVrMQ8wDQYDVQQIEwZMb25kb24xDzANBgNVBAcTBkxvbmRvbjEQMA4GA1

..................................

HbkMNVTiHWS6gxcBlWMo0blCXuvF571gioA4nkRsIk+aGcrSF7BJg+6hESu/sU2vTqi
tSNEmtqwYvuTKaPl5XVMYRlH4zpiU838+48IzvAtUS4CyQxKfGvYHzo7cDfcQqNqy1G
XQl+ldtzNVKyGf5UBPmJsJxH16X8zSX5TvxCI</ResponseData><CSCResponse><N
IB id="NIB_F8C3B821A28E70139D1CC437F8340E23B42CE885_1"
version="2.0"><ContextInfo
msggrpid="2BAD252ABFCF8A2B3931516F0F0BC462CC92EDFE"
msgid="1001349411141"></ContextInfo><StartTime><LocalTime
id="LocalTime_F8C3B821A28E70139D1CC437F8340E23B42CE885_1"
time="20010924162955Z"/></StartTime><MsgTime><LocalTime
id="LocalTime_F8C3B821A28E70139D1CC437F8340E23B42CE885_2"
time="20010924163651Z"/></MsgTime></NIB><Signature
xmlns="http://www.w3.org/2000/02/xmldsig#"><SignedInfo><Canonicaliz
ationMethod
Algorithm="http://search.ietf.org/internet-drafts/draft-ietf-trade-
hiroshi-dom-hash-03.txt"></CanonicalizationMethod><SignatureMethod
Algorithm="http://www.w3.org/2000/02/xmldsig#rsa-sha1"></SignatureM
ethod><Reference
URI="#NIB_F8C3B821A28E70139D1CC437F8340E23B42CE885_1"><Transforms><
Transform
Algorithm="http://search.ietf.org/internet-drafts/draft-ietf-trade-
hiroshi-dom-hash-03.txt"></Transform></Transforms><DigestMethod
Algorithm="http://www.w3.org/2000/02/xmldsig#sha1"></DigestMethod><
```

```
DigestValue>Ou6H7IQ2U95LvkfwjW0i6DtfUE8=</DigestValue></Reference><
Reference
URI="#Response_D85200FD60A1AEC4FCD7293EADA68B1D05E8DA13_1"><Transfo
rms><Transform
Algorithm="http://search.ietf.org/internet-drafts/draft-ietf-trade-
hiroshi-dom-hash-03.txt"></Transform></Transforms><DigestMethod
Algorithm="http://www.w3.org/2000/02/xmldsig#sha1"></DigestMethod><
DigestValue>GUrLs/8UEnjBJENkoyY/cCUkFW0=</DigestValue></Reference><
/SignedInfo><SignatureValue>HOxQsKKycayWJYhXeyNdc52eWFHv3Y1Nz9CcigO
JQHz+bKV9ewkeKoOSzbngYdufk1hyB8OloYprYIcpVXwKKFeQ7hP+7yC6ODQI1uv1LS
Pi41PUlJH2Q5B7yMHZjyAbxpsudoxThHtOQ+i09KZVJSkO5+Xn1J0QDt8OOMSwtdM=<
/SignatureValue><KeyInfo><X509Data><X509IssuerSerial><X509IssuerNam
e>C=GB,O=iPlanet,OU=Payments Services,CN=Payments
Root</X509IssuerName><X509SerialNumber>9</X509SerialNumber></X509Is
suerSerial></X509Data></KeyInfo></Signature><CertBundle><X509Data><
X509IssuerSerial><X509IssuerName>C=GB,O=iPlanet,OU=Payments
Services,CN=Payments
Root</X509IssuerName><X509SerialNumber>9</X509SerialNumber></X509Is
suerSerial><X509Certificate>MIIDNjCCAp+gAwIBAgIBCTANBgkqhkiG9w0BAQU
FADBTMQswCQYDVQQGEwJHQjEQMA4GA1U

.....................................

nJRKnCCsg==</X509Certificate></X509Data><X509Data><X509IssuerSerial
><X509IssuerName>C=GB,O=iPlanet,OU=Payments Services,CN=Payments
Root</X509IssuerName><X509SerialNumber>1</X509SerialNumber></X509Is
suerSerial><X509Certificate>MIICkjCCAfugAwIBAgIBATANBgkqhkiG9w0BAQU
FADBTMQswCQYDVQQGEwJHQjEQMA4GA1UEC

.....................................

s7vANiPFskDCg1nxDW0m0dlHBTAYlGeDMOU77wxYAxwD7kn8zMrlB/uUwOEqsc=</X5
09Certificate></X509Data></CertBundle><Response
id="Response_D85200FD60A1AEC4FCD7293EADA68B1D05E8DA13_1"><ResponseD
ata>MIIE/QoBAKCCBPYwggTyBgkrBgEFBQcwAQEEggTjMIIE3zCCAQ+hgZUwgZIxCzA
JBgNVBAYTAnVrMQ8wDQYDVQQIEwZMb25kb24xDzANBgNVBAcTBkxvbmRvbjEQMA4GA1

.................................

U2vTqitSNEmtqwYvuTKaPl5XVMYRlH4zpiU838+48IzvAtUS4CyQxKfGvYHzo7cDfcQ
qNqy1GXQl+ldtzNVKyGf5UBPmJsJxH16X8zSX5TvxCI</ResponseData></Respons
e></CSCResponse></Response></Acknowledgement>

RESPONSE END

----------------------------------------------------------------

ragnarok# exit

ragnarok#

script done on Mon 24 Sep 2001 17:31:20 BST
```

iTPS Reinstallation

iWS 4.1 Reinstall

For those versions of software placed on an iWS 4.1

1. consult

   http://docs.iplanet.com/docs/manuals/fasttrak/41/ig/unix.htm

2. Remove all Web Server instances.
3. Reinstall the Web Server and all its configured instances.
4. Reinstall iAS, iTTM and iMQ
5. Reinstall the iTPS

iWS 6.0 Reinstall

For those versions of software placed on an iWS 6.0 it may be possible to reinstall iTPS without having to reinstall the entire Web Server and all its instances.

1. Consult

   http://docs.iplanet.com/docs/manuals/enterprise/50/ig/unix.htm

2. In this case it is possible to Reinstall the components

- BiaB Admin Tool,
- Tooledup,
- BiaB Backend
- CPI
- BFI

When the plugin is installed on iWS 6.0 it uses the answers to the questions you give at install time to configure but it actually uses “wdeploy” to deploy the application, this means that if you uninstall any of the components, the simplest way is to use the wdeploy delete option.

iTPS Backup

Make a backup copy of the iTPS installation and all its associated database tables.

A list of tables can be found as follows:

```
su oracle
sqlplus tbase/tbase
select TABLE_NAME from ALL_TABLES;
exit;
```

To see what other tables need to be backed up please refer to “Database Check Points,” on page 87

Chapter 3

Configuration

Configuration Overview

The following configurations need to take place:

- Certificates and Bank Back End Authorisation
- Database checkpoints that allow you gain access to information about the cause of error during runtime
- Screen Services
- Payments Screens
- Customers Authorisation Services

Certificate Configuration

Before testing your installation is complete you will need to configure the system with the appropriate certificates.

Buyers Bank Certificates

You'll need a signing certificate issued to you by your Bank and containing a Trusted Identrus Root. This is a certificate hierarchy and contains a PKCS12 signing chain containing private key for signing information.

Sellers Website Tooled Up Certificates

This is dealt with in the Install procedure for tooledup.

iPlanet Trustbase Transaction Manager Certificates

Certificates need to be configured as described in the Identrus four corner model. See
http://docs.iplanet.com/docs/manuals/trustbase/221/install/ittm22cb.htm#157035

iPlanet Trustbase Payment Services Certificates

None required

BiaB Certificates

Please refer to "Installing Bank in a Box back office simulator," on page 52 and "Installing Bank in a Box Admin Tool," on page 55 for details about this

CPI certificates

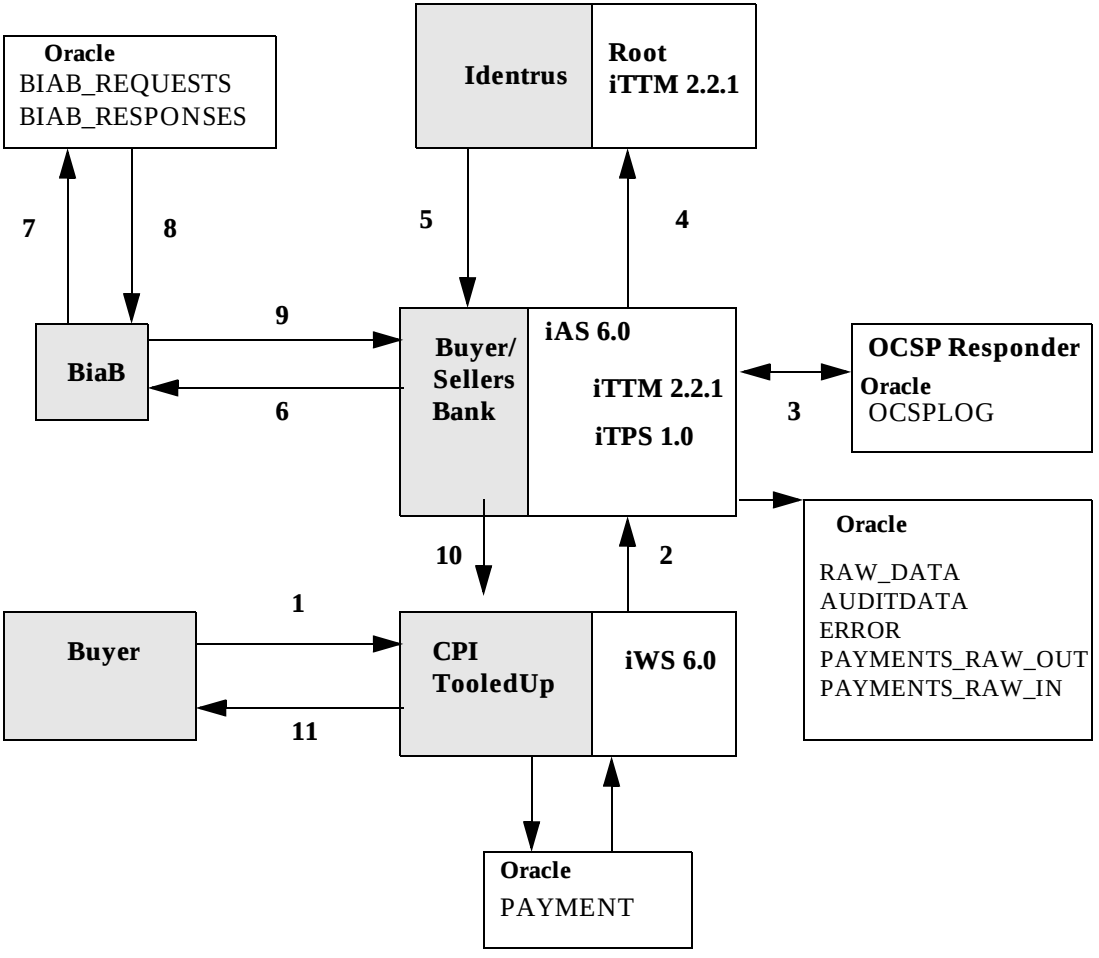
Please refer to “Installing the CPI API,” on page 69 for details of how to install certificates

Database Check Points

This data flow checkpoint diagram and list of points will be used during a transaction and can be used to help locate where an event or error occurs in the system. This is only applicable to synchronous messages. When messages are sent from the Seller web site, messages can be traced at the following points, illustrated using the three corner model in Figure 3-1 “Data Checkpoint List for the three corner model,” on page 88:

Checkpoint	Action/Step/Entry	Table/Location to view data/Message
1	Buyer posts a request to the seller’s web site.	Web server log (web server): *Iws/logs/access*
2	Message (transaction) is sent from CPI to iTTM/iTPS (TC). The message is logged.	Kxs access is written with a timestamp: *ias/logs/kxs* Kjs jvm log file with system.out text: *ias/logs/kjs* iTTM Oracle table: RAW_DATA
	Initially message is validated. If error in validation, log to audit. If error in system, log to error	iTTM Oracle table: AUDITDATA iTTM Oracle table: ERROR
3	TC performs validation check using OCSP to check the status of the Seller’s (Tooled Up) signing certificate.	OCSP log (See OCSP vendor logs)
4	CSC request to Identrus root (freshness).	iTTM Oracle table: RAW_DATA
5	CSC response from Identrus root	iTTM Oracle table: RAW_DATA
6	TC sends message to BiaB.	iTTM Oracle table: PAYMENTS_RAW_OUT
7	BiaB receives message and logs to BiaB Oracle	BiaB Oracle table: BIAB_REQUESTS. Note this message is identical to the message in PAYMENTS_RAW_OUT (step 6). BiaB terminal shows that a message is received.
8	BiaB processes message and logs response to TC in BiaB Oracle.	BiaB terminal shows that a message is ready. BiaB Oracle table: BIAB_RESPONSES
9	TC receives message from BiaB and logs incoming message.	iTTM Oracle table: PAYMENTS_RAW_IN. Note the message is identical to the message in BIAB_RESPONSES (step 8).
10	TC sends message to web server and logs the outgoing message.	iTTM Oracle table: RAW_DATA
11	Seller (tooled up) logs payment message and status.	Seller Oracle table: PAYMENTS

Figure 3-1 Data Checkpoint List for the three corner model



Configuration Pre-requisites

iTTM and iTPS, and all their associated components such as iAS and iWS, should be running while configuring services within iTTM.Thus, you must make sure the following software is up and running in the order specified

1. Oracle 8i
2. nCipher HSMs on all machines
3. iMQ for Java 2.0 on both the Buyer and Seller Bank machines
4. Bank in a Box and iWS 6.0 on the Buyer and Seller Bank machines
5. Bank in a Box administration tool server and iWS 6.0 on the Buyer and Seller Bank machines
6. iTPS on the Buyer and Seller machines
   a. iWS 4.1 on both the Buyer and Seller Bank machines
   b. iAS 6.0 on both the Buyer and Seller Bank machines
   c. iTTM 2.2.1 on both the Buyer and Seller Bank machines
7. JMQ Proxy on both the Buyer and Seller Bank machines
8. Buyer web site (BFI) web server
9. Tooledup Seller web site web server

System Configuration

iPlanet Payment Services Configuration can be accessed from the following menu

Figure 3-2 Payment Main Menu

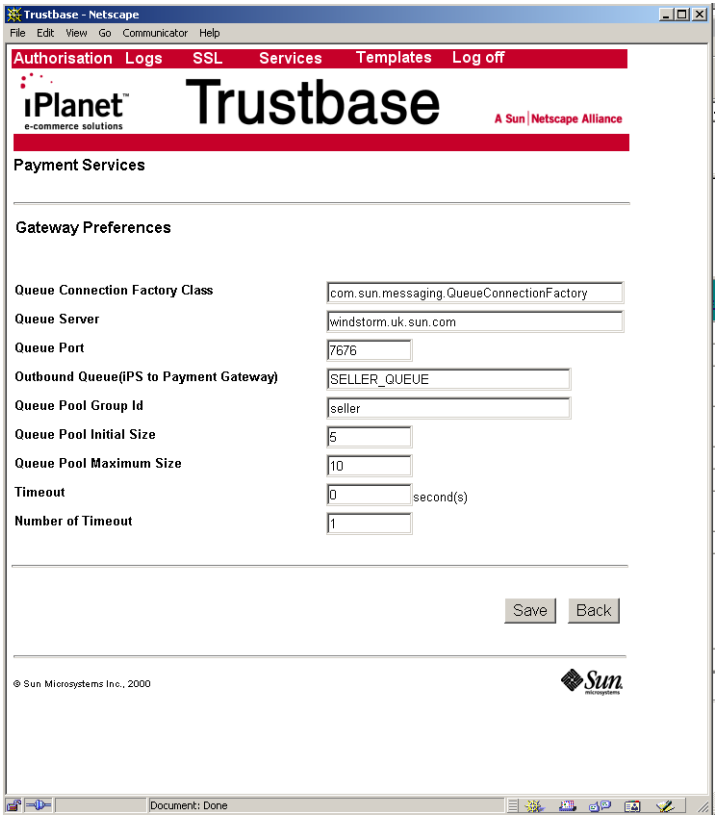
Payment Gateway Preferences

The system administrator may configure

· The JMS queue identifiers used to communicate with the payment gateway.

· The timeout to be used when communicating with the payment gateway.

Figure 3-3 Payment Gateway Preferences Screen



This is already configured from the original installation and is set here as the default.

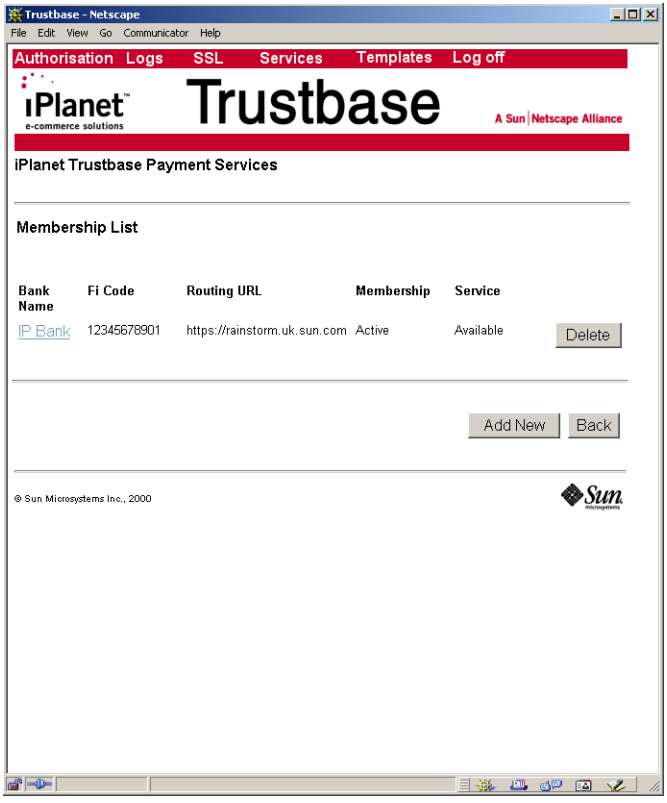
- · Outbound Queue may not be empty

- Timeout is how long you are prepared to wait for a response when you send a message to your JMS queue. 0 is the default and this means it waits indefinitely until it gets a response.
- Number of Timeouts is the number of times you send a message to a queue to get a response. When the timeout is set to 0 it does not matter what you enter here.

Scheme Membership List

The system administrator may add, modify or delete entries from the Scheme Membership List.

Figure 3-4 Scheme Membership List Screen



When the user clicks on a bank name, the Scheme Member Details screen is shown.

When the user clicks the <delete> link, a confirmation dialog is displayed and, if confirmation is given, the corresponding entry is removed from the list.

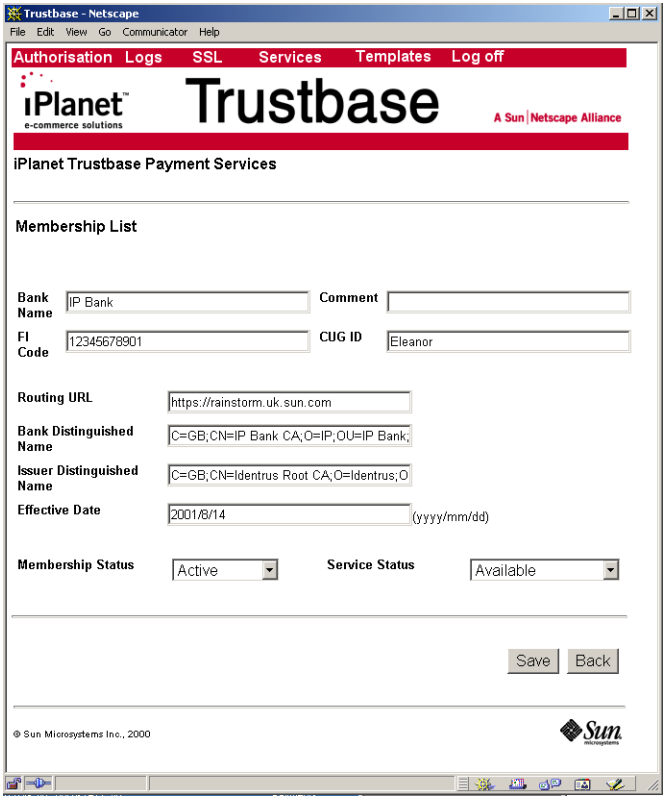
When the user clicks on the <add a new entry> link, the same Scheme Member Details screen is shown with empty fields.

On submission, the Scheme Member Details form is validated according to the following rules:

- Bank Name needs to be present
- Comment is optional
- FI Code needs to be present. FI Code needs to be 11 digits.
- CUG ID needs to be the membership scheme e.g. Eleanor.
- Routing URI needs to be present and a valid URI according to RFC 2396.
- Bank Distinguished Name and Issuer Distinguished Name needs to be present and have a valid distinguished name according to RFC 2253.
- Date is checked.

Note that this does not need to be applied if you are doing 3 corner models. If the node acts in the four corner model then the other SFI's and BFI's that you will communicate with need to be registered.

Figure 3-5 Member Details



Membership Status can be one of

- Active
- Pending
- Suspended
- Terminated

Service Status determines whether the member is still operating within the scheme. Members can be:

- Available
- Unavailable

To enable membership checking for this iTPS, a modification is required to the file

<ittm_install_directory>/<hostname_directory>/identrus.properties

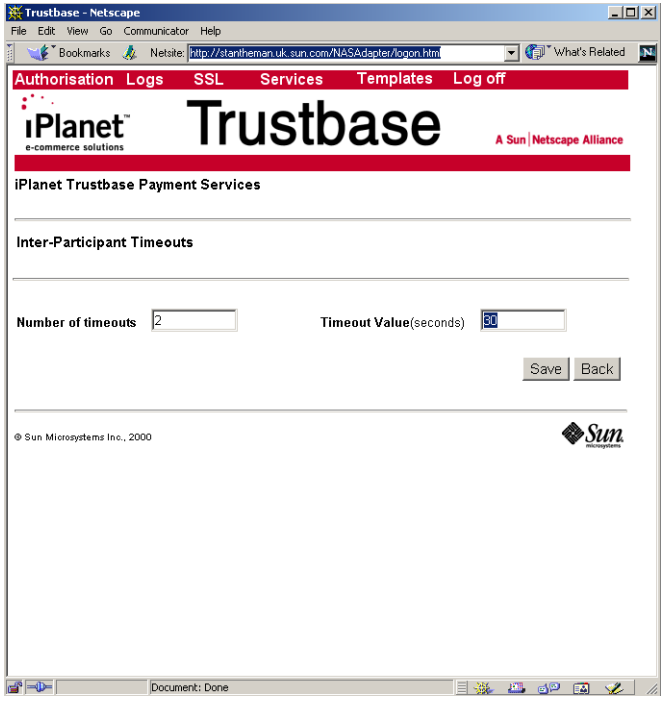
```
Membership.Check=true
```

This has the effect of only allowing payments for financial institutions that are listed within this screen.

Inter-Participant Timeouts

The system administrator may configure the timeout values used when communicating with other Scheme participants.

Figure 3-6 Inter-Participant Timeouts Screen



These parameters determine how often you need to retry sending an interbank message. there are two values: (a) Timeout value: the length of time one waits before sending an interbank message again (b) Number of timeouts: the number of times one keeps retrying to send the interbank message. The Timeout value can be greater than 0. If set to 0, then Timeout is considered infinite. This can be left as the default and under normal circumstances does not need to be changed.

Settlement Chain

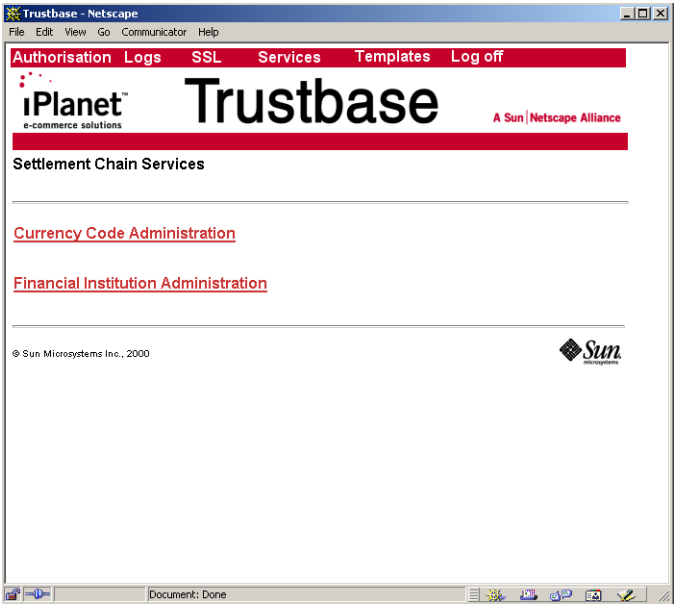
Before making payments, a currency settlement chain needs to be established. This requires, in the first instance, deciding which financial institution you want to use to accept payments in a particular currency. You can then

1. select a currency
2. assign it to the financial institution (provided the financial institution can trade in that currency)
3. finally add it to your currency settlement chain.

In some instances you may want payment initiation messages to be settled in many different currencies. Under such circumstances, iPlanet Trustbase Payment Services needs to be configured to accept a number of financial institutions that are used to settle payments in a variety of currencies.

1. Select <Services><Payments><Settlement Chain>

Figure 3-7 Settlement Chain Main Menu

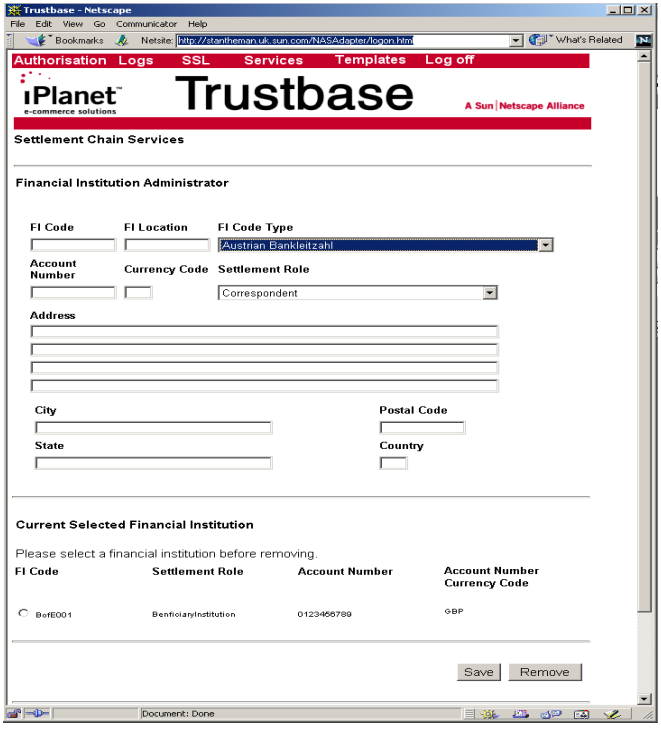


Settlement chains are used by the SFI to inform the BFI of the SFI details of where to make payment, or if the currency supplied in the transaction cannot be accepted at the SFI, a chain of F.I> that can convert the payment into a currency that the SFI can accept.

2. There are four steps for adding a currency to the payment settlement chain:

a. First select <Financial Institution Administration> assign the account number of the institution that is to settle your currency

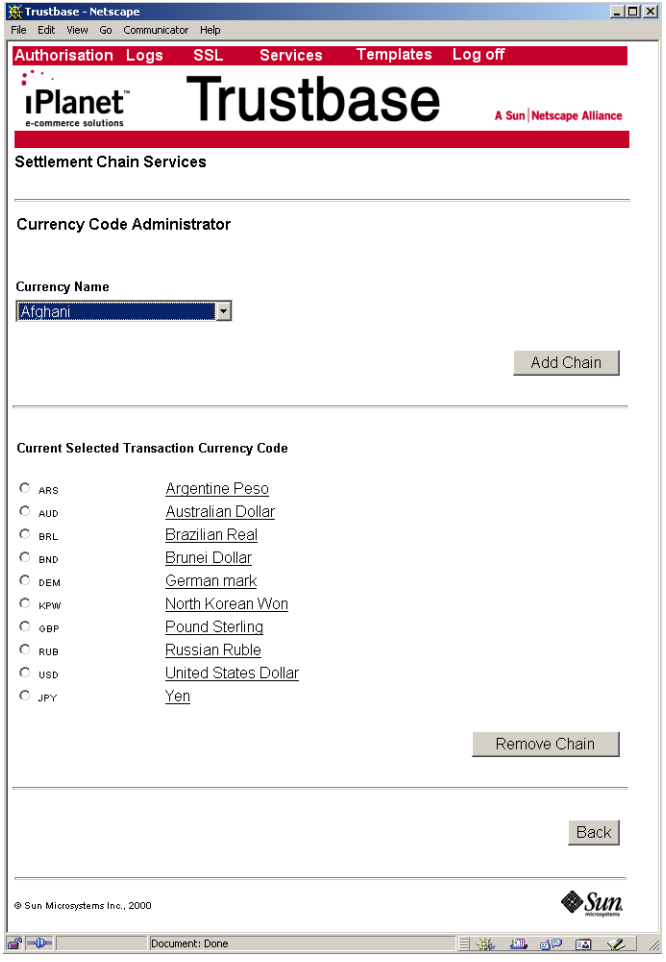
Financial Institution Administration



The SFI settlement role is always set to the Beneficiary Institution. Each financial Institution entered may be entered several times for different currencies and associated accounts into which to place the payment.

b. Second, define the currencies that can be accepted through this SFI. Select <Currency Code Administration>. Each currency code must have at least the SFI associated with the currency, effectively making a settlement chain of one financial Institution, itself.

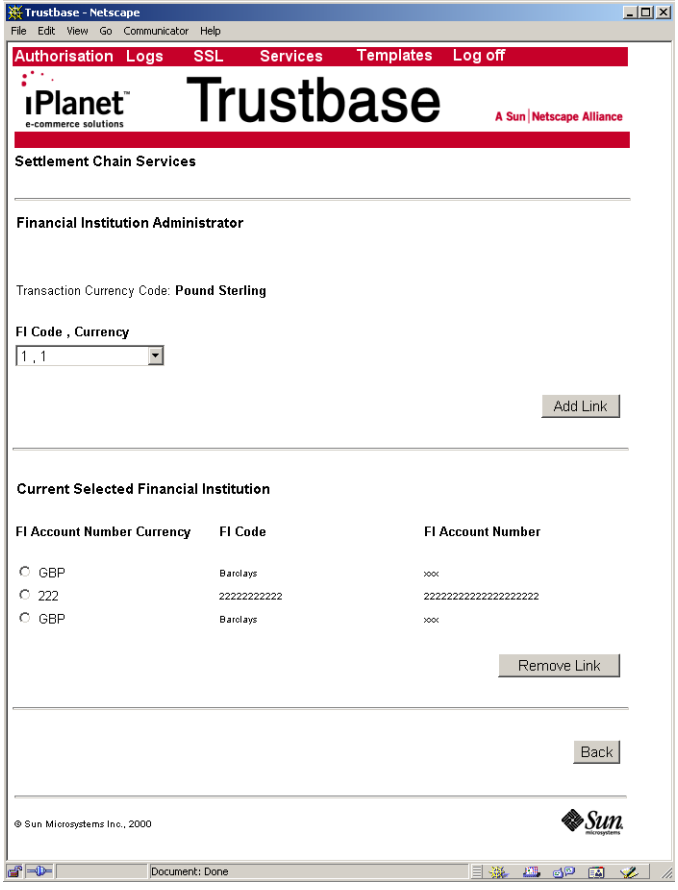
Figure 3-8 Currency Code Administration



c. Third select a currency that you want to assign a financial institution to. Select <Add Chain>

d. Fourth select a currency, from the screen on the previous page, by double clicking on it and match the currency to the financial institution by selecting <Add Link> as illustrated below:

Figure 3-9 Adding a currency code to the Settlement Chain



If a SFI can accept a currency directly, then it alone has to be associated with that currency. If the SFI cannot directly accept a payment in a specific currency then a settlement chain of additional financial institutions has to be created in order to convert the payment into a currency that can be accepted by the SFI.

3. Adjusting some of the data that appears in some of the tables can be done via the Oracle tables that are installed via the SQL script

   ```
   <Trustbase_install_directory>/TTM/V2.2/Config/sql/Payments.sql
   ```

The following table may need to be reconfigured:

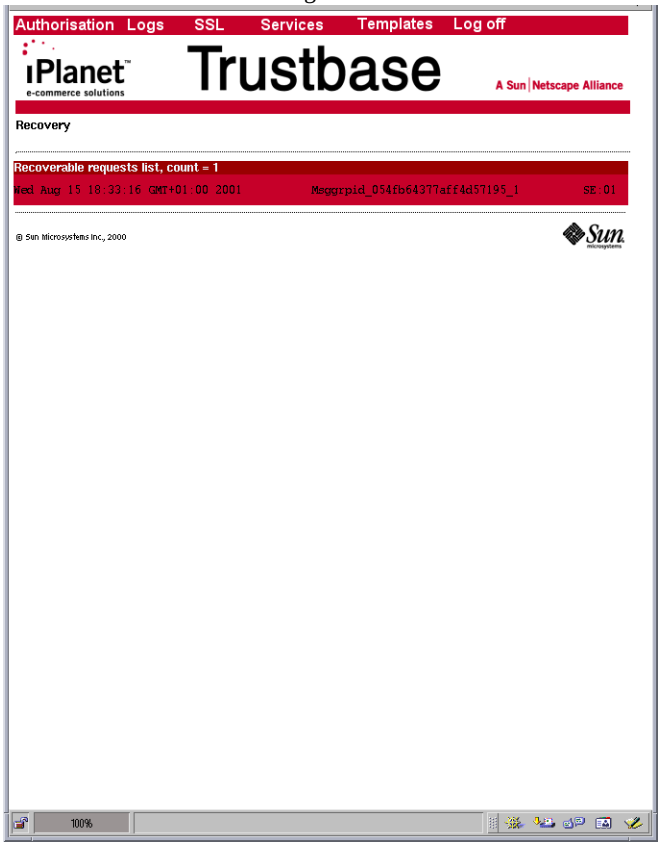
- SC_CURRENCY_CODE_LIST - This contains all the possible currency codes that a user can select from when forming a chain or creating a transaction currency code.

Payment Recovery

iTPS provides the facility to re-present a payment request in the event that it failed on the first attempt. This can occur for example, if the buyers financial institution is not contactable at the time of the payment request.

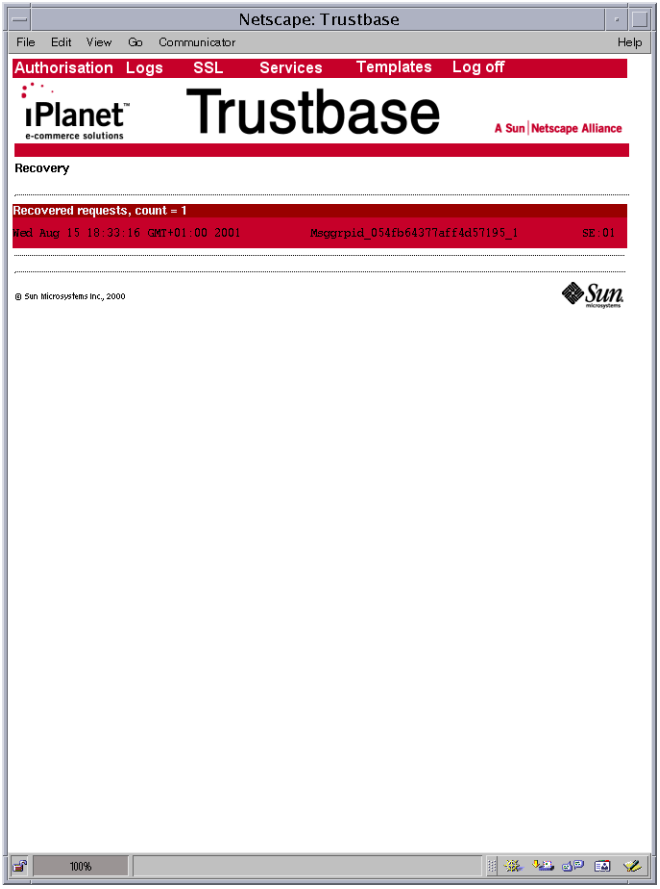
By select the <List Recoverable messages> from the payments menu, any transactions that started but failed to complete will be listed, as shown below.

Figure 3-10 Recoverable Messages



To initiate a recovery, select the <Start recovery process> option from the payments menu. This may take some time as it re-presents any recoverable payment requests through the system. Upon completion a list of recovered transactions is displayed, as shown below.

Figure 3-11 Recovered messages



Customer Authorisation Service for iTPS

Authorising Customers to the payment services is achieved by matching certificate name details. Upon successful match it assigns the customer with a customer ID. This is different from iPlanet Trustbase Transaction Manager's authorisation facility as iTPS has individual authorisation rather than generic corporate authorisation.

1. Using SQLPlus go into your iPlanet Trustbase Transaction Manager Server

```
SQL> describe cust_dn_mapping;
```

Figure 3-12 cust_dn_mapping

Name	Null	Type
DN	NOT NULL VARCHAR2	(1024)
CUST_ID	NOT NULL VARCHAR2	(36)

```
SQL> insert into cust_dn_mapping values ('CN=RC from
IP;O=IP;OU=IP Bank;','1');
1 row created.

SQL> commit;
Commit complete.

SQL>
```

2. Modify the properties file identrus.properties by changing the following to

```
customer.auth.checking = true
```

3. The DN column accepts the customers signing certificate subject name as its input.

Chapter 4

Running the System

Once you've installed and configured, this chapter shows you how you can test your system is up and running correctly and processing payments requests as expected.

Starting the system

Before checking any particular component you must bring the individual components up and make sure that the system is actually running. Starting the system must be performed in a particular order otherwise components will fail to communicate properly. The order for starting the system is:

1. Oracle 8i
2. nCipher HSMs on all machines
3. iMQ for Java 2.0 on both the Buyer and Seller Bank machines
4. Bank in a Box and iWS 6.0 on the Buyer and Seller Bank machines
5. Bank in a Box administration tool server and iWS 6.0 on the Buyer and Seller Bank machines
6. iTPS on the Buyer and Seller machines
    a. iWS 4.1on both the Buyer and Seller Bank machines
    b. iAS 6.0 on both the Buyer and Seller Bank machines
    c. iTTM 2.2.1 on both the Buyer and Seller Bank machines
7. JMQ Proxy on both the Buyer and Seller Bank machines
8. Buyer web site (BFI) web server
9. Tooledup Seller web site web server

The following sections provide instructions for checking that components are running, starting and stopping each component.

Oracle 8i

Oracle 8i is a complex product and the instructions are intended as a quick list of items that are useful when trying to determine the status of the Oracle installation.

Useful information to check on the installation and make a note of:

Information Type	**Example Set-up Value for Oracle 8i**
Install directory	Oracle program files: /opt/oracle/app/product/8.1.7/bin Oracle data files: /identrusdb/orcl
Oracle user login	Username: oracle, Password: oracle
Sqlplus - dba admin	Username: sys, Password: change_on_install
Sqlplus - tbase user	Username: tbase, Password: tbase
Operational ports	Oracle ports: 1521
SID	orcl

Useful commands for starting and stopping Oracle. Checking Oracle is running can be performed by looking at the running processes using the process grep or process list commands. If Oracle is not running then you will need to log in as the Oracle superuser and start the Oracle.

Action or check	**Command or output**
To start server	As oracle user: svrmgrl; Connect internal; startup; exit lsnrctl; start; exit
To stop server	As oracle user: lsnrctl; stop; exit svrmgrl; connect internal; shutdown; exit
Processes grep	ps -ef | grep oracle
Process list - there will be an oracle orcl for each application connection.	oracle 9862 1 0 12:48:10 ? 0:00 orcl (DESCRIP-TION=(LOCAL=no)(ADDRESS=(PROTOCOL=BEQ))) oracle 764 1 0 Mar 07 ? 0:01 /opt/oracle/bin/tnslsnr LISTENER -inherit oracle 751 1 0 Mar 07 ? 0:00 ora_pmon_orcl oracle 753 1 0 Mar 07 ? 0:00 ora_dbw0_orcl oracle 755 1 0 Mar 07 ? 0:00 ora_lgwr_orcl oracle 757 1 0 Mar 07 ? 0:22 ora_ckpt_orcl oracle 759 1 0 Mar 07 ? 0:02 ora_smon_orcl oracle 761 1 0 Mar 07 ? 0:00 ora_reco_orcl oracle 9771 1 0 12:47:58 ? 0:00 oracleorcl (DESCRIP-TION=(LOCAL=no)(ADDRESS=(PROTOCOL=BEQ)))
Tables of interest	Auditdata**:** Contains internal audit information and indicates what the TC has processed. Error: Shows unexpected errors e.g. cannot communicate with Certificate Authority Error_support**:** Shows any java stack trace associated with the error table.

nCipher

Information Type	Example Set-up Value nCipher
Install directory	/opt/nfast
Operational ports	9000
To start server	nfast start
To stop server	nfast stop
Processes grep	ps -ef | grep hard
Process list	nfast 4241 1 0 Mar 05 ? 0:22 ../sbin/hardserver –llogfile nfast 4246 4241 0 Mar 05 ? 0:10 ../sbin/hardserver -llogfile
Documentation	nCipher KeySafe 1.0 http://www.ncipher.com

To check that the nCipher is running perform a process list on each machine. If no nFast process is in the list you will need to start the nFast hard server using the appropriate command.

iMQ for Java 2.0

iMQ for Java 2.0 needs to be started before iTPS can be run. The following illustrates this:

cd /opt/SUNWjmq/bin

./jmqbroker

```
[06/Sep/2001:12:50:14 GMT]

================================================================

iPlanet Message Queue for Java

Copyright 2001

Version:  2.0 SP1 (Build 321-b)

Sun Microsystems, Inc.

Compile:  Fri Aug 3 10:30:43 PDT 2001

All Rights Reserved

This product includes code licensed from RSA Data Security.

================================================================

Java Runtime Version: 1.3.0_02 Sun Microsystems Inc.
/opt/SUNWjmq/jre

[06/Sep/2001:12:50:15 GMT] [B1060]: Loading persistent data...

[06/Sep/2001:12:50:16 GMT] [B1039]: Broker
"jmqbroker@windstorm:7676" ready.
```

This will create a JMQ broker that was the default port 7676. If you want to specify a different port then use:

```
./jmqbroker -port <portnumber>
```

Bank in a Box

To run the Bank in a Box, run the biab script located in the scripts directory. The script accepts the following arguments, although none are required for normal operation

-verbose	verbose output
-debug	debug output
-quiet	only display warnings and errors
-logfile <file>	Specify the name of a file for log output
-admin	Enter user administration mode

If the server was started in admin mode, user management may be performed at the BiaB command line. The following commands are accepted:

adduser <username> <password>	Adds a user to the system
enableuser <username>	Enables a user account
disableuser <username>	Disables a user account
listusers	Displays a list of user accounts
version	Displays the version of the software

Bank in a Box Back End can be started as follows:

```
bash-2.03# ./scripts/biab -debug
[AUDIT] Starting BIAB [V1.0-1001500003703-18]
```

Bank in a Box administrator tool

The Bank in a Box administrator tool is a Web server application running on iAS 6.0. To check that the Web Server is running use the grep command given below. If the server is not running then start the admin server and use the tools within the adminserver to manage the web server

Information Type	**Example Set-up Value iws6**
Install directory	/opt/netscape/server6
Administration logon	Username: iwsadmin, Password: identrus
Operational ports	Server: 80, Admin: 8888
To start server	/opt/netscape/server6/https-<Host-Name>/start
To stop server	/opt/netscape/server6/https-<Host-Name>/stop
To start admin server	/opt/netscape/server6/https-admin/start
To stop admin server	/opt/netscape/server6/https-admin/stop
Processes grep	ps -ef | grep iws
Process list	nobody 9876 1 0 12:52:08 0:00 ./uxwdog -d /opt/net-scape/server6/https-<Host-Name>/config nobody 9877 9876 0 12:52:08 0:01 ns-httpd -d /opt/net-scape/server6/https-<Host-Name>/config also /opt/netscape/server6/https-admin/config if the admin is running
Install logs	/opt/netscape/server6/setup/WebServer/
Log directory	/opt/netscape/server6/https-<Host-Name>/logs
Document root	/opt/netscape/server6/docs
Installation and Con-figuration Documents	❍ http://docs.iplanet.com/docs/manuals/enterprise/50/ig/contents.htm ❍ http://docs.iplanet.com/docs/manuals/enterprise/50/ag/esgstart.htm#1003083

iTPS

The iTPS is reliant on three components running:

- iWS 4.1
- iAS 6.0
- iTTM 2.2.1.

If all these components have been started correctly then the iTPS component should be available. To check to ensure that the components are running, use the grep commands shown in the tables below. If iTTM is running, but iAS is not, shutdown the iTTM and restart the components starting with iAS 6.0 .

iWS 4.1

Information Type	Example Set-up Value for iWS4.1
Install directory	/opt/netscape/server4
Administration logon	Username: iwsadmin, Password: identrus
Operational ports	Server: 80, Admin: 8888
To start server	/opt/netscape/server4/https-<Host-Name>/start
To stop server	/opt/netscape/server4/https-<Host-Name>/stop
To start admin server	/opt/netscape/server4/https-admin/start
To stop admin server	/opt/netscape/server4/https-admin/stop
Processes grep	ps -ef | grep iws
Process list	nobody 9876 1 0 12:52:08 0:00 ./uxwdog -d /opt/netscape/server4/https-<Host-Name>/config nobody 9877 9876 0 12:52:08 0:01 ns-httpd -d /opt/netscape/server4/https-<Host-Name>/config also /opt/netscape/server4/https-admin/config if the admin is running
Install logs	/opt/netscape/server4/setup/WebServer/
Log directory	/opt/netscape/server4/https-<Host-Name>/logs
Document root	/opt/netscape/server4/docs
Installation and Configuration Documents	❍ http://docs.iplanet.com/docs/manuals/fasttrak/41/ig/contents.htm ❍ http://docs.iplanet.com/docs/manuals/fasttrak/41/ag/esgstart.htm#998517

iAS 6.0

Information Type	Example Set-up Value for iAS6.0
Install directory	/opt/iplanet/ias6
Administration logon	Username: admin, Password: password
Operational ports	Directory Admin: 20000, kas admin:10817, Directory server: 389
To start server	/opt/Trustbase/TTM/Scripts/startias
To stop server	/opt/Trustbase/TTM/Scripts/stopias
Installation logs	/opt/iplanet/ias6/setup/
Processes grep	ps -ef | grep ias To get just the 'kiva' processes (the ones that do the jvm work) do a ps -ef | grep k.s
Process list	root 10066 10064 0 14:33:21 0:03 /opt/iplanet/ias6/ias/bin/.kjs -cset CCS0 root 10059 9504 0 14:33:16 pts/6 0:00 /opt/iplanet/ias6/ias/bin/.kas root 9504 1 0 12:47:38 pts/6 0:00 /bin/sh /opt/iplanet/ias6/ias/bin/kas root 10070 1 0 14:33:25 0:00 /bin/sh /opt/iplanet/ias6/ias/bin/kcs -cse t CCS0 -eng 2 root 10064 1 0 14:33:21 ? 0:00 /bin/sh /opt/iplanet/ias6/ias/bin/kjs -cset CCS0 -eng 1 root 10061 1 0 14:33:19 ? 0:00 /bin/sh /opt/iplanet/ias6/ias/bin/kxs -cset CCS0 -eng 0 root 10072 10070 0 14:33:25 ? 0:00 /opt/iplanet/ias6/ias/bin/.kcs -cset CCS0 -eng 2 root 10062 10061 0 14:33:19 ? 0:01 /opt/iplanet/ias6/ias/bin/.kxs -cset CCS0 -eng 0 nobody 8174 1 0 12:45:04 ? 0:04 ./ns-slapd -f /opt/iplanet/ias6/slapd-unix d02/config/slapd.conf -i /opt/iplanet/ias6/slapd-<Machine-name> (check?)
Logged processes	kxs_0_CCS0: Contains information about the incoming message and the plugin start and stop kjs_0_CCS0: Contains the standard out from any running java process – can contain some debug information.
Installation Document	`http://www.iplanet.com/products/infrastructure/app_servers/index.html`

iTTM 2.2.1

Information Type	Example Set-up Value for iTTM 2.2.1
Install directory	/opt/Trustbase
Administration logon via web	Username: administrator, Password: administrator
Certificate manager	/opt/Trustbase/TTM/Scripts/runcertmanager
Operational ports	Admin via web: 80 (http://10.211.20.50/NASAdapter/logon.html)
To start server	/opt/Trustbase/TTM/Scripts/starttbase
To stop server	/opt/Trustbase/TTM/Scripts/stoptbase
Property file location	/opt/Trustbase/TTM/<Host-Name>/
Processes grep	ps -ef | grep java
Process list	root 9658 1 0 12:47:48 pts/6 0:04 /usr/bin/../java/bin/../jre/bin/../bin/sparc/native_threads/java uk.co.jcp.app. root 9713 1 0 12:47:53 pts/6 0:08 /usr/bin/../java/bin/../jre/bin/../bin/sparc/native_threads/java uk.co.jcp.tbas root 9790 1 0 12:48:03 pts/6 0:12 /usr/bin/../java/bin/../jre/bin/../bin/sparc/native_threads/java uk.co.jcp.secu
Installation Document	http://docs.iplanet.com/docs/manuals/trustbase/221/install/contents.htm

Enabling the JMSProxy

To run the JMS proxy, run the jmsproxy script located in the scripts directory as

```
<jms_install_directory>/jmsproxy/scripts/jmsproxy
```

Buyer and Seller web applications

These Web applications are both deployed on top of the iWS 6.0 installations on the Buyer and Seller Web site machines. In order to check that these applications are available, use a browser to go to the appropriate URL.

Information Type	**Example Set-up Value iws6**
Install directory	/opt/netscape/server6
Administration logon	Username: iwsadmin, Password: identrus
Operational ports	Server: 80, Admin: 8888
To start server	/opt/netscape/server6/https-<Host-Name>/start
To stop server	/opt/netscape/server6/https-<Host-Name>/stop
To start admin server	/opt/netscape/server6/https-admin/start
To stop admin server	/opt/netscape/server6/https-admin/stop
Processes grep	ps -ef | grep iws
Process list	nobody 9876 1 0 12:52:08 0:00 ./uxwdog -d /opt/net-scape/server6/https-<Host-Name>/config nobody 9877 9876 0 12:52:08 0:01 ns-httpd -d /opt/net-scape/server6/https-<Host-Name>/config also /opt/netscape/server6/https-admin/config if the admin is running
Install logs	/opt/netscape/server6/setup/WebServer/
Log directory	/opt/netscape/server6/https-<Host-Name>/logs
Document root	/opt/netscape/server6/docs
Installation and Con-figuration Documents	❍ http://docs.iplanet.com/docs/manuals/enterprise/50/ig/contents.htm ❍ http://docs.iplanet.com/docs/manuals/enterprise/50/ag/esgstart.htm#1003083

If the Web Servers are not running then use the process grep (on the host machine) to check that the web server is running. If the Web Server process is not running then start the webserver using the admin console.

Running the Models

We now describe how to run the system for each main kind of Payment Model

Running the Three Corner Model

In this situation the Buyer's Bank is the same as the Seller's Bank, i.e. the buyer and the seller have both been issued with certificates from the same Financial Institution.

1. User interfaces with the Seller's Website, in this case TooledUp, and initiates a payment
2. Payment Message gets sent to the iPlanet Trustbase Payment Services Server
3. iPlanet Trustbase Transaction Manager informs its backend system or in this example Bank in a Box.
4. Bank in a Box then sends confirmation of payment to TooledUp
5. The status of this payment initiation is returned back to the seller and hence buyer.

Running the Four Corner Model (SFIM)

1. Buyer interfaces with Seller's Website, in this particular instance TooledUp, and initiates a payment.
2. Payment Message gets sent to iPlanet Trustbase Payment Services Server at the Seller's Bank informs its back end systems that in turn informs the Buyers Bank.
3. Buyers Bank informs back end system, in this case Bank in a Box.
4. A response is returned to its financial institution
5. The SFI on receiving the response from the BFI informs its back end systems and response gets sent to the Sellers Website confirming payment.

Making a Payment via the Buyers Bank (BFIM)

1. If the Subscriber signed data is signed by the Buyer then
   a. Buyer initiates payment from the Buyers Bank Website
   b. Payment Message is sent to iPlanet Trustbase Payment Services that in turn informs the Buyers Bank back end systems.
   c. Response gets returned to Buyers Bank Website
2. If the seller has signed the subscriber signed data then
   a. Buyer initiates payment from the Buyers Bank Website
   b. Payment Message is sent to iPlanet Trustbase Payment Services that in turn informs the Buyers Bank back end systems.
   c. The BFI informs the seller's SFI
   d. The SFI informs its back end systems
   e. Response sent back to the BFI
   f. BFI responds back to the buyer

NOTE More Information about how each payment scheme defines its Models and Payment products can be found at http://www.identrus.com

Example supported Schemes include:

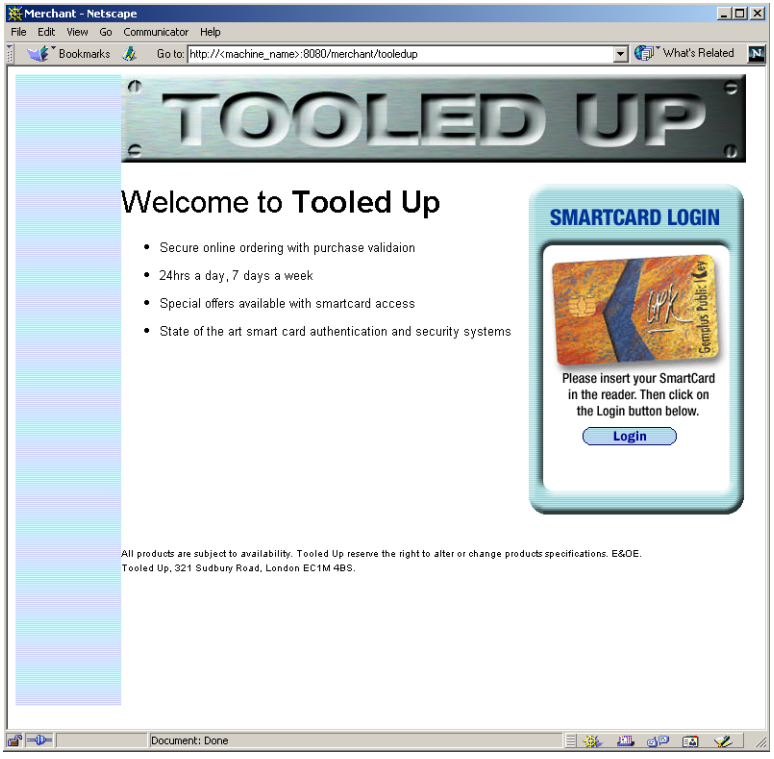
Eleanor Payment Reference Specification

Initiating Payment via Sellers Website TooledUp

You can test the system has been installed correctly by going to the Tooledup Website and initiating a payment as follows.

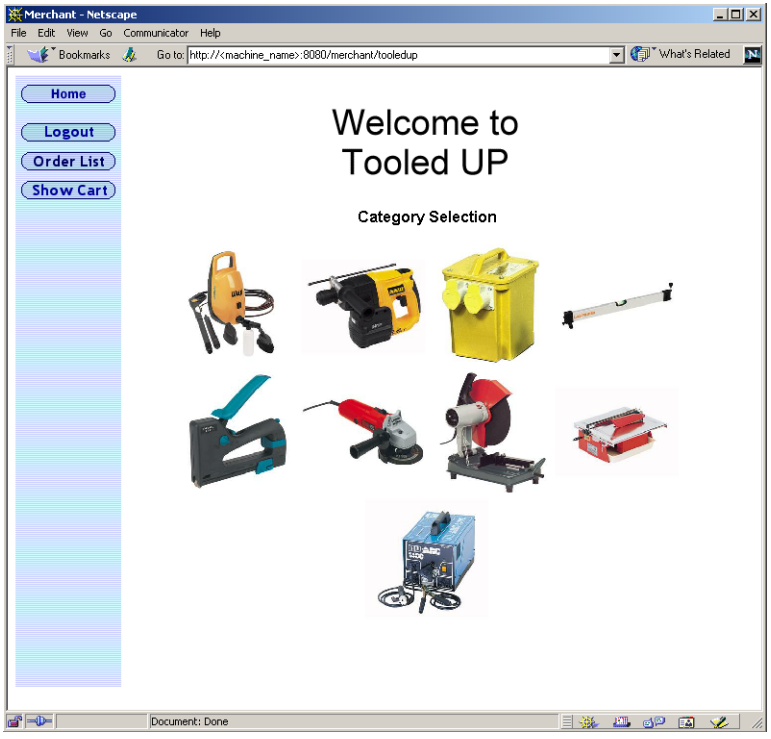
1. Go to TooledUp http://<server_name>:<port>/<uri_path>/tooledup

Figure 4-1 TooledUp Main Menu



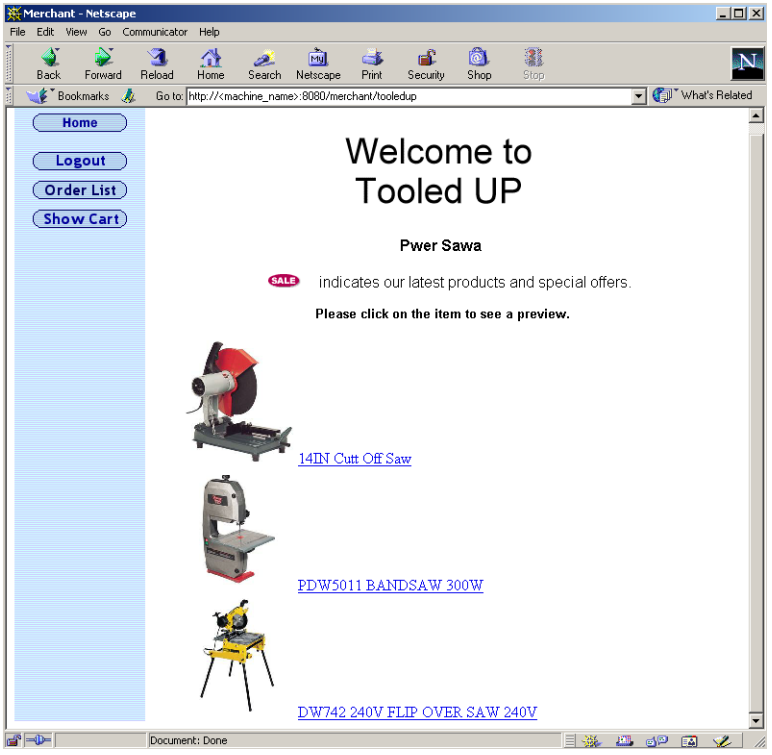
2. Insert Your Smart Card and login. The following menu appears

Figure 4-2 TooledUp Ltd Catalogs



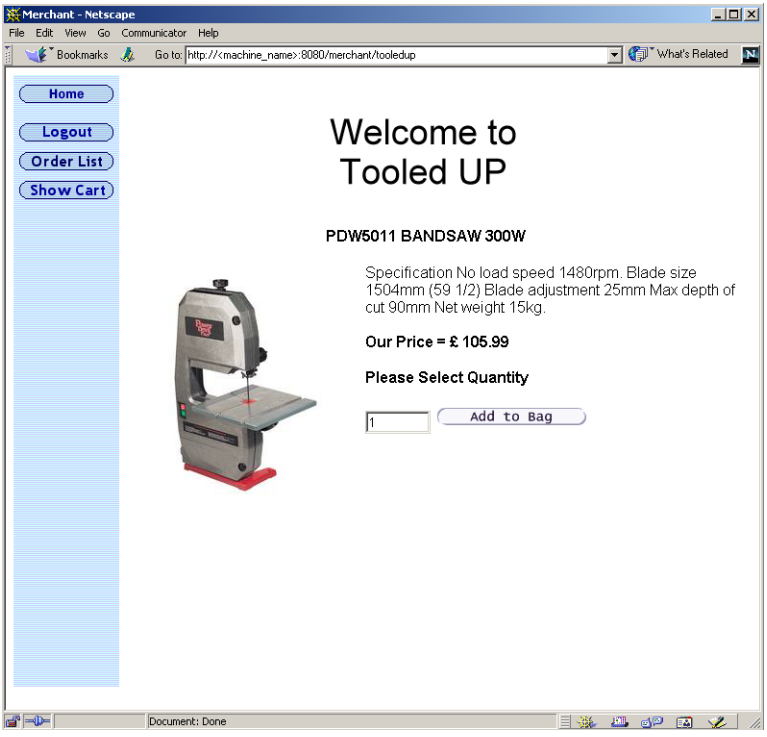
3. Select a product to purchase

Figure 4-3 TooledUp Category Selection



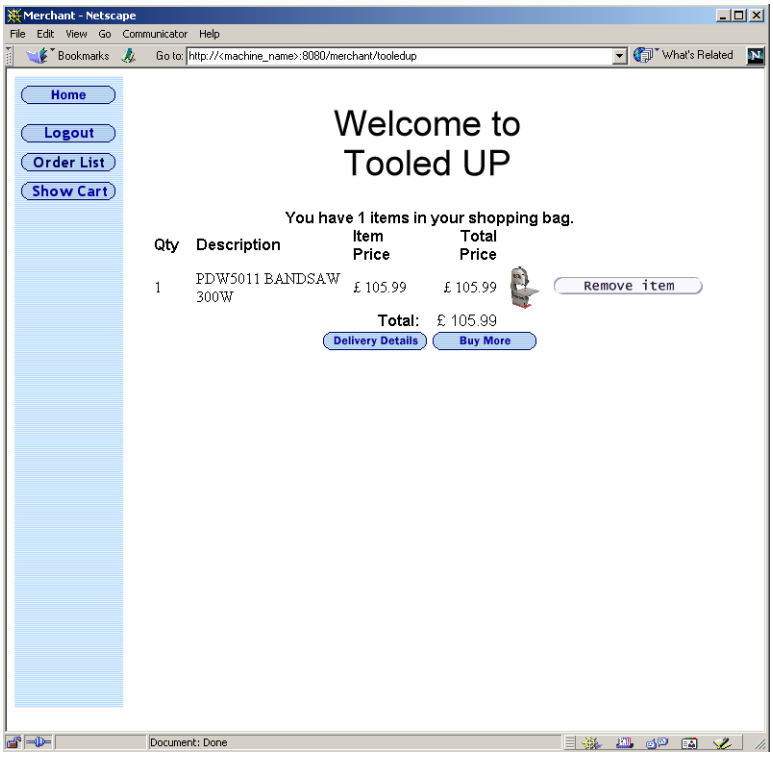
4. Add it to the Shopping Basket

Figure 4-4 Add to Shopping Basket



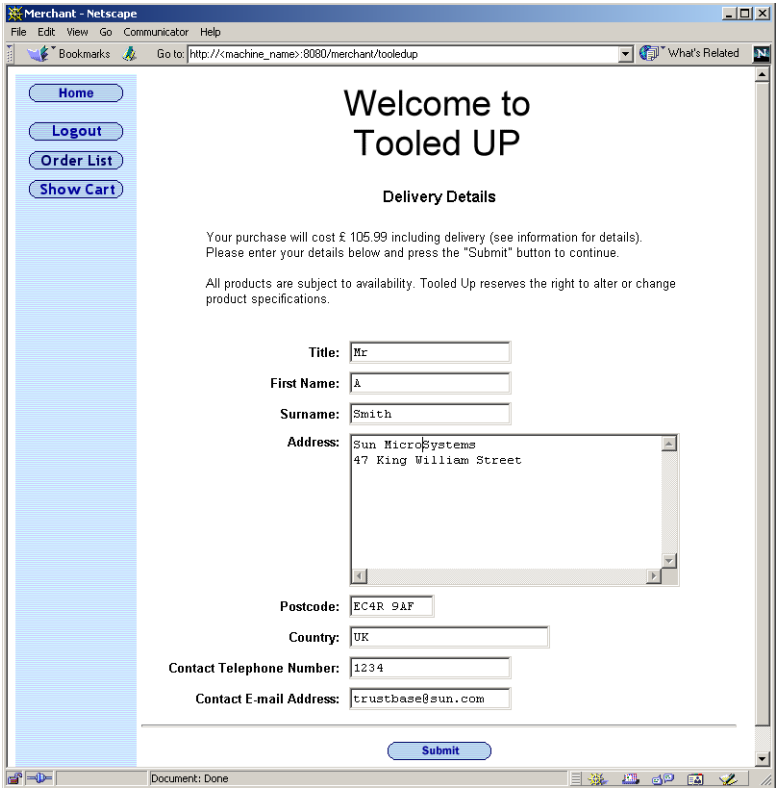
5. Shopping Basket Details

Figure 4-5 Shopping Bag Details



6. Make delivery Details

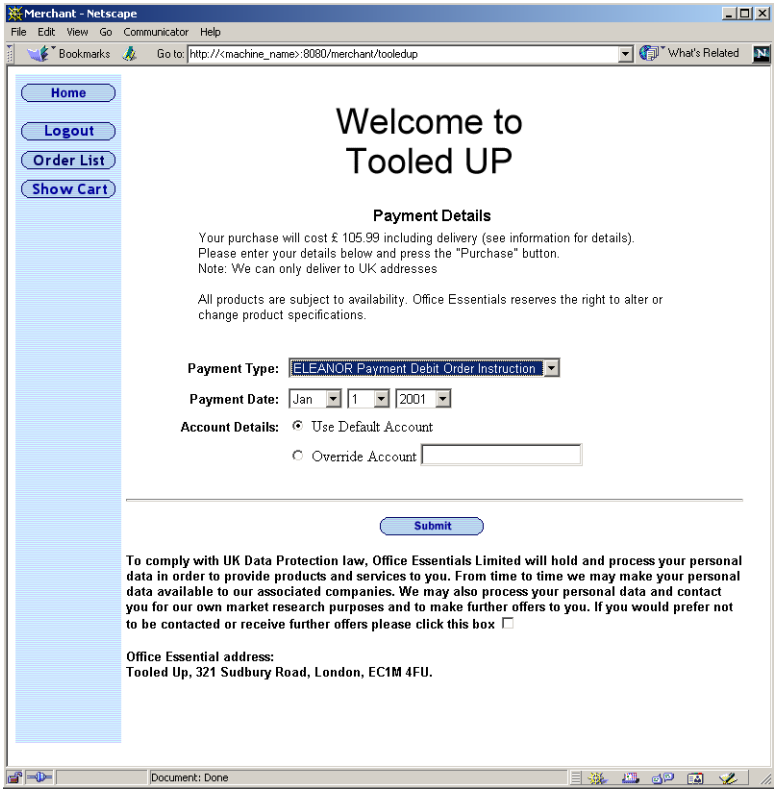
Figure 4-6 Enter Delivery Details



7. Make payment. Select Submit at the bottom of the Delivery screen menu

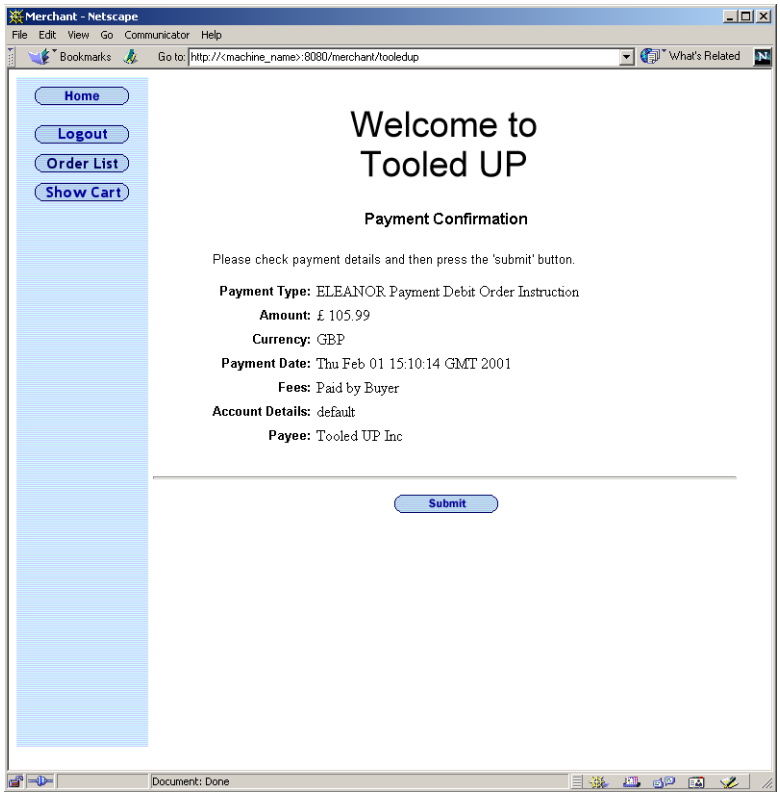
8. Confirm Delivery Details and Payment type

Figure 4-7 Payment Type



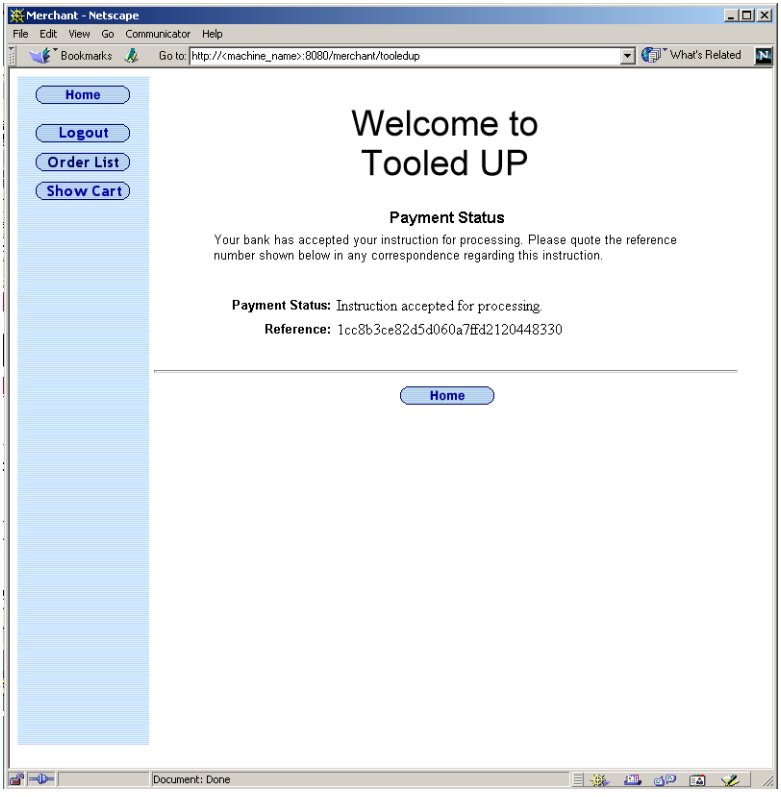
9. Confirm Delivery Details

Figure 4-8 Confirm Delivery Details



10. Payment Accepted

Figure 4-9 Payment Accepted



11. Payment Confirmation

The XML message from this transaction can be confirmed in a number of different ways:

a. Via your API com.iplanet.trustbase.initiator.cpi

b. Viewing the Identrus raw_data log (see your iPlanet Trustbase Transaction Manager Developer Guide http:/ / docs.iplanet.com / docs / manuals / trustbase / 221 / dev / ittm22dn.htm#131923)

c. Editing IWS6 startup UNIX script

<IWS6_ Install_Directory> / https-<Server_Name>start

by adding a debug feature as follows:

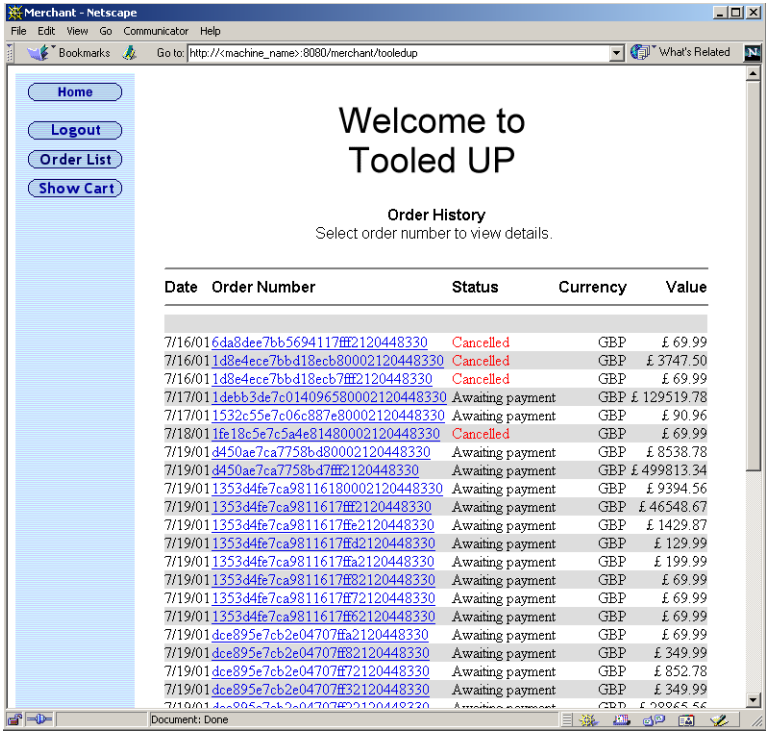
```
case $arg in
          -debug)
                ./ns-httpd -d $PRODUCT_SUBDIR/config
                exit 0
                ;;
        -start)
                ./$PRODUCT_BIN -d $PRODUCT_SUBDIR/config $@
                if test $? -ne 0 ; then
                        exit 1
                fi
```

then run the script as

```
./start -debug
```

12. Check Order List. Finally there is a Tooled Up screen to display confirmed payment requests.

Figure 4-10 Order List



Running Bank in a Box Back End

Please refer to “Installing Bank in a Box back office simulator,” on page 52

Running Bank in a Box Admin Tool

The Bank in a Box (BiaB) has been expanded to allow it to present a user interface permitting examination of messages received, and sending of response messages. This allows a standard installation of iTPS to be used in a live system, by requiring manual intervention between the BiaB interface and the real bank back end infrastructure. Clearly, this approach is only feasible for very low transaction volumes, but does allow evaluation of the product prior to full scale integration with the existing back end infrastructure. The system also allows you to acknowledge Payments. The following provides a walkthrough of this operation

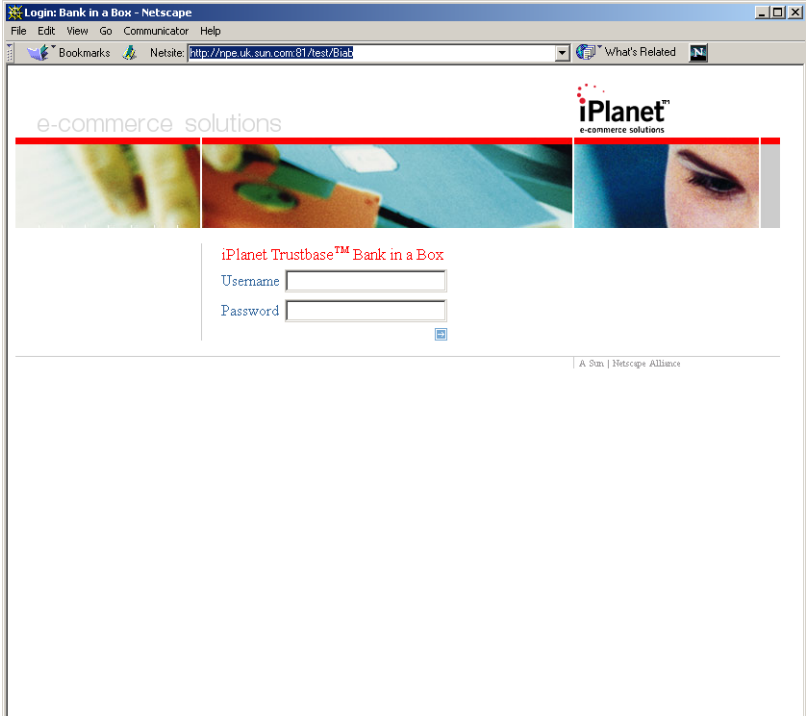
1. Make Sure your BiaB Backend Server is running and a username and password has been allocated to. This can be changed by starting the BiaB in Admin mode and typing

   ```
   adduser <username> <password>
   ```

2. Load the following URL in your browser:

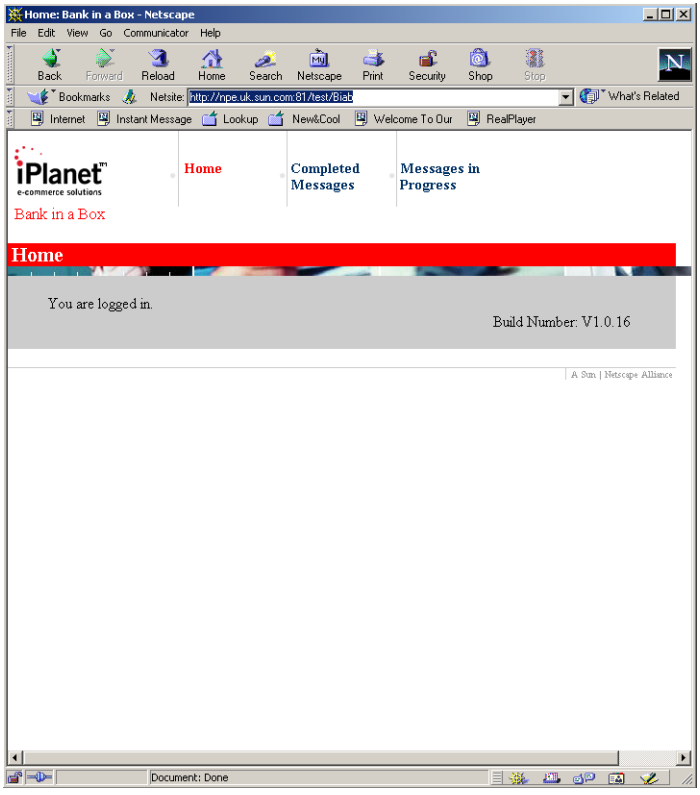
   http://<hostname><port>/<uri_path>/Biab

Figure 4-11 Bank in a Box MainMenu



3. Type in the username and the password. The following menu appears

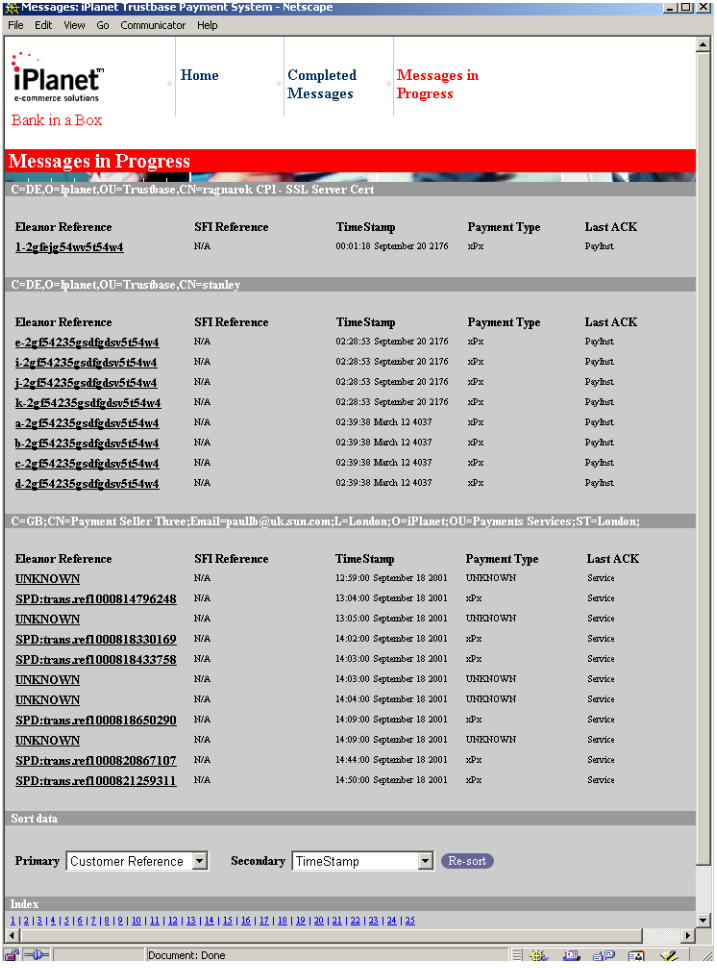
Figure 4-12 Bank in a Box Admin Tool Homepage



4. Select <messages in progress>

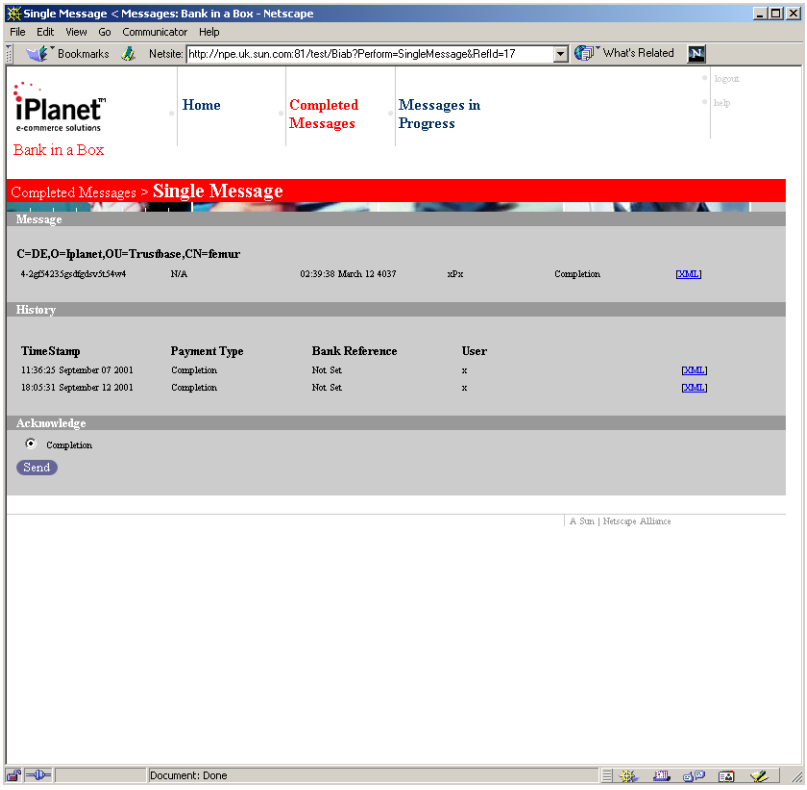
5. An example screen containing some messages now follows. Clearly the first time there will be no messages.

Figure 4-13 BiaB Message Screens



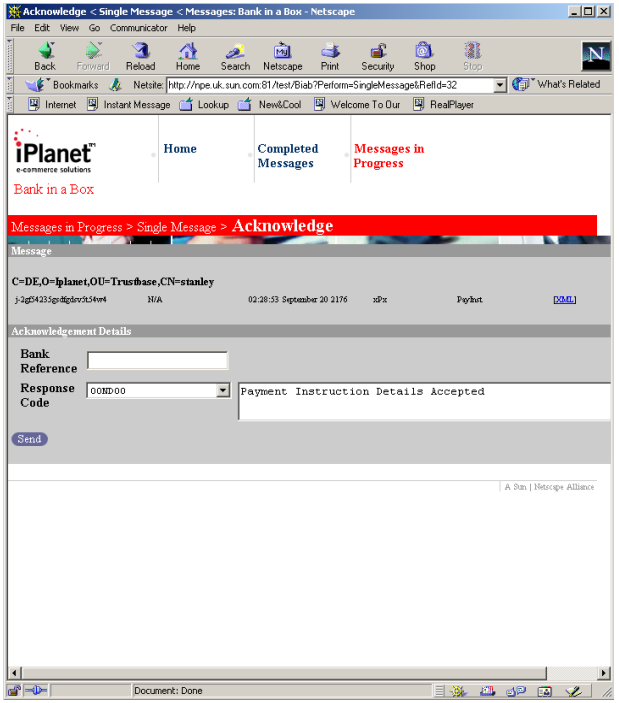
6. Select an individual message to view and the following screen appears:

Figure 4-14 BiaB Message Details



7. Below are listed the acknowledgments that could be sent from the BiaB
   a. Complete
   b. Execution
   c. PayInst
   d. Cancellation
   e. Obligation
   f. Services

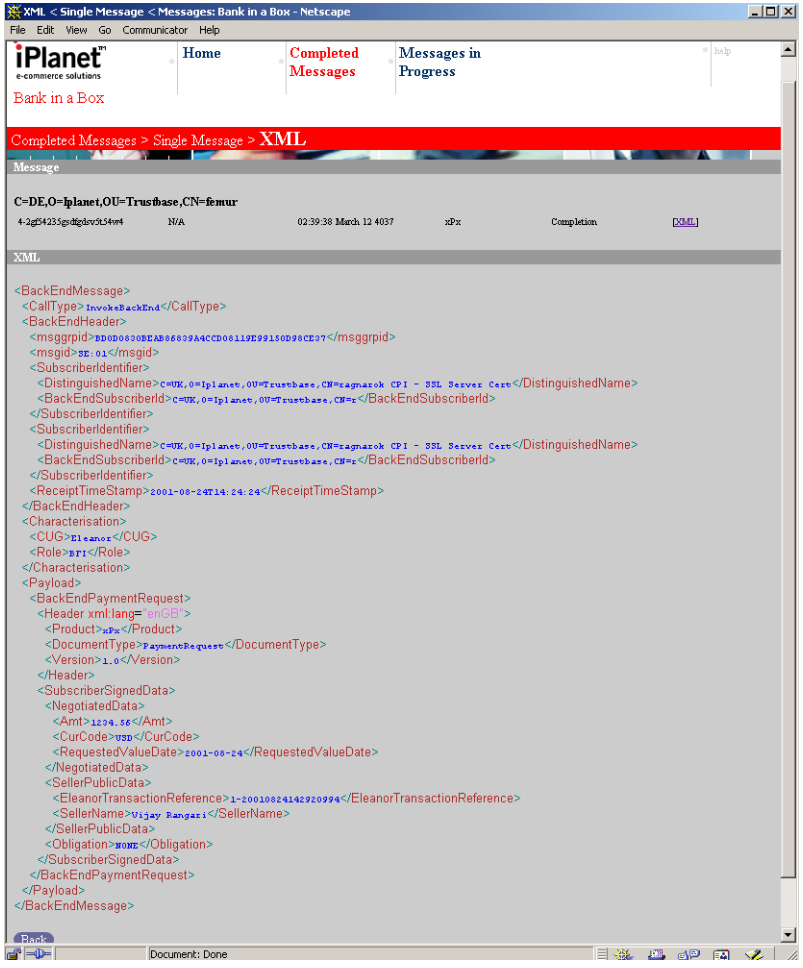
Figure 4-15 Acknowledging a message



Depending on the type of the request message some options may not be available. The precise definitions of each of these options can be found in the Eleanor Technical Specification

8. Each individual message can be viewed in more detail by selecting the <XML> tag

Figure 4-16 An XML Message



9. Options are available to sort and retrieve messages from an index.

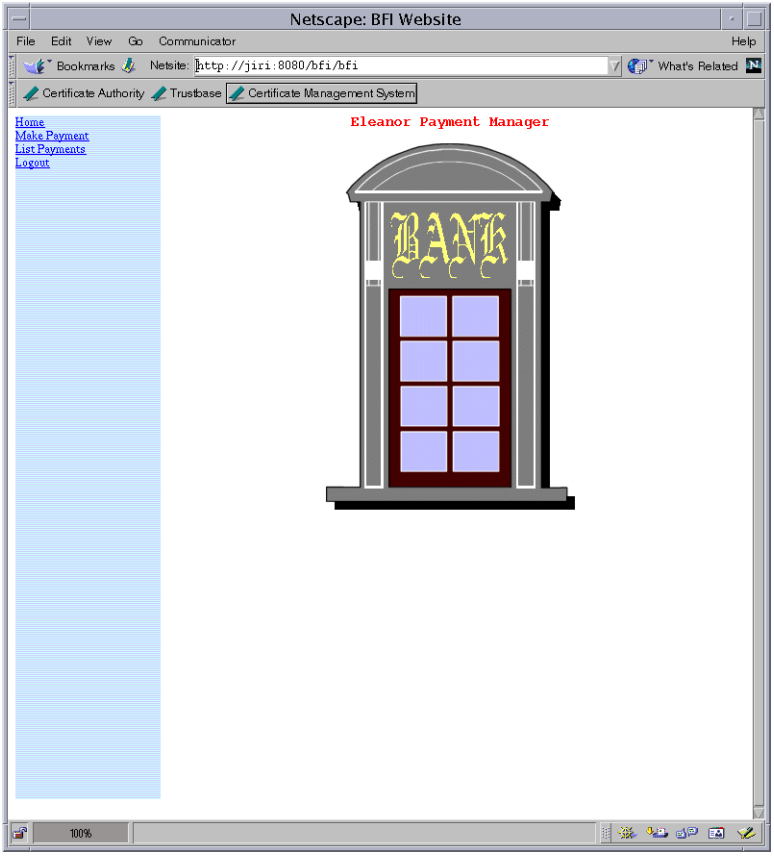
Initiating Payment via Buyers Bank Website

This example is of a Web Site hosted by the Buyer's bank accessed by Buyers who belong to the Eleanor Payment Scheme. It provides the ability for the buyer to initiate payment requests and cancellations directly with its bank.

1. Type in the URL of the Buyer's Bank Website. For example

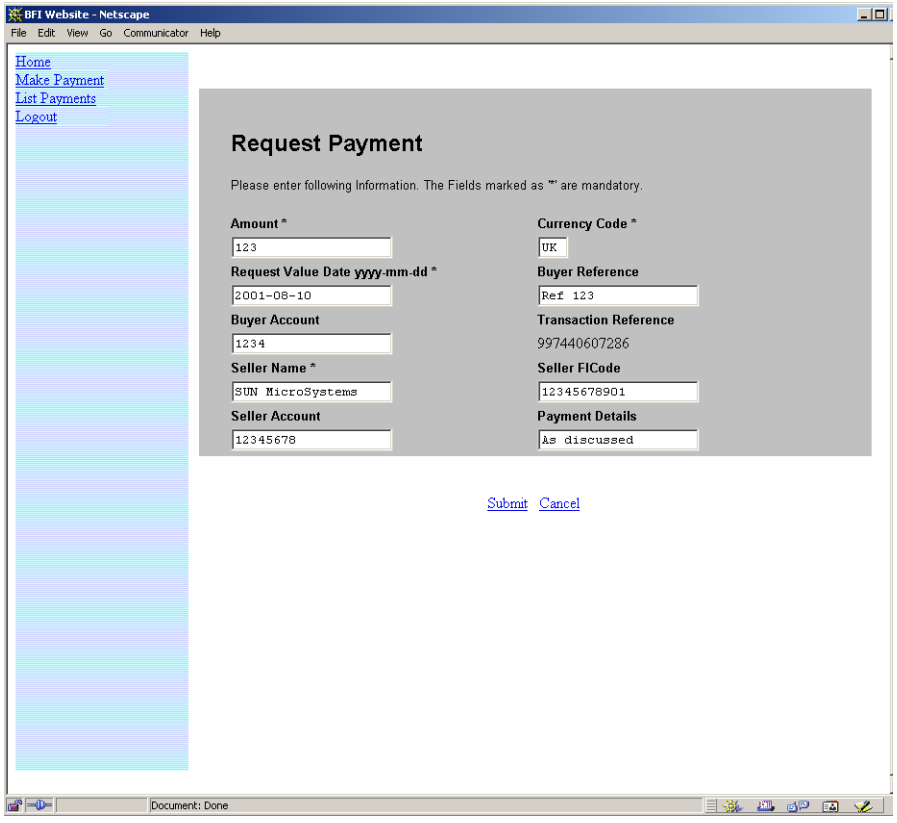
   `http://<server_name>:<port>/<uri_path>/bfi`

Figure 4-17 Buyers Bank Website Homepage



2. Select <Make Payment>

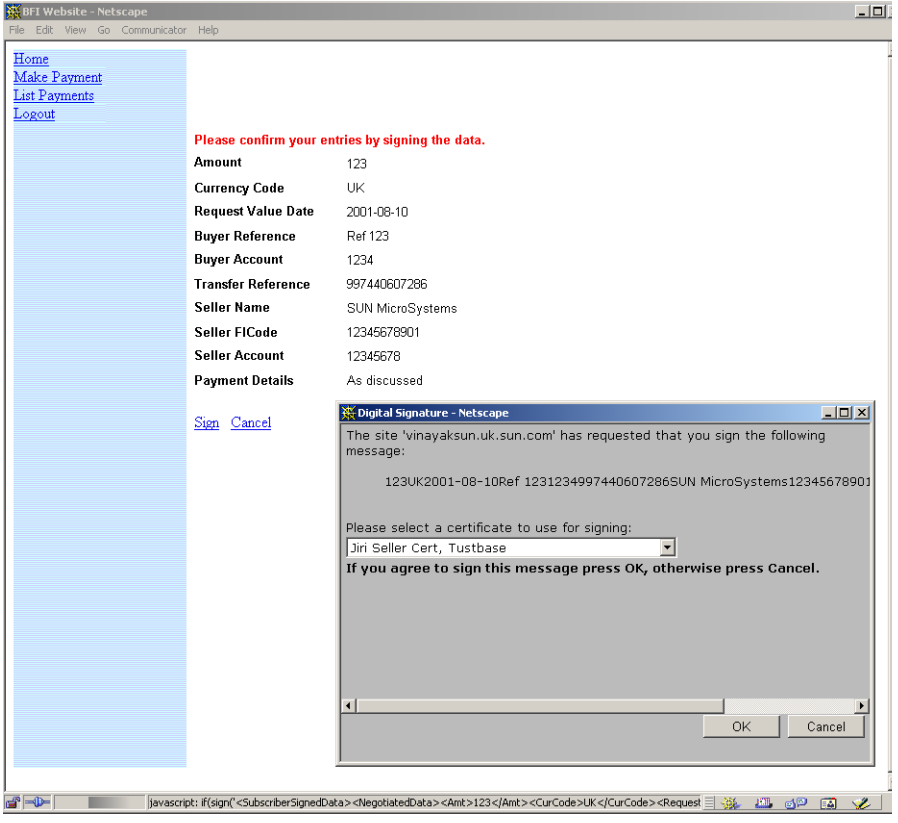
Figure 4-18 Initiate Payment



Details of what each of these fields mean can be found in your payment Scheme Specification

3. Check the details you have entered are correct and sign the payment using your buyers certificate you configured in the previous chapter

Figure 4-19 Sign Payment



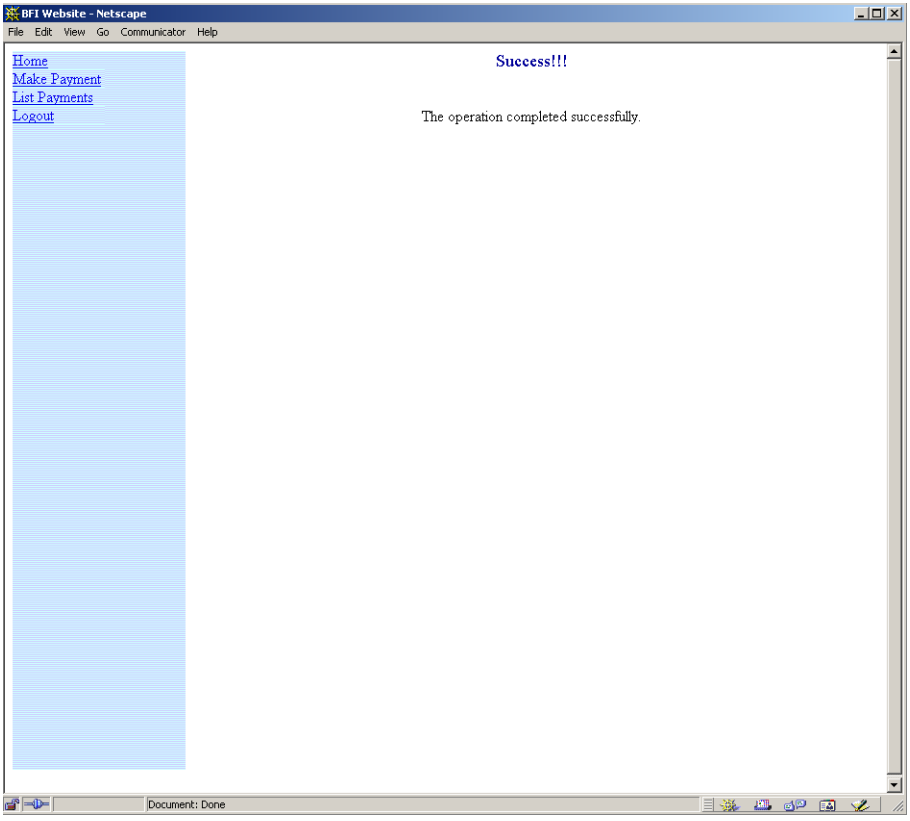
4. Finally when the payment has been initiated a Payment message will be sent to the URL of the Buyers Bank in which you installed iPlanet Trustbase Payment Services on. The following steps take place:

   a. Buyer Website sends Payment to CPI Library located on the Buyers Website Webserver

   b. CPI Library forwards this to iPlanet Trustbase Payment Services

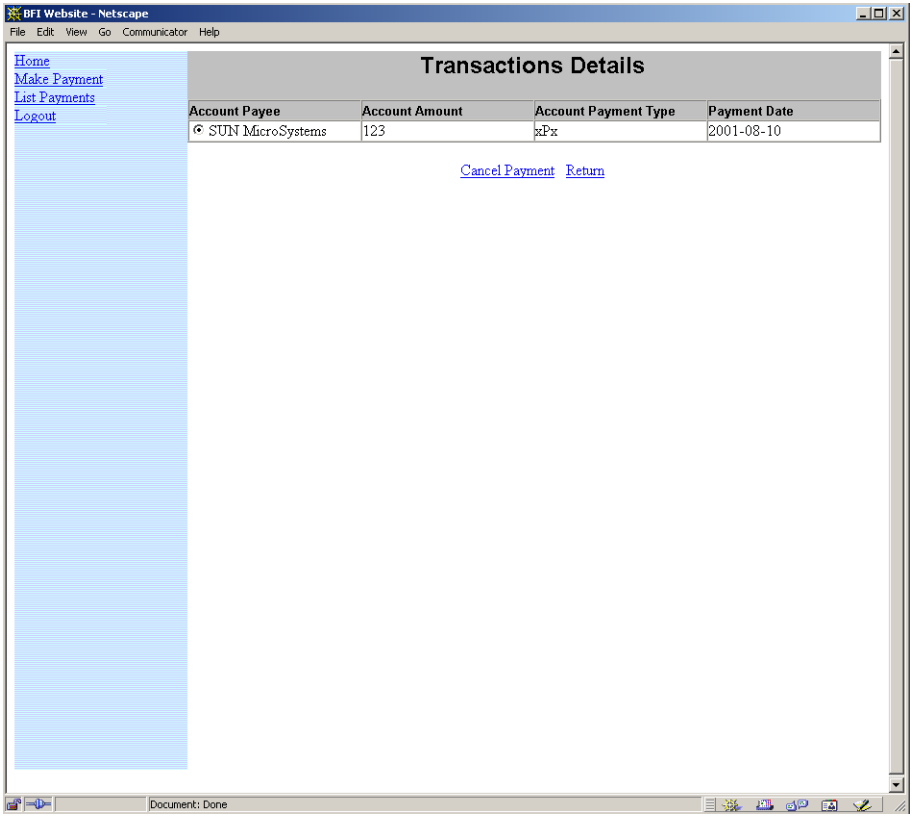
   c. iPlanet Trustbase Payment Services processes the message and forwards the reply to the Buyers CPI Library located on the Buyers Webserver

Figure 4-20 Payment Initiation completed successfully



5. Information appears on the Buyers Screen confirming payment. Select <List Payment> to check the information that you have entered has been processed as a payment.

Figure 4-21 List Payment



Running the CPI Test program

Please refer to “Installing the CPI API,” on page 69

Chapter 5

Interfacing with Existing Systems

This chapter illustrates a typical payment being processed and describes how the payment can be initiated using the iPlanet Trustbase Payment Services Corporate Payment Initiation Library API (CPI). There are three kinds of situations that may warrant the use of the CPI library.

Figure 5-1 Buyer buys something from Sellers Website

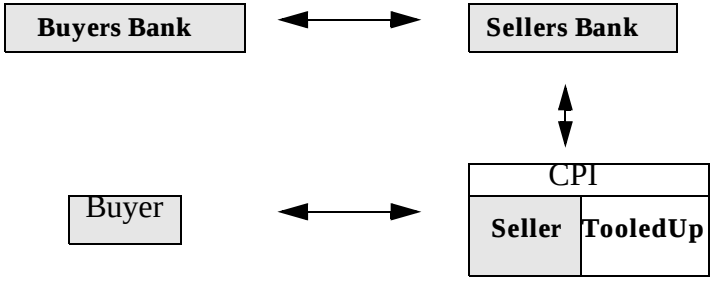


Figure 5-2 Buyer is making a payment with a Sellers signature

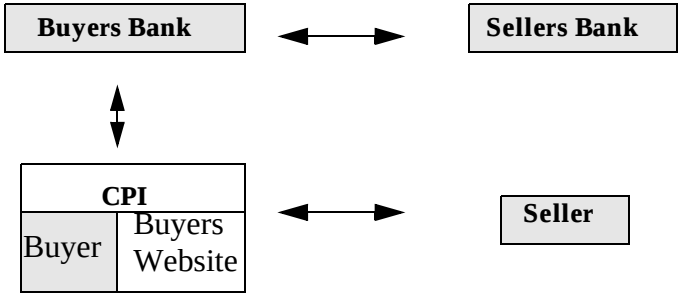
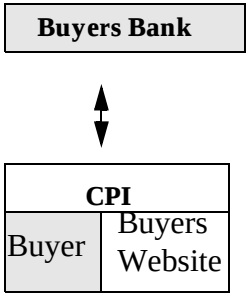


Figure 5-3 Buyer trying to make a payment without the need for the seller's signature



Using the iTPS API to initiate a Payment

The Corporate Payment Initiation Library (CPI) allows a merchant (seller) web site or another application (such as a bank web site) to submit payment requests to a participant (such as the seller / buyer bank).

In order to submit a payment, typically a merchant performs the following steps:

1. Collects payment data from a customer (e.g. from HTML forms submitted to the merchant's web site).
2. Creates an XML element that represents the payment.
3. Sends the XML to the customer for signing
4. Creates any additional XML elements as required (e.g. SellerPrivateData, RemittanceData).
5. Creates a ConfigAdapter. The config adapter allows the merchant application to specify the location (URL) of his financial institution, the certificate to be used for signing payment messages and initialises the SSL subsystem.
6. Optionally defines the transport over which the message is sent using TransportAdapter
7. Submits the XML, signed data and ConfigAdapter to the PaymentInitiator which will construct a signed payment request and send it to the merchant's financial institution.
8. Receives from the PaymentInitiator a Status object, which contains the XML reply content and raw data suitable for logging.

9. Parses the XML reply content to establish the status of the payment request using other API methods e.g. responseReceived()

NOTE Consult your iTPS and iTTM API for more information on this:

com.iplanet.trustbase.initiator

com.iplanet.trustbase.initiator.cpi

com.iplanet.trustbase.initiator.config

com.iplanet.trustbase.initiator.transport

The above iTPS API can be found within the docs directory on your CD-ROM drive. The Trustbase API can be found within the iTTM installation directory or from your CD-ROM drive

Parameters needed to send a message

In order to send a message the following parameters need to be defined

1. role - identifies the role of the customer (for example ‘BU:01’ = Buyer, ‘SE:01’= Seller)
2. doctype - the doctype of the request, for example

   ```
   <!DOCTYPE PaymentRequest PUBLIC \"-//IDENTRUS//ELEANOR PAYMENT
   REQUEST DTD//en"
   "http://www.identrus.com/Eleanor/1.0/ver1.0/PaymentRequest.dtd">
   ```

3. data - the data signed by the signature(s) in the pkcs7 parameter
4. elements - additional XML elements that belong in the request (e.g. Header, RemittanceData).
5. pkcs7 - array of the base64 encoded pkcs7 signed object data, each of which represents the buyer’s signature. If multiple signatures are given, the multipleSignaturesType needs to be one of

   ALL_SIGN_SAME_DATA, SIGN_PRECEDING_SIGNATURES.

   If the pkcs7 array parameter contains one signature, the multipleSignaturesType parameter is ignored. Multiple signatures needs to be either: all sign the same application data (but not sign preceding signatures) or each (except the first) sign the preceding signatures, forming a chain.
6. config - the com.iplanet.trustbase.initiator.config.ConfigAdapterImpl for this operation the following properties needs to be present:
   a. Keystore Domain Space
   b. Signing Certificate
   c. Verification Certificate
   d. Optionally an SSL signing certificate
7. multipleSignaturesType - describes what type the multiple signatures are in the pkcs7 array parameter. This will be one of
   a. ALL_SIGN_SAME_DATA
   b. SIGN_PRECEDING_SIGNATURES.
8. transport - the com.iplanet.trustbase.initiator.TransportAdapter for this operation. This is Optional

Certificate Verification

The CPI packages behavior can be altered by setting properties in the ConfigAdapter. The recognised properties are defined in

`com.iplanet.trustbase.initiator.PropertyCodes`

This package contains details of how the Property codes can be used

How to use the API

The following sample code illustrates how to use the API and illustrate initiating a payment and sending it to iPlanet Trustbase Payment Services. There are two main aspects to this:

- How to build the CPI Library parameters
- How to send a Payment Initiation Message

```
class PaymentInitiatorExample
{
     public Status processPaymentRequest()
     {

// Set up the parameters required to send a PaymentRequest
// The role. In this case we are acting as the seller, hence the role is "SE:01"
          String role ="SE:01";

// The XML DOCTYPE for the message to be sent. The doctype identifies the type of the message,
// and the DTD that defines the structure of the message. In this case we are sending a payment
// request, for which the public ID is -//ELEANOR PAYMENT REQUEST DTD//en and the System ID
// is http://www.identrus.com/Eleanor/1.0/ver1.0/PaymentRequest.dtd.
          String doctype = "<!DOCTYPE PaymentRequest PUBLIC \"-//IDENTRUS//ELEANOR PAYMENT REQUEST
DTD//en\""
                         +" \"http://www.identrus.com/Eleanor/1.0/ver1.0/PaymentRequest.dtd\"> ";

// Get the data describing the payment. This is the data that will be signed by the customer
          String customerData = createCustomerData(getPaymentAmount());

// Get the other elements to be included in the request, such as the Header block.
          String[] elements = getElements();

// Request that the customer signs the payment request. This signature is performed by the
// smartcard plugin in the customer's browser.
          String[] customerSig = getCustomerSignature(customerData);

// The signature type we are using. This defines how signatures are applied if there
// are multiple signatures in the customerSig array. In this example code there is a single signature
// from the customer, so the parameter is not used regardless of the value being set to
// PaymentInitiator.ALL_SIGN_SAME_DATA as per below
          int multipleSignatureType = PaymentInitiator.ALL_SIGN_SAME_DATA;

// Get the config adapter for this request. The config adapter is the means by which the
// PaymentInitiator class accesses the underlying PKI and properties of the system.
          ConfigAdapter config = getConfig();

// The transport adapter defines the transport over which the message is sent. Here we
// will use HTTP
          TransportAdapter transport = new HTTPTransportAdapter();

// Get an instance of the Payment Initiator,
```

```
          PaymentInitiator pi= PaymentInitiatorManager.getPaymentInitiator();


// Send the request to the Payments server using the information gathered above.
          Status status = pi.send(role,
                                        doctype,
                                        customerData,
                                        elements,
                                        customerSig,
                                        multipleSignatureType,
                                        config,
                                        transport);

// Retrieve the response data from the Payments Server check to see if the response is true or false
          if (status.responseReceived())
          {
// Retrive the actual content as a String object
               String response = status.getContent();

               // Business logic to deal with the response
               // ...
               // ...
               // ...
          }
          else if   (status.hasConnectionFailed())
          {
               // No response - log an error
          }
          else
          {
               // fall through error handling
          }
     }


     public ConfigAdapter getConfig()throws Exception
     {
// The default implementation of ConfigAdapter works from information supplied in a Properties
// object. Certificates are retrieved from a file on disk.

          Properties props = new Properties();

props.put(PropertyCodes.INITIATOR_KEYSTORE_DOMAIN_SPACE,"mykeystore");
props.put(PropertyCodes.INITIATOR_KEYSTORE_SIGNING_CERTIFICATE,"server-cert);
props.put(PropertyCodes.INITIATOR_KEYSTORE_VERIFICATION_CERTIFICATE + ".1","IdentrusRoot");
return new ConfigAdapterImpl(props);
     }

     public String createCustomerData(String paymentAmount)
     {
// Construct the CustomerSignedData XML block for the customer to sign.
// There are a number of ways of constructing this block, including the
```

```
// iPlanet JAXHIT technology.  There are a number of advantages to using
// this technology from iPlanet, such as type checking and XML validity checking.
// For brevity we will just create the block as a String in this example.
// The values here are hard coded for example only and would typically be
// supplied by the calling Seller application

          // The ISO currency specified in the Eleanor Tech Spec, here we're using US $
          String currency = "USD";

          // The valueDate is taken from the transaction attributes agreed between Buyer and
          // Seller
          String valueDate = "2001-07-25";

          // The unique Eleanor ID for the request as per the Eleanor Tech Spec
          String eleanorRef = this.getEleanorRef();

          // The name of the seller company (i.e. us)
          String sellerName = "ACME Soap Co";

          // The type of obligation. In this case it is "NONE", as we are sending a plain revocable
          // payment request
          String obligation = "NONE";

          String customerData = "<SubscriberSignedData><NegotiatedData><Amt>"
                                      +paymentAmount
                                      +"</Amt><CurCode>"
                                      +currency+
                                      "</CurCode><RequestedValueDate>"
                                      +valueDate+

"</RequestedValueDate></NegotiatedData><SellerPublicData><EleanorTransactionReference>"
                                      +eleanorRef+
                                      "</EleanorTransactionReference><SellerName>"
                                      +sellerName
                                      +"</SellerName></SellerPublicData><Obligation>"
                                      +obligation+
                                      "</Obligation></SubscriberSignedData>";
          return customerData;

     }

     public String getElements()
     {
          // Construct the header block for the message
          // Eleanor product code as defined in the Eleanor Specification. "xPx" is the code
          // for Payment Request
             String product = "xPx";

          // The XML doc type (the type of the root element of the message)
             String docType = "PaymentRequest";

          // The version
```

```
            String version = "1.0";

        // Build the header block
            return new String[]{"<Header xml:lang=\"enGB\"><Product>"
                                    +Product
                                    +"</Product><DocumentType>"
                                    +DocType
                                    +"</DocumentType><Version>"
                                    +Version
                                    +"</Version></Header>"};
    }


    private String getEleanorRef()
    {
// return a unique Eleanor reference number as per the Eleanor Tech Spec
    }

    public String getPaymentAmount()
    {
// Get the payment amount attribute from the transaction
    }

    public String[] getCustomerSignature(String customerData)
    {
        // Request that the customer signs the customerData with their smartcard.
        // ...
        // ...
        // ...
    }
}
```

Test.java

The full source for this worked example can be found in:

```
cdrom/cdrom0/iTPS/cpi/cpi.tar
```

The script to run test.java can be found in:

```
<cpi_install_dir>/bin/test.sh
```

The source can be found in:

```
<cpi_install_dir>/example/com/example/example1/Test.java
```

Glossary

Asynchronous Message (See also Eleanor Technical Specification and also Synchronous message) means that messages can be sent over a delayed time frame. Following a request, a response can be sent at any time. Whereas with a Synchronous message following a request the system waits until it gets a response.

Bank Name The name of the Scheme Member to whom this entry relates.

BIC-11 is an 11 digit Bank Identifier Code. The Bank Identifier Code is a unique address which, in telecommunication messages, identifies precisely the financial institutions involved in financial transactions. The BIC Directory Site can be found at http://www.bicdirectory.swift.com/.

BFI (See also Eleanor Technical Specification) The Buyers Bank or in Identrus terminology the IP.

BFIM (See also Eleanor Technical Specification) This is the model used for making a payment. The instruction is submitted by the submitting Corporate through the Buyer Financial Institution.

Biab Bank in a Box

Buyer The Buyer wants to purchase goods or services from the Seller. The Buyer is in possession of an Identrus smartcard. The Buyer is granted access to the payment facility by his bank. This is also abbreviated as BU in the Eleanor Scheme and as the SC in the Identrus four corner model

Buyers Bank The Buyer's Bank manages the Buyer's account from which payment is to be made. The Buyer's Bank issues the Buyer with his Identrus smartcard. The Buyer's Bank is a member of the Identrus schemes.

cache A cache is a 'local' store used to hold recently accessed information, so that if further access is required a local copy may be used rather than a 'remote' copy. In most schemes this specifically refers to a local store containing copies of responses from the Identrus Root to certificate status checks.

Cancellation A cancellation is the revocation of a Payment Request. It may only be applied to a Payment Request, not the payment. Therefore payments that have been paid cannot be cancelled by this method.

CPI Corporate Payment Initiation API Library

Customer For the purposes of this document this term describes a customer of a Member.

FI Code A BIC-11 code owned by the Scheme Member to whom this entry relates.

Identrus four-corner Model This describes the four parties involved in an Identrus enabled transaction, namely two banks and their respective customers. Where two customers have the same bank, only three parties are involved, and this is known as a 'three party' model.

Identrus Root The Identrus Root manages membership of the Identrus scheme. The Buyer's Bank and Seller's Bank verify each other's identity via the Identrus Root.

Issuing Participant (IP) In the Identrus four-corner model this refers to the level 1 Identrus participant that issued the certificate to the Subscribing customer. This equates to "Buyer's Bank" in other payment transaction

iWS iPlanet Web Server

iAS iPlanet Application Server

iTTM iPlanet Trustbase Transaction Manager

iMQ iPlanet Message Queue

iTPS iPlanet Trustbase Payment Services

JMS is an API. Many implementations of JMS are available from different providers (like IBM MQ and iMQ) and any of them can be used with iTPS

Membership The current status of the institution concerned. This can take the values Pending, Active, Suspended, Terminated.

Relying Customer (RC) In the Identrus four-corner model this is the organisation that relies on the identity of the Subscribing customer in order to conduct trade. This equates to 'Seller' in a payment transaction.

Relying Participant (RP) This refers to the Relying Customer's bank. This equates to "Seller's Bank" in a payment transaction.

Service The current status of a service. This is used to mark temporary service interruptions. The allowed values are Available and Unavailable.

Seller The Seller wants to supply goods or services to the Buyer. The Seller is in possession of an Identrus private key. The Seller is granted access to the payment facility by his bank. This is also abbreviated as SE in the Eleanor Scheme and as the RC in the Identrus four corner model.

Sellers Bank The Seller's Bank manages the Seller's account into which payment is to be made. The Seller's Bank is a member of the Identrus schemes.

SFI (See also Eleanor Technical Specification) The Sellers Bank or in Identrus terminology the RP

SFIM (See also Eleanor Technical Specification) This is the model used for initiating a payment.The instruction is submitted by the submitting Corporate through the Seller Financial Institution. This is essentially the same as the Identrus four corner model.

Scheme Register (See also Eleanor Technical Specification) The Eleanor Scheme has membership criteria. Each participating Financial Institution ensures that corresponding institutions are members of the scheme. It does so via an XML message that is referred to as a scheme register.

Subscribing Customer (SC) In the Identrus four-corner model this is the end customer of the Identrus enabled service. This equates to the 'Buyer' in a payment transaction.

Synchronous Message (See also Eleanor Technical Specification) See Asynchronous Message.(See also Eleanor Technical Specification and also Synchronous message) means that messages cannot be sent over a delayed time frame. Following a request, a response is sent immediately before the system can proceed. Whereas with an asynchronous message, following a request, a response can be sent at anytime.

Timeout is the amount of time (in seconds) for which a client will wait for a server to respond before dropping the connection and returning with a timeout error. In the context of these screens, it is the amount of time the Seller's Bank will wait for the Buyer's Bank to respond.

Transaction This refers to a complete end to end transaction, potentially involving multiple request-response pairs. Where a single message in such a transaction is referred to the term 'message' is used.

Transaction Reference (See also Eleanor Technical Specification) The provision and handling of transaction references is key to allowing the Eleanor Payment products to be used effectively from third party systems and to assist in reconciliation of data

Index

G

H

I

J

L

M

O

P

R

S

T

U