Implementing Customizations in the Trading Community Architecture using Workflow Business Events

Dave Hebden

Protégé Software Services

Booth # 1426

Implementing Customizations in the Trading Community Architecture using Workflow Business Events

- Introduction
- No More User Hooks
- An Event For Virtually Everything
- Event Parameters
- Event Subscriptions
- Beware the Loop!
- Conclusion

Introduction

Previous to release 11.5.7 and TCA mini-pack 11i.HZ.G, customizations for the processing of Customer information in the Trading Community Architecture were implemented by User Hooks. These hooks were procedure calls from the seeded TCA PL/SQL package code to program stubs that could be enhanced by users with custom code. These hooks have been removed and replaced by Business Event Callouts that are handled by the Workflow Business Event System. The use of this powerful new feature will be discussed in detail.

No More User Hooks

Oracle once provided a facility known as User Hooks within the TCA APIs. Below is a block of code (commented out) from the old HZ_LOCATION_PUB PL/SQL package which contained the APIs for manipulating Location information in the TCA.

```
/*************** Comment out CRMHK********
    IF ((fnd_profile.value('HZ_EXECUTE_API_CALLOUTS') = 'Y') AND
       (l_location_rec.content_source_type =
          HZ_PARTY_PUB.G_MISS_CONTENT_SOURCE_TYPE)) THEN
      hz_location_crmhk.create_location_pre(
                      l_location_rec,
                 x_return_status,
                 x_msg_count,
                  x_msg_data);
.
**********************************************/
```

No More User Hooks

As you saw above, the API once made a direct call to the 'CREATE_LOCATION_PRE' procedure. Since the inception of Business Event Callouts, however, these direct calls have been removed and are now handled by the raising of a Business Event. Below is a block of code from the newer HZ_LOCATION_V2PUB:

```
IF x_return_status = FND_API.G_RET_STS_SUCCESS
THEN

    HZ_BUSINESS_EVENT_V2PVT.create_location_event (
    l_location_rec);
END IF;
```

No More User Hooks

… which in turn invokes this block:

```
    IF G_EXECUTE_API_CALLOUT = 'Y' THEN
.
        -- Raise Event
        HZ_EVENT_PKG.raise_event(
            p_event_name       => l_event_name,
            p_event_key        => l_key,
            p_parameters       => l_list );
.
```

As is evident in the code above, a profile option can be set to enable or disable these callouts on a global basis.

An Event for Virtually Everything

Each time a Person Party, Organization, Relationship, Customer Account, Address, Contact, Customer Profile, etc. is created or updated, a particular Business Event is raised by a call to a procedure in the HZ_BUSINESS_EVENT_V2PVT package.

For example, the HZ_LOCATION_V2PUB.CREATE_LOCATION API procedure invokes the following procedure in that package:

```
PROCEDURE create_location_event (
    p_location_rec IN  hz_location_v2pub.location_rec_type
                );
```

An Event for Virtually Everything

The HZ_LOCATION_V2PUB.UPDATE_LOCATION API procedure invokes the following:

```
PROCEDURE update_location_event (
    p_location_rec  IN  hz_location_v2pub.location_rec_type,
    p_old_location_rec IN hz_location_v2pub.location_rec_type
          );
```

Notice that for UPDATE events, two records are passed where one contains the pre-update version of the data and the other contains the post-update version. This allows custom logic to be performed based on the exact changes that occurred to the Location.

See the White Paper for a complete list of events.

Event Parameters

When each event is triggered in the TCA API, a group of parameters is written to a database table called HZ_PARAM_TAB. This table is generic in structure so as to accommodate the different parameter signatures of the various events.

The table layout is:

ITEM_NAME	- unique identifier for the event instance
PARAM_NAME	- column name
PARAM_CHAR	- contains column value if Character type
PARAM_NUM	- contains column value if Number type
PARAM_DATE	- contains column value if Date type
PARAM_INDICATOR	- ‘NEW’ or ‘OLD’ (‘OLD’ for update only)

Event Parameters

When each event is triggered in the TCA API, a group of parameters is written to a database table called HZ_PARAM_TAB. This table is generic in structure so as to accommodate the different parameter signatures of the various events.

For example:

ITEM_NAME	- oracle.apps.ar.hz.Location.update39306
PARAM_NAME	- P_LOCATION_REC.LOCATION_ID
PARAM_CHAR	-
PARAM_NUM	- 1080
PARAM_DATE	-
PARAM_INDICATOR	- NEW

Event Parameters

When each event is triggered in the TCA API, a group of parameters is written to a database table called HZ_PARAM_TAB. This table is generic in structure so as to accommodate the different parameter signatures of the various events.

For example:

ITEM_NAME	- oracle.apps.ar.hz.Location.update39306
PARAM_NAME	- P_LOCATION_REC.LOCATION_ID
PARAM_CHAR	-
PARAM_NUM	- 1080
PARAM_DATE	-
PARAM_INDICATOR	- OLD

Event Parameters

When each event is triggered in the TCA API, a group of parameters is written to a database table called HZ_PARAM_TAB. This table is generic in structure so as to accommodate the different parameter signatures of the various events.

For example:

ITEM_NAME	- oracle.apps.ar.hz.Location.update39306
PARAM_NAME	- P_LOCATION_REC.CITY
PARAM_CHAR	- Essen
PARAM_NUM	-
PARAM_DATE	-
PARAM_INDICATOR	- NEW

Event Parameters

When each event is triggered in the TCA API, a group of parameters is written to a database table called HZ_PARAM_TAB. This table is generic in structure so as to accommodate the different parameter signatures of the various events.

For example:

ITEM_NAME	- oracle.apps.ar.hz.Location.update39306
PARAM_NAME	- P_LOCATION_REC.CITY
PARAM_CHAR	- Esen
PARAM_NUM	-
PARAM_DATE	-
PARAM_INDICATOR	- OLD

Event Subscriptions

- To have your custom code invoked when a TCA event is fired, you must create a subscription to the appropriate event.

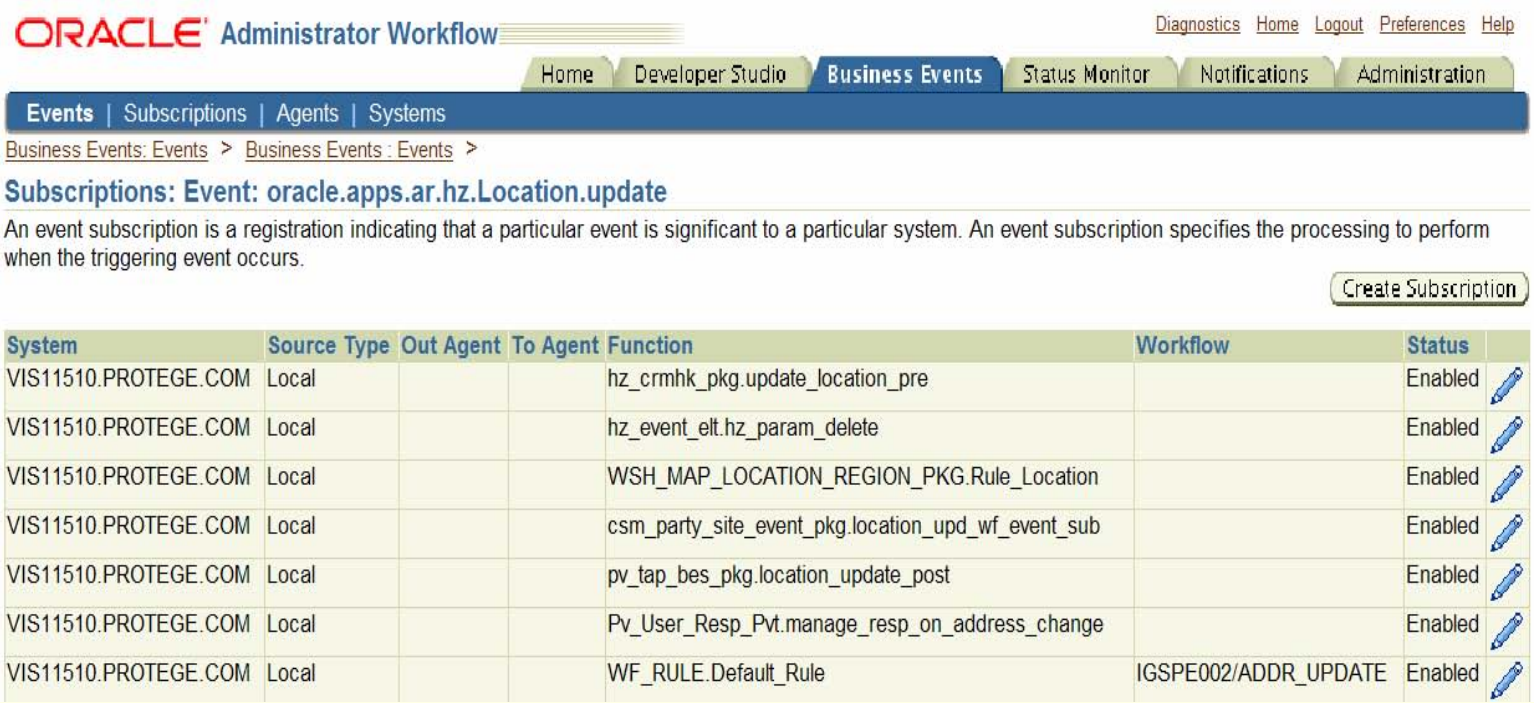
- To maintain event subscriptions, you must use a Workflow Administrator responsibility and you must have System Administration access (typically logging in as the SYSADMIN user).

- Under Administrator Workflow, choose Business Events and then perform a search for the desired event.

- Once the desired event has been located, click the Subscription icon to show a list of all subscriptions to that event.

Event Subscriptions

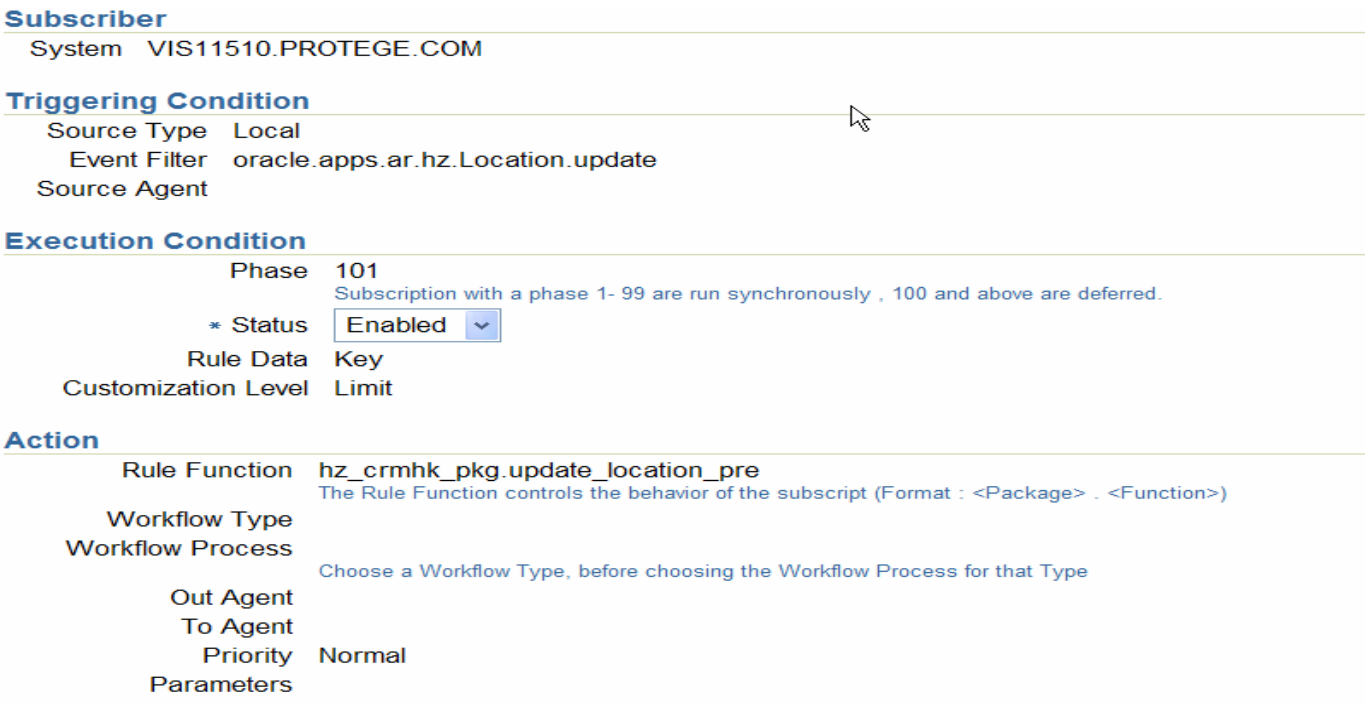
Below is a screenshot of the list of subscriptions in a test system for the Update Location event:



System	Source Type	Out Agent	To Agent	Function	Workflow	Status
VIS11510.PROTEGE.COM	Local			hz_crmhk_pkg.update_location_pre		Enabled
VIS11510.PROTEGE.COM	Local			hz_event_elt.hz_param_delete		Enabled
VIS11510.PROTEGE.COM	Local			WSH_MAP_LOCATION_REGION_PKG.Rule_Location		Enabled
VIS11510.PROTEGE.COM	Local			csm_party_site_event_pkg.location_upd_wf_event_sub		Enabled
VIS11510.PROTEGE.COM	Local			pv_tap_bes_pkg.location_update_post		Enabled
VIS11510.PROTEGE.COM	Local			Pv_User_Resp_Pvt.manage_resp_on_address_change		Enabled
VIS11510.PROTEGE.COM	Local			WF_RULE.Default_Rule	IGSPE002/ADDR_UPDATE	Enabled

Event Subscriptions

By clicking on the Update icon, we get details for the subscription (in this case, the first one listed above):



Event Subscriptions

In addition to your own logic, the rule function code must contain logic to retrieve the parameter values from the HZ_PARAM_TAB table to be able to use the Location data that was Updated when the event was fired. Below is some of the code from the first procedure listed above that is used to read the parameter table:

```
hz_param_pkg.GetParameter(p_item_key  => l_key,
                          x_param_tab => l_tab);
location_rec_fill
( P_LOCATION_REC    =>  P_LOCATION_REC,
  l_tab             =>  l_tab,
  p_ind             =>  'NEW' );
location_rec_fill
( P_LOCATION_REC    =>  P_OLD_LOCATION_REC,
  l_tab             =>  l_tab,
  p_ind             =>  'OLD' );
```

Event Subscriptions

- To facilitate the order in which subscriptions are processed for each event, a Phase Number is assigned to the subscription.

- Subscriptions are processed in order of Phase Number from lowest to highest.

- A Phase Number of 100 or higher means that the subscription processing is done in Deferred mode by the Workflow Agent (i.e. run asynchronously of the transaction that fired the event). If the Phase Number is set to 99 or lower, the subscription is processed synchronously.

- Care should be taken in using Phase Numbers of 99 or less as the transaction that fired the event will not complete until the subscription is processed.

Event Subscriptions

- One of the subscriptions listed in the screenshot above is for a function HZ_EVENT_ELT.HZ_PARAM_DELETE to be invoked. You will see that this subscription is seeded into most of the TCA events.

- This function is used to remove (clean up) the rows from the parameter table for the event. It is important that this subscription be given a higher Phase Number than any other subscription for the event so as not to delete the parameters for the event before all subscriptions that may use them are processed.

- The seeded Phase Number for this cleanup function is 500. It is very important that this subscription exists in every event that writes parameters to the HZ_PARAM_TAB table. If rows are left in this table for events that have been completely processed, they could rapidly build up and cause severe performance problems.

Beware the Loop!

- Sometimes it might be necessary to want to update something about an object when any other update is done to that same object.

- Suppose that every time a Party is created in the TCA, it is desired that a Descriptive Flex Field in the HZ_PARTIES table keep track of how many times that Party has been updated.

- It would be simple to create a subscription to the Update Party event that runs some custom code to update the DFF. If proper coding standards are used, this would be accomplished by a call to the Update Party public API.

- The problem is that this in itself would fire another Update Party Business Event which would invoke the subscription again and so on, causing a loop. In this case, it would probably be best prior to calling the Update Party public API in your custom code to check to see if anything else other than the value of the DFF has changed.

Conclusion

Business Event Callouts are a flexible and powerful feature in the TCA. They allow custom processing to occur in an asynchronous manner under the dominion of the Workflow Agent. Since User Hooks are no longer available, these Business Event Callouts are the method of choice for manipulating objects as activity occurs in the TCA.

Q & A

- Questions & Answers

Dave Hebden

Protégé Software Services, Inc

Woburn,MA

Booth # 1426

www.protege.com

781 305 8100

- Thank You