Version: 2.2

(version for CE 7.1 SP 0 - 3)

August 1, 2007

Authors:

See section 'Authors'

Contacts: Ruth Groene / Patrick Zimmer

iCOD (Industry Composite Development)

Architecture Guideline for Model-Driven Composite Development in the Composition Environment

History

Version	Status	Date
1.00 NW04s	First reviewed and released version	2005-12-09
1.20 NW04s	Final version on NW04s	2006-11-06
2.0 NW CE 1.0	NW CE draft 2	2007-02-05
2.1 NW CE 1.0	NW CE final	2007-05-15
2.2 NW CE 7.1	Valid for CE 7.1 SP 0 - 3	2007-08-02

Contents

1 Revision Log & Authors

Version	Date	Who	Remarks
2.0	Feb 5, 2006	[Authors]	First released version for NetWeaver CE1.0
2.1	May 15, 2007	[Authors]	Updates in CAF chapter, additions of logging and tracing information (appı p. 38), addition of TAM (technical architecture modeling)
2.2	August 2, 2007	[Authors]	CAF topics (chapters 9.4, 9.5), security (20.1.12), Backend Abstraction Lay

Authors and Stakeholders (for the current version of this guideline)

Name	Role
Sebastian Speck	[Author, Reviewer]
Ruth Groene	[Central Contact, Author]
Patrick Zimmer	[Author Business Logic & Process Layer]
Peter Csontos	[Author Backend Abstraction Layer]
Bela Tolvaj	[Author Collaboration Scenarios]
Robert Lorch	[Contact User Interface]
Volker Stiehl	[Author Guiding Principles, Architectural Guidelines]
Frederik Thormaehlen	[Author Reuse]
Pia Kinkel	[Author Portal]
Bernhard Fuhge	[Author Analytics]
Gergely Pap	[Author Globalization]
Gabor Faludi	[Author Security, GP topics]
Marton Pinter	[Contact Backend Abstraction Layer]
Robert Foldvari	[Author UME]
Martin Czekalla	[Author Software Components]
Robert Lorch	[Author Development Landscape]
Silvia Goetz	[Author Product Standards]
Csaba Hegedus	[Author Testing]
Axel Schuller	[Author]
Ulf Fildebrandt	[Reviewer]
Andreas Gebauer	[Program Lead]
Kai Schneider	[Contact for GP related topics]
Sven Krueppel-Berndt	[Project Lead enterprise SOA@PTU IS]

2 References & Related Documents

References - Composition Environment

Here you can find information about the NetWeaver Composition Environment in general:

NW CE	Link
Central Access point for Documentation about the Composite Application Framework (contact: Zornitsa Nikolova)	https://bis.wdf.sap.corp/twiki/bin/view/Main/CompositeApplicationFramework
Summary of the Development Handover workshop "Composition Environment" on November 14/15, 2006	https://portal.wdf.sap.corp/irj/go/km/docs/room_project/cm_stores/documents/workspaces/9163c421-9248-2810-cdbe-aa3356d48314/Architecture/Material/NW_CE-Development-Handover-Workshop.pdf
Complete Material of the Development Handover workshop "Composition Environment" on November 14/15, 2006 (contact: Kevin Flaherty)	https://portal.wdf.sap.corp/irj/go/km/docs/room_project/cm_stores/documents/workspaces/9163c421-9248-2810-cdbe-aa3356d48314/Architecture/Material/NW_CE-DevHandover-2006-11-14.zip
e-Learning "enterprise SOA for mySAP" (technical basics of enterprise SOA for mySAP Business Suite)	https://learningportal.wdf.sap.corp/sap(bD1lbiZjPTAwMSZkPW1pbg==)/bc/bsp/sap/HCM_LEARNING/LESO.HTM?plvar=01&otype=D&objid=70003200&CORE=trainingtype%2ehtm
Implementation Guideline mySAP Service Enabling (Word)	\\DWDF029\Erp_architec\PUBLIC\ERP_Service_Enabling\Architecture\ReleasedDocuments\mySAPServiceEnablingGuide.doc
Implementation Guideline mySAP Service Enabling (PowerPoint)	\\DWDF029\Erp_architec\PUBLIC\ERP_Service_Enabling\Architecture\ReleasedDocuments\mySAPServiceEnablingGuide.ppt
Bluebook mySAP Business Suite Service Provisioning	https://portal.wdf.sap.corp/irj/servlet/prt/portal/prtroot/com.sap.km.cm.docs/corporate_portal/WS%20PTG/Product%20Architecture/Knowledge%20Transfer/Bluebook/mySAP%20Business%20Suite%20Service%20Provisioning.pdf

Related Documents

List of links pointing to a couple of other documents or deliverables that are either important for the program or deliverables of the Architecture Guidelines Project @ iCOD:

Document	Link
iCOD Glossary (containing terms relevant for the complete program)	\\Dwdf051\isidm\PTU_IS_Composite_Dev_Program\35_Architecture\11_ArchitectureGuidelines_LinksAndOtherDeliverables\2006\GlossaryCompositesArchitectureGuidelines.xls
NW Master plan	https://bis.wdf.sap.corp/twiki/bin/view/Sapinternal/NwMp
Location of the current released version of the NW04s version of the iCOD Architecture Guideline	https://portal.wdf.sap.corp/irj/servlet/prt/portal/prtroot/com.sap.km.cm.docs/room_project/cm_stores/documents/workspaces/9163c421-9248-2810-cdbe-aa3356d48314/Architecture/ArchGuideline_CompDevProg_PTU_IS_Current.doc
Guideline for Specifying Composite Applications	https://weblogs.sdn.sap.com/pub/wlg/4356
ISV Enabling Guide	https://www.sdn.sap.com/irj/sdn/go/portal/prtroot/docs/library/uuid/f4a5b25f-0c01-0010-f4a9-dd7a930c46c5

3 Scope of the Document

Composite applications are defined as applications that sit on top of other applications and reuse their functionality by service calls. They empower companies to drive innovative business processes and leverage existing IT-investments. SAP's NetWeaver Composition Environment provides a toolset and runtime for developing, running, and managing efficiently composite applications using SAP's enterprise SOA. With this new breed of application, architects responsible for the technological architecture of a composite application are faced with the challenge of choosing the right technologies for the respective layers of such a solution and their optimal interplay. The layers making up a composite application are:

- Backend Layer
- Backend Abstraction Layer
- Business Objects and Service Layer (comprising the business logic)
- User Interface Layer
- Process Layer
- Portal Layer

This guide aims to explain in detail how to approach composites from an architect's and developer's perspective. This is done by guiding the reader through each layer, discuss possible options for each layer, and give recommendations for architectural decisions based on experience and best practices acquired in first composite implementation projects.

The centerpiece of this guideline are the "Technology Selection" chapters in which each composite's layer will be briefly introduced followed by a choice of technologies available for the layer. Each technology will then be roughly explained including pros and cons and advices about their proper usage within a composite. In order to make this guideline not too theoretical and to overcome possible misunderstandings, an example scenario will be provided which helps to explain how decisions will be made for a concrete application.

It is not the intention of this guideline to go into technical details about how to implement certain functionalities or describe functions and features of particular tools. It is solely focused on the architectural concepts of a composite application, especially the layering of an application and where to implement which logic. Additionally it helps the architect to understand how to translate

a given composite specification delivered by a product manager or business expert into a proper software architecture that fulfils all required functions and features. Discussions and how-to-guides about implementation details can be found in the NetWeaver Developer Guide (https://www.sdn.sap.com/irj/sdn/devguide). Although these technical details are omitted, it should be clear that the intention is to help with architecture decisions for composites being primarily built with SAP's NetWeaver Composition Environment. Therefore the guideline covers tools such as Guided Procedures for process modelling, Visual Composer, Web Dynpro, and Interactive Forms for UI modelling, and the Composite Application Framework for developing the business objects and business services.

The reader benefits most of this guide if he is already familiar with the above mentioned frameworks and technologies. He will also take full advantage of this guide if he has a fair knowledge of composite specifications. These specifications are written by either product managers or business experts and explain in detail the composites functionality. For this it is recommended to have a look at the paper "Guidelines for Specifying Composite Applications" (https://www.sdn.sap.com/irj/servlet/prt/portal/prtroot/docs/library/uuid/20844e88-0d01-0010-de9a-eb2d302df7b7) which details in a less technical manner, how to specify all composite relevant aspects for the architect (e.g. process flow, roles, user interfaces, business objects, and services).

It is recommended to read this guide at least once from beginning to end to be prepared for finding the right architecture for your composite. After you worked through it once it can also be used as a reference book, strongly modularized following the layered approach mentioned above, so that you can jump directly to the chapters you currently need for your decision.

The subsequent chapters provide the following: Chapter *Architecture Goals and Guiding Principles* gives a short introduction to composite applications. It discusses the motivation behind and the basic assumptions about this new breed of solutions. Chapter *Technology Decisions and Constraints* is the heart of the guide: it describes the layering and the technologies applicable for each layer followed by recommendations, the reasoning for the recommendations, as well as tips and tricks. Each layer and their technologies will be explained in great detail. Also covered are cross-topic concerns such as error/exception handling, transaction handling, security, or user management.

The forthcoming chapters will in addition discuss further architecture relevant topics as there are:

- How can components of the several layers be reused?
- How to slice the composite in layers and components?
- Impact of the SAP product standards on composite development
- How does the development landscape of a composite look like?
- How can supportability for a composite be assured?
- How can appropriate testing of a composite be achieved?

Finally the appendix collects links to more information relevant for the development of composite applications as well as a glossary.

3.1 Exceptions & Feedback

This document describes the rules **all** composites developed by iCOD have to fulfill before they are shipped to a customer. Within the Prototyping Phase of a Development Project deviation from the rules are allowed.

Questions about this guideline and how to adopt it in development projects should be discussed with the members of the iCOD Architecture Network (distribution list: *DL PTU IS iCOD Architecture*).

Please note: Especially if certain projects cannot follow one of the given rules, this should be addressed to this iCOD Architecture team.

This Guideline has to evolve iteratively based on our growing knowledge, so the feedback from the development projects is crucial.

4 Overview of Architecture Goals and Guiding Principles

This chapter lays out the general architecture goals and the general composite architecture. The architecture is described with no concrete reference to actual or future technologies to be used in composite development. The concrete technologies are evaluated and described in detail later in the 'Technology Selection' chapters for the individual layers.

4.1 Architecture Goals and Target Architecture

For the sake of this chapter let's begin with a summary of the key statements that characterize a composite application: a composite application is an application making use of data and functions provided as services by platforms and applications and combining these into usage-centric processes and views, supported by its own business logic and specific user interfaces.

A composite application has to fulfill the following characteristics:

- ❑ Own lifecycle
- ❑ Loosely coupled with backend systems
- ❑ Integration with backend via stateless service calls
- ❑ Backend independency
- ❑ Easy to adopt/enhance for customers
- ❑ Model-driven Architecture

These characteristics will highly influence the decisions to be taken in the forthcoming chapters.

4.1.1 Architecture Goals

Based on the key characteristics of a composite application mentioned in the previous section and under consideration of the available tool support for composite development, the overall architecture goals can be broken down to the following high-level guiding principles:

- Optimize development efficiency by using **model-driven development** tools
- Increase flexibility and backend independency by an **enterprise SOA-compliant architecture**
- Improve user acceptance by following an **user-centric approach**

- Combining existing services to new processes, extending a composite by a model-driven approach, and a simple installation and configuration process at the customer's side all contribute to a **short-time-to-value**.
- Composites will typically face the following challenges:
    - They will live in different customer landscapes
    - Typically the processes a customer needs will slightly differ from the shipped solutions implemented in the composite
    - The user interface is very often target for change, especially if a UI has to follow certain UI guidelines

  Taking these challenges into account an easy **adaptability** is of paramount importance for a composite application.
- Follow the KISS principle (keep it short & simple) in first place – **simplicity** is the key criteria during the decision taking process.

Each of those principles will be discussed in greater detail in the sections to come.

4.1.1.1 Model-Driven Development

The composites shall be developed following the paradigm of model driven development. So the amount of handcrafted coding shall be reduced and the amount of configured or modeled parts of the composite shall be increased leading to an increase of the overall productivity and quality.

Model-driven development should be used on all layers of a composite. For this the SAP NetWeaver Composition Environment comprises the following tools supporting model-driven development:

- CAF Core for business objects and service composition
- Visual Composer for simple, straightforward online user interfaces
- Web Dynpro for more advanced online user interfaces
- SAP Interactive Forms for form based offline user interfaces
- Guided Procedures for process modeling and service orchestration

For sure the possibility to reach this goal is dependent on the capabilities of the available tools, but it is the clear mission of this guide to address applications which benefit from the model-driven development paradigm and to use the existing model driven tools wherever possible.

4.1.1.2 Enterprise SOA-Compliant Architecture

The composites shall be built according to an enterprise SOA-compliant architecture. To fulfil this the composites have to:

- Integrate with the backend via Enterprise Services.

  With mySAP ERP 2005 SAP delivered an enterprise services enabled business process platform. As other vendors of ERP systems follow the same service oriented approach, the enterprise SOA compliance ensures high interoperability in heterogeneous landscapes. According to this approach the composites make a step into the direction of backend independency. This enables the customer to integrate the composite with systems not directly implementing these services, e.g. Non-SAP-Systems or SAP-Systems with older releases.

In addition composites reuse existing functionality and add their own business logic where standard processes fall short. This is done in a non-invasive way allowing independent life-cycles of the composite and the integrated systems.

- Separate concerns clearly.

  By the definition of the different layers of a composite and the rules for the communication between them, it is ensured, that the composites have a complete decoupling of business logic, UI, and process logic and are loosely coupled to the backend.

4.1.1.3 User-Centricity

Composites shall provide instant usability to its users. The goal is to combine the services included from the different backend systems and those developed in the composite to homogeneous processes and user interfaces. On the one hand side the user interfaces should be **task oriented**, so only providing the information or requiring that data that is relevant for the current task of the user. Task oriented UI's will typically be built with either Visual Composer, Web Dynpro, or Interactive Forms depending on the specific needs. On the other hand side the user interfaces should be **process oriented**, meaning that they should guide the user through the relevant process steps. These kinds of UI's are covered by the Guided Procedures framework.

One of the main value propositions of a composite is to streamline or extend a process available in the backend and provide it in a role-based, user-friendly and process-oriented way to its users. To achieve this, an early and profound UI design is essential.

4.1.1.4 Short-time-to-Value

Another value proposition of composite applications is the reuse and combination of existing assets of the customer's system landscape to implement new business processes. To achieve this reuse, the composites have to integrate into the existing landscape without requiring either a major upgrade or a modification of the existing backend systems like an ERP (non-invasiveness of a composite). So the goal of the composites is to support existing and shipped releases of SAP and Non-SAP systems. The reuse of existing functionality contributes significantly to the "Short-time-to-Value" goal, as existing functionality doesn't have to be re-implemented again, so that development can concentrate on the missing new functionality.

"Short-time-to-Value" can also be achieved by reducing the time to get a composite running at the customer site. This time should be as short as possible and can be achieved by:

- Simple configuration
- Model Driven Extensibility
- Simple installation process
- Integration into existing system landscape without requiring additional hardware or instances (TCO)

Obviously these points have to be supported by the respective development tools and frameworks.

Besides the advantages of the Short-Time-To-Value for the customer, it also makes sense for vendors like SAP to aim for Short-time-to-Value, e.g. it allows vendors to plan composites with a period of 6 month from decision to delivery.

4.1.1.5 Adaptability

As a consequence of the model-driven approach outlined in chapter Model-Driven Development (4.1.1.1), composites shall be easy to change and extend by a customer in a model-driven way

as well, therefore requiring no additional programming efforts. By this a higher flexibility will be provided to the customer optimizing the overall adaptability of composites. The need as well as the available technical possibilities differ for the different layers of a composite application. The following examples showcase typical scenarios:

- Backend Abstraction Layer: adapt the composite to run against a large number of backend systems
- Business Objects and Service Layer (comprising the business logic): adapt business objects and services to the customer's needs: add additional attributes to pre-delivered business objects or enhance the business logic of an existing service
- User Interface Layer: adapt the UI in a way required by the customer, e.g. remove useless fields and add new fields
- Process Layer: add new steps to or remove needless steps from the shipped processes

4.1.1.6 Simplicity

KISS - Keep it short & simple!

On all aspects of the composite development and all decisions taken, the guiding principle should be that the final product should look and behave simple by avoiding unnecessary complexity or hiding needed complexity behind the scenes.

4.1.2 Basic Architecture of a Composite

The anatomy of the composite (see Figure 1 for details) is a high-level description and is independent from the tools used for implementation of the different layers. Which technology to use per layer and for the communication between the layers is described in the "Technology Selection" chapters.

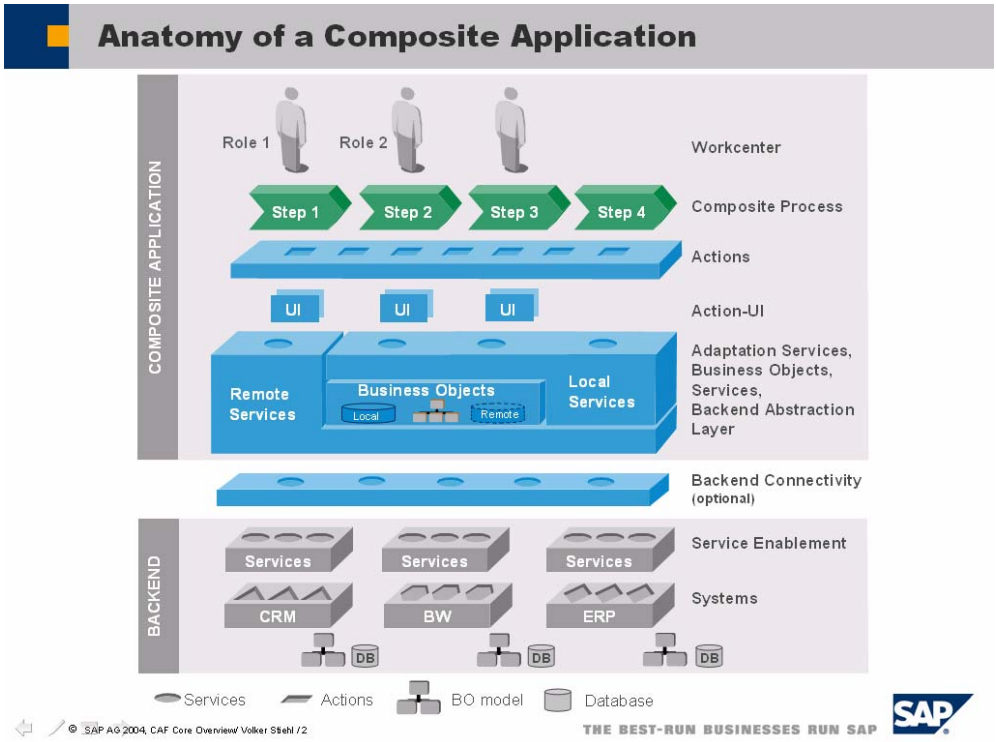


Figure 1 Layers of a composite application

4.1.2.1 Service Enablement Layer

The service enablement layer provides the data and functionalities available in the backend systems, which are used in the composite, by services. As figure 1 already indicates, the development of the services is not part of the development of the composite itself. Instead the services have to be provided by the backend systems, which contain the core business logic and data.

4.1.2.2 Backend Connectivity Layer

As composite applications are defined as applications sitting on top of other applications which reuse existing functionality via service calls, the question comes up how the consumption of those services technically looks like. The technology of choice is the **direct connection of the business logic layer with the backend via standards-based Web Services technology** and it is indeed the recommended solution of choice for composite applications.

Although this might be suitable for the majority of use cases, an alternative for situations in which simple Web Service calls cannot fulfill the requirements must be found. As an example consider an IT landscape at the customers side, which doesn't offer a certain business functionality required by the business logic layer to be executed by exactly one service call. Instead a series of calls, probably with parameter adaptations in-between, is necessary to achieve this goal. In this case a message broker comparable to SAP's Exchange Infrastructure (XI) might be used as a messaging middleware for service communication, connectivity, transformation and portability in order to connect the composite with its backend systems. As the above-mentioned

example illustrates this software is useful in case the simple web service call is not sufficient for the landscape found at the customer's side and functionality such as sophisticated parameter mapping/transformation, default value handling, and message routing is needed which is usually found in these kinds of integration software. Typically this is not part of a final composite application, it's rather the customer's responsibility to customize the application in a way that it involves communication with a messaging infrastructure.

In case a full-blown message broker is not an option, hand-crafted solutions based on the service framework provided by the business logic layer are also an alternative. How to construct such an architecture by following a design-pattern approach to overcome the integration limitations of the business logic layer will be discussed in this paper as well and is especially useful for switching easily between SAP systems of different releases. This backend abstraction layer as well as the motivation for it, and a possible architecture will be discussed in detail in chapter 6.1.

By default, the composite's business objects and services layer makes direct calls to the provided backend enterprise services and by this abstracts the calls to the higher layers (UI-/process layer). However, it is also possible for the higher layers to call backend services directly. Due to the lost flexibility this architecture should be avoided if possible. In the target Enterprise SOA based architecture the technology of choice to connect composites with the backend systems is Web Services.

However, independent of the solution chosen for overcoming these connectivity challenges, the main idea of composites is in all cases the same: no technology related interfaces appear on higher levels (e.g. business logic layer, process layer)! The higher levels can rely on stable interfaces which will not change in case the underlying technology, by which the services are called, changes. Following this approach a decoupling of functionality and technology will be achieved.

4.1.2.3 Business Logic Layer

Within the business logic layer the business logic and the business objects specific for the composite are implemented. The **unified business object model** provides the flexibility to make transparent usage of business objects with local or remote persistency. The **unified service model** provides service abstraction and shields higher layers from service implementation details making them replaceable. So it is recommended to make use of this abstraction in the UI- and process layer to benefit from its flexibility to adapt the final solution to different target IT-landscapes.

The business objects can have different complexity with regard to their persistency:

- No persistency in the composite.
  The business objects serve for backend abstraction and to provide services to the UI-/process layer in the right granularity. They are only persisted in the backend.
- Persistency only in the composite.
  The business objects do not exist in the backend system and are local to the composite.
- Persistency in the composite and the backend systems.
  By that replication and synchronization between the composite and the backend systems is needed adding additional complexity to the implementation of the application. Strategies to avoid replication and how to deal with it if it's needed are described in a chapter 7.2.
  However, there are situations thinkable for which the persistency for the same business object in a distributed environment isn't an issue at all if it is ensured that the same data is only changed in one place at one point of time. This can be ensured by a proper data model or process design.

On this layer model driven development shall be used to model the business objects (attributes, nodes, and services) and generate and manage the local as well as the remote persistency of the business objects.

4.1.2.4 User Interface Layer

New user interfaces can be created on top of the services provided by the business logic layer. By only using services of the business logic layer a clear decoupling between the UI and the business logic is implied. The user interface layer comprises online as well as offline UIs.

Experiences made during the development of first composite applications showed that sometimes an additional UI-related logic layer is necessary. This layer sits in-between the UI- and the business logic layer and links the UI-agnostic services of the business logic layer with the UI. It provides specialized functionality optimized for a particular UI which, without such a layer, had to be implemented in the UI itself making it hard to distinguish between mainly UI related logic (e.g. screen flow, layout, user interactions, events) and data adaptation and modification logic. Therefore the UI and this new UI-adapter layer are inseparable connected.

We can approach this problem also from a different perspective: business logic services have to be independent from UI's (otherwise their reuse is almost impossible). On the other hand UI's have requirements regarding supporting services in order to provide an optimal user experience for the end user. In most cases the requirements of the UI-layer don't fit with the services provided by the business logic layer. Therefore the UI-adapter layer takes over this part to mediate between the two. As this functionality will be provided as services, they will typically be implemented in the business logic layer. This fact is indicated by the adaptation services being part of the business logic layer in figure 1.

On this layer model driven development shall be used to model the User Interface screens including the screen flow and the possible user interaction.

4.1.2.5 Process Logic Layer

Within the process logic layer it is defined, which process steps are executed in which sequence by which roles and how the context data of the process is passed between the participating process steps.

Within each process step exactly one action is executed. An action is a wrapper around a so-called callable object which either represents a user interface or a service. Such an action has importing and exporting parameters. To indicate how data can be passed from a previous action (a) to the next action (b), the appropriate parameters can be assigned to each other on the process layer (output parameters of action (a) are mapped to the input parameters of action (b)).

One of the benefits of actions is that they decouple process steps from services and user interfaces to allow business experts to model processes on a non technical level.

On this layer model driven development shall be used to model the process, the process steps and the actions thus avoiding hard-coded process flows within the business logic layer.

4.1.2.6 Portal Access Layer

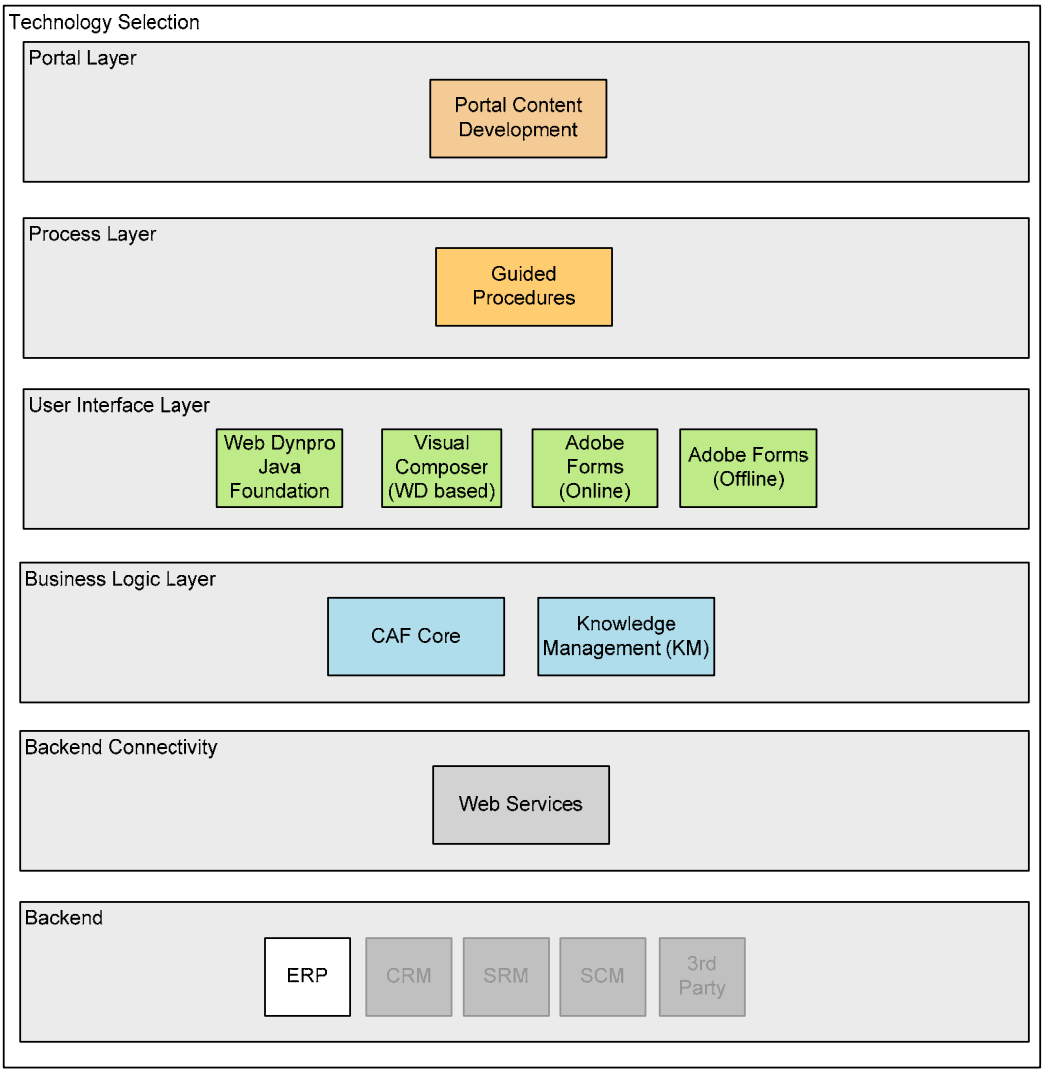
Within the portal access layer the user interfaces and processes are provided in a role-based manner using the work- and control-center concept. Let's have a closer look at this concept: a user may have different roles. For each role a work center exists. The work center provides a role and task-oriented view of data and activities (e.g. customer service care, employee self service, purchase order management). It consists of a set of pages organizing and supporting

the user's activities in a certain area (work set), e.g. the work set ‘employee self service’ with the activities e.g. leave request and address change. On the other hand side the control center provides an overview of all work centers the user is utilizing.

Besides the provisioning of the control center/work center concept by the portal layer, it also takes care of the central entry point for work items the end user has to care of. This is covered by the so-called Universal Work List which is needed as the means of forwarding follow-up process steps between users.

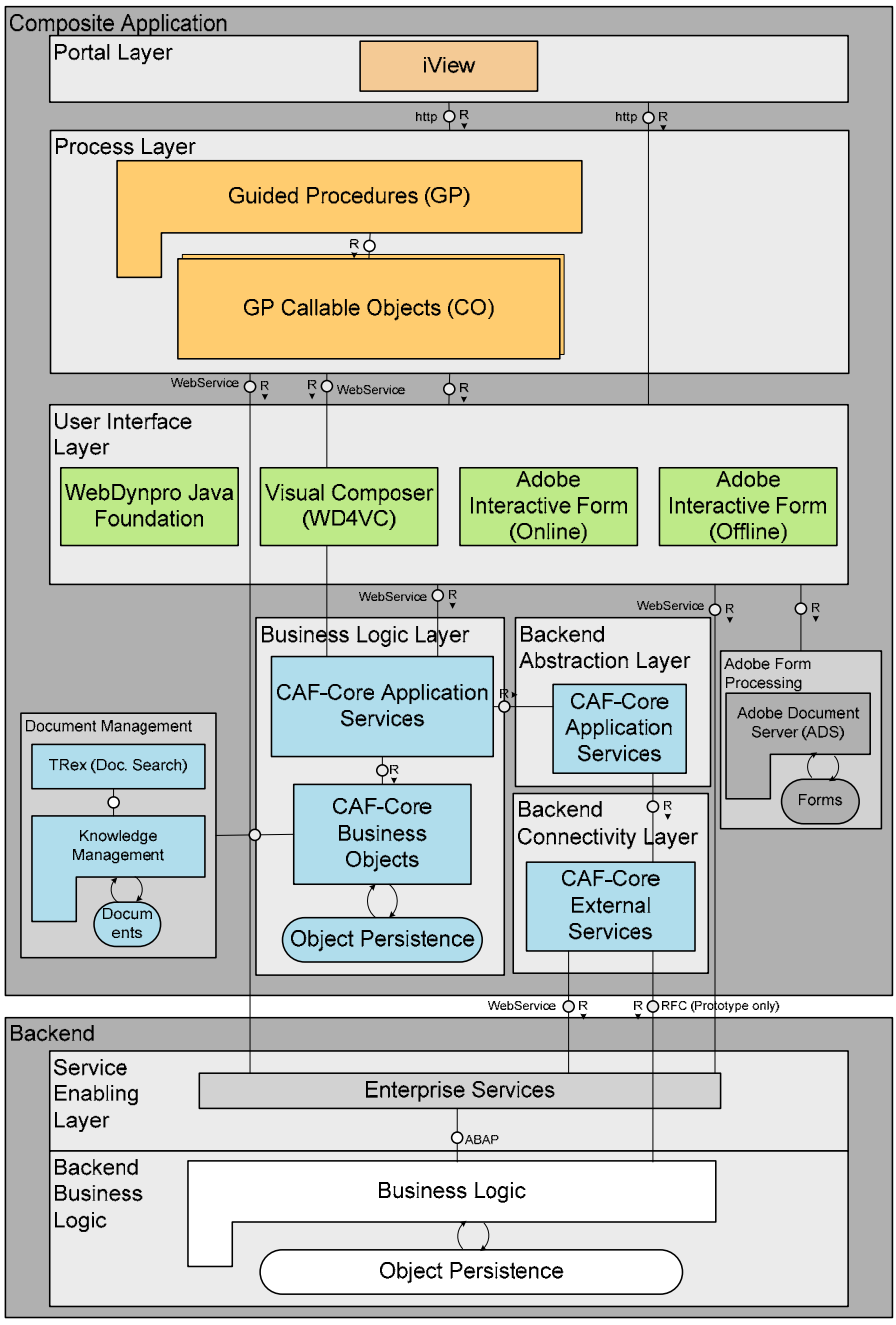
5 Overview of the Technology Selection

For each of the layers of the Composite Application Architecture it has been decided which technology to use. In this diagram the technology selection for iCOD is depicted.



✘ **The usage of any technology not mentioned in this diagram and released for usage in the following chapters is forbidden.**

The following figure illustrates how the individual layers work together:



In the following chapters the following questions will be answered per Layer:

- Why the specific technology has been selected

- Why other technologies have been ruled out
- For which use case, which technology has to be used, if more than one technology is allowed
- What are the most important rules for usage of a specific technology
- Where to find more information

6 Technology Selection for Backend Connectivity

A Composite is reusing the functionality and data provided by its backends. Within enterprise SOA this is done by consuming Enterprise Services.

An Enterprise Service is defined as a service which provides business functionality and which is published by SAP in the “SAP Enterprise Service Inventory” (currently ES Workplace). Enterprise Services are structured according to a harmonized enterprise model based on process components, business objects and global data types. They are well documented, guarantee quality and stability and are based on open standards.

☑ **Synchronous Enterprise Services consumed via Web Service Protocol have to be used for backend connectivity**

The consistent definition of Enterprise Services is ensured by usage of Patterns, which are described in this presentation:
https://portal.wdf.sap.corp/irj/go/km/docs/room_project/cm_stores/documents/workspaces/9163c421-9248-2810-cdbe-aa3356d48314/Architecture/Material/Enterprise_Services.ppt

The most important patterns for the usage within a composite are for example Manage Business Object, containing operations to Create, Read and Update a Business Object Instance.

☑ **The Composite only acts as client calling the Enterprise Services**

✘ **No outbound calls or events from backend to composite are supported currently**

Events sent from the backend to the composite would be especially needed in cases, where the backend wants to inform the composite about exceptional situations, so that the composite is able to handle those.

As of today no coherent event infrastructure exists, which allows to provide events within the backend and consume them in the composite. As part of NW CE 2.0 Project Galaxy plans to provide this infrastructure.

✘ **No asynchronous communication between a composite and a NW 04s-based backend system (e.g. mySAP ERP 2005) is supported currently**

Asynchronous communication would be especially needed, to decouple the Composite from the backend in cases where the composite is not dependent on the direct result of the service call, e.g. when Updating a Business Object. By this the following advantages could be achieved:

- Performance by non-blocking execution of the service
- Reliable Transportation of several messages (in order)
- Ability to send and queue messages, if the receiver is not available

Support of WS-RM (Web Service-Reliable Messaging) for asynchronous reliable communication (Exactly-Once or Exactly-Once-In-Order) is under discussion for CE 7.1.1 and as downport for the NW 04s ABAP Stack. This WS-RM downport is the only chance to consume asynchronous services from within CE as asynchronous XI services can't be handled by the WS-Runtime as they have a proprietary implementation of "reliable" communication.

☑ **RFC-enabled Function Modules and BAPIs consumed via RFC protocol may only be used during development and for pilot projects until the Enterprise Services are available**

RFCs and BAPIs is another way supported by NW CE to communicate with SAP backends, but as this is not compliant with enterprise SOA and only usable for SAP-systems, the usage is only allowed as temporary workaround.

Besides Enterprise Services also services provided over the internet could be consumed by a composite. By usage of those a Composite can become a MashUp, which is defined as a web-site or web application that seamlessly combines content from more than one source into an integrated experience.

Some details, examples and ideas about MashUps@iCOD are described in this presentation.
https://portal.wdf.sap.corp/irj/go/km/docs/room_project/cm_stores/documents/workspaces/9163c421-9248-2810-cdbe-aa3356d48314/Architecture/Material/Mashups%40iCOD.ppt

? **If you have ideas or plans to integrate web content into your composites, please contact iCOD Architecture**

6.1 Backend Abstraction Layer

In order to enable a number of pilot customers to adopt composites early, it is allowed as a temporary solution to ship iCOD applications using BAPIs and remote-enabled function modules. The main reason for this is that enterprise services used by composites are being developed in parallel with the composites themselves and sometimes they are not released until the pilot customer starts using the product.

This approach bears a risk: when the services are ready, it needs too much effort to migrate the business logic layer of the application from a BAPI-based implementation to an enterprise service-based one. As a solution for this problem, the concept of Backend Abstraction Layer has been established. Adding this new layer to the application, the migration effort can be significantly reduced.

Moreover, adding this new layer also brings other values: it helps to design the business layer better and more easily up front even when only BAPIs are available. It provides a consistent view of different connectivity types and different conventions followed in BAPIs; hides differences in types used, parameter syntax (i.e. leading zeros), parameter semantics (like search criteria connected with logical AND or logical OR operation), return values and identifier codes. BAL also makes it possible to change between different systems (ERP, SCM etc) providing the same functionality without modifying the core business implementation. It also gives valuable information for the definition of ESA services, and makes unit testing of the business logic easier.

☑ **Backend Abstraction Layer concept introduces an additional layer of indirection to consistently separate backend connectivity from business logic**

☑ **Backend Abstraction Layer (BAL) is to be implemented in all iCOD composites**

☑ **BAL is to be implemented using CAF Core and EJB according to the pattern described in the following documents:**

- A detailed document describing the guidelines for the Backend Abstraction Layering: https://portal.wdf.sap.corp/irj/go/km/docs/room_project/cm_stores/documents/workspaces/9163c421-9248-2810-cdbe-aa3356d48314/Architecture/Material/iCOD_BackendAbstractionLayerGuideCE.doc
- A summary Powerpoint presentation about the concept: https://portal.wdf.sap.corp/irj/go/km/docs/room_project/cm_stores/documents/workspaces/9163c421-9248-2810-cdbe-aa3356d48314/Architecture/Material/BackendAbstractionLayerGuidelines.ppt

6.2 Backend Adapter

Not in all cases right at the beginning of the development phase all needed Enterprise Services will be available. In this case the usage of existing RFC-enabled Function Modules and BAPIs is allowed as described before, but also these may not cover all the needs. To circumvent this temporary issue the backend Adapter (aka Z-connectivity) was introduced. The Backend Adapter provides the connectivity within the Backend to compensate missing Enterprise Services.

The most important decision is, if you need a Backend Adapter at all:

If you can emulate the needed functionality of an Enterprise Service with already existing RFCs:

- ✘ **Don‘t make changes in the backend**
- ☑ **Call these RFCs from Backend Abstraction Layer via RFC protocol and add the additional needed logic for mapping there**

If you have to do any kind of changes on the backend (e.g. missing functionality, remote enabling …):

- ☑ **Do all the coding to emulate the Enterprise Service in one new RFC on the backend**
- ☑ **Call this one RFC from Backend Abstraction Layer via RFC protocol**

In case you need own RFCs, these are the most importance rules how to implement those:

- ☑ **The Backend Adapter should be as thin as possible**

  In the worst case every customer has to do the same manually in his system

- ✘ **Don’t change any existing objects in the Backend**

  Only creation of new objects is allowed. Before using appends, implementing BAdIs or any other enhancement technology contact iCOD Architecture

- ✘ **Don‘t provide functionality you don’t have requested as enterprise services**
- ✘ **Don‘t use features you won‘t have with enterprise services (e.g. transaction handling, static/global variables)**

  Exception: If you do it within one RFC substituting on Enterprise Service to emulate his functionality

- ☑ **Ensure that the Backend Adapter runs on the lowest supported backend release of your composite**
  - **Ideally develop it on the lowest release (4.6C)**
  - **But to be on the safe side – test it in all supported releases**
- ☑ **Do NOT copy function modules, WRAP them**

As the Backend Adapter is not part of the iCOD product, special means have to be taken to provide it to the customer:

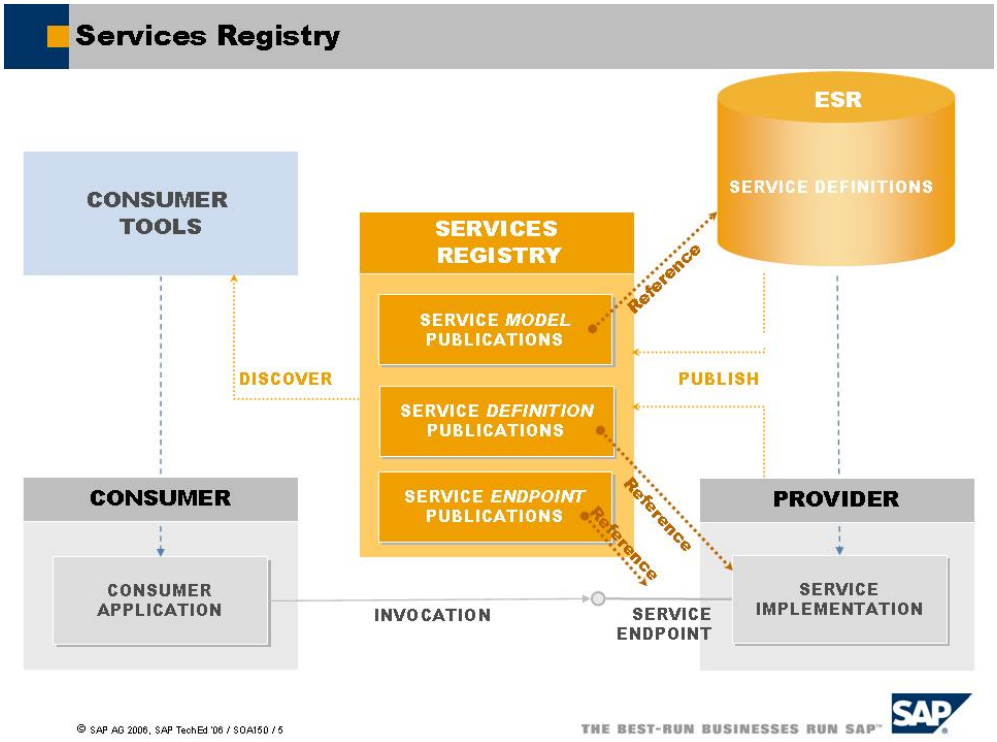
- ☑ **Put all backend developments into exactly one local package per composite**
  - **To easily get an overview of the Backend Adapter**
  - **To ease software logistics**

- ☑ **Use the prefix <Z + composite abbreviation + '_'> for all objects you create (e.g. ZMOP_*)**
  - **If you just wrap a function module for RFC enabling, keep the name of the function module but add the prefix**
- ☑ **All backend developments are provided to a customer via a note**
  - **Use note 987379 as template (w/o web service creation)**

More details on the Backend Adapter can be found in this presentation: https://portal.wdf.sap.corp/irj/go/km/docs/room_project/cm_stores/documents/workspaces/9163c421-9248-2810-cdbe-aa3356d48314/Architecture/Material/BackendAbstractionLayerGuidelines.ppt

6.3 Enterprise SOA System Landscape

The high level architecture of the communication between service provider and service consumer at design time and run time is depicted in the following diagram:



Services can either be called directly via the Web Service Infrastructure or managed via XI.

By doing it managed via XI you gain the following advantages:

- The mapping could be used by a customer to map the service calls of the Composite to a different (e.g. Non-SAP) system in his system landscape
- The routing could be used to determine the target systems for service calls.

On the other hand the following disadvantages come with the usage of XI (at least in NW 04s):

- No passing of the user calling the service to the backend, which is a requirement for security reasons (Authorizations and Audit Trail)
- Additional TCO for the customer, if he has to install and configure XI just for the Composite

XI Integration Builder (aka Enterprise Services Registry) is used as Design Time for the definition of GDTs and interfaces of Enterprise Services. XI Integration Server is not used as a runtime tool to invoke the Enterprise Services from the Composite.

☑ **Enterprise Services will be called directly via the Web Service Infrastructure**

☑ **Customers do not have to install XI as prerequisite to run the composites**

☑ **Enterprise Services are published from the service provider to the Services Registry and discovered from there by the service consumer**

Web services must be activated for the SOAP runtime to be consumable from the outside. This activation has to be done per client in the Backend system. As a result of this activation the WSDL gets enlarged by the Runtime-Part (containing the physical server and binding).

- The activation can be done using transaction WSCONFIG for a single web service.
- The activation can be done by executing report RSWSCONFIG with transaction SE38 for several web services.
- It's planned to have a consumer-driven mass-activation of web services with CE 7.1.1. This will be the recommended solution for mass configuration by then.

One of the goals of NW CE is to unify the backend connectivity and integrate all the different configurations existing within 04s into one common connectivity configuration framework. Details on the status reached with CE 7.1 SP 0 - 3 can be found in this presentation.
https://portal.wdf.sap.corp/irj/go/km/docs/room_project/cm_stores/documents/workspaces/9163c421-9248-2810-cdbe-aa3356d48314/Architecture/Material/CE1.0WSConnectivity.ppt

6.3.1 Backend Connectivity Configuration

The configuration concept is based on the idea to use logical destinations within all web service consuming tools to define the backend to call. Logical destinations are used as abstraction layer above the physical endpoints. Within configuration a customer than is able to assign physical destinations for his landscape to these logical destinations.

☑ **Logical Destinations have to be used for all consumed Enterprise Services and BAPIs/RFCs**

As of NW CE 7.1 SP 0 - 3 there is no means to transport the Logical Destinations. They have to be created manually by the customer within the NW Administrator based on the documentation provided within the installation guide. With CE 7.1.1 it's planned to enable delivery of Logical Destinations with the Composite via NWDI.

To ease the life for the system administrator potentially installing several composites, the logical destinations shall follow a naming convention.

☑ **All Logical Destinations have to be named SAP_<ProcessComponent>, where ProcessComponent is defined by the Enterprise Service used.**

This recommendation is based on the idea, that all Services provided by the same process component reside in the same physical system at a customer and by that can be configured to point to the same physical destination.

6.4 Transaction Handling

Details on Handling of Transactions within a Composite can be found in this presentation. https://portal.wdf.sap.corp/irj/go/km/docs/room_project/cm_stores/documents/workspaces/9163c421-9248-2810-cdbe-aa3356d48314/Architecture/Material/Transactions%40C42_iCOD.ppt

☑ **Enterprise Services are stateless and atomic**

Stateless means that no state is kept in the backend between two service calls.
Atomic means that each service updating the database does its own commit.

As consequence it's not possible to combine several service calls into one Logical Unit of Work, which is ended by a Commit or Rollback. Neither it is possible to keep locks in the Backend between the point of time data has been read until it's updated.

✘ **There is no generic support for distributed ACID transactions between a composite and its backends**

In Database Transactions ACID stands for Atomicity, Consistency, Isolation, and Durability to ensure database integrity. Databases normally uses locking to ensure it.

Within the loosely coupled world of enterprise SOA using stateless and atomic services potentially distributed over several systems connected via a network, the same level of integrity can not be provided.

6.4.1 Locking

There are two ways to prevent concurrent updates: pessimistic and optimistic locking. Pessimistic locking is where a record is locked immediately when the lock is requested, while an optimistic lock is where a record is only locked when the changes made to that record are updated. The stateless nature of Enterprise Services makes pessimistic locking infeasible, cause there's no state which keeps the lock between the reading and the updating service call.

✘ **Enterprise Services don't support pessimistic locking**

☑ **Use an Update <BO> service operation to apply optimistic locking**

The change service operations do not support optimistic locking. It does not check, if somebody else has changed the same BO instance since the last read. This means if somebody else has changed the same BO instance those changes will be overwritten without any warning.

The update service operation checks for concurrent updates and by that supports optimistic locking. It checks, if somebody else has changed the same BO instance since the last read based on a ChangeStateID passed by the Read <BO> service. This means if somebody else has changed the same BO instance there will be an error message send back. It is then up to the composite to react on this:

- Ask the user and call the change operation
- Cancel the changes (data entered by user is lost)
- Offer a screen to merge the changes.

This means in cases with a high probability of concurrent updates, or in cases where the "last-one-wins" logic of the change service operation is not acceptable, request an Update <BO> service.

The differences between a change and an update service operation are described in more detail in this document.

https://portal.wdf.sap.corp/irj/servlet/prt/portal/prtroot/com.sap.km.cm.docs/corporate_portal/WS%20Application%20Platform/03_Architecture/Concept%20papers/Business%20Process%20Management/ChangeServices.doc

All Enterprise Services change and update service operations have the following behaviour:

- For elements which are not part of the message payload the corresponding fields in backend databases won't be changed.
- For elements which are part of the payload but have initial value the corresponding fields in the backend databases will also be set to initial value.
- And of course for elements which are part of the payload with non initial value the corresponding fields in the backend databases will be changed accordingly.

This behavior is described in more detail including a description of table handling is described in this document:

https://portal.wdf.sap.corp/irj/go/km/docs/room_project/cm_stores/documents/workspaces/9163c421-9248-2810-cdbe-aa3356d48314/Architecture/Material/behaviour_of_update_services.doc

6.4.2 Idem potency

For asynchronous A2A or B2B communication brokered by XI, the Exchange Infrastructure ensures reliable messaging, including guaranteed delivery and exactly-once execution of each message.

However, in direct point-to-point communication, where Web services are called directly using the SOAP protocol, there is no guarantee that each request message sent by the consumer will arrive exactly once at the provider. A message may get lost or may arrive several times in case of network problems. Even if the request message arrives at the provider, the response message may get lost during network transport. Hence the consumer might assume that its request did not arrive and resend its request.

For many services, however, it is crucial that the service is executed exactly once when requested. For example, if a new purchase order shall be created via a service invocation, it is neither acceptable to have no purchase order created, nor to have two or more duplicates created due to repeated processing of exactly the same request.

☑ **Enterprise Services support idem potency to ensure exactly-once execution of a service operation**

Service consumers have to explicitly "request" idempotent behavior by providing a non-empty value for the message UUID

☑ **Ensure by idem potency that the same service call is executed under the same Business Document Message UUID from the Business Document Message Header**

Especially in a Browser-based User Interface there's the risk, that after getting no response for some time, the user presses the Refresh-Button of the Browser and sends the same. By using idem potency it can be prevented that the same object is created twice.

☑ **Use java.util.UUID.randomUUID() to create the UUID per distinct Service Call**

✘ **No generic support of idem potency by the Web Service Consumption tools in CE 7.1 SP 0 - 3**

? **iCOD Architecture further evaluates the handling transactions using Enterprise Services**

☑ **Repeat failed synchronous Enterprise Service calls until they are executed successfully, ensuring by idem potency that they are executed exactly once**

Details on the implementation of idem potency within Enterprise Services can be found in this document:

https://portal.wdf.sap.corp/irj/go/km/docs/room_project/cm_stores/documents/workspaces/9163c421-9248-2810-cdbe-aa3356d48314/Architecture/Material/Exactly%20Once%20Implementation%20Guideline.doc.

6.4.3 Transaction Handling in CAF Core

☑ **Use a Check <BO> service operation to minimize the potential of errors by modifying services and to give early feedback about erroneous input to the end user**

Enterprise services do not have a “Testrun” parameter. This Parameter existed in BAPI’s and if set, all checks of the BAPI were executed, but no changes to the database were done.

Especially in interactive forms with a check and a submit button, in a UI consisting of several screens and in GP with separated UI and Background Callable Objects, the additional check service is needed.

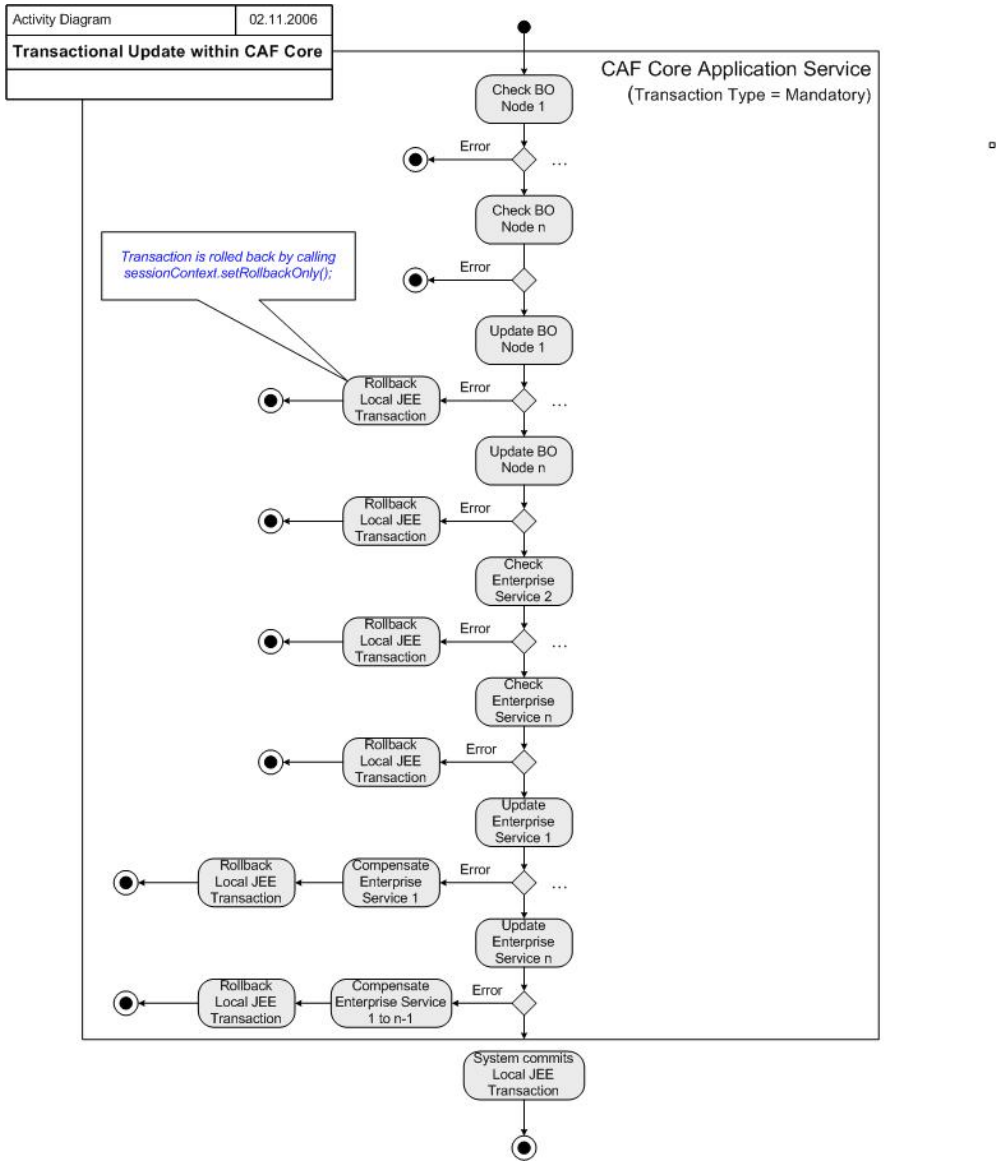
If you want to update within one Application Service both the local CAF Business Objects as well as the backend via Enterprise Services, apply the following rules:

☑ **Set Transaction Type of the Application Service to Mandatory**

☑ **Implement the Logic within an Update Application Service in the following sequence:**

1. **Local Checks**
2. **Local Updates of CAF Business Objects**
3. **Remote Checks by Enterprise Services (if more than one Update Enterprise Service is called)**
4. **Remote Updates by Enterprise Services**

Do the local updates first as your able to roll them back. Do the remote checks as this reduces the risk that you have to compensate an Enterprise Service call.

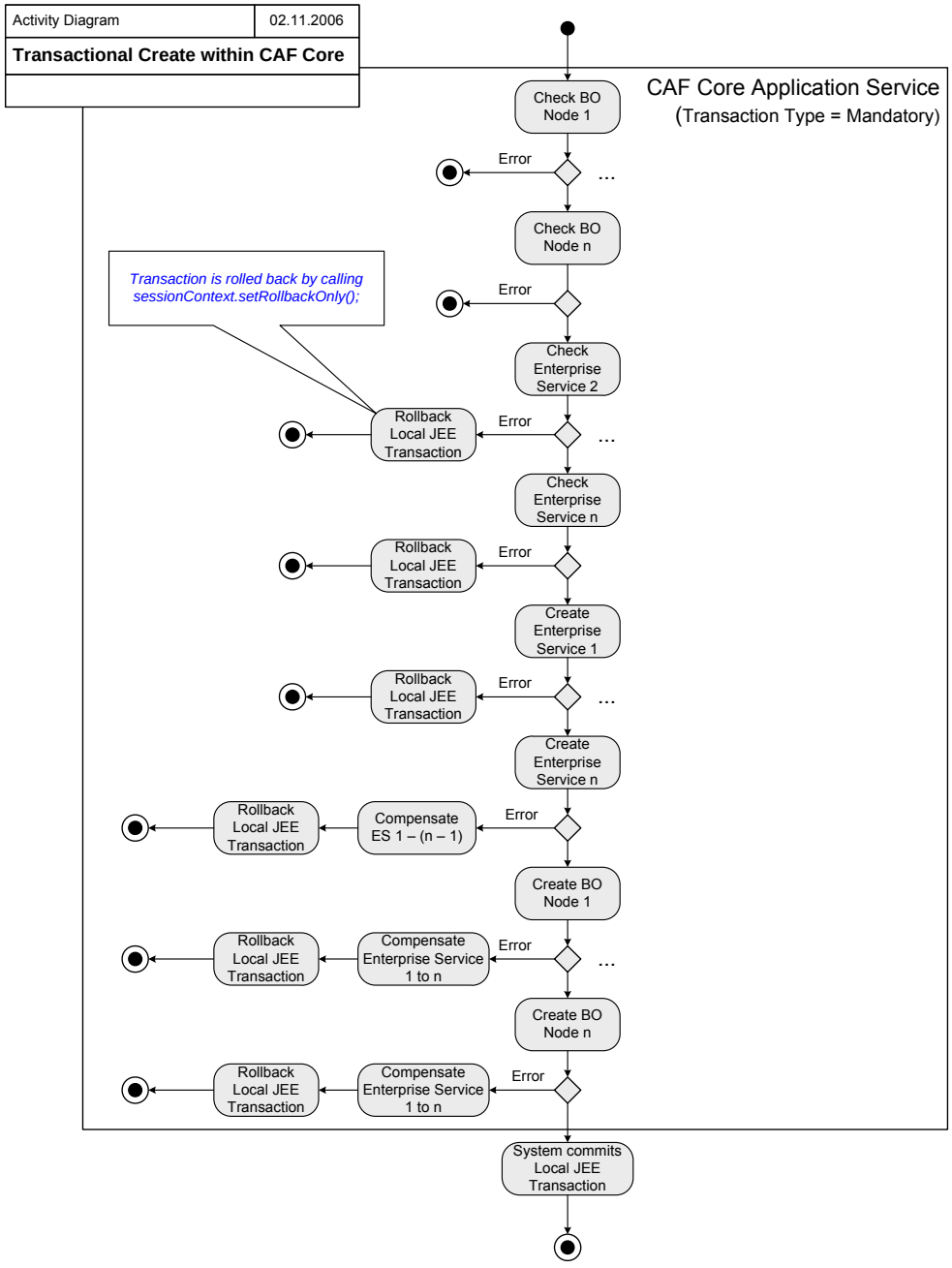


It is not possible to follow this sequence when creating a business object in the backend and storing its backend-generated key in the entity. Change of the sequence or client-side key generation is needed in this case.

☑ **Implement the Logic within a Create Application Service in the following sequence:**

1. **Local Checks**
2. **Remote Checks by Enterprise Services (if more than one Update Enterprise Service is called)**
3. **Remote Creates by Enterprise Services**

4. Local Creates of CAF Business Objects



☑ **Call compensate Enterprise Services to "rollback" modifying service calls**

? **iCOD Architecture further evaluates the usage of compensate Enterprise Services**

☑ **Call sessionContext.setRollbackOnly() to rollback the local J2EE transaction**

Details on Compensation Services can be found in this presentation. https://portal.wdf.sap.corp/irj/go/km/docs/room_project/cm_stores/documents/workspaces/9163c421-9248-2810-cdbe-aa3356d48314/Architecture/Material/ESA%20Service%20Cut%20-%20Compensation%20Services.ppt

6.5 Error/Exception handling

Enterprise Services distinguish between error messages (including messages of type Information, Success and Warnings) and exceptions.

- Exceptions shall be used in situations which normally should not happen and where end users do not have the possibility to correct the situation by any user action. Typical exception situations are database inconsistencies, invalid customizing settings, code errors, …
- Error messages are used when the end users can correct the situation for example by entering different values or by waiting until an object is no longer locked. Typical error situations are application errors and locking problems.

ABAP messages of type A and X are not supported. Exceptions have to be used instead of them.

There are two different ways to indicate error messages and exceptions:
1. Error messages (including messages of type I, W and S) are passed back as part of the output message of a service. All output messages does have the following rough structure:

<message header data (GDT BusinessDocumentMessageHeader)>
<message specific data (GDT)>
….
<Log (GDT Log)>
….
</message specific data>

Log (see GDT Log in the GDT Catalog) can contain messages of type E, W, I, S. It does have the following structure:

CDT		Cat.	Object Class	Property Qualifier	Property	Representation/ Association	Type	Type Name	Len.	Card.	Remarks
Log			Log			Details					
	MaximumLogItemSeverityCode	E	Log	Maximum	LogItem Severity	Code	GDT	LogItemSeverityCode		0..1	
	Item	E	Log		Item	Details	GDT	LogItem		1..n	

6.251.5 Detailed Description and Value Ranges

MaximumLogItemSeverityCode	Coded representation of the maximum severity of a log message in a given log.
Item	Individual log message (see GDT LogItem).

LogItem (see GDT LogItem in the GDT Catalog) does have the following structure:

GDT		Cat.	Object Class	Property	Representation/ Association	Type	Type Name	Len.	Card.	Remarks
LogItem					Details					
	TypeID	E	Log Item	Type Identification	Identifier	CCT	Identifier	1..40	0..1	Restricted
	SeverityCode	E	Log Item	Severity	Code	GDT	LogItemSeverityCode		0..1	
	Note	E	Log Item	Note	Note	GDT	Note	1..200	1	Restricted
	WebAddress	E	Log Item	WebAddress	WebAddress	CCT	WebAddress		0..1	

6.266.5 Detailed Description and Value Ranges

TypeID Unique identification of the type of a log entry (within the application that generates the log).
For example, when a catalog is published, a log can be generated containing several items concerning the successful publication of a catalog item. Since these log entries are similar, they all have the same TypeID, although the respective catalog items are inserted dynamically in a text pattern that corresponds to the message type.
The LogItemTypeID must not be confused with sequential numbering for the messages in a log.
The LogItemTypeID does not require any of the attributes listed for the CCT Identifier, since these are taken from the context.

SeverityCode The severity of the log message.

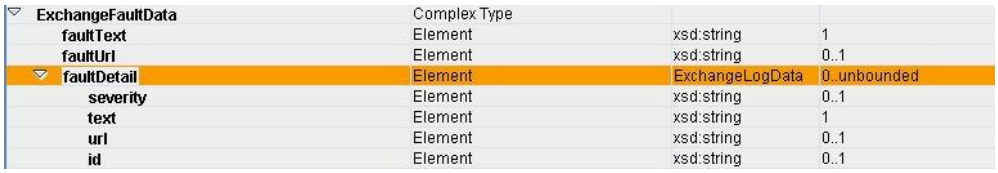
Note A short text for the log message.
The LogItemNote restricts the length permitted in the GDT Note.

WebAddress The address for a document available on the Internet that contains more information about the log entry.
The only URI schemas permitted are "http" and "https."

The LogItemSeverityCode is the coded representation of the severity in a log message on the execution of an application.
The allowed values of the SeverityCode are described in this table:

Code	Name	Description
1	Information	Notification of the execution of an application or an application step if no errors or error possibilities have occurred.
2	Warning	Warning of the possibility of an error or an error source in the execution of an application or an application step. The respective result is to be viewed with reservation.
3	Error	Notification of the occurrence of an error during the execution of an application or an application step - usually with a more precise description of the type of error.
4	Abort	Notification of a premature or unforeseen termination of the execution of an application.

2. Exceptions are indicated by XI fault message types. This means in case of an exception situation in the backend a common XI fault message is raised. XI fault messages types does always have the structure ExchangeFaultData which includes ExchangeLogData



ExchangeFaultData	Complex Type		
faultText	Element	xsd:string	1
faultUrl	Element	xsd:string	0..1
faultDetail	Element	ExchangeLogData	0..unbounded
severity	Element	xsd:string	0..1
text	Element	xsd:string	1
url	Element	xsd:string	0..1
id	Element	xsd:string	0..1

6.6 Value Help

☑ **Request an Query <BO>, Query <Configuration Object> or Specific Value Help Service for <BO> service, if you need more than fixed value list for a GDT**

Enterprise Services will provide two kinds of value help services:

- A single generic context-free service operation to retrieve a value list for Domains with fixed values, where you pass in a GDT as parameter and get back one ID field and one language dependent text field.
- If more complex parameters are needed a specific service has to be requested.

Technical Details about the design of the generic service and its availability are still open.

? **iCOD Architecture will evaluate**

6.7 Metadata

Currently it is discussed, how a solution for retrieval of metadata for a service can be provided. Metadata can be for example the label texts of a GDT to be displayed in the UI.

? **iCOD Architecture will evaluate**

6.8 Default Values for Service Calls

BAPIs / RFCs and Enterprise services may have mandatory parameters that composites do not „want" to expose to the End User within a UI.

✘ **No hard-coding of default values is allowed**
Exception:

- **Enterprise Services: If the possible entries of the parameter are specified as a code list within GDT-Catalog**
- **BAPIs/RFCs: If the possible entries of the parameter is specified within a system table or as a domain fixed values in the Backend**

☑ **Make sure that all default values you wish to hard-code can really be fixed from a business point of view. In doubt, make them configurable**

☑ **In these exceptional cases: Use constants to refer to these default values**

☑ **All other default values have to be configurable**

7 Technology Selection for the Business Logic Layer

☑ **Use CAF Core to implement the Business Logic**

☑ **It is allowed to add/use 'plain' EJB 3.0/JPA functionality inside CAF projects which is not (yet) supported or for workarounds. Talk to iCOD Architecture in such cases beforehand.**

For ICOD composites, CAF Core is the default technology in order to showcase model-driven development also on business logic layer.

7.1 Overview

CAF Core provides the ability to import services provided by the backend systems as external services, model Business Objects and implement custom logic in Java as application services.

Main Advantages:

- Modelling of Business Objects and generation of the persistency, CRUD- and optionally Query methods for this model.
- Modelling of services incl. the data types needed for their signatures
- Exposing the logic as (web) services to the user interface and/or process layer
- Authorization concept for Business Objects and services
- Integration to KM (only for Document access)

Main Disadvantages:

- Support of external data access other than trough RFC/BAPI or web service (e.g. database access): Only via custom code in application service operations
- No Business Objects with distributed persistency
- No integrated concept for the product standards archiving and configuration yet
- No extensibility concept (to come not before NW CE 2.0).
- No integration to BI (planned for CE 7.1.1), TREX and Collaboration

7.2 Business objects

A Business Object (BO) in CAF Core is essentially a data persistency model, defined by a set of attributes and relationships to other Business Objects, the lifecycle methods Create, Read, Update, Delete (CRUD), each working on a single Business Object instance per call, and query methods to retrieve multiple instances based on search conditions.

Regarding persistency, you actually have two options: remote and local.

Remote persistency means the actual Business Object instances reside in an external system, which has to be accessed by means of so-called External Services mapped to the CRUD and query methods of the CAF Core BO (hence, such CAF Core BOs are called “Remote Business Objects”). As of today, CAF Core offers the possibility to either connect to RFCs or to document-style Web Services via External Services.

Local persistency means CAF Core will create an own database model for you that reflects the structure of your BO, which is therefore called “Local Business Object” (e.g. simple attributes with cardinality 0..1 or 1..1 will be reflected by database table columns, whereas cardinality 0..n or 1..n, as well as a reference to another Business Object will lead to an additional table linked via foreign key relationship).

Moreover, there is a special persistency type “Custom”, which basically allows you to redefine the out-of-the-box CRUD methods, i.e. you can implement your own persistency logic that way.

When creating Business Objects in CAF Core, please keep in mind then following rules:

✘ **Don’t use Remote Business Objects**

Remote Business Objects have the following disadvantages:

- only query and CRUD operations with quite strict signatures allowed, which makes mapping to complex signatures of Enterprise Services difficult/impossible
- local persistency footprint for each data record read from the remote/backend system via the mapped service, which means sub-optimal performance
- at this time, there seems to be no clear benefit over an application service operation mapped to an External Service, therefore they are a somewhat redundant concept.

☑ **Only use local Business Objects to store data existing in your Composite**

This means: if you have to persist some data specific to your composite, CAF Core local BOs are to be used for that.

☑ **Handle potential lock conflicts with local Business Objects.**

If there is an optimistic lock conflict (which will occur when you try to update a BO based on 'dirty read' data, i.e. by a data record older than the most currently committed version), CAF Core will throw a CAFOptimisticLockException. In this case, you'll typically have to notice the user that the data submitted by him is based on an outdated state of the respective BO and present him the current state, so he can merge his entered data with that.

In case a pessimistic lock occurs (which can only be the case if a pessimistic lock for a BO instance has been requested pro-actively, i.e. there is no inherent "automatism" like for optimistic locks), the framework will throw a CAFPessmisticLockException. In this case, you can only presently a message to the user that the respective object is locked at the moment and he has to re-try later. In general

- o pessimistic locks should be set/tested before data entry by a user.
- o pessimistic locks should only be set if it's clear that optimistic locks won't be sufficient for data integrity, which should only be the case rarely (but you should always think about it carefully!). Talk to iCOD architecture before using pessimistic locks.

Cf. chapter 'Transaction handling' for more details on optimistic locking.

Also notice that optimistic locking only works with authenticated users, i.e. anonymous requests will never notice lock conflicts.

✘ **XML-DAS is not part of NW CE 7.1 SP 0 - 3 delivery, so business scenarios which may potentially be affected by Product Standard "Archiving" are out of scope for CE 7.1 SP 0 - 3.**

XML-DAS is the de-facto standard archiving solution on NW WAS Java and we should refrain from implementing some custom work around for its absence in CE 7.1 SP 0 - 3 by ourselves.

More details on Product Standard "Archiving" can be found in the chapter Data Archiving

✘ **Don't create an own implementation of your Business Object. In particular, don't implement own persistency logic.**

As said, in NW CE it's possible to overwrite the generated CRUD methods of a Business Object.

However, we try to keep all custom coding in Application Services because of consistency reasons and to reduce potential error sources (like forgetting optimistic lock conflict checks, permission checks, tracing CRUD method calls, …). Moreover, it would reduce the model-driven character of our Business Logic Layer in general.

If you cannot circumvent custom persistency logic by all means, talk to iCOD architecture beforehand.

- **Don't directly expose Business Object operations to the upper layers of your composite.**

  Main reasons:

  - Fixed/inflexible operation signatures, partly not complying with our needs/requirements towards service consumption (e.g. you cannot add a result sub-structure adhering to GDT Log as needed by our error handling concept).
  - No out-of-the-box possibility to restrict result size for query operations (would only be possible by creating a custom BO implementation, but we want to refrain from that, cf. last point). Since CAF also doesn't support some kind of paging mechanism (because of the statelessness of its services), this may lead to huge data sets transported to the UI at once.

- **Avoid pure external data replication**

  Replication of persistent data between a Composite Application and its Backend systems adds complexity since the (semantically) same instance of a business object can be changed independently at several places, thus risking out-of-sync situations, i.e. depending on the respective scenario more or less severe inconsistencies.

  Try to avoid this kind of replication whenever possible. If you feel there is no other option, contact iCOD architecture beforehand.

  Ways to avoid the need for data replication:

  - a process design creating the Business Objects in the Backend system not before the process within the Composite is finished
  - Storing only references/keys to the Business Objects in the Backend Systems and not persisting redundantly their data in the Composite

  Nevertheless there may be cases where data replication is sensible:

  - Scenarios where you'd otherwise would need to perform a huge number of backend accesses or you would have to transport a large amount of data again and again (remember CAF Core is stateless, so keeping this data in session context is not possible), leading to unacceptable performance/scalability
  - Collaboration Objects, which are a representation of a backend object, but which shall not be automatically updated with every change within the backend (i.e. where out-of-sync situations are uncritical).

  Generally, the following points apply to data replication:

  - In case of rather short-living but high-amount transactional data (e.g. purchase orders), replication is most problematic since frequent re-syncs are not feasible on the one hand but out-of-sync situations are much more likely on the other. Therefore, try to avoid data replication by all means in this case.
  - For long-living master data, out-of-sync situations are not that likely, so replication might be allowed and designed on a case-by-case basis.

7.3 Application Services

Application services allow for custom programming, e.g. to transform/combine data from local Business Objects, External services and other Application services or performing any kind of custom business logic.

- ☑ **Use Application Service operations to implement the business logic which is specific to the composite**
- ☑ **Adhere to the SUN Java code conventions for your custom coding which can be found under http://java.sun.com/docs/codeconv/**

When working with application services, the following rules apply:

- ☑ **Put simple business logic directly into an Application service operation**

  Under the "Implementation" tabstrip of your Application Service, click the <Application service name>BeanImpl.java link to create a class deriving from an abstract generated class containing the CAF Core specific parts. In the created class, you can override the methods corresponding to the operations of your Application service then.

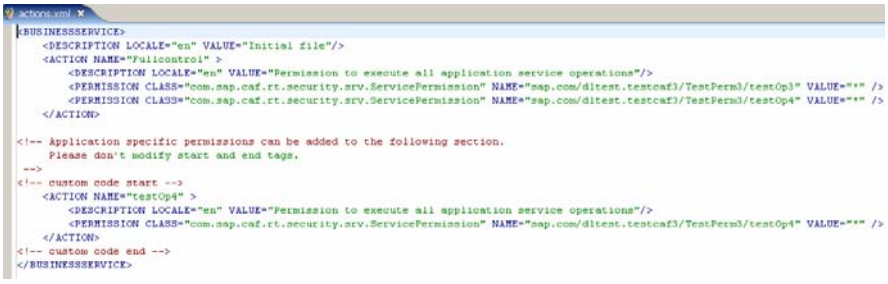
- ☑ **If business logic gets more complicated, distribute (parts of) it to additional methods and/or classes to get a clearer design**

  Just create the needed additional classes/packages under the src folder of the ejbmodule DC created for your CAF project. They will be added to the build-result of the corresponding ear project on the next build then.

- ☑ **Clarify with your product owner which operations need authorization checks**
- ☑ **In case there are such operations: make sure authorizations can be configured per operation**

  To enable an administrator to assign different authorized roles to different App. Service operations in a modification-free way, you have to do the following:

  - In the permission DC of your CAF project, you can find a file named "actions.xml" located in folder /src. By default, this file contains one action entry "fullcontrol" having one permission per Application Service operation with permission checks enabled.
  - After deploying the permission DC's SDA file [you have to do that explicitly, i.e. it won't be automatically deployed from Composite Application perspective with your CAF project at the time being], you can find the mentioned "fullcontrol" action in the User Management tool. When assigning this action to a UME role, each user assigned to that role will be allowed to call all operations of your App. Service. This is typically undesirable.
  - Therefore, you have to add an action per operation to the "custom code" section of the actions.xml. Use the "fullcontrol" action as a template and name your action exactly like the operation it is referring to:



After re-building and re-deploying your permission DC, you should be able to assign the new action to an UME role in the User Management tool, allowing the respective users only to execute the one referred operation.

☑ **Enable all Application services as Web Service if they are to be accessed by Process Logic and/or UI layer.**

In principle, there are the following options to consume an Application service operation from UI and/or process layer:

- Enable the Application service as Web Service. This can be achieved by choosing “Expose service as Web Service” from the Application service context menu in the CAF Composite Application Explorer view, which will start a wizard where you can add all (applicable) operations to the Web Service interface.
- JNDI lookup & direct call of EJB interface (since each Application Service is an EJB Session Bean underneath).

For iCOD, we want to use the **first option only**. Besides further underlining the composite idea/Enterprise SOA story, this enables Visual Composer to consume Application Services, which is not possible in another way at the time being.

☑ **Application services operations should access other Application services operations by EJB method calls (not through web service interface)**

☑ **Web services based on Application Services will not be PICed.**

However, by adhering to ICOD Naming Conventions for Application Services which are leaned towards the respective ESA guidelines, we are going to get web service interfaces similar to PICed ESA services automatically.

☑ **Provide all info, warning and error messages in a sub-structure of your Application Service operation output parameter, adhering to the GDT ‘Log’,**

Re-use the Log structure type generated when importing an Enterprise Service as External Service (typically, all Enterprise Services use the GDT Log in their signatures).

Temporarily import some Enterprise Service as External Service to get this Log type, even if you actually don’t need to connect to a backend service (delete the External Service and all unnecessary generated types again in such (probably rare) case).

✘ **Don’t throw exceptions in Application Service operations exposed as Web Service operations. Use GDT ‘Log’ with SeverityCode 3 (Error) or 4 (Abort) instead.**

Note that exceptions would be communicated as SOAP Fault Messages, which cannot be handled as desired by us in some cases (for example, you can only react with Process Exceptions to SOAP Fault Messages in GP, cf. chapter “Technology Selection for Process Layer”).

Moreover, CAF Core only supports passing a string as payload of SOAP Fault Messages for App. Service-based Web Services, whereas GDT Log also allows for passing additional data (e.g. WebAddress pointing to some detail information on the error).

☑ **Only add operations of type 'CUSTOM' with proper signature to Web Service interface. Cut these operations in a coarse granular manner according to the needs of your composite application.**

"Proper" means you should try to stick to Enterprise Service guidelines as close as possible. Especially, try to re-use (parts of) those types created when importing Enterprise Services as External Services.

"Coarse granular" design means to avoid distributing logic that should belong to one transaction over several operations if possible. Note that all accesses to Business Objects performed from one Application Service operation can be executed in the context of one JEE transaction (and hence, be rolled back if necessary).

Enterprise Services are built for reuse per Business Object applying standardized patterns and not for a single consumer only. In contrast to that the Application Service Operations of the composite are built dedicated for its User Interface or Process Logic only. So you should take the freedom to cut those according to the needs of your applications potentially crossing BO boundaries.

☑ **Make use of logging and tracing**

http://aiokeh.wdf.sap.corp:50000/SAPIKS2/contentShow.sap?_CLASS=IWB_EXTHLP&TMP_IWB_TASK=PREVIEW&_LOIO=F4D1E6418B7ECA17E10000000A155106&_SLOIO=6203AE42E5ADCD6AE10000000A155106&RELEASE=712&LANGUAGE=EN&_SEQNUM=0&_SCLASS=IWB_STRUCT gives an overview of what should be logged and traced and how to achieve this with the SAP Logging API. Summing it up:

- Traces are primarily targeted towards developers, whereas logs are targeted towards administrators.
- Log messages are to be emitted via class com.sap.tc.logging.Category (since Categories describe the origin of messages from a more semantical point of view, e.g. '*/System/Database'*), trace messages are to be emitted via com.sap.tc.logging.Location (since Locations describe the origin of messages from a more technical point of view, like e.g. a fully qualified Java class name).
- The emission methods of com.sap.tc.logging.Category all take an additional input parameter 'loc' of type com.sap.tc.logging.Location ➔ log messages are also traced at the same time, given that their priority is not below the threshold of the passed Location (which is 'Warning' by default, so e.g. info messages will not go to the trace).
- Categories as well as Locations are maintained (but not created) in NW Administrator under 'Problem Management' → 'Logs and Traces' → 'Log Configuration'. Both are organized in hierarchies, inheriting settings (especially destination and maximum sizes) from their respective parent nodes.
- By using the methods Category.getCategory or Location.getLocation (overloaded with various signatures), you can instantiate existing Categories/Locations or create new ones ad-hoc implicitly (if the specified one doesn't exist yet).
- Rules for Category and Location selection:

☑ **Use Location.getLocation(<Current class name>.class) to instantiate a tracing location**

This will use the fully qualified name of the given class to retrieve or create a location hierarchy along the package path as needed, giving you the leaf location. This will be fine as long as your fully qualified classname complies to iCOD naming guidelines (→ com.sap.is.cmp.<composite name abbreviation>.<arbitrary part>.<class name>)

☑ **Use Category.getCategory(Category.APPLICATIONS, <composite name abbreviation>) to instantiate a logging category**

The new category will hence be created under root category 'APPLICATIONS', which is present by default after NW CE installation.

- For when to use what severity level (and hence, what emission method), cf.

  http://bis.wdf.sap.corp:1080/twiki/bin/view/Techdev/LoggingFAQ#q1-5

  Examples of things that should be logged:

  - Access to Local BOs (BO instance(s) successfully created, read, updated, deleted or failure), permission check results (success, failure), access to External Services (which one, success, failure), KM access (accessed resource info, success, failure), important/business-relevant state changes (e.g. you should not only log "BO xy updated", "Service z called", but "Changes to Sales Order xy have been booked successfully, additional documents D1, …, Dn have been linked".)

  Examples of things that should be traced only:

  - Entering and leaving methods (note: CAF framework will do this for Appl. service operations automatically using an Interceptor – cf. https://www.sdn.sap.com/irj/sdn/weblogs?blog=/pub/wlg/5803 for more details on Interceptors in EJB 3.0, since it's an elegant & recommendable concept), creating & releasing locks if you're using pessimistic locking, debug information like values/value changes of important variables, variables set to default values implicitly, retries on calling External Services (e.g. when calling idempotent web services – only log definitive failure, but trace each retry of the call), …

For more details on the subject in general cf.

http://bis.wdf.sap.corp:1080/twiki/bin/view/Techdev/JavaMonitoringAndManagement

Contact person for unresolvable questions is Nikola Marchev.

7.4 External Services

☑ **Only Web Service based External Services are allowed in the delivery/GA version of ICOD composites**

This rule applies since one of the basic requirements towards the ICOD program is to showcase and prove the enterprise SOA story.

☑ **An External Service is to be mapped to one Web Service Destination created in NW Administrator**

Cf. http://aiokeh.wdf.sap.corp:50000/SAPIKS2/contentShow.sap?_SCLASS=IWB_STRUCT&_SLOIO=499E4E20BF004717B031E948DBBC2AAA&TMP_IWB_TASK=PREVIEW&RELEASE=712&LANGUAGE=EN&_SEQNUM=77&_LOIO=586BEF1807C441859DE1AA2BBC167728&_CLASS=IWB_EXTHLP on how to do this destination mapping.

If an External Service call should be mapped to multiple endpoints, this has to be done by means of XI on customer side.

Possible scenario:

A customer desires one of our composites to read and combine data for business object "Supplier" both from an ERP and a SRM system.

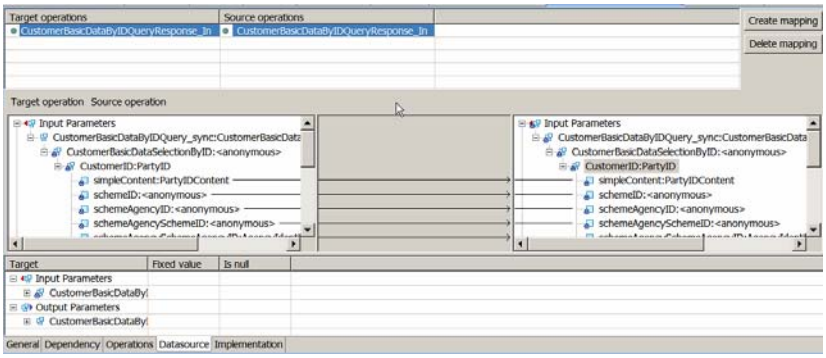
By default, our composite would do only one service request to get the supplier data (from an ERP system typically) however. So the customer can modify our composite, extend the backend abstraction layer by a custom connectivity DC or use XI to split the original request into two new ones as well as combining their results, where the latter one is maybe the "most elegant" solution if he already has an XI system in his landscape.

☑ **External Services may only be accessed out of Application Services using mapped operations**

☑ **Re-use types generated on External Service import when defining Application Service operations to be mapped.**

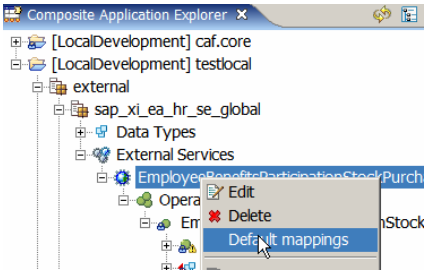
Note that programmatic calls of External Services are not supported in NW CE anymore (i.e. there is no proxy class generated for an External Service). Instead, an operation of an External Service can only be accessed through a mapped Application Service operation.

To create such mapping, uncheck the "Implemented" checkbox of the Application Service operation and switch to the "Datasource" tabstrip to assign the desired External Service operation. By clicking this assignment, you'll get a view where you can perform the actual parameter mapping.



Note you can re-use types CAF design time generates on External Service import to define Application Service operations. This will facilitate the mapping to the respective External Service in many cases.

It is also possible to create fully mapped application service operations ad-hoc from an External Service, hence giving you an equivalent to a type proxy.



✘ **There is no standard way for "pushing" data to UI layer as of today. Don't try to implement one on your own.**

This means new/updated data has to be requested by an UI from the Business Logic Layer pro-actively. Pushing data to the application UI, which means changing displayed data without explicit (i.e. user trigger) or implicit (e.g. by means of WD TimedTrigger Control) UI event is not feasible at the time being. You should also refrain from investing into a custom implementation of such push mechanism, e.g. by "service-enabling" your UI layer and using such service via an External Service.

? **iCOD architecture will continue to examine/drive ways for event consumption from CAF Core**

7.5 Persistency of unstructured content

Unstructured content, which essentially means files like PDFs, Word documents, Excel sheets, etc., is to be stored by means of SAP NW Knowledge Management and Collaboration (KMC).

☑ **Only use Knowledge Management and Collaboration (KMC) of SAP Enterprise Portal for persistency of unstructured content specific to the composite**

☑ **Store unstructured content related to a backend Business Object in Document Management System (DMS), given a DMS integration for this BO is available.**

This is done by means of specific Enterprise Services, cf. ERP Process Component "Document Management":

http://erp.esworkplace.sap.com/socoview(bD1lbiZjPTgwMCZkPW1pbg==)/smdisplay.asp?id=C99184DA624A11DA36BB000F20DAC9EF&fragID=&packageid=DBBB6D8AA3B382F191E0000F20F64781&context=&iv=

Note this requires the respective backend Business Object instance to exist before you can store content related to it in DMS.

☑ **Regarding KMC access, use the out-of-the-box CAF Core-KMC integration if sufficient**

For persistency of files, this means using the standard BO "Document" and the standard Application Service "DocContent".

Cf. http://aiokeh.wdf.sap.corp:50000/SAPIKS2/contentShow.sap?_SCLASS=IWB_STRUCT&_SLOIO=499E4E20BF004717B031E948DBBC2AAA&TMP_IWB_TASK=PREVIEW2&RELEASE=716&LANGUAGE=EN&_SEQNUM=63&_LOIO=1CAA6EAF373C416CBFA416665887AB1C&_CLASS=IWB_EXTHLP for details on their usage.

Note that other CAF Core-KMC integration artifacts (e.g. collaboration entities like Discussion, DiscussionRoom, Topic, …) don't work as of CE 7.1 SP 0 - 3.

☑ **Use WebDAV access to KMC repositories as fallback if CAF Core-KMC integration doesn't offer the needed functionality. Inform ICOD Architecture in such cases.**

By means of using the WebDAV standard, you can access KMC repositories via HTTP to retrieve, create & update documents. Cf.

http://help.sap.com/saphelp_nw2004s/helpdata/en/4e/05f44941f4bd4890c9e54015810800/frameset.htm

for some more details.

✘ **Don't use KMC APIs or KMC Web Services directly.**

KMC (and hence, the KMC API) is not planned to be part of the NW CE delivery, so KMC access is only supported within a Federated Scenario for our composites, having KMC residing on a NW 04s instance remotely accessed from the composite on a NW CE instance.

For this remote access, it's not recommendable to use KMC Web Services directly since they will probably have incompatible signatures in their 04s and NY versions, so it would require additional custom configuration for a composite to differentiate which version is used (with CAF, this is going to be a central configuration setting).

7.6 Extensibility

Extensibility related capabilities of SAP NetWeaver modelling and development tools have been evaluated by iCOD Architecture. The findings can be found in this presentation
https://portal.wdf.sap.corp/irj/go/km/docs/room_project/cm_stores/documents/workspaces/9163c421-9248-2810-cdbe-aa3356d48314/Architecture/Material/Extensibility_and_Supportability.ppt

✘ **As of today, CAF Core offers no suitable extensibility concept. Don't implement your own.**

☑ **If your composite has a specific use case/requirement in terms of extensibility, contact ICOD architecture to discuss a solution.**

For a later version of NW CE (probably not before 2.0), it is planned to have a possibility to extend a CAF Core project based on a 'Java-like' inheritance concept, meaning you can keep/inherit some parts of the parent project and overwrite others, comparable to the way you can do it with Java classes.

However, only one-level inheritance is going to be supported.

? **It has yet to be clarified how the requirement: SAP Delivery → ISV extension → Customer extension can be fulfilled with the planned extensibility concept.**

7.7 Business configuration

✘ **As of today, CAF Core offers no concept for application-specific configuration**

☑ **Use Local BOs as custom configuration persistency**

☑ **Build your own UI using Web Dynpro to maintain configuration data based on these BOs**

The assumption is that configuration UIs should typically be less sophisticated than normal application UIs, hence using VC should actually be fine in most cases. However, in CE 7.1 SP 0 - 3, we don't intend to use VC at all, so you have to go for a Web Dynpro based approach here, too.

☑ **For delivery of default configuration, you have to import BO content to NWDI as follows**:

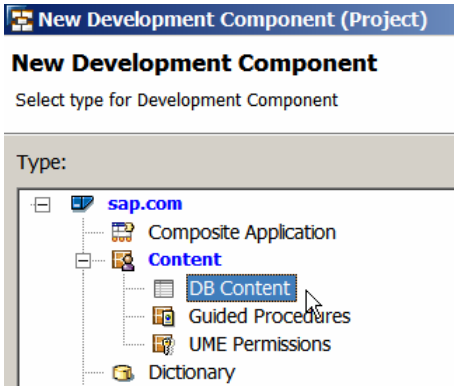
Go to "Content transport for local business objects" under CAF Administrative Tools on your WebAS (cf. http://<your WebAS host>:<port>/caf)

- In the left-hand tree, drill down to your BO and use one of its query operations (e.g. findAll) to get a list persisted entities
- Mark all desired persisted entities and press "Add Selected to Transport"
- Go to next step "Inspect selected content" by pressing the respective button. Afterwards, directly go on to "Transport Content".
- Choose "Create transport file" and download it as XML file afterwards by clicking the respective link → you'll get a file named "dbcontent.xml".
- In your IDE, create a new DC of type "DBContent" and overwrite the empty dbcontent.xml file in its src folder with the file you obtained in the previous step.

Note: the DC type "DBContent" is only available SAP internally and has to be added to your IDE explicitly. To do so, extract the archive

\\dwdf051\isidm\public\90_Misc\40_IDEPatches\DBContent_DC_Type.rar

to your IDE installation directory. After restarting the IDE, you should be able to find the respective DC type in the wizard under the "Content" node:



? **Since the DBContent DC type is only available SAP internally, an ISV won't be able to deliver database content via NWDI until further notice.**

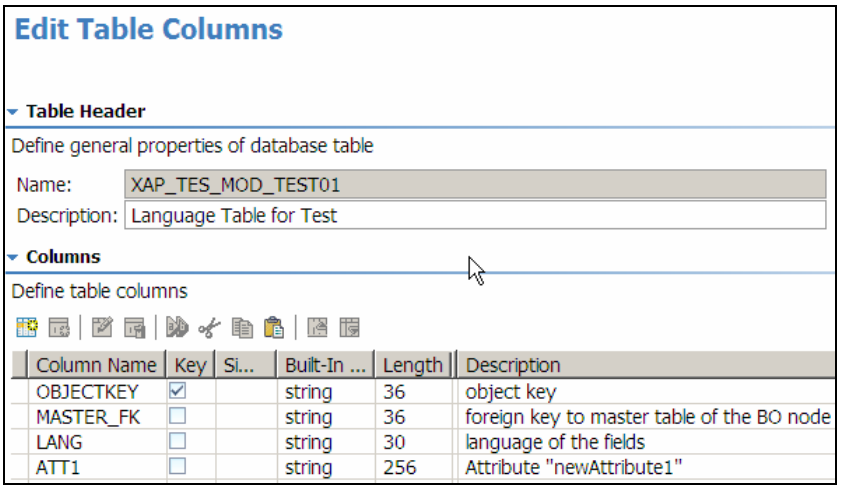
It is planned to overcome this restriction, but details are still pending (however, availability in CE 7.1 SP 0 - 3 is out of scope).

7.8 Internationalization

✘ **Language dependent attributes of BO content cannot be translated by this process. You will have to translate them by yourself if you need to ship such BO content.**

By defining a BO attribute to be language dependent, another database table is created under the hood linked via foreign key to the actual BO data table. This table may contain one

dedicated entry for each desired language. However, its content cannot be integrated into the standard NWDI translation process as of today.



For CE 7.1.1, it's planned to get rid of this restriction.

☑ **Custom texts intended for display to the users have to be externalized into a .properties file and converted into a .xlf file by means of the S2X tool of the IDE.**

For each Java class, you can use "Source -> Externalize Strings…" from the context menu of the IDE Java editor to create a .properties resource bundle file containing the string literals occurring in your class source code (you can select which ones to externalize) as well as an access class for the resource bundle. Moreover, this action will refactor your code to replace the string literals by respective method calls to your resource bundle access class.

From the context menu of the generated .properties resource bundle file, you can choose "I18N Tools -> Run S2XTool" to start the S2X editor for creating an .xlf file based on your resource bundle.

Also cf. chapter "Globalization" for more details.

☑ **All rules given by chapter "Globalization" have to be taken into account**

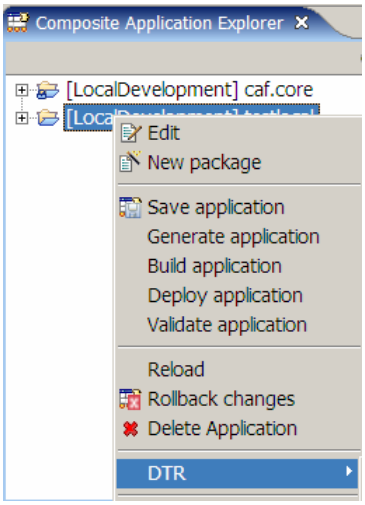
7.9 Naming guidelines

☑ **The naming guideline recommendations for CAF Core Objects are described in the iCOD Naming Guidelines**

7.10 Working with CAF Core Development Components

☑ **Use "Add local changes" function before checkin to make sure all needed files are added to your activity.**

You will find this function under the "DTR" of your CAF project's context menu.



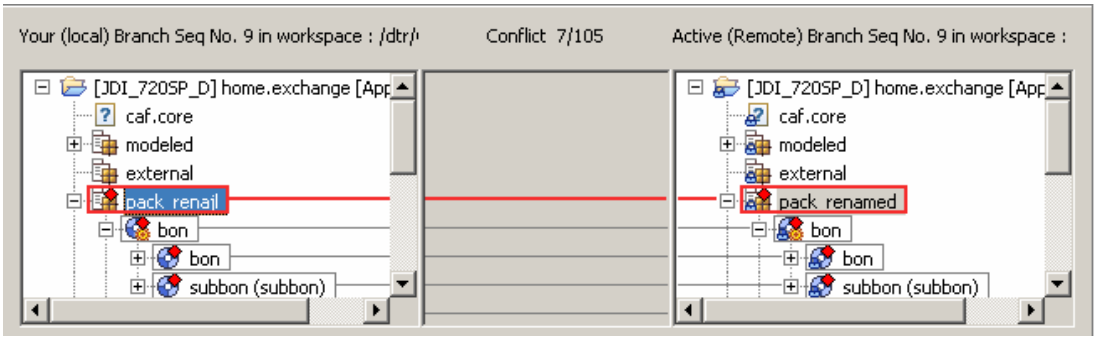
☑ **Only sync CAF projects either from DTR perspective using Forced Sync or from CAF Composite App. Explorer view under "DTR" project context menu entry.**

This will prevent you from getting a lot of unnecessary sync failure messages caused by MMR files that are in edit mode locally (CAF keeps them in edit mode by default). Since they are overwritten at each re-generation of your CAF project anyway, it's no problem they're overwritten by a forced sync.

As said, this rule doesn't include the first time sync, which you do in default way, i.e. by selecting the 5 DCs of your CAF project in Development Configurations perspective and choosing 'Create project' from the context menu.

☑ **Team development is supported now with CAF Core projects due to a new merge tool that allows for resolving concurrent modifications on model artifacts.**

A potential model conflict will be indicated on your CAF project's design-time model file ca_model.cafmm if you try to checkin your DTR activity. You can then start the model merge tool from the context menu of the respective file activity entry under "Resolve Conflict" → "CAF Resolve". For look&feel sample, cf. screenshot below.



☑ **Pay attention that the CAF merge tool is only suited for the CAF model file ca_model.cafmm.**

Concurrent modifications to Java source files have to be resolved by the standard text-based merge tool that you can start from the context menu of the respective file activity entry under “Resolve Conflict” → “Merge”.

✘ **Despite that there is a model merge tool now, try to avoid conflicting modifications on your CAF project.**

This recommendation is actually quite common for software development in a team generally.

It basically means developers should agree with each other who works on what objects to prevent incompatible parallel changes to the same object (merging those is often a somewhat tedious or even tricky task a merge tool can only support to a certain extent, whereas compatible changes can be merged automatically, also with CAF merge tool).

In any case, pay attention that the merge result may not be correct, e.g. if the contributing developers have not carefully agreed on changes to objects depending from each other.

☑ **Always test merge results carefully; never just rely on their correctness.**

7.11 References

You will find quite some tutorials and How-Tos in the CAF Core RKT repository located under https://websmp105.sap-ag.de/~form/sapnet?_SHORTKEY=01100035870000671791&. (TODO add tutorials for NW CE when they become available).

Moreover, you might want to have a look at the documentation in Knowledge Warehouse (CE preview version, with many white spaces at the time being):

http://aiokeh.wdf.sap.corp:50000/SAPIKS2/contentShow.sap?TMP_IWB_TASK=PREVIEW&_CLASS=IWB_EXTHLP&_LOIO=5A03AE42E5ADCD6AE10000000A155106&_SLOIO=6203AE42E5ADCD6AE10000000A155106&LANGUAGE=EN&RELEASE=712&_SCLASS=IWB_STRUCT

8 Technology Selection for the User Interface Layer

8.1 Overview

The user interface (UI) layer provides components for human-machine interaction within the composite application. The UI of the composite runs in a web browser uses web services to retrieve and update data in the connecting layers (Business Layer and Backend Abstraction Layer).

☑ **The following technologies can be used to implement user interface**

- **Visual Composer Freestyle Modeling**
- **Adobe Interactive Form (Online/Offline Forms)**
- **WebDynpro Java Foundation**

The decision on the technology for the user interface depends on the concrete use case. The overall design approach in iCOD requires to build UI components that are simple to use and easily adaptable to the needs of the user without changing the BL. Other important requirements are easy configuration, customization and the integration with other technologies like Portal and Guided Procedure.

The evaluation of all of these criteria has led to the following guideline:

☑ **The standard UI technology used within iCOD composites is the Visual Composer Freestyle modeling**

☑ **Use Web Dynpro Java Foundation for user interfaces which are too complex for VC storyboard approach or there are technology limitations**

Check Chapter Visual Composer Freestyle Modeling for the limitations of VC modeling.

☑ **Use Adobe Interactive Form for form based (i.e. paper like) UIs or for offline processes**

In certain cases (technology limitation, design restriction) the UI can only be based on Adobe Interactive Form (AIF):

- UIs which design has to be very close to the paper-based forms which are already in use
- Data should be available in a printable or signed format
- Data should be entered offline
- Data are sent to a process participant for archiving

8.2 Overall design guidelines

There are two major type of UI pattern recognized for composites.

☑ **Guided-activity-floor-plan interface is the UI design pattern for step-by-step interface**

☑ **Object-instance-floor-plan is the UI design pattern for complex interfaces or random data access**

The guided-activity-floor-plan (implemented by WD Roadmap UI element) leads the user through the process using simple, intuitive interfaces. Each interface gives a very limited choice

for the user, which makes it less error prone. Also it makes easier to adapt it to new requirements and it is simpler to adapt its UI design to GP requirements.

Use the object-instance-floor-plan when the process cannot be determined, or it has too many choices/exceptions. It allows users to view and edit all properties of a business object, and they can navigate between different views of the object data, navigate to related objects/documents and secondary activities as well as search for other objects.

(sample screenshots are missing)

- ☑ **All interface technologies (except AIF) has to use relative layout**

Relative layout cannot render at pixel-precise positions and sizes but it results in more consistent UIs which is device-independent. Using relative layout requires different approaches from the UI designer and the developer as well. To build complex UI layout developer has to use nested VFLOW/HFLOW containers, and the so called MeltingGroup of Form layouts (available in CE 7.1.1). MeltingGroup denotes a number of controls occupying a 'cell' of a single control in a Form layout.

- ☑ **1024x768 is the standard resolution for all UIs of the composite**

This resolution is the standard for all browser based UI within SAP.

For detailed description of UI design see chapter User Interfaces Design.

8.3 General decisions

The following decisions have to be followed for all UI technologies:

- ☑ **Use Web Services to access Business Logic Layer**
- ✘ **Do not call any services which are outside of the Business Logic Layer if the Backend Abstraction Layer is implemented**

BAL encapsulates all backend and enterprise service calls, but the ‘abstract’ interface which represents all business methods is located in the Business Layer (see chapter Backend Abstraction Layer).

- ☑ **Use Logical Destinations of NetWeaver Administrator to configure WS Destinations**
- ☑ **All UI layer objects have to be put into development components (DCs) and checked into the DTR**
- ☑ **Use consistent UI rendering (HTML or Flex) per role**

In other words do not mix HTML rendering with Flex rendering for the given role in order to create consistent UI usage. Mixing WD and VC screens is still possible as long as the rendering is kept consistent.

- ☑ **Provide design that enables implicit personalization of Web Dynpro whenever possible**

Check section **Implicit Personalization**.

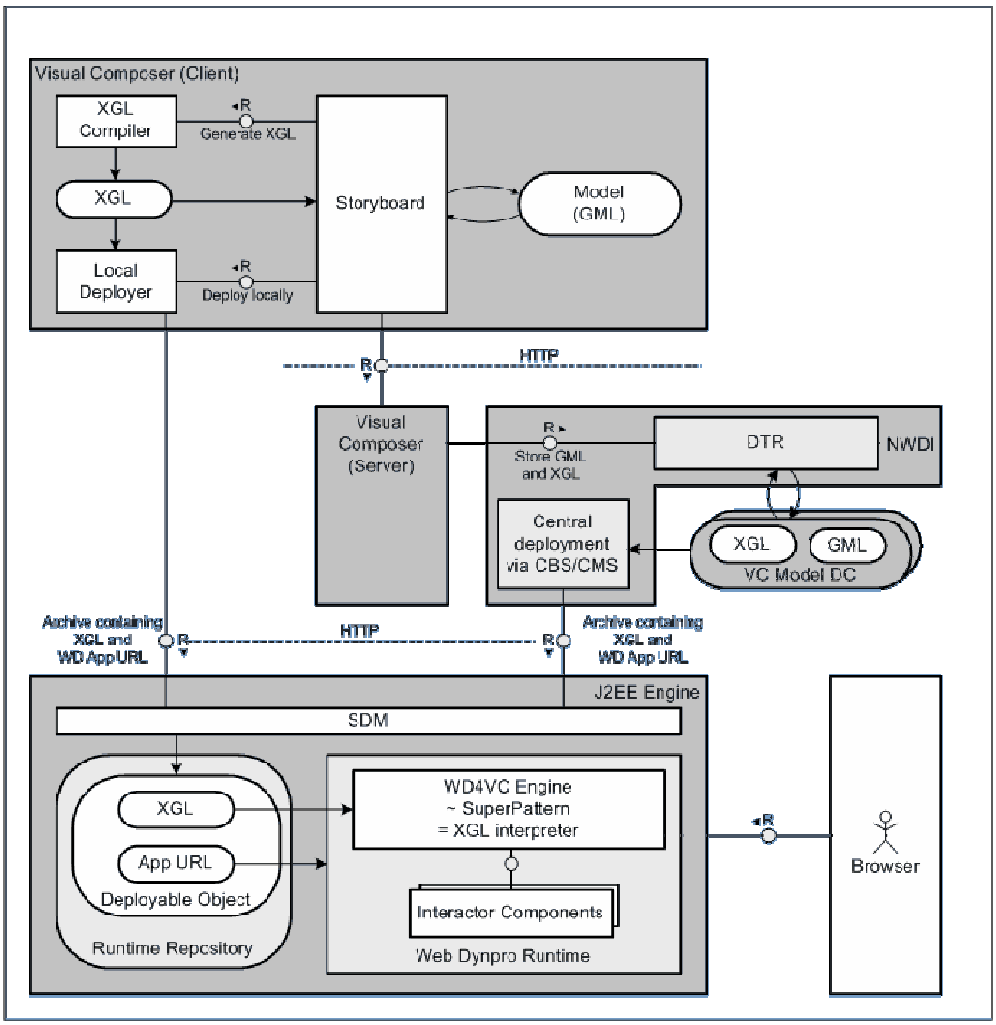
- ☑ **Always retrieve values from Business Logic Layer for dynamic or configurable value helps**

8.4 Visual Composer Freestyle Modeling

8.4.1 Limitation Note

Because of layout restrictions Freestyle Visual Composer will not be used as UI technology by iCod in CE 7.1 SP 0 - 3. A dedicated rollout will follow when VC is approved for iCod usage.

8.4.2 Overview



Visual Composer (VC) enables the fully model-driven development of UI applications. VC models are built using the VC design-time which is running inside a web browser. The heart of the VC architecture at run-time is the so called WD4VC component (see figure), which can interpret XGL files and create either Flex or Html rendering.

VC is fully compliant with the component model of SAP. New models can only be created within an SC/DC. The new DC can be saved either in the user workspace or in a shared workspace. Models can also be checked into the DTR directly or added to it later.

Main Advantages:

- High degree of Model Driven Development (no programmer skills)
- Clear separation of UI Layer from Business Logic Layer
- Can be adapted to user needs

Main Disadvantages:

- Missing UI controls like File Browser, Value Help
- VC meta-model has its own scope which means that it is more difficult to adapt to new requirements which were not foreseen by the meta-model
- The graphical model becomes unreadable pretty quickly as the complexity of the UI grows, reaching the level easily when the customer has no chance to modify it

It is not easy to define the limits of the VC modelling capabilities. One of the key indicator that the developer has reached this point, that more-and-more services are created in the Business Logic Layer which are clearly belong the UI layer.

8.4.3 Componentization

VC supports 4 different types of model but only the first 2 are relevant for the current scope of iCOD:

- **Scenario** represents reusable UI component
- **Service** represents reusable service component
- **Voice Application** and **Voice Components** are to create UI application based on voice recognition

Scenario and Service components are the basic building blocks for VC development in iCOD. VC support reuse of these components within a DC or between DCs. In fact creating a new Nested View will create automatically a new Scenario component. Components communicate between each other through Signal In and Signal Out connectors.

☑ **Use VC components whenever possible to create a modular design which is easier to maintain, update and customize**

☑ **All service calls has to be wrapped within a Service component**

Creating a Service component ensures that the model is less vulnerable to changes of the parameters of the service.

8.4.4 Backend and BL connectivity

The rules of General Decision chapter apply.

8.4.5 Process layer integration

☑ **Always create iView for both Flex and HTML rendering**

In CE 7.1 SP 0 - 3 the choice between Flex or HTML rendering is done through different URLs (see bellow). To ease the configuration of the composite create one URL iView containing the Flex rendering and on for HTML for each VC screen.

Example to access a VC screen in HTML rendering:

http://iltlvh110.tlv.sap.corp:50000/webdynpro/dispatcher/sap.com/zsoltrma_impl/Overview?sap-wd-client=SmartClient

Example to access a VC screen in Flex rendering:

http://iltlvh110.tlv.sap.corp:50000/FlexClientWAR/FlexClient.html?URL=http://iltlvh110.tlv.sap.corp:50000/webdynpro/dispatcher/sap.com/zsoltrma_impl/Overview?sap-wd-client=SmartClient

Unfortunately different

Also check the Chapter Technology Selection for the Process Layer.

8.4.6 Portal layer integration

☑ **Always create iView for both Flex and HTML rendering**

See explanation in previous Chapter Process layer integration. Also check the Chapter Technology Selection for the Portal Access Layer.

8.4.7 Value Help

Visual Composer has no support for value helps in CE 7.1 SP 0 - 3. The value help has to be implemented by the developer of the composite.

Check this presentation for some guideline:

https://portal.wdf.sap.corp/irj/go/km/docs/room_project/cm_stores/documents/workspaces/9163c421-9248-2810-cdbe-aa3356d48314/Architecture/Material/CE_VC_ValueHelp.ppt

8.4.8 Customizing / Personalization

VC models are translated to WD4VC therefore WD runtime provides the implicit personalization feature. For more details see Chapter Customizing / Personalization.

8.4.9 Translation / Internationalization

VC models are translated to WD4VC therefore WD runtime provides the translation and internationalisation feature. For more details see the WDJ chapter.

8.4.10 Extensibility

Extensibility of VC models is not supported by the tool itself. Furthermore there are no tools to merge the models neither. This means that the designer of the composite has to provide ways for the user to extend the UI functionality by making the model as modular as possible.

One solution would be to adapt extensibility techniques used by the ABAP development. (link missing)

Other way of doing extensibility is to implement extension points. VC provides model reuse within or between DCs which can be used to implement extension point through empty models.

Also VC SDK is available to build so called ‘kits’ which are the extension of the VC modeller itself to provide extra shapes for example required by other domains. Check Voice application

model types. At the moment only Netwevear team is allowed to develop kits, so before committing such a development, contact the architecture team.

☑ **Use Blackbox components of VC to implement missing UI or modelling elements**

VC will support Blackbox components in CE 7.1.1 through a tight integration of GP CO concept.

8.4.11 References

For more details check the Twiki page of WD4VC:

https://bis.wdf.sap.corp/twiki/bin/view/Sapinternal/Wd4Vc

Also have a look on the SDN forum of Visual Composer

https://forums.sdn.sap.com/forum.jspa?forumID=58&start=0

Visual Composer Veterans Guide:

https://sapmats-de.sap-ag.de/download/download.cgi?id=7GTASQ8ZGBG59FRU63TKBU8IAHSSIXUCCKC8MZ7M4HMRKTVC75

Modelers Guide:

http://aiokeh.wdf.sap.corp:50000/SAPIKS2/contentShow.sap?_SCLASS=IWB_STRUCT&_SLOIO=44D9586D3EF05F4DE10000000A11466F&TMP_IWB_TASK=PREVIEW2&RELEASE=712&LANGUAGE=EN&_SEQNUM=688&_LOIO=4504134E8A5741DEE10000000A1553F6&_CLASS=IWB_EXTHLP

8.5 WebDynpro Java Foundation (WDJ)

8.5.1 Overview

WebDynpro Java foundation (WDJ) allows creating complex user interfaces on top of web services. In contrast to the visual composer it allows more flexibility in creating UIs.

Main Advantages:

- Supports more communication protocols
- Stability and usability proven by FS-CAA, ERP Self-Services
- Clear separation of UI Layer and Business Logic Layer implied by architecture
- Supports more UI controls than VC
- Supports explicit personalisation and field extensions
- Better support of complex data type
- Enables dynamic UI development

Main Disadvantages:

- Less Model Driven Development of UI than VC
- Requires Java programmer skills

8.5.2 General guidelines

Remember that the WDJ application is only used for the UI layer. All necessary business logic or object persistence on top of the backend layer has to be implemented in the business layer.

✘ **No business logic or object persistence is allowed within Web Dynpro Java components**

☑ **Restrict yourself to the coding absolutely necessary to call the web services and create the user interface**

Therefore, the WDJ application shall only contain the WDJ model to access the web service, some context mapping from the model node to the corresponding component and view nodes, and views build with the WDJ design-time. The component and view controllers should only contain coding necessary to execute the model and provide the UI. In other words, any code which implements business logic or persistency has to be moved to the Business Layer.

If necessary, dynamic modification of the views is allowed but the recommendations for WDJ should be taken into consideration especially in respect to performance (see the references for more details). This restriction makes sure the user interface can be moved to other more model-driven technologies at a later point in time.

8.5.3 Backend and Business Layer connectivity

The rules of section General decisions apply.

8.5.4 Process layer integration

The process layer uses Guided Procedures (GP) as process framework. WDJ components can be included in a GP process by creating a callable object of type “Web Dynpro Component (GP Interface)”. This requires that the WDJ component implements a WDJ interface which is provided by GP.

This interface allows the WDJ component to interact with the GP process. The implementation of the interface is described in detail in the following “How-to”-document:

\\Dwdf051\isidm\PTU_IS_Composite_Dev_Program\35_Architecture\20_HowToDocuments_DevGuides\30_GuidedProcedures\HowTos\2005\HowTo - Implement_a_CO_Interface.pdf

8.5.5 Value Help

☑ **Use SVS, EVS and OVS for all value help scenarios**

- Simple Value Selector (SVS)

  It is based on a DropDownByKey UI element. The texts to be displayed in the dropdown list are retrieved from a simple data type's enumeration of key/display text pairs.

- Extended Value Selector (EVS)

  Like the SVS, this input help is used for selecting a key/display text pair of a simple data type. This input help displays a popup UI with a built-in function for browsing and filtering large value sets in a table.

- Object Value Selector (OVS)

  Unlike the SVS and EVS, the Object Value Selector is not entirely based on a declarative approach. To embed this sophisticated value help in the Web Dynpro application, some lines of code has to be implemented in the corresponding OVS controller. In re-

turn for this programming effort the Web Dynpro runtime automatically renders a generic OVS UI. This user interface is based on a special OVS core component belonging to the Web Dynpro Java runtime environment.

8.5.6 Customizing / Personalization

8.5.6.1 Implicit Personalization

WDJ offers the so-called “implicit personalization”. This allows changing the user interface on administrator or end-user level in a modification-free way. This feature is available only if WDJ UI is embedded in an iView. All personalization data is stored in the Portal Content Directory (PCD). That means in detail that the data is stored in the IView and in the end stored as IView properties. These properties are NOT visible in the standard iView editor but they are stored with the iView object.

Prerequisites

Personalization is automatically available for each Web Dynpro application **on the Enterprise Portal**. You can disable personalization by setting AllowUserPersonalization in the Web Dynpro runtime environment configuration screen to false. The default setting for this value is true.

Features

The scope of the personalization functions depends on whether personalization is performed by an administrator based on roles, or by an individual user for that user only. If the administrator makes a setting but does not define it as final, this setting can be overwritten by the user. Here is what each of them can do.

Administrators

To do personalization setting, administrator starts an IView in preview mode and opens the personalization dialog by choosing Ctrl while simultaneously pressing the secondary mouse button. The following options are available:

1. Remove a UI element from the user interface by setting visible to NONE. (UI elements which are defined as invisible by the Web Dynpro application cannot be made visible.)
2. Modify individual properties like read only, Tool tip text, Mandatory, text direction, width of input field, Label text, explanation text of a UI element. The properties that you can change depend on the UI element in question. However, you can always reset a property to original.
3. Rearrange columns in a table or lines in a form.
4. Insert additional UI elements using decorate. These UI elements are only for decorative or informative purposes and have no impact on the functionality of an application.
5. Marking settings as final. This prevents end users from making any further changes of their own.
6. Edit customer enhancement fields. This feature is available only for adaptive RFC and not for web services. After choosing Edit Customer Enhancement Fields in the personalization dialog window, you see the context nodes that contain customer enhancement fields. You can then select the appropriate UI elements for these fields and add them to the IView.

End Users

To personalize Web Dynpro applications on the Enterprise Portal, the user positions the mouse pointer on the relevant UI element and presses Ctrl + secondary mouse button.

The following options are available:

- Remove a UI element from the user interface by setting visible to NONE.
- Arrange the table columns according to specific requirements..
- Reset some or all personalization settings to original.

Based on the above explanation, it is evident that WDJ UI implementing GP interface and WD4VC application integrated into GP will not have the administrator personalization since these UIs will not be an iView which is a pre-requisite for doing the administrator personalization. Administrator personalization in GP is available only if the WDJ UI/WD4VC UI is integrated as an iView callable object in GP.

End user personalization is available in all cases but not much can be done in End user personalization. All the main features are available for an administrator personalization.

	Administrator Personalization	End User Personalization
GP interface callable object	Not possible	Possible
WD4VC callable object	Not possible	Possible
Portal iView as callable Object	Possible	Possible

8.5.6.2 Explicit Personalization

For explicit personalization for WDJ check the following link: Explicit Personalization

Explicit personalization is not available for VC.

8.5.7 Translation / Internationalization

WebDynpro Java is fully integrated into the standard SAP translation process. All frontend texts within a WDJ component are stored in language-dependent resource files, which are then automatically included in the translation process (via import into the translation system).

☑ **All texts visible on the user interface are translated using the NWDI translation process**

WDJ makes sure all texts you are entering on view design level, e.g. label texts, button texts, tooltip texts are included in these files automatically. If a text is displayed on the UI by any other means, e.g. displaying an error message in the message area, this text needs to be created in a message-pool.

☑ **Use message-pools in order to add user interface texts dynamically**

✘ **Don't use literals in your WDJ coding**

Further details on this can be found at the corresponding chapter of the online help in the developer studio.

8.5.8 Extensibility

In general, there is no extensibility concept for WDJ applications. Only the adaptive RFC, adaptive Webs Service and the UI decoration features add some kind of extensibility to this technology.

☑ **Use adaptive Web Service to call services in the Business Layer**

The UI decoration feature adds only the possibility to enhance the user interface by very simple UI elements which don't trigger server event.

8.5.9 References

These links refer to several documents describing WDJ in more detail:

- **Development Guidelines for Web Dynpro Java (NW 2004s)**
  https://bis.wdf.sap.corp/wd4java/java70/documents/WD_NW 04s_DevelopmentGuidelines.ppt
  Please follow the guidelines and tips contained in the presentation (the chapter regarding RFC usage is not relevant as the composite development only uses web services). The presentation also contains link to further documentation and examples.
- **Web Dynpro in the Spotlight**
  Introduction into Web Dynpro from a beginner's perspective
  https://sapneth3.wdf.sap.corp/webdynpro
- **Web Dynpro for Java Knowledge Base (NW 04s)**
  https://bis.wdf.sap.corp/twiki/bin/view/NW 04S/WebDynproForJava

8.6 WebDynpro and Visual Composer integration

See this tutorial:

8.7 Adobe Interactive Form

8.7.1 Overview

Interactive Forms by Adobe (AIF) are used for all form-driven process steps within a composite. They may be implemented as online or offline forms. Online forms are used whenever system access is essential, e.g. for value help or data validation. Offline forms are used whenever a system connection is not available or desired.

The use cases addressed via AIF are:

1. Process Initiation: Submission of an AIF triggers a composite application business process. The data from the form is populated into the process context.
2. Offline Step Interaction: In the course of a running business process, a participant receives via e-mail an offline AIF, where he can fill information or approve/reject only; submission (by URL or Mail-Reply) of the filled out form/data triggers the completion of the step and populates the process context with the form's data.
3. Generating a final AIF artifact: At the end of a composite application business process, an AIF is created as an artifact to be sent out as the end result of the composite application
4. On-Line Approve/Review of an AIF: As part of an online process step, a user can view a read-only version of an AIF as an integral part of the UI step (main panel) and perform an action based on his review (e.g. Accept, Reject, etc..).
5. On-Line Data Entry using an AIF: As part of an online process step, an AIF is displayed to the user as an integral part of the UI step (main panel). The user can fill/modify information in the form, and submit it.

8.7.2 General guidelines

- ☑ **Online forms are implemented by adding an AIF control within a Web Dynpro Java component**
  Data retrieval from and to the form is realized by interaction with the component context. The backend connectivity is implemented within the WDJ component and the rules for WDJ apply.
- ☑ **Offline forms are implemented by creating a callable step of type "Interactive Form" within Guided Procedures**
  All data are retrieved from the GP context and written back into the GP context.
- ☑ **An offline AIF may be submitted either via a HTTP submit or by sending a email to the GP server**

  Regarding email submission of an AIF to the GP server, you can find a HowTo at \\Dwdf051\isidm\PTU_IS_Composite_Dev_Program\35_Architecture\07_ArchitectureGuidelines_Organisation\2005\20_ArchitectureWorkingAndEvaluationGroups\30_UI_Layer\50_AdobeFormsOnline\HowTo use an IF CO in an offline scenario.doc
- ☑ **Minimize the amount of scripting within the form, especially don't call Web services**

8.7.3 Value Help

Value helps in Adobe Interactive Forms are similar to EVS controls of Web Dynpro.

8.7.3.1 Dynamic Value help for offline AIF

Populating the value help of a dropdown list directly within a CO of type Interactive Form works only partially (see internal CSS message 3972470 2006)

So use the following (see CSS 3689564 2006): A call a WS to fill a hidden table with the value help of a dropdown list on the Interactive Form. A JavaScript on the Interactive Form maps the value help from the hidden table to the dropdown list. The processor of message 3972470 2006 will explain how to bind the value help directly with the dropdown list.

8.7.4 Translation / Internationalization

AIF can be translated by a specialized translation tool (ORBITT), but integration with the standard translation process is missing.

8.7.5 Digital signatures

- ☑ **If a user-specific digital signature is needed, use an AIF with signature field**
- ✘ **"Qualified signatures" (i.e. legally binding in any case) are not possible at the time being**

As of today, Adobe Reader is the only "UI control" available for WebAS Java stack allowing for user-based digital signatures by enabling the user to choose a locally stored certificate from his local PC (note that for user-specific signatures, the resp. certificates have always to be stored locally (e.g. on a smartcard or in a safe folder of the local PC) because of security/confidentiality reasons).

However, since signing a PDF in Adobe Reader means simply signing the PDF binary, there might be some Javascript code contained in the PDF that modifies the displayed data dynamically without breaking the signature (since the actual binary would remain unchanged).

The workaround proposed by Adobe for this issue is that the author applies a signature to his original PDF, i.e. if a user trusts the respective author; he may fill out the form with relatively low danger.

However, e.g. with respect to the German law regarding digital signatures, this author signatures are not sufficient for cases where so called “qualified signatures” are needed. This has to be kept in mind when positioning our current composites at the market.

8.7.6 References

AIF – GP integration Requirements & Guidelines
\\Dwdf051\isidm\PTU_IS_Composite_Dev_Program\35_Architecture\07_ArchitectureGuidelines_Organisation\2005\20_ArchitectureWorkingAndEvaluationGroups\30_UI_Layer\50_AdobeFormsOnline\AIF - GP Integration Requirements & Guidelines.doc

9 Technology Selection for the Process Layer

☑ **In case there are multiple roles/different processors contributing to a business process of a composite, Guided Procedures (GP) is the only allowed option**

Note this also holds true if potential customer extensions to a business process cannot be excluded that might require adding further roles to the process

✖ **Do not "hardcode" process logic in another layer, e.g. do not combine UIs and their sequence that might be assigned to different roles/processors by the customer.**

Guided Procedures is a framework consisting of two main parts: the GP Design time which allows for defining user-centric processes (i.e. it's not suited for A2A scenarios) and the GP runtime that executes these process models.

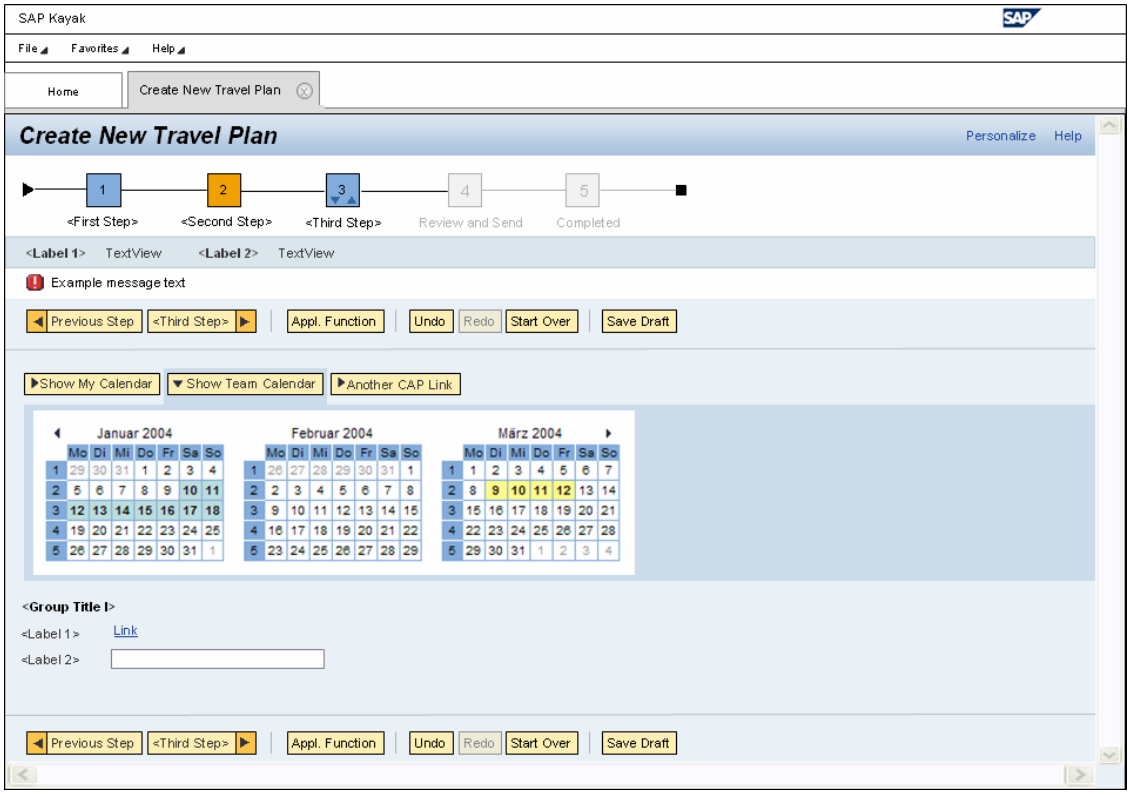
Other alternatives that have been ruled out:

- Internet Service Request (ISR): based on SAP WAS ABAP, whereas GP runs on SAP WAS Java, which makes it the preferable choice for our Java-based composites.
- ccBPM: not suited for user centric processes, focused on system integration. Moreover, it depends on XI, which we don't require to be present in a customer's landscape.

☑ **In case there is only one role/processor contributing to a (simple sequential) business process that is to be executed by this processor "in one go", use a Guided Activity pattern instead of GP**

☑ **If in doubt, talk to ICOD architecture**

A Guided Activity is a UI floor plan which basically consists of a Roadmap control indicating the current step of a sequential process and buttons for navigating back and forth between the steps or completing/canceling the whole sequence. Cf. the following screenshot to get a rough impression, but cf. ICOD UI guidelines for details:



Since there is no underlying workflow engine as it is the case with GP, it is not possible to hand the process over to another contributor. On the other hand, if it's clear that there will be only one contributing processor, you gain a smaller overhead (in terms of performance and development effort) compared to GP usage as well as the ability to navigate backwards through the steps.

Possible indicator for such "exactly-one-contributor" processes: the UIs of the individual steps could actually be combined into only one UI, to be processed by one user "in one go".

9.1 Callable Object / Action Design

☑ **By default, separate callable objects (COs)/actions for UI and update service should be defined. Only omit this separation in case of technical restrictions that make the needed context parameter consolidation impossible.**

Note that this holds true for any update service, i.e. also for persistency local to the composite, so calling a CAF Core service that updates a CAF Business Object (BO) should also be separated out.

☑ **Callable objects offering an UI shall ensure data consistency by calling check services**

✘ **Do only separate UI and update steps, do not do further separation like putting check services into separate COs/actions.**

✘ **Don't split up updates to CAF Core BOs: there should be only one CO/action calling one CAF Core service combining all necessary CAF BO updates**

Note that combining CAF BO updates into one CAF Core application service operation allows for rolling back the changes if one of these BO updates fails. Cf. chapter "Transaction handling" for more details on that, as well as chapter "Business Logic Layer" on Application Service cut recommendations.

☑ **After the update service call step, add a process control CO of type "Decision (comparison to pre-defined value)" or "Business Logic" to handle service call error messages**

Since in chapter "Business Logic Layer" the recommendation is given that all exceptions (from the backend as well as from the composite business logic itself) are to be caught and passed as GDT Log part of the (Web Service-exposed) CAF Application Service output structure, handling error messages only based on GDT Log should be sufficient here.

✘ **GP Exception handling is not recommended since the alternative described above offers essentially the same functionality.**

Cf.
http://aiokeh.wdf.sap.corp:50000/SAPIKS2/contentShow.sap?_SCLASS=IWB_STRUCT&_SLOIO=015F6F4169E25858E10000000A1550B0&TMP_IWB_TASK=PREVIEW2&RELEASE=712&LANGUAGE=EN&_SEQNUM=33&_LOIO=443D3936C5C14A8FE10000000A1553F6&_CLASS=IWB_EXTHLP

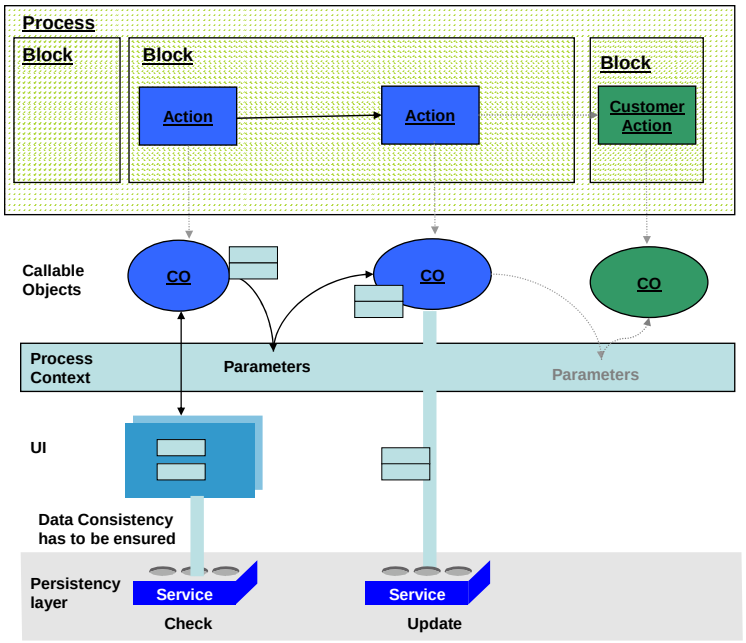
for details on the CO type "Business Logic" (it should also give you some hints on the available logical expressions).

☑ **When using the CO type "Web Service", only use Logical Destinations (instead of direct WSDL references).**

Cf.
http://aiokeh.wdf.sap.corp:50000/SAPIKS2/contentShow.sap?_CLASS=IWB_EXTHLP&TMP_IWB_TASK=PREVIEW2&_LOIO=F87889CE12924273A379C1C590675573&_SLOIO=015F6F4169E25858E10000000A1550B0&RELEASE=712&LANGUAGE=EN&_SEQNUM=24&_SCLASS=IWB_STRUCT

on how to create Logical Destinations in the SAP NW Administrator.

In the following, customer enhancement/extensibility possibilities by separating UI from backend calls are illustrated (note that this could also be useful to create different industry enhancements of one "core process")

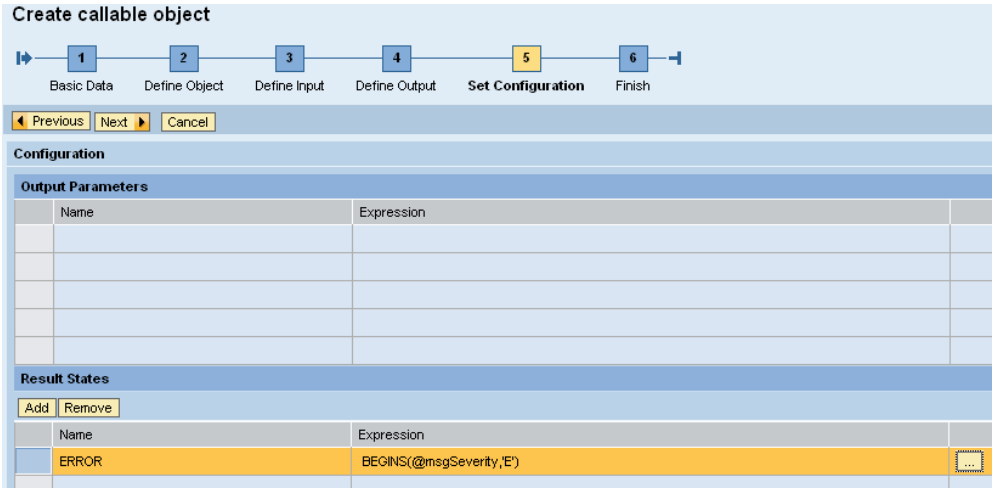


Process as delivered by SAP

Here, we have two callable objects/actions, one handling the UI and performing the appropriate check services, the other one handling the backend update service call. This separation offers high flexibility to the process flow definition; customers could e.g. add an approval step between data maintenance and the update service call.

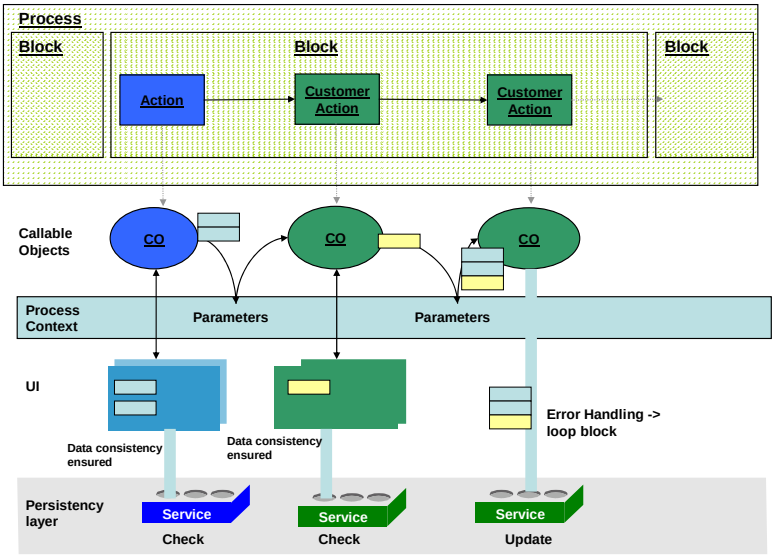
Calling appropriate check services before the update service call should minimize the possibility that the update will fail, but this cannot be excluded obviously (example: in highly parallel scenarios where many users are accessing one Business object at the same time, it might happen sometimes that you have retrieved a version ID by calling Read<BO> service (that we also count as “check service” in this context) in the UI action but this version is already out-of-date when the process continues with the Update<BO> service call in the update action).

These error cases are typically expressed by means of a dedicated service return parameter (in case of Enterprise Services, this parameter adheres to the GDT Log, in case of BAPIs, it has the type BAPIRET2 defined in ABAP DDIC). Based on this parameter, you can add an additional process control callable object of type “String comparison” or “Business Logic” to define a result state fired in case of the respective error condition. The screenshot below illustrates that for the CO type “Business Logic”.



Following figure shows how a customer enhancement of the process delivered by SAP could look like.

The original UI step collecting and checking the data is still present, but after that, the customer has added another UI step that puts additional data to the GP context in the end. After that, both the data from the original UI step as well as the data from the customer UI step is consolidated into the backend update step.



Delivered process enhanced by customer

9.2 Starting a GP process

☑ **GP processes can be triggered by the following ways (note the preference specified for each of them):**

- Via Web Service: For each (active) process template, there is a dedicated "start web service" generated and registered to the central GP wsil document /gpcore/GPProcessDiscoveryWSIL.

  This means: creating a web service destination in NW Administrator pointing to /gpcore/GPProcessDiscoveryWSIL will give you access to all GP process-start web services from all SAP tools that can handle web service destinations.

  This is the most preferable way to start GP processes, especially in case you need to supply parameters only known at runtime.

  Find more information under in the (preliminary) GP documentation under

  http://aiokeh.wdf.sap.corp:50000/SAPIKS2/sync.sap?_CLASS=IWB_EXTHLP&_LOIO=4458F9CC0D95311CE10000000A155369&TMP_IWB_TASK=PREVIEW2&RELEASE=712&LANGUAGE=EN

- Via API: GP framework offers a Java-based API for various tasks, one of them is starting a process - also parameterized if needed.

  Cf.
  http://aiokeh.wdf.sap.corp:50000/SAPIKS2/contentShow.sap?_SCLASS=IWB_STRUCT&_SLOIO=015F6F4169E25858E10000000A1550B0&TMP_IWB_TASK=PREVIEW2&RELEASE=712&LANGUAGE=EN&_SEQNUM=141&_LOIO=43FCDF77FC6510B3E10000000A11466F&_CLASS=IWB_EXTHLP

  for details.

  Only use the GP APIs in case the Web Service based process instantiation doesn't offer the needed functionality. However, please inform ICOD architecture in case you're facing such situations.

- Via URL: A process can be started via a so-called instantiation URL that you can retrieve for each (active) process template from the GP design time, looking like this:

  ../../sap.com/caf~eu~gp~ui~inst/AInstantiation?process.template.id=DA479C30036411DAA699000BCD3B046D

  Note this way of process instantiation makes only sense in case there are only parameters that can be set at design time (to append all input parameters of your process template with their respective default values to the URL, set the checkbox "Include default parameters" when retrieving the instantiation URL in GP design time).

  Only choose this instantiation type in case it's not feasible to use the process start web service or the GP APIs; moreover, please inform iCOD architecture in such cases.

- Via Interactive Form: A process can be started by submitting an Interactive Form, which in our case always means an Adobe PDF Form; supplying parameters to the process – or better mapping form fields to the input parameters of the process – is supported, too. Moreover, it's possible now to create the necessary form template directly from GP design time.

  Cf.

  http://aiokeh.wdf.sap.corp:50000/SAPIKS2/contentShow.sap?_SCLASS=IWB_STRUCT&_SLOIO=015F6F4169E25858E10000000A1550B0&TMP_IWB_TASK=PREVIEW2&RELEA

SE=712&LANGUAGE=EN&_SEQNUM=87&_LOIO=44BCA11FC60B7006E10000000A155369&_CLASS=IWB_EXTHLP

for details.

Before using this way of process instantiation, clarify

- Where/how the user can get/download the necessary form.
- If there has to be an approval step for the received data as a first step in the process.
- How long the submission is valid, i.e. if there should be a due date after which submitting the form shouldn't trigger the process.
- If repeated submission – and thus, triggering several process instances – of the same form is ok or not.
- If you need to make sure that only trusted addresses can trigger processes by submitting the form (especially if the form can be submitted from outside the own company).

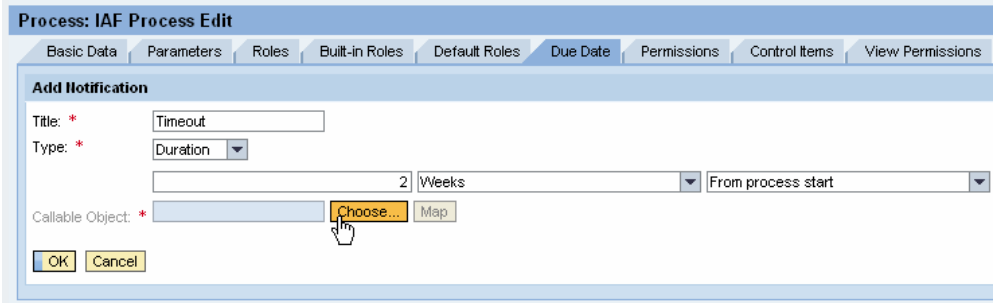
  It is recommended to use Bounced Mail Framework (BMF) for setting up lists of trusted addresses since there is a native integration between GP and BMF. Cf. http://aiokeh.wdf.sap.corp:50000/SAPIKS2/contentShow.sap?_SCLASS=IWB_STRUCT&_SLOIO=44D9586D3EF05F4DE10000000A11466F&TMP_IWB_TASK=PREVIEW2&RELEASE=712&LANGUAGE=EN&_SEQNUM=60&_LOIO=444F6E3C14A35DACE10000000A11466F&_CLASS=IWB_EXTHLP for configuration of BMF usage from GP and http://aiokeh.wdf.sap.corp:50000/SAPIKS2/contentShow.sap?_SCLASS=IWB_STRUCT&_SLOIO=44D9586D3EF05F4DE10000000A11466F&TMP_IWB_TASK=PREVIEW2&RELEASE=712&LANGUAGE=EN&_SEQNUM=636&_LOIO=44450862F1F014BCE10000000A155369&_CLASS=IWB_EXTHLP for BMF details and administration in general.

  Apart from that, using this way of process instantiation is fine.

- GP runtime: The GP runtime application has it's own native entry UI that also allows for instantiating processes from all available process templates. Because of usability reasons, this (too generic) UI should never be accessed by an end user however, so this is no option for integration into ICOD composites.

☑ **Add a relative due date/duration (e.g. "3 weeks from process start") to each process template, which terminates processes after a sensible timeout to prevent orphaned instances**

This is done by registering a CO of type "Terminate process" with a due date defined for the respective process template.



One example for the origin of orphaned processes: when a GP Process is integrated into the portal via an URL iView (pointing to the process instantiation URL), it is possible to instantiate it by simply navigating to this iView. However, users may be unaware of this more or less implicit instantiation, which leads to unexpected processes/entries in their or other users' workflow inbox

.

? **It is currently under investigation how to dynamically retrieve the GP process template be instantiated (so hard-coding the respective template ID can be omitted).**

9.3 GP Context

At runtime, passing of data between process steps is done via the so-called GP context, which is in a nutshell a data persistency with a structure that can be modeled and mapped in GP design time for COs, actions, blocks and process templates.

☑ **GP context stores all intermediate data by default.**

☑ **Only use CAF Core as intermediate storage in case of technical limitations preventing you from complying with the previous rule.**

During the lifetime of a GP process instance, store all new data (i.e. data maintained by the users, newly created Adobe forms, etc.) in the GP context, given that data is (or may be) relevant for other GP steps.

This has the major advantage that in case the running process instance is cancelled somehow, GP framework will perform all clean-up; moreover, it's easier for a customer to enhance the process template if all data is "at hand" in the context.

Do only use other stores in case the data that you put there is consistent from technical and business perspective (also see next point) or technical limitations (like restrictions in VC regarding parameter exposition → only flat table supported in CE 7.1 SP 0 - 3) prevent you from using the GP context.

☑ **Save only consistent data in Business Logic layer and/or the backend by default. Exceptions are only allowed in case GP context cannot be used as intermediate storage (cf. previous rule).**

As already mentioned, before writing intermediate data collected during a running GP process instance to another store than GP context, you should make sure

1) the data is consistent not only from a technical point of view, but also from a business perspective (e.g. don't persist some purchase order items without also persisting the corresponding purchase order header referencing these items [normally, this should be prevented by a well-designed Business Logic layer anyhow]). As mentioned, exceptions are only allowed in case of technical limitations with respect to GP context usage/parameter exposition to GP context.
2) there is some other way of accessing this intermediate data apart from your (running) GP process (e.g. some worklist UI displaying the intermediate data or business objects, allowing to take them up again).

These measures (especially the 2nd one) should prevent the need for a custom rollback mechanism that handles data clean-up in the case where a GP process instance is terminated abnormally.

✘ **Do not replicate data of a Backend or CAF Core Business Object into the GP context. Restrict yourself to keys**

By means of passing only key values through the GP process context instead of replicating the respective full (business object) data from Business logic layer or the backend, re-reading this data based on the passed keys in every process step where it's needed, you should be able to minimize out-of-sync situations on the one hand and keep the GP context lean on the other.

☑ **Use consistent names for the context parameters of the different process steps to be able make use structure mapping. If you face problems doing so, please contact iCOD architecture.**

GP is able to map structured context parameters automatically, given they are identical with respect to hierarchy and component names (component namespaces may be different).

This can be achieved by selecting several equally structured context parameters and clicking the "Structure mapping" button afterwards in the parameter consolidation UI of the GP design time.

9.4 GP runtime considerations

With running processes you should take care of the followings:

☑ **Provide descriptive and unique work item titles for the end user**

Note that by default a user will only see the design-time description of the current GP action as workflow item text in his inbox, which is undesirable from a usability point of view (since it means that items arising from parallel processes which have arrived at the same step can only be distinguished based on their creation dates).

Cf.
http://aiokeh.wdf.sap.corp:50000/SAPIKS2/contentShow.sap?_SCLASS=IWB_STRUCT&_SLOIO=015F6F4169E25858E10000000A1550B0&TMP_IWB_TASK=PREVIEW2&RELEASE=712&LANGUAGE=EN&_SEQNUM=47&_LOIO=43FB05B590C410B3E10000000A11466F&_CLASS=IWB_EXTHLP

for details on how to parameterize action descriptions (i.e. refer to input parameters), which is the most preferable way of providing dynamic work item titles.

This feature works using parameters with cardinality 1..1. Please note that if you don't have such parameter available defined by the CO wrapped in the given Action, you may also add this necessary parameter on the Action level.

☑ **Assign default UME roles to all processor roles of your process template to avoid the need for manual assignment at process instantiation**

☑ **In case you need to dynamically set the user(s) for a process step at runtime, use CO type "Assign users to Process Role"**

Note that the CO type "Assign users to Process Role" requires a list of unique UME IDs corresponding to the respective users that you typically have to retrieve by means of UME API (the CO types "Choose a user" and "Read User Information" offered by GP only allow for retrieving UME IDs for individual users, not for lists of users).

9.5 Usage of UWL

☑ **The UWL shall be used as workflow inbox for all processors of GP process actions.**

- ✘ **UWL is not part of the CE 7.1 SP 0 - 3 delivery, meaning usage of UWL as GP inbox only works in a Federated Portal Network (FPN) scenario with an 04s consumer portal hosting the UWL.**

Cf. chapter “Portal Access Layer” for more details on FPN**.**

- ☑ **The GP runtime and administration portal worksets (and thus, the GP inbox therein) should only be accessible by administrators.**

Note that the UWL has to be configured to retrieve workflow items generated by the GP framework.

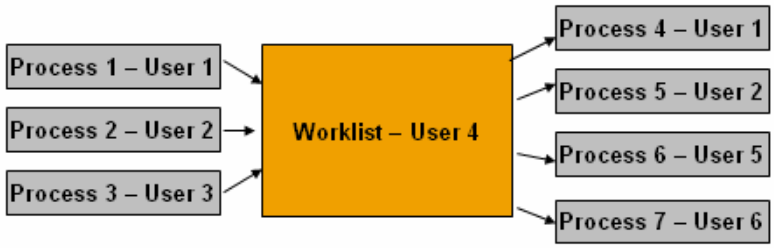
Cf.
http://aiokeh.wdf.sap.corp:50000/SAPIKS2/contentShow.sap?_SCLASS=IWB_STRUCT&_SLOIO=8DD084428254B767E10000000A155106&TMP_IWB_TASK=PREVIEW2&RELEASE=712&LANGUAGE=EN&_SEQNUM=6&_LOIO=43EF06A7860C7061E10000000A1553F6&_CLASS=IWB_EXTHLP

for details.

- ✘ **UWL cannot be used as mass processing tool, each work item has to be processed individually**

Note that GP does not offer the possibility for work item mass-processing in general, which means that in such cases, a custom solution is needed.

The following example illustrates that:



User 4 in the second step should be able to “batch assign” approvers to some business objects, each one provided via one individual process instance Process 1-Process 3.

As said, GP doesn’t have a functionality to work on several process instances/workitems in parallel, so it’s not sufficient to simply have some “Assign approver” action in the corresponding process template (which would give User 4 one separate workitem for each business object).

Hence, User 4 in the second step should have custom multi-selection enabled worklist UI – which is not part of the GP process– that displays the business objects provided by Process 1-Process 3 and allows for assigning approvers to them. After this has been done, follow-up GP processes are started for each business object, sending respective workitems to the defined approvers (User 1, User 2, User 5 and User 6.in the example).

It should also be possible to use the GP APIs to collect and complete running process instances for such a custom worklist.

Cf.
http://aiokeh.wdf.sap.corp:50000/SAPIKS2/contentShow.sap?_CLASS=IWB_EXTHLP&TMP_IWB_TASK=PREVIEW2&_LOIO=449A3276A5726573E10000000A11466F&_SLOIO=015F6F4169E25858E10000000A1550B0&RELEASE=712&LANGUAGE=EN&_SEQNUM=143&_SCLASS=IWB_STRUCT

on retrieving process instances and

http://aiokeh.wdf.sap.corp:50000/SAPIKS2/contentShow.sap?_SCLASS=IWB_STRUCT&_SLOIO=015F6F4169E25858E10000000A1550B0&TMP_IWB_TASK=PREVIEW2&RELEASE=712&LANGUAGE=EN&_SEQNUM=142&_LOIO=440D3E1626821C9DE10000000A11466F&_CLASS=IWB_EXTHLP

on how to complete actions programmatically. However, this has still to be evaluated.

- ✘ **UWL is "inbox only", no actions in UWL are possible effecting the GP process instance, e.g. approve, reject**

GP process steps cannot be completed within the UWL directly, e.g. no ad-hoc Approve/Reject function in the UWL is supported. It is also not supported to forward the GP workitem to another processor.

- ☑ **In case forwarding of GP workitems is desired, use the process control item "Delegate Role**"
- ☑ **If a given steps need to be taken over by some third party not defined in the process, this can only be done via API.**

The process control item "Delegate Role" allows the respective processor to assign his current workitem to other users from the GP runtime UI (where he automatically navigates to when clicking a GP workflow item in the UWL).

Cf.

http://aiokeh.wdf.sap.corp:50000/SAPIKS2/contentShow.sap?_SCLASS=IWB_STRUCT&_SLOIO=015F6F4169E25858E10000000A1550B0&TMP_IWB_TASK=PREVIEW2&RELEASE=712&LANGUAGE=EN&_SEQNUM=58&_LOIO=42CEC68872C56BAEE10000000A1553F6&_CLASS=IWB_EXTHLP

for details.

The API for takeover is available via the **IGPRuntimeManager** class, and its **changeTaskProcessor** method.

Cf.

http://aiokeh.wdf.sap.corp:50000/SAPIKS2/content_get.sap?TMP_IWB_TASK=PREVIEW2&_CLASS=IWB_EXTHLP&_LOIO=43FCDF77FC6510B3E10000000A11466F&LANGUAGE=EN&RELEASE=716

For more information on UWL in general, cf UWL homepage:

http://nspad261.pal.sap.corp:1080/uwl/index.html

9.6 Delivering GP Content

- ☑ **All GP content shall be delivered in active state**
- ☑ **GP content is delivered via NWDI infrastructure**

Cf.
https://bis.wdf.sap.corp/twiki/bin/view/Sapinternal/GuidedProceduresFAQ#How_is_Guided_Procedures_content_delivered_to_customers_

for details.

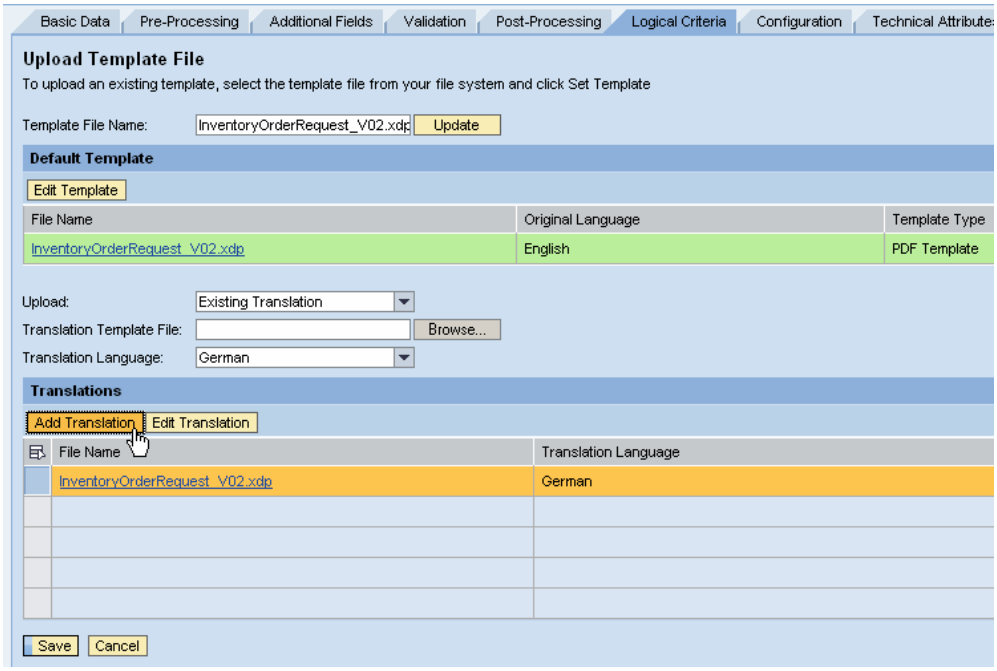
- ☑ **Only Design Time Objects are delivered**
- ☑ **Standard (.xlf-based) NWDI translation process is used**

This is automatically achieved when using NWDI delivery for all language-dependent texts apart from Interactive Forms.

- ✘ **As of now, Interactive Forms cannot be translated by the .xlf-based NWDI translation process**
- ☑ **We will deliver only an English version of each Interactive Form template. It is up to the customer to translate this template (or use an alternative one) as required.**

Interactive Form templates for different languages can be added to the "Interactive Form" object referenced by your Interactive Form CO under the tab "Logical Criteria" in GP Designtime. At runtime, the correct template according to the user's locale will be instantiated then.

As said, only add an English version here for delivery.



- ✘ **Don't use the GP proprietary translation tool mentioned in the GP documentation**

9.7 Naming Conventions and Folder Structure

- ☑ **Each Composite has a root folder with name of the composite**
- ☑ **All Folders for GP objects of your composite are created under this root folder**
- ☑ **Folder Names are: Blocks, Actions, Callable Objects, Content Objects**
- ☑ **Process Templates are created directly under the composite root folder.**

9.8 References

BIS page: https://bis.wdf.sap.corp/twiki/bin/view/Sapinternal/GuidedProcedures

RKT Materials

Guided Procedures Documentation

Twiki Page: Developers Guide, Installation Instructions, Javadocs for APIs

For API-Documentation see also http://www.sdn.sap.com/irj/sdn/javadocs.

CAF SDN forum.

GP also delivers some examples (Order Office Material Process, Time-Off Process). They are available in Design Time under the folder “Examples”.

10 Technology Selection for the Portal Access Layer

☑ **In the Portal Layer, the SAP Enterprise Portal is used as single choice of technology.**

In the Portal Layer, the Enterprise Portal (on NW04s and on NW CE) will be used as single choice of technology and the methodology of role-based User Interface Design will be applied.

For more information about portal content development, the iCOD program provides a portal development guide containing necessary information required for implementing role-based portal content:

https://portal.wdf.sap.corp/irj/go/km/docs/room_project/cm_stores/documents/workspaces/9163c421-9248-2810-cdbe-aa3356d48314/Architecture/Material/iCOD_PortalDevGuide_NWCE.doc

11 Technology Selection for Analytics

11.1 Consumption of BI Data in a Composite

11.1.1 BI Queries and Query Views

Analytic applications are created by including BI query elements as source of data (similar to an Enterpise service) in a Visual Composer model. The selection of the BI query or query view in the BW backend defines the data to be provided.

☑ **Analytics data is sourced by existing or new BI standard queries or query views, delivered with BI Content.**

All used BI objects must be available to the customer with an appropriate BI Content release. Feature packs of the current correction system (BI Content 7.0.3 at the moment) are normally used to deliver BI Content quickly.

BI Content development information and schedules:

https://bis.wdf.sap.corp/twiki/bin/view/Netweaverdev/CrossComponentBIBusinessContent

☑ **For performance reasons the BI queries and query views are tailored to the application, i. e. avoid to select unneeded data (in rows or columns).**

☑ **Use query views to adapt queries to the exact selection or to create variants of a query for similar selections. This keeps the number of new queries limited.**

Ideally existing queries are used. In case appropriate queries do not exist, new queries should select the data that is needed. Query views are a good way to limit existing or new queries to the needed selection.

☑ **Calculations and restrictions are normally better implemented in BW (e. g. calculated key figures, restricted key figures and especially time variables) than in Visual Composer**

☑ **Displayed characteristics are normally used in rows, pure input characteristics are used in free characteristics (input port)**

☑ **Refresh BI query element in Visual Composer after each change in BW to keep both parts aligned**

- ✘ **With the number of BI queries and query views used within one VC model the performance slows down, as each must be executed**
- ✘ **Use queries with structures or characteristics in columns only in exceptional cases, as their usage is inflexible**
- ✘ **BW Hierarchies can't be used in VC currently, a workaround could be to use grouping characteristics of different levels (e. g. 0BP_GRP_A05, 0BP_GRP_A06)**

SAP xApps Analytics Development Guidelines

More detailed guidelines by the BI Suite Optimization team can be found on this page:

https://bis.wdf.sap.corp/twiki/bin/view/Netweaverdev/SAPXAppAnalyticsGuidelines

11.1.2 Direct Access to BI InfoCubes

If the BI system is connected through XMLA connector, it is possible to directly access data of an InfoCube. The following link describes how to use InfoCubes instead of queries or query views:

https://www.sdn.sap.com/irj/sdn/wiki?path=/display/VC/Use+InfoCubes&

The InfoCubes can be inserted and used in VC models similar to queries or query views.

- ☑ **Consider this option for simple analytics sections of a composite, as this avoids developing and delivering BI content**
- ☑ **For complex and more flexible analytic functions the use of queries and query views is recommended**

11.1.3 Access BI Data and Queries from CAF vs. VC

The following document describes how to create a CAF application service to access a BI system:

How To….Create CAF Application Service Accessing BI InfoCubes

- ☑ **Use this option to combine BI data with transactional data in CAF**
- ☑ **To display pure BI data consume the data from VC**
- ✘ **Do not implement operations (e. g. aggregation) in CAF, but create an appropriate BI query for performance reasons**

11.2 Load CAF Data into a BI system

Data stored within the local persistency of CAF Core can be loaded into BI InfoProviders. The following tutorial explains how to setup an extractor:
https://websmp210.sap-ag.de/~sapidb/011000358700005903192005E.pdf

12 Technology Selection for User Management

The User Management Engine is the identity management application of any SAP system, which is based on the SAP NetWeaver Application Server Java, e.g. the SAP NetWeaver Portal or the Composite applications. This base service runs on top of the J2EE engine and can be set to store user related data in different types of repositories, like database, LDAP, SAP R/3.

- ☑ **The User Management Engine (UME) as User Administration Tool is to be used** UME has an administration console for managing user accounts.
- ☑ **Do not implement any custom solution for user management**

By means of configuration UME offers various options to integrate different data stores. Typical scenarios of identity management are:

- UME using central LDAP system as data store - SSL communication is recommended to encrypt the data transferred
- UME using the user management component of a backend system (with R/3 adapter) Roles existing in the ABAP systems are visible as Groups in the UME. If the backend has the User – Business Object assignment

The UME has an **application programming interface** (API) that can be used in programmatic environment – typically in the Business Logic Layer and Backend Connectivity Layer – enabling identity management, that means the management of UME Objects like Users, Groups, Roles, Security Policy, SAP Logon Tickets, User mapping.

- ✘ **In CAF Core do not implement any custom access control.**

The CAF Core provides a built in abstraction over the UME and handles authority checks and access control without manual coding. However, would it be necessary to use UME API for access control please contact the Architecture team for advise.

- ☑ **Use the User, Group and Role concepts to implement refined access to such parts of the SAP applications like Portal Content, Business Objects, Process steps**

For the Guided Procedures it is crucial to distinguish between roles that are the processors of the certain steps. You use UME Roles for this purpose.

For the CAF Core it is desirable to refine access to the certain set of operations of an Application Service with the help of the UME actions.

For the Portal Content of a Composite application it is important to build up a role based content structure and control access to its items by assigning users to these Portal Roles in the UME.

- ☑ **Instead of assigning Portal Roles to Users, assign them to UME Groups or UME Roles, because these two UME Objects can in all cases be used at the Permission setting of a Portal Content**
- ☑ **If there is a need, configure UME to replicate every persistence related action taking place (create, update, delete) to an external system, such as BI, SRM, CRM in the form of XML files.**

  Replication of User Data:

  http://help.sap.com/saphelp_nw04/helpdata/en/e0/3820a37edeec45a91dd6e45fba15b8/content.htm

  Defining To Which SAP Systems User Data is Replicated:

  http://help.sap.com/saphelp_nw04/helpdata/en/6a/ea79c8e9782d449549b425189ffb32/content.htm

More Information on the UME can be found here:

- Identity Management (Service Marketplace): http://service.sap.com/~form/sapnet?_SHORTKEY=01100035870000668401&

- UME (SAP Online Help): http://help.sap.com/saphelp_nw2004s/helpdata/en/a6/9d86f7b9d0224fbf74f9024b0b0a6a/frameset.htm
- Tutorial: "Protecting Access to the Web Dynpro Application Using UME Permissions": http://help.sap.com/saphelp_nw2004s/helpdata/en/f3/a64d401be96913e10000000a1550b0/frameset.htm

13 Configuration of Composites

Configuration refers to the possibility to adapt the behavior of a composite to the business needs of a customer without the need to change coding. In the mySAP Business Suite configuration is referred to as customizing, but for the composites the term configuration will be used. There's grey zone between configuration and extensibility, currently without a clear line of separation.

☑ **As little Configuration in Composite as possible**

☑ **Existing Configuration in Backend should be reused by service calls**

One of the major goals for composites is short-time-to-value, meaning that the time to get a composite running productively at a customer should be minimized. By introducing a customizing concept with the same complexity as in the backend, this goal will not be achievable. So also in the configuration of a composite, we have to achieve simplicity by hiding the complexity from the one, who is doing the configuration.

Nevertheless configuration can't be avoided completely, because otherwise we would have to hard-code some logic and the composite could not be adapted to the customers needs.

The need for configuration is dependent on the complexity of the composite, but even in the lightest composites some kind of configuration is needed, e.g. system configuration to define the backend to communicate with.

☑ **No single place for configuration.**
Each type of configuration has to be located, where it logically belongs to.

The type of configuration corresponds to the layers of the composite anatomy:

- Process Configuration defining the process flow and the processors of the process logic layer
- Personalization to define user-specific settings within the Personalization of the User Interface Layer
- System Configuration to define the backend connections of the backend connectivity layer
- Business Configuration to define business logic specific settings within Business Logic Layer

☑ **For all configuration needs, answer the following questionnaire:**

  - **Are we able to implement the 80% case and the rest can be done by the customer via extensibility?**
    → If so, don't provide configuration.
  - **Does the configuration influence the process flow of the composite?**
    → If so, it shall be put into the Process Configuration in GP
  - **Does the configuration define the communication with the Backends?**
    → If so, it shall be put into the System Configuration using the corresponding NW tools

- **Is the configuration user-specific or affects the screen layout only?**
  → If so, it shall be put into the Personalization using implicit (and explicit) personalization
- **→ Else, put it in the Business Configuration in CAF-Core**

✘ **There's currently no concept to ship backend customizing content with the Composite, for guidance see the information on backend abstraction layer.**

☑ **All Dependencies of the Composite on Backend have to be stated in the Documentation of the Composite**

Details about Configuration of Composites in iCOD can be found in this presentation:

https://portal.wdf.sap.corp/irj/go/km/docs/room_project/cm_stores/documents/workspaces/9163c421-9248-2810-cdbe-aa3356d48314/Architecture/Material/iCOD_ConfigurationCustomization.ppt

Details how to implement business configuration in CAF-Core can be found in this chapter.

Details about the implicit and explicit personalization features of WebDynpro Java can be found in this chapter.

Details how to handle default values for service calls can be found in this chapter.

14 Extensibility

Details can be found in this presentation:

https://portal.wdf.sap.corp/irj/go/km/docs/room_project/cm_stores/documents/workspaces/9163c421-9248-2810-cdbe-aa3356d48314/Architecture/Material/Extensibility_and_Supportability.ppt

15 Security/Collaboration

More details can be found in this presentation on Collaborative Scenarios:

https://portal.wdf.sap.corp/irj/go/km/docs/room_project/cm_stores/documents/workspaces/9163c421-9248-2810-cdbe-aa3356d48314/Architecture/Material/CollaborativeScenario_060529.ppt

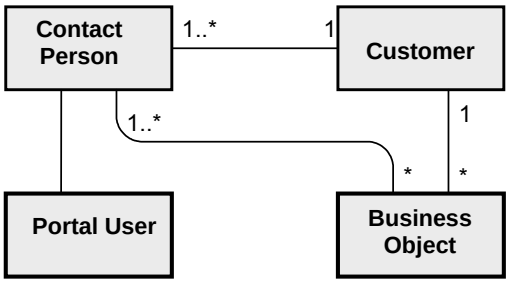
Several iCOD composites structure the collaboration between a company and its customers. In a collaborative scenario the employees of the customers of the company access the company's portal to carry out operations on Business Objects which they are allowed to manipulate themselves. E.g. they can place, alter or cancel own purchase orders, but they are not allowed to access orders of others.

- **Portal Users (Users of the Composite)** can be categorized into 2 groups:
  - **External User (Employee of the Customer, self-service user)** He is the employee of the customer, who logs in to the company's portal.
  - **Internal User (e.g. Sales Agent**) He is e.g. the sales agent of the vendor company which is operating the portal and responsible for one or several customers. He is usually an existing user in backend system.
- **Backend User:** He is a user in the backend system. It can be a real user or a technical user used by the composite for all Portal Users or "group account" used by all employees of the same customer. (see subchapter about user mapping.)

This topic is split into subchapters according to the following main questions:

- How to map Portal Users to Backend Users?

- How to implement (instance level) authority check for accessing Business Objects?
- How to assign Business Object instances to a specific user (including indirect assignment, see an example in the picture below)



- How to assign new users to the corresponding Business Objects?
- How can the proper security level of the External Facing Portal be ensured?

The following subchapters provide several solutions to these main questions and they also describe the dependencies between them represented by “if” clauses.

15.1 Mapping Portal user to Backend User

☑ **Each Portal User can be mapped to one Technical User in the Backend** by putting the technical user into the configuration of the service calls and selecting “Basic” authentication.

☑ **Portal User can connect using SSO to the Backend** if the Portal User is already available in the backend with the same UserID or its creation is allowed in the corresponding customer landscape. More information about how to configure the Portal Server and the underlying backend system for SSO can be found here: http://help.sap.com/saphelp_nw2004s/helpdata/en/89/6eb8e1af2f11d5993700508b6b8b11/frameset.htm

After both the portal server and the backend system is configured for SSO, the SSO has to be selected as authentication method for the respective service destinations.

☑ **It is recommended to use SSO if auditing is required,** because it makes visible which user did what in the backend.

☑ **Use SSO if functionality for authority check is available in the Backend** (see following two chapters)

If SSO is used, then there are various methods to eliminate the double user maintenance in the backend and in the portal by integrating their user management:

- Central User Data Store
  - *Central LDAP server*

    http://help.sap.com/saphelp_nw04s/helpdata/en/74/642441cd87a12be10000000a1550b0/frameset.htm
  - *Central User Administration (CUA)*

http://help.sap.com/saphelp_nw04s/helpdata/en/07/622441cd87a12be10000000a1550b0/frameset.htm

- Portal uses the Backend's user store
  - *UME with R/3 adapter*

    http://help.sap.com/saphelp_nw04s/helpdata/en/e4/ed484196158739e10000000a1550b0/frameset.htm

The selection of the right method is highly depending on the existing customer landscape, but it does not influence the design and the development of the composite.

In such scenarios, where the company does not want to create new account for each External User, both SSO and group mapping has to be used depending on the type of the user, e.g.

- SSO is used for internal users and all external users are assigned to the same technical user in the backend
- SSO is used for internal users and each employee of a specific customer is assigned to the backend user representing the customer in the backend

In NW CE the backend connectivity will be administered centrally in the NetWeaver Administrator tool, compared to NW 04s, where several alternatives were available depending on the corresponding layer. The unification has the following negative side effects:

- ✘ **User and group mapping, which was offered earlier by Portal System Objects, is not available in NW CE for WS calls from VC anymore.**

  Portal System Object can be used for mapping Portal Users or Groups to individual Backend Users in NW CE **only when accessing BI from VC**.

  For user mapping only the R/3 reference system concept is offered by the framework. More information can be found here:

  http://help.sap.com/saphelp_nw2004s/helpdata/en/ed/845896b89711d5993900508b6b8b11/frameset.htm

  The basic idea is that upon logon to the portal, an SSO ticket with two user ids is created, namely the one used for portal logon itself and the respective one mapped for the reference system.

  Disadvantages:

  - all connected backend systems have to accept either the portal logon user id or the user id of the reference system (since these are -as mentioned- the only two available in the SSO ticket).
  - mapping of groups or roles is not feasible (i.e. you can only have mappings of individual portal user ids to backend user ids, but a "group account" is not allowed).

  ? It has to be clarified if this concept is working in CE and in the FPN scenario

- ✘ **Mapping of Portal Groups to individual Backend Users is not supported by the framework in NW CE.**
- ✘ **Using SSO for Internal Users and mapping a technical user to all External Users in the same enterprise Service call is not supported.** Either SSO or a single technical user has to be set for all users.

? Group and user mapping methods as well as these in combination with SSO for Internal Users, can be implemented as custom user mapping logic by adding custom code into

the generated External Services code of the Backend Abstraction Layer, which does the mapping and calls the enterprise Service with the appropriate userid. It has to be evaluated if the custom userid mapping can be implemented in the CAF Core based BAL. iCOD Architecture has to be contacted before choosing this solution. CE 7.1.1 may offer a built-in solution.

15.2 Assigning Users to Business Objects

User and Business Object assignment can be persisted in the backend as well as in the composite:

- ☑ **It is recommended to maintain the Portal User to Business Object assignment in UME** if the assignment model of the business scenario can be reflected in UME Objects (Users, Groups and Roles), e.g. each "Customer" is represented by a "UME Group". The benefit of this solution is that this way the CAF Authorization framework can be used for authority check (see next chapter).

If each UME User is assigned only to one UME Group, the respective Business Objects (e.g. "Customers") can be propagated as Groups automatically into UME by implementing the TradingPartnerDirectory Interface.

More information about Trading Partner Directory Interface can be found here:

http://help.sap.com/saphelp_nw2004s/helpdata/en/d6/249f40c18a2a54e10000000a1550b0/frameset.htm

- ☑ **It is recommended to (re-)use the User to Business Object assignment if it is available in the backend by exposing and consuming the assignments via enterprise Services.**
- ☑ **Implement CAF local Business Object based assignment** if the backend does not offer this already and the assignment model of UME Objects does not fit the business scenario.

Do not replicate or store the User to Business Object assignment in(to) the UME if it is already in the backend. In this case, instead of using CAF authorization framework, filtering has to be implemented also for local Business Objects similarly to the Business Objects in the backend. If the usage of CAF authorization framework can lead to higher benefit than the disadvantages of data replication in the specific scenario, then contact the iCOD architecture team.

15.3 Checking Authority of Portal Users to Business Objects

Collaborative scenario requires authority check on Business Object instance level. E.g. the Portal User can access only his related Purchase Orders.

- ☑ **Authority checks on Local CAF Business Objects instances can be implemented with CAF Authorization Framework.** UME Users, Groups and Roles can be assigned to Business Objects instances and CAF will do the authority check. This option is only feasible if you maintain the Portal User to Business Object assignment in UME. (see previous chapter). It has to be evaluated how it can work exactly. Potentially the usage of Business Rules is required. You can find more information here:

  http://help.sap.com/saphelp_nw2004s/helpdata/en/44/356564baf848a0e10000000a114a6b/frameset.htm

- ✘ **In most of the cases Enterprise Services do not offer an authority check on Business Object Instance Level**
  (There are some examples for exceptions, like CRM.)
- ✘ **Do not implement own authorization framework in the backend, because development in backend system is generally out of scope of iCOD.**
- ☑ **Authority check can be implemented for backend Business Objects by filtering within the Composite.**
- ☑ **It is recommended to use the existing selection criteria of the enterprise Service**, because of performance reasons**.** If the corresponding service does not have the required selection criteria, then the filtering has to be implemented in the composite.

Filtering logic has to use the User – Business Object assignment, which is described in the previous chapter.

- ✘ **Do not provide direct access for the Portal Users to the Backend System if the Portal and Backend user is mapped by SSO and there is no instance level authority check in the backend.** The user might logon directly to the Backend, this way he avoids the filtering and gain access to data of other users. A firewall can prevent External Users to access the Backend directly. Internal Users may be allowed to see the data of other customers depending on the specific scenario.
- ☑ **Assign only the necessary Roles (PFCG) to the backend users.** However please note that it does not ensure instance level authority check. The Security Guide should inform the customer about this fact.

15.4 Lifecycle of (external) Portal Users

The following activities have to be managed manually or automatically when creating a new Portal User:

- ☑ **Creation of Backend User with the same userid is required when creating a new Portal User** if SSO is used, Backend User did not exist yet and Portal User and Backend User storage is not unified in the landscape.
- ☑ **New Backend user has to receive the corresponding roles (PFCG) in SU01**
- ☑ **New user has to be assigned to the corresponding Business Object(s).** If Portal User and Business Object assignment is described in UME by User and Group assignment, and each Portal User is assigned only to one Group, then the “Delegated User Administration” and “Self-registration” concepts can be used to manage this activity without additional custom development.

  More information about “Delegated User Administration” concept:

  http://help.sap.com/saphelp_nw2004s/helpdata/en/57/6aa430a3a8c1498d4f3eb08c41be95/frameset.htm

  More information about “Self-registration” concept:

  http://help.sap.com/saphelp_nw2004s/helpdata/en/c6/37eaeccab21849847802ec3586676a/frameset.htm

15.5 Secure Network Architecture for External Facing Portals

There are general guidelines available about how a secure network architecture has to be set up for running an External Facing Portals:

https://www.sdn.sap.com/irj/servlet/prt/portal/prtroot/docs/library/uuid/7c3ba590-0201-0010-f4b1-953e107b9b57 (slide 14)

https://www.sdn.sap.com/irj/servlet/prt/portal/prtroot/docs/library/uuid/9d5d9398-0701-0010-2d87-fd75c112d68d (page 24-25)

? Compared to generic portal scenarios, in the case of External Facing Portal the security can be increased in a way that the Internal and External Portal Users access different portals. One is in the DMZ and another one is behind all firewalls. The GP process, which they interact in, is integrated in a way that both portals connect to the same work-flow engine. Federated Portal Network might be used to connect the two portals as well and increase security. These approaches have to be evaluated.

15.6 Miscellaneous

The External Facing Portal concept has not been evaluated yet. The evaluation will be done in the scope of eTax project. Beyond the statements of the previous subchapters, the evaluation will investigate

- What is the proper UI technology for an External Facing Portal? Where are the limitations of VC/WD concerning performance and branding (pixel perfect design)?
- Is it possible to integrate a composite into a non-SAP Portal?

16 Reuse Decisions and Constraints

Reuse is using an item more than once. Within CE 7.1 SP 0 - 3 based iCOD projects reuse is desirable to increase development efficiency. iCOD Architecture differentiates the following reuse categories.

1. Reuse by Concept
2. Reuse by Copy and Paste
3. Reuse by Reference
4. Reuse provided by NetWeaver

16.1 Reuse by Concept

Example:

The Backend Abstraction Layer is a concept for solving a specific problem. Having this problem in your project you should reuse the described concept. If you have another problem where a solution is not described in this document

- Check SDN for web logs and tutorials
- Get in contact with other iCOD or NetWeaver developers to find a solution concept

- Try to find best practices in the Mini Reuse Wiki

  http://pwdf0626.wdf.sap.corp:1100/MiniReuse

If you have found a best practice by your own write an article in the Mini Reuse Wiki by your own to share it.

16.2 Reuse by Copy and Paste

Example:

In the following example you will find a piece of code that you can copy and paste in your own program if you have the following problem: Displaying the current Date Using Data Binding

http://help.sap.com/saphelp_nw04/helpdata/en/60/57fb027b6a8144b5c01e593709e7d0/content.htm

If you have another problem

- Check SDN for web logs and tutorials
- Ask other iCOD or NetWeaver developers to find a piece of code that solves your problem. Try to find Java code snippets in the Mini Reuse Wiki

  http://pwdf0626.wdf.sap.corp:1100/MiniReuse

☑ **If you have own coding that could help others to increase their development efficiency write an article in the Mini Reuse Wiki to share it.**

16.3 Reuse by Reference

Example: Number Sequencer, Alpha Conversion etc.

iCOD plans to set up a track for reusable software components to establish a reuse or foundation layer that can be referenced by many iCOD projects. In addition a process has to be defined how to handle reusable software components. Which reusable software components can be found in the iCOD foundation layer and how they are referenced and integrated is documented in the Mini Reuse Wiki

http://pwdf0626.wdf.sap.corp:1100/MiniReuse/index.php/Main_Page

☑ **If you are a developer in the iCOD foundation layer write an article in the Mini Reuse Wiki for each reusable software component that you are providing to the iCOD projects.**

16.4 Reuse provided by NetWeaver

Example:

Reuse provided by NetWeaver can be reuse by concept, reuse by copy and paste and reuse by reference. The most valuable reuse provided by NetWeaver is the reuse by reference.

☑ **In any case where you identify a reuse candidate put a short description of it in the central iCOD xls list. Get in contact with iCOD Architecture to discuss if the reusable software component can be provided by NetWeaver in the next release.**

Comment [SAP1]: This central excel is currently under discussion. Use this place for the time being – in the future, the Mini Reuse Wiki may be used.

17 Conceptual and Design Modeling

In addition to the executable models used by the different technologies like CAF or Guided Procedures during the course of a development project modeling on conceptual and design level is needed to express the intended structure and behavior of a composite.

In order to standardize these models Technical Architecture Modeling (TAM) was defined. TAM defines a common language and a graphical notation for communication on model level. The main target is to replace the ubiquitous free-style notations by a UML standard-based set of diagram.

Details on TAM can be found here:

- TAM-Homepage:
  http://ency.wdf.sap.corp:1080/wiki/TAM
- TAM-Flyer:
  http://ency.wdf.sap.corp:1080/wiki/upload/d/d9/TAM-Flyer.pdf
- TAM standard document:
  http://ency.wdf.sap.corp:1080/wiki/upload/3/3d/ModelingStandard_V8.pdf
- TAM Visio Shapes:
  http://ency.wdf.sap.corp:1080/wiki/Visio
  (The Visio Standard edition is sufficient for all of the shapes. Please order your personal copy in B2B, look for Office Software.)

☑ **Use Technical Architecture Modeling (TAM) for all iCOD Composites to model on conceptual and design level**

The usage of following diagrams types is mandatory:

☑ **Use Component / Block Diagrams to describe the high level static architecture of your composite including the communication with the involved backend systems**

Here's an example of a Component / Block Diagram from the iCOD Composite eTAX: eTax Systemarchitecture

☑ **Use Class Diagrams to describe**

- **the Business Objects with their attributes**
- **the Application Services with their operations and**
- **their relationships**

of your composite

Here's an example of a Class Diagram from the iCOD Composite xILR: xILR Class Diagram

☑ **Use Activity Diagrams or BPMN Diagrams to describe the process flow of your composite**

You can also use BPMN notation, if the expressiveness of BPMN is sufficient. Even if BPMN is not part of TAM, as it is the de-facto standard for business process modeling and will be the notation of Galaxy Designtime, it is released for iCOD.

Details on BPMN can be found here:

- BPMN Introduction:
  \\dwdf034\BPEM\BPEM_Public\BPMN\Introduction%20to%20BPMN.pdf

- BPMN Visio Shapes:
  \\dwdf034\BPEM\BPEM_Public\BPMN\GBPM Stencil (Full Set).vss
- BPMN Homepage:
  http://www.bpmn.org
- BPMN Spec:
  \\dwdf034\BPEM\BPEM_Public\BPMN\OMG%20Final%20Adopted%20BPMN%201-0%20Spec%2006-02-01.pdf

Here's an example of a BPMN Diagram for the iCOD Composite xPICM: xPICM BPMN diagram

The usage of following diagrams types is optional:

☑ **Use Package Diagrams to describe the SC and DC structure of your composite**

☑ **Use Activity Diagrams to describe the inner logic of your application services, if this logic is non-trivial**

☑ **Use State Machine Diagrams to describe the behavior of your Business Objects including its states and transitions, if this behavior is non-trivial**

The usage of following diagrams types is not recommended:

✘ **Usage of Sequence Diagrams is not recommended as their value-add compared to the effort of creation is rather low**

✘ **Usage of Use Case Diagrams is not recommend as their value-add is rather low**

18 Code Conventions

Even so iCOD is striving for model-driven development, there is and will always be some Java code to be written by developers. To ensure quality, readability, maintainability and reduce the potential of errors code conventions are an essential ingredient.

Whereas there are already SAP wide Programming Guidelines for ABAP, unfortunately there are no SAP wide Programming Guidelines for Java. And we also don't have yet specialized ones for iCOD.

But there are general ones provided by SUN, which shall be used also in iCOD:

http://java.sun.com/docs/codeconv/

☑ **Apply the SUN Java Code Conventions for all Java code written in iCOD**

Besides that there is a rather good paper by Scott Ambler on writing robust Java code:

http://www.ambysoft.com/downloads/javaCodingStandards.pdf

And if you really want a good and comprehensive overview about how to write high quality code, read the book Code Complete by Steve McConnell:

Code Complete, Second Edition

Even if this is written by a Microsoft guy, it's applicable for all programming languages and should be the bible for all developers writing product quality code.

? **iCOD Architecture will work on provisioning of iCOD specific Code Conventions**

19 Software Layers and Components

19.1 Software Layers

For a general description see Chapter Basic Architecture.

19.2 List of Components

19.2.1 Own Application and Software Components

☑ **For iCOD each logical layer will be packed into a separate Software Component (SC). There will be separate SCs for each layer**

☑ **All SCs have the name pattern "sap.com_ISC<Project><Layer>".**

Normally you create up to 6 SCs:

- **Portal Layer SC (pattern "ISC*EP") containing**
  - Portal Content (iViews, pages, worksets, roles)
  - UWL configurations
  - KM content (not in Portal light based on NW CE) → not recommended
- **Process Layer SC (pattern "ISC*PL") containing**
  - GP objects
- **UI Layer SC (pattern "ISC*UI") containing**
  - Web Dynpro Foundation UI code (including implementations of ICallableObject)
  - VC models
  - Adobe Form
- **Business Logic Layer including the Backend Abstraction Layer (pattern "ISC*BL") containing**
  - CAF
  - EJBs
- **Backend Connectivity Layer (pattern "ISC*BC")**
- **Unit test SC (pattern "test.sap.com_ISC*TST") containing all unit tests.** (This SC will not be delivered.)

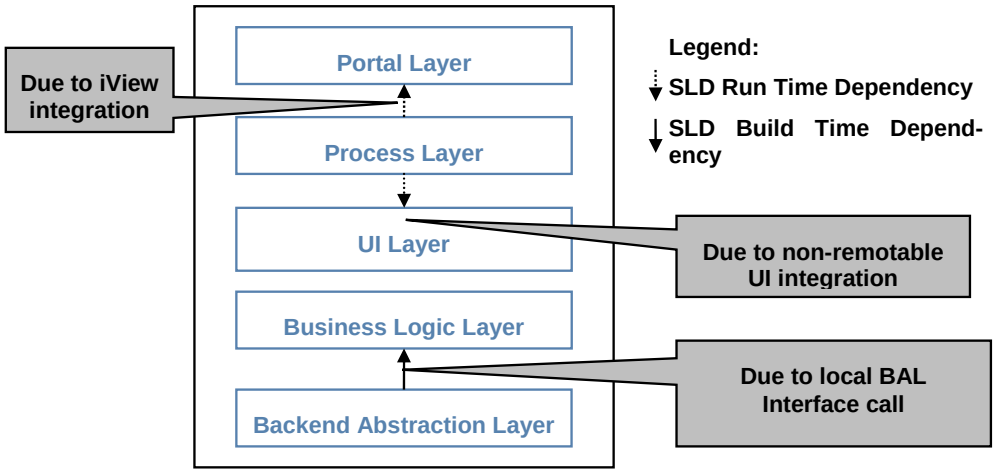


Figure 2: SC layers and dependencies

Figure 2 contains the maximal allowed SLD Build and Run Time Dependencies. Run Time Dependencies have to be maintained, if needed. Further needed Dependencies have to be discussed with iCOD Architecture.

The motivation for splitting the SCs can be found in the following two points:

- Separate patching of each layer is possible
- Distribution over several machines would be possible, see in the following section 19.3

Please note that for the FPN (Federated Portal Network) some special problems may come up with the UWL and the OBN (Object based navigation). These problems must be investigated. For a detailed overview of the software components, refer to the iCOD System Landscape Guide (refer to the link below).

☑ **All delivered SCs will contain the source code and models**

Only by this, we give the customers the chance to adopt it to specific requirements.

If you request some new SCs, please contact Martin Czekalla or Steffen Weidenbach.

19.3 Deployment Scenarios

- The default scenario is the deployment of all SCs on one server.
- Additionally a separated deployment to two servers is possible:
  - Server 1 contains the Business Logic Layer (BL) and the Backend Abstraction Layer (BC)
  - Server 2 contains the Portal layer, the Process Layer and the UI Layer

For a detailed overview of the deployment scenarios, refer to the iCOD System Landscape Guide (refer to the link below).

20 Relations to Standards and Guidelines

20.1 Product Standards

SAP Product standards define a common understanding on central non-functional characteristics and overall quality criteria requested by the market that every SAP product has to fulfil. They are designed to ensure smooth operations and use of SAP software for all parties involved, both on customer and SAP side, from administrator to end user. Requirements to SAP software such as optimizing TCO or legal compliance are addressed via such standards.

Some of these product standards have direct impact on the architecture of a product and require specific knowledge of each and every developer. These standards are described in more detail here.

Details about all the Product Standards can be found in the Corporate Portal in the Workspace "Product & Technology Group" >> "Standards & Tools".

Each project needs to state a planned compliance to each product standard before entering the development phase (usually in the Q-Gate P2D), act accordingly in the development and test phase, and report on the reached compliance at the end of the development and test phase (Q-Gate D2P, Readiness Meeting for Piloting, resp.). Please refer to the Q-Gate P2D presentations (program part) to get details on the planned compliance. The Q-Gate presentations are

stored at \\Dwdf051\isidm\PTU_IS_Composite_Dev_Program\20_Program\10_Meetings\.

For each product standard a contact within iCOD has been nominated and responsibilities for each project and the product standard contacts have been described. Please refer to the presentations in the iCOD cRoom's private section Program > Organization, or access this link for the Product Standard Contacts and this link for the responsibilities. All product standard contacts in iCOD can be reached by the distribution list DL PTU IS iCOD Product Standard Contacts.

Storage of information for Product Standards:

- Q-Gates' presentation (program presentation)
- public section of iCOD's cRoom > Information > iCOD_ProductStandards.

Tests for Product Standards are a main activity to verify the reached compliance. The tests are in the responsibility of the projects (see presentation), the iCOD Product Standard contact can support the tests.

Product Standard	Impact on Architecture
Accessibility	See Chapter Accessibility
Application Integration & Interfaces	See Chapter Backend Connectivity
Customizing & Configuration	See Chapter Configuration of Composites
Data Archiving	See Chapter Data Archiving
Development Environments	See Chapter Technology Decisions and Constraints
Documentation	See Chapter Documentation
Functional Correctness	See Chapter Functional Correctness
Globalization	See Chapter Globalization
Multiple Clients	-
Open source/Third Party	See Chapter Open source/Third Party
Performance	See Chapter Performance
Security	See Chapter User Management
IT Service/Appl. Management	See Chapter IT Service and Application Management (ITSAM)
Technical Implementation and Change Management (TICM)	See Chapter Technical Implementation and Change Management (TICM)
Usability	See Chapter UI Design

20.1.1 User Interfaces Design

What Does the Standard Define?

In order to keep the User Interface of the Composite Applications consistent and provide a high quality UI, the SAP User Experience group provides some general rules and guidelines. The UI Checklist for iCOD developers is a short and straightforward document to do a self-check of the User Interface at any time. The UI Checklist covers labels, alignment, abbreviations, buttons, tables, messages and more. The UI Checklist does not cover the overall UI Design paradigm and UI Design concepts. Information on these topics can be found in the complete Standards and Guidelines 1.1 http://nvpal067.pal.sap.corp:8080/sng/sg11/default.htm. Following the UI Checklist will improve the quality of the UI and we can reduce the effort for UI reviews. This document cannot replace the work of an User Interface Designer, therefore continue to set up separate UI review meetings with your responsible User Interface Designer.

Guidelines for Fulfilling the Standard within iCOD

☑ **Follow the UI Checklist for iCOD Developers:**

https://portal.wdf.sap.corp/irj/go/km/navigation/room_project/cm_stores/documents/workspaces/a11af347-0843-2810-f68a-ae8287a6dbab/User%20Interface%20Design/UI%20Checklist?StartUri=/room_project/cm_stores/documents/workspaces/a11af347-0843-2810-f68a-ae8287a6dbab

For more information about User-Centered Design see http://mysap.wdf.sap.corp:2080/UCD/.

20.1.2 Accessibility

(Details)

What Does the Standard Define?

Accessibility refers to the possibility for everyone, including and especially people with disabilities, to access and use technology and information products.

What does it mean for a screen in an SAP product to be accessible?

- The user must be able to get to everything using only the keyboard (keyboard access and navigation).
- The user must be able to see and/or hear what everything is (identification).
- If an action is available, the user must be able to perform that action (keyboard access).

Guidelines for Fulfilling the Standard within iCOD

Compliance with this standard is dependent on the UI technology used. Based on the decisions described in the chapter User Interface Layer

☑ **The following guidelines have to be known and followed by every developer:**

- Accessibility - Standard Overview Presentation for iCOD
- Accessibility for Web Dynpro Appl. Developer (L225)
- Accessibility for PDF Interactive Forms Technology (L231)
- Accessibility Rules for Designing Applications with Visual Composer

20.1.3 Data Archiving

(Details)

What Does the Standard Define?

The standard defines a uniform and complete data archiving solution for all SAP application components. To comply with the standard, every application component must provide a data

archiving solution or a delete program (if applicable) for the expected fast-growing data volumes.

Guidelines for Fulfilling the Standard within iCOD

Within the standard there are reasons listed to do archiving.

☑ **Each Composite, which persist Business Objects has to provide either a delete program or an archiving solution for those Business Objects**

It's a business decision, which of the alternatives is appropriate for a specific Business Object.

Within Java the technology for archiving solutions is XML DAS:

- Introduction to XML-Based Archiving
- Developing XML-based Archiving Solutions for Java

There's not yet a generic integration between CAF Core and XML-DAS.

? **iCOD Architecture evaluated, how CAF Core Entity Service instances can be archived with XML-DAS. The result is that this concept is feasible; however no iCOD project has implemented it yet and XML-DAS is not available in CE 7.1 SP 0 - 3.**

20.1.4 Documentation

What Does the Standard Define?

- To enable SAP customers to install, implement and use SAP software, SAP must provide complete, up-to-date and accurate product documentation; also of a legal point of view.

Guidelines for Fulfilling the Standard within iCOD

☑ Raw documentation needs to be provided by the development team. The template and best practices for the raw documentation is the latest version of composite application documentation at \\Dwdf051\isidm\PTU_IS_Composite_Dev_Program\40_KnowledgeManagement\10_Projects.
The Standards & Guidelines for writing need to be followed as much as possible, see \\Dwdf051\isidm\PTU_IS_Composite_Dev_Program\40_KnowledgeManagement\00_S&Gs

☑ The questions of the information developer for creating the final documentation need to be answered by the development team.

☑ The information developer creates the final composite application documentation and does language editing for non-native speakers.

20.1.5 Functional Correctness

What Does the Standard Define?

- perform static checks and remove all identified severe errors
- perform functional tests and remove all identified severe errors
- document and publish whenever limitations exist due to unresolved problems

- ensure by design and testing (at least adequate samples) that, after a component has been upgraded or a patch has been implemented, all old functions needed by other components/scenarios still work as expected.
- test cases with complete coverage of the requirement specifications as part of complete and traceable development documentation serve as an essential basis

Guidelines for Fulfilling the Standard within iCOD

- ☑ outline test cases in the SRS (Software Requirement Specification) document and provide data for traceability to software requirements and use cases in SRS
- ☑ perform developer tests and unit tests as described in iCOD's Test Approach (iCOD's cRoom private section QualityManagement > Test > General, Concepts) and in Section **Error! Reference source not found. Error! Reference source not found.**.
- ☑ provide test cases for integration testing (also called "functional tests"); due to the SCRUM methodology they have to be provided with each sprint
- ☑ perform integration/functional tests as described in iCOD's Test Approach, document results in GTP and CSS; support of TCC colleagues is possible
- ☑ provide test cases for an Acceptance Test
- ☑ perform an Acceptance Test (with project external business experts, e.g., customers), document results in GTP and CSS
- ☑ have JLIN run and remove the priority 1 and priority 2 errors - as much as possible <still under investigation>
- ☑ list limitations in the Q-Gate D2P, resp., "Readiness Meeting for Piloting"
- ☑ perform and support tests during validation as required
- ☑ define a smoke test for validation (purpose: quick verification of the successful installation of the composite)
- ☑ define "top X" processes (most important processes - for a later regression test), if not all test cases should be taken
- ☑ for a release higher than the first one: cover all "top X" functions by a regression test

20.1.6 Open source/Third Party

(Details Open Source) + (Details 3rd party)

What Does the Open source Standard Define?

The standard defines how to deal with open source and Third Party Software, whereby open source software is defined as software

- that is delivered with SAP Software Products / Client Solutions AND is "free" of licensing costs
- AND has a license that SAP must accept

OR has no license restrictions at all (but there are still risks related to it)

What Does the 3rd party Standard Define?

For SAP, as a solution provider for numerous companies that operates in various industries, licensing of third-party products enables reduced time-to-market, increased market acceptance of the solution (more industry coverage, or leveraging partner's market position, for example), and lower development and maintenance costs.

It is clear that the quality of licensed products when embedding or reselling, of tools for software engineering, and of services like the outsourcing of product maintenance, determines the quality of the whole SAP solution and all services SAP provides for the customer.
With SAP's international scope, contracts with third-party vendors must meet minimum standards with regard to international functionality, legal issues (liability, warranty), platform, delivery (revenue recognition), and the business model.

Guidelines for Fulfilling the Standard within iCOD

- ✘ **Don't deliver any Open Source Software, Freeware & Third Party Software within the Composites developed by iCOD**

20.1.7 IT Service and Application Management (ITSAM)

What Does the Standard Define?

A standardized management concept to operate the business processes and the underlying system landscape with optimal availability, optimal performance and lowest cost of ownership

What Are the Basic Principles of the Standard?

1. Solution Monitoring
   - Monitoring and alerting for critical issues are possible with CCMS
   - Heartbeat (availability) functionality with GRMG
   - Provide monitoring templates
   - Automatic monitoring setup during installation
   - Registration at SLD (System Landscape Directory)
2. Problem & Performance analysis
   - Central log administration and viewing (separate logs and traces are required)
   - Instrumentation for performance measurement
   - Activity and resource monitoring for own managed caches or queues
   - Information for integration into Solution Manager Diagnostics
3. Management of mySAP Technology
   - Component configuration
   - Capability of online backup & restore for all business application data
   - Managing of outdated data
   - High availability concept
   - Support for Adaptive Computing
4. Business Process Management
   - Guaranteed data consistency and data processing efficiency (batch processing, allow parallel processing)
   - Concepts for monitoring and error handling and for restart and recovery in case of asynchronous interfaces

5. Software Change Management
   - Transport (distribute) configuration settings within the system landscape
   - Managing customer modifications, enhancements
6. Remote Support
   - All support tools must be accessible via standard SAP remote infrastructure
7. License Auditing
   - Measurement of the license compliance is required

Guidelines for Fulfilling the Standard within iCOD

☑ **The following guideline has to be known and followed by every developer:**

- Guide for Component Checklist (L216)

The Project Issue Management composite application was tested by the production testing service unit regarding the ITSAM product standard.

Details regarding the scope of the testing can be found in this presentation:

Copy (version January 15, 2007) in the iCOD Collaboration Room:
https://portal.wdf.sap.corp/irj/go/km/docs/room_project/cm_stores/documents/workspaces/9163c421-9248-2810-cdbe-aa3356d48314/Architecture/Material/ITSAM_TestScope_iCOD10_PIM.ppt

Details regarding the result of the testing can be found in this presentation:

Copy (version January 15, 2007) in the iCOD Collaboration Room:
https://portal.wdf.sap.corp/irj/go/km/docs/room_project/cm_stores/documents/workspaces/9163c421-9248-2810-cdbe-aa3356d48314/Architecture/Material/ITSAM_TestServiceReport_iCOD10_PIM.ppt

20.1.8 Technical Implementation and Change Management (TICM)

(Details)

What Does the Standard Define?

The main goal of TICM is to ensure smooth and reliable software logistics processes within customer solution landscapes.

- Technical implementation means implementation of the technical infrastructure (installation of software components).
- Change management deals with the upgrade (release update) of components and system landscapes, the import of corrections, such as patches and support packages, and the transport system

Guidelines for Fulfilling the Standard within iCOD

☑ **The following guideline has to be known and followed by every developer:**

- TICM Development Guideline for Java Development

20.1.9 Supportability

What Does the Standard Define?

Supportability in general means to build applications providing mechanisms for support engineers and developers for analyzing the application in case of error conditions or performance bottlenecks.

More information about supportability can be found here:

http://bis.wdf.sap.corp:1080/twiki/bin/view/NW2004S/SupportabilitY

Guidelines for Fulfilling the Standard within iCOD

- ☑ Supportability is covered in the product standard ITSAM (IT Service and Application Management).

Also see this presentation on extensibility and Supportability

https://portal.wdf.sap.corp/irj/go/km/docs/room_project/cm_stores/documents/workspaces/9163c421-9248-2810-cdbe-aa3356d48314/Architecture/Material/Extensibility_and_Supportability.ppt

20.1.10 Performance

(Details)

What Does the Standard Define?

The pragmatic product standard Performance defines criteria that ensure good performance and scalability of applications. This refers to both new solutions and existing ones.

The Key Performance Indicators (KPI) which are relevant for iCOD are as follows:

- ☑ **All database query execution should take less than 10 ms.** In case a deviation is measured, then it shall be checked if all frequently executed accesses to the persistence layer are supported by an *appropriate index, and* all accesses to database have *complete WHERE clauses.*
- ✘ **No identical accesses to the persistence layer.** There are no identical accesses to the persistence layer within a business transaction.
- ☑ **Parallel processing enabled.** Does the system scale as expected, e.g. does the design of object/database locks and load balancing mechanisms allows scalable parallel processing?
  - ☑ Try to avoid static code blocks or static variables and synchronized code blocks or minimize the operations executed in a synchronized block.
  - ☑ Use the built-in optimistic locking of CAF Core, minimize implementation of pessimistic locking.
  - ☑ Deployable proxies are not thread safe. Use different proxy instances in each thread.
- ☑ **Two round trips per user interaction step.** There are at most two round trips between front-end and application layer per user interaction step. Between two servers there are two round trips in average per user interaction step.
  - ☑ **Do not call enterprise Services from a loop,** which makes the number of server-to-server roundtrips depending on the data volume.
  - ✘ **Use a single "bulk" read** to retrieve all necessary Business Object instances in a single roundtrip.
- ☑ **Average dialog response time.** The average response time per user interaction step should be below two seconds. Breakdown response time to identify the respective layer, if 2 sec end-to-end response time exceeded. (SQLTrace, JARM, Profiler)

- ☑ **Sizing procedure available.** There is a sizing procedure for disk space, CPU consumption, and memory consumption available.
- ☑ **Linear dependency.** The processing time and the memory required depends linearly on the amount of processed data.

In order to avoid high response time and CPU consumption when using Online Interactive Form by Adobe:

- ✘ **Do not use more than ~50 entries in Drop Down List.** Drop Down List uses scripting, which is executed both during the client and the server side rendering. Execution time depends non-linearly on the number of entries.
- ☑ **If you use more than ~50 entries, then use the Enumerated Drop Down List from the Zero Client Installation (ZCI) palette.** A ZCI based form contain less generated script, thus it improves the performance.
- ☑ **If you use more than ~50 entries, then set the global parameter of the WebDynpro Framework "sap.valuesetlimit.maxondemand" to ~50.** This way if the Enumerated Drop Down List contains more than 50 entries, than the entries will not be propagated into the Drop Down List during the rendering, but they will be loaded from the server and displayed in an EVS (Extended Value Selector) first when the user click on the value help arrow.
- ☑ **Do not use any ActiveX controls, use ZCI instead**. ZCI reduces the rendering time both at client and server side. It does not require installation of xACF, therefore it can work also with browsers which do not support ActiveX technology.

Please note that ZCI based Drow Down List is not stable at the moment, see Internal Message 231152 for current status.

The usage of some specific technologies can be heavily resource consuming. Therefore it is recommended the do performance measurements for the dialog steps where the following technologies are used:

- **CAF Guided Procedures:** This is a very database intensive technology; however performance improvement was introduced with SP09 and SP10.
- **Knowledge Management:** Storing of large files is database intensive.
- **Backend BAPI calls**: Response time can be high from several reasons, e.g. lack of caching.
- **Interactive Form by Adobe:** Size of the same form increases by factor 2-3 between two dialog steps. (See Internal Message 2282109 for current status.)

For a detailed overview concerning fulfilling the standard within iCOD and measurement tools, refer to the following documents:

- **Performance - Standard Guideline for iCOD**

20.1.11 Globalization

(Details)

What Does the Standard Define?

- **Goal:** Address the global market
- **Globalization** = Internationalization + Localization

The concept of globalization includes the process of developing software that addresses all business issues associated with making a company worldwide in scope. Globalization enables a company to run its systems simultaneously, anywhere in the world. And it involves developing multilingual applications and software products that companies can implement – alone or in combination – to reflect any business strategy and line of business. Globalization is the result of an effective internationalization and localization strategy. The design of a system needs to be culturally and technically neutral – internationalized

Internationalization topics:

- Unicode enabling
- Data Communication & Encoding
- Text Processing
- Cultural Awareness
- Display
- Input
- Printing
- Time zones
- Translatability
- Localizability

Localization topics:

- Functional localization: Some business processes differ from country to country. This may mean that SAP has to localize the functions in the application components to match.
- Translation*:* SAP's products have to be delivered in a language spoken by the users.

How to comply with the Globalization Product Standard?

Comply with the Standard requires from the developers:

- ☑ **Utilize the Net Weaver capabilities**
- ☑ **Create workaround if the technology does not give solution (if applies)**
- ☑ **Implement function localization (if applies)**
- ✘ **Do not reinvent the wheel**
- ☑ **Check the Guideline first, if a solution is needed**

Dealing with these factors can increase the complexity of the product, and increase the development time as well. The Globalization Guideline gives information about how to unitize the NetWeaver built-in capabilities and how to create workarounds. The functional localization is related to the project's requirements, and that document is not indented to give advice on it.

The NetWeaver technologies - Globalization Overview (Evaluation is based on technologies used by iCOD: NetWeaver 04s SP8, RFC/BAPI backend connectivity)

Technology	Standard subtype	Comments
Web Dynpro Java	Internationalization	Good - Data can render easily according to user's locale.
	Translation	Good – SAP standard translation process works well on WDJ projects.
Adobe Interactive Forms	Internationalization	Limited - Localizability sometimes requires workaround.
	Translation	(?) – Forms never translated at iCOD. (Translation of forms are unclear at the mome
Composite Application Framework Core	Internationalization	Limited – Date formatting problems with RFC/BAPI. Never tested with ESA. The „co data type" rule is required with Communication with other layers.
	Translation	Not applicable – CAF Core has no built-in translation feature. However, resource bundles can be added to CAF – works only if the CAF Core is at determine the caller's locale.
Visual Composer with Flex Compiler (WD4VC is not used currently)	Internationalization	Very limited - date conversion error, currency/number formatting problems
	Translation	(?) - VC models can be translated like portal content, but VC was never translated a iCOD.
Guided Procedures	Internationalization	Good – („correct data type" rule is required between COs)
	Translation	Limited - If you modify a translated process -> the whole translation will be lost. Developers easily forget to add the *develcontac*t to XLFs.

Guidelines for Fulfilling the Standard within iCOD:

Globalization Roll-out presentation:

https://portal.wdf.sap.corp/irj/go/km/docs/room_project/cm_stores/documents/workspaces/9163c421-9248-2810-cdbe-aa3356d48314/Architecture/Material/Globalization_rollout_1207.ppt

Globalization Guideline (GlobalizationGuideline_iCOD.doc):

https://portal.wdf.sap.corp/irj/go/km/docs/guid/20fc4ab4-4e67-2910-3ea4-941994dce564

20.1.12 Security

(Details)

What Does the Standard Define?

The security standard is a product standard which defines a uniform minimum level of security for all SAP applications. In general, much of the security-related aspects of a usual web application are covered by NetWeaver. Special care need to be taken on the input validation and authorization check of the Application services, during designing processes and when determining the initial configuration. So:

Guidelines for Fulfilling the Standard within iCOD

- ☑ **Permission checks for Application Services should be enabled**

  (for more details see chapter Application services)

- ☑ **Documentation should contain Security Guide as a subchapter**
- ☑ **Should carefully develop input validation** – as Application Service operations of a Composite Application might be accessed by not only their respective UIs, these operations should not assume that their input parameters have passed validation on the UI, and therefore should carefully validate any and all input data.
- ☑ **If the application cannot avoid explicitly storing personal data, data protection and privacy regulations should be taken into account** – personal data should not be exposed to unauthorized parties during storage transmission or normal operations of the application. Also such data needs to be purged upon request from the concerned party.

The main topics of security are:

Identity Management

On a basic level, this requirement is about distinguishing between users with different roles from the aspect of the process. This requirement also favors the use of UME and Single-Sign-On technologies.

- ☑ **For authentication and authorization purposes make use of UME features and artifacts exclusively.**

Network and Infrastructure Security

For all iCOD Composites it is advised to have secure communication turned on by default. The tools used by iCOD provide enough support for securing communication channels.

- ✘ **Do not implement any custom encryption code.**

Secure Programming

The technologies recommended above for developing iCOD Composites provide protection against Cross Site Scripting and SQL injection attacks if used properly. In the normal case the developer does not have to take care about secure programming. One example: avoid custom coding which accesses SQL statements directly, use CAF artifacts to access Entities.

In general, all input should be validated on both the UI and the Business Logic Layer.

Special knowledge is required, as there are other vulnerabilities which might be exploited with malformed input like XPath injection, etc.

More information:
https://portal.wdf.sap.corp/irj/servlet/prt/portal/prtroot/com.sap.km.cm.docs/corporate_portal/WS%20PTG/Controlled%20Documents/PIL%20Guidelines/L192%20Secure%20Programming.PDF

In addition, make use of the built-in virus scanning facilities if the application has to deal with uploaded files directly.

Audit and Accountability

Logging [see also chapter Application Services] should be implemented carefully in order to comply the possible requirements of Security Auditing.

More information:
https://portal.wdf.sap.corp/irj/servlet/prt/portal/prtroot/com.sap.km.cm.docs/corporate_portal/WS%20PTG/Controlled%20Documents/PIL%20Guidelines/L194%20Security%20Logs.PDF

https://portal.wdf.sap.corp/irj/servlet/prt/portal/prtroot/com.sap.km.cm.docs/corporate_portal/WS%20PTG/Controlled%20Documents/PIL%20Guidelines/L188%20Legal%20Auditing.PDF

Summary:

- ☑ **Read-only access shall be available for all data.**

✘ **Avoid the use of technical users. Use SAP Logon Tickets or certificate-based authentication instead if possible.**

Secure Configuration and Delivery

The Security Guide default delivered configuration should contain only the minimal amount of rights applied to roles in the system, by which the application is still functional. Also the Security Guide should prescribe only the necessary minimum authorizations.

Data Protection & Privacy

In general Composites should avoid storing sensitive/confidential personal information locally.

More information:
https://portal.wdf.sap.corp/irj/servlet/prt/portal/prtroot/com.sap.km.cm.docs/corporate_portal/WS%20PTG/Controlled%20Documents/PIL%20Guidelines/L184%20Data%20Protection%20and%20Privacy.PDF

Summary:

☑ **Configuration of permission checks for CAF Business Objects should be enforced by delivering the application with CAF BO permission checks turned on. The Security Guide should contain the initial configuration of the CAF BO access control.**

✘ **Avoid handling of personal data. Manage its confidentiality and lifespan if you can't.**

Secure processes

Design processes supported by an application in a way to disable scenarios, where a user might perform actions to which he/she is not entitled, including maliciously providing authorization to himself/herself/someone else.

21 Development landscape

The development scope of the iCOD program will be primarily development on the Java Stack. Basically, the development landscape design is based on the following basic assumptions:

☑ **Short Release Cycle:**
Within iCOD, we'll assume the production of one composite requires ~300 person days to develop, with an overall development and ready to ship phase of 3-6 months.

☑ **Delivery via NWDI:**
All development objects for an iCOD composite can be kept and delivered via NetWeaver Development Infrastructure (NWDI)

☑ **Translation**:
For all development objects a translation process is set in place.

☑ **Development Language is English, all objects shall be created in English**

? **Reuse Objects**:
Reuse of software or development components was out-of scope for iCOD projects in NW04s and is still under discussion for NW CE.

☑ **Strong coupling of composite applications to NW features:**
It is expected that the iCOD composite applications will heavily depend on latest features provided by NetWeaver. This requires a strategy for rapid upgrading of the development infrastructure.

☑ **Support Strategy:**
It is a declared goal to support at least a couple of SP releases in synchronization with NetWeaver CE cycles. This also implies that there will be as many SP and VAL tracks as supported SPs.

☑ **Shipment of Single Composites is possible:**
It is the aim of the program, that single Composites will be shipped to the customer.

For a detailed overview of the development landscape, refer to this document provided by the iCOD program:
https://portal.wdf.sap.corp/irj/go/km/docs/room_project/cm_stores/documents/workspaces/9163c421-9248-2810-cdbe-aa3356d48314/Architecture/Material/iCOD_DevLandscapeGuideCE.doc

22 Testing

Test activities of iCOD for manual and unit tests were outlined in iCOD's Test Approach document in iCOD's cRoom private section QualityManagement > Test > General, Concepts.

22.1 Manual Testing

Details on the manual testing can be found in Section **Error! Reference source not found. Error! Reference source not found.**.

22.2 Automated Unit Testing

In order to reach the goal of supporting a multitude of iCOD Composite Applications automation of testing is key for ensuring a good product quality.

☑ **Use JUnit for Unit Tests**
✖ **Do not deliver any code related to JUnit (like test classes) within the product.**
☑ **For checking integrity of running applications at the customer's use available supportability Features like Logging & Tracing, Heartbeat Monitoring, etc. instead**
? For the future it should be detailed out, which main use cases for unit tests in the composite development exist.

JUnit and other test related source code must be placed in the test SC (pattern **test.sap.com_ISC*TST)** and stored in the DTR, but it is not part of the customer delivery.

For iCOD, JUnit for Unit Tests shall be used. JUnit is approved by the SAP Software Regulation Process for internal usage Process and is widely accepted in the Java development community. Apart from that, JUnit is also integrated into the SAP Netweaver Developer Studio.

More Information about JUnit can be found here:

- http://www.junit.org
- Various weblogs on SDN on JUnit, for example
  - https://www.sdn.sap.com/irj/sdn/weblogs?blog=/pub/wlg/1355
  - https://www.sdn.sap.com/irj/servlet/prt/portal/prtroot/docs/library/uuid/9ef1967a-0901-0010-6c86-ed2251898764

Another alternative tool for unit testing is the ATS (Automated Test System). This is a JUnit based in-house tool which suites better for server based unit testing.

The tool has the capability to run the tests inside the JVM of the J2EE engine and supports/ provide different types of tests (like basic, client, server and application test).

ATS is integrated with the Netweaver Developer Studio, provides sophisticated reporting and has web front-end for mass test execution.

The CAF team planned to integrate CAF and ATS (automatic generation of unit test for CAF), but the feature was postponed to CE 7.1.1 according to Malte Christian Kaufmann.

The ATS tool should be evaluated inside the iCOD program.

More Information about JUnit can be found here:

https://bis.wdf.sap.corp/twiki/bin/view/Techdev/ATS

22.3 Automated Functional Testing

Mercury Quick Test Professional is under evaluation by the Test Competence Center (TCC) colleagues. There is an issue with recording Adobe Interactive Forms. Discussion was initiated with Mercury to solve the issue.