

**Billing to Cash Collection Project**

**Interface Detail Design Document**

*Development ID I065 – Contact Sync*

*IB Interfaces: Siebel CRM to SAP External Contacts*

[1) Approvals 3](#_Toc285533916)

[2) Review List 3](#_Toc285533917)

[3) Related Documentation 3](#_Toc285533918)

[4) Glossary 3](#_Toc285533919)

[5) Introduction 4](#_Toc285533920)

[5.1 **Document Context** 4](#_Toc285533921)

[**5.2** **Document Structure** 4](#_Toc285533922)

[**5.3** **Scope** 4](#_Toc285533923)

[**5.4** **Out of Scope** 5](#_Toc285533924)

[**5.5** **Assumptions** 5](#_Toc285533925)

[6) External Contacts 5](#_Toc285533926)

[**6.1** **Interface Definition** 5](#_Toc285533927)

[6.1.1 Interface Purpose 5](#_Toc285533928)

[6.1.2 Context 5](#_Toc285533929)

[6.1.3 Functional View 6](#_Toc285533930)

[6.1.4 Interface Characteristics 6](#_Toc285533931)

[**6.2** **Interface Design** 6](#_Toc285533932)

[6.2.1 High Level Interface Design 6](#_Toc285533933)

[6.2.2 Data Definitions 8](#_Toc285533934)

[**6.3** **Interface Design Components** 9](#_Toc285533935)

[8.3.1 CCRMCM\_DequeueContactRequest 10](#_Toc285533936)

[8.3.2 CCRMCM\_ContactDispatcher 11](#_Toc285533937)

[8.3.3 SAPCM\_EventManager 12](#_Toc285533938)

[8.3.4 SAPCMExt\_Invoker 13](#_Toc285533939)

[8.3.5 SAPCM\_ContactStatus 15](#_Toc285533940)

[8.3.6 CCRMCM\_PropagateContactErrorStatus 15](#_Toc285533941)

[8.3.7 PL-SQL 16](#_Toc285533942)

[7) Developer Notes 17](#_Toc285533943)

[ **Code management** 17](#_Toc285533944)

[ **Deployment Process:** 17](#_Toc285533945)

[8) Issues/Workarounds: 17](#_Toc285533946)

Document History

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Change Description | Author | Date Released |
| 1.0 | Initial Draft | Albin I | Dec 15 2010 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Approvals

The individuals listed below will be required to review and approve this document.

|  |  |  |  |
| --- | --- | --- | --- |
| Role | Name | Signature | Date |
|  |  |  |  |
|  |  |  |  |

# Review List

|  |  |
| --- | --- |
| Name | Position |
|  |  |

# Related Documentation

| Ref | Title | Author | Version |
| --- | --- | --- | --- |
| I046 | I065\_ARCASH\_ Siebel to SAP Contact Management | Melodie Martinez | 1.0 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Glossary

| Abbreviation | Definition |
| --- | --- |
| CCRM | Commerce and Customer Relationship Management |
| EAI | Enterprise Application Integration |
| B2CC | Bill To Cash Collection |
| SAP | SAP System |

# Introduction

##### **Document Context**

This document is one of a set of Interface Definition Documents providing the definition and High Level Design of the interfaces required for the Siebel CRM to SAP B2CC This document details the interfaces which support the propagation of changes (creation, update and delete) of internal and external contacts from master system Siebel CRM to receipt systems (SAP B2CC ). The interface will also indicate the association and de-association of a contact to an account and the contact function changes.

##### **Document Structure**

This document describes about the requirements of an interface in terms of scope, functionality and characteristics. This document focus on what the interface does from an end-to-end perspective. Also covers the design of interface and the technical integration implementation.

##### **Scope**

The scope of this interface document defines the creation and update of internal and external contacts into SAP from SIEBEL. As Siebel is the system of record it will hold the most accurate information which needs to be reflected in SAP.

This interface defines the scope of contact propagation from SIEBEL to SAP (request component) alone.

Only the contacts with accounts having the functions as Bill payer and administrator will be propagated to SAP.

##### **Out of Scope**

* NA-

##### **Assumptions**

SAP system will ONLY subscribe to Bill Payer Contact Functions via this interface.

Siebel will only publish Contacts if the associated account is SAP-integrated as well.

# External Contacts

##### **Interface Definition**

* 1. Interface Purpose

The interface will allow keeping external contacts (the Bill payer, viewer and administrator) information in sync between the two systems without any manual re keying. As Siebel is the system of record it will hold the most accurate information which needs to be reflected in SAP.

EAI will handle the process of propagating contact data updates from Siebel to B2CC that is subscribed to receive it. Siebel will publish updates to EAI using one standard interface and EAI will transform this data to fit the interface(s) exposed by the consuming systems like SAP B2CC and manage the distribution of the data through EAI interface components.

* 1. Context

The interface will allow keeping external contacts (the Bill payer) information in sync between the two systems without any manual re keying. As Siebel is the system of record it will hold the most accurate information which needs to be reflected in SAP.

* 1. Functional View



* 1. Interface Characteristics

EAI dequeues the Contact data from MQ e\_CRM\_EAI\_027\_CON\_PUB\_RQST\_Q to which Siebel enqueues request message. The BPEL process, CCRMCM\_DequeueContactRequest dequeues the Contact data. The Contact interface is offered by EAI to Siebel to allow sync up the contact data in SAP.

The messaging style used for the External contact interface is the ‘Command Style’, where a general purpose message format is used with a standard data payload and the required functionality is specified by an ‘action code’.

##### **Interface Design**

* 1. High Level Interface Design

**Request**



NOTE: **BLUE**: Existing BPEL Components

**Response**

`

NOTE: **BLUE**: Existing BPEL Components

* 1. Data Definitions

The message specifications used in the interfaces between EAI and the external systems are defined in the API specification [6].

Latest versions of the XML schemas for the messages used in the interface are stored in the Reuters Developer Network Subversion repository (https://int.thomsonreuters.com) project called sami (https://sami.cdt.int.thomsonreuters.com). The definitions for this interface can be found in subversion at:

https://sami.cdt.int.thomsonreuters.com/svn/eai\_eai/branches/6.1/design/EAI1.0/Interfaces

##### **Interface Design Components**

|  |  |  |
| --- | --- | --- |
| **Module Name/Method** | **Module Type** | **Existing/New** |
| CCRMCM\_DequeueContactRequest | BPEL Process | Existing BPEL Component |
| CCRMCM\_ContactDispatcher | BPEL Process | Existing BPEL Component |
| SAPCM\_EventManager | BPEL Process | Existing BPEL Component |
| SAPCM\_Invoker | BPEL Process | Existing BPEL Component |
| SAPCMExt\_Invoker | BPEL Process | New BPEL Component |
| SAPCM\_ContactStatus | BPEL Process | New BPEL Component |
| CCRMCM\_PropagateContactErrorStatus | BPEL Process | New BPEL Component |

1. CCRMCM\_DequeueContactRequest

* **Purpose**

The BPEL process dequeues the User Contact request from the MQ and process for validating the data to create the contacts in SAP.

* **Program Logic**

**AS-IS Process:**

The process dequeues the message from the queue e\_CRM\_EAI\_027\_CON\_PUB\_RQST\_Q to which Siebel has enqueued. ‘e’ is the environment P- prod, B – blue, BL – Blue Lite, A – UAT, Q – System test, D – Support, C – Dev and NF- Non Functional.

This process transforms data into a canonical contact event (EAI.MSG.CM.197) and invoke CCRMCM\_ContactDispatcher.

* **DVM**

NA

* **Data Mapping (Transformation)**

NA

* **Scope:**

NA

* **Exception Handling:**

For all the scopes in CCRMCM\_DequeueContactRequest catch all types of exceptions and throwing the fault message.

Selection Failure**:**

For any Selection Failure fault exception while transformation catches the exception and throwing remote fault exception and appending the fault message to the title.

remoteFault:

For any Remote fault exception while transformation catches the exception and throwing remote fault exception and appending the fault message to the title.

runtimeFault:

For any run time fault exception while transformation catch the exception and throwing run time fault exception and appending the fault message to the title.

bindFault:

For any data binding or data validations exceptions while transformation catch the exception and throwing remote fault exception and appending the fault message to the title.

NOTE: After Fault the BPEL instance is ready for Re-initiate, Manually user/Ops team will be re-initiating after all validations done.

* **Input/output Schemas:**

EAI.MS.CCRM.194.xsd

EAI.MS.CCRM.197.xsd

* **Partner links**

**ContactEventTopic.wsdl**

**DequeueContactRequest.wsdl**

1. CCRMCM\_ContactDispatcher

* **Purpose**

This process propagates the external Contact request which has dequed from the MQ and propogates it to SAP through CCRMAM\_Dispatcher.

* **Program Logic**

The process Propogates the message to SAP through SAPCM\_EventManager

The necessary message transformations are performed .

* **DVM**

None

* **Data Mapping (Transformation)**

NA

* **Exception Handling:**

All the scope in CCRMAM\_ContactDispatcher catches all types exceptions and throws the fault message.

RemoteFault:

For all the remote fault messages captured at the catch block, the relevant message will be captured and set as Title to the faulted instance..

RuntimeFault:

For all the runtime fault messages captured at the catch block, the relevant message will be captured and set as Title to the faulted instance.

BindFault:

For all the bind fault messages captured at the catch block, the relevant message will be captured and set as Title to the faulted instance.

For the above mentioned and for any other exceptions caught, the exceptions message is customized, concatenated with the unique ContactID in the process and the submitted as the Title of the process. This gives the user an ease to search the status of the Contact Data submitted to SAP.

The user can use the same ContactID, navigate to the instance tab and search from the BPEL Console of the respective environment for the ContactID. The title will then show the status for the status.

* **Input/output Schemas:**

EAI.MSG.CM.197\_ContactEvent.xsd

EAI.MSG.CM.197\_ContactEvent.xsd

* **Partner links**

SAPCM\_EventManager.wsdl

1. SAPCM\_EventManager

* **Purpose**

Receives contact events and initiates calls to the SAP Invoker to update SAP based on the contact events. The SAPCM Contact Manager uses the EAI Event Sequencer so that multiple instances can be run concurrently whilst maintaining sequential integrity of updates that could potentially conflict with the other.

* **Program Logic:**

**As-is**

* The process receives a PublishContactEventRequest (EAI.MSG.SAP.CM.198), checks with the EAI Event Sequencer whether there is a current in-progress middleware process that might affect this contact and waits if necessary until the event sequencer notifies it that it can processed. This check is done by sending a CheckSequenceRequest (EAI.MSG.UTL.185) to the Event Sequencer and then waiting for a BasicResponse (EAI.MSG.EAI.169) before proceeding.
* The data from CCRMAM\_ContactDispatcher transformed in the interface\_buffer table schema and then sent to the table using a database adapter.
* Also the sequence checker (SC) is invoked from the SAPCM\_EventManager and SC is initiated.
* The sequence is released in Invoker only upon completion of the process and the sequence is released for the corresponding request instance.
* Although the SAPCM Contact Manager is responsible for acquiring the lock with the EventSequencer, it is not responsible for releasing it. This is done by the SAPCM Invoker after it has completed processing the contact update.

To-Be

* If the contact is having at least one account that is SAP published and with the Function “Billing Contact - Payer” then the contact will be transformed in the interface\_buffer table schema and then sent to the table using a database adapter.
* Also the sequence checker (SC) is invoked from the SAPCM\_EventManager and SC is initiated.
* The sequence is released in CCRMCM\_PropagateContactErrorStatus only upon receiving the response from SAP.
* DVM “SAPCMFunctionFilterMap” is used to filter the contacts based on the function.
* **Data Mapping (Transformations)**

Process passes the same message which has been received from SAPCM account event subscriber but with the original tracing ID replaced with a unique correlation ID generated by this process.

* **DVM:**

SAPCMFunctionFilterMap

* **Partner links:**

DequeueContact.wsdl – MQ Adapter,

SequenceChecker.wsdl.

InterfaceBuffer.wsdl –Database Adapter

* **Exception Handling:**

For All the scopes in SAPCM\_EventManager catch all types exceptions and throwing the fault message.

remoteFault:

For any Remote fault exception while transformation in Parallel flows catch the exception and throwing remote fault exception and appending the fault message to the title.

runtimeFault:

For any run time fault exception while transformation in Parallel flows catch the exception and throwing run time fault exception and appending the fault message to the title.

bindFault:

For any data binding or data validations exceptions while transformation in Parallel flows catch the exception and throwing remote fault exception and appending the fault message to the title.

NOTE: After Fault the BPEL instance is ready for Re-initiate, Manually user/Ops team will be re-initiating after all validations done.

1. SAPCMExt\_Invoker

* **Purpose**
* This process receives a general contact sync event and calls the appropriate SAP web service based on the update type.
* **Program Logic:**

The SAPCMExt\_Invoker performs logic to prepare the contact details for SAP because the logical data model required by SAP is quite different to the logical model published by Siebel.

SAPCM\_EventManager which inserts the SIEBEL data into the interface buffer table.

SAPCMExt\_Invoker polls data from SAPCMExt\_Polling\_V (View Created for Interface buffer) for all the messages with the interface\_id=’SAPCMExt’ , state set as ‘N’ or ‘R’ for every 10 seconds.

It then checks for any running instances which would affect the initiation of this instance. If found any, the process waits for the completion of the same and in the other case the process in initiated.

After the completion of the entire flow, SAPCMExt\_Invoker invokes the Sequence Releaser which releases the sequence.

The above process follows the update call to the interface buffer table. During which the state is changed to ‘C’ upon completion of process.

* **Exception Handling:**

For All the scopes in SAPCMExt\_ Invoker catch all types exceptions and throwing the fault message.

remoteFault:

For any Remote fault exception while transformation in Parallel flows catch the exception and throwing remote fault exception and appending the fault message to the title.

runtimeFault:

For any run time fault exception while transformation in Parallel flows catch the exception and throwing run time fault exception and appending the fault message to the title.

bindFault:

For any data binding or data validations exceptions while transformation in Parallel flows catch the exception and throwing remote fault exception and appending the fault message to the title.

* **Data Mapping (Transformations)**

TransformationContactToSAP.xsl

* **Partner links:**

SAPCMExt\_POLLING (DB Adapter)

(SAP Provided WSDL i.e. external wsdl)

* **Database:**

Interface\_buffer,

Interface\_Config

1. SAPCM\_ContactStatus

* **Purpose**
* This ESB process receives a general contact response from SAP web service.
* **Program Logic:**

This ESB process receives a general contact response from SAP web service and route the message to SAPCM\_PropagateContactStatus.

* **Exception Handling:**
* **Data Mapping (Transformations)**

MT\_Contact\_To\_PublishContactRequestReponse.xsl

* **Partner links:**

SAPCM\_PropagateContactStatus

* **Database:**

NA

1. CCRMCM\_PropagateContactErrorStatus

* **Purpose**
* This process receives a contact response message from SAPCM\_ContactStaus ESB process and propagate the same to SIEBEL MQ.
* **Program Logic:**
* This process receives a contact response message from SAPCM\_ContactStaus ESB process and if the response is error,transform the message to SIEBEL format and propagates the same to SIEBEL MQ.
* Update the error message to BPEL sensor table for Monitoring purpose.
* **Exception Handling:**

For All the scopes in CCRMCM\_PropagateContactErrorStatus catch all types exceptions and throwing the fault message.

remoteFault:

For any Remote fault exception while transformation in Parallel flows catch the exception and throwing remote fault exception and appending the fault message to the title.

runtimeFault:

For any run time fault exception while transformation in Parallel flows catch the exception and throwing run time fault exception and appending the fault message to the title.

bindFault:

For any data binding or data validations exceptions while transformation in Parallel flows catch the exception and throwing remote fault exception and appending the fault message to the title.

* **Data Mapping (Transformations)**

TransformationContactErrorStatus.xsl

* **Partner links:**

ContactStatusUpdate.wsdl

Queue Name: e\_EAI\_CRM\_CONT\_ERR\_RESP\_Q

* **Database:**

1. PL-SQL

**SAPCMExt\_POLLING\_V**

CREATE OR REPLACE FORCE VIEW SAPCMExt\_POLLING\_V ("ID", "CREATED", "PAYLOAD", "INTERFACE\_ID", "REFERENCE\_ID", "STATE", "LAST\_UPDATED") AS

select ib.id, ib.created, ib.payload, ib.interface\_id, ib.reference\_id, ib.state, ib.last\_updated

from eaiowner.interface\_buffer ib, eaiowner.interface\_config ic

where ic.ID = 'SAPCMExt'

and ic.state = 'ON'

and ic.IS\_AVAILABLE = 'Y'

and ib.INTERFACE\_ID = ic.ID

and ib.STATE in ('N','R')

and ib.SEND\_ATTEMPTS < ic.MAX\_SEND\_ATTEMPTS;

**SAP CM Interface Config:**

Insert into INTERFACE\_CONFIG (ID,STATE,LAST\_UPDATED,IS\_AVAILABLE,SHUTOFF\_THRESHOLD\_SECONDS,SHUTOFF\_THRESHOLD\_MESSAGES,MAX\_SEND\_ATTEMPTS) values ('SAPCMExt','ON',null,'Y',null,10,10);

**Sequencer**

INSERT INTO SEQR\_MSG\_TYPE (MESSAGE\_TYPE,CONTEXT) VALUES

('SAP.ContactROWID','SAPCMEventManager');

# Developer Notes

##### **Code management**

For Code management and usage of subversion refer the document.

##### **Deployment Process:**

The deployment process for BPEL, ESB or web services components and as well as for any other deployments like DB Scripts, Business rules configurations: refer EAI- Build and Deployment Process.doc in https://sami-crm6-eai.reutersdev.net/svn/sami-crm6-eai/trunk/dev/EAI-CRM61/R2releases/BuildDocs folder in SVN.

# Issues/Workarounds:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S.No | Summary | Status | Assigned to | Solution |
| 1 | Siebel CRM Schemas | Closed | CRM team |  |
| 2 | SAP WSDLs | Closed | SAP Team |  |