Best Practices Using the Oracle Clinical API

Bill Caulkins, Ahila Selvaraj
DBMS Consulting
04 October 2009
OCUG Tutorial Session
Session T05

Acknowledgements

- Many thanks to the OCUG for this opportunity to present for the OCUG A&CM group.
- Many thanks to the OCUG Planning and Review Committee and OCUG A&CM Focus Group Chairs for their infinite patience in receiving and expeditious review of this presentation.
- Many thanks to everyone who participated in the development of presentation.

Types of APIs in OC

- Object Naming and Validation API.
- External Application Configuration API.
- External Workflow Data Entry API.
- Data Capture API.

Object Naming and Validation API

- This API is a database package that let you to define own object-naming validation for Oracle Clinical Installation. Through this API package one can implement local naming conventions for:

1. Object Names.
2. Customer Record Formatting (CRF) document number validation.
3. Document Number derivation.
4. Invoking a scanned image.
5. CRF page tracking.

Object Naming and Validation API

- The database package OCL_CLIENT_PACK consists of two script files in the RXC_INSTALL directory.
- **ocl_client_ps.sql**
  - Container of Package specification of OCL_CLIENT_PACK.
  - Creating the public synonym.
  - Granting execute privilege to both RXCLIN_MOD and RXCLIN_READ database roles.
- **ocl_client_pb.sql**
  - Container for the function bodies for the OCL_CLIENT_PACK package.
- These database package scripts should be execute in RXC database schema account .
- When initially installing the Oracle Clinical application.
- Once after each time you modify the file.

Object Naming and Validation API

- The functions available in the package to customize validations and derivations are:
  - ValidateName.
  - ValidateDocument.
  - DeriveDocumentNumber.
- The functions, to customize for page tracking are:
  - DefaultBookPageNumber.
  - TrimPageNumber.
  - ChangePageStatus .
  - AddPhysPageNumber.

Object Naming and Validation API

ValidateName

- This function is called by the validation triggers on the naming of each of the Oracle Clinical objects.
- You can add your own validation to enforce your company naming standards as an example:
  - Maintain DVG.
  - Maintain Questions.
  - Maintain Question Groups.
  - Maintain DCMs.
  - Maintain DCIs.
  - Maintain Procedures (both Derivation and Validation).
  - Maintain Where Clauses.
  - Maintain Labs.
  - Mass Changes.

Object Naming and Validation API

ValidateDocument

- This function is called by the Validation Procedures on the document number in the CRF Log-In form and the Batch Data Load program.

DeriveDocumentNumber

- With this function you can derive CRF document numbers from the key fields entered in the Log-In form and the Batch Data Load program.

DefaultBookPageNumber

- This function assigns the correct page number to the first page of a DCI in the DCIbook.

TrimPageNumber

- This function trims the page number off its suffix according to the numbering scheme. It is called each time Oracle Clinical needs to add one unit to a page number and each time Oracle Clinical needs to assign a default page number.

Object Naming and Validation API

ChangePageStatus

- This function, sets the client-specific status of a new page, depends on the current page status, the RDCI status, and the blank flag, according to user-defined rules. It returns NULL if successful; otherwise, it returns an error message.

AddPhysPageNumber

- This function adds a number to a page number, according to the numbering scheme— when DCIs are added to a DCI book, and also when pages assigned at data entry correspond to one of the following unplanned events: the DCI is not in the DCI book; the DCI is present in the DCI book but not for the specific visit; or the subevent number is zero.

External Application Configuration API

- This API provides the interfaces with external application in the areas of:
  - Study planning.
  - Investigator management.
  - Drug supply.
- The OC tables that are part of the Interface Configuration interface with external applications will start with the prefix **OCL_.**
- The OC Interface tables can be used in the following ways with external applications.
  - Synonyms to customer tables.
  - Views on customer tables.
  - Batch/parallel maintenance of the OCL_ tables from customer applications.

External Application Configuration API

Study planning

- Study planning is the process of controlling and managing all studies within a company and across all company locations.
- Oracle Clinical requires that study planning provide:
  - A two-tier hierarchy of studies to which users can be assigned as an alternative to maintaining security by individual user assignments to studies.
  - A list of studies that are approved for definition in Oracle Clinical.

Below are the list of tables that can be used in ***Study Planning Tables***

Table	Description
OCL_STUDIES	Master list of all studies
OCL_STUDY_REGIONS	List of regions where each study is to be conducted
OCL_ORGANIZATION_UNITS	List of company departments that can support studies
OCL_PROGRAMS	Upper tier where studies can be grouped and security managed
OCL_PROJECTS	Lower tier where studies can be grouped and security managed
OCL_PROGRAM_PRODUCT_MASTERS	List of the compounds for which each program is responsible

External Application Configuration API

Drug supply

- A drug supply system interacts with Oracle Clinical in two ways:
  - At the front end of the clinical process, the drug supply system provides a list of drugs formulated for a clinical study.
  - At the back end of the process, the drug supply system takes the randomization generated in the design subsystem of Oracle Clinical and produces from it the packaged and labeled supplies for the study.

Table	Description
OCL_DOSAGE_FORMS	Look-up list of dosage forms
OCL_PRODUCT_MASTERS	List of active substances (raw chemicals) and formulated products (medications) that a clinical study can use

External Application Configuration API

Investigator management

- An investigator management system is responsible for maintaining a list of the investigators who can work on studies and the records of their assignments to studies.

Table	Description
OCL_INVESTIGATORS	List of potential and actual investigators who can work on studies
OCL_SITES	List of locations where a clinical study can be conducted
OCL_STUDY_SITES	List of the assignments of sites to a particular clinical study
OCL_STUDY_SITE_ROLES	List of which investigators have been or are responsible for each study site

External Application Configuration API

Interface Procedures

- These API procedure maintain referential integrity between Oracle Clinical and another external application.

Some of the procedures are listed below.

Procedure Name	Uses
LocalStudyDeletion.LocalDeletionOK	Prevent study deletion
LocalStudyDeletion.DeleteOCLRecords	Delete interface records
LocalStudyDeletion.DeletelocalRecords	Delete local records
LocalRegionUpdateDeletion.LocalRegDeletionOK	Prevent region deletion
LocalRegionUpdateDeletion.LocalRegionCascadeUpdate	Update region code
LocalTreatPattDeletion.LocalTreatPattDeletion	Treatment pattern deletion check

External Workflow Data Entry API

- Oracle Clinical data entry process can be integrated with external workflow and imaging systems API.
- The PL/SQL Library used by this API is **RXCLBCLI.PLL**
- This API allows to
  - Data Entry Workflow process Customization.
  - Menu Customization.
- Using **RXCLBCLI.PLL** PL/SQL Library imaging work flow can be integrated.

External Workflow Data Entry API

Enabling external systems

- To enable an external workflow system, you set the INVOKE_WORKFLOW option to Y in the OCL_STATE reference codelist.
- To enable an imaging system, you set the INVOKE_IMAGE option to Y in the OCL_STATE reference codelist.

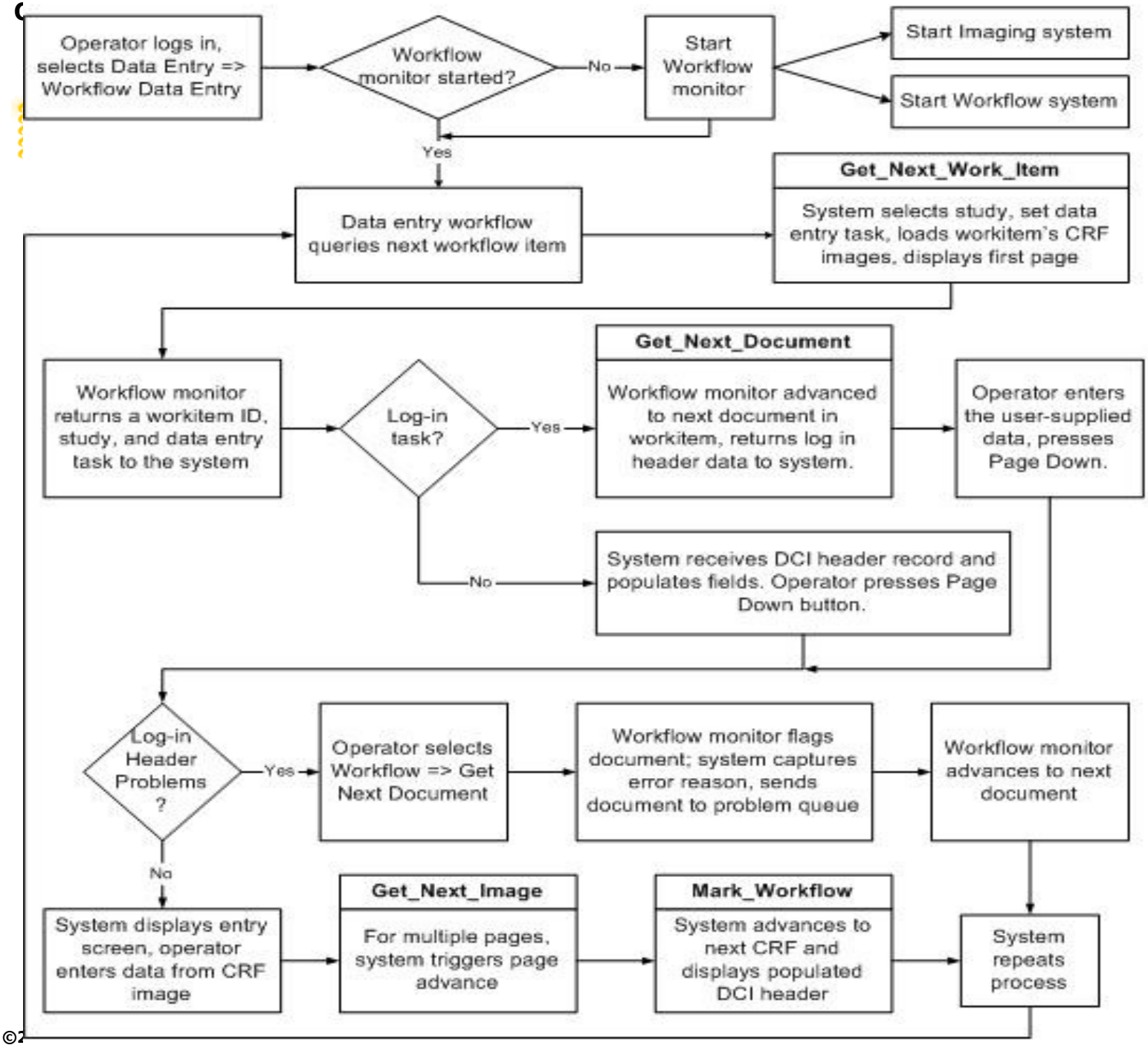
To enable these options:

1. Select Admin => Reference Codelists => Local Codelists.
2. Type OCL_STATE in the Name field, and execute a query.
3. To enable an external workflow system, scroll through the values to INVOKE_WORKFLOW. To use an imaging system, scroll through the values to INVOKE_IMAGE
4. Set the value to Y.
5. Save.

External Workflow Data Entry API

Some of the terms used in work flow are below

- A **workitem** is a collection of DCIs, also referred to as documents. All DCIs in the workitem are for the same task and the same study, but they may be for different patients
- A **document** is a single DCI within a workitem, and can be composed of one or more physical Customer Report Forms (CRF) pages.
- When you are performing data entry through the workflow system the following standard Oracle Clinical data entry options are unavailable from a data entry form:
  - changing studies.
  - changing tasks.
  - turning DCI book on.
  - entering or executing a query.
- The key change task is also not available for use in a workflow system.


Operator logs in, selects Data Entry => Workflow Data Entry
Workflow monitor started?
No
Start Workflow monitor
Start Imaging system
Start Workflow system
Yes
Data entry workflow queries next workflow item
Get_Next_Work_Item
System selects study, set data entry task, loads workitem's CRF images, displays first page
Workflow monitor returns a workitem ID, study, and data entry task to the system
Log-in task?
Yes
Get_Next_Document
Workflow monitor advanced to next document in workitem, returns log in header data to system.
Operator enters the user-supplied data, presses Page Down.
No
System receives DCI header record and populates fields. Operator presses Page Down button.
Log-in Header Problems ?
Yes
Operator selects Workflow => Get Next Document
Workflow monitor flags document; system captures error reason, sends document to problem queue
Workflow monitor advances to next document
No
System displays entry screen, operator enters data from CRF image
Get_Next_Image
For multiple pages, system triggers page advance
Mark_Workflow
System advances to next CRF and displays populated DCI header
System repeats process

External Workflow Data Entry API

Some of the Workflow API functions are

get_next_document

- The purpose of this function is to retrieve information about the next workflow work item.
- The function returns a null value if it succeeds and an error message if it fails.

mark_workflow

- This routine is called at the end of the commit process, which occurs whenever changes are committed to the database.
- The workflow function updates visit date, status, and other information from the Oracle Clinical received DCI record.

External Workflow Data Entry API

get_next_CRF_work_item

- Execution of this routine enables the following:
  - The workflow monitor starts if this module is not yet started. (The workflow monitor starts the image viewer, if necessary.)
  - The workflow monitor retrieves the next workitem from the workflow system for the specified user.
  - Oracle Clinical initializes the task and sets study context, then calls the get_ next_document function.

External Workflow Data Entry API

get_next_image

- This API function displays the image of the next page of the document.
- It is called when the submenu item Get Next Image is selected from the Workflow menu.

get_previous_image

- This API function displays the image of the previous page of the document.

External Workflow Data Entry API

Menu Customization

- Using the pl/sql library RXCLBCLI.PLL Oracle clinical menu can be customized.
- These integrating functions can be programmed in the library, using OLE, C, PL/SQL, or DDE.
- Oracle Clinical provides 5 menu stubs—Custom Program Units— that can be customized for use with the external system.
- These customized menu stubs can be used in naming and displaying the menu items, and to writing the code to fully implement the menu.

External Workflow Data Entry API

Menu Customization

- Using the library **RXCLBCLI.PLL**, there are five functions that are defined to allow you to add your own menu options and functionality to the Data Entry menu.
- The functions, are named CustomPgmUnit1 through CustomPgmUnit5.
- The CustomPgmUnit# functions are provided for each custom menu option as a shell. You can to add functionality into the data entry system.

External Workflow Data Entry API

CustomPgmUnit1

CustomPgmUnit1(*document_number, nreceived_dci_id, nError_type)*

CustomPgmUnit2

CustomPgmUnit2(*ndcf_id nError_type)*

CustomPgmUnit3

CustomPgmUnit3(*nError_type)*

CustomPgmUnit4

CustomPgmUnit4(*nError_type)*

CustomPgmUnit5

CustomPgmUnit5(*nError_type)*

Data Capture API

- The Data Capture API provides
  - API functions to update data in Oracle Clinical.
  - A stable interface to other application.
- The combination of the Data Capture API and the stable interface provides the tools to integrate Oracle Clinical with other systems.

Data Capture API (2)

- The API can be with programs with Oracle Pro*C and Microsoft Visual C++.
- The following files are available on the Oracle Clinical Web Server in the ***opa_home/oc/dcapi***
  - **dcapi.h:** Include this header file in the C files and set the header file path in your Microsoft Visual C++ project definition to the directory that contains dcapi.h.
  - **dcapi.lib:** Include this library file in the list of files used to link the program and set the library path of your Microsoft Visual C++ project definition to the directory that contains dcapi.lib.
  - **dcapi.dll:** Include this file in your system path during runtime.

Data Capture API (3)

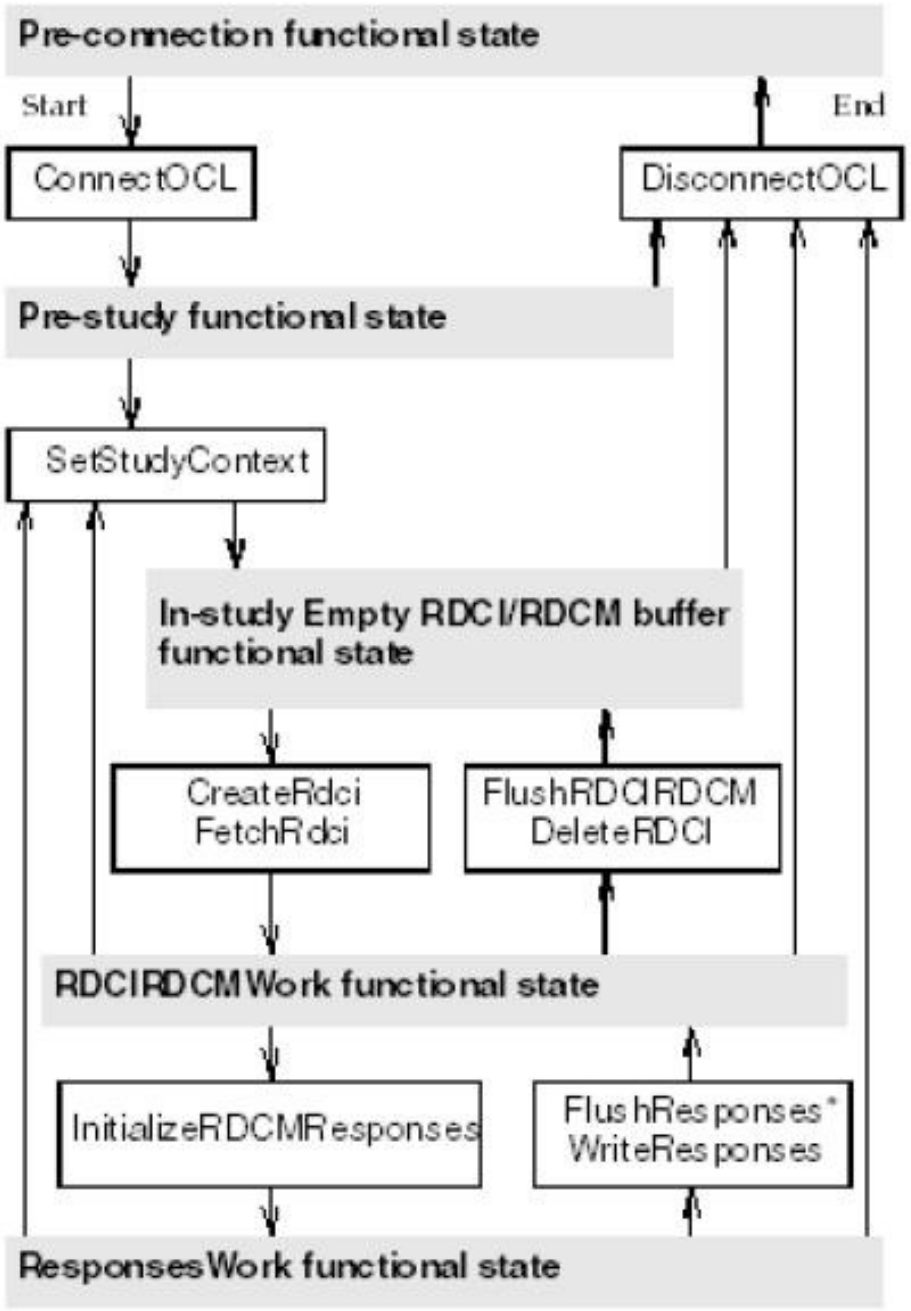
- API functions can be used to create and update the following in Oracle Clinical Data.
- Received RDCI and RDCM Log-In information.
  - Actual events.
  - Patient enrollment.
  - Data responses.
  - Data discrepancies.
  - Comment information.

Data Capture API (4)

Database roles for the Data Capture API

- The Data Capture API provides security mechanisms for assigning roles to users to determine the tasks they can perform.
- This assignment is accomplished through database roles. The roles necessary to perform tasks through the API are different from the roles in Oracle Clinical.

Database Role Name	OC Manual Name
OCLAPI_INITIAL_LOGIN	Initial Log-In
OCLAPI_KEY_CHANGES	Key Changes
OCLAPI_FIRST_PASS_ENTRY	First-Pass Entry
OCLAPI_UPDATE	Update
OCLAPI_BROWSE	Browse
OCLAPI_PATIENT_ENROLLMENT	Patient Enrollment
OCLAPI_EXECUTE_MULTIVARIATE	Execute Multivariate



Data Capture API (6)

API Functional State transition

- The Data Capture API enforces rules about when can call an API function. The API considers the data to be in one of the following states.

Pre-Connection

- This state occurs when the application is running but not connected to the Oracle Clinical database.
- The API function **ConnectOCL** connects you to the Oracle
- Clinical database.

Pre-Study

- This state occurs immediately after your program connects to the Oracle Clinical database
- The **SetStudyContext** API function provides this context information.

Data Capture API (7)

In-Study with Empty RDCI/RDCM Buffer

- This state occurs when the program has set the study context but has not called any API functions that update RDCI/RDCM data.
- ***EnrollPatient, SetActualEvent, SetPageStatus, ExecuteMultivariate, SetExternalContext*** can be used in this state.

RDCIRDCMWork

- This state occurs when the program is retrieving or updating RDCI/RDCM data
- ***WriteRdciRdcm, FlushRdciRdcm*** functions are used to save and discard the response data.
- Once all pending changes have been saved or discarded, the program can call ***EnrollPatient, SetActualEvent, SetPageStatus, and ExecuteMultivariate*** to commit data.

Data Capture API (8)

ResponsesWork

- This state occurs when your program is retrieving or updating response data.
- Once all pending changes have been saved or discarded using ***WriteResponses*** or ***FlushResponses*** the program can commit data by calling ***EnrollPatient, SetActualEvent, SetPageStatus,*** and ***ExecuteMultivariate.***
- The program can return to either the RDCIRDCMWork state or In-Study with Empty RDCIRDCM Buffer state by calling, saving, or discarding any pending changes.

Data Capture API (9)

- Some of the API functions called in transition states are below.

ConnectOCL

- Connects the user to an Oracle Clinical database in either Production or Test mode.

DisconnectOCL Can be called in:

- Pre-Study.
- In-Study with Empty RDCI/RDCM Buffer.
- RDCI/RDCM Work, but only if there are no changes pending.
- Responses Work, but only if there are no changes pending.

SetStudyContext Can be called from:

- Pre-Study
- In-Study with Empty RDCI/RDCM Buffer
- RDCI/RDCM Work, but only if there are no changes pending.
- Responses Work, but only if there are no changes pending.

Data Capture API (10)

CreateRdci Can be called in:

- In-Study with Empty RDCI/RDCM Buffer
- RDCI/RDCM Work

FetchRdci Can be called in:

- In-Study with Empty RDCI/RDCM Buffer
- RDCI/RDCM Work,

FlushRdciRdcm

- Can be called in: RDCI/RDCM Work

DeleteRdci

- Can be called in: RDCI/RDCM Work, but only if there are no changes pending.

InitializeRdcmResponses

- Can be called in: RDCI/RDCM Work

FlushResponses

- Can be called in: Responses Work.

WriteResponses

- Can be called from: Responses Work.

Data Capture API (11)

Sample Code for ConnectOCL API function

```
void connectOCL_setstudy()
{
    StudyRecord study_rec;
    double session_id ;
    short int ret_status ;
    fprintf(Log, "Connecting to the Database \n");
    fflush(Log) ;
    memset(&study_rec,0,sizeof(StudyRecord));
    ret_status = ConnectOCL("OPS$QAUSER", "QAUSER", "DCAPI_DB", OCL_PROD,
    &session_id) ;
    if (ret_status != SUCCESS)
      handle_error(ret_status);
    fprintf(Log, "Setting Study Context \n");
    fflush(Log) ;
    ret_status = SetStudyContext("DCAPI_SMOKE", &study_rec) ;
    if (ret_status != SUCCESS)
      handle_error(ret_status);
    else
      db_connection = TRUE ;
}
```

Data Capture API (12)

Sample Code for Delete DCI

```
void soft_delete()
{
    short int ret_status ;
    double received_dci_id ;
    RdciRecord rdci_rec;
    RdcmArr rdcm_arr ;
    fprintf(Log, "\n Soft deleting document: %s ----------------\n", g_doc_num) ;
    fflush(Log) ;
    connectOCL_setstudy() ;              // Connect to Oracle Clinical and Set Study Context
    memset(&rdci_rec,0,sizeof(RdciRecord));
    memset(&rdcm_arr,0,sizeof(RdcmArr));
    received_dci_id = get_received_dci_id() ;
    ret_status = FetchRdci(received_dci_id, TRUE, KEY_CHANGES, &rdci_rec, &rdcm_arr);
    if (ret_status != SUCCESS)
          handle_error(ret_status);
    ret_status = DeleteRdci() ;
    if (ret_status != SUCCESS)
          handle_error(ret_status);

    disconnectOCL() ;          // Disconnect from the database

    fprintf(Log, " Document Soft Deleted  ---------------------------\n");
    fflush(Log) ;
}
```

Question and Answers

Ahila Selvaraj
aselveraj@clinicalserver.com
+1-732-372-9976

Bill Caulkins
Bill.caulkins@clinicalserver.com
+1-919-323-0598

Biographies

Ahila Selvaraj, Senior OHS Developer, DBMS Consulting, Inc.

- Ahila is a Senior OHS Developer for DBMS Consulting, specializing in integration of OC to other OHS systems, with over 10 years of PL/SQL experience.

Bill Caulkins, Director, Global Application Development and Deployment, DBMS Consulting, Inc.

- Bill has worked with Oracle databases since 1987 as an application developer, database administrator, systems analyst, and project manager. He has worked in the pharmaceutical industry since 1991, primarily in the clinical trials arena. His interests and areas of expertise include information sharing, data integration, performance tuning, and getting the most out of PL/SQL. He enjoys implementing solutions which enable people to focus more of their time on business challenges, and less of their time on technology.