

Open Catalog Interface



ADDON.EBP_OCI

Release 3.0



Copyright

© Copyright 2001 SAP AG. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

Microsoft[®], WINDOWS[®], NT[®], EXCEL[®], Word[®], PowerPoint[®] and SQL Server[®] are registered trademarks of Microsoft Corporation.

IBM[®], DB2[®], OS/2[®], DB2/6000[®], Parallel Sysplex[®], MVS/ESA[®], RS/6000[®], AIX[®], S/390[®], AS/400[®], OS/390[®], and OS/400[®] are registered trademarks of IBM Corporation.

ORACLE[®] is a registered trademark of ORACLE Corporation.

INFORMIX[®]-OnLine for SAP and Informix[®] Dynamic Server[™] are registered trademarks of Informix Software Incorporated.

UNIX[®], X/Open[®], OSF/1[®], and Motif[®] are registered trademarks of the Open Group.






HTML, DHTML, XML, XHTML are trademarks or registered trademarks of W3C[®], World Wide Web Consortium, Massachusetts Institute of Technology.

JAVA[®] is a registered trademark of Sun Microsystems, Inc.

JAVASCRIPT[®] is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

SAP, SAP Logo, R/2, RIVA, R/3, ABAP, SAP ArchiveLink, SAP Business Workflow, WebFlow, SAP EarlyWatch, BAPI, SAPPHIRE, Management Cockpit, mySAP.com Logo and mySAP.com are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other products mentioned are trademarks or registered trademarks of their respective companies.

Icons

Icon	Meaning
	Caution
	Example
	Note
	Recommendation
	Syntax

Open Catalog Interface	5
Open Catalog Interface: Outbound Section	6
Open Catalog Interface: Inbound Section	9
Example of a Catalog Interface	16

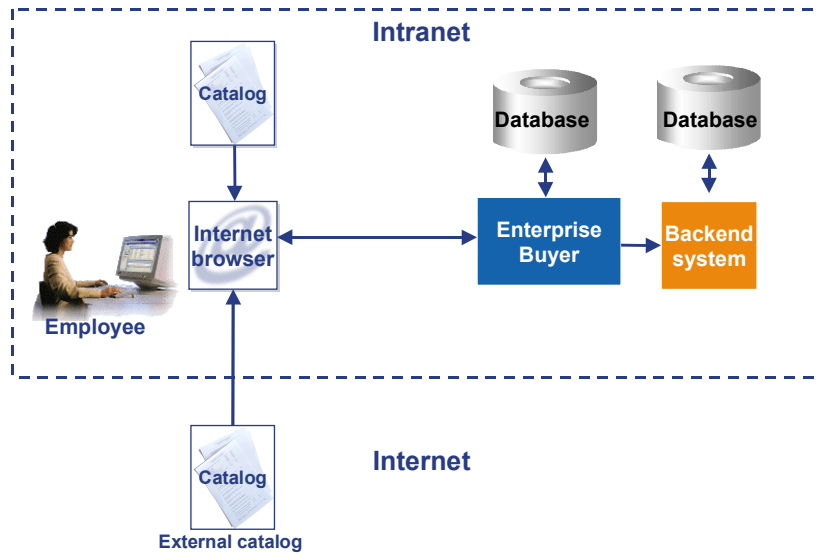
Open Catalog Interface

Purpose

The Open Catalog Interface (OCI) is the interface between catalogs and SAP Markets Enterprise Buyer (professional edition). SAP's Open Catalog Interface uses standard Internet protocols. It has already been implemented between several catalog applications and Enterprise Buyer.

Integration

The graphic below shows how catalogs are integrated with Enterprise Buyer.



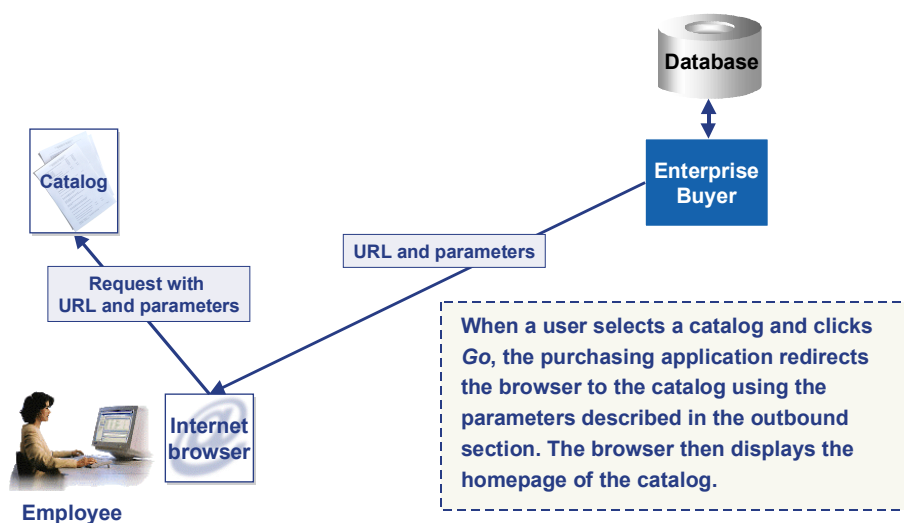
Structure

The catalog interface consists of two separate and distinct sections: the **outbound section** and the **inbound section**.

- The [outbound section \[Page 6\]](#) defines the information being sent from the Enterprise Buyer application to the catalog application. This includes information such as the catalog URL and logon data.
- The [inbound section \[Page 9\]](#) consists of the information being sent from the catalog application to the Enterprise Buyer application. This section contains data on the items selected in the catalog, such as the item descriptions, quantities ordered, and prices. For more details, see the interface diagram under *Inbound Section*.

Open Catalog Interface: Outbound Section

The following graphic shows how the purchasing application redirects the browser from Enterprise Buyer to the catalog using the parameters of the outbound section:



The outbound section consists of information that is sent to the catalog application by the Enterprise Buyer application. This information originates in the Enterprise Buyer system, where it is created and maintained using Customizing activity *Define catalogs*. You also have to define user attribute CAT to make the catalog visible for the user in the Enterprise Buyer application.

Previously catalogs were accessed using HTTP method HTTP GET. Now the standard catalog call method is HTTP POST. However, you can switch to HTTP method HTTP GET by selecting the flag *Call catalog using GET* in Customizing activity *Define catalogs*.

Using this information, the Enterprise Buyer application constructs a URL call to the catalog application and redirects the browser to this URL. The CGI script of the catalog application then has to parse and decode this information.

In the Enterprise Buyer system, you can set up this information in Customizing.

Path in the SAP Implementation Guide (IMG):

Enterprise Buyer professional edition → *Master Data* → *Define Catalogs*.

Note the following when setting up this information:

- Many of the field names and values are dependent on the particular catalog application.
- Every field contains a name and a value (*Contents* column) and has a type. Possible types are:
 - URL
 - The URL of the catalog or link to the information you want to access.
 - SAP field
 - A system field, such as SY-LANGU, that is filled with the correct value, in this case the language, at runtime.
 - Fixed value
 - The value this field contains is transferred.
 - Return URL
 - The URL used to return to the Enterprise Buyer application from the catalog.
- You must structure the fields in the order shown in the table below. In particular, define the catalog-specific information directly after the catalog URL and the ~OkCode and ~Target, and ~Caller fields directly after the Return URL field.

- Typically, field values are entered as literals. For example, the LOGIN field can have a literal value of GUEST. However, system variables such as SY-UNAME can also be used as values for a field, allowing the actual value to be determined at runtime. In this case, you should assign the type *SAP field* to the relevant field.
- The name/value pair OCI_VERSION="<Release>" (for example, OCI_VERSION="3.0") is passed to the catalog. This indicates that the Enterprise Buyer system involved is a Release 3.0 system.

The outbound data you maintain is structured as described in the table below. In this table, *Fixed* indicates that the field name must be exactly as specified, whereas *Variable* indicates that the field name is catalog-specific.

Description	Mandatory	Catalog-specific	Field Name	Field Name is Fixed/Variable	Meaning
Catalog URL	Yes	Yes	<blank>	Fixed	The URL of the catalog. This should refer to the location of the catalog CGI script file.
All catalog-specific fields	As relevant	Yes		Variable	The set of catalog-specific fields. An example of catalog-specific fields is provided in the table below.
Return URL	Yes	No	HOOK_URL	Variable	The URL used to return to the Enterprise Buyer application from the catalog application. Set the value of this field to blank. It is automatically filled at runtime by the procurement application. (See also Note below.)
OK Code	Yes	No	~OkCode	Fixed	Contains the transaction code indicating that the function <i>Add Items to SAP shopping basket</i> is to be performed. Must be set to ADDI for Enterprise Buyer.
Target	Yes	No	~TARGET	Fixed	Specifies the frame to which a catalog is to return in a frame-based environment. If this field is not set, the catalog application must provide a default target of <i>_top</i> .
Caller	Yes	No	~CALLER	Fixed	Indicates that the data was sent by an external catalog. Content must be set to CTLG .



The HOOK_URL can have a different name, but the type of this field must be set to Return URL. The HOOK_URL is encoded so that special characters such as ':' and '/' are represented by '%' and the hex digits for the ASCII code for the characters. The encoding method is standard URL encoding. The catalog application is responsible for decoding this string into a valid URL. Note that the catalog application must not make any assumptions about the HOOK_URL. For example, it cannot assume that a question mark is present to delineate the URL variables.



Example of catalog-specific parameters:

Description	Field Name	Field Name is Fixed/Variable	Meaning
Request type	REQTYPE	Variable	Requests a login when the catalog is first accessed.
Login	USERNAME	Variable	The user ID for logging in to the catalog.
Password	PASSWORD	Variable	The login password.
Database identifier	SERVICE	Variable	The database identifier. This may be required for some catalogs.

Additional Catalog Functions

An external catalog may support additional functionality. This additional functionality can be triggered by several name/value pairs which are passed over to the catalog within the query string of the catalog URL or form when the functionality is needed. The following functions are possible:

- Detail of a product

If this field is filled, the application jumps directly to the correct point in the catalog.

The following name/value pairs are additionally transferred to the catalog:

FUNCTION="DETAIL" and

PRODUCTID="database key of product in the catalog"

- Validation of a product

When a template containing catalog data is used again, the catalog is accessed and the current data (such as the current price) is returned to Enterprise Buyer.

The following name/value pairs are additionally transferred to the catalog:

FUNCTION="VALIDATE" and

PRODUCTID="database key of product in the catalog"

- Sourcing of a product

If you enter a search term, such as pen in Enterprise Buyer, a search is made directly in the catalog from the application and matching items found in the catalog are returned to Enterprise Buyer.

The following name/value pairs are additionally transferred to the catalog:

FUNCTION="VALIDATE" and

SEARCHSTRING="string to directly start the catalog search" and

VENDOR="vendornumber in the buyer's system"



The functions "DETAIL" and "VALIDATE" will only work if the parameter NEW_ITEM-EXT_PROD_ID[n] was filled from the catalog in a previous call.

If a catalog supports one or more of these three additional functions, the corresponding flag has to be set in Customizing. Path in the SAP Implementation Guide (IMG): *Enterprise Buyer professional edition* → *Master Data* → *Define Catalogs*.

The expected data to be returned from the catalog is described in this document under [Inbound Section](#)

[Page 9]

Open Catalog Interface: Inbound Section

Use

The inbound section consists of information that is sent to the Enterprise Buyer application by the catalog application.

For each item selected in the catalog and sent to the Enterprise Buyer application, all mandatory fields have to be sent, along with the relevant optional fields.

The fields can be sent back to the purchasing application by either the GET or the POST method (because to the limitations of GET, POST is strongly recommended).

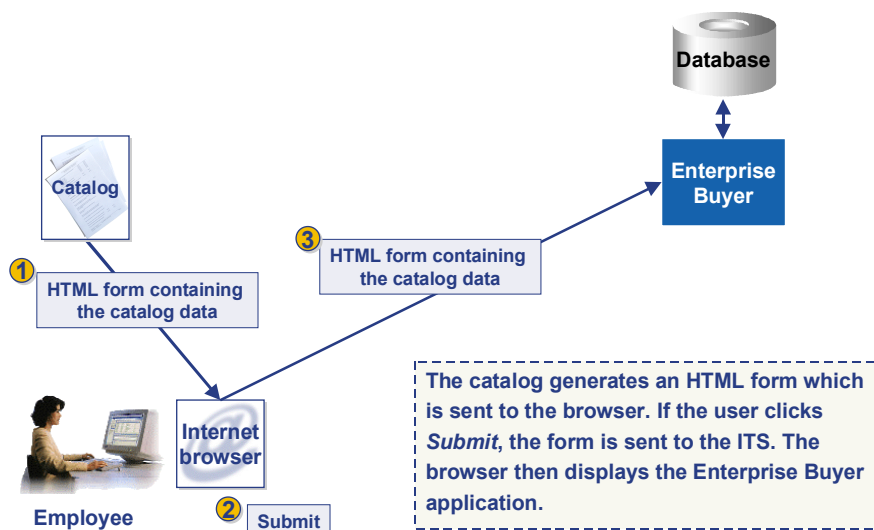


Note the following:

- Fields should not contain leading spaces.
- All numeric fields that allow fractional amounts must explicitly include the decimal point "." at the correct byte position.
- Apart from the decimal point, numeric fields may **not** include commas or any other non-numeric characters.
- As of Release 2.0, no distinction is made between a service master and a material master. Both will be stored as a product master.
- As of Release 2.0 the inbound section can be sent back to the Enterprise Buyer system as follows:
 - As an HTML version (as in Release 1.0)
 - As an XML version

HTML version

The following graphic illustrates the data flow of the inbound section:



To handle multiple items selected in the catalog, each field name must be an array using the bracket subscript convention as per the C programming language. Subscripts must start at 1 (not zero). For example, the description of an item would be returned as a name such as "NEW_ITEM-DESCRIPTION [1]" (refer to the [Example of a Catalog Interface \[Page 16\]](#)).

Item	Name	Required/Optional	Details	Type Length
Description	NEW_ITEM-DESCRIPTION[n]	Required if the SAP product master number is not specified.	The description of the item to be ordered	CHAR - 40
Product master	NEW_ITEM-MATNR[n]	Required if the <i>Description</i> field is not specified.	The SAP product master number in the buyer system	CHAR - 18
Product group	NEW_ITEM-MATGROUP[n]	Optional	The SAP product (material) group.	CHAR - 10
Quantity	NEW_ITEM-QUANTITY[n]	Required	Quantity to add. There can be a maximum of 11 digits to the left of the decimal point, and there must be 3 digits to the right. If the quantity contains decimal places, there must be a decimal point, not a comma. Otherwise problems will occur.	CHAR - 15
Unit of measure	NEW_ITEM-UNIT[n]	Required if product master field is not filled	Unit of measure of the item. Must be the standard ISO code.	CHAR - 3
Price	NEW_ITEM-PRICE[n]	Optional	Price of the item in the catalog. There can be a maximum of 11 digits to the left of the decimal point, and there must be 3 digits to the right.	CHAR - 15
Price unit	NEW_ITEM-PRICEUNIT[n]	Optional	The number of units to be purchased at the given price. If no value is returned by the catalog, the value defaults to one.	CHAR - 9
Currency	NEW_ITEM-CURRENCY[n]	Required if a price is returned, otherwise optional	Must be the ISO code for the currency.	CHAR - 5
Lead time	NEW_ITEM-LEADTIME[n]	Optional	Number of days from today until the product will be available. If not specified, no assumptions will be made about the lead time.	CHAR - 5

Item	Name	Required/Optional	Details	Type Length
Vendor	NEW_ITEM-VENDOR[n]	Optional	If the product is purchased through a multi-supplier catalog, this field contains the business partner number in the buyer system.	CHAR - 10
Vendor product number	NEW_ITEM-VENDORMAT[n]	Optional	The vendor product part number	CHAR - 40
Manufacturer's code	NEW_ITEM-MANUFACTCODE[n]	Optional	The manufacturer's code in the buyer system	CHAR - 10
Manufacturer's product number	NEW_ITEM-MANUFACTMAT[n]	Optional	The manufacturer's product part number	CHAR - 40
Contract number	NEW_ITEM-CONTRACT[n]	Optional	The number of the contract with the vendor.	CHAR - 10
Item of a contract	NEW_ITEM-CONTRACT_ITEM[n]	Optional	The number of an item within a contract with the vendor.	CHAR - 5
Service flag	NEW_ITEM-SERVICE[n]	Optional	A flag which indicates if the line refers to a service or to goods.	CHAR - 1
Quotation	NEW_ITEM-EXT_QUOTE_ID[n]	Optional	A reference to an external quotation ID. Example: The catalog is able to create a quotation in the selling system. This is a reference to this quotation.	CHAR - 35
Quotation item	NEW_ITEM-EXT_QUOTE_ITEM[n]	Optional	A reference to an external quotation item. Example: The catalog is able to create a quotation in the selling system. This is a reference to this quotation.	CHAR - 10
Product ID	NEW_ITEM-EXT_PRODUCT_ID[n]	Optional	Internal database key to identify a product in the catalog. Using this key you can jump directly to the product in the catalog during workflow approval, for example.	CHAR - 40
Description	NEW_ITEM-LONGTEXT_n:132[] (see Note below)	Optional	Description of a configuration, for example.	CHAR - No restriction

Item	Name	Required/Optional	Details	Type Length
Attachment	NEW_ITEM-ATTACHMENT	Optional	The field contains a URL to an attachment. The buyer system (SAP Business Connector) connects to the URL. You could use this field to append a particular configuration as an XML file, for example.	CHAR - 255
Attachment title	NEW_ITEM-ATTACHMENT_TITLE	Optional	If the attachment title is transferred, this field contains this title. Otherwise, the field contains the file name taken from the field NEW_ITEM-ATTACHMENT.	CHAR - 255
Attachment purpose	NEW_ITEM-ATTACHMENT_PURPOSE	Optional	If an attachment refers to a configuration, for a PC or car, for example, this field contains the letter c.	CHAR - 1
External schema type	NEW_ITEM-EXT_SCHEMA_TYPE	Optional	This field contains a schema name, as it appears in the procurement system.	CHAR - 10
External category ID	NEW_ITEM-EXT_CATEGORY_ID	Optional	Unique key for a category ID. To be used if you have performed a schema import and are using external product categories within Enterprise Buyer as material groups or if you have imported external product categories for mapping purposes.	CHAR - 60
External category	NEW_ITEM-EXT_CATEGORY	Optional	Unique key for a category ID. To be used if you have performed a schema import and are using external product categories within Enterprise Buyer as material groups or if you have imported external product categories for mapping purposes. This key is version-dependent.	CHAR - 40

Item	Name	Required/Optional	Details	Type Length
Customer-specific field	NEW_ITEM-CUST_FIELD1[n]	Optional	Customer-specific field, which may be handled in a business add-in in the Enterprise Buyer system	CHAR - 10
Customer-specific field	NEW_ITEM-CUST_FIELD2[n]	Optional	As above	CHAR - 10
Customer-specific field	NEW_ITEM-CUST_FIELD3[n]	Optional	As above	CHAR - 10
Customer-specific field	NEW_ITEM-CUST_FIELD4[n]	Optional	As above	CHAR - 20
Customer-specific field	NEW_ITEM-CUST_FIELD5[n]	Optional	As above	CHAR - 50



- Across the supply chain, many different IDs exist for a material. The manufacturer, vendor, and buyer all use different material or part numbers. Rarely are any of these numbers the same. To reflect this, different fields are provided in the OCI for these different IDs (MATNR for the buyer, VENDORMAT for the vendor, MANUFACTMAT for the manufacturer). Of all these IDs, the product number for the buyer has special significance since the shopping cart is sent to the buyer.
- The field NEW_ITEM-LONGTEXT forms an exception. In this case, the index must be attached with an underscore followed by 132 and empty brackets: NEW_ITEM-LONGTEXT_n:132[] (see also the [Example of a Catalog Interface \[Page 16\]](#)).
- The functions DETAIL and VALIDATE will only work if the parameter NEW_ITEM_EXT_PRODUCT_ID[n] was filled from the catalog in a previous call.
- The index of the line items is shown as 'n' in the table above.
- The content of the five customer fields can be handled using business add-in BBP_CATALOG_TRANSFER.
- You can add several name/value pairs and read specific values for these pairs using the Business Add-In CAT_CALL_ENRICH.

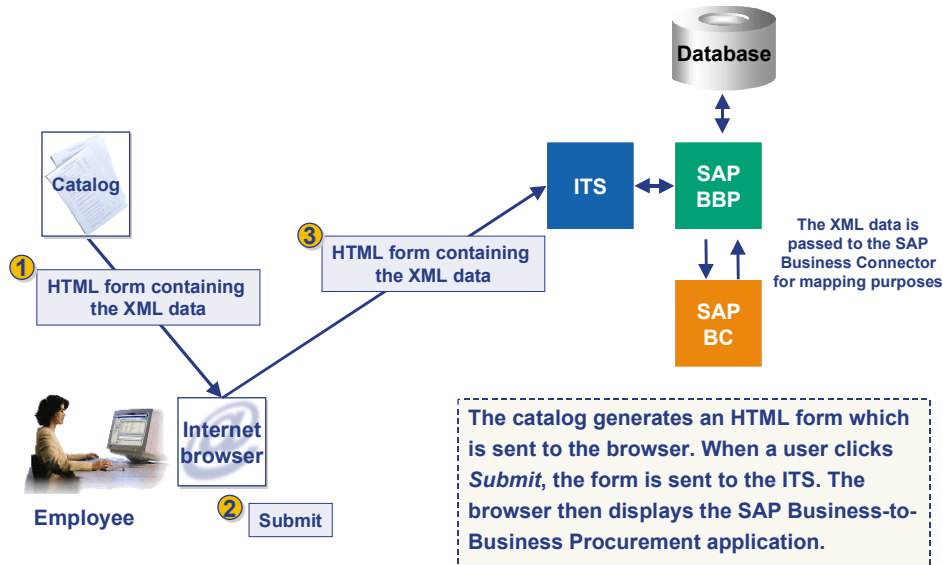
For further documentation on how to develop Business Add-Ins, see:

Implementation Guide (IMG): *Enterprise Buyer professional edition* → *Business Add-Ins for Enterprise Buyer* → *Copy Shopping Cart from Catalog in Enterprise Buyer*

XML Version

The interface is also able to process XML as input. The prerequisite for this is that you install the SAP Business Connector on the receiving side, which is used as a mapping tool.

The following graphic illustrates the data flow of the inbound interface:



If XML is used, the XML file has to be passed over to the browser in an HTML form in an input field with the name `~xmlDocument`. (This should have the type `hidden`.) The content of the generated XML file is expected to be bound to the input field `~xmlDocument` as a base64-encoded string (see the [Example of a Catalog Interface \[Page 16\]](#)). The encoding can be done directly on the server side, or, as in the example, on the client side.

In addition the type of XML to be transferred has to be passed over in another input field. The name of this field is `~xmlType`. This variable is used to distinguish between several XML schemas on the SAP Business Connector. Possible values for this field are: ESAPO (Encoded SAP Object) and ERNO (Encoded RosettaNet Object). To see the SAP XML schema, refer to the file `pdi_oci.xsd` at <http://www.sap.com/softwarepartner/scenarios> *mySAP E-Procurement* → *Enterprise Buyer Professional* → *B2B-OCI*.

Additional Catalog Functions

The table below shows what data is expected to be returned to the catalog if one of the additional functions is called:

	Function	Data To Be Returned From The Catalog
Detail of a product	DETAIL	No data has to be returned from the catalog. The detail view of the product has to be shown in the browser.
Validation of a product	VALIDATE	The application expects the complete product data to be returned. If the product no longer exists, the catalog is not expected to return any data. Data transfer must be done in the background.
Sourcing of a product	VALIDATE	The catalog is expected to transfer the data in the <i>normal</i> way.

Example of a Catalog Interface

The following excerpts are examples of the HTML and XML source code generated by a catalog engine after a user has finished selecting items for a shopping cart. You can extract the source code and display the data in a Web browser.

HTML version

In the HTML version, all the line items are placed as name/value pairs in an HTML form as shown below. This can be done as text or hidden fields. The parameter `HOOK_URL` has to be placed in the action parameter of that form, so that the HTML form can be sent to the SAP Business-to-Business Procurement system.

HTML Form (interface.asp):

```
<HTML>
<HEAD>
<HEAD>
<H1>SAP Open Catalog Interface: HTML - Example</H1>
</HEAD>
<P>This is an example, what needs to be generated by the catalog engine, if
the user is done with picking items in his shopping basket.</P>
<P>View the source of this file to see all the text inputs.</P>
<BODY bgcolor="#DED6C2">
<%
If Request.QueryString("HOOK_URL") Then
Hook = Request.QueryString("HOOK_URL")
Version = Request.QueryString("OCI_VERSION")
Else
Hook = Request.Form("HOOK_URL")
Version = Request.Form("OCI_VERSION")
End If
%>
<br>
<FORM action="<%=Hook%>" method=post target=_top>
<!-- For more information about the interface fields,
refer to the catalog interface description -->
<input type="hidden" name="~OkCode" value="ADDI">
<input type="hidden" name="~target" value="_top">
<input type="hidden" name="~CALLER" value="CTLG">
<!--Item number 1-->
<input type="hidden" name="NEW_ITEM-DESCRIPTION[1]" value = "Pen, Easytouch,
Medium, Black">
<input type="hidden" name="NEW_ITEM-MATNR[1]" value = "">
<input type="hidden" name="NEW_ITEM-MATGROUP[1]" value = "">
<input type="hidden" name="NEW_ITEM-QUANTITY[1]" value = "1">
```



```
<input type="hidden" name="NEW_ITEM-UNIT[1]" value = "EA">
<input type="hidden" name="NEW_ITEM-PRICE[1]" value = "50.00">
<input type="hidden" name="NEW_ITEM-PRICEUNIT[1]" value = "5">
<input type="hidden" name="NEW_ITEM-CURRENCY[1]" value = "USD">
<input type="hidden" name="NEW_ITEM-LEADTIME[1]" value = "1">
<input type="hidden" name="NEW_ITEM-VENDOR[1]" value = "">
<input type="hidden" name="NEW_ITEM-VENDORMAT[1]" value = "">
<input type="hidden" name="NEW_ITEM-MANUFACTCODE[1]" value = "4711">
<input type="hidden" name="NEW_ITEM-MANUFACTMAT[1]" value = "4712">
<input type="hidden" name="NEW_ITEM-CONTRACT[1]" value = "111">
<input type="hidden" name="NEW_ITEM-CONTRACT_ITEM[1]" value = "1111">
<input type="hidden" name="NEW_ITEM-SERVICE[1]" value = "">
<input type="hidden" name="NEW_ITEM-EXT_QUOTE_ID[1]" value = "111">
<input type="hidden" name="NEW_ITEM-EXT_QUOTE_ITEM[1]" value = "1111">
<input type="hidden" name="NEW_ITEM-EXT_PRODUCT_ID[1]" value = "11111">
<input type="hidden" name="NEW_ITEM-LONGTEXT_1:132[]" value =
"longtext_1:The best pen in our program, lightweight and easy to handle">
<input type="hidden" name="NEW_ITEM-CUST_FIELD1[1]" value = "custf1.1">
<input type="hidden" name="NEW_ITEM-CUST_FIELD2[1]" value = "custf1.2">
<input type="hidden" name="NEW_ITEM-CUST_FIELD3[1]" value = "custf1.3">
<input type="hidden" name="NEW_ITEM-CUST_FIELD4[1]" value = "custf1.4">
<input type="hidden" name="NEW_ITEM-CUST_FIELD5[1]" value = "custf1.5">
<input type="hidden" name="NEW_ITEM-ATTACHMENT[1]" value =
"http://www.heise.de/ct/motive/01/04/p800.jpg">
<input type="hidden" name="NEW_ITEM-ATTACHMENT_TITLE[1]" value =
"pinguine...">
<input type="hidden" name="NEW_ITEM-ATTACHMENT_PURPOSE[1]" value = "">
<input type="hidden" name="NEW_ITEM-EXT_SCHEMA_TYPE[1]" value="UNSPSC">
<input type="hidden" name="NEW_ITEM-EXT_CATEGORY_ID[1]" value="50101610">
<input type="submit" value="TransferItemstoB2Bshoppingbasket"
id=submit1name=submit1><br>
</FORM>
&copy;2001,SAPMarkets
</BODY>
</HTML>
```

XML version

In the XML version, the data from a shopping cart can be generated into a separate file which is then linked to the HTML form. The XML data is expected to be in the value of an input field with the name `~xmlDocument`. It can be set there dynamically using the function `SAP_encode_b64(Str)`, as shown in the example below. In the input field `~xmlType`, you have to specify the type of schema used to generate the XML data. Possible values for this field are: ESAPO (Encoded SAP Object) and ERNO (Encoded RosettaNet Object). To view the SAP XML schema refer to the file `pdi_oci.xsd` in SAPNet. To access this files, enter the URL <http://www.sap.com/softwarepartner/scenarios>, specifying the aliases and choose the menu options *mySAP E-Procurement* → *Enterprise Buyer Professional B2B-OCI*.

HTML Form (interface.asp)

```
<HTML>

<script language="JavaScript1.2">
function SAP_encode_b64(Str) {
var encStr = "";
var base64 = [
'A','B','C','D','E','F','G','H','I','J','K','L','M',
'N','O','P','Q','R','S','T','U','V','W','X','Y','Z',
'a','b','c','d','e','f','g','h','i','j','k','l','m',
'n','o','p','q','r','s','t','u','v','w','x','y','z',
'0','1','2','3','4','5','6','7','8','9','+','/' ];
for (var i = 0; i < Str.length; i += 3) {
    encStr += base64[(Str.charCodeAt(i) >>> 2)];
    if(!Str.charAt(i+1)) {encStr += '=='; break;}
    encStr += base64[(((Str.charCodeAt(i) & 0x03) << 4) | Str.charCodeAt(i+1)
>>> 4)];
    if(!Str.charAt(i+2)) {encStr += '='; break;}
    encStr += base64[(((Str.charCodeAt(i+1) & 0x0F) << 2) |
Str.charCodeAt(i+2) >>> 6)];
    encStr += base64[(Str.charCodeAt(i+2) & 0x3F)];
}
return encStr;
}
</script>

<HEAD>
<H1>SAP Open Catalog Interface: XML - Example</H1>
</HEAD>

<P>This is an example, what needs to be generated by the catalog engine,
if the user is done with picking items in his shopping basket.</P>
<P>View the source of the file 'example1.xml' to see all the items.</P>
```

```

<!-- insert the order -->
<BODY bgcolor="#DED6C2">

<XML ID=xmlid src="example1.xml"></XML>

<!-- Transfer the order to the B2B Application -->
<form action="%=Request.QueryString("HOOK_URL")%" method="post"
name="OrderForm" onSubmit="OrderForm['~xmlDocument'].value =
SAP_encode_b64(xmlid.xml) ">

    <input type="hidden" name="~xmlDocument" value="" >
    <input type="hidden" name="~xml_type" value="ESAPO">
<input type="submit" value="Transfer Items to B2B shopping basket"
id=submit1 name=submit1><br>

</form>
&copy; 2000, SAP AG
</BODY>
</HTML>

```

XML data (example1.xml)

```

<?xml version = "1.0"?>
<BusinessDocument>
  <CatalogHeader>
  </CatalogHeader>
  <Catalog>
    <CatalogID>11</CatalogID>
    <Product ProductType = "Good">
      <CatalogKey>11111</CatalogKey>
      <ParentCategoryID>44120000</ParentCategoryID>
      <Description Language = "EN">Pen, Easytouch, Medium,
Black</Description>
      <ShoppingBasketItem RefVendorDescription = "0"
RefManufacturerDescription = "1">
        <Quantity UoM = "EA">1</Quantity>
        <NetPrice>
          <Price Currency = "USD">0.50</Price>
          <PriceUnit>5</PriceUnit>
        </NetPrice>
        <LeadTime>1</LeadTime>
        <Quote>
          <QuoteID>111</QuoteID>
          <QuoteItemID>1111</QuoteItemID>

```

```

        </Quote>
        <ItemText Language = "EN">
            The best pen in our program, lightweight and easy to
handle
            </ItemText>
        </ShoppingBasketItem>
        <ManufacturerDescription ID = "1">
            <PartnerProductID Code = "Other">4712</PartnerProductID>
            <PartnerID Code = "Other">4711</PartnerID>
        </ManufacturerDescription>
        <VendorDescription ID = "0">
            <PartnerProductID Code = "Other">648570</PartnerProductID>
            <PartnerID Code = "Other">1768</PartnerID>
            <LeadTime>1</LeadTime>
            <BuyerContract>
                <ContractID>111</ContractID>
                <ContractItemID>1111</ContractItemID>
            </BuyerContract>
        </VendorDescription>
    </Product>
    <Product ProductType = "Good">
        <ProductID Code = "Buyer">KB-SPEZ</ProductID>
        <CatalogKey>22222</CatalogKey>
        <Description Language = "EN">Palm Pilot, the second
item</Description>
        <ShoppingBasketItem RefVendorDescription = "2"
RefManufacturerDescription = "3">
            <Quantity UoM = "EA">1</Quantity>
            <NetPrice>
                <Price Currency = "USD">225</Price>
                <PriceUnit>222</PriceUnit>
            </NetPrice>
            <LeadTime>1</LeadTime>
            <Quote>
                <QuoteID>222</QuoteID>
                <QuoteItemID>2222</QuoteItemID>
            </Quote>
            <ItemText Language = "EN">

```

The award-winning Palm Computing organizers, designed as companion products to personal computers, enable mobile users to manage their schedules, contacts and other critical personal and business information on their desktops and remotely. Palm Computing organizers automatically synchronize their information with a personal computer locally or over a local or wide area network at the touch of a button.

Their most distinguishing features include shirt-pocket size, instant response, an elegant graphical user interface and an innovative desktop docking cradle which facilitates two-way synchronization between the PC and organizer.

```

        </ItemText>
    </ShoppingBasketItem>
    <ManufacturerDescription ID = "3">
        <PartnerProductID Code = "Other">222</PartnerProductID>
        <PartnerID Code = "Other">2222</PartnerID>
    </ManufacturerDescription>
    <VendorDescription ID = "2">
        <PartnerProductID Code = "Other">12345</PartnerProductID>
        <PartnerID Code = "Other">1768</PartnerID>
        <LeadTime>1</LeadTime>
        <BuyerContract>
            <ContractID>222</ContractID>
            <ContractItemID>2222</ContractItemID>
        </BuyerContract>
    </VendorDescription>
</Product>
<Product ProductType = "Good">
    <CatalogKey>33333</CatalogKey>
    <Description Language = "EN">Palm Pilot, the third
item</Description>
    <ShoppingBasketItem RefVendorDescription = "4"
RefManufacturerDescription = "5">
        <Quantity UoM = "EA">77</Quantity>
        <NetPrice>
            <Price Currency = "USD">225</Price>
        </NetPrice>
        <LeadTime>1</LeadTime>
        <Quote>
            <QuoteID>333</QuoteID>
            <QuoteItemID>3333</QuoteItemID>
        </Quote>
        <ItemText Language = "EN">

```

The award-winning Palm Computing organizers, designed as companion products to personal computers, enable mobile users to manage their schedules, contacts, and other critical personal and business information on their desktops and remotely. Palm Computing organizers automatically synchronize their information with a personal computer locally or over a local or wide area network at the touch of a button. Their most distinguishing features include shirt-pocket size, instant response, an elegant graphical user interface and an innovative desktop docking cradle which facilitates two-way synchronization between the PC and organizer.

```
                </ItemText>
    </ShoppingBasketItem>
    <ManufacturerDescription ID = "5">
        <PartnerProductID Code = "Other">333</PartnerProductID>
        <PartnerID Code = "Other">3333</PartnerID>
    </ManufacturerDescription>
    <VendorDescription ID = "4">
        <PartnerProductID Code = "Other">12345</PartnerProductID>
        <PartnerID Code = "Other">1768</PartnerID>
        <LeadTime>1</LeadTime>
        <BuyerContract>
            <ContractID>333</ContractID>
            <ContractItemID>3333</ContractItemID>
        </BuyerContract>
    </VendorDescription>
</Product>
</Catalog>
</BusinessDocument>
```