Oracle® Exchange

Punchout Guide

Release 6.2

September 2001

Part No. A92190-01

ORACLE®

Exchange, Release 6.2

Part No. A92190-01

Primary Author: Richard Sears, Vic Mitchell

Contributing Authors: Warren Perkinson, Sanjay Bhasin, Sam Andrus.

Contents

3 Defining an Oracle iProcurement Punchout

A Oracle Exchange Punchout DTDs

B Oracle iProcurement Punchout DTDs

C Sample Oracle iProcurement XML Documents

Index

Send Us Your Comments

Oracle Exchange Punchout Guide, Release 6.2

Part No. A92190-01

Oracle Corporation welcomes your comments and suggestions on the quality and usefulness of this document. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most?

If you find any errors or have any other suggestions for improvement, please indicate the document title and part number, and the chapter, section, and page number (if available). You can send comments to us in the following ways:

- Electronic mail: richard.sears@us.oracle.com
- FAX: (650) 506-7816 Attn: Oracle Exchange
- Postal service:
  Oracle Corporation
  Oracle Exchange Documentation
  300 Oracle Parkway, 3OP16
  Redwood Shores, CA 94065
  USA

If you would like a reply, please give your name, address, telephone number, and (optionally) electronic mail address.

If you have problems with the software, please contact your local Oracle Support Services.

Preface

This manual describes how to use the punchout capability of Oracle Exchange to provide Exchange members and iProcurement users punchout access to supplier-hosted catalogs.

Intended Audience

This manual is intended for suppliers who wish to set up punchout access to their local catalogs and iProcurement managers who wish to enable and control punchout capability from an iProcurement instance.

Structure

This manual describes the following:

Chapter 1 Using Oracle Exchange Punchouts

This chapter explains punchout technology and describes the different punchout models and the benefits of each.

Chapter 2 Defining an Oracle Exchange Punchout

This chapter describes the Oracle Exchange to supplier punchout model and explains the process a supplier uses to implement this type of punchout.

Chapter 3 Defining an Oracle iProcurement Punchout

This chapter explains the different punchout models supported by Oracle iProcurement and how to implement each.

Appendix A Oracle Exchange Punchout DTDs

This appendix contains examples of the DTDs used when executing Oracle Native XML punchout transactions from Oracle Exchange.

Appendix B Oracle iProcurement Punchout DTDs

This appendix contains examples of the DTDs used when implementing an XML punchout from Oracle iProcurement.

Appendix C Sample Oracle iProcurement XML Documents

This appendix conatins example XML documents used during a punchout from Oracle iProcurement (sample XML documents used for native XML punchout from Oracle Exchange are contained in Chapter 2).

Related Documents

For additional information on installing and implementing Oracle Exchange and Oracle iProcurement, see the following manuals:

- *Oracle iProcurement for Release 11i Installation Guide*
- *Oracle iProcurement for Release 11i Implementation Guide*
- *Oracle Exchange Installation Guide*

1

Using Oracle Exchange Punchouts

While procurement systems and online exchanges typically offer the ability to store information about their suppliers' items or services locally, the task of defining and maintaining these local catalogs is both time consuming and, in the case of some products that are continually changing, extremely difficult. Additionally, some items and services are not well suited to this hosted catalog model (for example books, where the catalog changes frequently). Yet Exchange suppliers and procurement system content providers know that their users still want the ability to purchase such items. The problem is how to allow access to all the items but without the high cost overhead of having to maintain catalogs locally. The solution is punchout technology.

Topics covered in this chapter include:

- Overview of Oracle Exchange Punchouts
- Punchout models
- Benefits comparison of the punchout models
- References
- Software requirements

Overview of Oracle Exchange Punchouts

Punchout enables buyers to connect to the supplier's catalog and search for items directly on the supplier's site. When an item or items are identified, the item information is returned to the buyer's system to be added to any additional items being purchased. Punchout benefits both the supplier and the buyer by allowing suppliers to maintain and host their own catalog information, while buyers can search for items from within their own ERP system, procurement software, or online exchange purchasing operation.

Additionally, a supplier can utilize the large customer base existing within an Oracle Exchange to increase the exposure that their products receive while reducing the configuration effort required on the part of the supplier. A supplier can define a punchout on an Oracle Exchange once and all Exchange members and Oracle iProcurement users on that exchange can use this punchout definition to access that supplier directly.

Using the punchout capability provides the following:

- Exchange members have access to a broader range of potential products and services not just products whose information is stored locally on the exchange.
- Self service requisitioners within a buying organization are able to search on both the exchange and supplier sites for products which meet their requirements while Purchasing Managers can control contract leakage be ensuring that only items from approved suppliers are purchased.
- Suppliers can utilize the large customer base existing within an Oracle Exchange to increase the exposure for their products while reducing their configuration and maintenance effort. A supplier can define a punchout once and all members accessing this exchange can use this punchout definition to access the suppliers' catalogs.

Punchout Benefits

Products which are configurable or include highly variable or dynamic pricing are difficule to maintain within a static, hosted catalog environment. These types of items are particularly well suited to a punchout as this allows the supplier to control the configuration and pricing.

Catalogs which are continually changing are costly and time-consuming to maintain in a hosted environment. These types of catalogs are more accurately maintained by the supplier to ensure that the latest catalog content and pricing are available.

Obtaining information directly from a supplier using a punchout ensures that the product information is accurate and up-to-date, which eliminates downstream inefficiencies due to incorrect product or pricing information . The burden of maintaining the hosted catalog is removed from the buying organization, which reduces both catalog maintenance and data storage costs. Products which previously had been difficult to maintain within a locally hosted catalog can now be purchased from within the buyer's ERP system, procurement software, or directly from the exchange.

Punchout does not suit every product or supplier however. By allowing the supplier to host catalog information locally, the buyer or buying organization must work closely with the supplier to control the content and pricing of products.

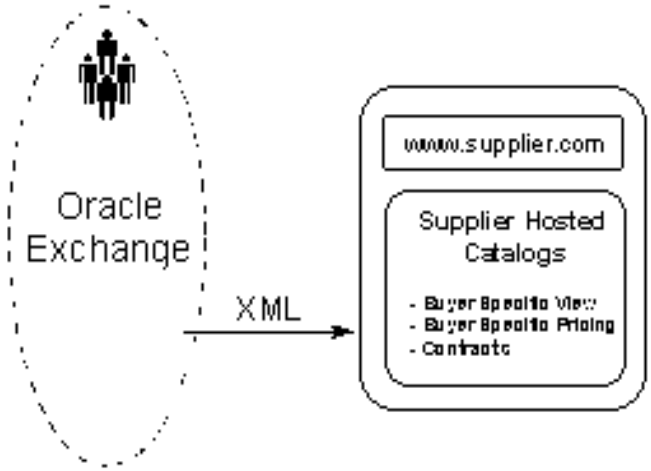
Table 1-1 Supplier Hosted versus Locally Hosted Catalogs

Supplier Hosted Catalogs	Locally Hosted Catalogs
Best suited for products requiring a high degree of configuration (such as computer hardware), specialized services, or products with fluctuating prices.	Best suited for commodity-type items (such as office supplies), products with pre-negotiated or stable prices, MRO items, or products built to a predetermined specification (such as mass produced mechanical parts).
Supplier closely manages the content presented to buyers.	Supplier avoids managing a complex web site and web content.

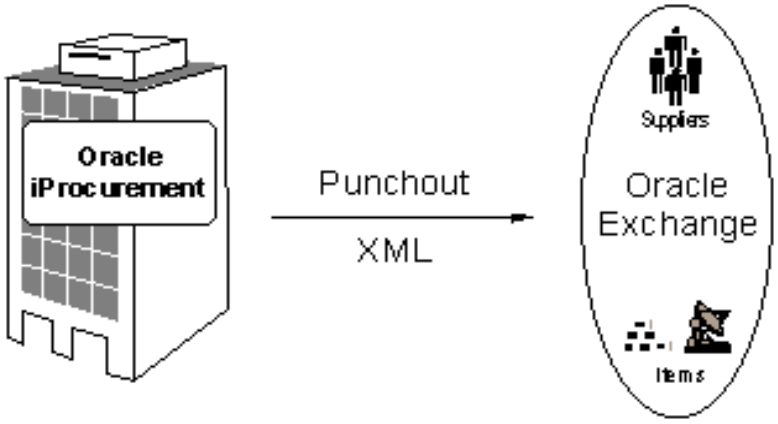
Punchout Models

Oracle Exchange supports various punchout models allowing both suppliers and buyers to provide and obtain goods and services most efficiently.

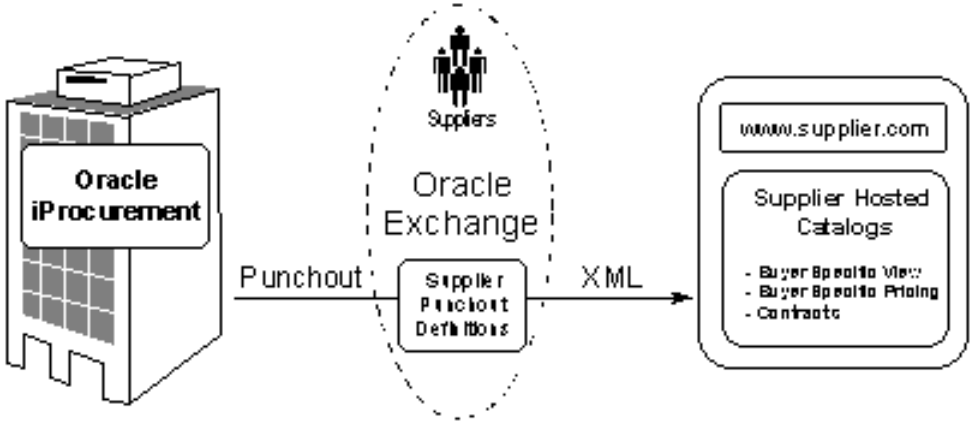
Model 1: Oracle Exchange punchout to Supplier-hosted catalogs.



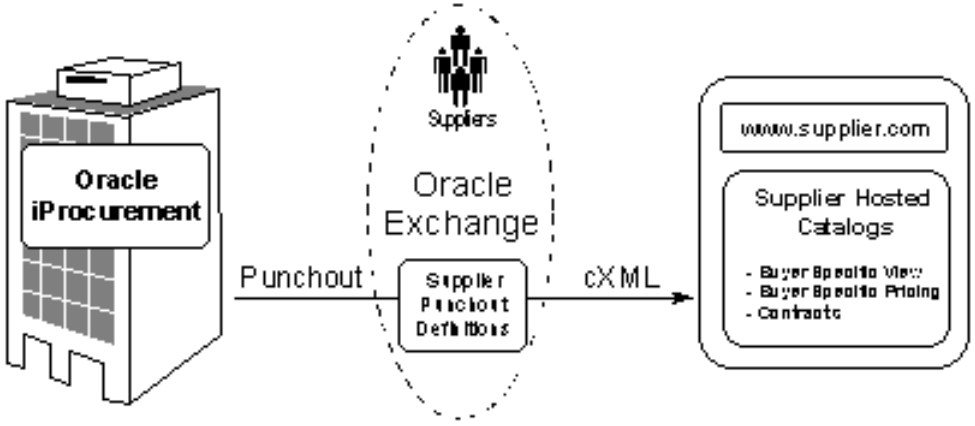
Model 2: Punchout from Oracle iProcurement to Oracle Exchange in XML format



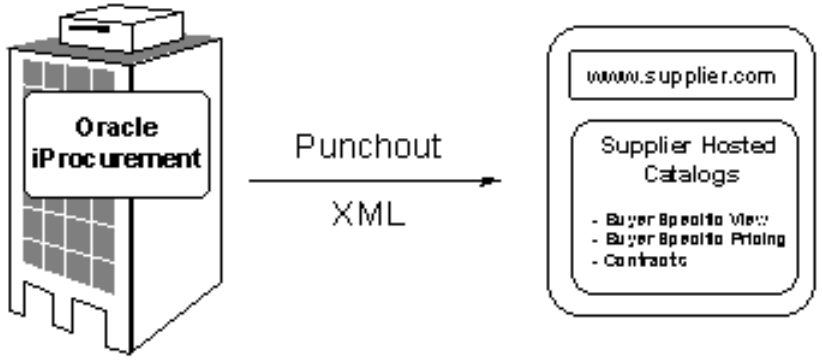
Model 3: Punchout from Oracle iProcurement to a Supplier-hosted catalog via Oracle Exchange using XML format



Model 4: Punchout from Oracle iProcurement to a Supplier-hosted catalog via Oracle Exchange using cXML format



Model 5: Punchout from Oracle iProcurement directly to a Supplier-hosted catalog using XML



Benefits Comparison of Punchout Models

Model	Buyer Benefits	Supplier Benefits	Comments
1 Oracle Exchange punchout to supplier-hosted catalogs	Buyer members automatically have access to centralized collection of punchouts simply by registering with Exchange.	Suppliers define punchouts once to Exchange and are then visible to potentially all Exchange members.	Suppliers can control access to their punchouts. Purchasing Managers and Company Admins can control buyer access.
2 iProcurement punchout to Oracle Exchange hosted catalogs	iProcurement requisitioners have access to all supplier catalogs defined on Exchange.	As soon as a supplier defines a catalog to Exchange, iProcurement users have access to it. Catalogs are also available to registered Exchange users.	
3/4 iProcurement punchout to supplier hosted catalogs via Oracle Exchange	iProcurement users have access to centralized collection of punchouts.	Suppliers only have to define their punchouts on the Exchange once. An individual configuration within each iProcurement instance is not required.	Punchout can be via Oracle native XML or cXML
5 iProcurement punchout directly to supplier hosted catalogs	eContent managers can control their users' access to punchouts from within iProcurement.	Supplier can control access by determining who obtains the punchout information. Any punchout changes must be communicated to all buyer organizations.	Available using XML format only.

References

Oracle iProcurement for Release 11*i* Installation Guide

Oracle iProcurement for Release 11*i* Implementation Guide

Oracle Exchange Implementation and Setup Guide

Software Requirements

Oracle Exchange 6.1, Patchset C or above

Oracle iProcurement Release 11*i*, Patchset J or above

2

Defining an Oracle Exchange Punchout

This chapter explains how to implement a punchout from Oracle Exchange to a supplier. Topics in this chapter include:

- Introduction to Oracle Exchange punchout
- Functional overview of Oracle Exchange punchouts
- Steps to perform on Oracle Exchange
- Steps to perform on your web site
- Steps to configure your punchout definition
- Oracle Exchange punchout setup checklist

Introduction

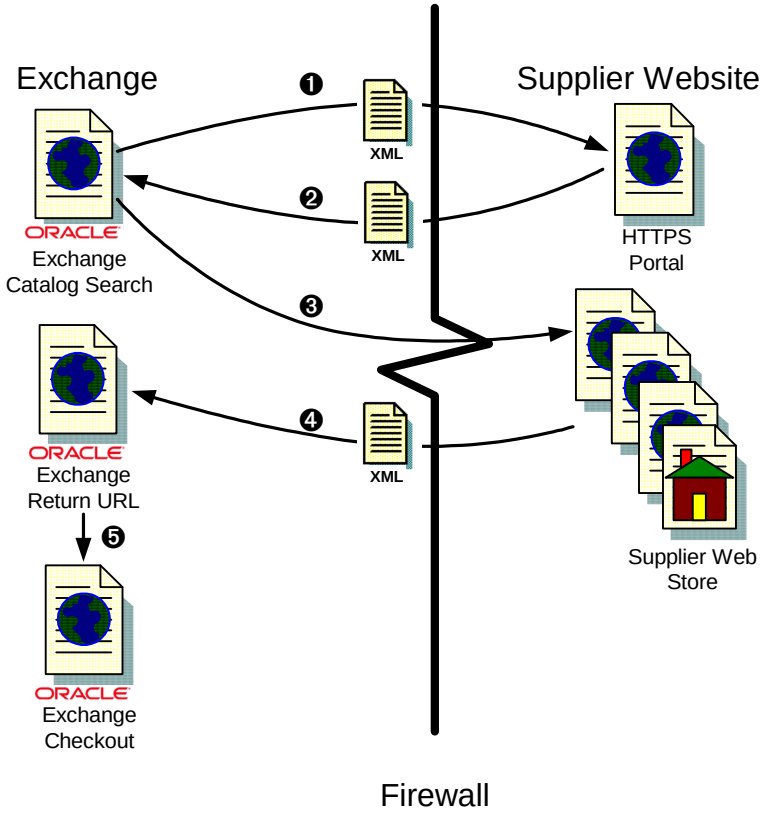
Suppliers using Oracle Exchange typically choose to host their catalogs of goods and services physically on Exchange. This relieves them of many of the catalog management tasks. If you wish, however, to still host your own web catalog and yet allow that catalog to be accessible to customers browsing Exchange, you can set up a punchout from the Exchange to your supplier-hosted catalogs.

While buyers are browsing the Exchange, they enter keywords to search for particular products and services. The Exchange search engine will present a list of items which match the keywords entered by the buyer. If items from your supplier-hosted web catalogs match the search criteria, the buyer is presented with a link from the Exchange catalog to navigate to your supplier-hosted catalog.

When the buyer completes selecting items to purchase, your supplier-hosted site returns the shopping cart contents to the Exchange to complete the buyer’s transaction.

Oracle Exchange uses XML transactions to pass information between Exchange and your supplier-hosted site. Exchange supports both XML and cXML protocols for exchanging information.

Functional Overview



1. When the keywords you defined when you defined your punchout on the Exchange meet the search criteria of a buyer shopping on Exchange, a link to your site is displayed.
2. When the buyer clicks the link to your site, the Exchange will post an XML or cXML document to the URL address that you specified for your web catalog. This document will include among other things, a connection password, the buyer's Exchange username, the buyer's company name, and a return URL

   For examples of this document in both XML and cXML, see "Process the incoming Punchout Request/PunchOutSetupRequest transaction" later in this chapter.

3. Your site must be able to process this XML or cXML document, validate that the password is correct, create an HTTP session to handle the request, and respond with a document specifying a destination URL and session information.

   The session information is important for two reasons.

   - If required, it will allow you to provide buyer-specific information such as unique or pre-selected configurations, buyer-specific pricing or products, or buyer-specific advertising or links.
   - It will allow your site to recognize buyers as having linked to your site from Exchange, and to return the buyers back to the Exchange along with the contents of their shopping carts once they have completed selecting their items.

- For examples of this document in both XML and cXML, see "Provide an appropriate Punchout Request Response/PunchOutSetupResponse transaction" later in this chapter.

4. The Exchange will redirect the buyer's browser to the URL passed to it from the your site to allow the buyer to continue to shop on your web catalog.

5. When the buyer finishes shopping and is ready to return to the Exchange, your site must post an XML or cXML document containing all the selected items in the shopping cart back to the URL the Exchange passed in the initial call to your site.

   For examples of this document in both XML and cXML see, "Return the XML Shopping Cart/PunchOutOrderMessage information" later in this chapter.

6. Once the buyer has returned to the Exchange, additional purchases can be made, or the buyer can proceed to checkout.

Steps to Perform on Exchange

Step 1: Register your company on Exchange

To enable buyers on Exchange to access your site, you must exist as a registered company on the Exchange (if you are already registered with Exchange, no additional or special type of registration is necessary).

To register with Exchange:

1. Navigate to the URL of the Exchange with which you wish to participate.
2. From the Marketplace home page, click the "Sign me up!" link.
3. On the **Exchange Registration** page, click the appropriate radio button selection, and then click Proceed to Registeration.
4. Use the online help pages available by clicking the Help icon at the top of the page to guide you through the remainder of the registration process.

Step 2: Create code mappings

If you plan to integrate the Exchange with your ERP system and web catalog, it is necessary to map the codes which are used within Exchange to the codes that are used for items in your catalog. This allows Exchange to recognize codes that are passed to it by your site when users have finished selecting the items they wish to purchase.You need to create mappings for currency codes and unit of measurement codes.

Note that Currency Code and Unit of Measurement are only two of the codes which can be mapped. Mapping values you specify are also used for other inbound and outbound XML documents passed between Exchange and your company.

You must be signed on to Exchange as a Company Administrator to define these mappings.

Map currency codes

1. Sign on to Exchange using a signon with Company Admin authority. If you just registered your company in the previous step, the signon you used will have Company Admin authority
2. From the Marketplace homepage, click the "Company Admin" link, then click the "Application Setup" tab. The **Application Setup** page appears.

3. Click the “Data Mapping” link. The **Data Mapping** page appears.
4. Click the Currency Code radio button and then click the “Edit Mapping” button. The **Edit Mapping (Currency Code)** page appears.
5. Display the currency code you wish to map by entering the code in the “Currency Code Search” datafield and clicking the “Go” button, or by clicking the letter link that corresponds to the first letter of the code. The resulting display shows all currency codes that matched your search criteria.

Description	Exchange Value	My Company to Exchange Value	Exchange to My Company
Text description of the currency code	The value used internally by Exchange to represent this currency	The value you will pass to Exchange	The value Exchange will forward to you if sending a transaction from the Exchange (not used for Punchout)

If your company does not use the code shown in the second column, enter the code you use in the third column.

For example, your company may use US as the code for US dollars while Exchange uses USD for US dollars. In this case, you would enter US in the third column’s datafield to define the correct mapping.

After you have specified all your code mappings, click the “Apply Changes” button.

Map unit of measurement codes

In addition to mapping your currency codes, you must also map the unit of measurement codes. You must map all unit of measurement codes that will be passed between your company and Exchange. The process to map unit of measurement codes is similar to that of mapping currency codes.

1. Sign on to Exchange using a signon with Company Admin authority.
2. From the Marketplace homepage, click the “Company Admin” link. Click the “Application Setup” tab. The **Application Setup** page appears.
3. Click the “Data Mapping” link. The **Data Mapping** page appears.
4. Click the Unit of Measurement radio button and click the “Edit Mapping” button. The **Edit Mapping (Unit of Measurement)** page appears.

5. If you do not see the unit of measurement code you wish to map, enter the code in the "Unit of Measurement Search" datafield and click the "Go" button, or click the letter link that corresponds to the first letter of the code. The resulting display shows all unit of measurement codes that match your search criteria.

Description	Exchange Value	My Company to Exchange Value	Exchange to My Company
Text description of the unit of measurement code	The value used internally by Exchange to represent that unit of measurement	The value you will pass to Exchange	The value Exchange will forward to you if sending a transaction from the Exchange (not used for Punchout)

If your company does not use the code shown in the second column, enter the code you use in the third column.

For example, your company may use CSE as the code for case while Exchange uses CS for case. In this case, you would enter CSE in the third column's datafield to define the correct mapping.

After you have specified all your code mappings, click the "Apply Changes" button.

Steps to Perform at Your Web Site

Exchange uses XML and cXML-based transactions when communicating with host sites so you will need to modify your site to support this processing. This includes installing an XML parser if one is not already in place and storing the DTD specifications for the transactions involved.

The complete set of steps includes:

1. Install an XML parser.
2. Create and store the necessary DTD specifications for the transactions to be used.
3. Create a new URL to handle the incoming punchout request.
4. Create the code necessary to process and generate the necessary XML or cXML documents to communicate with Exchange.
5. Determine the appropriate method for redirecting buyers back to the Exchange and create the code necessary to accomplish this.

Prerequisites

- You have decided how you want to validate user access.
- You have decided how you want to redirect users to Exchange.

Step 1: Install an XML Parser

Communication between Exchange and web catalogs makes extensive use of XML processing. You must have an XML parser tool installed to decode the XML documents passed to you. You can use Oracle XML Parser for Java, Apache AML Xerces Java Parser®, or another commercially available parser.

To download either the Oracle or Apache XML parsers contact one of the sites below. Use the instructions on the site's web page when downloading and installing the parser.

To download the Oracle XML Parser software and documentation and installation instructions see:

`http://technet.oracle.com/tech/sml/parser_java2/`

To download the Apache XML Parser for Java documentation and software see:

`http://xml.apache.org`

Step 2: Create .dtd files for use with your XML Parser

XML parsers can use definitions contained in .dtd files to test the validity and completeness of data contained in an .xml file.

There are three XML transactions that are used when communicating between a web catalog site and Exchange. The name of the transaction depends on the protocol you are using, either XML or cXML.

Table 2–1 Oracle Native XML and cXML Transaction Names

Oracle Native XML Name	cXML Name
XML Punchout Request	PunchOutSetupRequest
XML Punchout Response	PunchOutSetupResponse
XML Punchout Shopping Cart	PunchOutOrderMessage

The XML .dtd specifications used for validating the Oracle Native XML transactions are included in Appendix B. You can download the DTD specifications for the cXML transactions from
`http://www.cXML.org/.`

Either way, you must ensure the appropriate dtd's are used when parsing the inbound transaction and generating the outbound transactions.

To create the appropriate .dtd files:

1. If using the Oracle Native XML protocol, create a file for each of the .dtd examples described in Appendix B. You can create these files using Notepad or another text editor, but save the files with an extension of .dtd.

   If using the cXML protocol, download the specifications for each transaction and create the corresponding .dtd. The cXML specifications can be downloaded at `http://www.cXML.org/`.

2. Place the .dtd files in the directory required by your particular XML parser. For example, if you store the dtd for the inbound XML Punchout Request in the `/etc/templates/` directory as file `punchreq.dtd`, the Java code to parse the incoming request may include the following:

```
url = new URL
("file://localhost/etc/templates/punchreq.dtd");
urlConnection = url.openConnection();
InputStream urlinstr = urlConnection.getInputStream();
dp.parseDTD(urlinstr,rootElementName);
```

International Support

At this time only American English XML is supported. The XML language tag <language code> is not used. Also, only UTF-8 character encoding is supported as in the following example

```
<?xml version="1.0" encoding="UTF-8"?>
```

Step 3: Create a URL to accept incoming punchout documents

You must create a URL to accept the incoming XML Punchout Request or the cXML PunchOutSetupRequest transaction sent from Exchange. The Exchange will establish a server to server communication with this URL and send the request using the HTTP protocol. Your URL will need to receive the HTTP Post punchout document, parse it against the appropriate .dtd to ensure that the document is well formed, construct an XML Punchout Response or a cXML PunchOutSetup/Response document and post this back to the Exchange via HTTP post.

Using a secure connection

To provide the necessary security, we recommend that you create an HTTPS page to handle this request. Digital certificates for the creation of an HTTPS site can be obtained from a certification authority such a Verisign® at:

`http://www.verisign.com`

> **Note:** Although you can use an HTTP page for your web catalog portal, HTTP does not provide a secure connection. Since password information is passed between Exchange and your portal, using an unsecure connection may allow the transmission to be intercepted and compromise the security of your site**.**

If your site accepts HTTPS requests, when the Exchange initiates communication with your site, your server will return your digital certificate identification number which Exchange will compare to a list of trusted certificates contained in the file `pomdigcrt.txt` on the Exchange.

> **Note:** This is the default certificate file name. The name can be changed by the Exchange Operator.

If this digital certificate is recognized, the certificate will be used to generate a unique session key (to be used only for the duration of this connection between your site and the Exchange), and the Exchange will transmit this session key back to your site encrypted using your server's public key. All subsequent communication between the Exchange and your site will be encrypted.

Step 4: Create the XML processing code

You will need to create code to perform the XML processing required to communicate with Exchange. This processing includes:

- interpreting the XML Punchout Request or cXML PunchOutSetupRequest document passed to you from Exchange.
- creating the XML Punchout Response or cXML PunchOutSetupResponse document and passing it back to Exchange.

- creating the XML Punchout Shopping Cart or cXML PunchOutOrderMessage document.
- identifying a method for redirecting the buyer and passing the XML Punchout Shopping Cart or cXML PunchOutOrderMessage document back to Exchange.

Step 4A: Process the incoming XML Punchout Request or cXML PunchOutSetupRequest transaction

The XML Punchout Request/cXML PunchOutSetupRequest transaction is created on Exchange when a buyer clicks the link to your catalog. This transaction contains information about the user and the user's company. You should use this information to validate the proper level of user access. The Exchange will establish a server to server communication with your site and return either the XML Punchout Request or cXML PunchOutSetupRequest document using the HTTP protocol.

XML Punchout Request document example

An example XML Punchout Request .xml document is shown below:

```
<?xml version = '1.0' encoding = 'ISO-8859-1'?>
<request>
  <header version="1.0">
<action>SHOPPING</action>
      <login>
         <username></username>
         <password>supplier_password</password>
      </login>
      <language>EN</language>
   </header>
   <body>
      <loginInfo>
         <exchangeInfo>
            <exchangeName>Exchange Name</exchangeName>
         </exchangeInfo>
         <userInfo>
            <userName>Joe Smith</userName>
            <userContactInfo>
               <userPhone>800-123-4568</userPhone>
               <userEmail>
                  jsmith@mail.com
               </userEmail>
            </userContactInfo>
            <userCompany>
               <companyName>
```

```
                  Buying Corporation
               </companyName>
               <companyDUNS/>
               <contactName>JSMITH</contactName>
               <contactPhone>800-123-4567</contactPhone>
            </userCompany>
         </userInfo>
<returnURL>
            http://ap903sun.us.oracle.com:6290/orders
/PunchoutCallback.jsp?app=buying
         </returnURL>
      </loginInfo>
      <searchKeywords>
         paper pens
      </searchKeywords>
   </body>
</request>
```

Table 2–2 XML Punchout Request Header Data Elements

Field	Required	Type	Comment
<username	Required but not used. Can be empty.	Character	Not currently used. Reserved for future development.
<password>	Required	Character	This is the password into your web catalog which your site must validate. Note that every user on the Exchange will provide the same password - there is no need to register individual accounts on your web site. For details on where you register this password, see "Define your link information to Exchange" later in this chapter.
<action>	Constant	SHOPPING	
<language>	Required	Character	This is the ISO language preference of the user for browsing. The XML document itself only supports American English for character sets and number formats.

Table 2–3 XML Punchout Request Body Data Elements

Field	Required	Type	Comment
<exchangeName>	Optional	Character	This is the name of the Exchange from which the punchout was initiated.
<userName>	Required	Character	Buyer's username on the Exchange.
<userPhone>	Optional	Number	Buyer's contact phone number.
<userEmail>	Optional	Character	Buyer's email address.
<companyName>	Optional	Character	Buyer's company name.
<companyDUNS>	Optional	Number	Buyer's DUNS number.
<contactName>	Optional	Character	The buyer company administrator name.
<contactPhone>	Optional	Character	The buyer company administrator phone number.
<returnURL>	Required	Character	This is the URL to which your site will pass the shopping cart XML document when the user exits from your web catalog.
<searchKeywords>	Optional	Character	If this user searched for a particular set of keywords prior to this punchout, these keywords will be included here.

cXML PunchOutSetupRequest document example

An example cXML PunchOutSetupRequest is shown below:

```
<?xml version = '1.0' encoding = 'ISO-8859-1'?>
<!DOCTYPE cXML SYSTEM "http://xml.cxml.org/schemas/cXML/1.1.007/cXML.dtd">
<cXML version="1.1.007" xml:lang="en-US" payloadID="Wed Apr 04 12:52:46
GMT-08:00 2001" timestamp="Wed Apr 04 12:52:46 GMT-08:00 2001">
   <Header>
      <From>
         <Credential domain="DUNS">
            <Identity>654321</Identity>
         </Credential>
      </From>
      <To>
```

```
            <Credential domain="DUNS">
               <Identity>123456</Identity>
            </Credential>
         </To>
         <Sender>
            <Credential domain="OracleExchangeUserId">
               <Identity>buyer@acme.com</Identity>
               <SharedSecret>supplier_password</SharedSecret>
            </Credential>
            <UserAgent>Exchange_name</UserAgent>
         </Sender>
      </Header>
      <Request>
         <PunchOutSetupRequest operation="create">
            <BuyerCookie/>
            <Extrinsic name="User">JSMITH</Extrinsic>
            <BrowserFormPost>
               <URL>
http://iprocurement.company.com/oa_
servlets/oracle.apps.icx.por.punchout.PunchOutServlet?callBack=shopping
               </URL>
            </BrowserFormPost>
            <Contact>
               <Name xml:lang="en-US">jsmith</Name>
               <Email>buyer@acme.com</Email>
            </Contact>
         </PunchOutSetupRequest>
      </Request>
```

```
</cXML>
```

Table 2–4 cXML PunchOutSetupRequest Data Elements

Element	Required?	Type	Comment
<From> <Credential>	Required	Character	Identifies the buying organization. The domain attribute will be either: DUNS If the company has DUNS number on Oracle Exchange, the identity element will have the DUNS number NAME If there is no DUNS number, the identity element will have the name of the company on Oracle Exchange.
<To> <Credential>	Required	Character	Identifies the supplier organization. The rules determining the identity attribute are the same as above.
<Sender> <Credential>	Required	Character	Identifies the Exchange. The domain attribute identifies the Exchange domain value, e.g., OracleExchange.com. The Shared Secret attribute is the password set by the supplier for this exchange.
<Sender> <UserAgent>	Required	Character	Specifies the name of the Exchange, e.g., OracleExchange.com
<BrowserFormPost> <URL>	Required	Character	This is the URL to which your site will pass the shopping cart XML document when the user exits from your web catalog.
<Extrinsic name="User">	Required	Character	This is the username of the user on the Exchange.

Table 2–4 cXML PunchOutSetupRequest Data Elements

Element	Required?	Type	Comment
<Contact>	Optional		This Name and Email elements within the contact element are passed to the Supplier.

Note: The table above is not a complete list of all elements included in the cXML PunchOutSetupRequest document. For DTD details on the cXML transaction above, see http://www.cXML.org/.

Your site may optionally validate that the connection password included is the same as the password you defined when you set up the punchout to your catalog. Your portal may also choose to validate other information, for example, you may wish to check the username and/or company name if you are hosting buyer specific content or price breaks.

Step 4B: Provide an appropriate XML Punchout Response or cXML PunchOutSetupResponse

Provided that the password is correct, your portal should then respond back to Exchange with either an XML Punchout Response or cXML PunchOutSetupResponse document.

XML Punchout Response document example

An example XML Punchout Response document is shown below:

```
<?xml version = '1.0'?>
<response>
  <header version="1.0">
    <action>
       LOGIN_RESPONSE
    </action>
  </header>
  <body>
     <loginInfo>
        <loginURL>
           http://130.35.117.113:6290/orders/
           testscripts/SupplierCallback.jsp?
```

```
          sessionId=123
        </loginURL>
      </loginInfo>
    </body>
  </response>
```

Table 2–5 XML Punchout Response Header Data Elements

Field	Required	Type	Comment
<action>	Constant	LOGIN_RESPONSE	

Table 2–6 XML Punchout Response Body Data Elements

Field	Required	Type	Comment
<loginURL>	Required	Character	This is the URL to which the Exchange user will be directed on your web catalog. As part of the validation process, a session should have been created that will be used to identify the user, and this session should be incorporated into this URL (e.g.,www.mysite.com?sessionID=123)

cXML PunchOutSetupResponse document example

An example cXML PunchOutSetupResponse file is shown below:

```
<?xml version="1.0" ?>
<cXML payloadID="2000-08-15" timestamp="2000-08-15">
  <Response>
    <PunchOutSetupResponse>
      <StartPage>
        <URL>
          http://www.oracle.com/orders/
             SupplierShopping.jsp?sessionID=123
        </URL>
      </StartPage>
    </PunchOutSetupResponse>
    <Status code="200" text="success" />
  </Response>
```

```
</cXML>
```

Table 2–7 cXML PunchOutSetupResponse Data Elements

Element	Required?	Type	Comment
<StartPage><URL>	Required	Character	This is the URL to which the Exchange user will be directed on your web catalog. As part of the validation process, a session should have been created that will be used to identify the user, and this session should be incorporated into this URL (e.g. www.mysite.com?sessionID=123)

For DTD information on the cXML transaction above, see `http://www.cXML.org/.`

Ideally your portal should create an individual session through which the buyer will access your site. This method ensures that the buyers connection is secure (because the session will be closed as soon as the buyer returns to the Exchange), and will also be used to identify the buyer as having navigated to your site from the Exchange. In the above example the session is identified with a session ID.

If an error is discovered (for example, the password is invalid), an XML Punchout Response similar to the example below should be returned to Exchange.

```
<?xml version="1.0"?>
<response>
 <header version="1.0">
  <return returnCode=″A″>
  <return>
 </header>
 <body>
  <loginInfo>
   <loginURL>
   </loginURL>
  </loginInfo>
 </body>
```

```
</response>
```

Table 2–8 XML Punchout Response Header Data Elements

Field	Required	Type	Comment
<return>	Not currently used	Character	Provision for a failure message in the future. Not used currently.
<return returnCode="">	Required Attribute	Character	Valid Values: "S" = Success "E" = Error "W" = Warning "A" = Requires Approval

Table 2–9 XML Punchout Response Body Data Elements

Field	Required	Type	Comment
<loginURL>	Optional	Character	Ignored

The cXML equivalent would be:

```
<?xml version="1.0"?>
<cXML payloadID=”99999” timestamp=”2000-08-15”>
  <Response>
    <PunchoutSetupResponse>
      <StartPage>
        <URL></URL>
      </StartPage>
    </PunchOutSetupResponse>
    <Status code=”401” text=”Invalid Shared Secret” />
  </Response>
</cXML>
```

For detailed information about the elements in the cXML example above, see the cXML 1.1 User’s Guide available at http://www.cXML.org/

If you deny access to your web catalog as in the example above, the buyer will be shown an error message and will be unable to access your web catalog.

Step 4C: Determine the redirect method

Once buyers have completed browsing your site, they must return to the Exchange to complete the purchase process for the items they added to their shopping cart from your web catalog.

Depending on the structure of your web catalog, you may need to determine whether a buyer navigated to your site from the Exchange or from another site. If you site accepts buyers from both the Exchange and other sites, your checkout process will be different than if you only accept buyers from the Exchange.

If you accept buyers from both Exchange and alternate sites, your Exchange buyers should see a "Return to Exchange" button instead of a generic "Proceed to Checkout" button. You could achieve this as follows:

1. When the user initially establishes a connection with your web catalog, they could have a session ID defined for their connection (refer to "Create a URL to accept incoming punchout documents" earlier in this chapter).
2. When the user reaches the point at which they would normally proceed with the checkout process, they will see instead a "Return to Exchange" button.

   The following sample Java code will conditionally display a button if the value of the variable "sessionID" is set to '123'.

```
<%
if(request.getParameter("sessionID").equals("123"))
{ %>
<form name="form_name">
<input type="submit"
name="submit_button"
value="Return to Exchange">
<! Put other appropriate HTML form commands here>
</form>
<% }
%>
```

If your web catalog will only be accessed from Exchange, no such conditional processing is necessary.

Step4D: Return the XML Punchout Shopping Cart or cXML PunchOutOrderMessage information

Once the user clicks the "Return to Exchange" button, your site must send a transaction to the Exchange containing details of all the items the buyer selected from your web catalog.

For Oracle Native XML transactions, the URL to which this XML transaction should be sent is the URL identified by the <returnURL> tag in the original XML Punchout Request transaction. It is important that the document is url-encoded and the parameter name for the HTTP Post is ‘oracleCart.

For cXML transactions, the URL to which this transaction should be sent is identified by the <BrowserFormPost> tag in the original cXML PunchOutSetup Request. It is important that the document is either url-encoded or Base-64 encoded, and that the parameter name for the HTTP Post is either ‘cxml-urlencoded’ or ‘cxml-base64’ respectively.

Once the transaction is returned to the Exchange, the selected items will then be added to the shopping cart, and the buyer can complete the purchase process.

XML Punchout Shopping Cart document example

An example .XML Punchout Shopping Cart transaction file is shown below:

```
<?xml version="1.0"?>

<response>

 <header version="1.0">
   <action>
   </action>
 </header>

 <body>
  <OrderLinesDataElements>
   <orderLine>
    <item quantity="1">
     <itemNumber>
       <supplierItemNumber>
            <itemID>B1324</itemID>
       </supplierItemNumber>
       <manufacturerItemNumber>
         <itemID>X456</itemID>
         <manufacturerName>
          Bob’s Office Supplies Factory
         </manufacturerName>
       </manufacturerItemNumber>
      </itemNumber>
    <itemDescription>
     Whiteboard 6’ x 4’
    </itemDescription>
```

```
  <unitOfMeasure>
     <supplierUnitOfMeasure>
         <supplierUOMType >
          Each
         </supplierUOMType>
     </supplierUnitOfMeasure>
   </unitOfMeasure>
   <hazardClass>UVW456</hazardClass>
  </item>

  <category>
   <categoryCode>MAS-736</categoryCode>
  </category>

  <price>
   <currency>USD</currency>
   <unitPrice>50.99</unitPrice>
  </price>

  <supplier>
   <supplierDUNS>12345</supplierDUNS>
         <supplierName>supplier_name</supplierName>
  </supplier>

</orderLine>

<orderLine>
  <item quantity="10">
   <itemNumber>
     <supplierItemNumber>
          <itemID>C1324</itemID>
     </supplierItemNumber>
     <manufacturerItemNumber>
       <itemID>X456</itemID>
       <manufacturerName>
        Acme Office Supplies
       </manufacturerName>
     </manufacturerItemNumber>
    </itemNumber>
  <itemDescription>
   Red Whiteboard Markers, Chisel Tip
  </itemDescription>
  <unitOfMeasure>
     <supplierUnitOfMeasure>
         <supplierUOMType >
```

```
            Each
           </supplierUOMType>
         </supplierUnitOfMeasure>
       </unitOfMeasure>
       <hazardClass>ABC123</hazardClass>
      </item>

      <category>
       <categoryCode>MAB-7836</categoryCode>
      </category>

      <price>
       <currency>USD</currency>
       <unitPrice>5.99</unitPrice>
      </price>

       <supplier>
       <supplierDUNS>12345</supplierDUNS>
             <supplierName>supplier_name</supplierName>
      </supplier>

    </orderLine>
   </OrderLinesDataElements>
  </body>
</response>
```

Table 2–10 XML Punchout Shopping Cart Header Data Elements

Field	Required	Type	Comment
<action>	Not currently used	Character	Reserved for future development

Table 2–11 XML Punchout Shopping Cart Body Data Elements

Field	Required	Type	Comments
<item quantity>	Optional	Number	Default ="1" Decimals are allowed
<supplierItemNumber	Conditionally optional	Character	Supplier’s internal item number

Table 2–11 XML Punchout Shopping Cart Body Data Elements

Field	Required	Type	Comments
<itemID>	Optional	Character	If this item is a configured item, you may wish to use this field to provide a unique reference to the configuration that the buyer has selected. This will allow you to identify the exact configuration that was originally selected. The maximum length of this field is 25 characters.
<manufacturerItemNumber>	Optional	Character	Manufacturer's part number
<manufacturerName	Optional	Character	Manufacturer's name
<itemDescription>	Conditionally Optional	Character	Description of item. If this is a configuration item this field can be optionally used to describe the components of the configuration. The maximum length of this description is 240 characters.
<supplierUOMType>	Conditionally Operational	Character	This is the Unit of Measure code that is used in your web catalog and ERP applications.
<hazardClass>	Do not use	Character	This tag is required but should not contain any data.
<categoryCode>	Required	Number	Type of product being purchased.

Table 2–11 XML Punchout Shopping Cart Body Data Elements

Field	Required	Type	Comments
<currency>	Conditionally Optional	Number	This is the currency code which you defined within the Data Mapping on the Exchange.
<unitPrice>	Conditionally Optional	Number	Price per unit of this item
<supplierDUNS>	Conditionally Required	Number	Supplier DUNS number. Either <supplierDUNS> or <supplierTradingPartner Code> is required.
<supplierTradingPartnerCode>	Conditionally Required	Character	Code for this trading partner. Either <supplierDUNS> or <supplierTradingPartner Code> is required.

Note: In the chart above, all elements noted as conditionally optional are only optional if there are no items in the shopping cart.

Note: For configured items (such as a laptop PC), we recommend that you return a single line in the shopping card which represents the entire configuration. You must provide enough information to ensure that the correct configuration will be shipped to the buyer and allow the buyer to identify the item. This can be achieved through a combination of information contained in the <itemID> and <itemDescription> fields. It is recommended that the <itemID> field contain a code which uniquely identifies the specific configuration selected by the buyer and the <itemDescription> field contain a user understandable description of the configured item.

cXML PunchOutOrderMessage document example

An example cXML PunchOutOrderMessage file is shown below:

```
<?xml version = "1.0" encoding = "UTF-8"?>
<cXML version = "1.1.007" xml:lang = "en-US"
     payloadID = "99999" timestamp = "2000-08-15">
  <Header>
    <From>
      <Credential domain = "DUNS">
        <Identity>888</Identity>
      </Credential>
    </From>
    <To>
      <Credential domain = "DUNS">
        <Identity>888</Identity>
      </Credential>
    </To>
    <Sender>
      <Credential domain = "DUNS">
        <Identity>888</Identity>
      </Credential>
      <UserAgent>Workchairs cXML</UserAgent>
    </Sender>
  </Header>
  <Message>
    <PunchOutOrderMessage>
      <BuyerCookie>ICX</BuyerCookie>
      <PunchOutOrderMessageHeader
           operationAllowed = "edit">
        <Total>
          <Money currency = "USD">112.4</Money>
        </Total>
      </PunchOutOrderMessageHeader>
      <ItemIn quantity = "3">
        <ItemID>
          <SupplierPartID>5555</SupplierPartID>
        </ItemID>
        <ItemDetail>
          <UnitPrice>
            <Money currency = "USD">763.20</Money>
          </UnitPrice>
          <Description xml:lang = "en">
             Leather reclining chair
          </Description>
          <UnitOfMeasure>EA</UnitOfMeasure>
```

```
          <Classification domain = "UNSPSC">
            513333
          </Classification>
        </ItemDetail>
      </ItemIn>
    </PunchOutOrderMessage>
  </Message>
</cXML>
```

Table 2–12 cXML PunchOutOrderMessage Data Elements

Element	Required?	Type	Comment
<SupplierPartID>	Required	Character 25	This is a unique item reference number which can be used to identify the item or item configuration.
<SupplierPartAuxillaryID>	Not used		This element is not used by the Exchange. It is therefore necessary to pass a unique reference within the <SupplierPartID> element.

Note: For DTD details on the cXML transaction above, see http://www.cXML.org/.

Note: For configured items (such as a laptop PC), it is recommended that you return a single line in the shopping card which represents the entire configuration. You must provide enough information to ensure that the correct configuration will be shipped to the buyer and allow the buyer to identify the item. This can be achieved using the information contained in the <SupplierPartID> element. It is recommended that the <SupplierPartID> element contain a code which uniquely identifies the specific configuration selected by the buyer and that the <Description> element contain a user understandable description of the configured item.

Steps to Configure Your Punchout Definition

The final set of steps is necessary to make Exchange aware of your web catalog's punchout capability and to control accessibility. These setup steps are documented below:

1. Configure Punchout Definition.
2. Test Punchout Definition.
3. Assign Search Keywords.
4. Control Availability.

Prerequisites

- You have a signon with Catalog Authoring authority, and the Catalog Authoring job function includes the Configure Catalog Linkout job task. You also have the "Create Add-to-Cart Order job function

The Configuration steps listed about are performed from the **Configure Catalog Punchout** page.

1. In the Markeplace Selling application, click the Catalogs tab.
2. From the **Catalogs** page, click the "Configure Catalog Punchout" or the "Setting up a catalog punchout" link. The **Configure Catalog Punchout** page appears. The four steps to configuring your punchout are accessible from this page.

Step 1: Configure Punchout Definition

For users to punch out directly to your catalog, you must provide the necessary punch out information.

1. Click the "Click here to configure your punchout" link on the **Configure Catalog Punchout** page. The **Configure Punchout Definition** page appears.
2. Enter values in the Punchout URL and Punchout Password fields. The Punchout URL is the address of your web catalog's portal page. This is the URL you created in a prior step and is the address Exchange will contact when processing a punchout request. The punchout password is the value Exchange will pass to your web catalog for access validation.
3. Select the communication protocol you will use to pass information to Exchange. Exchange supports both XML and cXML.

4. When you have entered the URL, password and protocol values, scroll to the bottom of the page.
5. You may optionally enter the following additional information using the User Interface Identification section of the **Configure Punchout Definition** page. This information appears when your company is displayed as a punchout enabled supplier.

- You can provide an alternate company name in the Display Name field. This name will be used in place of your company's registered name.
- You can supply the location of a company logo image in the Logo URL field. This image must be in .gif format and no larger than 125 pixels wide and 25 pixels high. This URL address must be outside your firewall so Exchange can access the image.
- You can add a brief description of your catalog and the items you sell in the Company Description field.

When you have enter the required information, click the Apply Changes button. On the Confirmation page, click "Return to Configure Catalog Punchout."

Step 2: Test Punchout Definition

After you have configured your punchout definition you should test it. You can test your definition while still within the configuration process. This allows you to test your definition before opening the punchout to users. To test your punchout, you must have the "Create Add-to-Cart Order" authorization.

1. From the **Configure Catalog Punchout** page, click the "Click here to test your punchout" link. The **Test Catalog Punchout** page appears.
2. Click the Begin Punchout Test Now button.

If your definition has been correctly defined, you will be able to punchout to your site and select items from your catalog. Once you have selected several items, return to Oracle Exchange by clicking the Return to Oracle Exchange button (or whatever name you used when creating your redirect earlier). Verify your shopping cart contents have been successfully added to an Oracle Exchange shopping cart. Once you have done this, you should empty the cart so you don't create a purchase order accidentally. If you receive an error message at any point in the punchout process described above, make a note of the message to assist in resolving the problem.

Step 3: Define Your Search Keywords

Defining appropriate search keywords for items in your web catalog is important to ensure that buyers browsing the Exchange are presented with the link to your catalog. If a buyer's

search criteria includes at least one of the keywords you define, your web catalog will be displayed in the list of suppliers presented to the buyer.

Keywords define the items that you have available on your web catalog, and you can enter up to 2000 characters of information for each language for which you require keywords. For example if you sell paper products to both English and French speaking customers, you may want to include the following keywords in American English:

ream paper photocopy Xerox A5 A4 A3 A2 A1 Letter

and the following keywords in French:

rame papier enveloppe photocopie A5 A4 A3 A2 A1

If a buyer searches on the Exchange in a foreign language for which you have defined no keywords, no link to your catalog will be displayed. In the example above, if the buyer searches for 'papel' in Spanish, a link to your catalog is not returned because you defined no keywords for Spanish.

1. To create search keywords, you must sign on to Exchange as a user with Catalog Authoring authority.
2. On the **Configure Catalog Punchout** page, click the "Click here to define keywords for your punchout" link. The **Trading Partner Keywords** page appears.
3. Select the language for which you wish to define keywords from the "Language" drop down menu.
4. Enter all appropriate keywords in the text area.
5. When you have completed defining your keywords, click the "Apply Changes" button.
6. Repeat steps 3 -5 for any additional languages your web catalog supports.

Step 4: Control Punchout Availability

Your punchout definition is not initially available to users. Only the punchout author can access your punchout for testing purposes. You must publish your punchout definition before any buyers can use it.

1. From the **Configure Catalog Punchout** page, click the "Click here to set the availability of your punchout link." The **Control Punchout Availability** page appears.
2. Click the radio button next to "Yes, publish my punchout to buying companies on the Exchange."
3. Click the Apply Changes button.

4. Your punchout is now available

Revoking a punchout used by an Exchange member

You can revoke your punchout definition at any time. This prevents the punchout from being seen and used by buyers. The definition itself remains on Exchange.

If a buyer has created an order which includes an item from the punchout, it is possible for the buyer to reorder the order at a later stage. If you revoke or unpublish a punchout which was been previously used to create an order, a buyer will be prevented from reordering any of the items on this order.

Buyer Steps to Control Punchout Access

Purchasing managers need to be able to specify which suppliers' punchouts can be used by buyers within their company. They can do this by enabling punchouts only from approved suppliers.

When a new company registers with Exchange, no punchouts are available until they are enabled, and any new supplier punchouts that are defined are not visible to buyers until specifically enabled by the Purchasing Manager. Purchasing Manages can remove or add punchout definitions at any time.

You must have the Control Punchout Access Exchange task to identify the punchouts available to buyers within your company. Note also that Company Administrators can control access to punchouts by granting or revoking the Punchout to Supplier Site Exchange task to individual users.

1. In the Marketplace Buyer application, click the Purchases tab.
2. Under Catalogs, click the "Control Punchout Access" link.The **Control Punchout Access** page appears. This page lists all punchouts defined for Exchange.
3. Click the Select checkbox next to the punchouts you wish to enable for buyers in your company.
4. Once you have selected all the punchouts your company will use, click the Review and Submit button.
5. A confirmation page appears. Your users can now access the punchouts.

Effect of a Supplier Disabling a Punchout

If a supplier disables a punchout it will immediately be unavailable for use by any buyer. It will continue to appear on the Select Supplier Punchouts page although it will be identified as having been disabled by the supplier.

Once the Purchasing Manager deselects the disabled punchout links and commits the change, the links are not available for reselection until the supplier re-enables the punchout.

Reordering from a Punchout Disabled by the Purchasing Manager

A situation can arise when a Purchasing Manager enables access to a supplier's punchout, a buyer then creates an order from this punchout, and the Purchasing Manager subsequently disallows this punchout. If, after the Purchasing Manager has disallowed this punchout, the buyer then clicks the Reorder button to reorder these items, the buyer will get an error message and be prompted to view those items which could not be copied onto the new order.

If the buyer then clicks the link, a list with all the items which could not be copied onto a new purchase order displays.

For those items created from a punchout which has subsequently been disallowed by the Purchasing Manager, the following message displays:

> The item was selected from an external site which has since been disabled by your Purchasing Manager. Please select another item.

Oracle Exchange Punchout Setup Checklist

The following is a checklist of the tasks you must complete to integrate your web catalog with Exchange.

1. Register your company with Exchange (if necessary).
2. Map your currency codes (if necessary).
3. Map your unit of measurement codes (if necessary).
4. Obtain and install an XML Parser.
5. Create necessary .dtds (see Appendix B for details if using Oracle Native XML, see `http://www.cXML.org/`. if using cXML).
6. Determine how you are going to validate user access using the information passed to you.
7. Determine any additional processing you may want to perform using the information passed to you.
8. Determine what method to use when returning buyers to Exchange.
9. Create the URL to process transaction validation.
10. Create code to process:

- XML Punchout Request/PunchOutSetupRequest
- XML Punchout Response/PunchOutSetupResponse
- XML Punchout Shopping Cart/PunchOutOrderMessage

11. Define the punchout information for your web store on Exchange
12. Test your punchout.
13. Define your search keywords for every language your site supports.
14. Enable your punchout definition.

3

Defining an Oracle iProcurement Punchout

This chapter explains the punchout process used by Oracle iProcurement for each of the punchout models described in the Overview chapter. This chapter includes both an architectural overview of the model and the steps to implement each model.

Topics in this chapter include

- Model 2: Punch out from Oracle iProcurement to Oracle Exchange: XML
- Model 3: Punch out from Oracle *i*Procurement to Supplier via Oracle Exchange: XML
- Model 4: Punch out from Oracle *i*Procurement to Supplier via Oracle Exchange: cXML
- Model 5: Punch out from Oracle *i*Procurement directly to Supplier: XML
- Installation/Impelmentation Steps
- Oracle Exchange Setup Steps
- Oracle iProcurement Setup Steps
- Oracle eCommerce Gateway Setup Steps

Model 2: Punch out from Oracle iProcurement to Oracle Exchange: XML

An Oracle Exchange is setup as a punchout (or external) supplier hub in iProcurement and the Oracle Exchange link is displayed on the iProcurement homepage. This enables iProcurement users to punch out to Oracle Exchange, fill a shopping cart with requisition items, and return the cart to iProcurement for approval routing and purchase order creation.

Setup iProcurement Buying Organization as Buyer on Oracle Exchange

Before an iProcurement adminstrator can setup a punchout to Oracle Exchange, they must first register their organization on Oracle Exchange. This registration process will provide company and user accounts and a password, which are required to setup the punchout process. Refer to the Oracle Exchange you want to participate on for details on the registration requirements.

Setup Supplier on Oracle Exchange

The iProcurement buying organization must also have their suppliers register and load catalogs on the Oracle Exchange they will punch out to. The supplier catalog can include buyer specific pricing for the iProcurement customer and is loaded on Oracle Exchange using either a tab-delimited text file (spreadsheet) or XML file. Refer to the Exchange you want to register on for details on loading a catalog.

Setup Oracle Exchange as a Punchout (external) Supplier in iProcurement

The Oracle Exchange you want to punch out to must first be setup in iProcurement. For details on the setup required, refer the implementation and installation section later in this document.

Punchout Process When a user clicks the link to Oracle Exchange, the 'PunchOutServlet' servlet is executed. First, this calls Oracle SSL APIs to establish secure connection to Oracle Exchange. These APIs request a Certification Authority Certificate from the Oracle Exchange site and compare that to Certificates stored in iProcurement.

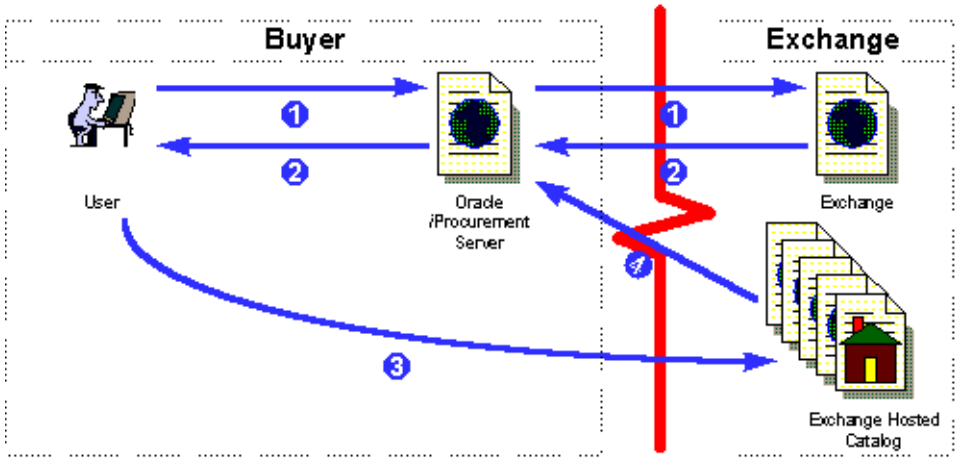
After a secure connection between iProcurement and Oracle Exchange is established, 'PunchOutServlet' then generates the 'OEXLoginRequest' XML document, that contains user details and login credentials for Oracle Exchange as well as a return URL. Once generated, this is passed to Oracle Exchange ('LinkinLogin.jsp') through the secure connection.

Oracle Exchange uses the LinkinAuthBean java class to authenticate the iProcurement user. Once authenticated Oracle Exchange uses the LinkinMainBean java class to generate the 'LoginResponse' XML document that is returned to iProcurement through the secure connection.

After the login response is received by the iProcurement instance, the user's browser is re-directed to the login URL provided by the Exchange in the 'LoginResponse' XML document. Now, the user can search and add items to a shopping cart on Oracle Exchange. The Exchange functionality available to the user is based on a profile previously setup in Oracle Exchange.

After the user finishes adding items to their shopping cart, the 'ReqTransaction' java class in Oracle Exchange calls the 'ExportOrder' java class, which creates the 'OrderLinesDataElements' XML document, that contains the shopping cart items added from Oracle Exchange. This is sent back to iProcurement, based on the return URL.

iProcurement executes the 'AppsManager' java class to add the items to the requisition. The EDI setups in Oracle Applications are referenced to do the required data conversion (mapping) between the external code values coming from the supplier and the internal code values setup in Oracle iProcurement. If the lock_item_flag was not set during configuration of the punchout for this Exchange, then the user will be able to make any changes to the item returned from Oracle Exchange. The requisition then goes through the normal workflow and approval processes configured in iProcurement, before being turned into a purchase order.



1. The *i*Procurement user starts on their homepage, which displays a link to an Oracle Exchange. When the user clicks the 'Oracle Exchange' link, the browser connects to the *i*Procurement server and the 'PunchOutServlet' is executed. This servlet first calls Oracle SSL APIs to establish a secure connection with Oracle Exchange. The APIs request the Certification Authority (CA) digital certificate (public key) from Oracle Exchange. This is compared to certificate authorities stored in the ca-bundle.crt file in iProcurement.

   After the secure connection is established the 'PunchOutServlet' generates the 'OEXLoginRequest' XML document, which includes a base set of user details and the

return URL for the *i*Procurement instance. This XML document is passed to the Oracle Exchange site.

'OEXLoginRequest' sample:

```
<?xml version = '1.0'?>
<request>
   <header version="1.0">
      <login>
         <username>buyer1</username>
         <password>WELCOME</password>
      </login>
      <action>shopping</action>
      <language>US</language>
      <userArea>
         <operatingUnit>204</operatingUnit>
         <shipTo>V1- New York City</shipTo>
         <deliverTo>V1- New York City</deliverTo>
      </userArea>
   </header>
   <body>
      <loginRequest>
         <userInfo>
            <userName>Green, Mr. Terry</userName>
            <appUserName>TGREEN</appUserName>
            <userContactInfo>
               <userEmail>tgreen@vision.com</userEmail>
            </userContactInfo>
         </userInfo>
         <companyInfo>
            <companyName>IP</companyName>
         </companyInfo>
         <reqToken>
template=tpn*@_action=addToOrder*@_function=addToOrder*
 @_reqSessionId=0*@_reqHeaderId=0*@_emergencyReq=N*@_itemSourceId
=1001
         </reqToken>
         <returnURL>
http://iprocurement.company.com/oa_servlets/oracle.apps.icx.por.
punchout.PunchOutServlet?callBack=shopping
         </returnURL>
      </loginRequest>
   </body>
</request>
```

Note: The DTDs are described in detail in the Appendix 'A' of this document.

Some additional optional user data can also be sent to Oracle Exchange if the punchout was setup in iProcurement to send it. The sample XML document for a login request with additional optional user data is attached in Appendix 'B' of this document. For details on how to include the additional information see the Installation and Implementation section.

2. Oracle Exchange receives the 'OEXLoginRequest' with the 'LinkinLogin.jsp' and executes the 'LinkinAuthBean' java class to authenticate the *i*Procurement user. The user id and password used to authenticate the *i*Procurement user are the same user id and password that is created by the *i*Procurement administrator when they register their company on Oracle Exchange. Once authenticated, 'LinkinMainBean' java class generates the 'LoginResponse' XML and sends it back to the Oracle iProcurement.

   'LoginResponse' sample:

```
<?xml version = '1.0'?>
<response>
  <header version="1.0">
    <action>LOGIN_RESPONSE</action>
  </header>
  <body>
    <loginInfo>
      <loginURL>
http://www.oracleexchange.com/orders/LinkinCallback.jsp
?sessionId=3fe0a73f6fa1c003.164.982359182161&action=shopping
     </loginURL>
    </loginInfo>
  </body>
</response>
```

3. Now that a secure and trusted connection is established and the user has been assigned a session, their browser is re-directed to the loginURL on Oracle Exchange and they are allowed to search and add items to their shopping cart on Oracle Exchange. The Exchange functionality available to the user is based on the job functions assigned to the Oracle Exchange user that was used during the initial authentication of the punch out. The *i*Procurement administrator on Oracle Exchange controls the job functions.

4. When the user finishes adding items to their cart on Oracle Exchange, the items in the shopping cart (sspcart.jsp) are posted in a form to iProcurement (using the parameter oracleCart). This is controlled by the 'ReqTransaction' java class, using the callback URL received from iProcurement during the initial punch out (see step 1 of this section). The data is sent back in the 'OrderLinesDataElements' XML document (generated by ExportOrder.java) which is 'URL-encoded'.

   'OrderLinesDataElements' sample:

```
<?xml version = '1.0' encoding = 'ISO-8859-1'?>
<response>
   <header>
      <action>SHOPPING_RESPONSE</action>
   </header>
   <body>
      <OrderLinesDataElements>
         <catalogTradingPartner>OracleExchange</catalogTradingPartner>
         <orderLine>
            <contract>
               <catalogType>CONTRACTED</catalogType>
            </contract>
            <item lineType="GOODS" quantity="1.0">
               <itemNumber>
                  <supplierItemNumber>
                     <itemID>kb_Item2</itemID>
                  </supplierItemNumber>
               </itemNumber>
               <itemDescription>
                   Item #2 : kbenjami- list, TBD b.s.
               </itemDescription>
               <unitOfMeasure>
                  <buyerUnitOfMeasure/>
                  <supplierUnitOfMeasure>
                     <supplierUOMType>Bushel</supplierUOMType>
                     <supplierUOMQuantity></supplierUOMQuantity>
                  </supplierUnitOfMeasure>
               </unitOfMeasure>
            </item>
            <category>
               <categoryCode categoryCodeIdentifier="SPSC">
                   Ball Point Pens
               </categoryCode>
            </category>
            <price>
               <currency>USD</currency>
```

```
            <unitPrice>100.0</unitPrice>
         </price>
         <supplier>
            <supplierDUNS>2820</supplierDUNS>
            <supplierName>Kareem's Corporation</supplierName>
         </supplier>
      </orderLine>
   </OrderLinesDataElements>
</body>
</response>
```

> **Note:** See Appendix 'A' for the DTD of the shopping cart sent back from Oracle Exchange to iProcurement.
>
> See Appendix 'B' for a sample XML document (shopping cart) sent back from Oracle Exchange to iProcurement.

5. *i*Procurement executes the 'AppsManager' java class to add the items to the requisition. The EDI setups in Oracle Applications are referenced to do the required data conversion (mapping) between the external code values coming from the supplier and the internal code values setup in Oracle iProcurement. And if the lock_item_flag (stored in the ICX_POR_ITEM_SOURCES table) was set during configuration of the punchout for this Exchange, then the user will not be able to make any changes to the items returned from Oracle Exchange. The requisition then goes through the normal workflow and approval processes configured in iProcurement.

> **Note:** See the Installation and Setup section of this chapter for details on setting the lock_item_flag and using the EDI setups to create data maps.

Model 3: Punch out from Oracle *i*Procurement to Supplier via Oracle Exchange: XML

Oracle *i*Procurement users can also punch out to an XML supplier through Oracle Exchange. Using Oracle Exchange for the punchout simplifies the initial setup process and as the authentication and maintenance of the punchout. To punch out to a supplier through the Exchange, the *i*Procurement customer must setup Oracle Exchange as a supplier hub in *i*Procurement and their suppliers must setup a punchout from Oracle Exchange to their web store.

Setup Suppliers on Oracle Exchange

The suppliers can setup their punchout definitions one time on Oracle Exchange. Then any *i*Procurement administrator can download these definitions, which will automatically setup the punchout from *i*Procurement to that supplier.

> **Note:** See the chapter 'Defining an Oracle Exchange Punchout' for details on setting up a supplier for punchout from an Oracle Exchange.

Setup Oracle Exchange as a Supplier Hub in iProcurement

The Oracle Exchange you want to use as a supplier hub must first be setup in *i*Procurement. For details on the setup required, refer the implementation and installation section later in this document

Setup Punchout Suppliers from Oracle Exchange in iProcurement

The iProcurement administrator logs on to the *i*Procurement eContent Manager and initiates the process of setting up punchout suppliers they want to enable. *i*Procurement executes 'PunchOutServlet' which first establishes a secure connection with Oracle Exchange. Once established, 'PunchOutServlet then generates the 'OEXLoginRequest' XML document and sends it to Oracle Exchange. After the *i*Procurement user is authenticated on Oracle Exchange their browser is re-directed to Oracle Exchange, using the login URL provided in the 'LoginResponse' XML. The *i*Procurement administrator selects the punchout suppliers they want to enable in *i*Procurement. Oracle Exchange sends the suppliers' setups to *i*Procurement in the 'SupplierSyncUpResponse' XML document. *i*Procurement then executes 'PunchOutServlet' which creates all the supplier punchout related setups in *i*Procurement.

Part of this setup process includes the download of keywords the supplier has associated with their company during their setup on Oracle Exchange. The first time a supplier is setup in *i*Procurement these keywords will be downloaded into *i*Procurement. The *i*Procurement customer can keep these keywords or change them to meet their own user's needs. This information is not downloaded from Oracle Exchange again.

Additional setup information maintained by the supplier on Oracle Exchange (such as their punchout URL and site level user id and password) will remain on the Exchange. This is referenced each time a punchout occurs, so the *i*Procurement administrator does not have to maintain any supplier specific login details themselves.

Punchout Process

The supplier is automatically setup in Oracle *i*Procurement as a punchout supplier in the above step and the Supplier's link is displayed on the *i*Procurement homepage.

When a user clicks the link to a supplier, the 'PunchOutServlet' servlet is executed. First, this calls Oracle SSL APIs to establish secure connection to Oracle Exchange. These APIs request a Certification Authority Certificate from the Oracle Exchange site and compare that to Certificates stored in the *i*Procurement.

After a secure connection between the *i*Procurement server and Oracle Exchange is established, 'PunchOutServlet' then generates the 'OEXLoginRequest' XML document, which contains user details and login credentials for Oracle Exchange as well as a return URL. Once generated, this is passed to Oracle Exchange ('LinkinLogin.jsp') through the secure connection.

Oracle Exchange uses the 'LinkInAuthBean' java class to authenticate the iProcurement user. Once authenticated Oracle Exchange uses the 'PunchOutLoginBean' java class to initiate a connection with the supplier site. If required (based on the setup of the supplier), Oracle Exchange will request a Certification Authority Certificate from the supplier site and compare that to Certificates stored in Oracle Exchange.

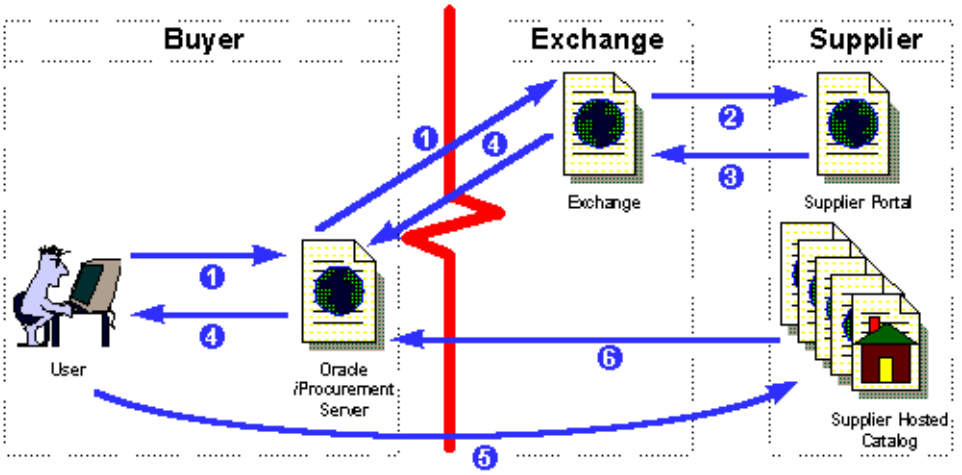
Once a secure connection is established, the 'PunchOutLoginBean' java class will generate and deliver the 'LinkRequest' XML document to the supplier site, which contains the *i*Procurement return URL. The supplier site generates a response in the form of a 'LinkResponse' XML document which contains the login URL and session information for the supplier's site.

Oracle Exchange uses the LinkinMainBean java class to generate the 'LoginResponse' XML document that is returned to *i*Procurement.

Upon receipt of this 'LoginResponse' XML document by *i*Procurement, the iProcurement user's browser is re-directed to the login URL provided by the supplier. The Oracle Exchange involvement in this process is transparent to the *i*Procurement user. The user can shop the supplier's site and add items to a shopping cart on the supplier site. When the user finishes adding items to their shopping cart, the supplier executes a program, which takes the items in the shopping cart and creates an 'OrderLinesDataElements' XML document with the shopping cart item details. This document is returned to *i*Procurement, based on the return URL.

*i*Procurement executes the 'AppsManager' java class to add the items to the requisition. The EDI setups in Oracle Applications are referenced to do the required data conversion (mapping) between the external code values coming from the supplier and the internal code values setup in Oracle *i*Procurement. If the lock_item_flag was not set during configuration of the punchout for this Exchange, then the user will be able to make any changes to the

item returned from Oracle Exchange. The requisition then goes through the normal workflow and approval processes configured in *i*Procurement.



1. The *i*Procurement user starts on their homepage, which displays a link to a supplier site. When the user clicks the link, the browser connects to the *i*Procurement server and the 'PunchOutServlet' is executed. This servlet first calls Oracle SSL APIs to establish a secure connection with Oracle Exchange. The APIs request the Certification Authority (CA) digital certificate (public key) from Oracle Exchange. This is compared to certificate authorities stored in the ca-bundle.crt file in iProcurement.

   After the secure connection is established the 'PunchOutServlet' generates the 'OEXLoginRequest' XML document, which includes a base set of user details and the return URL for the *i*Procurement instance. This XML document is passed to the Oracle Exchange site.

   'OEXLoginRequest' sample:

```
<?xml version = '1.0'?>
<request>
   <header version="1.0">
      <login>
         <username>buyer1</username>
         <password>WELCOME</password>
      </login>
      <action>shopping</action>
      <language>US</language>
      <userArea>
```

```
         <operatingUnit>204</operatingUnit>
         <shipTo>V1- New York City</shipTo>
         <deliverTo>V1- New York City</deliverTo>
      </userArea>
   </header>
   <body>
      <loginRequest>
         <userInfo>
            <userName>Green, Mr. Terry</userName>
            <appUserName>TGREEN</appUserName>
            <userContactInfo>
               <userEmail>tgreen@vision.com</userEmail>
            </userContactInfo>
         </userInfo>
         <companyInfo>
            <companyName>IP</companyName>
         </companyInfo>
<partySiteId>
        2963
</partySiteId>
         <reqToken>
template=tpn*@_action=addToOrder*@_function=addToOrder*
 @_reqSessionId=0*@_reqHeaderId=0*@_emergencyReq=N*@_itemSourceId
=1001
         </reqToken>
         <returnURL>
http://iprocurement.company.com/oa_
servlets/oracle.apps.icx.por.punchout.PunchOutServlet?callBack=shopping
         </returnURL>
      </loginRequest>
   </body>
</request>
```

Note: The DTDs are described in detail in the Appendix ‘A’ of this document.

Some additional optional user data can also be sent to Oracle Exchange if the punchout was setup in iProcurement to send it. The sample XML document for a login request with additional optional user data is attached in Appendix ‘B’ of this document. For details on how to include the additional information see the Installation and Implementation section.

Oracle Exchange receives the ‘OEXLoginRequest’ with the ‘LinkinLogin.jsp’ and executes the ‘LinkinAuthBean’ java class to authenticate the *i*Procurement user.

2. After the *i*Procurement user is authenticated on Oracle Exchange, the Exchange uses ‘LinkinLogin.jsp to establish a secure connection to the supplier site. It calls Oracle SSL APIs (if required based on the supplier's setup) which requests the Certification Authority (CA) digital certificate (public key) from the supplier site. The Oracle Exchange operator compares this to certificate authorities setup. If validated a secure connection is opened between Oracle Exchange and the supplier site.

   The ‘PunchOutLoginBean’ java class then generates the ‘LinkRequest’ XML document. It includes the site level user name and password that the supplier requires accessing their web store.

   Note: The site level user name is stored and maintained on Oracle Exchange by the supplier. There is no need for the iProcurement sysadmin to store this information.

   The ‘LinkRequest’ also includes user and company details from the *i*Procurement user that were included in the initial ‘OEXLoginRequest’ from *i*Procurement to Oracle Exchange.

   ‘LinkRequest’ XML sample:

```
<?xml version="1.0"?>
<response>
  <header version="1.0">
   <login>
    <username/>
    <password>
     supplier_password
    </password>
   </login>
   <action/>
 </header>
 <body>
  <loginInfo>
   <exchangeInfo>
    <exchangeName>
     Oracle Exchange
    </exchangeName>
   </exchangeInfo>
```

```
   <userInfo>
    <userName>Joe Smith</userName>
    <userContactInfo>
     <userPhone>6505062000</userPhone>
     <userEmail>
      jsmith@hotmail.com
     </userEmail>
    </userContactInfo>
    <userCompany>
     <companyName>
      ABC Incorporated
     </companyName>
     <companyDUNS>12345</companyDUNS>
    </userCompany>
   </userInfo>
   <returnURL>
http://iProcurment.company.com/oa_
servlets/oracle.apps.icx.por.punchout.PunchOutServlet?callBack=shopping
   </returnURL>
  </loginInfo>
 </body>
</response>
```

3. The supplier site generates a response in the form of a ‘LinkResponse’ XML document that contains the login URL with session information for the supplier’s site.

‘LinkResponse’ XML sample:

```
<?xml version="1.0"?>
<response>
  <header version="1.0">
   <action/>
</header>
 <body>
  <loginInfo>
   <loginURL>
    http://www.suppliersite.com/orders/SupplierShopping.jsp?sessionID=123
   </loginURL>
  </loginInfo>
 </body>
</response>
```

4. The ‘LinkResponse’ is returned to the ‘PunchoutCallback.jsp.’ Oracle Exchange then uses the ‘LinkinMainBean’ java class to generate the ‘LoginResponse’ XML and sends

it back to iProcurement using the return URL. The XML document contains the session aware URL which the *i*Procurement user will use to connect to the supplier site.

'LoginResponse' XML sample:

```
<?xml version = '1.0'?>
<response>
  <header version="1.0">
    <action>LOGIN_RESPONSE</action>
  </header>
  <body>
    <loginInfo>
      <loginURL>
http://www.suppliersite.com/orders/SupplierShopping.jsp?sessionID=123
     </loginURL>
    </loginInfo>
  </body>
</response>
```

5. Using the session aware login URL provided in the 'LoginResponse' XML, *i*Procurement executes the 'PunchOutServlet,' that re-directs the *i*Procurement user's browser to the supplier site using the supplier's login URL. The user is allowed to search and add items to their shopping cart on the supplier site.

6. When the user finishes adding items to their cart on the supplier site, the items in the shopping cart must be sent back to *i*Procurement by posting a form to the return URL received from *i*Procurement during the initial punchout. The parameter for the post should be 'oracleCart'. The data is sent back in the 'OrderLinesDataElements' XML document which is 'url-encoded'.

   'OrderLinesDataElements' XML sample:

```
<?xml version = '1.0' encoding = 'ISO-8859-1'?>
<response>
   <header>
      <action>SHOPPING_RESPONSE</action>
   </header>
   <body>
      <OrderLinesDataElements>
         <catalogTradingPartner><suppier name></catalogTradingPartner>
         <orderLine>
            <contract>
               <catalogType>CONTRACTED</catalogType>
            </contract>
            <item lineType="GOODS" quantity="1.0">
```

```
                  <itemNumber>
                     <supplierItemNumber>
                        <itemID>kb_Item2</itemID>
                     </supplierItemNumber>
                  </itemNumber>
                  <itemDescription>
                      Item #2 : kbenjami- list, TBD b.s.
                  </itemDescription>
                  <unitOfMeasure>
                     <buyerUnitOfMeasure/>
                     <supplierUnitOfMeasure>
                        <supplierUOMType>Bushel</supplierUOMType>
                       <supplierUOMQuantity></supplierUOMQuantity>
                     </supplierUnitOfMeasure>
                  </unitOfMeasure>
               </item>
               <category>
                  <categoryCode categoryCodeIdentifier="SPSC">
                      Ball Point Pens
                  </categoryCode>
               </category>
               <price>
                  <currency>USD</currency>
                  <unitPrice>100.0</unitPrice>
               </price>
               <supplier>
                  <supplierDUNS>2820</supplierDUNS>
                  <supplierName>Kareem's Corporation</supplierName>
               </supplier>
            </orderLine>
         </OrderLinesDataElements>
   </body>
</response>
```

Note: See Appendix 'A' for DTD of the shopping cart sent back from the supplier to *i*Procurement.

See Appendix 'B' for a sample XML document (shopping cart) sent back from the supplier to *i*Procurement.

---

7. *i*Procurement executes the 'AppsManager' java class to add the items to the requisition. The EDI setups in Oracle Applications are referenced to do the required data conversion

(mapping) between the external code values coming from the supplier and the internal code values setup in Oracle *i*Procurement. If the lock_item_flag (stored in the ICX_ POR_ITEM_SOURCES table) was set during configuration of the punchout for this supplier, then the user will not be able to make any changes to the item returned from the supplier. The requisition then goes through the normal workflow and approval processes configured in *i*Procurement.

Note: See the Installation and Setup section for details on setting the lock_item_flag and using the EDI setups to create data maps.

Model 4: Punch out from Oracle *i*Procurement to Supplier via Oracle Exchange: cXML

Oracle *i*Procurement users can also punch out to a cXML supplier through Oracle Exchange. Oracle Exchange simplifies the initial setup process as well as the authentication and maintenance of the punchout. To punch out to a supplier through the Exchange, the *i*Procurement customer must setup Oracle Exchange as a supplier hub in *i*Procurement and their suppliers must setup a punchout from Oracle Exchange to their web store.

Setup Suppliers on Oracle Exchange

The suppliers can setup their punchout definitions one time on Oracle Exchange. Then an *i*Procurement administrator can download these definitions, which will automatically setup the punchout from *i*Procurement to that supplier.

Note: See the chapter 'Defining an Oracle Exchange Punchout' for details on setting up a supplier for punchout from an Oracle Exchange.

Setup Oracle Exchange as a Supplier Hub in iProcurement

The Oracle Exchange to use as a supplier hub must first be setup in *i*Procurement. For details on the setup required, refer the implementation and installation section later in this document

Setup Punchout Suppliers from Oracle Exchange in iProcurement

The *i*Procurement administrator logs on to *i*Procurement and initiates the process of setting up suppliers they want to enable for punchout. *i*Procurement executes 'PunchOutServlet' which first establishes a secure connection with Oracle Exchange. Once established, 'PunchOutServlet then generates the 'OEXLoginRequest' XML document and sends it to

Oracle Exchange. After the *i*Procurement user is authenticated on Oracle Exchange their browser is re-directed to Oracle Exchange, using the login URL provided in the 'LoginResponse' XML. The *i*Procurement administrator selects the punchout suppliers they want to enable in *i*Procurement. Oracle Exchange sends the suppliers' setups to *i*Procurement in the 'SupplierSyncUpResponse' XML document. *i*Procurement then executes 'PunchOutServlet' that creates all the supplier punchout related setups in iProcurement.

Part of this setup process includes the download of keywords the supplier has associated with their company during their setup on Oracle Exchange. The first time a supplier is setup in *i*Procurement these keywords will be downloaded into *i*Procurement. The *i*Procurement customer can keep these keywords or change them to meet their own user's needs. This information is not downloaded from Oracle Exchange again.

Additional setup information maintained by the supplier on Oracle Exchange (such as their punchout URL, site level userid, and password) will remain on the Exchange. This is referenced each time a punchout occurs, so the *i*Procurement administrator does not have to maintain any supplier specific login details themselves.

Punchout Process

The supplier is automatically setup in Oracle *i*Procurement as a punchout supplier in the above step and the Supplier's link is displayed on the *i*Procurement homepage.

When a user clicks the link to a supplier, the 'PunchOutServlet' servlet is executed. First, this calls Oracle SSL APIs to establish secure connection to Oracle Exchange. These APIs request a Certification Authority Certificate from the Oracle Exchange site and compare that to Certificates stored in *i*Procurement.

After a secure connection between the *i*Procurement server and Oracle Exchange is established, 'PunchOutServlet' then generates the 'OEXLoginRequest' XML document, which contains user details and login credentials for Oracle Exchange and a return URL. Once generated, this is passed to Oracle Exchange ('LinkinLogin.jsp') through the secure connection.

Oracle Exchange uses the 'LinkInAuthBean' java class to authenticate the *i*Procurement user. Once authenticated Oracle Exchange uses the 'PunchOutLoginBean' java class to initiate a connection with the supplier site. If required (based on the setup of the supplier), Oracle Exchange will request a Certification Authority Certificate from the supplier site and compare that to Certificates stored in Oracle Exchange.

Once a secure connection is established, the 'PunchOutLoginBean' java class will generate and deliver the 'LinkRequest' XML document to the supplier site, which contains the *i*Procurement return URL. The supplier site generates a response in the form of a

‘LinkResponse’ XML document which contains the login URL and session information for the supplier’s site.

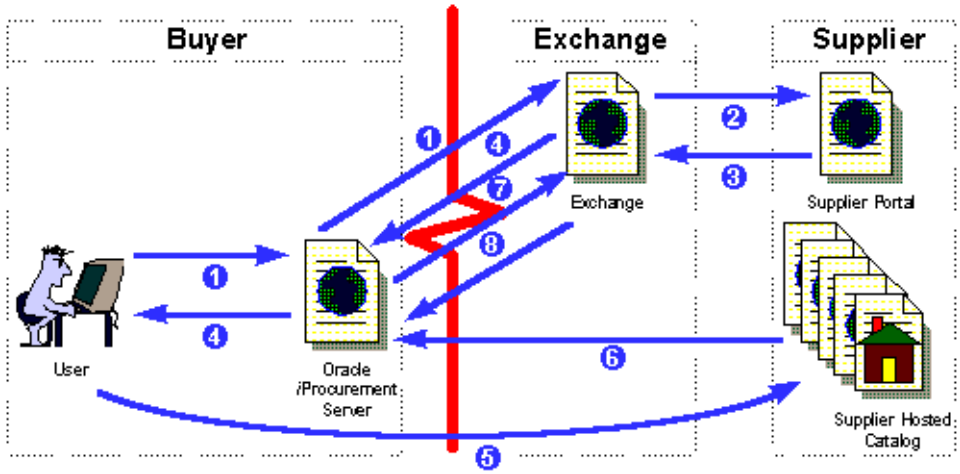
Oracle Exchange uses the LinkinMainBean java class to generate the ‘LoginResponse’ XML document that is returned to *i*Procurement.

Upon receipt of this ‘LoginResponse’ XML document by *i*Procurement, the iProcurement user’s browser is re-directed to the login URL provided by the supplier. The Oracle Exchange involvement in this process is transparent to the *i*Procurement user. The user can shop the supplier’s site and add items to a shopping cart on the supplier site. When the user finishes adding items to their shopping cart, the supplier executes a program, which takes the items in the shopping cart and creates a ‘PunchOutOrderMessage’ cXML document with the shopping cart item details. This ‘url encoded’ document is posted to the *i*Procurement return URL with the ‘cxml-urlencoded’ parameter.

The ‘cxml-urlencoded’ parameter used during the return of the shopping cart lets *i*Procurement know this is a cXML shopping cart. Since *i*Procurement cannot process the ‘PunchOutOrderMessage’ cXML document, ‘PunchOutServlet’ receives and re-directs the shopping cart to Oracle Exchange. First, the ‘PunchOutServlet’ establishes a secure and trusted connection with the Exchange, and then it forwards the cXML shopping cart.

On Oracle Exchange, ‘PunchoutMainBean’ and ‘ExportOrder’ are executed to convert the shopping cart from cXML to XML and also do the code conversions (UOM and currency) defined on the Oracle Exchange. This shopping cart is then returned to iProcurement through the secure connection.

*i*Procurement executes the ‘AppsManager’ java class to add the items to the requisition. The EDI setups in Oracle Applications are referenced to do the required data conversion (mapping) between the external code values coming from the supplier and the internal code values setup in Oracle *i*Procurement. If the lock_item_flag was not set during configuration of the punchout for this Exchange, then the user will be able to make any changes to the item returned from Oracle Exchange. The requisition then goes through the normal workflow and approval processes configured in *i*Procurement.



1. The *i*Procurement user starts on their homepage, which displays a link to a supplier site. When the user clicks the link, the browser connects to the *i*Procurement server and the 'PunchOutServlet' is executed. This servlet first calls Oracle SSL APIs to establish a secure connection with Oracle Exchange. The APIs request the Certification Authority (CA) digital certificate (public key) from Oracle Exchange. This is compared to certificate authorities stored in the ca-bundle.crt file in *i*Procurement.

After the secure connection is established the 'PunchOutServlet' generates the 'OEXLoginRequest' XML document, that includes a base set of user details and the return URL for the *i*Procurement instance. This XML document is passed to the Oracle Exchange site.

'OEXLoginRequest' sample:

```
<?xml version = '1.0'?>
<request>
   <header version="1.0">
      <login>
         <username>buyer1</username>
         <password>WELCOME</password>
      </login>
      <action>shopping</action>
      <language>US</language>
      <userArea>
         <operatingUnit>204</operatingUnit>
         <shipTo>V1- New York City</shipTo>
```

```
         <deliverTo>V1- New York City</deliverTo>
      </userArea>
   </header>
   <body>
      <loginRequest>
         <userInfo>
            <userName>Green, Mr. Terry</userName>
            <appUserName>TGREEN</appUserName>
            <userContactInfo>
               <userEmail>tgreen@vision.com</userEmail>
            </userContactInfo>
         </userInfo>
         <companyInfo>
            <companyName>IP</companyName>
         </companyInfo>
<partySiteId>
        2963
</partySiteId>
         <reqToken>
template=tpn*@_action=addToOrder*@_function=addToOrder*
 @_reqSessionId=0*@_reqHeaderId=0*@_emergencyReq=N*@_itemSourceId
=1001
         </reqToken>
         <returnURL>
http://iprocurement.company.com/oa_
servlets/oracle.apps.icx.por.punchout.PunchOutServlet?callBack=shopping
         </returnURL>
      </loginRequest>
   </body>
</request>
```

Note: The DTDs are described in detail in the Appendix ‘A’ of this document.

Some additional optional user data can also be sent to Oracle Exchange if the punchout was setup in iProcurement to send it. The sample XML document for a login request with additional optional user data is attached in Appendix ‘B’ of this document. For details on how to include the additional information see the Installation and Implementation section.

Oracle Exchange receives the ‘OEXLoginRequest’ with the ‘LinkinLogin.jsp’ and executes the ‘LinkinAuthBean’ java class to authenticate the *i*Procurement user.

2. After the *i*Procurement user is authenticated on Oracle Exchange, the Exchange executes the 'PunchOutLoginBean' java class. To establish a secure connection to the supplier site, it calls Oracle SSL APIs (if required based on the supplier's setup) which requests the Certification Authority (CA) digital certificate (public key) from the supplier site. This is compared to certificate authorities setup by the Oracle Exchange operator. Once certified a secure connection is opened between Oracle Exchange and the supplier site.

   The 'PunchOutLoginBean' java class then generates the 'PunchOutSetupRequest' cXML document. It includes the site level user name and password that the supplier requires to access their web store.

   Note: The site level user name is stored and maintained on Oracle Exchange by the supplier. There is no need for the iProcurement sysadmin to store this information.

   The 'PunchOutSetupRequest' includes the *i*Procurement return URL as well as user and company details from the *i*Procurement user that were included in the initial 'OEXLoginRequest' from *i*Procurement to Oracle Exchange.

   'PunchOutSetupRequest' cXML sample:

```
<?xml version = '1.0' encoding = 'ISO-8859-1'?>
<!DOCTYPE cXML SYSTEM "http://xml.cxml.org/schemas/cXML/1.1.007/cXML.dtd">
<cXML version="1.1.007" xml:lang="en-US" payloadID="Wed Apr 04 12:52:46
GMT-08:00 2001" timestamp="Wed Apr 04 12:52:46 GMT-08:00 2001">
   <Header>
      <From>
         <Credential domain="DUNS">
            <Identity>654321</Identity>
         </Credential>
      </From>
      <To>
         <Credential domain="DUNS">
          <Identity>123456</Identity>
         </Credential>
      </To>
      <Sender>
         <Credential domain="OracleExchangeUserId">
            <Identity>buyer@acme.com</Identity>
            <SharedSecret>supplier_password</SharedSecret>
```

```
         </Credential>
         <UserAgent>Exchange_name</UserAgent>
      </Sender>
   </Header>
   <Request>
      <PunchOutSetupRequest operation="create">
         <BuyerCookie/>
         <Extrinsic name="User">JSMITH</Extrinsic>
         <BrowserFormPost>
            <URL>
http://iprocurement.company.com/oa_
servlets/oracle.apps.icx.por.punchout.PunchOutServlet?callBack=shopping
            </URL>
         </BrowserFormPost>
         <Contact>
            <Name xml:lang="en-US">jsmith</Name>
            <Email>buyer@acme.com</Email>
         </Contact>
      </PunchOutSetupRequest>
   </Request>
</cXML>
```

3. The supplier site generates a response in the form of a 'PunchOutSetupResponse' cXML document, which contains the login URL with session information for the supplier's site.

'PunchOutSetupResponse' cXML sample:

```
<?xml version="1.0" ?>
<cXML payloadID="2000-08-15" timestamp="2000-08-15">
  <Response>
    <PunchOutSetupResponse>
      <StartPage>
        <URL>
http://www.suppliersite.com/orders/LinkinCallback.jsp
?sessionId=3fe0a73f6fa1c003.164.982359182161&action=shopping
        </URL>
      </StartPage>
    </PunchOutSetupResponse>
    <Status code="200" text="success" />
  </Response>
</cXML>
```

4. The 'PunchOutSetupResponse' is returned to the 'LinkinLogin.jsp' on Oracle Exchange. Oracle Exchange then uses the 'LinkinMainBean' java class to generate the

'LoginResponse' XML and sends it back to iProcurement using the return URL. The XML document contains the session aware URL which *i*Procurement user will use to connect to the supplier site.

'LoginResponse' XML sample:

```
<?xml version = '1.0'?>
<response>
  <header version="1.0">
    <action>LOGIN_RESPONSE</action>
  </header>
  <body>
    <loginInfo>
      <loginURL>
http://www.suppliersite.com/orders/LinkinCallback.jsp
?sessionId=3fe0a73f6fa1c003.164.982359182161&action=shopping
     </loginURL>
    </loginInfo>
  </body>
</response>
```

5. Using the session aware login URL provided in the 'LoginResponse' XML, *i*Procurement executes the 'PunchOutServlet', which redirects the *i*Procurement user's browser to the login URL provided by the supplier. The user is allowed to search and add items to their shopping cart on the supplier site.

5. When the user finishes adding items to their cart on the supplier site, the items in the shopping cart are sent back to *i*Procurement by posting a form to the return URL received from *i*Procurement during the initial punchout. The parameter for the post should be 'cxml-urlencoded'. The data is sent back in the 'PunchOutOrderMessage' XML document which is 'url encoded'.

'PunchOutOrderMessage' XML sample:

```
<?xml version = "1.0" encoding = "UTF-8"?>
<cXML version = "1.1.007" xml:lang = "en-US"
     payloadID = "99999" timestamp = "2000-08-15">
  <Header>
    <From>
      <Credential domain = "DUNS">
        <Identity>888</Identity>
      </Credential>
    </From>
    <To>
      <Credential domain = "DUNS">
        <Identity>888</Identity>
```

```
      </Credential>
    </To>
    <Sender>
      <Credential domain = "DUNS">
        <Identity>888</Identity>
      </Credential>
      <UserAgent>Workchairs cXML</UserAgent>
    </Sender>
  </Header>
  <Message>
    <PunchOutOrderMessage>
      <BuyerCookie>ICX</BuyerCookie>
      <PunchOutOrderMessageHeader
           operationAllowed = "edit">
        <Total>
          <Money currency = "USD">112.4</Money>
        </Total>
      </PunchOutOrderMessageHeader>
      <ItemIn quantity = "3">
        <ItemID>
          <SupplierPartID>5555</SupplierPartID>
        </ItemID>
        <ItemDetail>
          <UnitPrice>
            <Money currency = "USD">763.20</Money>
          </UnitPrice>
          <Description xml:lang = "en">
             Leather reclining chair
          </Description>
          <UnitOfMeasure>EA</UnitOfMeasure>
          <Classification domain = "UNSPSC">
            513333
          </Classification>
        </ItemDetail>
      </ItemIn>
    </PunchOutOrderMessage>
  </Message>
</cXML>
```

Note: See Appendix 'A' for DTD of the shopping cart sent back from the supplier to *i*Procurement.

See Appendix 'B' for a sample XML document (shopping cart) sent back from the supplier to *i*Procurement.

6. The 'cxml-urlencoded' parameter used to post the shopping cart lets *i*Procurement know this is a cXML shopping cart. Since *i*Procurement cannot process the 'PunchOutOrderMessage' cXML document, 'PunchOutServlet' receives and re-directs the shopping cart to Oracle Exchange. The 'PunchOutServlet' establishes a secure and trusted connection with the Exchange, then it forwards the cXML shopping cart.
7. Oracle Exchange executes 'PunchoutMainBean' and 'ExportOrder' which convert the shopping cart from cXML to XML and also do the code conversions (UOM and currency) defined on the Oracle Exchange. This shopping cart is then returned to iProcurement through the secure connection.
8. *i*Procurement executes the 'AppsManager' java class to add the items to the requisition. The EDI setups in Oracle Applications are referenced to do the required data conversion (mapping) between the external code values coming from the supplier and the internal code values setup in Oracle *i*Procurement. If the lock_item_flag (stored in the ICX_ POR_ITEM_SOURCES table) was set during configuration of the punchout for this supplier, then the user will not be able to make any changes to the item returned from the supplier. The requisition then goes through the normal workflow and approval processes configured in iProcurement.

Note: See the Installation and Setup section for details on setting the lock_item_flag and using the EDI setups to create data maps.

Model 5: Punch out from Oracle *i*Procurement directly to Supplier: XML

A supplier is setup as a punchout (or external) supplier in *i*Procurement and the supplier site link is displayed on the *i*Procurement homepage. This allows *i*Procurement users to punch out to a supplier's web store, fill a shopping cart with requisition items and return the cart to *i*Procurement for approval routing and purchase order creation.

This model has been supported in Oracle *i*Procurement since patch 'H' and has been re-architected in patch 'J'. The old model however continues to be supported as well. The new architecture is described below.

Setup Supplier as a Punchout (external) supplier in iProcurement

The supplier you want to punch out to must first be setup in *i*Procurement. For details on the setup required, refer the implementation and installation section later in this document.

Punchout Process

When a user clicks the link to the supplier, the 'PunchOutServlet' servlet is executed. If required this will first call Oracle SSL APIs to establish a secure connection to the supplier site. These APIs request a Certification Authority Certificate from the supplier site and compare that to Certificates stored in *i*Procurement.

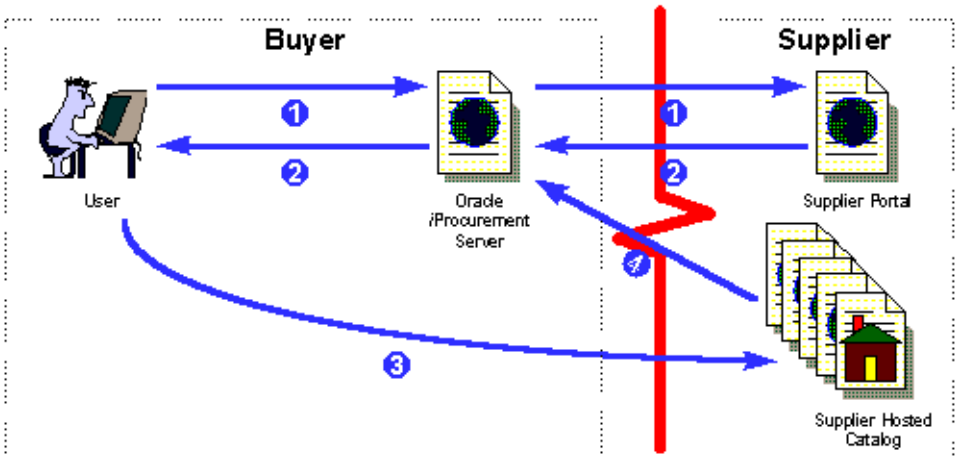
After a secure connection between *i*Procurement and the supplier is established, 'PunchOutServlet' then generates the 'SupplierLoginRequest' XML document, which contains user details and login credentials for the supplier site and a return URL. Once generated, this is passed to the supplier through the secure connection.

The supplier can authenticate the *i*Procurement user. Once authenticated the supplier must generate the 'LoginResponse' XML document that is returned to *i*Procurement

After successfully establishing a secure connection and authenticating the *i*Procurement instance, the *i*Procurement user's browser is re-directed to the supplier's site using the login URL provided by the supplier in the 'LoginResponse' XML document. The user is then allowed to search and add items to a shopping cart on the supplier site.

After the user finishes adding items to their shopping cart, the supplier must create the 'OrderLinesDataElements' XML document, which contains the shopping cart items added from their site. This is posted to the return URL on the *i*Procurement server using the 'oracleCart' parameter.

*i*Procurement executes the 'AppsManager' java class to add the items to the requisition. The EDI setups in Oracle Applications are referenced to do the required data conversion (mapping) between the external code values coming from the supplier and the internal code values setup in Oracle *i*Procurement. And if the lock_item_flag was not set during configuration of the punchout for this supplier, then the user will be able to make changes to the item returned from the supplier site. The requisition then goes through the normal workflow and approval processes configured in *i*Procurement.



1. The *i*Procurement user starts on their homepage, which displays a link to an external supplier. When the user clicks the supplier link, the browser connects to the *i*Procurement server and the 'PunchOutServlet' is executed. This servlet first calls Oracle SSL APIs to establish a secure connection with the supplier site (if required). The APIs request the Certification Authority (CA) digital certificate (public key) from the supplier site. This is compared to certificate authorities stored in the ca-bundle.crt file in *i*Procurement.

   After the secure connection is established the 'PunchOutServlet' generates the 'SupplierLoginRequest' XML document, which includes a base set of user details and the return URL for the *i*Procurement instance. This XML document is passed to the supplier site.

   'SupplierLoginRequest' sample:

```
<?xml version = '1.0'?>
<request>
   <header version="1.0">
      <login>
         <password>
            welcome
         </password>
      </login>
      <action>
         SHOPPING
```

```
      </action>
      <language>
         US
      </language>
      <userArea>
         <operatingUnit>
            204
         </operatingUnit>
         <shipTo>
            V1- New York City
         </shipTo>
         <deliverTo>
            V1- New York City
         </deliverTo>
      </userArea>
   </header>
   <body>
      <loginInfo>
         <exchangeInfo>
            <exchangeName>
               Oracle
            </exchangeName>
         </exchangeInfo>
         <userInfo>
            <userContactInfo>
               <userPhone/>
               <userEmail>
                  tgreen@vision.com
               </userEmail>
            </userContactInfo>
            <userCompany>
               <companyName>
                  Oracle
               </companyName>
               <companyDUNS/>
               <contactName/>
               <contactPhone/>
            </userCompany>
            <userName>
               Green, Mr. Terry
            </userName>
         </userInfo>
<returnURL>
http://iprocurement.oracle.com/oa_
servlets/oracle.apps.icx.por.punchout.PunchOutServlet?callBack=shopping
```

```
        </returnURL>
      </loginInfo>
   </body>
</request>
```

> **Note:** The DTDs are described in detail in the Appendix 'A' of this document.
>
> Some additional optional user data can also be sent to Oracle Exchange if the punchout was setup in iProcurement to send it. The sample XML document for a login request with additional optional user data is attached in Appendix 'B' of this document. For details on how to include the additional information see the Installation and Implementation section.

2. The supplier receives the 'SupplierLoginRequest' and authenticates the *i*Procurement user. Once authenticated, the supplier must generate the 'LoginResponse' XML and send it back to the Oracle *i*Procurement.

'LoginResponse' sample:

```
<?xml version = '1.0'?>
<response>
  <header version="1.0">
    <action>LOGIN_RESPONSE</action>
  </header>
  <body>
    <loginInfo>
      <loginURL>
http://www.suppliersite.com/orders/LinkinCallback.jsp
?sessionId=3fe0a73f6fa1c003.164.982359182161&action=shopping
     </loginURL>
    </loginInfo>
  </body>
</response>
```

3. Now that a secure and trusted connection is established, the *i*Procurement user's browser is re-directed to the session aware login URL provided by the supplier. Now the user is allowed to search and add items to their shopping cart on the supplier site.

4. When the user finishes adding items to their cart on the supplier's site, the items in the shopping cart are posted to the *i*Procurement server (with the 'oracleCart'parameter) using the return URL received from *i*Procurement during the initial punch out (see step 1 of this section). The data must be sent back in the 'OrderLinesDataElements' XML document which is 'url encoded'.

   'OrderLinesDataElements' sample:

```
<?xml version = '1.0' encoding = 'ISO-8859-1'?>
<response>
   <header>
      <action>SHOPPING_RESPONSE</action>
   </header>
   <body>
      <OrderLinesDataElements>
         <catalogTradingPartner>
OracleExchange
  </catalogTradingPartner>
         <orderLine>
            <contract>
               <catalogType>CONTRACTED</catalogType>
            </contract>
            <item lineType="GOODS" quantity="1.0">
               <itemNumber>
                  <supplierItemNumber>
                     <itemID>kb_Item2</itemID>
                  </supplierItemNumber>
               </itemNumber>
               <itemDescription>
                   Item #2 : kbenjami- list, TBD b.s.
               </itemDescription>
               <unitOfMeasure>
                  <buyerUnitOfMeasure/>
                  <supplierUnitOfMeasure>
                     <supplierUOMType>Bushel</supplierUOMType>
                     <supplierUOMQuantity></supplierUOMQuantity>
                  </supplierUnitOfMeasure>
               </unitOfMeasure>
            </item>
            <category>
               <categoryCode categoryCodeIdentifier="SPSC">
                   Ball Point Pens
               </categoryCode>
            </category>
            <price>
```

```
                <currency>USD</currency>
                <unitPrice>100.0</unitPrice>
             </price>
             <supplier>
                <supplierDUNS>2820</supplierDUNS>
                <supplierName>Kareem's Corporation</supplierName>
             </supplier>
          </orderLine>
       </OrderLinesDataElements>
    </body>
</response>
```

Note: See Appendix 'A' for DTD of the shopping cart sent back from the supplier to *i*Procurement.

See Appendix 'B' for a sample XML document (shopping cart) sent back from the supplier to *i*Procurement.

5. *i*Procurement executes the 'AppsManager' java class to add the items to the requisition. The EDI setups in Oracle Applications are referenced to do the required data conversion (mapping) between the external code values coming from the supplier and the internal code values setup in Oracle *i*Procurement. If the lock_item_flag (stored in the ICX_ POR_ITEM_SOURCES table) was set during configuration of the punchout for this supplier, then the user will not be able to make any changes to the item returned from the supplier site. The requisition then goes through the normal workflow and approval processes configured in *i*Procurement.

Note: See the Installation and Setup section for details on setting the lock_item_flag and using the EDI setups to create data maps.

Installation / Implementation Steps

For each of the punchout methods discussed in the previous section, there is setup required in *i*Procurement and either Oracle Exchange or on the Supplier's web store site. This section addresses the setup that needs to occur for each type of punchout model. (Model 1, punchout from Oracle Exchange to supplier, does not involve Oracle iProcurement).

- Model 2: Punch out from Oracle *i*Procurement to Oracle Exchange: XML

- Model 3: Punch out from Oracle *i*Procurement to Supplier via Oracle Exchange: XML
- Model 4: Punch out from Oracle *i*Procurement to Supplier via Oracle Exchange: cXML
- Model 5: Punch out directly to Supplier: XML

In the following section each scenario is described in more detail and includes a list of the setup processes to complete to make it work. The setup processes are detailed in the subsequent section.

Model 2: Punch out from Oracle *i*Procurement to Oracle Exchange: XML

When you setup a punchout from *i*Procurement to Oracle Exchange, it allows *i*Procurement users to search and fill a shopping cart with catalog items hosted on an Oracle Exchange.

When the authenticated *i*Procurement user punches out to Oracle Exchange, an XML document is sent with the user's company details to Oracle Exchange. The company information sent from *i*Procurement is the same as the company defined in Oracle Exchange. A proxy user with a certain profile is attached to this company in Oracle Exchange and the *i*Procurement user, by default, inherits the proxy user's job functions on the Exchange. This enables the *i*Procurement administrator to restrict the types of activities an *i*Procurement user can actually perform on Oracle Exchange.

In most cases, the *i*Procurement user is restricted to only being able to view items with the *i*Procurement company's pre-negotiated pricing. Typically they would not access Oracle Exchange functionality that allows them to shop open market catalogs, create or bid in auctions, use Supply Chain Collaboration tools or Product Design collaboration tools.

To set this up, complete the following:

Oracle Exchange Setups

- Setup Company on Oracle Exchange
- Setup Site Level User for *i*Procurement on Oracle Exchange

*i*Procurement Setups

- Setup Profile Options
- Maintain Supplier Hubs
- Resequence punchout display (optional)
- Maintain punchout suppliers (ongoing)

Oracle eCommerce Gateway Setups

- Setup Trading Partner
- Setup Conversion Code

Note: The setup processes listed above are detailed in the subsequent section.

Model 3 & 4: Punch out from Oracle *i*Procurement to Supplier via Oracle Exchange: XML or cXML

When you setup a punchout from *i*Procurement to a supplier's web store it allows *i*Procurement users to search and add items to a shopping cart on the suppliers web site and then bring those details back to *i*Procurement to complete the requisition process and have a purchase order created.

The *i*Procurement user clicks a link to the supplier site on their *i*Procurement home page. *i*Procurement will then authenticate to Oracle Exchange and Oracle Exchange will authenticate to the supplier site. The user will be presented with the supplier's web site and allowed to fill and return a shopping cart to *i*Procurement.

To set this up, complete the following:

Oracle Exchange Setups

- Setup Company on Oracle Exchange
- Setup Site Level User for *i*Procurement on Oracle Exchange

*i*Procurement Setups

- Setup Profile Options
- Maintain Supplier Hubs
- Download Punchout Enabled Suppliers from Hub
- Resequence punchout display (optional)
- Maintain punchout suppliers (ongoing)

Oracle eCommerce Gateway Setups

- Setup Trading Partner
- Setup Conversion Code

Note: The setup processes listed above are detailed in the subsequent section.

Model 5: Punch out directly to Supplier: XML

In this case the *i*Procurement user clicks a link to a supplier site from the *i*Procurement home page and is taken directly to the suppliers site. Once there they can add items to a shopping cart and return the cart to *i*Procurement to complete the requisitioning process.

To set this up, complete the following:

*i*Procurement Setups

- Setup Profile Options
- Create Direct Punchout Suppliers
- Resequence punchout display (optional)
- Maintain punchout suppliers (ongoing)

Oracle eCommerce Gateway Setups

- Setup Trading Partner
- Setup Conversion Code

Note: The setup processes listed above are detailed in the subsequent section.

Oracle Exchange Setup Steps

Setting up your company on Oracle Exchange

- On the Oracle Exchange home page, click ‘Registration’
- Select ‘I want to Register a new company as an independent entity’
  - Step 1: Accept the Administrator Role--whoever registers their company for the first time on Oracle Exchange is the default administrator for the company account. Responsibilities include approving new user accounts for company employees and administering the Company’s account on the Exchange
  - Step 2: Company Information--provide company information including address, phone number, etc.
  - Step 3: Exchange Payment—specifies how the company will pay for charges they incur using the Exchange. This information will be used to collect any transaction fees that accrue from using the Exchange.
  - Step 4: Personal Information—the personal information related to the person creating the account. This will be used to setup an individual user account for the company administrator, in addition to the company account being created.
  - Step 5: Legal Agreement—accept the Terms and Conditions of use applicable to the Exchange where the registration is occurring. These are established by the Oracle Exchange Operator.
  - Step 6: Review and Submit—review all the information that has been entered during the registration process. Once verified, submit the registration for approval by the Exchange operator.

After the registration information is submitted, the approval of the company account is not automatic. Each Oracle Exchange operator can decide whether or not to approve or reject a company's registration. The time for approval of an account on the Exchange will vary by operator.

Setting up a site level user for iProcurement on Oracle Exchange

When the company account is approved on Oracle Exchange, then a proxy user must be setup in Oracle Exchange for the company created in the above step.

- On the Oracle Exchange home page, click 'Registration.'
- Select 'I think my company is already registered, and I want to register myself as a new user.'
- Step 1: Find Your Company—you are registering as a user of the company account created in the step above. On this screen search for and select the company name you previously created.
- Step 2: Personal Information—enter the personal information for this user account. You may want to specify the name as IP User, or Proxy User, or some variation.
- Step 3: Legal Agreement—this agreement should have been reviewed and accepted on behalf of the company during the registration of the company account. It is provided during the user registration for informational purposes.
- Step 4: Review and Submit—review all the information that has been entered during the registration process. Once verified, submit the registration for approval by the company administrator.

The company administrator (usually the person that registered the company in the previous step) must approve this user on Oracle Exchange. To approve this user so the proxy account is active on the exchange, the company administrator should:

- Login to the Oracle Exchange where the company's account was created.
- Navigate to the Admin (or Company Account) tab. Under the 'Users' heading, click 'Approve Users.'
- Select the user and click 'Approve' button.
- Select the user again and click the 'Assign Job Functions' button.
- Under the 'Job Functions' heading, choose 'Select and add existing job functions.'
- At this point the administrator is presented with a list of possible job functions to be assigned to the *i*Procurement proxy user. If you want your company employees to be able to shop on Oracle Exchange, but only from those catalogs that have buyer specific

pricing available for your company, select the following job functions: Buyer with Restricted Pricing View.

At this point you have completed all the setup necessary on Oracle Exchange.

Note: Any job functions assigned to this user on Oracle Exchange will be available to all *i*Procurement users that punch out from *i*Procurement to Oracle Exchange.

Oracle *i*Procurement Setup Steps

Now that you have created company and user accounts on Oracle Exchange, you can setup the Exchange as an external supplier, which will be available to users from the *i*Procurement home page or as part of the search results.

Note: *i*Procurement users that punch out from *i*Procurement to Oracle Exchange must be associated with an Oracle ERP employee.

Setting up profile options

3 new system level profile values in the *i*Procurement setup section :

- POR : Proxy Server Name - Proxy server name if the customer has a proxy setup
- POR : Proxy Server Port - Proxy server port if the customer has a proxy setup
- POR : CA Certificate File Name - Absolute file path of ca-bundle.crt where all the certificates' names are stored

Maintaining supplier hubs

To setup an Oracle Exchange as a punchout supplier, navigate to the eContent Manager in *i*Procurement.

- From the home page of the eContent Manager, select the link for 'Maintain Supplier Hubs.'
- Click 'Create New' button to setup a new supplier hub.

- Enter information in the fields on this page to complete the setup.

Table 3–1 Supplier Hub Setup

Field Name	Required	Description
Supplier Hub Name	Yes	The name that will be displayed on the iProcurement home page and in search results during searches performed by iProcurement users
Hub Description	No	Displayed on the external supplier directory page in iProcurement. Only visible when a user requests a list of all external suppliers (example: The Exchange for Big Business, Small Business, All Business)
URL	Yes	The URL that iProcurement will use to login to Oracle Exchange. For any Oracle Exchange, it should be the following: http://www.<exchangename>.com/orders/LinkinLogin.jsp Example: http://exchange.oracle.com/orders/LinkinLogin.jsp
Image URL	No	Optional field to specify the web address of the Exchange's logo. For Oracle Exchange's the URL will be: http://www.<exchangename>.com/brands/<xx>/marketing/<yy>/logo.gif Where <xx> represents the database of the exchange instance. Ask the Exchange operator you are creating a link to for this information. Where <yy> represents the language code (e.g. US for American English, D for German, F for French, JA for Japanese, etc.). There may be a different logo for each language the Exchange supports. (example: https://exchange.oracle.com/brands/ox/marketing/US/logo.gif)
Company Name	Yes	The Company name of the proxy user established on Oracle Exchange. This must be entered exactly the same as it was entered during the registration process on Oracle Exchange. If you are unsure of the registered company name, it can be found in the Trading Partner Directory on the Oracle Exchange (link from the Exchange home page).
Username	Yes	The username of the proxy user created on Oracle Exchange. This will be used each time an iProcurement user punches out to Oracle Exchange and is validated during the setup of the punchout session.

Table 3–1 Supplier Hub Setup

Field Name	Required	Description
Password	Yes	The password for the proxy user account created on Oracle Exchange. This is used in combination with the Username to establish the punchout session.
Retype Password	Yes	Retype the password.
Keywords	No	Any words entered in this field will be included when an iProcurement user searches for items on iProcurement. If any words match the search criteria entered by the iProcurement user, the link to this supplier hub will be displayed along with the other search results.
Always display this hub with search results	No	If checked, this Supplier Hub will be displayed in the search results for any search performed in iProcurement
Items added to shopping cart from external punchout sites can be modified in iProcurement	No	If checked, the iProcurement user will be allowed to modify shopping cart items returned from any external site before the cart is submitted. If you do not want the iProcurement user to modify price, quantity, etc. of the items added from external sites, leave this box unchecked. This is the 'lock_item_flag' earlier referenced in the document.
Send optional user and company information when connecting to the hub site	No	If checked, the login request XML will include additional user data when it is sent to the supplier hub or supplier web store. See details of additional information in Appendix B: Sample XML Documents

Once all the data is entered, click the ‘Create’ button to finish the setup

Now your *i*Procurement users will see a link to this supplier hub on the *i*Procurement home page or in the external supplier's directory.

Downloading punchout enabled suppliers from hub

If you have suppliers that have already enabled a punchout from Oracle Exchange to their web store, you can use this setup to download that setup information into *i*Procurement.

Once you complete this setup you can punch out from *i*Procurement to a supplier’s web store and add items to your shopping cart from their site and then return the shopping cart to *i*Procurement to go through the approval process.

Note that this functionality will only work with Oracle Exchanges (version 6.1c and higher) that have an Oracle Exchange setup as a supplier hub and with punchout suppliers enabled on the Exchange.

- Login to *i*Procurement and navigate to the eContent Manager Home Page and then select 'Download Punchout Enabled Suppliers from the Hub.'
- Select 'Supplier Hub Name' to view a list of suppliers available on that Oracle Exchange.
- You will be presented with a list of Suppliers that have a punchout setup with the Oracle Exchange you selected.
- Select the suppliers you want to download setup information for into *i*Procurement. Each of the suppliers you select at this point will be made available to your *i*Procurement users as an external supplier.
- Select 'Sync Suppliers' button. This will return you to *i*Procurement and display a confirmation message, letting you know the setup information for the suppliers you selected was successfully saved.

Creating direct punchout suppliers

When you have a supplier you would like to setup a direct punchout to, you can setup their punchout information in this section. This option requires additional development efforts by the supplier on their web site.

If you are going to do a direct punchout to a supplier site, they must be able to receive the login request detailed in the solution section of this document. They must also be able to generate a login response in the format *i*Procurement expects, as well as be able to generate and send a shopping cart in the format supported by iProcurement.

Once a supplier has met these criteria on their site, use this functionality to setup the parameters and enable them as an external punchout supplier in *i*Procurement.

- Navigate to the iProcurement eContent Manger home page.
- Select the link to 'Create Direct Punchout Suppliers.'
- Enter information in the following fields to complete the setup.

Table 3–2 Direct Punchout Setup

Field Name	Required	Description
Punchout Supplier	Yes	The name of the supplier that will be displayed on the iProcurement home page and in search results during searches performed by iProcurement users (example: Supplier Web Store)

Table 3–2 Direct Punchout Setup

Field Name	Required	Description
Description	No	Displayed on the external supplier directory page in iProcurement. Only visible when a user requests a list of all external suppliers (example: The Big Business Supplier)
URL	Yes	The URL that iProcurement will use to login to the supplier site. It should include not only the URL, but the program name that will receive and process the login request from iProcurement. (example: http://www.suppliersite.com/buyerLogin.jsp)
Image URL	No	Optional field to specify the web address of the Supplier's logo. (example: http://www.suppliersite.com/logo.gif)
Company Name	Yes	The Company name or user id the supplier will validate during setup of the punchout. The supplier you are setting up a punchout with should provide the value.
Password	Yes	The password for the company name provided above. This will be used as a site level password at the supplier site. The supplier you are setting up a punchout with should provide the value.
Retype Password	Yes	Retype the password.
Keywords	No	Any words entered in this field will be included when an iProcurement user searches for items on iProcurement. If any words match the search criteria entered by the iProcurement user, the link to this supplier site will be displayed along with the other search results.
Always display this supplier with search results	No	If checked, this Supplier will be displayed in the search results regardless of the search criteria entered by the iProcurement user.
Items added to shopping cart from external punchout sites can be modified in iProcurement	No	If checked, the iProcurement user will be allowed to modify shopping cart items returned from the supplier site before the cart is submitted. If you do not want the iProcurement user to modify price, quantity, etc. of the items added from the supplier site, leave this box unchecked. This is the 'lock_item_flag' earlier referenced in the document.
Send optional user and company information when connecting to the supplier site	No	If checked, the login request XML will include additional user data when it is sent to the supplier site. See details of additional information in Appendix B: Sample XML Documents

Maintaining punchout suppliers

This page allows the *i*Procurement administrator to maintain any of the setup information for a punchout supplier setup previously. This encompasses both direct punchout suppliers and suppliers downloaded from Oracle Exchange.

- Navigate to the *i*Procurement eContent Manager home page.
- Select the link to 'Maintain Punchout Suppliers'
- The *i*Procurement is presented with a list of all punchout suppliers enabled in iProcurement.
  - Supplier column--downloaded from Oracle Exchange or entered in the punchout supplier field during a direct punchout supplier setup
  - Description—downloaded from Oracle Exchange or entered in the description field during a direct punchout supplier setup
  - Source – identifies where the supplier setup information originated. If the supplier was downloaded from an Oracle Exchange this column will contain the supplier hub name entered during set up of the hub. If the supplier was setup as a direct punchout supplier in *i*Procurement, the value in this column is 'Internal'
- Click the supplier name to edit the supplier-specific details.
  - Punchout supplier--the name that is displayed to *i*Procurement users. For Oracle Exchange suppliers this is automatically downloaded during the setup process. For direct suppliers, the *i*Procurement administrator enters this during the setup process.
  - Description--used when this supplier is displayed in the complete list of external suppliers. For Oracle Exchange suppliers this is automatically downloaded during the setup process. For direct suppliers, the *i*Procurement administrator enters this during the setup process.
  - Keywords—used when an *i*Procurement user is searching for catalog items. The search engine will include these words and if a match is found during the search, a link to the supplier site will be included on the search results page. For Oracle Exchange suppliers this is automatically downloaded during the setup process, but can be subsequently change by the iProcurement administrator. For direct suppliers, the *i*Procurement administrator enters this during the setup process.
  - See step 3 under the 'Create Direct Punchout Suppliers' section for descriptions of the remaining options.
- Once you have completed your changes, click the Update button to make your changes.

Re-Sequencing the punchout display

This page controls the order in which all punchout suppliers (including Oracle Exchange) are listed in *i*Procurement. The first few suppliers in the list appear on the *i*Procurement home page. Once the list is too long to display all suppliers on the home page a 'more…' link is displayed after the first few suppliers.

Clicking this link provides a list of all punchout suppliers enabled in *i*Procurement. This page will let you control the order these suppliers are listed.

- Navigate to the *i*Procurement eContent Manager home page.
- Choose the link to 'Re-sequence Punchout Display.'
- A list of all punchout suppliers enabled in *i*Procurement is displayed.
- Enter the sequence number in the left column, according to how you want the list of suppliers ordered when it is displayed to *i*Procurement users.
- Click the 'Update' button to re-order the list.

Note: After supplier setup the documents exchanged between *i*Procurement and the supplier can be seen in the debug log file and jvm error log file. The location of debug log file is determined in ssp_init.txt.

Oracle eCommerce Gateway Setup Steps

- Setup Trading Partner
- Setup Conversion Code

Setting up your trading partner

Navigate to: eCommerce Gateway > Setup > Trading Partner

Trading Partner: Enter the Supplier name here. The trading partner (supplier) first needs to be defined in Oracle Purchasing. Supplier details will then default here. These are the suppliers of the items selected from Oracle Exchange.

Important! The site for this supplier must have the EDI Location set to either the DUNS number for this trading partner (if they have a DUNS number and that DUNS number is associated with this trading partner on the Exchange) or the Oracle Trading Partner id from the exchange (if this is a punchout to the Exchange) or an agreed unique reference that the supplier will return within the <SupplierDUNS> tag (if this is a punchout to a supplier).

Setting up your code conversions

This setup maps the Supplier side codes to Oracle *i*Procurement.

- Navigate to: eCommerce Gateway > Setup > Code Conversion > Define Code Conversion Values
- Category: Here you can choose from a host of values like Item Categories, UOM, and Currency.
  - All Item Categories must be mapped between the supplier and *i*Procurement. Without mapping the requisition will fail validation. See the example below.
- Description: Enter the description of the mapping.
- Direction: Enter the direction as IN or BOTH. Use IN direction if there are more than one supplier code values to map into one value of Oracle Purchasing codes.
- Key 1: Enter 'OracleExchange' if this mapping is for a punchout to Oracle Exchange. Enter the supplier name if the punchout is to a supplier. Key 2 through Key 5 and External 2 through External 5 are not used.
- Internal: Oracle Purchasing codes values
- External Values: Supplier side codes values

See the *Oracle eCommerce Gateway User's Manual* for more details.

Code conversion examples:

Table 3–3

Category	Direction	Key1	Internal Value	External 1
ITEM_ CATEGORY	IN	Catalog Trading Partner code (i.e. OracleExchange) or supplier name (if catalog trading partner code does not exist)	Internal category code, for example: MISC.MISC	External category code for example: Bond Paper

Table 3–3

Category	Direction	Key1	Internal Value	External 1
UOM	IN	Catalog Trading Partner code (i.e. OracleExchange) or supplier name (if catalog trading partner code does not exist).	Internal UOM code, for example: BOX	External UOM code, for example: BX
VENDOR_ NAME	IN		Internal supplier name, for example: Staples	External supplier name, for example: Staples Corp.
VENDOR_ SITE	IN		Internal supplier site name, i.e HQ	External supplier name, i.e. Staples Corp.

A

Oracle Exchange Punchout DTDs

This Appendix contains the XML DTD specifications that govern the format and content of the XML transactions used to pass information between Exchange and your host site.

- Punchout Request (Pomlognr.dtd)
- Punchout Response (Pomlogns.dtd)
- Punchout Shopping Cart (pomcarts.dtd)
- Pomxmlrr.dtd
- Shoppingcart.dtd

You can find information on the cXML DTD specifications at:
`http://xml.cXML.org/schemas/cXML/1.1.007/cXML.dtd`

XML Punchout Request (pomlognr.dtd)

See also pomxmlrr.dtd for common DTD elements.

```
<!ENTITY % common SYSTEM "pomxmlrr.dtd">
<!ENTITY % body "<!ELEMENT body (loginInfo,searchKeywords?)>">
<!ELEMENT loginInfo (exchangeInfo,userInfo,returnURL)>
<!ELEMENT exchangeInfo (exchangeName)>
<!ELEMENT exchangeName (#PCDATA)>
<!ELEMENT userInfo (userName,userContactInfo,userCompany)>
<!ELEMENT userName (#PCDATA)>
<!ELEMENT userContactInfo (userPhone,userEmail)>
<!ELEMENT userCompany (companyName,companyDUNS,contactName?,contactPhone?)>
<!ELEMENT userPhone (#PCDATA)>
<!ELEMENT userEmail (#PCDATA)>
<!ELEMENT companyName (#PCDATA)>
<!ELEMENT companyDUNS (#PCDATA)>
```

```
<!ELEMENT contactName (#PCDATA)>
<!ELEMENT contactPhone (#PCDATA)>
<!ELEMENT returnURL (#PCDATA)>
%common;
%request;
```

XML Punchout Response (pomlogns.dtd)

See also pomxmlrr.dtd for common DTD elements.

```
<!ENTITY % common SYSTEM "pomxmlrr.dtd">
<!ENTITY % body "<!ELEMENT body (loginInfo)>">
<!ELEMENT loginInfo (loginURL)>
<!ELEMENT loginURL (#PCDATA)>
%common;
%response;
```

Shopping Cart DTD (pom carts.dtd)

See also pomxmlrr.dtd and shoppingcart.dtd for common and included elements.

```
<!ENTITY % common SYSTEM "pomxmlrr.dtd">
<!ENTITY % shoppingcart SYSTEM "shoppingcart.dtd">
%shoppingcart;
<!ENTITY % body "<!ELEMENT body (OrderLinesDataElements)>”>
%common;
%response;
```

POMXMLRR.DTD

```
<!-- $Header: pomxmlrr.dtd 115.2 2001/02/05 18:42:11 mangupta noship $ -->
<!-- common dtd file for all the request/response dtds -->
<!ENTITY % message "(header,body?)">
  <!-- both response and request are modeled similarly, with a
       header and a body -->
<!ENTITY % body   "<!ELEMENT body  ANY>">
  <!-- any request can override this body element -->
<!ENTITY % request "<!ELEMENT request %message;>">
  <!-- request dtd can instantiate the request element and the
       response dtd can instantiate the response element -->
<!ENTITY % response "<!ELEMENT response %message;>">
  <!-- see request entity -->
<!ELEMENT header
(requestID?,login?,(action|return),cookies?,language?,userArea?)>
```

```
  <!-- header contains
an optional requestID which is a sequence number to match
a request and a response in an asynchronous communication.
         an optional login info (since a response doesn't have to send back any
login info),
         action indicating what the request is about,
         optional cookies (the response can set some info that may not be
relavant to the client agent, but must be passed back in the subsequent
requests. This is modeled after the cookies in the http protocol)
         returnStatus     return status from the response
         language         the preferred language of the user
         userArea         user specific information
   -->
<!ATTLIST header version CDATA #REQUIRED>
   <!-- version of the particular dtd being used -->
<!ELEMENT requestID (#PCDATA)>
   <!-- requestID to match the response with the corresponding request in case
of asynchronous communication -->
<!ELEMENT login ((username,password)|session)>
   <!-- user name and password or the session details -->
<!ELEMENT action (#PCDATA)>
   <!-- action required from the request -->
<!ELEMENT cookies (cookie*)>
   <!-- cookies set from response. see header element for more info -->
<!ELEMENT cookie (#PCDATA)>
<!ATTLIST cookie name CDATA #REQUIRED>
<!ELEMENT language (#PCDATA)>
<!ELEMENT userArea ANY>
<!ELEMENT username (#PCDATA)>
<!ELEMENT password (#PCDATA)>
<!ENTITY % session "<!ELEMENT session (sessionID)>">
%session;
<!ELEMENT sessionID (#PCDATA)>
<!ELEMENT return (returnMessage?)>
<!ENTITY % returnMessage "<!ELEMENT returnMessage (#PCDATA)>">
   <!-- returnMessage is made an entity so that if a specific response desires
to return multiple messages or an xml document itself it can override this
entity -->
%returnMessage;
<!ATTLIST return returnCode (S|E|U|W|A) #REQUIRED>
   <!-- return code for the request. S = Success, E = Error, U = Unexpected, W =
Warning and A means require authentication. Authentication returncode is
required in case the session is expired and the user has to use
username/password info instead of the session info
    -->
```

```
%body;
   <!-- body entity has to be customized by each request/response dtd -->
```

SHOPPINGCART.DTD

```
<!ELEMENT OrderLinesDataElements (catalogTradingPartner?,orderLine*)>
<!ELEMENT catalogTradingPartner (#PCDATA)>
  <!ELEMENT orderLine
(contract?,item,category,price,supplier?,additionalAttributes?)>
    <!ELEMENT contract ((supplierContract |
buyerContract),buyerContractLineNumber?,catalogType?)>
      <!ATTLIST contract contractNumberIdentifier (KNOWN | UNKNOWN |
INFORMATIONAL | NONE) #IMPLIED>
      <!ELEMENT supplierContract (contractNumber)>
      <!ELEMENT buyerContract (contractNumber)>
      <!ELEMENT contractNumber (#PCDATA)>
      <!ELEMENT buyerContractLineNumber (#PCDATA)>
      <!ELEMENT catalogType (#PCDATA)>

    <!ELEMENT item
(itemNumber,itemDescription,unitOfMeasure,hazardClass?)>
     <!ATTLIST item lineType (GOODS | SERVICES_AMOUNT | SERVICES_QUANTITY)
"GOODS">
     <!ATTLIST item quantity CDATA "1">
      <!ELEMENT itemNumber (supplierItemNumber , manufacturerItemNumber? ,
buyerItemNumber?)>
         <!ELEMENT supplierItemNumber (itemID, supplierReferenceNumber?)>
           <!ELEMENT itemID (#PCDATA)>
           <!ELEMENT supplierReferenceNumber (#PCDATA)>
           <!ELEMENT manufacturerItemNumber (itemID,manufacturerName)>
           <!ELEMENT manufacturerName (#PCDATA)>
         <!ELEMENT buyerItemNumber (itemID,buyerItemRevision?)>
           <!ELEMENT buyerItemRevision (#PCDATA)>
           <!ELEMENT itemDescription (#PCDATA)>
           <!ELEMENT unitOfMeasure (buyerUnitOfMeasure? ,
supplierUnitOfMeasure)>
         <!ELEMENT buyerUnitOfMeasure (#PCDATA)>
         <!ELEMENT supplierUnitOfMeasure (supplierUOMType,
supplierUOMQuantity?)>
            <!ELEMENT supplierUOMType (#PCDATA)>
            <!ELEMENT supplierUOMQuantity (#PCDATA)>
      <!ELEMENT hazardClass (#PCDATA)>
    <!ELEMENT category (categoryCode)>
    <!ELEMENT categoryCode (#PCDATA)>
```

```
     <!ATTLIST categoryCode categoryCodeIdentifier (SPSC | SUPPLIER | BUYER)
#IMPLIED>
    <!ELEMENT price (currency,unitPrice)>
     <!ELEMENT currency (#PCDATA)>
     <!ELEMENT unitPrice (#PCDATA)>
   <!ELEMENT supplier ((supplierDUNS |supplierTradingPartnerCode),supplierName,
contactName?, contactPhone?)>
      <!ELEMENT supplierDUNS (#PCDATA)>
 <!ELEMENT supplierTradingPartnerCode (#PCDATA)>
      <!ELEMENT supplierName (#PCDATA | supplierSite)*>
      <!ELEMENT supplierSite (#PCDATA)>
      <!ELEMENT contactName (#PCDATA)>
      <!ELEMENT contactPhone (#PCDATA)>
    <!ELEMENT additionalAttributes (attribute1?, attribute2?,attribute3?,
attribute4?, attribute5?, attribute6?, attribute7?, attribute8?, attribute9?,
attribute10?, attribute11?, attribute12?, attribute13?, attribute14?,
attribute15?)>
      <!ELEMENT attribute1 (#PCDATA)>
      <!ELEMENT attribute2 (#PCDATA)>
      <!ELEMENT attribute3 (#PCDATA)>
      <!ELEMENT attribute4 (#PCDATA)>
      <!ELEMENT attribute5 (#PCDATA)>
      <!ELEMENT attribute6 (#PCDATA)>
      <!ELEMENT attribute7 (#PCDATA)>
      <!ELEMENT attribute8 (#PCDATA)>
      <!ELEMENT attribute9 (#PCDATA)>
      <!ELEMENT attribute10 (#PCDATA)>
      <!ELEMENT attribute11 (#PCDATA)>
      <!ELEMENT attribute12 (#PCDATA)>
      <!ELEMENT attribute13 (#PCDATA)>
      <!ELEMENT attribute14 (#PCDATA)>
      <!ELEMENT attribute15 (#PCDATA)>
```

B

Oracle iProcurement Punchout DTDs

Common DTD file for all Request / Response DTDs (pomxmlrr.dtd)

```
<!-- $Header: pomxmlrr.dtd 115.0 2001/02/02 01:16:04 mahmad noship $ -->
<!-- common dtd file for all the request/response dtds -->
<!ENTITY % message "(header,body)">
  <!-- both response and request are modeled similarly, with a header and a body
-->
<!ENTITY % body   "<!ELEMENT body ANY>">
  <!-- any request can override this body element -->
<!ENTITY % request "<!ELEMENT request %message;>">
  <!-- request dtd can instantiate the request element and the response dtd can
instantiate the response element -->
<!ENTITY % response "<!ELEMENT response %message;>">
  <!-- see request entity -->
<!ELEMENT header
(requestID?,login?,(action|return),cookies?,language?,userArea?)>
<!ATTLIST header version CDATA #REQUIRED>
   <!-- version of the particular dtd being used -->
<!ELEMENT requestID (#PCDATA)>
<!ELEMENT login ((username,password)|session)>
   <!-- user name and password or the session details -->
<!ELEMENT action (#PCDATA)>
   <!-- action required from the request -->
<!ELEMENT cookies (cookie*)>
   <!-- cookies set from response. see header element for more info -->
<!ELEMENT cookie (#PCDATA)>
<!ATTLIST cookie name CDATA #REQUIRED>
<!ELEMENT language (#PCDATA)>
<!ELEMENT userArea ANY>
<!ELEMENT username (#PCDATA)>
```

```
<!ELEMENT password (#PCDATA)>
<!ENTITY % session "<!ELEMENT session (sessionID)>">
%session;
<!ELEMENT sessionID (#PCDATA)>
  <!-- any additional user information can be passed in this element -->
<!ELEMENT return (returnMessage?)>
<!ENTITY % returnMessage "<!ELEMENT returnMessage (#PCDATA)>">
   <!-- returnMessage is made an entity so that if a specific response desires
to return multiple messages or an xml document itself it can override this
entity -->
%returnMessage;
<!ATTLIST return returnCode (S|E|U|W|A) #REQUIRED>
%body;
```

Table B–1 pomxmlrr.dtd Elelments

Field	Required	Comment
<requestID>	N	A sequence number to match a request and response in an asynchronous communication. Not currently used.
<action>	Y	For the OEXLoginRequest either shopping or suppSync. Shopping if the request is to add items to a shopping cart, suppSync if the iProcurement administrator is downloading new punchout suppliers from Oracle Exchange to iProcurement. For the LoginResponse, the value should be LOGIN_ RESPONSE. This is the response returned to iProcurement from either Oracle Exchange or the supplier site. For the OrderLinesDataElements (shopping cart) this is SHOPPING_RESPONSE. Included in the XML that returns the shopping cart to iProcurement. For the supplierSyncUpResponse this is SYNC_ RESPONSE. Included in the XML that is returned to iProcurement from Oracle Exchange during the initial setup of Exchange punchout suppliers.
<cookie>	N	Required attribute is the cookie name
<language>	Y	Language code for the site level user.
<userName>	Y	The site level user name on Oracle Exchange. Created when the proxy user is created on Oracle Exchange
<password>	Y	Password associated with the site level username above
<SessionID>	N	Oracle Exchange session ID if the session is already established.

Table B–1 pomxmlrr.dtd Elelments

Field	Required	Comment
<returnMessage>	N	Valid return codes are S (success), E (error), U (unexpected), W (warning), A (requires authentication)
<operatingUnit>	N	Part of the <userArea>, this is the iProcurement user's operating unit.
<shipTo>	N	Part of the <userArea>, this is the iProcurement user's Ship to location
<deliverTo>	N	Part of the <userArea>, this is the iProcurement user's Deliver to location

OEXLoginRequest.dtd

```
<!-- $Header: OEXLoginRequest.dtd 115.0 2001/02/03 00:57:04 skaushik noship $
-->
<!ELEMENT % common SYSTEM "pomxmlrr.dtd">
<!ELEMENT % body SYSTEM "<ELEMENT body (loginRequest)>">

<!ELEMENT loginRequest(companyName, userInfo, companyInfo?, userArea?,
partySiteId?, reqToken?, language?,
searchKeywords?, cxmlCart?, returnURL)>

<!ELEMENT companyName(#PCDATA)>
<!ELEMENT userInfo(userName, appUserName, userContactInfo?)>
<!ELEMENT userName(#PCDATA)>
<!ELEMENT appUserName(#PCDATA)>
<!ELEMENT userContactInfo(userPhone?, userEmail?)>
<!ELEMENT userPhone(#PCDATA)>
<!ELEMENT userEmail(#PCDATA)>
<!ELEMENT companyInfo(companyDUNS?, contactName?, contactPhone?)>
<!ELEMENT companyDUNS(#PCDATA)>
<!ELEMENT contactName(#PCDATA)>
<!ELEMENT contactPhone(#PCDATA)>
<!ELEMENT userArea ANY>
<!ELEMENT partySiteId (#PCDATA)>
<!ELEMENT reqToken(#PCDATA)>
<!ELEMENT language(#PCDATA)>
<!ELEMENT searchKeywords(#PCDATA)>
<!ELEMENT % cxmlCart SYSTEM "cXML.dtd">
```

```
<!ELEMENT returnURL(#PCDATA)>
```

Table B–2 OEXLoginRequest.dtd Elements

Field	Required	Comment
<companyName>	Y	Company name of iProcurement user (e.g. Big Buyer, Inc.)
<userName>	Y	Name of iProcurement user (e.g. Terry Green)
<appUserName>	Y	Oracle Application user name of iProcurement user (e.g. tgreen)
<userPhone>	N	IProcurement user's phone number
<userEmail>	N	IProcurement user's email address
<companyDUNS>	N	DUNS number of iProcurement company
<contactName	N	Contact Name for iProcurement company
<contactPhone>	N	Phone number of company contact
<partySiteId>	N	OracleExchange trading partner id. Used only if punchout is to supplier via Oracle Exchange.
<reqToken>	N	Session information sent from iProcurement to Oracle Exchange that will also be sent back to iProcurement by the Exchange
<language>	N	Language of the iProcurement user
<searchKeywords>	N	Search keywords entered during initial search on iProcurement, will be used to re-execute search on Oracle Exchange
<returnURL>	Y	URL that Oracle Exchange will send the LoginResponse XML document back to
<fullName>	N	Full name of iProcurement user as stored from iProcurement user record.
<title>	N	Title of iProcurement user
<manager>	N	Name of iProcurement user's manager, as stored in iProcurement
<managerEmail>	N	iProcurement user's manager's email address
<location>	N	Location of iProcurement user from HR information
<currency>	N	Currency of iProcurement user
<dateFormat>	N	Date format of iProcurement user

LoginResponse.dtd

```
<!-- $Header: LoginResponse.dtd 115.0 2001/02/02 01:12:45 mahmad noship $ -->
<!ENTITY % common SYSTEM "pomxmlrr.dtd">
<!ENTITY % oracle SYSTEM "OrderLinesDataElements.dtd">
<!ENTITY % body "<!ELEMENT body (loginInfo)>">
<!ELEMENT loginInfo (loginURL)>
<!ELEMENT loginURL (#PCDATA)>
%common;
%response;
%oracle;
```

Table B–3 LoginResponse.dtd Elements

Field	Required	Comment
<loginURL>	Y	URL that the user is re-directed to on Oracle Exchange for shopping/supplier sync up

OrderLinesDataElements.dtd

```
<!-- $Header: OrderLinesDataElements.dtd 115.0 2001/02/02 01:14:01 mahmad noship
$ -->
<!-- OrderLinesDataElements.dtd -->
<!ELEMENT OrderLinesDataElements (catalogTradingPartner?, orderLine*)>
<!ELEMENT catalogTradingPartner (#PCDATA)>
<!ELEMENT orderLine (contract?, item, category, price, supplier?,
additionalAttributes?)>

<!-- DTD for element CONTRACT -->
<!ELEMENT contract
((supplierContract|buyerContract),buyerContractLineNumber?,catalogType?)>
  <!ATTLIST contract contractNumberIdentifier (KNOWN|UNKNOWN|INFORMATIONAL|NONE)
 #IMPLIED>
  <!ELEMENT supplierContract (contractNumber)>
  <!ELEMENT buyerContract (contractNumber)>
  <!ELEMENT contractNumber (#PCDATA)>
  <!ELEMENT buyerContractLineNumber (#PCDATA)>
  <!ELEMENT catalogType (#PCDATA)>

<!-- DTD for element ITEM -->
<!ELEMENT item (itemNumber, itemDescription, unitOfMeasure, hazardClass?)>
  <!ATTLIST item lineType (GOODS|SERVICES_AMOUNT|SERVICES_QUANTITY) #IMPLIED>
  <!ATTLIST item quantity CDATA "1">
```

```
  <!ELEMENT itemNumber
(supplierItemNumber|manufacturerItemNumber|buyerItemNumber)>
  <!ELEMENT supplierItemNumber (itemID, supplierReferenceNumber?)>
  <!ELEMENT itemID (#PCDATA)>
  <!ELEMENT supplierReferenceNumber (#PCDATA)>
  <!ELEMENT manufacturerItemNumber (itemID, manufacturerName)>
  <!ELEMENT manufacturerName (#PCDATA)>
  <!ELEMENT buyerItemNumber (itemID, buyerItemRevision?)>
  <!ELEMENT buyerItemRevision (#PCDATA)>

  <!ELEMENT itemDescription (#PCDATA)>

  <!ELEMENT unitOfMeasure (buyerUnitOfMeasure|supplierUnitOfMeasure)>
  <!ELEMENT buyerUnitOfMeasure (#PCDATA)>
  <!ELEMENT supplierUnitOfMeasure (supplierUOMType,supplierUOMQuantity?)>
  <!ELEMENT supplierUOMType (#PCDATA)>
  <!ELEMENT supplierUOMQuantity (#PCDATA)>

  <!ELEMENT hazardClass (#PCDATA)>

<!-- DTD for element CATEGORY -->
<!ELEMENT category (categoryCode)>
<!ELEMENT categoryCode (#PCDATA)>
<!ATTLIST categoryCode categoryCodeIdentifier(SPSC|SUPPLIER|BUYER) #IMPLIED>

<!-- DTD for element PRICE -->
<!ELEMENT price (currency?, unitPrice)>
<!ELEMENT currency (#PCDATA)>
<!ELEMENT unitPrice (#PCDATA)>

<!-- DTD for element SUPPLIER -->
<!ELEMENT supplier ((supplierDUNS | supplierTradingPartnerCode), supplierName,
contactName?, contactPhone?)>
<!ELEMENT supplierDUNS (#PCDATA)>
<!ELEMENT supplierTradingPartnerCode (#PCDATA)>
<!ELEMENT supplierName (#PCDATA | supplierSite)*>
<!ELEMENT supplierSite (#PCDATA)>
<!ELEMENT contactName (#PCDATA)>
<!ELEMENT contactPhone (#PCDATA)>


<!-- DTD for element ADDITIONAL ATTRIBUTES -->
<!ELEMENT additionalAttributes (attribute1?, attribute2?, attribute3?,
attribute4?, attribute5?, attribute6?, attribute7?,
attribute8?, attribute9?, attribute10?, attribute11?, attribute12?,
```

```
attribute13?, attribute14?, attribute15?)>
<!ELEMENT attribute1 (#PCDATA)>
<!ELEMENT attribute2 (#PCDATA)>
<!ELEMENT attribute3 (#PCDATA)>
<!ELEMENT attribute4 (#PCDATA)>
<!ELEMENT attribute5 (#PCDATA)>
<!ELEMENT attribute6 (#PCDATA)>
<!ELEMENT attribute7 (#PCDATA)>
<!ELEMENT attribute8 (#PCDATA)>
<!ELEMENT attribute9 (#PCDATA)>
<!ELEMENT attribute10 (#PCDATA)>
<!ELEMENT attribute11 (#PCDATA)>
<!ELEMENT attribute12 (#PCDATA)>
<!ELEMENT attribute13 (#PCDATA)>
<!ELEMENT attribute14 (#PCDATA)>
<!ELEMENT attribute15 (#PCDATA)>
```

Table B–4 OrderLinesDataElements.dtd Elements

Field	Required	Comment
<catalogTradingPartner>	N	The catalog trading partner is used for catalog hosting service websites (such as TPN) or another Exchange. It is used to determine which items category mapping should be used within Oracle iProcurement when bringing items into Oracle iProcurement. This element should not normally be populated for a punchout to a supplier site.
<contract>	N	Identifies the beginning of the contract section, contains the attribute 'contractNumberIdentifier' with possible values: KNOWN, UNKNOWN, INFORMATIONAL, or NONE
<contractNumber>	N	Supplier or buyer contract number, depending on the parent tag
<buyerContractLineNumber>	N	Line number of buyer's contract if it exists
<catalogType>	N	Specifies whether or not this is from a contract catalog or non-contract catalog
<item lineType = "GOODS" quantity = "1.0">	Y	Identifies the beginning of the item section; contains the following attributes: **LineType -** Possible values: GOODS, SERVICES_ AMOUNT, SERVICES_QUANTITY with GOODS as default with nothing is specified. **Quantity** - The number of items ordered.
<itemID>	Y	Supplier, buyer, or manufacturer item number, depending on parent tag.

Table B–4 OrderLinesDataElements.dtd Elements

Field	Required	Comment
<supplierReferenceNumber>	N	Reference number provided by the supplier for this item or order, is stored by iProcurement in the supplier reference number field.
<manufacturerName>	N	The name of the manufacturer for this item, will populate the Manufacturer Name in iProcurement
<buyerItemRevision>	N	If provided by the supplier, populates the item revision field in iProcurement.
<itemDescription>	Y	The supplier's description of this product, it will populate the item description in the iProcurement shopping cart.
<buyerUnitOfMeasure>	Y	Identifies the buyer's unit of measure
<supplierUOMType>	Y	Specifies the supplier's unit of measure type (i.e. each, dozen, bushel, etc.)
<supplierUOMQuantity>	N	Specifies the suppliers quantity in the specified unit of measure
<hazardClass>	N	Hazard class name for the item, if provided by the supplier.
<category categoryCodeIdentifier="SPSC">	Y	Specifies the category used to classify this item. Valid attributes are: SPSC, SUPPLIER, or BUYER. If code is not specified SPSC is assumed.
<categoryCode>	Y	Category code value.
<currency>	N	The currency the price is specified in.
<unitPrice>	Y	The price per unit of the item
<supplierDUNS> <supplierTradingPartnerCode>	Y	The supplier's DUNS number. Used to get supplier name and supplier site information in iProcurement. Either <supplierDUNS> or <supplierTradingPartnerCode> must be provided in the DTD. <supplierTradingPartnerCode> is used if the supplier does not have a DUNS number, and this element should be populated with a unique alphanumeric code which the buyer will use to identify the internal supplier code in their ERP system.
<supplierName>	Y	The supplier's company name. This is validated against vendors defined in iProcurement.
<supplierSite>	N	Supplier's site as defined in iProcurement
<supplierNumber>	N	Supplier number as defined in iProcurement

Table B–4 OrderLinesDataElements.dtd Elements

Field	Required	Comment
<supplierTradingPartner>	N	Not currently used.
<contactName>	N	Contact name for supplier
<contactPhone>	N	Supplier contact's phone number
<attribute1>…<attribute15>	N	Up to 15 additional attributes can be included with each item in the shopping cart

LinkRequest.dtd

See the '*Defining an Oracle Exchange Punchout*' chapter for details on this XML document

LinkResponse.dtd

See the '*Defining an Oracle Exchange Punchout*' chapter for details on this XML document

SupplierSyncUpResponse.dtd

```
<!-- $Header: SupplierSyncUpResponse.dtd 115.0 2001/02/02 01:15:09 mahmad noship
$ -->

<!ENTITY % common SYSTEM "pomxmlrr.dtd">
<!ENTITY % body "<!ELEMENT body (supplier*)>">
<!ELEMENT supplier (supplierPartyId, supplierImageUrl,
supplierLanguageSpecificInfo*)>
<!ELEMENT supplierLanguageSpecificInfo
(language,supplierName,supplierDescription,supplierKeywords)>
<!ELEMENT supplierName (#PCDATA)>
<!ELEMENT supplierDescription (#PCDATA)>
<!ELEMENT supplierKeywords (#PCDATA)>
<!ELEMENT supplierPartyId (#PCDATA)>
<!ELEMENT supplierImageUrl (#PCDATA)>
%common;
```

```
%response;
```

Table B–5 SupplierSyncUpResponse.dtd Elements

Field	Required	Comment
<language>	Y	Language of the information being retrieved
<supplierName>	Y	The supplier's company name in the specified language (the Oracle Exchange registered name of the company)
<supplierDescription>	Y	The supplier's description in the specified language (entered by the supplier on Oracle Exchange)
<supplierKeywords>	Y	Keywords provided by the supplier on Oracle Exchange
<supplierPartyId>	Y	The supplier's party ID on Oracle Exchange
<supplierImageUrl>	Y	The URL pointing to the supplier's logo.

SupplierLoginRequest.dtd

```
<!-- $Header: SupplierLoginRequest.dtd 115.0 2001/02/03 00:57:54 skau noship $
-->
<!ENTITY % common SYSTEM "pomxmlrr.dtd">
<!ENTITY % body "<!ELEMENT body (loginInfo,searchKeywords?)>">
<!ELEMENT loginInfo (exchangeInfo,userInfo,returnURL)>
<!ELEMENT exchangeInfo (exchangeName)>
<!ELEMENT exchangeName (#PCDATA)>
<!ELEMENT userInfo (userName,userContactInfo,userCompany)>
<!ELEMENT userName (#PCDATA)>
<!ELEMENT userContactInfo (userPhone,userEmail)>
<!ELEMENT userCompany (companyName,companyDUNS,contactName?,contactPhone?)>
<!ELEMENT userPhone (#PCDATA)>
<!ELEMENT userEmail (#PCDATA)>
<!ELEMENT companyName (#PCDATA)>
<!ELEMENT companyDUNS (#PCDATA)>
<!ELEMENT contactName (#PCDATA)>
<!ELEMENT contactPhone (#PCDATA)>
<!ELEMENT returnURL (#PCDATA)>
%common;
%request;
```

Table B–6 SupplierLoginRequest.dtd Elements

Field	Required	Comment
<exchangeName>	Y	The iProcurement company name that is trying to login into the supplier site
<userName>	Y	The name of the iProcurement user (e.g. Terry Green)
<userPhone>	N	The iProcurement user's phone number
<userEmail>	N	The iProcurement user's email address
<companyName>	N	The iProcurement user's company's name
<companyDUNS>	N	The company's DUNS number
<contactName>	N	Name of IProcurement company's contact
<contactPhone>	N	The company's contact's phone number
<returnUrl>	Y	The URL where iProcurement expects to receive the login response document

PunchOutSetupRequest.dtd

http://xml.cXML.org/schemas/cXML/1.1.007/cXML.dtd

PunchOutSetupResponse.dtd

http://xml.cXML.org/schemas/cXML/1.1.007/cXML.dtd

PunchOutOrderMessage.dtd

http://xml.cXML.org/schemas/cXML/1.1.007/cXML.dtd

C

Sample Oracle iProcurement XML Documents

OEXLoginRequest

This document is also used as the Supplier Sync up Login, the only difference being that the action is `suppSync` instead of `shopping`.

Sample XML

```
<?xml version = '1.0'?>
<request>
   <header version="1.0">
      <login>
         <username>
            buyer1
         </username>
         <password>
            WELCOME
         </password>
      </login>
      <action>
         shopping
      </action>
      <language>
         US
      </language>
      <userArea>
         <operatingUnit>
            204
         </operatingUnit>
         <shipTo>
            V1- New York City
```

```
            </shipTo>
            <deliverTo>
               V1- New York City
            </deliverTo>
         </userArea>
      </header>
      <body>
         <loginRequest>
            <userInfo>
               <userName>
                  Green, Mr. Terry
               </userName>
               <appUserName>
                  TGREEN
               </appUserName>
               <userContactInfo>
                  <userEmail>
                     tgreen@vision.com
                  </userEmail>
               </userContactInfo>
            </userInfo>
            <companyInfo>
               <companyName>
                  IP
               </companyName>
            </companyInfo>
            <reqToken>


template=tpn*@_action=addToOrder*@_function=addToOrder*@_reqSessionId=0*@_
reqHeaderId=0*@_emergencyReq=N*@_itemSourceId=1001

            </reqToken>
            <returnURL>

http://ap100jvm.us.oracle.com:4632/oa_
servlets/oracle.apps.icx.por.punchout.PunchOutServlet?callBack=shopping

            </returnURL>
         </loginRequest>
      </body>
   </request>
```

OEXLoginRequest (with optional extended data)

Sample XML

```
<?xml version = '1.0'?>
<request>
   <header version="1.0">
      <login>
         <username>
            buyer1
         </username>
         <password>
            WELCOME
         </password>
      </login>
      <action>
         shopping
      </action>
      <language>
         US
      </language>
      <userArea>
         <operatingUnit>
            204
         </operatingUnit>
         <shipTo>
            V1- New York City
         </shipTo>
         <deliverTo>
            V1- New York City
         </deliverTo>
         <fullName>
            Green, Mr. Terry
         </fullName>
         <title>
            MR.
         </title>
         <manager>
            Baker, Ms. Catherine
         </manager>
         <managerEmail>
            cbaker@vision.com
         </managerEmail>
         <location>
```

```
            V1- New York City
         </location>
         <language>
            US
         </language>
         <currency>
            USD
         </currency>
         <dateFormat>
            DD-MON-RRRR
         </dateFormat>
      </userArea>
   </header>
   <body>
      <loginRequest>
         <userInfo>
            <userName>
               Green, Mr. Terry
            </userName>
            <appUserName>
               TGREEN
            </appUserName>
            <userContactInfo>
               <userEmail>
                  tgreen@vision.com
               </userEmail>
            </userContactInfo>
         </userInfo>
         <companyInfo>
            <companyName>
               IP
            </companyName>
         </companyInfo>
         <reqToken>


template=tpn*@_action=addToOrder*@_function=addToOrder*@_reqSessionId=0*@_
reqHeaderId=0*@_emergencyReq=N*@_itemSourceId=1001

         </reqToken>
         <returnURL>

http://ap100jvm.us.oracle.com:4632/oa_
servlets/oracle.apps.icx.por.punchout.PunchOutServlet?callBack=shopping
```

```
        </returnURL>
      </loginRequest>
   </body>
</request>
```

LoginResponse

Sample XML

```
<?xml version = '1.0'?>
<response>
  <header version="1.0">
    <action> LOGIN_RESPONSE </action>
  </header>
  <body>
    <loginInfo>
      <loginURL>

http://ap904sun.us.oracle.com:6250/orders/LinkinCallback.jsp?sessionId=3fe0a73f6
fa1c003.164.982359182161&action=shopping


     </loginURL>
    </loginInfo>
  </body>
</response>
```

OrderLinesDataElements

Sample XML

```
<?xml version = '1.0' encoding = 'ISO-8859-1'?>
<response>
  <header>
    <action>
      SHOPPING_RESPONSE
    </action>
  </header>
```

```
<body>
  <OrderLinesDataElements>
    <catalogTradingPartner>
      OracleExchange
    </catalogTradingPartner>
    <orderLine>
      <contract>
        <catalogType>
          CONTRACTED
        </catalogType>
      </contract>
      <item lineType="GOODS" quantity="1.0">
        <itemNumber>
          <supplierItemNumber>
            <itemID>
              kb_Item2
            </itemID>
          </supplierItemNumber>
        </itemNumber>
        <itemDescription>
          Item #2 : kbenjami- list, TBD b.s.
        </itemDescription>
        <unitOfMeasure>
          <buyerUnitOfMeasure/>
          <supplierUnitOfMeasure>
            <supplierUOMType>
              Bushel
            </supplierUOMType>
```

```
            <supplierUOMQuantity>

            </supplierUOMQuantity>
          </supplierUnitOfMeasure>
        </unitOfMeasure>
      </item>
      <category>
        <categoryCode categoryCodeIdentifier="SPSC">
          Ball Point Pens
        </categoryCode>
      </category>
      <price>
        <currency>
          USD
        </currency>
        <unitPrice>
          100.0
        </unitPrice>
      </price>
      <supplier>
        <supplierDUNS>
          2820
        </supplierDUNS>
        <supplierName>
          Kareem's Corporation
        </supplierName>
      </supplier>
    </orderLine>
```

```
      </OrderLinesDataElements>
    </body>
  </response>
```

LinkRequest

See PunchoutRequest DTD in Appendix A for details on this XML document.

LinkResponse

See Punchout Response DTD' in Appendix A for details on this XML document.

SupplierSyncUpResponse

Sample XML

```
<?xml version = '1.0'?>
<response>
   <header version="1.0">
      <action>
         SYNC_RESPONSE
      </action>
   </header>
   <body>
      <supplier>
         <supplierPartyId>
            2817
         </supplierPartyId>
         <supplierImageUrl>
            http://www.yahoo.com
         </supplierImageUrl>
         <supplierLanguageSpecificInfo>
            <language>
               F
            </language>
            <supplierName>
               TestCo
            </supplierName>
            <supplierDescription>
               Yahoo Life Nothing Pen
            </supplierDescription>
```

```
            <supplierKeywords/>
         </supplierLanguageSpecificInfo>
         <supplierLanguageSpecificInfo>
            <language>
               PT
            </language>
            <supplierName>
               TestCo
            </supplierName>
            <supplierDescription>
               Yahoo Life Nothing Pen
            </supplierDescription>
            <supplierKeywords/>
         </supplierLanguageSpecificInfo>
         <supplierLanguageSpecificInfo>
            <language>
               US
            </language>
            <supplierName>
               TestCo
            </supplierName>
            <supplierDescription>
               Yahoo Life Nothing Pen
            </supplierDescription>
            <supplierKeywords>
               Monkey Donkey
            </supplierKeywords>
         </supplierLanguageSpecificInfo>
      </supplier>
   </body>
</response>
```

SupplierLoginRequest

Sample XML

```
<?xml version = '1.0'?>
<request>
   <header version="1.0">
      <login>
         <password>
            welcome
         </password>
      </login>
```

```
   <action>
      SHOPPING
   </action>
   <language>
      US
   </language>
   <userArea>
      <operatingUnit>
         204
      </operatingUnit>
      <shipTo>
         V1- New York City
      </shipTo>
      <deliverTo>
         V1- New York City
      </deliverTo>
   </userArea>
</header>
<body>
   <loginInfo>
      <exchangeInfo>
         <exchangeName>
            Oracle
         </exchangeName>
      </exchangeInfo>
      <userInfo>
         <userContactInfo>
            <userPhone/>
            <userEmail>
               tgreen@vision.com
            </userEmail>
         </userContactInfo>
         <userCompany>
            <companyName>
               Oracle
            </companyName>
            <companyDUNS/>
            <contactName/>
            <contactPhone/>
         </userCompany>
         <userName>
            Green, Mr. Terry
         </userName>
      </userInfo>
      <returnURL>
```

```
            http://ap100jvm.us.oracle.com:4634/oa_
servlets/oracle.apps.icx.por.punchout.PunchOutServlet?callBack=shopping
         </returnURL>
      </loginInfo>
   </body>
</request>
```

PunchOutSetupRequest (cXML)

Sample XML

```
<?xml version = '1.0' encoding = 'ISO-8859-1'?>
<!DOCTYPE cXML SYSTEM " http://xml.cxml.org/schemas/cXML/1.1.007/cXML.dtd
<http://xml.cxml.org/schemas/cXML/1.1.007/cXML.dtd> ">
<cXML version="1.1.007" xml:lang="en-US" payloadID="Thu Apr 19 10:29:03
GMT-08:00 2001" timestamp="Thu Apr 19 10:29:03 GMT-08:00 2001">
<Header>
<From>
<Credential domain="Name">
<Identity>OEXCHANGE</Identity>
</Credential>
</From>
<To>
<Credential domain="DUNS">
<Identity>015996366</Identity>
</Credential>
</To>
<Sender>
<Credential domain="ap903sun.us.oracle.com">
<Identity>6.2 Dev env</Identity>
<SharedSecret>snow</SharedSecret>
</Credential>
<UserAgent>6.2 Dev env</UserAgent>
</Sender>
</Header>
<Request>
<PunchOutSetupRequest operation="create">
<BuyerCookie>12345678</BuyerCookie>
<Extrinsic name="User">KBENJAMI</Extrinsic>
<BrowserFormPost>
<URL>http://ap903sun.us.oracle.com:6290/orders/PunchoutCallback.jsp?app=buying</
URL>
</BrowserFormPost>
<Contact>
```

```
<Name xml:lang="en-US">KareemAndreBenjami</Name>
<Email>dumas4@yahoo.com</Email>
</Contact>
</PunchOutSetupRequest>
</Request>
</cXML>
```

PunchOutSetupResponse (cXML)

Sample XML

```
<?xml version="1.0" ?>
<cXML payloadID="2000-08-15" timestamp="2000-08-15">
  <Response>
    <PunchOutSetupResponse>
      <StartPage>
        <URL>
          http://www.oracle.com/orders/
             SupplierShopping.jsp?sessionID=123
        </URL>
      </StartPage>
    </PunchOutSetupResponse>
    <Status code="200" text="success" />
  </Response>
</cXML>
```

PunchOutOrderMessage (cXML)

Sample XML

```
<?xml version = "1.0" encoding = "UTF-8"?>
<cXML version = "1.1.007" xml:lang = "en-US" payloadID = "99999" timestamp =
"2000-08-15">
  <Header>
    <From>
      <Credential domain = "DUNS">
        <Identity>888</Identity>
      </Credential>
    </From>
    <To>
      <Credential domain = "DUNS">
```

```
        <Identity>888</Identity>
      </Credential>
    </To>
    <Sender>
      <Credential domain = "DUNS">
        <Identity>888</Identity>
      </Credential>
      <UserAgent>Workchairs cXML</UserAgent>
    </Sender>
  </Header>
  <Message>
    <PunchOutOrderMessage>
      <BuyerCookie>ICX</BuyerCookie>
      <PunchOutOrderMessageHeader operationAllowed = "edit">
        <Total>
          <Money currency = "USD">112.4</Money>
        </Total>
      </PunchOutOrderMessageHeader>
      <ItemIn quantity = "3">
        <ItemID>
          <SupplierPartID>5555</SupplierPartID>
        </ItemID>
        <ItemDetail>
          <UnitPrice>
            <Money currency = "USD">763.20</Money>
          </UnitPrice>
          <Description xml:lang = "en">
             Leather reclining chair
          </Description>
          <UnitOfMeasure>EA</UnitOfMeasure>
          <Classification domain = "UNSPSC">
            513333
          </Classification>
        </ItemDetail>
      </ItemIn>
    </PunchOutOrderMessage>
  </Message>
</cXML>
```

Index

P

R

U

X