Composite Application Development Cookbook

Applies to:
SAP NetWeaver 7.0(04s) / SAP NetWeaver CE 7.1

Summary
This document explains the process of developing composite applications step-by-step from specification till the packaging phase referring the best practices with relevant examples from real composite development projects. It also provides the links to relevant resources and documents for composite development from SDN and SAP Help. This document is a one-stop guide for SAP partners and customers while developing composite applications on SAP NetWeaver platform.

Author: Dipankar Saha
Company: IBM India
Created on: 20 December 2007

Author Bio
Dipankar Saha is working in IBM India as Advisory System Analyst on architecture and development of composite applications using CAF, GP, Web Dynpro and SAP xMII. Previously he worked for SAP Labs India and was involved in development of SAP xMII 12.0 and ISA95 business package and various other Adaptive Manufacturing composite applications. Dipankar has about 6 years of experience in enterprise application software development of which 5 years in SAP NetWeaver domain. Dipankar is a SAP certified Enterprise Architect
# Table of Contents

1. Introduction .................................................................................................................. 4
2. Specifying Composite Application .............................................................................. 4
   2.1 Specify the Business Problem .................................................................................. 4
   2.2 Define the Business Process & Process User Roles ............................................... 4
   2.3 Define the Process Steps ....................................................................................... 5
   2.4 Define the User Interfaces ..................................................................................... 5
3. Designing Composite Application .............................................................................. 6
   3.1 Service Identification ............................................................................................ 6
   3.2 Service Decomposition .......................................................................................... 6
   3.3 Service Mapping .................................................................................................... 7
   3.4 Defining the Business Objects ............................................................................... 8
   3.4 Designing the Process .......................................................................................... 9
4. Architecture of the Composite Application .................................................................. 10
5. Developing Composite Application ............................................................................ 11
   5.1 Setting up the Development Infrastructure ......................................................... 11
      5.1.1 Using Local Development Components ....................................................... 11
      5.1.2 Setting up NWIDI ........................................................................................ 11
   5.2 Developing the Service Layer ............................................................................... 12
      5.2.1 Reusing CAF Core Projects .......................................................................... 12
      5.2.2 Developing the Business Objects/Entity ....................................................... 20
         5.2.2.1 Modeling Business Objects ........................................................................ 20
         5.2.2.2 Specifying Language Dependency in CAF Core ...................................... 20
         5.2.2.3 Defining User/Application Messages ....................................................... 22
      5.2.3 Importing the External Services ..................................................................... 22
         5.2.3.1 Using the UDDI Registry ........................................................................... 23
         5.2.3.2 Integrating SAP BI with CAF Core ........................................................... 23
         5.2.3.3 Integrating Knowledge Management with CAF Core ............................... 23
      5.2.4 Developing the Application Service ............................................................... 23
         5.2.4.1 Defining Input and Output Messages for Application Service Operations ... 24
         5.2.4.2 Exposing Application Services as Web Services .................................... 25
         5.2.4.3 Defining the Service Security .................................................................... 25
         5.2.4.4 Service Eventing and Notification ............................................................ 25
   5.3 Developing the User Interface Layer .................................................................... 25
      5.3.1 Selecting the UI technology for composite user interface development .......... 25
      5.3.2 Web Dynpro Java as UI for composite applications ...................................... 26
         5.3.2.1 Creating the Model ................................................................................... 26
         5.3.2.2 Implementing Value Help ......................................................................... 26
         5.3.2.3 Integrating Adobe Interactive Form in Web Dynpro Java ....................... 26
         5.3.2.4 Implementing the Web Dynpro Java Callable Object ................................ 27
1. Introduction

Composite Applications are a new breed of business applications which are developed on existing assets and solutions, having their own lifecycle and extend the functionality of the existing assets. SAP NetWeaver provides a framework called the Composite Application Framework or Composition Environment which supports the development of composite applications on SAP NetWeaver platform based on SAP assets like SAP ERP, SAP CRM, etc as well as non-SAP assets available in the landscape. In this document we will discuss how to develop a composite application in SAP NetWeaver platform starting from the specification till the packaging phase.

2. Specifying Composite Application

Specification is the first step in developing a new composite application. While specifying a composite application the required functionality and features should be documented which will help in identifying the technical components and assets to be used for the development. To start with it should be remembered that a composite application is always based on a specific business process. So the definition of a composite application starts with defining a business process.

The following steps should be followed while specifying a composite application:

2.1 Specify the Business Problem

Describe the business problem in detail which the composite application will address. It is good to state the criticality of the business problem also so that it may help in understanding the demand of the solution in market. Also identify the target users and the business benefits achieved by the composite application.

2.2 Define the Business Process & Process User Roles

Composite application is typically a business process. In this step business process which will address the issue should be described at a high level. A business process diagram is useful to understand the scenario. An example of the business process diagram is as below:

**Employee Travel Request Management**

- **Employee**
  - Create Flight Booking Request
- **Manager**
  - Approve/Reject Booking Request
- **TravelAgent**
  - Book Flight & Notify Employee

The business process diagram will give a basic idea of the process and process roles. Each high level step can be broken down into granular sub-steps which will be done in later sections. Also define the user roles for business process for each process step from the business process diagram.
2.3 Define the Process Steps

The business process described in section 2.2 should be decomposed into granular steps. The process steps should be depicted in a detailed business process flow diagram as below:

![Business Process Flow Diagram](image)

Also describe each of the process steps in the following table format:

<table>
<thead>
<tr>
<th>Step No</th>
<th>Step Title</th>
<th>Step Description</th>
<th>Interactive (I)</th>
<th>Mandatory</th>
<th>Due date handling required</th>
<th>Role</th>
<th>Exits</th>
<th>Next Step</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Search Flight</td>
<td>Employee enter the search criteria and submit search</td>
<td>I</td>
<td>Y</td>
<td>N</td>
<td>Employee</td>
<td>Default</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>Create Flight Booking Request</td>
<td>Employee selects a flight from the list and submit booking request</td>
<td>I</td>
<td>Y</td>
<td>N</td>
<td>Employee</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Decide On Booking Request</td>
<td>Manager decides on the Request</td>
<td>I</td>
<td>Y</td>
<td>Y (within 24 hours)</td>
<td>Manager</td>
<td>Approved</td>
<td>4,5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Rejected</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Receive Notification</td>
<td>Employee receives decision</td>
<td>I</td>
<td>Y</td>
<td>N</td>
<td>Employee</td>
<td>Default</td>
<td>end</td>
</tr>
<tr>
<td>5</td>
<td>Book Flight</td>
<td>Clerk updates original order</td>
<td>I</td>
<td>Y</td>
<td>Y (within 24 hours)</td>
<td>Clerk</td>
<td>Default</td>
<td>end</td>
</tr>
</tbody>
</table>

The above table will help in understanding the detail activity for each process step and the process flow information like due date handling, subsequent step and whether a step is interactive (UI based) or background action.

2.4 Define the User Interfaces

The user interfaces for each interactive process step should be specified using screen mock-ups. This will help while designing and developing the user interface.

The above steps will specify the composite application features and functions at a high level and should be typically performed by the Business Analysts or the Business Process Experts.

The Guidelines for Specifying Composite Applications is a good resource provided by SAP to capture the specification of a composite application to be developed.
3. Designing Composite Application

After the specification of the composite application is complete the design phase is started. In the next steps the process is decomposed into granular blocks and the services underlying are identified.

3.1 Service Identification

Composite applications are based on services and follow SOA principles. Each process or process step is based on a service. Identify the coarse-grained services underlying each process step. For example Create Travel Booking Request is a service consumed by the first step of the composite application. These are the composite services which provide complete business functionality. List down all the identified services in a table like below:

<table>
<thead>
<tr>
<th>Service Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CreateTravelBookingRequest</td>
<td>Creates a flight booking request</td>
</tr>
<tr>
<td>ApproveTravelBookingRequest</td>
<td>Approves or rejects a travel booking request</td>
</tr>
<tr>
<td>CreateTravelBooking</td>
<td>Creates a flight booking based on the request</td>
</tr>
</tbody>
</table>

3.2 Service Decomposition

In the next step decompose the composite services identified in the previous step into granular atomic services if possible. An identified service may perform a set of activities which are executed by a series of atomic services. List down all the atomic services for each of the composite services identified in the above step in a table as below. Also specify the input, output and exception handling logic for each of the services:
<table>
<thead>
<tr>
<th>Composite Service</th>
<th>Atomic Services</th>
<th>Description</th>
<th>Input</th>
<th>Output</th>
<th>Exception Handling</th>
</tr>
</thead>
<tbody>
<tr>
<td>CreateTravelBookingRequest</td>
<td>SearchFlight</td>
<td>Search Flight</td>
<td>Start date, end date</td>
<td>Flight List</td>
<td>Start date earlier than end date</td>
</tr>
<tr>
<td>CheckEmployeeEligibility</td>
<td></td>
<td>Checks the employee eligibility for the selected flight</td>
<td>employeeID</td>
<td>Flight class</td>
<td>Employee ID is valid</td>
</tr>
<tr>
<td>CreateTravelBookingRequest</td>
<td></td>
<td>Creates a booking request</td>
<td>Flight ID, dates, employee ID, comments</td>
<td>Booking ID</td>
<td>Flight ID, employee ID and dates are valid</td>
</tr>
<tr>
<td>SubmitTravelBookingRequest</td>
<td></td>
<td>Submit the booking request for approval</td>
<td>Booking ID</td>
<td>Status, approver</td>
<td>Booking ID is valid</td>
</tr>
<tr>
<td>ApproveTravelBookingRequest</td>
<td>ReviewTravelBookingRequest</td>
<td>Reviews the booking request</td>
<td>Booking ID</td>
<td>Booking details</td>
<td>Booking ID is valid</td>
</tr>
<tr>
<td>ApproveTravelBookingRequest</td>
<td></td>
<td>Approves or rejects the travel request</td>
<td>Booking ID</td>
<td>Confirmation</td>
<td>Booking ID is valid, status is Created</td>
</tr>
<tr>
<td>CreateTravelBooking</td>
<td>BookFlightForEmployee</td>
<td>Books a flight for the employee</td>
<td>Booking ID</td>
<td>Booking Confirmation</td>
<td>Booking ID is valid, status is Approved</td>
</tr>
<tr>
<td>UpdateBookingRequest</td>
<td></td>
<td>Updates the booking request with booking details</td>
<td>Booking ID, flight confirmation</td>
<td>Confirmation, status</td>
<td>Booking ID is valid, status is Approved</td>
</tr>
<tr>
<td>CreateInvoice</td>
<td></td>
<td>Creates invoice for the flight booking</td>
<td>Booking ID, invoice details</td>
<td>Invoice number</td>
<td>Booking ID is valid, status is Booked</td>
</tr>
<tr>
<td>NotifyEmployee</td>
<td></td>
<td>Notifies the employee</td>
<td>Booking ID</td>
<td>Confirmation</td>
<td>Booking ID is valid, status is Booked</td>
</tr>
</tbody>
</table>

### 3.3 Service Mapping

In the next step map each atomic service identified in section 3.2 to services from existing assets wherever available as below. Existing assets can be the enterprise systems like SAP ERP, SAP CRM or legacy, etc. Wherever a corresponding service is not available from any existing asset it should be provided by the composite application i.e. the services developed in the composite application. You can use the SDN ES WorkPlace or the Service Registry installed in your local environment to find out the Enterprise Services available. Refer section 5.2.3.1 to setup and use the UDDI registry.
3.4 Defining the Business Objects

Define the business objects used in the process. Each service is provided by a Business Object i.e. an atomic service is an operation of a business object. Some business objects are available in the backend while some are custom business objects to be modelled in the composite layer. The source of the business objects should be specified along with the persistency. Also the attributes of the business objects should be defined here.

<table>
<thead>
<tr>
<th>Business Object/Entity</th>
<th>Description</th>
<th>Attributes</th>
<th>Datatype</th>
<th>Custom findBy Methods</th>
<th>Related BO/Entity</th>
<th>Association Type</th>
<th>Persistency</th>
<th>Backend</th>
</tr>
</thead>
<tbody>
<tr>
<td>Employee</td>
<td>Business Object for employee</td>
<td>employeeID</td>
<td>String</td>
<td>findByEmployeeID</td>
<td>BookingRequest</td>
<td>Composition</td>
<td>Remote</td>
<td>SAP ECC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>employeeName</td>
<td>String</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>employeeDepartment</td>
<td>String</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BookingRequest</td>
<td>Business Object for travel booking request</td>
<td>requestID</td>
<td>String</td>
<td>findByRequestID</td>
<td></td>
<td></td>
<td>Local</td>
<td>NA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>requestText</td>
<td>String</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>requestType</td>
<td>String</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>flightNumber</td>
<td>String</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>startDate</td>
<td>DateTime</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>endDate</td>
<td>DateTime</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>status</td>
<td>String</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The Business Objects defined above are the Entity Services in NetWeaver 7.0 and Business Objects in NetWeaver CE. The related Business Objects/Entity has to be also specified here along with the association type. For details on association and composition of BOs in NetWeaver CE refer the following SDN blog: Modeling Business Objects in NetWeaver CE CAF Core

It is recommended not to model the remote business objects/entity in the composite layer. For business objects/entities which exist in the backend/existing assets the services should be directly consumed in the composite layer as external services. Modeling remote business objects/entities in composite layer will increase the complexity and some scenarios may not be supported.

Also the Business Objects/Entity should be represented in an E-R diagram as below:
### 3.4 Designing the Process

The process steps identified in section 2.3 along with the services and user interfaces identified in section 2.4 defines the composite process which will be developed using Guided Procedure. Specify the process blocks along with the actions and callable objects to be developed in Guided Procedure in the table format as below:

<table>
<thead>
<tr>
<th>Block Name</th>
<th>Block Type</th>
<th>Action</th>
<th>Callable Object</th>
<th>CO Type</th>
<th>ResultState</th>
<th>Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>CreateTravelRequest</td>
<td>Sequential Block</td>
<td>CreateTravelRequestAC</td>
<td>CreateTravelRequestCO</td>
<td>Web Dynpro Java (GP Interface)</td>
<td>Submit</td>
<td>ApproveTravelRequestAC</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Cancel</td>
<td>TerminateProcessAC</td>
</tr>
<tr>
<td>TerminateProcessAC</td>
<td></td>
<td>TerminateProcessCO</td>
<td>Terminate Process (Process Control)</td>
<td>Success</td>
<td>End</td>
<td></td>
</tr>
<tr>
<td>ApproveTravelRequest</td>
<td>Sequential Block</td>
<td>ApproveTravelRequestAC</td>
<td>ApproveTravelRequestCO</td>
<td>Web Dynpro Java (GP Interface)</td>
<td>Approve</td>
<td>CreateFlightBookingAC</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Reject</td>
<td>SendRejectionMailAC</td>
</tr>
<tr>
<td>SendRejectionMailAC</td>
<td></td>
<td>SendRejectionMailCO</td>
<td>Send Notification</td>
<td>Success</td>
<td>TerminateProcessAC</td>
<td></td>
</tr>
<tr>
<td>CreateFlightBooking</td>
<td>Sequential Block</td>
<td>CreateFlightBookingAC</td>
<td>CreateFlightBookingCO</td>
<td>Web Dynpro Java (GP Interface)</td>
<td>Success</td>
<td>NotifyBookingDetailsAC</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Failure</td>
<td>CreateTravelRequestAC</td>
</tr>
<tr>
<td>NotifyBookingDetailsAC</td>
<td></td>
<td>NotifyBookingDetailsCO</td>
<td>Send Notification</td>
<td>Success</td>
<td>No Target</td>
<td></td>
</tr>
</tbody>
</table>
4. Architecture of the Composite Application

A composite application can have the following components:

- CAF Core as composite service layer
- Web Dynpro/Visual Composer/Adobe Interactive Forms as user interface layer
- Guided Procedure as process layer

The following diagram explains the architecture of a composite application in SAP NetWeaver platform.

The backend systems are the enterprise systems like SAP ERP, SAP CRM, SAP BI, etc as well as non-SAP and legacy systems which provide existing business functionality as services which are consumed by the composite layer i.e. CAF Core and new composite service are developed over them. The composite services are consumed by the user interfaces which are part of the actions or process steps in the Guided Procedure. It is always a good practice to consume the backend service in the CAF layer only and not directly in the user interface layer.
5. Developing Composite Application

The best practices and steps for development of composite applications are discussed below.

5.1 Setting up the Development Infrastructure

5.1.1 Using Local Development Components

The development of SAP NetWeaver based composite applications can be done locally or using NWDI (NetWeaver Development Infrastructure) in a shared environment. Though it is always recommended to use NWDI for any productive development on SAP NetWeaver platform still some may opt to use the local development option for various reasons. In local development the Development Components are created locally and the sources are saved in the local machine where the application is developed. Later the local DCs can also be migrated to NWDI. Refer the following document to develop without using NWDI.

SAP NetWeaver Composition Environment: A Software Lifecycle Management Scenario without NWDI

To convert the local CAF DCs to NWDI DC refer the following document:

How To Convert Local CAF DC into NWDI DCs
How to Migrate Developed Software Components to a New Release in NWDI
Migrating LocalDCs onto JDI/NWDI Track

For local development of Web Dynpro projects to be used as callable objects in Guided Procedure the dependent DCs (which provides the GP interface) needs to be setup in the local system. Refer this document to set up the local development environment for Web Dynpro Java callable object development using GP Interface:

SAP Network Blog: Working with the APIs of CAF Guided Procedures: NWDI or Local Development?

5.1.2 Setting up NWDI

NWDI or NetWeaver Development Infrastructure gives an end-to-end framework with tools and methodology to develop any NetWeaver based applications. But setting up and administering NWDI needs considerable effort. Refer the following documents on NWDI installation, configuration and management:

NWDI Overview and Guidelines
Guide to Develop Applications using NWDI - Part 1
Guide to Develop Applications using NWDI - Part 2
Best Practices for Running the NWDI
Best Practices for NWDI: Track design for ongoing development
Installation Cookbook: Configure CAF / GP on NWDI 7.00
How To Install NWDI 2004s for CAF & GF Development
How to Setup SAP NetWeaver Development Infrastructure (NWDI) Permissions and Roles
Developing CAF Applications in the SAP NetWeaver Development Infrastructure
How To Develop a CAF Application Using NWDI
NetWeaver Developers Guide 2004s: The Web Dynpro Project in the NWDI
SAP NetWeaver Development Infrastructure (NWDI) and DevStudio
To set up NWDI for NetWeaver CE refer the following document:

How to Setup an NWDI Track for Composition Environment Development
Developing with CAF in Team-Oriented Environment

Using the Development and Production Infrastructure

5.2 Developing the Service Layer

The service layer in a composite application is developed using the CAF Core. CAF Core is the composite business logic layer where the Enterprise Services from backend systems are consumed and additional business logic for the composite application are developed.

5.2.1 Reusing CAF Core Projects

NetWeaver 7.0 (04s)

CAF Core projects can be reused in other CAF projects using the DC usage. Create public part in the CAF Core project which will be reused. To create a public part open the Development Configuration perspective in NetWeaver Developer Studio and select the ejbModule project. Select Public Parts node under DC metadata and click on new Public Part.
Specify a name of the Public Part and click on Next.

Select java Package Tree in the next screen and the package hierarchy to be reused. Click on Finish.

A new public part has been created for the ejbModule project.
Build the CAF project in which the public part is created.

In the consumer CAF Core project select the ejbModule project and Used DC node under DC MetaData. Click on Add Used DC.

Select the Public Part defined in the earlier steps. Select the Entities node, dependency type Design Time, Build Time and Run Time and "weak" as shown below and click on Finish.
A new DC reference will be added to the consumer project.

After adding the public part the application services, data types and entity services of the provider CAF project can be reused in the consumer CAF project. This may be helpful in modularizing the CAF development and reusing a CAF projects artifacts (entity, datatypes, application services) in multiple projects.

To access the services of the provider CAF project use the EJB interface as below:
getSAforCustomer(com.ibm.consumer.service.appsrv.datatypes.GetSAForCustomerRequestMessage)

    retVal = null;
GetSAForCustomerResponseMessage getSAForCustomerResponseMessage = new
GetSAForCustomerResponseMessage();
com.ibm.consumer.service.appsrv.datatypes.ReturnMessage returnMessage = new
com.ibm.consumer.service.appsrv.datatypes.ReturnMessage();
ArrayList salesAreaResultList = new ArrayList();
FindSalesAreaLocal provider_findSalesAreaLocal = null;
try{
    // initialize the context
    Context ctx = new InitialContext();

    // EJB lookup by JNDI reference
    FindSalesAreaLocalHome provider_findSalesAreaLocalHome = (FindSalesAreaLocalHome) ctx.lookup("java:comp/env/ejb/FindSalesAreaBean");

    // create the EJB interface
    provider_findSalesAreaLocal = provider_findSalesAreaLocalHome.create();

    // initialize the provider operation’s request message
    ReadSAforCustomerRequestMessage readSAforCustomerRequestMessage = new
    ReadSAforCustomerRequestMessage();

    // setting provider request parameter by consumer request
    readSAforCustomerRequestMessage.setCustNo(getSAForCustomerRequest.getCustNo());

    // initialize the provider operation’s response message
    ReadSAforCustomerResponseMessage readSAforCustomerResponseMessage = new
    ReadSAforCustomerResponseMessage();

    //call the provider service operation
    readSAforCustomerResponseMessage = provider_findSalesAreaLocal.readSAforCustomer(readSAforCustomerRequestMessage);
    } catch (javax.ejb.CreateException e) {
        returnMessage.setType("E");
        returnMessage.setLogMessage(e.getMessage());
    } catch (NamingException e) {
        returnMessage.setType("E");
        returnMessage.setLogMessage(e.getMessage());
    } catch (ClassCastException e) {
        returnMessage.setType("E");
        returnMessage.setLogMessage(e.getMessage());
    }
    retval = getSAForCustomerResponseMessage;
//@custom code end -
After writing the code, generation and build the EJB reference and the JNDI lookup name need to be added in the ejb-jar.xml and ejb-j2ee-engine.xml files respectively of the EJB project. This activity should be done after build and just before deployment only as the reference will go off once a new build is done.

Open the ejb-jar.xml file present in the ejbmodule project under the META-INF folder. Add the reference of the application service bean of the provider CAF project and save.
Open the ejb-j2ee-engine.xml and add th in the ejbmodule project under the META-INF folder. Add the reference of the application service bean of the provider CAF project and save.

Add the JNDI reference in the ejb-j2ee-engine.xml also present in the META-INF folder of the ejbmodule project. Specify a JNDI lookup name. The JNDI lookup name should be of the following pattern:

/vendor/CAF project name/Application service Name
**NetWeaver CE 7.1**

Reusing CAF Core projects in NetWeaver CE 7.1 is much simpler compared to NetWeaver 7.0. In NetWeaver CE 7.1 a new feature called “Reused Project” has been added. Any CAF Core project can be added as a dependent of another CAF Core project. All objects in the dependent CAF Core project are available in the parent CAF Core project’s design time.

Refer the following document for more details:
[Reuse Business Objects and Override Operations of a Business Object](http://sdn.sap.com)
5.2.2 Developing the Business Objects/Entity

5.2.2.1 Modeling Business Objects

Business Objects or Entity Services represent data objects used in the composite application. Based on the business objects defined in section 3.4 the entity services or BO has to be created in CAF Core. The following points should be considered while creating the Business Object/Entity:

If a Business Object/Entity is defined of type remote persistency then an external service operations has to be mapped with the operations of the Business Object/Entity.

Define custom key for a local Business Object/Entity if required. For remote Business Object/Entity custom key is mandatory.

Add the custom findBy operations for the Business Object/Entity as required. Refer the following document for more details on using the query operations of the Business Object/Entity: Filtering Business Object Instances in Application Service

If Permission Check is enabled for Business Object/Entity the security principal has to be set up. Refer section 5.2.4.3 for more details.

5.2.2.2 Specifying Language Dependency in CAF Core

Text defined as attributes of Business Objects/Entities in CAF Core sometimes need to be displayed to the user in the user's logon language. To define language dependent attribute of Business Object/Entity Service (which are of type shortText or String) select the Language Dependent property to true for the attribute.
Get the user language by the following code:

LanguageCallback lc = new LanguageCallback();
Locale locale = lc.getLocale();
String userlanguage = locale.getLanguage();

The following code snippets set the attribute value along with the corresponding language using two different methods:

//get the Entity Local Interface
DepartmentServiceLocal deptService = this.getEmployeeService();

//create new Department Entity
Department dept = deptService.create(deptID);
//create a new hashmap
Map deptMap = new HashMap();

//put the language specific text with the language key in the hashmap
deptMap.put(userlanguage, deptNameEN);

//set the hashmap as the language specific attribute with the language key
dep.setDepartmentNameMap(deptMap);

In the first option all the attribute values are set and then set the language. Otherwise as in second option for specific attribute set the attribute values along with corresponding languages in a hashmap.

Following code illustrates how to read the language dependent attribute:

//get the Entity Local Interface
DepartmentServiceLocal empService = this.getDepartmentService();

//read the Entity
Department dept = deptService.readByCustomKeys(deptID);

//get the language dependent attribute hashmap
HashMap deptMap = (HashMap) dept.getDepartmentNameMap();

//get the attribute value for specific language key
String deptNameEN = (String) deptMap.get(userlanguage);
5.2.2.3 Defining User/Application Messages

Application or user messages used in the composite application can be stored in the CAF layer. To define user messages in CAF Core use a Business Object/Entity Service with attributes message id and message text.

```java
//get UserMessages local interface
UserMessagesServiceLocal messageService = this.getUserMessagesService();

//create a new UserMessages Entity
UserMessages message = messageService.create(messageID);

//read the message by messageID
UserMessages message = messageService.readByCustomKeys(messageID);
String messageText = message.getMessageText();
```

Instead of hard-coding application messages in the application code use the Entity/Business Object to retrieve the messages and by message ID.

Define the messageText attribute of the Business Object/Entity as language dependent so that the user messages can be maintained in different languages. To maintain the messages in the Business Object/Entity create an application service.

5.2.3 Importing the External Services

The external services are the services from the backend systems and existing assets which are leveraged by the composite application. The services from the backend systems which are identified in section 3.3 should be imported as external service in CAF Core. All backend services should be imported into CAF Core instead of consuming them directly from the UI layer. Even if an external service can be consumed directly by the UI without adding any custom logic it should be imported into CAF Core and the application service will be a wrapper over the external service which will be consumed in the UI layer. This will enable backend abstraction for the composite application.

Refer the following documents for importing RFC and Web Services as external services in CAF Core:

- Integrating a Web Service in a Composite Application
- Using RFCs as External Services
5.2.3.1 Using the UDDI Registry

The external services can be imported from the Services Registry if available. A UDDI registry is available along with NetWeaver CE 7.1 which can be configured and web services from backend systems can be published in it. Refer the following blog to configure and publish enterprise services in the NetWeaver CE 7.1 UDDI registry:

Configuring & Publishing Enterprise Services from SAP Business Suite system into NWCE (NetWeaver Composition Environment) Services Registry

Configuring the Services Registry

Refer the following blog for importing external services in NetWeaver CE from the services registry and using them in CAF Core:

Using External Service in NetWeaver CE CAF Core

CAF services can also be published to the UDDI registry directly from Composition Environment. Refer the following document for more details:

Publishing Web Services in Service Registry with CAF

5.2.3.2 Integrating SAP BI with CAF Core

SAP BI services can also be used in CAF Core projects. The RFC/BAPI and the Web Services from SAP BI system can be imported in CAF Core as external services. Also the BI queries can be accessed from CAF using the BI-Java SDK. Refer the following documents for more details on CAF BI integration using BI-Java SDK:

How To… Use the BI Java SDK in a J2EE Application
Build Custom Analytical Applications with the New Business Intelligence (BI) Java SDK
CAF Core and SAP Business Information Warehouse Integration

5.2.3.3 Integrating Knowledge Management with CAF Core

KM data types and services can be integrated into CAF Core if required. Refer the following documents for more details:

Integrating CAF and KM
Integrating KM Documents in CAF
Integrating KM with CAF UI Patterns

5.2.4 Developing the Application Service

The application services are the composite services which provide the business logic of the composite application. The application services are based on the external services and business objects/entities. The operations of the applications services are identified in section 3.1 and section 3.2. Define the dependency of the entity/business objects and the external services on the corresponding application services. One of the best practices to access the external services from the application (composite) service operations is to import all the external services in a separate CAF Core project and create application services in it which serve as the access methods for external service operations. These custom external service access methods can be called from the composite service operations implemented in another CAF Core project. Accessing application service operations from one CAF Core project to another can be achieved by reusing CAF projects as explained in section 4.2.1. This will create another level of backend abstraction in the composite service layer. Refer the following SAP Help documentation on encapsulating external services in application service operations:

Encapsulating External Services as Application Services
The Business Objects/Entities should be accessed inside the implementation of the operations of the application service. While calling the update method of the Business Object/Entity from the application service operation it may be required to lock the record which is being updated. Refer the following SAP Help documentation for more details on locking Business Object/Entity: [Locking Entity Services](#)

Transaction handling can be implemented in application service operations. Refer the following SAP Help documentation for more details: [Transaction Support](#)

### 5.2.4.1 Defining Input and Output Messages for Application Service Operations

Instead of using discrete parameters or Entities in Application Service operations’ input and output create message data structures and use them in the operations’ interface. This conforms to the web service standards where each operation must have an input and an output message. Also it may also help while passing multiple parameters in the operation output.

Add the relevant attributes in the data structure and set the collection type (cardinality) list or none as required. Use the following naming convention while defining the messages:

- `<operationname>RequestMessage` – input message type for operation
- `<operationname>ResponseMessage` – output message type for operation

In this way each operation of the application service will have exactly one input parameter (complex data structure). Also add a log structure to the Response/output message which will be populated with the application log details.

Do not use any parameter with the name return in the application service operations. This may lead to a problem while using the application services exposed as web services in Web Dynpro using Adaptive Web Service model.
5.2.4.2 Exposing Application Services as Web Services

Application Services developed in CAF Core should be exposed as web services so that those can be consumed by the UI layer or the process layer. While exposing the application services as web services the security profiles should be configured as required. Refer the following SAP Help documentation on exposing application services as Web Services in NetWeaver 7.0:

Generating Web Services from Application or Entity Services

For NetWeaver CE 7.1 refer the following document:

Exposing Application Services or Business Objects as Web Service

5.2.4.3 Defining the Service Security

Business Objects/Entities and application services developed in CAF Core can be configured to define permission rules or security principals to restrict access. For Business Objects/Entities permissions check can be enabled from the Permissions tab in the design time and correspondingly the security principals need to be setup from the CAF administration. For application service operations the security check needs to be specified for each operation.

Refer the following SAP Help documentation for service protecting in NetWeaver 7.0:

Service Protecting

Refer the following SAP Help documentation for service protecting in NetWeaver CE 7.1:

Composite Application Service Security

5.2.4.4 Service Eventing and Notification

Framework events like data change in Business Objects/Entities due to CRUD operations can be notified and subscribed using Java Message Service. Refer the following SAP Help document for more details:

Service Eventing and Notifications

Refer the following documents on service composition in NetWeaver CE CAF Core:

Service Composition with SAP Composition Application Framework (SAP CAF) of SAP NetWeaver Composition Environment

SAP NetWeaver Composition Environment 7.1: What's new in CAF?

Refer the following documents for additional information on developing composite services using CAF Core:

EDM EMM Scenario Part II: How to Create Composite Application Services (CAF) with SAP NetWeaver 2004s
How-to-Guide CAF Core

5.3 Developing the User Interface Layer

5.3.1 Selecting the UI technology for composite user interface development

The user interface defined in section 2.5 should be developed after developing the service layer in composite application. The user interface of a composite application on NetWeaver platform can be developed using the following user interface technologies:

- Web Dynpro Java
- Visual Composer
• Adobe Interactive Forms

Each of the above mentioned UI technologies can be used separately or in combination with others developing the user interface of the composite applications.

5.3.2 Web Dynpro Java as UI for composite applications

Web Dynpro Java is a most commonly used user interface technology for composite application development. It provides lot of flexibility for the developer to design the UI and develop the presentation logic. It follows the model-view-controller (MVC) architecture where the model represents the data access layer or the composite service layer. Use Web Dynpro as the UI technology for your composite application if the user interface is complex and requires screen validations.

5.3.2.1 Creating the Model

The composite application services exposed as web services are accessed from Web Dynpro using the Adaptive Web Service model. To create adaptive web service model in Web Dynpro refer the following documents:

- Importing Adaptive Web Service Models
- Creating an E-Mail Client Using Web Dynpro's Adaptive Web Service Model
- WDJ - Highlights of Adaptive Web Service Model
- FAQ - Models - Adaptive Web Service

5.3.2.2 Implementing Value Help

Value help or F4 help are very common in composite applications, which provide a set of values to the user at runtime for an input field. The value help or OVS is implemented in Web Dynpro by implementing an inner class in the component controller and calling services (RFC/CAF Web Service) to get the value set.

Refer the following documents on how to implement OVS/F4 help in Web Dynpro Java:

- Advanced Input Help - The Object Value Selector (OVS)
- How to Create Web Dynpro Value Help Using CAF Entity Service Method

5.3.2.3 Integrating Adobe Interactive Form in Web Dynpro Java

Adobe Interactive forms can also be integrated with Web Dynpro Java using the Interactive Form control available in Web Dynpro design time. To embed an Adobe Form in the Web Dynpro application add the Interactive Form control in the view. Create a context node with the relevant attributes which will be displayed in the Adobe Form. Specify the context node in the DataSource property of the Interactive Form element. Also create another context attribute of type binary which should be specified in the pdfSource property of the Interactive Form control. Editing the Interactive form control will open the Adobe Form Designer in the NetWeaver Developer Studio in which the form layout can be designed and the context node can be bound to the UI elements of the form.

Refer the following documents on more details on Adobe integration with Web Dynpro Java:

- How To... Create Online and Offline Forms in Web Dynpro for Java
- Creating Interactive Forms in Web Dynpro for Java
- Dynamically Generated Forms in Web Dynpro
- Web Dynpro Java Working with Adobe Forms
- Design Concepts of Adobe Interactive Forms Using Web Dynpro Java (Adobe Live Cycle Designer 7.1)
- Tips & Tricks for Developing Interactive Forms (Adobe) in SAP Web AS 6.40 Java
5.3.2.4 Implementing the Web Dynpro Java Callable Object

All Web Dynpro Java components created as the user interface of the composite application has to be exposed as callable object to be used in Guided Procedure. Refer the following documents on how to expose the Web Dynpro Java component as a callable object for Guided Procedure in NetWeaver 7.0:

Implementing a Web Dynpro Callable Object which Implements the GP Interface

Refer the following blog on how to implement Web Dynpro Java callable object in NetWeaver CE 7.1:
Implementing Web Dynpro Java Callable Object (GP Interface) in NetWeaver CE

5.3.2.5 Using Enterprise Portal Services in Web Dynpro Java

Enterprise Portal services like portal navigation, portal eventing, etc can be used from Web Dynpro Java. Refer the following documents for more details:

Using Portal Services in Web Dynpro Java
Web Dynpro Java in the Portal

Refer the following document for highlights of the new features available in Web Dynpro Java in NetWeaver CE:
Highlights of Web Dynpro for Java in SAP NetWeaver Composition Environment 7.1

5.3.3 Visual Composer as UI for composite applications

Visual Composer is a tool for developing user interfaces for composite applications using model driven development and without any coding. The user interface in Visual Composer is developed consuming the composite services i.e. application services exposed as web services, as well as consuming the BAPI, RFC, Enterprise Services or BI Queries.

Visual composer is now available both in NetWeaver 7.0 and NetWeaver CE 7.1, with VC 7.1 having lots of enhanced capabilities than the previous release. The most important features of Visual Composer 7.1 are componentization for reusing VC models and NWDI integration to support the development of Visual Composer models by design time repository, change management, central build services, etc.

For more information on user interface development using Visual Composer 7.0 (NW 7.0) refer the following documents:

SAP NetWeaver Visual Composer – Reference Guide
SAP NetWeaver Visual Composer - How to Guides

For more information on user interface development using Visual Composer 7.1 (NW CE 7.1) refer the following documents:

Visual Composer for SAP NetWeaver Composition Environment
Veteran's Guide Visual Composer for SAP NetWeaver Composition Environment
Visual Composer for NetWeaver CE: Getting Started with a Typical Workflow
Visual Composer Modeling - From Prototype to Full-fledged Application

Visual Composer is also very useful modeling BI reports as part of the composite applications. The BI queries can be directly access in Visual Composer to model BI applications. Refer the following documents for more details on modeling Visual Composer 7.0 (NW 7.0) using BI queries:

Modeling Business Intelligence Applications

Refer the following document for more details on modeling Visual Composer 7.1 (NW CE 7.1) composite views using BI data services:

Modeling Composite Views Using BI Data Services
The composite services developed in CAF Core and Enterprise Services can be consumed in Visual Composer.

Refer the following document for more details on consuming Enterprise Service in VC in NetWeaver 7.0:
How to Consume an SAP Enterprise Service using Visual Composer and the Composite Application Framework

Refer the following document for more details on consuming Enterprise Service from ES Registry in VC in NetWeaver CE 7.1:
How to Browse an Enterprise Services Registry in Visual Composer

5.3.3.1 Visual Composer Callable Object

NetWeaver 7.0 (04s)

To create Visual Composer models as callable object to be used in Guided Procedure set the Add Guided Procedure Information property as below in Tools-> Options menu in Visual Composer storyboard:

![Options menu screenshot]

When deployed in the J2EE engine automatically an iView gets created in the PCD of the Enterprise Portal which can be added as a callable object in Guided Procedure.

Refer the following SAP help documents for modeling Visual Composer as callable object in Guided Procedure:
Modeling Visual Composer iViews for Guided Procedures
Exposing WD4VC Applications as Callable Objects

NetWeaver CE 7.1

In NetWeaver CE 7.1 Visual Composer models always generate a Web Dynpro application which can be either rendered by Web Dynpro runtime or the Flash runtime. The default runtime can be specified from Tools-> Options menu as shown below:
Selecting Web Dynpro runtime the user will get both the options of running the application as a Web Dynpro with Flash rendering and Web Dynpro with HTML rendering after deployment.

The Visual Composer views created using Visual Composer 7.1 can be integrated with Guided Procedure using the WD4VC callable object. Refer the following document for more information on exposing the VC view as a WD4VC callable object:

**Exposing WD4VC Applications as Callable Objects**

For new features of Visual Composer in NetWeaver CE 7.1 refer the following documents:

**Visual Composer for SAP NetWeaver CE**
**Visual Composer (VC) Models Componentization using NWDI**

5.3.4 Adobe Interactive Forms as UI for composite applications

Adobe Interactive forms can be used for composite applications supporting business processes where lots of paper based forms are used. Adobe Interactive Form can replace these paper based forms as online interactive electronic pdf files. Offline processes can also be supported using the Adobe Interactive Forms.

**How-to-Guide Developing with Interactive Forms**
**Enhancing the Job Application Process**
**Tips On Interactive Forms Development Based On Adobe Software - How to Handle Table Input and Output Using Forms in Workflows and Guided Procedures**

Adobe Interactive Forms can also be embedded in Web Dynpro components and used in conjugation with Web Dynpro views. Refer section 5.3.2.3 for more details on how to develop Adobe Interactive Form UI embedded in Web Dynpro Java.

The Adobe forms created should be exposed as callable object to be used in Guided Procedure. Refer the following document on how to create callable object for Adobe Interactive Forms:

**Creating a Callable Object: Interactive Form**

Refer the following document for using composite forms in NetWeaver CE 7.1:

**Integrating Offline and Form-Based Activities**

5.4 Developing the Process Layer

The process layer of a composite application defines the flow of the business process for composite application. To develop multi-role process Guided Procedure is used. The Guided Procedure defines the process flow and developed in the SAP Enterprise Portal which runs on the BPM engine of NetWeaver. The services and user interfaces developed as part of the composite application are used in the Guided Procedure as callable objects for each process step. Different types of callable objects are available in Guided Procedure e.g. Web Dynpro (GP Interface), Visual Composer, Portal iView, Composite Application Service, etc to use the user interfaces and the application services in Guided Procedure.
To develop single user processes for the composite application Guided Activity should be used. Guided Activity does not involve any workflow or BPM engine and implemented by the Roadmap control in Web Dynpro with buttons for navigating back and forth the process steps.

The composite business process is developed according to the process steps designed in section 2.4

### 5.4.1 Designing Blocks

Different types of blocks define the flow conditions e.g. sequential, parallel, looping, and alternate in Guided Procedure.

**Sequential Block**

Sequential block is the most common block used in Guided Procedure. It defines the process with sequential flow. Any type and any number of actions can be defined under a sequential block. A sequential block can be also used to logically group the actions in a process.

**Parallel Block**

Parallel flow can be defined by two types of blocks – parallel block and parallel dynamic block. Parallel block should be used when multiple actions are executed in parallel by the same or multiple processors.

**Parallel Dynamic Block**

Parallel dynamic block should be used when the number of parallel instances for same set of actions is determined at runtime for different processors. User assignment is done at runtime only, for parallel dynamic block.

**Conditional Loop Blocks**

Loop flow can be defined by two types of blocks – pre-conditional loop block and post-conditional loop block. Pre-conditional loop block requires a decision action before the loop body block which will determine the loop
condition. Post-conditional loop block has the decision action executed after the loop body block. The decision action can be a decision dialog or business logic callable object. The decision action should have only two result states - continue and break which will determine at runtime whether the loop will be continued or not. The result states of the decision action in a loop block are not shown at design time when it’s added to the loop block.

**Alternatives Block**

Alternatives block is used when the branching of the process flow is determined at runtime. It requires a decision action which determines which child block will be executed. The decision action should have a callable object like Decision Dialog or Business Logic which may give several result states. The alternate flow paths should be grouped in different child blocks (e.g. sequential blocks). The result state targets of the decision action should be set to different child blocks.

Alternatives Block can be used when the flow is redirected to a set of actions i.e. blocks as an alternative path based on the result state of the decision action.
Refer the following document for details on each types of block used in Guided Procedure:

**Designing a Block**

Exceptions can also be implemented in Guided Procedure to handle error conditions during the process flow. Refer the following document for more details:

**Implementing and Using Exceptions in Guided Procedures**

Refer the following blog on how to create dynamic approval workflow processes using conditional loop block:

**How To Create Dynamic Approval Process Using Conditional Loop Block in Guided Procedure**

### 5.4.2 Developing Callable Objects

The process flow in Guided Procedure is defined by the block types and resultstate target. The callable object can have multiple resultstates as defined in the configuration. For each resultstate a target action can be defined so that at runtime the flow is directed to the target action. Till NetWeaver 7.0 SP10 the resultstate target cannot be set to an action outside its own parent block. But from NetWeaver 7.0 SP11 and in NetWeaver CE 7.1 the target can be set to any action in the process. To set the resultstate target to an action outside the parent block select Other… from the resultstate target drop-down and select the action to which the flow should jump. To set the resultstate target to any action in the current block directly select the action from the resultstate target drop-down of the callable object. By default two resultstate targets are always present for any callable object – No Target and Terminal. Selecting No Target signifies a sequential flow and the flow will go to the next available action. Selecting Terminal will terminate the current block execution.

<table>
<thead>
<tr>
<th>Action</th>
<th>Result State</th>
</tr>
</thead>
<tbody>
<tr>
<td>Action1</td>
<td>No Target</td>
</tr>
<tr>
<td>Result States</td>
<td></td>
</tr>
<tr>
<td>Cancel_desc</td>
<td>Result State</td>
</tr>
<tr>
<td>Reject_desc</td>
<td>Result State</td>
</tr>
<tr>
<td>Release_desc</td>
<td>Result State</td>
</tr>
<tr>
<td>Submit_desc</td>
<td>Result State</td>
</tr>
<tr>
<td>Approve_desc</td>
<td>Result State</td>
</tr>
</tbody>
</table>

### 5.4.3 Starting GP Process

A GP process can be started in different ways as below:

- As a portal iView link
- From the GP Runtime Workcenter
- As a Web Service
- Using GP API

Refer the following documents for more details on each of the above methods to start GP process:

- Four Ways to Start Business Processes in Guided Procedures: Easy, Quickly, and Flexibly
- Starting and Terminating Processes Using the GP API
- Starting a Process Using Web Services
- How To Open Running GP Processes from Web Dynpro Applications

Remember these points while developing processes using Guided Procedure:

- Instead of specifying a UME role for initiator use “Initiator” for the first action and add the process iview to the specific portal role
- Do not pass huge data in GP context – use an unique ID
- Save the GP process instance ID in the CAF layer to open the running process from a custom UI
- Assign the administrator, overseer roles of the process to an actual administrator
- Process runtime dashboard is useful to monitor the running processes

Refer the following document on usage guide for Guided Procedure:

- Usage Guide for Guided Procedures on SAP NetWeaver 7.0

Refer the following document for an end-to-end guide on developing Guided Procedures:

- How-to-Guide Guided Procedures Version 2.0

Refer the following blog on dynamically assigning user to process roles in GP.

- Different ways to model "Dynamical assignment of user to process roles" using Composition Tool Guided Procedures - Part 1

### 5.5 Integrating in Enterprise Portal

The process template developed Guided Procedure and the stand-alone user interfaces developed as part of the composite application should be integrated to the SAP Enterprise Portal for the end-user to access them. iViews, pages and worksets are created in the Enterprise Portal to present the application to the end-user. The Web Dynpro, Visual Composer and the process templates are invoked through iviews from the portal. Refer the following SAP help documentation on integrating application in Enterprise Portal:

- Integrating Applications into the Portal

Refer the following blog on how to integrate the GP process in Enterprise Portal:

- How To Present Your GP Processes To The End User

Refer the following document for an overall guided on Portal Integration of composite applications:

- EDM EMM Scenario Part IV: Process Modeling and Portal Integration

In NetWeaver CE 7.1 portal content can also be modeled by Visual Composer using the Portal Content modeler available in VC 7.1. Refer the following documents for more details:

- Modeling Portal Content with Visual Composer
- SAP Help - Modeling Portal Content with Visual Composer
6. Configuring Composite Application

After the development is over the environment should be configured to deploy and run the composite application. Also these steps should be specified in the configuration guide of the composite application to configure the systems during implementation time.

CAF Core administration can be done from a central user interface provided by the framework. Refer the following SAP Help document for more details on CAF Core administration:
Administrative Tools in CAF Core

6.1 Configuring External Service Destinations

External Services imported in CAF Core should be configured in CAF Administration. Refer the following SAP help documentation for more details on how to configure the External Service destination in NetWeaver 7.0:
External Service Configuration
Using Single Sign-On for Web Services

Refer the following SAP Help documentation on how to create the External Service destination in NetWeaver CE 7.1:
Configuring External Services in CAF Runtime

6.2 Configuring CAF Core for BI Integration

Refer the following document for relevant configuration for CAF-BI integration:
Configuring CAF Core for BW Integration

6.3 Configuring Adaptive Web Service Destination

Adaptive Web Service destination needs to be configured for accessing Web Services (CAF application services) from Web Dynpro Java using Adaptive Web Service model.

NetWeaver 7.0 (04s)

The adaptive web service destination should be of type WSIL which needs to be created in the Visual Admin.

The WSIL URL for an ABAP WebAS is http://<server>:/<port>/sap/bc/srt/wsil?sap-client=<xxx>

The WSIL URL for a J2EE WebAS is http://<server>:/<port>/inspection.wsil
Server is the back-end server
Port is 50000 + back-end instance number
Client is the system client (3 digits). Not mandatory if you need the back-end default client

Create a new Web Service Client (DynamicWSProxy) in Visual Admin with the same name given in the model destination while importing it in Web Dynpro. Specify the WSIL URL and the authentication type as required.
The Adaptive Web Service destination can be either of type WSIL or WSDL. The web service destination has to be configured in NetWeaver Administrator (NWA).

Refer the following SAP help document for more details:

Destination Service
6.4 Proxy Configuration To Access External Services
For accessing external web server from NetWeaver the proxy server needs to be setup if required. Refer the following SDN blog to configure proxy in Visual Admin in NetWeaver 7.0
External Web Service Proxy Configuration for Visual Composer

Refer the following SAP Help documentation to configure proxy in NetWeaver CE 7.1 :
Configuring Proxy Settings

6.5 Configuring Guided Procedure for Adobe Interactive Forms
If Adobe Interactive Forms are used in the Guided Procedure Adobe Document Service and Guided Procedure runtime needs to be configured. Refer the following documents for configuring Guided Procedure and ADS:
Configuring Guided Procedures for Interactive Forms
Configuring ADS for Guided Procedures

6.6 Monitoring and Administrating Guided Procedure
Guided Procedure processes can be monitored from the NetWeaver Administrator (NWA). Refer the following SDN document for more details:
CAF Guided Procedure: System Monitoring
For other administration activities of Guided Procedure the Administration workset in the GP Enterprise Portal role should be used. Refer the following help documentation for more details:
Administration Workset

6.7 Configuring Enterprise Portal
6.7.1 Configuring UWL for Guided Procedure
Universal Work List can be configured to receive the Guided Procedure tasks along with other workflow items. If Guided Procedure connector for UWL is configured in the Enterprise Portal then the users can directly get the GP task in the UWL and can open the process by clicking on the link provided. To configure UWL for GP refer the following documents:
How to Configure UWL with the GP Engine
Guided Procedures: Runtime in Universal Worklist
6.7.2 Assigning the Security Roles

Proper security/UME roles need to be assigned to the users to develop, configure and run the composite applications. Separate roles are required for CAF Core, CAF GP and Visual Composer. Refer the following documents for more details:

Guided Procedures: Roles you need to get started
Setting up Roles for CAF, GP and Visual Composer
Security Guide for Guided Procedures

7. Transporting & Packaging Composite Application

For deploying and packaging composite application the development objects need to be transported from the development to quality and production systems and to distribute those to the third-party consumers or customers. If NWDI is used then transporting of the development content becomes very easy and all standard services for transport and build are provided by the NWDI itself. It is a good practice to create a software component (SC) in SLD and use that to add the CAF-Core, CAF-GP and Web Dynpro DCs in it. Typically for composite applications we need to take care of the following transports of development content:

7.1 Transporting CAF Core Development Objects

CAF Core content can be transported and packaged as DCs from NWDI and the deployment can be managed from the CAF Administration page. Refer the following document for more details:

Transporting CAF Content
Transporting CAF Content Between Different Composites

7.2 Transporting Guided Procedure Development Objects

CAF Guided Procedure process templates and objects can be transported either as local transport package or using NWDI (creating a Guided Procedure 2.0 DC). After transport also the administration activities include releasing the imported objects and setting up the security roles. Refer the following documents for more details:

How to Transport in GP
Transport Management in GP
Checking Transport Requests
Importing Transport Requests

7.3 Transporting User Interfaces

7.3.1 Web Dynpro Java

If NWDI is used for the development then the Web Dynpro Java components are also packaged as Web Dynpro DCs and transported to other systems or distributed externally. Locally developed Web Dynpro components either needs to be migrated to NWDI tracks for transport and packaging or can be packaged as zip achieve files.

7.3.2 Visual Composer

For Visual Composer objects no NWDI integration is available till NetWeaver CE 7.1 SP03. So VC models developed prior to that release should be exported as portal content for packaging and transport. From NetWeaver CE 7.1 SP03 NWDI (NetWeaver 7.0 SP 13 required) can be integrated to create the Visual Composer DC which can be transported and packaged in the same way as CAF or Web Dynpro.
Application Adaptation and Distribution
Transporting an application – Visual Composer Wiki

To setup NWDI for Visual Composer in NetWeaver CE 7.1 refer the following document:
Visual Composer for SAP NetWeaver Composition Environment

7.3.3 Adobe Interactive Forms
Adobe Interactive forms created with Web Dynpro Java will be transported and packaged along with the Web Dynpro DCs. The Adobe Interactive forms created stand-alone and used in Guided Procedure will be transported and packaged along with the Guided Procedure DCs as well.

7.3.4 Enterprise Portal Content
Enterprise Portal content e.g. iViews, worksets, pages, roles can be transported and packaged by creating a transport package for the portal content.

Transport of Portal Objects

7.4 Deployment & Packaging
For deploying the components refer the following document:
Deploying Components

For a composite application typically the following DCs will be transported or packaged for distribution:

- CAF Core DC
- Web Dynpro DC
- Visual Composer DC (only for NetWeaver CE 7.1)
- Guided Procedure DC
- Portal Content

Create the SDA file from all these development components (using NWDI) for distribution or deployment. In NetWeaver CE 7.1 a command line tool is available called CECLT to create deployment archive of the development components. Refer the following Sap help for more details:
Composition Environment Command Line Tool
Related Content

Composite Application Architecture: Introduction and Basic Overview
Architecture Guideline series for Composite Applications Introduction and Basic Overview
Architecture Guideline Series for Composite Applications Portal and Process Layer
Architecture Guideline Series for Composite Applications Business Logic, Abstraction Layer and Connectivity
Composition Environment Info Center
CAF Training Materials Online
Additional CAF Training Materials Online
Disclaimer and Liability Notice

This document may discuss sample coding or other information that does not include SAP official interfaces and therefore is not supported by SAP. Changes made based on this information are not supported and can be overwritten during an upgrade.

SAP will not be held liable for any damages caused by using or misusing the information, code or methods suggested in this document, and anyone using these methods does so at his/her own risk.

SAP offers no guarantees and assumes no responsibility or liability of any type with respect to the content of this technical article or code sample, including any liability resulting from incompatibility between the content within this document and the materials and services offered by SAP. You agree that you will not hold, or seek to hold, SAP responsible or liable with respect to the content of this document.