Chapter 3. Application System

An Application System is any system that would like to communicate with SAP EM. As mentioned earlier, the application system can be an existing SAP system or it can be a non-SAP system as shown in Figure 11.

Application System = Any system sending events to EM

There is almost no difference in the configuration on the SAP EM system if the application system is an SAP system or not.

The real difference lies in the various options available to you when you wish to communicate with SAP EM.

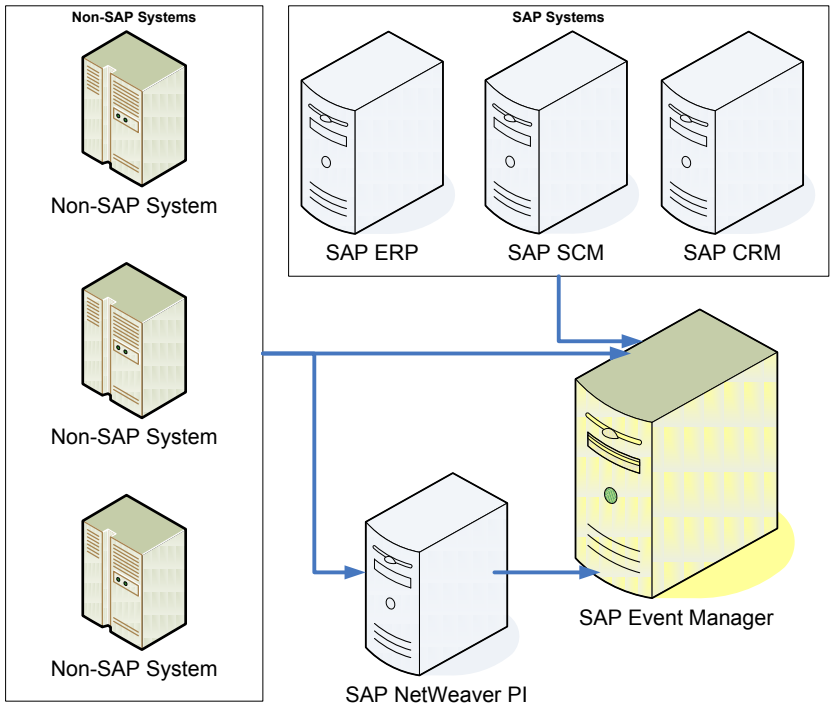

Figure 11: Application Systems

In this chapter we'll deal with the various components of an application system, starting with an SAP application system and wrapping up with the options at your disposal for non-SAP systems.

3.1 SAP Application System

An SAP system makes use of the Application Interface (AI) to communicate with SAP EM. It is described in detail in the following sections.

3.1.1 Application Interface (AI)

The application interface is SAP code by way of a SAP Basis Plug-in that provides for the integration with SAP EM.

The SAP Basis Plug-In enables your SAP system to communicate with SAP EM and is installed on a Web Application Server.

In earlier ERP releases, the SAP R/3 Plug-in provided the prerequisites needed to implement a content-based connection. E.g. BAdI implementations.

As shown in Figure 12 the AI is available in most SAP solutions including SAP CRM, SAP ERP, SAP 4.6C and SAP SCM. It uses the queued RFC and transactional RFC mechanism to communicate with SAP EM.

Note: SAP recommends using the Application Interface to communicate with SAP EM when communicating from other SAP systems.

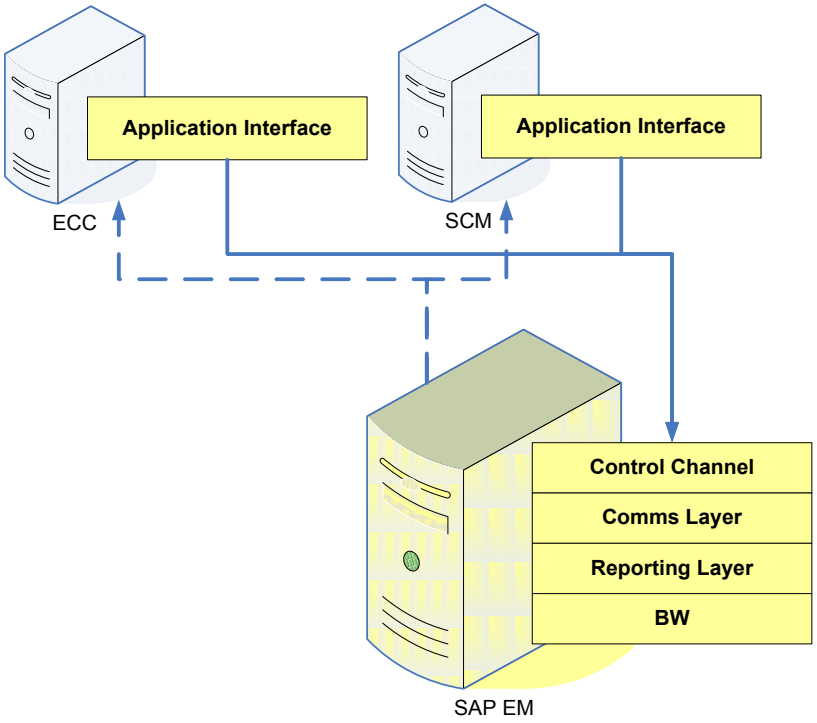

Figure 12: Application Interface (Functional View)

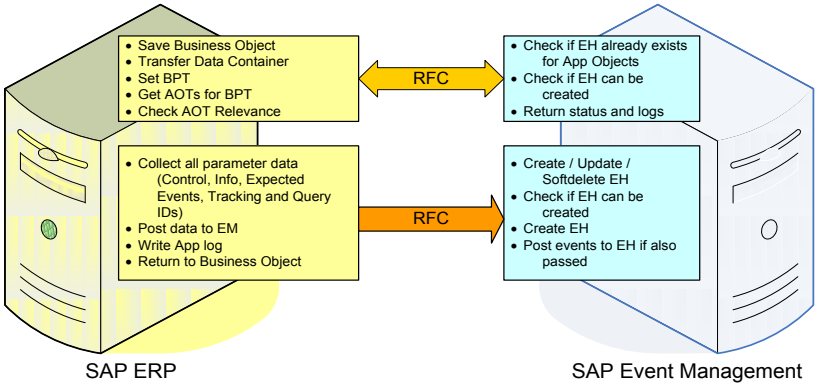

Figure 13: Application Interface (RFC Flow)

Figure 13 shows how the AI interfaces with SAP EM. Firstly it uses a Remote Function call to accomplish the following:

Once a business object (e.g. Sales Order or Delivery) is saved it performs the following functions:

- It transfers the data container to AI using function module /SAPTRX/EVENT_MGR_FILL_TABCONT. The data container has all the relevant data pertaining to the business object that was just save. As an example, if the business object was a sales order then the data container would hold all the data from the header (VBAK, VBUK, VBKD, …), the line items (VBAP, VBUP, …) and other relevant tables like VBEP, KONV, VBPA, VBFA and others
- Through function module /SAPTRX/EVENT_MGR_COMMUNICATE, it sets the Business Process Type (BPT) and retrieves the Application Object Types (AOTs) for the BPT

2 Main SAP EM AI function modules:
/SAPTRX/EVENT_MGR_FILL_TABCONT
/SAPTRX/EVENT_MGR_COMMUNICATE

For each AOT of the BPT it checks its relevance using the AOT Relevance check functionality of the AI

It then goes out to SAP EM, via tRFC, to check if an Event Handler (EH) already exists for the AOT. It also checks to see if an EH can be created as it, too, has a relevance check on SAP EM. The status and logs are returned to the AI for reporting.

Secondly, an RFC is called to actually perform the update to SAP EM once the above step has determined that it's relevant for SAP EM.

Data is collected from the BPT tables as described in the next section. This data is stored in control or info parameters.

In addition, the expected event plan is set up and tracking and query IDs are determined.

This data is then all posted to SAP EM, usually in update task (defined by configuration), via a qRFC.

Any issues or messages that are issued at this time are written to the application log and can be viewed using transaction /SAPTRX/ASAPLOG.

Note: It is possible to turn off these log entries in customizing of the AOT.

On the SAP EM side the result of this communication is simply 1 of 2 things, those being the creation, modification or deleting of an Event Handler (EH) or the creation and processing of an Event Message (EVM).

3.1.2 The detail

3 parts are implemented by SAP as part of the AI:

- Embedding of BAdI call in existing Business Object
- Implementation of the BAdI
- Customization for Application Object Type and Event Type creation

Embedding the BAdI call in existing Business Object

First a spot is needed in standard code that allows you to initiate the call to SAP EM. SAP has provided many places through the use of BAdIs, that we can leverage. All the most important business objects are already technically enabled.

Implementation of the BAdI

Once a BAdI has been found (E.g. LE_SHP_DELIVERY_PROC is the BAdI for deliveries), an implementation is created which will ultimately invoke the AI. *Note*: All the standard SAP EM implementations for SAP EM begin with /SAPTRX/* (E.g. /SAPTRX/LE_SHIPPING is the BAdI implementation for the delivery. The code invoking the Application Interface is found in method SAVE_AND_PUBLISH_DOCUMENT)

Examples of BAdI implementations in SAP ECC 6.0 are:

- /SAPTRX/BADIS_PTA
- /SAPTRX/BADI_PTA_BI
- /SAPTRX/LE_SHIPPING
- /SAPTRX/MM_INVOICE
- /SAPTRX/MM_MATDOC
- /SAPTRX/MM_PROC_PO
- /SAPTRX/PP_WOCONFIRM
- /SAPTRX/PP_WOGOODSMV
- /SAPTRX/PP_WOUPDATE
- /SAPTRX/RTI10_EHUPD
- /SAPTRX/SD_INVOICE
- /SAPTRX/SD_SALESORD
- /SAPTRX/TRA10_UPROF

The implementations for all AI BAdIs are similar and follow the same process as described in Figure 14.