XALT ORACLE CONNECTOR

Xalt's capabilities for mobile and cloud work as a cloud-based enterprise software platform that connects all your administrative database systems to your users on any PC browser or any mobile device. This part of our platform provides a flexible and secure environment for the creation and delivery of an unlimited number of business apps utilising a single download from the Apple App Store or Google Play.

ARCHITECTURE OVERVIEW

Connectors are pieces of Java code that facilitate the data movement between your data sources and the Xalt Cloud. Connectors reside on the Connector Gateway (Figure 1 below), where they translate the requests issued by the Xalt Cloud into a format consumable by the data source, and then translate the response back into the format needed by the Xalt Cloud. You may think of them as language packs that you snap in when you want to converse with a new system. In Figure 1, the connectors invoke the communication lines between the Connector Gateway and the various data sources. The connectors are also responsible for gathering the metadata used in Xalt. In most cases, a single connector is responsible for a single language (such as SQL), but in some cases where the language does not provide a programmatically efficient method of metadata discovery (such as REST web services), a connector must be custom built to the specifications of the data source.

The metadata gathered by the connector includes the data structures available from the data source, their properties, and the data types that make up those properties. Using a relational database as an example, the connector retrieves a list of tables or views, the columns that make up the tables, and the data types of each column. Aside from gathering metadata, a connector provides four basic data functions: create, read, update, and delete. These functions are the basic interfaces needed to allow Xalt to interact effectively with your data.

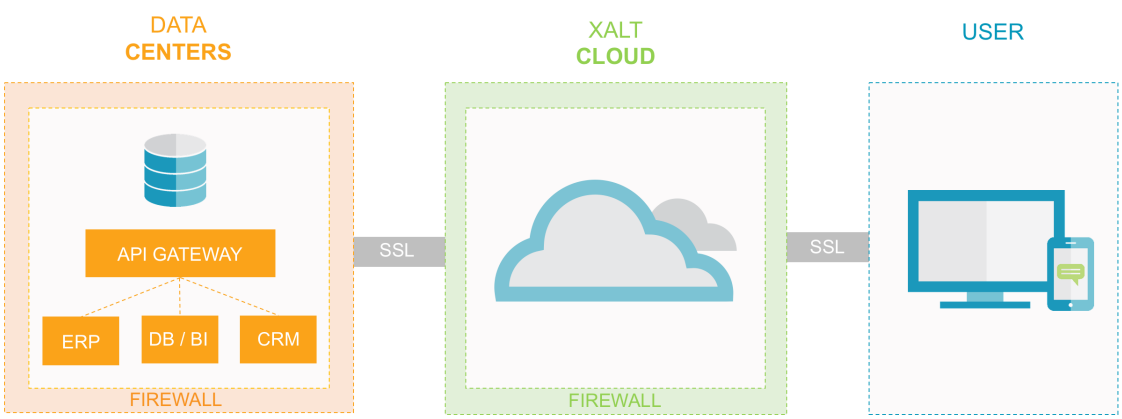

Figure 1

CONNECTING TO ORACLE

jdbc:oracle:thin:@ADDRESS:PORT:INSTANCE

Xalt – Use of existing Oracle Views

The Xalt connection provides access to the existing Oracle EBS view definitions, e.g. OE_ORDER_HEADERS_V, OE_ORDER_LINES_V and the Oracle EBS table definitions, e.g. HZ_CUST_ACCOUNTS, HZ_PARTIES.

XALT READ PROCEDURE

```
create or replace PROCEDURE CVQ_SP_LOAD_CUSTOMER
IS
 cursor c1 is
  SELECT hz_cust_accounts.cust_account_id, hz_cust_accounts.account_number, hz_cust_accounts.status,
   hz_cust_accounts.customer_type, NVL(hz_cust_accounts.PRIMARY_SALESREP_ID, hzb.PRIMARY_SALESREP_ID) as
salesrep_id, hz_cust_accounts.price_list_id,
   hz_cust_accounts.tax_code, hz_cust_accounts.tax_header_level_flag,
   hz_parties.party_name, hz_parties.tax_reference,
   hz_parties.address1, hz_parties.address2, hz_parties.address3, hz_parties.address4, hz_parties.city,
   hz_parties.postal_code, hz_parties.state, hz_parties.county, hzca.attribute1
   FROM hz_cust_accounts join hz_parties on hz_cust_accounts.party_id = hz_parties.party_id
   join hz_cust_acct_sites_all hzca on hz_cust_accounts.cust_account_id=hzca.cust_account_id
   join hz_cust_site_uses_all hzb on hzca.cust_acct_site_id = hzb.cust_acct_site_id;


 BEGIN

  FOR account_rec in c1
  LOOP
  EXIT WHEN c1%NOTFOUND;
   cnt := 0;
   BEGIN
    select count(*) into cnt from CVQ_Customer where COMPANY_NUMBER = account_rec.cust_account_id and
COMPANY_ID = 1;
   END;

   IF( cnt = 0 ) then
     BEGIN

       INSERT INTO CVQ_Customer.....
     END;
                                ELSE
     BEGIN

      UPDATE CVQ_Customer.....

     END;
                               END IF;

  END LOOP;

 RETURN;

END CVQ_SP_LOAD_CUSTOMER
```

XALT UPDATE PROCEDURE

Develop a Xalt stored-procedure to make the required call to Oracle API. This was developed using the standard Oracle API definition – further details in Appendix A below

Sample code below.

```
create or replace PROCEDURE      “CVQ_SP_SEND_QUOTE_TO_EBS”
-- ============================================
 -- Author:  jah
 -- Create date: <Create Date,,>
 -- Description:  <Description,,>
 -- ============================================
(
 v_p_line_number IN NUMBER,
 v_p_include_products IN NCHAR,
 v_p_success OUT NCHAR,
 v_p_errors OUT VARCHAR2
)
AS
-- Catavolt Variables removed from Sample

-- Oracle EBS Variables
l_hdr_rec         OE_Blanket_PUB.Header_Rec_Type;
l_hdr_val_rec      OE_Blanket_PUB.header_val_Rec_type;
l_hdr_rec_Null     OE_Blanket_PUB.header_rec_type;
l_line_tbl        OE_Blanket_PUB.line_tbl_Type;
l_line_tbl_Null    OE_Blanket_PUB.line_tbl_Type;
l_line_val_rec     OE_Blanket_PUB.line_val_Rec_type
l_line_rec        OE_Blanket_PUB.line_rec_Type;
l_line_val_tbl     OE_Blanket_PUB.line_Val_tbl_Type;
l_control_rec      OE_Blanket_PUB.Control_Rec_Type;
x_line_tbl        OE_Blanket_PUB.line_tbl_Type;
x_header_rec       OE_Blanket_PUB.header_Rec_type;
x_msg_count        NUMBER;
x_msg_data         VARCHAR2(2000);
l_return_status    VARCHAR2(2000);
x_return_status    VARCHAR2(30);
l_msg_count        NUMBER;
l_msg_data         VARCHAR2(2000);
l_message          VARCHAR2(2000);
l_msg_index_out    NUMBER;
G_DB_LOG_FLAG      Varchar2(240):= ‘LOG’;
V_List_Header_ID   Number;
V_Error_Msg        Varchar2(2000);
V_Process_Flag     Varchar2(1);
Kv              NUMBER;
V_St_Line_Number   NUMBER;
```

```
BEGIN
 -- BUSINESS LOGIC TO PROCESS,
 -- READ REQUIRED VALUES AND VALIDATE REQUEST

 -- USE CATAVOLT DATA TO CALL ORACLE EBS API

  -- setting OM debug level and writing debug info to a debug file
  --OE_DEBUG_PUB.DEBUG_ON;
  --OE_DEBUG_PUB.INITIALIZE;
  --L_FILE_VAL := OE_DEBUG_PUB.SET_DEBUG_MODE('FILE');
  --oe_debug_pub.setdebuglevel(5);
  --oe_debug_pub.add('Enter create BSA ',1);
  --dbms_output.put_line('The debug file is :'||OE_DEBUG_PUB.G_DIR||'/'||OE_DEBUG_PUB.G_FILE);
  Fnd_Global.apps_initialize(1245,50275,660);
  MO_GLOBAL.INIT('ONT');

-- SET BSA HEADER ATTRIBUTES

 -- CONSTANT EBS VALUES
  l_hdr_rec           := OE_Blanket_PUB.G_MISS_HEADER_REC;    -- Consider header record as missing
  l_hdr_val_rec       := OE_Blanket_PUB.G_MISS_HEADER_VAL_REC; -- Consider header val record as missing
  l_hdr_rec.operation    := OE_Globals.G_OPR_CREATE;          -- Header Operation

  l_hdr_rec.transactional_curr_code    := 'USD';
  l_hdr_rec.org_id                 := 101;
  l_hdr_rec.order_category_code       := 'ORDER';
  l_hdr_rec.enforce_price_list_flag := 'N';   -- will enforce price list on the releases.
  l_hdr_rec.payment_term_id       := 4;    -- Payment term NET 30
  l_hdr_rec.start_date_active     := Sysdate;
  l_hdr_rec.end_date_active       := Sysdate + 1;
  l_hdr_rec.transaction_phase_code := 'F';
  l_hdr_rec.open_flag := 'Y';
  l_hdr_rec.order_type_id  := 1001;
  l_hdr_rec.ship_from_org_id := 122;
  l_hdr_rec.attribute1  := '3'; -- Order Suffix (FLEX)

 -- DYNAMIC VALUES
  l_hdr_rec.sold_to_org_id := v_w_ebs_customer;
  l_hdr_rec.price_list_id  := v_w_Quote_price_list_id;
  l_hdr_rec.invoice_to_org_id := v_w_Invoice_to_org_id;
  l_hdr_rec.sales_document_name := v_w_SalesDocumentName;
  l_hdr_rec.salesrep_id := v_w_salespersonid; --v_w_SalesRepID;
```

```
  -- FLEX FIELDS
  l_hdr_rec.attribute3  := v_w_Quote_commencement; -- Notice of Commencement
  l_hdr_rec.attribute4  := v_w_quote_usage; -- Sales Usage Code --54
  l_hdr_rec.attribute16 := v_w_taxable;   -- Tax Exempt
  l_hdr_rec.attribute19 := v_w_ship_to_ohio; -- Ship to Location (for OHIO) -- 1046
  l_hdr_rec.attribute8 := v_w_comment1 || v_w_comment2 || v_w_comment3; -- Special Handling (150 chars)

 -- SET BSA LINES

  -- SELECT CATAVOLT LINE ITEMS

  DECLARE
   CURSOR db_cursor
    IS SELECT cust_prod.product_id ,
   cust_prod.QUOTE_PROD_PRODUCT ,
   cust_prod.QUOTE_PROD_QTY, --CVUTILS.CONVERT_TO_FLOAT(cust_prod.QUOTE_PROD_QTY,53) ,
   quarries.PL_LOCATION_ID ,
   cust_prod.QUOTE_PROD_PRICE, --CVUTILS.CONVERT_TO_FLOAT(cust_prod.QUOTE_PROD_PRICE,53) ,
   cust_prod.QUOTE_PROD_FREIGHT_PAY, --CVUTILS.CONVERT_TO_FLOAT(cust_prod.QUOTE_PROD_FREIGHT_PAY,53) ,
   cust_prod.QUOTE_PROD_FREIGHT_RATE, --CVUTILS.CONVERT_TO_FLOAT(cust_prod.QUOTE_PROD_FREIGHT_
RATE,53) ,
   cust_prod.QUOTE_PROD_UNIT,
   cust_prod.QUOTE_PROD_PRICE_LIST_ID
    FROM CVQ_QUOTE_PRODUCT cust_prod
    JOIN CVQ_PROJECTS_AND_LOCATIONS quarries ON quarries.pl_location_id = cust_prod.QUOTE_PROD_QUARRY
    WHERE  cust_prod.PL_ID = v_w_location_number
    AND cust_prod.QUOTE_LINE_NUMBER = v_p_line_number;

 BEGIN
   -- and the quote items
   IF ( v_sys_error = 0 ) THEN

   BEGIN
     wIndex :=1;
     OPEN db_cursor;
     FETCH db_cursor INTO v_w_product_id,v_w_product_name,v_w_qty,v_w_location_id,v_w_price,v_w_freight,v_w_
freight_pay,v_w_product_unit,v_w_prod_price_list_id;
     WHILE cvutils.fetch_status(db_cursor%FOUND) = 0

     LOOP
```

```
BEGIN

        -- SET EBS LINE ATTRIBUTES

        -- CONSTANT VALUES
        l_line_rec             := OE_Blanket_PUB.G_MISS_BLANKET_LINE_REC;
        l_line_val_rec         := OE_Blanket_PUB.G_MISS_BLANKET_LINE_VAL_REC;
        l_line_rec.operation   := OE_Globals.G_OPR_CREATE;

        -- DYNAMIC VALUES
        l_line_rec.sold_to_org_id   := v_w_ebs_customer;
        l_line_rec.inventory_item_id := v_w_product_id;
        v_w_prod_price_list_id := 109845;
        l_line_rec.price_list_id := v_w_prod_price_list_id;
        l_line_rec.blanket_min_quantity := 1;
        l_line_rec.order_quantity_uom := v_w_product_unit;
        l_line_rec.unit_list_price    := v_w_price;
        l_line_rec.pricing_uom        := v_w_product_unit;

        l_line_tbl(wIndex)    := l_line_rec;
        l_line_val_tbl (wIndex) := l_line_val_rec;

        FETCH db_cursor INTO v_w_product_id,v_w_product_name,v_w_qty,v_w_location_id,v_w_price,v_w_freight,v_w_
freight_pay,v_w_product_unit,v_w_prod_price_list_id;
         wIndex := wIndex + 1;

       END;
     END LOOP;
     CLOSE db_cursor;

    END;
  END IF;
 END;

-- ORACLE EBS PROCESS BLANKET

 oe_debug_pub.add('Before calling Process Blanket API',1);
 oe_msg_pub.initialize;
 OE_Blanket_PUB.Process_Blanket(
 p_org_id          => 101
 ,p_operating_unit    => NULL
 ,p_api_version_number => 1.0
 ,x_return_status     => x_return_status
 ,x_msg_count         => x_msg_count
 ,x_msg_data          => x_msg_data
```

```
  ,p_header_rec       => l_hdr_rec
  ,p_header_val_rec    => l_hdr_val_rec
  ,p_line_tbl         => l_line_tbl
  ,p_line_val_tbl     => l_line_val_tbl
  ,p_control_rec      => l_control_rec
  ,x_header_rec       => x_header_rec
  ,x_line_tbl         => x_line_tbl
  );

-- PROCESS RESULT

  dbms_output.put_line('Number of OE messages ');
  oe_debug_pub.add('Number of OE messages :'||x_msg_count,1);

  for k in 1 .. x_msg_count loop
   x_msg_data := oe_msg_pub.get( p_msg_index => k, p_encoded => 'F');
   dbms_Output.put_line('Message :'||x_msg_data);
   oe_debug_pub.add(substr(x_msg_data,1,255));
   oe_debug_pub.add(substr(x_msg_data,255,length(x_msg_data)));
  end loop;

 if x_return_status <> FND_API.G_RET_STS_SUCCESS then
   oe_debug_pub.add('Error in process blanket ',1);
   rollback;
   v_p_errors := '<Message type=''error''><Text>' || x_msg_data ||
            '</Text></Message>';
  else
   v_p_success := '1';
   dbms_output.put_line('New Sales Agreement Number is :'||x_header_rec.order_number||'(Header ID : '||x_header_rec.
header_id||')');
   oe_debug_pub.add('Line ID :'    ||x_line_tbl(1).line_id,1);
   oe_debug_pub.add('Header ID :'  ||x_header_rec.header_id,1);
   oe_debug_pub.add('Order number :'||x_header_rec.order_number,1);
   oe_debug_pub.add('Sold To :'    ||x_header_rec.sold_to_org_id,1);
   oe_debug_pub.add('Invoice To :' ||x_header_rec.invoice_to_org_id,1);
   oe_debug_pub.add('Ship To :'    ||x_header_rec.ship_to_org_id,1);
   v_p_errors := '<Message type=''information''><Text>Blanket Sales Agreement: ' || x_header_rec.order_number ||
            '</Text></Message>';

  end if;

EXCEPTION WHEN OTHERS THEN cvutils.handleerror(SQLCODE,SQLERRM);

END CVQ_SP_SEND_QUOTE_TO_EBS;
```

APPENDIX A

Standard Oracle Support Document

How to create Blanket Sales Agreement using public API OE_BLANKET_PUB.Process_Blanket in Oracle Order Management (Doc ID 790223.1)

Oracle Order Management - Version 12.1.1 and later
Information in this document applies to any platform.
Checked for relevance on 10-OCT-2012

Purpose

This note is intended to show how the creation of Blanket Sales Agreement action can be simulated using the newly available Public API in R12.1.1. .

Scope

The intended audience is for those with technical knowledge of Oracle Applications Code, the usage of public APIs, and those familiar with creating custom solutions for unique business requirements.

Details
Assumption:
==========
The following setups related to the Blanket Sales Agreement are not in the scope of this document and we assume that these setup steps are already in place.

1. Setup of order transaction type
2. Setup of document sequence
3. Setup of document assignment

The Blanket Sales Agreement feature has been around from Release 11.5.9, and its creation in Order Management has been supported from Blanket Sales Agreement Form. From Release 12.1.1, support has been provided to create BSA from backend using the newly available public API OE_BLANKET_PUB.Process_Blanket().

The sales Agreement functionality supports standard, ATO items and Kits and also includes the ability to create releases by order import and process order API.

Sales Agreement header created usually includes the following information:

- Customer, ship to , Bill to , Version e.t.c
- Effective Dates
- Payment and freight terms between the customer and supplier
- Sales Agreement Min and Max quantity
- Control flag to determine whether you can exceed the maximum value
- Pricing information such as Standard price lists or Sales Agreement price lists.

Sales Agreement Lines Include the following information:

- Item categories to store the categories and ALL ITEMS to cover all items.
- Min and Max quantity agreed by the customer and supplier
- Price including the choice of price list in addition to line level modifiers: Discount % and discount amount.
- Effective dates

Procedure Parameter Description:

****

OE_BLANKET_PUB.process_Blanket() accepts the following parameters

Parameter	In/ OUT	Type	Description
p_api_version_number	IN	NUMBER	API Version
x_return_status	OUT	VARCHAR2	Return Status
P_org_id	IN	NUMBER	Org Id (MOAC)
P_operating_Unit	IN	NUMBER	Operating Unit (MOAC)
p_header_rec	IN	OE_Blanket_PUB.header_rec_type	Header Rec
p_header_val_rec	IN	OE_Blanket_PUB.header_val_rec_type	Header Value Rec
p_line_tbl	IN	OE_Blanket_PUB line_tbl_Type	Inbound Line table
p_line_val_tbl	IN	OE_Blanket_PUB.line_val _tbl_Type	Inbound Line Values table
p_control_rec	IN	OE_Blanket_PUB.Control_Rec_Type	Inbound Control Record
x_header_rec	OUT NOCOPY	OE_Blanket_PUB.header_rec_type	Outbound Header Record
x_line_tbl	OUT NOCOPY	OE_Blanket_PUB line_tbl_Type	Outbound Line table
x_msg_count	OUT	NUMBER	Msg Count
x_msg_data	OUT	VARCHAR2	Msg Data
p_validate_desc_flex	IN	VARCHAR2	To Validate FF or not ?

Record Parameter description for header_rec_type:

==========

Attribute	Type
accounting_rule_id	NUMBER
agreement_id	NUMBER
attribute1- Attribute20	VARCHAR2(240)
context	VARCHAR2(30)
created_by	NUMBER
creation_date	DATE
cust_po_number	VARCHAR2(50)
deliver_to_org_id	NUMBER
freight_terms_code	VARCHAR2(30)
header_id	NUMBER
invoice_to_org_id	NUMBER
invoicing_rule_id	NUMBER
last_updated_by	NUMBER
last_update_date	DATE
last_update_login	NUMBER
order_category_code	VARCHAR2(30)
order_number	NUMBER
order_type_id	NUMBER
org_id	NUMBER
price_list_id	NUMBER
program_application_id	NUMBER
program_id	NUMBER
program_update_date	DATE
request_id	NUMBER
version_number	NUMBER
salesrep_id	NUMBER
shipping_method_code	VARCHAR2(30)
ship_from_org_id	NUMBER
ship_to_org_id	NUMBER
sold_to_contact_id	NUMBER
sold_to_org_id	NUMBER
transactional_curr_code	VARCHAR2(15)
return_status	VARCHAR2(1)
db_flag	VARCHAR2(1)
operation	VARCHAR2(30)
payment_term_id	NUMBER
shipping_instructions	varchar2(2000)
packing_instructions	varchar2(2000)
Price_list_Name	varchar2(240)
Price_list_description	varchar2(2000)
price_list_currency_code	varchar2(30)
conversion_type_code	varchar2(30)
blanket_max_amount	NUMBER

blanket_min_amount NUMBER
released_amount Number
fulfilled_amount Number
returned_amount Number
enforce_price_list_flag varchar2(1)
enforce_ship_to_flag varchar2(1)
enforce_invoice_to_flag varchar2(1)
enforce_freight_term_flag varchar2(1)
enforce_shipping_method_flag varchar2(1)
enforce_payment_term_flag varchar2(1)
enforce_accounting_rule_flag varchar2(1)
enforce_invoicing_rule_flag varchar2(1)
lock_control number
on_hold_flag VARCHAR2(1)
override_amount_flag varchar2(1)
start_date_active DATE
end_date_active DATE
revision_change_comments VARCHAR2(2000)
revision_change_date DATE
revision_change_reason_code VARCHAR2(30)
source_document_type_id NUMBER
source_document_id NUMBER
SALES_DOCUMENT_NAME VARCHAR2(240)
TRANSACTION_PHASE_CODE VARCHAR2(30)
USER_STATUS_CODE VARCHAR2(30)
flow_status_code VARCHAR2(30)
SUPPLIER_SIGNATURE VARCHAR2(240)
SUPPLIER_SIGNATURE_DATE DATE
CUSTOMER_SIGNATURE VARCHAR2(240)
CUSTOMER_SIGNATURE_DATE DATE
SOLD_TO_SITE_USE_ID NUMBER
DRAFT_SUBMITTED_FLAG VARCHAR2(1)
SOURCE_DOCUMENT_VERSION_NUMBER NUMBER
contract_template_id NUMBER
new_price_list_id NUMBER
new_price_list_name VARCHAR2(240)
new_modifier_list_id NUMBER
new_modifier_list_name VARCHAR2(240)
default_discount_percent NUMBER
default_discount_amount NUMBER
open_flag VARCHAR2(1)
Record Parameter description for line_Rec_type:
==========

Attribute	Type
accounting_rule_id	NUMBER
agreement_id	NUMBER
attribute1- attribute20	VARCHAR2(240)
context	VARCHAR2(30)
created_by	NUMBER
creation_date	DATE
cust_po_number	VARCHAR2(50)
deliver_to_org_id	NUMBER
freight_terms_code	VARCHAR2(30)
global_attribute_category	VARCHAR2(30)
header_id	NUMBER
inventory_item_id	NUMBER
invoice_to_org_id	NUMBER
invoicing_rule_id	NUMBER
ordered_item	VARCHAR2(2000)
ordered_item_id	NUMBER
last_updated_by	NUMBER
last_update_date	DATE
last_update_login	NUMBER
line_type_id	NUMBER
line_id	NUMBER
line_number	NUMBER
order_number	VARCHAR2(240)
order_quantity_uom	VARCHAR2(30)
org_id	NUMBER
payment_term_id	NUMBER
preferred_grade	VARCHAR2(150)
price_list_id	NUMBER
request_id	NUMBER
program_id	NUMBER
program_application_id	NUMBER
program_update_date	DATE
shipping_method_code	VARCHAR2(30)
ship_from_org_id	NUMBER
ship_to_org_id	NUMBER
sold_to_org_id	NUMBER
return_status	VARCHAR2(1)
db_flag	VARCHAR2(1)
operation	VARCHAR2(30)
item_identifier_type	VARCHAR2(30)
item_type_code	varchar2(30)
shipping_instructions	VARCHAR2(2000)
packing_instructions	VARCHAR2(2000)

salesrep_id number
unit_list_price number
pricing_uom varchar2(150)
lock_control number
enforce_price_list_flag varchar2(1)
enforce_ship_to_flag varchar2(1)
enforce_invoice_to_flag varchar2(1)
enforce_freight_term_flag varchar2(1)
enforce_shipping_method_flag varchar2(1)
enforce_payment_term_flag varchar2(1)
enforce_accounting_rule_flag varchar2(1)
enforce_invoicing_rule_flag varchar2(1)
override_blanket_controls_flag varchar2(1)
override_release_controls_flag varchar2(1)
qp_list_line_id NUMBER
fulfilled_quantity number
blanket_min_quantity NUMBER
blanket_max_quantity NUMBER
blanket_min_amount number
blanket_max_amount number
min_release_quantity NUMBER
max_release_quantity NUMBER
min_release_amount number
max_release_amount number
released_amount NUMBER
fulfilled_amount NUMBER
released_quantity number
returned_amount number
returned_quantity number
start_date_active date
end_date_active date
source_document_type_id NUMBER
source_document_id NUMBER
source_document_line_id NUMBER
transaction_phase_code VARCHAR2(30)
source_document_version_number NUMBER
modifier_list_line_id NUMBER
discount_percent NUMBER
discount_amount NUMBER
revision_change_comments VARCHAR2(2000)
revision_change_date DATE
revision_change_reason_code VARCHAR2(30)

Table Description for line_tbl_Type:
==========

Parameter	Type
Line_Rec_Type	Record

The Scripts provided here can be used as reference in order to create your own wrapper files to call the API. Also you can use these scripts to troubleshoot the API specific issues by analyzing the generated debug files.

Scenario 1: Creating Blanket Sales Agreement with Sales Agreement specific Price Lists (inline Pricing)with the following criteria.

Customer would need you to supply a min quantity of 499 and max quantity of 999 over a period of next 10 months.

```
SET SERVEROUTPUT ON SIZE 100000;

prompt ------------------------------------------------------------;
prompt This script creates a Blanket Sales Agreement with one Line;
 prompt Output : Sales Agreement Number;
 prompt ------------------------------------------------------------;

DECLARE
-- Input Variables

   l_hdr_rec             OE_Blanket_PUB.header_Rec_type;
   l_hdr_val_rec           OE_Blanket_PUB.header_val_Rec_type;

   l_line_tbl            OE_Blanket_PUB.line_tbl_Type;
   l_line_val_tbl          OE_Blanket_PUB.line_Val_tbl_Type;
   l_line_rec            OE_Blanket_PUB.line_rec_Type;
   l_line_val_rec          OE_Blanket_PUB.line_val_rec_Type;
   l_control_rec           OE_Blanket_PUB.Control_rec_type;


   -- Output Variables

   x_line_tbl            OE_Blanket_PUB.line_tbl_Type;
   x_header_rec            OE_Blanket_PUB.header_Rec_type;
   x_msg_count             NUMBER;
   x_msg_data            VARCHAR2(2000);
   x_return_status           VARCHAR2(30);
```

```
 -- Incremental variables
 i                   NUMBER;
 j                   NUMBER;

BEGIN
 -- setting OM debug level and writing debug info to a debug file
 oe_debug_pub.setdebuglevel(5);
 oe_debug_pub.add('Enter create BSA ',1);
 dbms_output.put_line('The debug file is :'||OE_DEBUG_PUB.G_DIR||'/'||OE_DEBUG_PUB.G_FILE);

 --Fnd_Global.apps_initialize(&user_Id,&responsibility_Id,&resp_appl_id);

 Fnd_Global.apps_initialize(1318,21623,660);
 MO_GLOBAL.INIT('ONT'); -- MOAC

 For j IN 1..1 LOOP

  l_hdr_rec             := OE_Blanket_PUB.G_MISS_HEADER_REC; -- header record as missing
  l_hdr_val_rec         := OE_Blanket_PUB.G_MISS_HEADER_VAL_REC; -- header val rec as missing
  l_hdr_rec.operation           := OE_Globals.G_OPR_CREATE; -- Header Operation
  l_hdr_rec.sold_to_org_id      := 3347;
  l_hdr_rec.order_type_id       := 3123;
  l_hdr_rec.ship_to_org_id      := 3730;
  l_hdr_rec.attribute1          := 'dr test';
  l_hdr_rec.start_date_active   := '01-MAR-2009';
  l_hdr_rec.end_date_active     := '01-JAN-2010';
  -- Create New Price list and modifier

  l_hdr_rec.new_price_list_name     := 'dr pricelist for BSA22';
  l_hdr_rec.new_modifier_list_name  := 'dr modifier list22';
  l_hdr_rec.default_discount_percent  := 15;


  -- populate line rec

  l_line_rec                := OE_Blanket_PUB.G_MISS_BLANKET_LINE_REC;
  l_line_val_rec            := OE_Blanket_PUB.G_MISS_BLANKET_LINE_VAL_REC;
  l_line_rec.operation          := OE_Globals.G_OPR_CREATE;
  l_line_rec.sold_to_org_id     := 3347;
  l_line_rec.inventory_item_id  := 149;
  l_line_rec.blanket_min_quantity   := 499;
  l_line_rec.blanket_max_quantity   := 999;
  l_line_rec.min_release_quantity   := 499;
  l_line_rec.max_release_quantity   := 999;
```

```
  -- Item pricing details

  l_line_rec.unit_list_price          := 888;
  l_line_rec.ITEM_IDENTIFIER_TYPE     := 'INT';
  l_line_rec.pricing_uom              := 'Ea';

  for i in 1..1
  loop
    l_line_tbl(i)     := l_line_rec;
    l_line_val_tbl (i) := l_line_val_rec;
  end loop;


oe_debug_pub.add('Before calling Process Blanket API',1);
oe_msg_pub.initialize;

OE_Blanket_PUB.Process_Blanket(
  p_org_id          => 204
 ,p_operating_unit    => NULL
 ,p_api_version_number => 1.0
 ,x_return_status     => x_return_status
 ,x_msg_count         => x_msg_count
 ,x_msg_data          => x_msg_data
 ,p_header_rec        => l_hdr_rec
 ,p_header_val_rec    => l_hdr_val_rec
 ,p_line_tbl          => l_line_tbl
 ,p_line_val_tbl      => l_line_val_tbl
 ,p_control_rec       => l_control_rec
 ,x_header_rec        => x_header_rec
 ,x_line_tbl          => x_line_tbl
 );

oe_debug_pub.add('Number of OE messages :'||x_msg_count,1);
for k in 1 .. x_msg_count
loop
  x_msg_data := oe_msg_pub.get( p_msg_index => k, p_encoded => 'F');
  dbms_Output.put_line('Message :'||x_msg_data);
  oe_debug_pub.add(substr(x_msg_data,1,255));
  oe_debug_pub.add(substr(x_msg_data,255,length(x_msg_data)));
end loop;
```

```
  if x_return_status <> FND_API.G_RET_STS_SUCCESS
  then
    oe_debug_pub.add('Error in process blanket ',1);
    dbms_output.put_line('Error in Process blanket, Check the debug log file ');
    rollback;
  else
    dbms_output.put_line('New Sales Agreement Number is :'||x_header_rec.order_number||'(Header ID : '||x_header_
rec.header_id||')');
    oe_debug_pub.add('Line ID :'    ||x_line_tbl(1).line_id,1);
    oe_debug_pub.add('Header ID :'  ||x_header_rec.header_id,1);
    oe_debug_pub.add('Order number :'||x_header_rec.order_number,1);
    oe_debug_pub.add('Sold To :'    ||x_header_rec.sold_to_org_id,1);
    oe_debug_pub.add('Invoice To :' ||x_header_rec.invoice_to_org_id,1);
    oe_debug_pub.add('Ship To :'    ||x_header_rec.ship_to_org_id,1);
  end if;

  end loop;
  commit;
End;
/
```

Scenario2; Creating Blanket Sales Agreement with Standard Price lists.

```
SET SERVEROUTPUT ON SIZE 100000;
prompt ------------------------------------------------------------;
prompt This script creates a Blanket Sales Agreement with one Line;
prompt Output : Sales Agreement Number;
prompt ------------------------------------------------------------;

DECLARE

  -- Input variables

  l_hdr_rec             OE_Blanket_PUB.header_Rec_type;
  l_hdr_val_rec           OE_Blanket_PUB.header_val_Rec_type;

  l_line_tbl            OE_Blanket_PUB.line_tbl_Type;
  l_line_val_tbl          OE_Blanket_PUB.line_Val_tbl_Type;
  l_line_rec            OE_Blanket_PUB.line_rec_Type;
  l_line_val_rec          OE_Blanket_PUB.line_val_rec_Type;
  l_control_rec           OE_Blanket_PUB.Control_rec_type;
```

```
 -- output variables

 x_line_tbl              OE_Blanket_PUB.line_tbl_Type;
 x_header_rec            OE_Blanket_PUB.header_Rec_type;
 x_msg_count             NUMBER;
 x_msg_data              VARCHAR2(2000);
 x_return_status         VARCHAR2(30);

 -- Incremental variables
 i                       NUMBER;
 j                       NUMBER;

BEGIN
 -- setting OM debug level and writing debug info to a debug file

 oe_debug_pub.setdebuglevel(5);

 oe_debug_pub.add('Enter create BSA ',1);

 dbms_output.put_line('The debug file is :'||OE_DEBUG_PUB.G_DIR||'/'||OE_DEBUG_PUB.G_FILE);

 --Fnd_Global.apps_initialize(&user_Id,&responsibility_Id,&resp_appl_id);

 Fnd_Global.apps_initialize(1318,21623,660);
 MO_GLOBAL.INIT('ONT'); -- MOAC

 For j IN 1..1
 LOOP
   l_hdr_rec                  := OE_Blanket_PUB.G_MISS_HEADER_REC; -- Consider header record as missing
   l_hdr_val_rec              := OE_Blanket_PUB.G_MISS_HEADER_VAL_REC; -- Consider header val record as missing
   l_hdr_rec.operation        := OE_Globals.G_OPR_CREATE; -- Header Operation
   l_hdr_rec.sold_to_org_id   := 3347;
   l_hdr_rec.order_type_id    := 3123;
   l_hdr_rec.ship_to_org_id   := 3730;
   l_hdr_rec.attribute1       := 'dr test';
   l_hdr_rec.start_date_active := Sysdate;
   l_hdr_rec.end_date_active  := '01-JAN-2010';
   l_hdr_rec.price_list_id    := 1000; -- Pass price_list_id for corporate price list
   l_hdr_rec.enforce_price_list_flag := 'Y';  -- will enforce price list on the releases.
   l_hdr_rec.payment_term_id  := 4;   -- Payment term NET 30
```

```
   -- populate line rec
   l_line_rec                  := OE_Blanket_PUB.G_MISS_BLANKET_LINE_REC;
   l_line_val_rec              := OE_Blanket_PUB.G_MISS_BLANKET_LINE_VAL_REC;
   l_line_rec.operation        := OE_Globals.G_OPR_CREATE;
   l_line_rec.sold_to_org_id   := 3347;
   l_line_rec.inventory_item_id := 149;

-- Item pricing details

   l_line_rec.unit_list_price  := 999;
   l_line_rec.ITEM_IDENTIFIER_TYPE := 'INT';
   l_line_rec.pricing_uom      := 'Ea';

   for i in 1..1
   loop
     l_line_tbl(i)     := l_line_rec;
     l_line_val_tbl (i) := l_line_val_rec;
   end loop;


  oe_debug_pub.add('Before calling Process Blanket API',1);
  oe_msg_pub.initialize;

  OE_Blanket_PUB.Process_Blanket(
    p_org_id          => 204
   ,p_operating_unit    => NULL
   ,p_api_version_number => 1.0
   ,x_return_status     => x_return_status
   ,x_msg_count         => x_msg_count
   ,x_msg_data          => x_msg_data
   ,p_header_rec        => l_hdr_rec
   ,p_header_val_rec    => l_hdr_val_rec
   ,p_line_tbl          => l_line_tbl
   ,p_line_val_tbl      => l_line_val_tbl
   ,p_control_rec       => l_control_rec
   ,x_header_rec        => x_header_rec
   ,x_line_tbl          => x_line_tbl
   );

  oe_debug_pub.add('Number of OE messages :'||x_msg_count,1);
  for k in 1 .. x_msg_count
  loop
    x_msg_data := oe_msg_pub.get( p_msg_index => k, p_encoded => 'F');
    dbms_Output.put_line('Message :'||x_msg_data);
```

```
    oe_debug_pub.add(substr(x_msg_data,1,255));
    oe_debug_pub.add(substr(x_msg_data,255,length(x_msg_data)));
  end loop;

  if x_return_status <> FND_API.G_RET_STS_SUCCESS
  then
    oe_debug_pub.add('Error in process blanket ',1);
    dbms_output.put_line('Error in Process blanket, Check the debug log file ');
    rollback;
  else
    dbms_output.put_line('New Sales Agreement Number is :'||x_header_rec.order_number||'(Header ID : '||x_header_
rec.header_id||')');
    oe_debug_pub.add('Line ID :'    ||x_line_tbl(1).line_id,1);
    oe_debug_pub.add('Header ID :'  ||x_header_rec.header_id,1);
    oe_debug_pub.add('Order number :'||x_header_rec.order_number,1);
    oe_debug_pub.add('Sold To :'    ||x_header_rec.sold_to_org_id,1);
    oe_debug_pub.add('Invoice To :' ||x_header_rec.invoice_to_org_id,1);
    oe_debug_pub.add('Ship To :'    ||x_header_rec.ship_to_org_id,1);
  end if;

  end loop;
  commit;
End;
/
```

Conclusion:

==========

Using Process Blanket Public API , users can only create Blanket Sales Agreement without having entering them through the Sales Agreement forms User Interface.

As of now, Only the "CREATE" operation is supported by the public API for BSA.

This also provides the ability to create releases using order import and process order API.

References:

==========

Order Management Implementation Guide - Release 12

Order Management User's Guide - Release 12

About Hexagon

Hexagon is a global leader in digital solutions that create Autonomous Connected Ecosystems (ACE), a state where data is connected seamlessly through the convergence of the physical world with the digital, and intelligence is built-in to all processes.

Hexagon's industry-specific solutions leverage domain expertise in sensor technologies, software, and data orchestration to create Smart Digital Realities™ that improve productivity and quality across manufacturing, infrastructure, safety and mobility applications.

Learn more about Hexagon (Nasdaq Stockholm: HEXA B) at hexagon.com and follow us @HexagonAB.