Guidelines for Specifying Composite Applications

Version 1.0

September 2006
### Table of Contents:

1 Introduction ........................................................................................................................................... 5

1.1 Scope of the Document ................................................................................................................... 5

2 Composite Development in a Nutshell ............................................................................................... 6

2.1 Motivation for Composite Applications ........................................................................................... 6

2.2 Basic Assumptions About Composite Applications .......................................................................... 7

3 Introducing the example scenario: Customer Service Care ............................................................... 8

4 Guideline for Writing a Composite Specification ................................................................................. 13

4.1 General Application Information ..................................................................................................... 15

4.2 Processes Overview ......................................................................................................................... 18

4.3 General Process Information ........................................................................................................ 20

4.4 Roles .................................................................................................................................................. 23

4.5 Visualization of Process Flow .......................................................................................................... 25

4.6 Step Data ........................................................................................................................................... 28

4.7 Exception Handling ......................................................................................................................... 31

4.8 Business Objects Overview ........................................................................................................... 34

4.9 Business Object ................................................................................................................................ 36

4.10 UI Mockup ........................................................................................................................................ 42

4.11 UI Description .................................................................................................................................. 45

4.12 Services ........................................................................................................................................... 48

5 Handover of Specification to Architect ............................................................................................... 51

6 Appendix ............................................................................................................................................... 52

6.1 Short Reference Guide ..................................................................................................................... 52

6.1.1 Form: General Application Information ....................................................................................... 52

6.1.2 Form: Process Overview ............................................................................................................... 53

6.1.3 Form: General Process Information .............................................................................................. 54

6.1.4 Form: Roles .................................................................................................................................... 55

6.1.5 Form: Visualization of Process Flow ............................................................................................... 56

6.1.6 Form: Step Data ............................................................................................................................ 57

6.1.7 Form: Exception Handling ........................................................................................................... 58
### 6.1.8 Form: Business Objects Overview ................................................................. 59
### 6.1.9 Forms: Business Object .............................................................................. 60
### 6.1.10 Form: UI Mockup ...................................................................................... 62
### 6.1.11 Form: UI Description ................................................................................ 63
### 6.1.12 Form: Services .......................................................................................... 64

### 6.2 Interactive Forms .......................................................................................... 65
Acknowledgement
The following colleagues contributed in various ways to make this document possible: Werner Aigner, Helen Day, Bernhard Drittler, Henrik Fessler, Ruth Groene, Markus Küfer, Markus Meisl, Gunther Piller, Uta Prigge, Janusz Smilek, Sebastian Speck, Volker Stiehl, Willy Strothmann, Atul Sudhalkar, Raphael Vogel, Patrick Zimmer
1 Introduction

1.1 Scope of the Document

Composite applications are defined as packaged 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 Composite Application Framework (SAP CAF) provides a methodology and toolset for developing and managing efficiently, composite applications using SAP's Enterprise Service Architecture. With this new breed of application, product managers responsible for the design of new business applications and business experts responsible for optimizing business processes within their respective companies are faced with the challenge of specifying composite applications in a way that can be clearly understood by solution architects who, in turn, are in charge of finding the right technology to implement the composite. This guide aims to explain in detail how to approach composites from a product manager (PM) / business expert (BE) perspective and how to provide all composite relevant information to the solution architect. This is done by guiding the PM / BE through a series of forms that he or she has to fill out in order to describe the composite in as much detail as possible.

For this guide we assume that the PM is an ISV (Independent Software Vendor) representative, responsible for the specification of new software products. The BE is a member of a company's business expert team. He or she is in charge of optimizing the internal business processes and therefore has to specify appropriate software solutions to accomplish this goal.

The centerpiece of this handbook will be a discussion of a methodology for specifying composite applications that can be built with SAP's composite toolset. This guideline will help to correctly describe new composite applications by following a checklist for the above mentioned forms. By using an example it will be demonstrated how to apply the checklist and what a final specification might look like. The examples presented in this guide will be written from a PM's perspective but they can easily be adapted to the BE’s needs.

It is not the intention of this guide to go into technical details or describe functions and features of particular tools. It is solely focused on the business concepts and processes and how to translate the business idea into implementation relevant key data for solution architects. Discussions about technology and tools can be found in the accompanying Architecture Guide. Although technical details are intentionally omitted, the first version of this guide is intended to help specifying composites suitable for being built with the SAP tools Guided Procedures for process modelling, Visual Composer for UI modelling, and the Composite Application Framework for developing the business objects and business services.

The guide is not designed to be read from beginning to end. Instead it consists of an introduction chapter (Composite Development in a Nutshell), which explains the basic concepts (composite applications in general) and basic terminology, but then it is more or less a reference book, strongly modularized, so that the PM / BE can jump directly to the chapters he or she currently needs for his/her decision/action.

The subsequent chapters provide the following: Chapter 2 gives a short introduction to composite applications. It discusses the motivation behind and the basic assumptions about composite applications. As this guideline is intended to be a practical handbook, the example which complements the theoretical part of the book is introduced in chapter 3. Chapter 4 is the heart of the guide: it describes how to fill out the forms that will make up the composite’s specification. Each form will be explained in great detail followed by an example based on the business scenario introduced in chapter 3. The appendix consists of the forms with a reference to the associated chapters and an additional chapter about interactive forms based on Adobe technology. Once the PM / BE is familiar with the main idea he or she should be able to fill out the forms simply by following the instructions for each form.
2 Composite Development in a Nutshell

2.1 Motivation for Composite Applications

Classical business applications are based mainly on the concept of manually entering data and reporting on that data. This concept led to the surge of two- and three-tier client-server ERP architecture, with SAP R/3 being one of the most prominent contributors. The high market motivation for this model was to enable business process reengineering and to empower employees. The focus was on data collection, control, reporting, and business process optimization based on analysis of the data reporting.

These kinds of business applications could provide only limited help to the relatively unskilled user and knowledge workers both in and outside the company. Their needs could only be met by tailored, hand-coded applications or, to a certain extent, by workflows based on a sequential programming model. Such workflow integration, however, did not give users an overview of the process but only showed them the individual steps to be executed and did not help process owners to take responsibility or handle exceptions effectively.

To summarize: classical business applications collect and analyze business process data effectively and can also run fully automated business processes, but they tend to fall short of fully involving the user in user-centric collaborative and iterative processes. Even today, many business processes are recorded in ERP systems but are actually executed using traditional paper-based or Microsoft Office documents in a fairly unstructured and uncoordinated way. Classical ERP systems do not encourage the establishment of novel or flexible business processes which customers need to gain competitive advantage. Last but not least, classical applications are self-contained and assume that all data and all business logic is owned by the application.

From a customer’s and end user’s perspective there are requirements for applications which the classical business application cannot meet. Such requirements include:

- providing a single, intelligent user experience across system and enterprise boundaries
- crossing functional, system and even company boundaries
- viewing the enterprise or supply chain as a whole rather than in its separate parts
- forcing enterprise-wide and process-wide system and human collaboration
- assembling a solution primarily from existing multi-source content
- being able to be adapted by business analysts

These requirements can be divided into two parts. Firstly, the user experience; the application has to operate in such a way as to reflect the user’s work context. This is not a new requirement and has already been addressed in different projects and by different technologies the role-based user interface approach and the Enterprise Portal, for example. But this is not sufficient. There is an additional requirement to provide applications that are fed by different systems and to unify them in such a way that the end user detects no boundaries or discrepancies between them. Combining end user experience and cross-system application integration needs leads compellingly to the topic of composite applications.

The end user’s requirements however, are not the only reason for developing composite applications. Advanced enterprises need to react quickly and effectively to new situations and require applications that can handle business events effectively, even unforeseen ones. This indicates the need for applications which allow users to be associated with automated business processes so that they can detect and handle exceptional situations. Composite applications, again, suit this need best, providing a process context and an overall process view, which crosses system boundaries intelligently. IT departments are also interested in gaining the ability to track and monitor processes which would otherwise fall to individuals. Introducing best practices and monitoring the quality of these processes provides important part of the reason for developing composite applications. Furthermore, as these processes are volatile, they need to be installed and changed in a very short period of time and often have limited validity.

While standard cross-industry applications can fulfill most standardized business requirements, there is also a need to provide and build applications which are either vertical solutions appropriate to a specific industry sector, country or region; or customer-specific developments that serve to create a competitive advantage for the customer. In this context, the ability to adapt applications and business processes to customers’ specific
needs becomes essential. The ability to penetrate highly vertical markets by providing a platform onto which ISVs can build composite applications is also an important aspect.

As a result, flexibility and a certain independence from underlying application systems is an essential aspect of composite applications. Flexibility is about defining and using components, but also about being able to offer an incremental platform, which gives ISVs and customers the required flexibility to create their solution independently of SAP release cycles and offerings. In addition, consistent monitoring and appropriate governance of the existence and use of composite applications are also required.

2.2 Basic Assumptions About Composite Applications

Looking at end users’, enterprises’ and software vendors’ requirements from a more technical perspective, it becomes clear that composite applications must have the following characteristics:

- Composite applications are based on services already provided by other applications or components. They combine available service operations with new application logic, user interfaces, and business process orchestration. These services can be provided by:
  - SAP’s new Business Process Platform
  - SAP’s enterprise SOA shipments (mySAP service enabling, see SAP’s Developer Network, Enterprise SOA – direct link: [http://esapreview.sap.com/socoview](http://esapreview.sap.com/socoview) for details of available services)
  - Service-enabled third-party solutions (service-enabled = self-contained enterprise functionality accessible via the Web Services standard)

- Composite applications are user-oriented applications that support cross-system collaboration. They have a user interface, although they might contain automated process steps requiring no user interaction.

- Composite applications are process-centric and manage business processes which span one or more underlying applications and/or components that are often data-centric. The process may include process steps from third-party or customer-proprietary software components.

- Composite applications have their own lifecycle. They can be built, packaged, deployed, and upgraded independently of the underlying applications or business process platform by SAP or ISVs. Consequently, composite applications establish a new logical application tier. This type of loose coupling also requires that there be no shared volatile states and no transactional locks between the composite application and its underlying components, thus imposing the requirement of a stateless service-oriented communication protocol.

- Composite applications often need to abstract from the components they are based on in order to be able to run in heterogeneous landscapes as well as various releases of the same backend system at customer sites without having to upgrade the main application systems. A particular requirement is that composite applications be able to cope with the existing configurations of the underlying components.

Composite applications require the service-enablement of the underlying applications or components. However, the applications on top of which we want to build composite applications often do not have interfaces a composite application can plug into, so that further development as well as further shipment and delivery processes are required to add those interfaces to the underlying components or applications.
3 Introducing the example scenario: Customer Service Care

In order to explain how to set about producing a specification for a composite application, we will illustrate each of the steps using an example. For this purpose we have chosen a “Customer Service Care” example. It covers the process of managing a customer complaint and is depicted as a swimlane diagram in Figure 1.

![Customer Service Care Diagram](image)

The background to the scenario is as follows: the fictive company ITelO produces computers and computer accessories. Thirty monitors have been delivered to a customer. Fifteen of them are damaged and Peter Miller, the customer’s Quality Manager, would like them to be replaced.

Peter Miller opens ITelO’s homepage, logs in and selects the appropriate link to enter a complaint. (Figure 2):
He enters his complaint in the window. (Figure 3 = process step 1):

**Enter Complaint**

- **Customer:** Panorama Studios
  - Peter Miller
  - 91602 Hollywood
  - Morning Drive 178
  - CA.

- **Order Number:** S0101658
- **Order Date:** 19.04.2006
- **Order Items:**
  - 10 HT-1030 20 Ergo Screen
  - 20 HT-1036 50 Flat Future

- **Complaint:** 15 of the Flat Future monitors don’t work. Please replace them!
  
  Best, P. Miller

**Figure 3**

When the complaint is saved in the database (process step 2), the process assigns the task to the customer’s account manager. Paul Burke will receive a notification telling him to look at the complaint. He checks a number of documents and reports on his customer (e.g. revenue figures for the last 5 years or
number of complaints made), but is still not sure what to do. Paul decides that he would like more information about the damage and to involve an independent expert from ITelO's insurance company. He adds an appropriate comment to the complaint and chooses the **Involve Expert** option (Figure 4 = process step 3).

![Figure 4](image)

The expert from the insurance company, Franklin Jones, receives an e-mail with a PDF document attached asking him to look at the damaged monitors. As it is a PDF document he can easily save it on his notebook and travel to the customer site to inspect the damage. On his train journey back he adds his recommendation (see Figure 5 = process step 6) and once he is online again he submits the data back to the application.
Figure 5

Paul Burke receives the recommendation from the expert and based on the information he has gathered so far, decides to accept the complaint. He informs a colleague about the case and asks him to initiate the replacement of the monitors (Figure 6 = process step 3).

Figure 6
As a result of Paul’s decision, the customer receives a notification that his complaint has been accepted (process step 4). An internal clerk will be informed that he has to update the order (from the context you can conclude that we are talking about a sales order; however for the purpose of simplicity we will use the term “order” throughout this document to refer to the sales order) for Peter Miller’s company (replacement of the monitors = process step 7).

This example demonstrates many aspects of a composite application and is thus ideal for explaining the main concepts involved in their development. Composites are:

- Highly collaborative
  The process involves several roles including the customer, the account manager, the expert, and the internal clerk and demonstrates neatly their interplay

- Reuse of existing functionality
  Complaints are always assigned to an existing order. Before the complaint is submitted to the composite the availability of the order the complaint is to be assigned to must be checked. This is an existing service provided by the backend system.

- Different business objects stored in different locations
  The composite will work on business objects such as customer, product, and order. These will typically exist in backend systems (and could also come from different backend systems). However, the complaint business object is new to the established ERP world and must therefore be modelled as a new BO stored in the composite itself.

- Interactive steps mixed with background steps
  As the process flow indicates (Figure 1) the complaint will be created by a service provided by the composite. All other steps are interactive steps. The responsible role is in charge of providing process relevant information by filling out appropriate UI’s. So the process consists of a mixture of background and interactive steps.

For the purpose of this guide and to keep the example to a reasonable size, we will focus on the first three process steps only.

**Note:** Although you do not have to explain the process in as great detail as it is done in this chapter, it might be a good idea to use this chapter as an example of how you can make your particular use case even more understandable to others. Feel free to describe your scenario in a word document (or any other tool you find convenient) following the approach outlined above. As UIs you can reuse the UI-mockups you create during the course of this guideline.
4 Guideline for Writing a Composite Specification

This guide is presented in form of an instruction manual which will guide you through the forms that have to be filled out in order to collect the information necessary to specify a composite. For each form field, we will also provide an example from the customer care scenario, as well as an explanation of the options from which the author can choose. The guide follows the typical sequence in which the forms have to be filled out. We recommend that you complete the forms in the order outlined below as this guarantees a structured approach from general to more detailed information. The forms appear in this order:

1. General Application Information
   Give a broad overview of the composite. Explain the basic idea and benefits one can expect from the final application.

2. Processes Overview
   A composite application usually consists of more than one business process. To get an overview of all the business processes that make up the composite, list them in this form with a short description of their function in the overall application.

3. General Process Information
   This form has to be filled out for every process that you listed in the previous form. It gathers more general information about the process: how is the process integrated and should the process be completed within a particular timeframe.

4. Roles
   Business process steps are either executed by people or by machines. It should be clear from the outset which roles are involved in a business process and who is responsible for each step. This form collects the role information for each process.

5. Visualization of Process Flow
   It is useful for business processes in particular, to display them graphically. To create a visualization of a process, the steps representing business activities must be identified and be arranged in a logical sequence. In addition the steps must be assigned to the roles listed in the 4th step. The visualized process flow, together with the role information give a very clear overview of the composite as it is planned.

6. Step Data
   For each of the process steps identified during the previous exercise a more detailed description is required. Typical questions are:
   - Is it an interactive or a background step?
   - Is it a mandatory step?
   - What happens from a business perspective during the execution of the step?
   - Is due date handling required?
   - …

7. Exception Handling
   For almost every service call, certain exceptional situations must be considered. It can happen, for example, that an order cannot be modified due to parallel changes being made in the backend system. These kinds of business exceptions must be specified, listed, and appropriate reactions need to be planned. This form helps to identify and define exceptions together with the reaction to them.

8. Business Objects Overview
   A composite application usually works on different business objects. To get an overview of all business objects that make up the composite, list them in this form with a short description of their relevance for the overall application.

9. Business Object
   Business applications work on data that comes from various business objects. Very often the application works only on a subset of the business object’s attributes. These attributes must be identified and specified in this form. Further, the relationships between business objects must be declared as well as information about possible business object representations in backend systems.

10. UI Mockup
    For interactive steps it is necessary to create a mockup of the UI. To do this you will need to identify
the fields required, possible field validations, and value help. The mockup will be described in more
detail by an accompanying form (see point 11).

11. UI Description
   The “UI description” form details the fields, tables, UI components etc. of the UI mockup. Additionally
   questions about the kind of access (read/write), whether a value help is needed, or regarding the
   kind of field validation have to be answered.

12. Services
   This final form lists all the services needed to fulfil the business functionality. In essence the input
   and output parameters are required together with a short description of the business logic. If this
   functionality is already encapsulated as a Web Service, the details about it should be provided as
   well.

**Note:** The questions the PM/BE has to answer over the course of the composite specification are sometimes
very detailed and therefore occasionally difficult to answer. If necessary answers can be left open or
postponed to a later stage. It is also a good idea to cooperate with the architect as early as possible on
certain more technically oriented topics such as business object specification or service specification. This
will increase the quality of the overall specification and result in less revisions being required at a later stage.
The aim of this guide is to achieve a high coverage of all aspects that influence a composite bearing all
essential criteria in mind. If you leave questions open you should be aware that this might have considerable
impact on the composite’s architecture and therefore on the effort to build it. The more information you
provide the architect with, the better he/she will be able to design the application.

We know that it is hard to find the right balance for the technical granularity of such a specification but by
explaining the questions in detail we hope to give you the right guidance.
4.1 General Application Information

**Form (Template): General Application Information**

- **1.** Describe the business problem the new solution addresses.
- **2.** Describe the idea behind the solution.
- **3.** Describe the market and the value of the solution (attach business case).
- **4.** Backend integration: solution should support several releases of the backend systems accessed.
- **5.** Basic numbers (e.g. #users, #business objects, access frequency).

**Figure 7**

This form is intended to summarize the basic idea behind the composite application. We recommend that you fill out this form and use it as a basis for discussions with management and other stakeholders. The preliminary decision on whether to embark on this project will be based on the content of this form. Once the decision has been made in favour of the composite, it makes sense to detail the solution by completing all the forms described in the following chapters.

**Fields**

1. In this field add a description of the problem the composite addresses. The description should explain the business environment (e.g. a business situation, a business domain, a particular situation) in which the deficiencies become obvious together with a summary of the consequences of this problem (e.g. loss of revenue, customer dissatisfaction, higher costs etc.). Point out whether there is an existing process that should be improved or if the process is missing entirely.

   **Example**: The Customer Complaint Management process as it is currently implemented in many companies, does not function efficiently. Typically data comes from different sources: telephone, e-mail, mail, etc. Additionally sometimes important information for making a decision on a complaint is not available or its retrieval is cumbersome. Sometimes the process comes to a halt because a responsible person is on vacation which leads to reduced customer satisfaction. Finally, the opportunity to gather valuable information about the quality of products is missed.

2. Once the problem has been identified in field 1 the probable solution must be outlined in this field. It is important to explain what the ideal process for this problem should look like. Describe how the current situation can be optimized. List arguments in favour of your proposed solution and explain why the new process improves the current situation. In essence external readers should be able to
grasp the fundamental idea of the composite by reading the explanations you enter in this field.

**Example:** By introducing a workflow system that is responsible for covering the entire Customer Complaint Management process, that can assign tasks to the right person at the right time, that can call services from different systems to collect appropriate information for decisions, and that allows quality relevant information to be gathered efficiently, the situation will be improved significantly!

3. If you want to sell the solution as a packaged composite application (and this will typically be the case for a PM) it is important to add some details about the target market. You should give an estimation about the expected market and a rough calculation about the expected revenue. The idea is to present a sketch of a business plan here. The BE on the other hand will provide calculations about the expected savings the new solution will bring.

In addition explain the values the new application will offer (e.g. improved efficiency or flexibility, reduced costs, increased customer satisfaction etc.).

**Example:** Market analysis shows that x% of companies have no or insufficient complaint management processes in place and that for y% of those companies customer service care processes play an important role. These companies are also willing to invest in appropriate solutions. From this the following figures can be derived:…

In summary the benefits of this solution will be: improved efficiency, reduced costs, increased quality of products, increased customer satisfaction

4. One of the key questions that has to be answered for a composite application is whether the final composite should be independent of the underlying IT landscape or not. By definition, composite applications sit on top of other applications. The question is now how deep the integration with the backend systems should be. If the final solution only needs to work within one company and as part of a well defined system landscape, it is absolutely valid to require no special precaution for landscape independency. On the other hand if you want to sell a solution to a number of customers or you want to ensure that the existing backend system could be replaced without impacting on the composite, it makes sense to plan an abstraction layer for the composite that shields the solution itself from the IT landscape. By introducing this abstraction layer you gain additional flexibility which inevitably comes at a price: from a development perspective all the data relevant to the composite must be explicitly modelled. So the development effort increases in proportion to the amount of data you handle within your application.

As a guideline for answering these questions, the following rule of thumb may help:
- Are you familiar with the system landscape (systems and releases) in which your composite will run?
  - if not, select **Yes** on the form (as you do not know the landscape, the composite should be as flexible as possible, so the additional abstraction layer makes perfect sense).
  - if you are familiar with the system landscape, you need to decide whether it will be satisfactory for the composite to run on this predefined system landscape.
  - if it will not be satisfactory for the composite to run on the existing system landscape, select **Yes** on the form (you know the landscape and you want your solution to be adaptable to other system landscapes)
  - otherwise select **No** on the form (you know the landscape and it will suffice if your solution is restricted to that particular landscape).

If this rule does not help you, we recommend that you discuss this topic with your system architect. However if you do decide to opt for a flexible solution that will run on several backend systems, it is useful to add a list of the systems you have in mind. Simply add the vendor, the application’s name and the release to this field (e.g. SAP R/3 4.6c or SAP mySAP ERP 2005).

5. Provide some basic figures for your composite. You should at least provide the expected number of users accessing the application and a rough estimate of the amount of data the application will have to handle (e.g. 1.5 million customer records; 750,000 products consisting of 2.7 million individual parts…). Any additional figures are very helpful (access frequency; availability requirements; response time requirements; etc.)

As an example of a completed form please see Figure 8:
Form (Example): General Application Information
(Executive Summary)

Describe the business problem the new solution addresses

The Customer Complaint Management process as it is currently implemented in many companies, does not work efficiently. Typically data comes in separately from different sources: telephone, e-mail, mail etc. In addition important information for making decisions on complaints may not be available or its retrieval is cumbersome. Sometimes the process comes to a halt because a responsible person is on vacation leading to reduced customer satisfaction. Finally, the opportunity to gather valuable information about the quality of products is missed.

Describe the idea behind the solution

By introducing a workflow system that covers the complete Customer Complaint Management process, that can assign tasks to the right person at the right time, can call services from different systems to collect appropriate information for decision making, and that allows quality relevant information to be gathered efficiently will improve the situation significantly.

Describe the market and the value of the solution (attach business case)

Market analysis show that x% of the companies have no or insufficient complaint management processes in place and that for y% of those companies customer service care processes play an important role. These companies are also willing to invest in appropriate solutions. From this the following figures can be derived:

In summary the benefits of this solution will be: improved efficiency, reduced costs, increased quality of products, increased customer satisfaction.

Backend integration: solution should support several releases of the accessed backend systems

<table>
<thead>
<tr>
<th>Yes</th>
<th>R/3 4.6c, mySAP ERP 2005</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td></td>
</tr>
</tbody>
</table>

Basic numbers (e.g. # users; # business objects; access frequency)

<table>
<thead>
<tr>
<th>#users: 200 per month...</th>
</tr>
</thead>
</table>

Figure 8
4.2 Processes Overview

**Form (Template): Processes Overview**

<table>
<thead>
<tr>
<th>Process title</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Enter the title of the process here&gt;</td>
<td>&lt;description comes here&gt;</td>
</tr>
</tbody>
</table>

**Figure 9**

This form is intended to give an overview of all processes that make up the composite application. The architect should get a first impression of the number of processes and the focus of each process. List all processes with the information as described in detail below.

**Fields**

1. Provide a meaningful name for your process. This name will be referenced from other forms, so make sure it is not too long.

   **Example**: Customer Service Care

2. Provide a short but precise description of what is to happen when the process is executed. Mention briefly the main steps in the process and their responsible roles.

As an example of a completed form please see Figure 10:
## Form (Example): Processes Overview

<table>
<thead>
<tr>
<th>Process title</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer Service Care</td>
<td>Focus: treatment of customer complaints. The customer enters the complaint which is handed over to the responsible account manager who is in charge of deciding on the complaint. If necessary, he/she can involve an independent expert to support him/her in making this decision. Once a decision is reached, the customer receives a notification and the order the complaint was associated to is updated.</td>
</tr>
</tbody>
</table>

**Figure 10**
4.3 General Process Information

Form (Template): General Process Information (to be filled out for each process)

Process title: <Enter title of process here>

<table>
<thead>
<tr>
<th>Information</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>When and how is the process integrated/started (consider any pre-conditions that have to be fulfilled before the process can start)?</td>
<td>&lt;description comes here&gt;</td>
</tr>
<tr>
<td>Due Date handling</td>
<td>&lt;description comes here&gt;</td>
</tr>
<tr>
<td>Number of parallel instances of this process (= number of concurrent users)</td>
<td>&lt;enter number here&gt;</td>
</tr>
</tbody>
</table>

Figure 11

This form collects basic data on each process that is part of the composite application. The information is needed to identify
- how the processes should be integrated and started
- whether the process should be finished within a certain timeframe and if so what should happen if this timeframe expires.

Fields

1. The process name the general process information should be applied to. Reuse the process name already provided on the “Processes Overview” form (see chapter 4.2 for details).

2. Describe when the process should be started and how the process should be integrated into the existing environment. There are two options here which are not mutually exclusive:

   - **Manually**: if you choose this start mode the process has to be started explicitly by pressing a button on a web form or following a link on a web/portal page for instance. In this case describe in detail where you would like to add this button/link and under which circumstances the link/button should be activated/visible.

   - **Automatically**: if you choose this option the process will be started when a particular event occurs. Such events can be the expiration of a timer for scheduled processes, the occurrence of certain business events (e.g. creation of an order, finishing data entry in a particular form etc.), or the occurrence of business exception (e.g. an order cannot be completed due to a delivery bottleneck of a certain product – see also field 4). Describe exactly when the event should be triggered.

If the process is to be executed on a regular, scheduled basis you have to define the following parameters:

**Schedule Type**: You can determine whether the process is to start only once or repeatedly. You can choose between several recurrence options, e.g. recurrent daily, weekly, monthly, or yearly.

**Priority**: Should the process run with low, middle, or high priority? If the process deals with critical situations, choose high; in all other cases decide for yourself how urgently a reaction is necessary.

**Error mode**: What should happen if the process cannot be started? For this scenario you can choose between different modes that either abort the processing or try to execute the scheduled process again. You can specify a maximum number of attempts to initiate the process in case of an error.

To fill out this field it might also be beneficial to think about pre-conditions which have to be fulfilled before the process can start. If you identify such pre-conditions list them here as well.
Example: Start option: manually. The process should be started every time a customer fills out the central complaint web page. Once the customer submits the form, the process should start. Pre-condition: An order must exist so that the customer can reference it in his complaint.

3. The purpose of due dates is to define a date by which a process or a process step must be completed. You can define a due date for the complete process and/or for each step in the process. For each due date an action has to be defined. This action will be executed once the due date expires. One typical example for such an action would be an email being sent out that notifies that the due date has expired. It is also possible to start another process in such a situation. Closely related to the due date itself is the notion of “notifications” being sent out or called prior to the expiration of the due date itself and acting as a reminder to the processor that the due date is close to expiration.

Example: For our Customer Service Care scenario we assume that a service level agreement has been negotiated between the manufacturer of the product and the customer. It states that the customer needs feedback on his/her complaint within 2 hours of making it and that the whole complaint management process must be completed within 2 days. Additionally it is a good idea to inform the account manager of this particular customer that the due date is about to expire, one hour in advance. The same is true for the expiration of the due date for the whole process. Half a day before the due date for the process expires, a notification should be sent to the current processor of that step so that he or she still has enough time to resolve the issue. The due date handling for this particular example would look like this:

At step level (e.g. the “Decide on Complaint” step):
Due Date: fires 2 hours after the start of the step. Action to be taken: send E-Mail.
First notification to be fired 1 hour after the start of the step. Action to be taken: send E-Mail,
(note: it is possible to define as many notifications as required)

At process level:
Due Date: fires 2 days after the start of the process. Action to be taken: send E-Mail.
First notification to be fired 36 hours after start of the process. Action to be taken: send E-Mail.

You can see from these examples that for each date (be it the due date itself or the notification) you have to define a point in time, when the due date is actually fired, combined with an action that has to be executed once the defined date is reached.

Options:
For defining due dates and notifications you have to consider the following:
1. In general due dates can be defined at process and step level
2. Due dates can be defined…
   a. …as absolute dates
   b. …as relative dates
3. You have to assign actions to each due date

Let us consider these options in more detail. The easiest way of defining a due date is to determine a concrete day such as 4/3/2006. To achieve this, add the following description to your step or process specification:

*Due Date Handling: Absolute, 4/3/2006; action: send E-Mail*

If you want to define a relative date, you have to provide information about the amount of time that must pass between a certain point in time and the due date firing. The general rule is:

[number] [Minutes, Hours, Days, Weeks, Months, Years] from [process start, action start]

To achieve this, add the following description to your step or process specification:

*Due Date Handling: Relative, 2 Hours from action start, action: send E-Mail*
Please remember that the **action** must be a callable service without a User Interface. It can be a call to a backend system, the start of another process, or a service for sending E-Mails. These services can either be pre-delivered by SAP (such as the service for sending E-Mails) or self-developed. As a rule, you can use all services that follow the Web Services standard (irrespective of the platform on which they are running as well as the programming language they are implemented in) as well as RFC’s.

4. Enter the expected number of parallel instances for this process in this field. You may restrict the number to exactly one instance. This is sometimes necessary to prevent the parallel modification of critical data. In this case, enter “**Maximum: 1**” in this field. In all other cases enter the estimated average number of parallel processes. This number indicates the expected load for the composite application and is identical to the number of concurrent users for the process.

As an example of a completed form please see Figure 12:

---

**Form (Example): General Process Information (to be filled out for each process)**

**Process title:** Customer Service Care

<table>
<thead>
<tr>
<th>Information</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>When and how is the process integrated/started (consider any pre-conditions that have to be fulfilled before the process can start)?</td>
<td>Start option: manually. The process should be started every time a customer fills out the central complaint web page. Once the customer submits the form, the process should start. Pre-condition: An order must exist so that the customer can reference it in his/her complaint.</td>
</tr>
<tr>
<td>Due date handling</td>
<td>Yes: due to SL agreements with customers this can vary. It would be great to keep it as a parameter. On average a complaint should be completed within 3 days: Due Date: fires 3 days after start of the process; action to be taken: send E-Mail First notification to be fired 36 hours after start of the process; action to be taken: send E-Mail</td>
</tr>
<tr>
<td>Number of parallel instances of this process (\text{= number of concurrent users})</td>
<td>approx. 60</td>
</tr>
</tbody>
</table>

**Figure 12**
4.4 Roles

Form (Template): Roles (to be filled out for each process)

Role Summary for Process: <Enter title of process here>

<table>
<thead>
<tr>
<th>Role</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;enter role name here&gt;</td>
<td>&lt;enter description here&gt;</td>
</tr>
</tbody>
</table>

Figure 13
For each process within the composite application, process roles have to be defined. These are the actors responsible for the execution of a process step. The roles have nothing to do with any users or groups configured in any user data store. Instead they should be defined solely from a business perspective. In the Customer Service Care process for example, a customer will fill out the complaint form and an account manager responsible for this particular customer is in charge of reacting to the complaint. So these roles are defined by the business case.

Fields
1. The process name the roles should be applied to. Reuse the process name from the “Processes Overview” form (see chapter 4.2 for details)
2. Provide a name for the role that describes it effectively
3. Describe the role, e.g. details about its responsibilities or relationships to other roles

As an example of a completed form have a look at Figure 14:
Form (Example): Roles (to be filled out for each process)

Role Summary for Process: Customer Service Care

<table>
<thead>
<tr>
<th>Role</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer</td>
<td>Customer who enters the complaint</td>
</tr>
<tr>
<td>Account Manager</td>
<td>Account Manager for that particular customer, who has to decide on the complaint</td>
</tr>
<tr>
<td>Expert</td>
<td>An external specialist for analyzing damaged hardware</td>
</tr>
<tr>
<td>Clerk</td>
<td>Account Manager’s colleague, responsible for order management</td>
</tr>
</tbody>
</table>

Figure 14
4.5 Visualization of Process Flow

Essential for the understanding of a business process is an appropriate visualization of the process. For this you can choose any suitable tool you like e.g. UML modelling tools, PPT, or Microsoft’s Visio with a BPMN shape (for details see http://www.bpmn.org/documents.htm; BPMN = Business Process Modeling Notation).

The visualization should at least provide the following information:

- the steps that make up the process and their execution sequence
- differentiation between interactive and background steps
- role assignment to the steps

As the list indicates, the key building blocks of a business process are steps. It is critical for the composite’s specification to identify the steps of the business processes it consists of. Consider that the granularity of a step is a business activity. Such a business activity is a self-contained business transaction like the creation of an order or the input of data for e.g. a flight booking. Additionally a business activity can be accomplished either interactively (executed by a person) or via service call (executed by a system).

As steps and business activities describe the same semantics in the domain of business processes, these terms will be used as synonyms from now on.

A business process flow defined as a sequence of business activities has to be distinguished from screen flows (also called “guided data entries” or “guided activities”: the terms will be used interchangeably throughout the text). Screen flows look from a user experience like business processes. In fact they simply collect data using different screens to finally accomplish one business activity. A typical example in the context of the Customer Service Care scenario could be the first step in the process - entering the complaint. This step could be separated in four screens:

- enter basic customer data
- enter basic order data the complaint should be associated with
- enter complaint data
- review all entered data and submit them to the backend system.

For each of the steps above a different screen can be used and you can navigate back and forth between the screens. Once you confirm the final screen, the business activity ends. The relationship between a business activity and a guided data entry is as follows:

A guided data entry makes up exactly one business activity and can be used as a step within a business process. A guided data entry is a collection of screens connected in form of a screen flow to assemble data for one business activity.

Following the above mentioned definition of business activities, you have to identify the steps that make up your process and put them in a proper sequence. If you use the swimlane diagram depicted in Figure 15 you can assign each of the steps to a particular lane. Each lane represents a particular role. As role names you choose the names already defined in the previous step – see 4.4 for details. By following this approach you automatically assign steps to roles. Background steps have to be assigned to the role on behalf of which the background call should be executed. As an example please see Figure 1 which depicts the process flow for our customer service care scenario. Different icons have been used to distinguish between interactive and background steps. We recommend that you assign numbers to each step for reference purposes. They will be reused as soon as you provide more detailed information for each step in the “Step Data” form (see chapter 4.6 for details).

From a process modelling perspective you can choose from the following patterns:

- activities can be executed in a sequence (definition: an activity in a workflow process is enabled after the completion of another activity in the same process)
- steps can be executed in parallel (definition: a point in the workflow process where a single thread of control splits into multiple threads of control which can be executed in parallel, thus allowing activities to be executed simultaneously or in any order)
  - the number of parallel branches can be provided as static information or be calculated during runtime
- synchronization (definition: a point in the workflow process where multiple parallel sub processes/activities converge into one single thread of control, thus synchronizing multiple threads. It is an assumption of this pattern that each incoming branch of a synchronizer is executed only once)
- exclusive choice (definition: a point in the workflow process where, based on a decision or workflow control data, one of several branches is chosen)
- simple merge (definition: a point in the workflow process where two or more alternative branches come together without synchronization. It is an assumption of this pattern that none of the alternative branches is ever executed in parallel)
- multi choice (definition: a point in the workflow process where, based on a decision or workflow control data, a number of branches are chosen)
- synchronized merge (closely related to multi choice; definition: a point in the workflow process where multiple paths converge into one single thread. If more than one path is taken, synchronization of the active threads needs to take place. If only one path is taken, the alternative branches should reconverge without synchronization. It is an assumption of this pattern that a branch that has already been activated, cannot be activated again while the merge is still waiting for other branches to compete)
- multiple merge (definition: A point in a workflow process where two or more branches reconverge without synchronization. If more than one branch gets activated, possibly concurrently, the activity following the merge is started for every activation of every incoming branch)
- arbitrary cycles (definition: a point in a workflow process where one or more activities can be done repeatedly)

* All definitions of the control flow patterns are taken from the web site “Workflow patterns” maintained by Prof. van der Aalst (http://is.tm.tue.nl/research/patterns/)
- multiple instances with a priori known design time knowledge (definition: for one process instance an activity is enabled multiple times. The number of instances of a given activity for a given process instance is known at design time. Once all instances are completed some other activity needs to be started)

- multiple instances with a priori known runtime knowledge (definition: for one case an activity is enabled multiple times. The number of instances of a given activity for a given case varies and may depend on characteristics of the case or availability of resources, but is known at some stage during runtime, before the instances of that activity have to be created. Once all instances are completed some other activity needs to be started)

- cancel case (definition: a case, i.e. workflow instance, is removed completely (i.e., even if parts of the process are instantiated multiple times, all descendants are removed)).

Once the visualization of the process flow is complete, the interactive and the background steps are identified, brought in a proper sequence, and assigned to the responsible roles. This result is used as the basis for defining more details about the steps. For this the form “Step data” has to be filled out. This is explained in the next chapter.

If you want to explain the process flow, the relevant business actions as well as the business story itself in greater detail we recommend that you write it in prose as we have done it in chapter 3 for our demo scenario. Take a look at chapter 3 for more information.

As an example for a visualized process flow please see the flow for the customer service care example on page 8, Figure 1.
4.6 Step Data

**Figure 16**

Once you have identified the business activities of a business process at a relatively high level you need to provide more detailed information about each step. For this the form depicted in Figure 16 has to be filled out with the following fields (one row for each step):

**Fields**

1. The process name the step data should be applied to. Reuse the process name already provided in the “Processes Overview” form (see chapter 4.2 for details)
2. For each step provide a number for referencing purposes. These must correspond to the numbers provided in the visualization of the process flow.
3. Provide a name for each step. Use meaningful names describing the business activity covered by this step and ones that also match the ones in your visualization. Very often you will be working on business data in a step so the action describing what you are doing with the business data supplemented by the business object’s name the data belongs to is ideal for a step name (i.e. a verb followed by a noun).

**Example:** “Enter Complaint”, “Create Complaint”, “Decide on Complaint” etc.

4. Describe in prose what should happen when each step is executed. Be guided by the question “Who is doing what how?”, or in more detail:
   a. On which data is the step working?
   b. What is the step doing with the data (create, read, update, delete)?

<table>
<thead>
<tr>
<th>Step No</th>
<th>Step Title</th>
<th>Step Description</th>
<th>Interactive (I) or Background (B)</th>
<th>Mandatory</th>
<th>Needed Services (numbers)</th>
<th>Due date handling required</th>
<th>Role</th>
<th>Exits Next Step</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
c. What does the business logic look like?

d. Which services are called when (see also field 7)?

e. If it is an interactive step, describe what you expect from the user.

    **Example:** The process role “Customer” enters a detailed description of the complaint.

5. Is the step executed interactively or does it call a service? Enter **I** for an interactive step and **B** for a background step.

6. Is it a essential to execute the step (e.g. important data has to be entered) or is the information not particularly relevant for the subsequent steps and could be left out? Enter **Yes** for mandatory steps and **No** for optional steps.

7. In the field **Needed Services** enter a list of numbers representing service calls. These numbers refer to the form 4.12 (Services) in which all the services required by the composite are listed. As we have not yet defined the services, you can leave this column empty for the time being and come back to it once the services have been identified and catalogued. However, with every number you enter in this field you express that the functionality provided by this step can only be achieved by using these services. During the step description (field 4) you should have explained when which service should be called and how these services work together to achieve a particular business goal.

8. The due date handling logic has been described in great detail already. Please refer to chapter 4.3 (General Process Information) field 3 for details.

9. Provide the role name of the actor for this step. The role name must be identical to the one provided during the visualization of the process flow.

10. You can assign so called **exits** or the number of the **next step** to a step. You will typically make use of these exits for interactive steps (but they are not restricted to interactive steps). Very often the user of a composite will make decisions during an interactive step. He or she accepts a leave request for example or he/she has to decide on a complaint. Depending on the decision the process flow will typically continue with a different step. To express these situations within this form, the **Exits** column has been included. If you want to make use of this functionality, give each of your planned exits a meaningful name such as “Involve Expert” or “Reject complaint” and add the number of the step the process should be continued with for that particular decision. If you do not intend to distinguish between several exits, add at least the “Default” exit supplemented by the number of the next step. Examples for a background steps with several exits are all calls to services which either return classifications (e.g. for our scenario we could check the customer’s classification by a background call which returns the classification such as class A-, B-, or C-customer) or which can calculate automatic decisions following certain business rules (e.g. if the customer’s revenue for the last five years is not constantly increasing return “Reject Complaint” otherwise return “Accept Complaint”). If you make use of these services provide all the possible exits plus their respective next steps as described before.

    **Example:** Step 3 of the Customer Service Care example called “Decide on Complaint” has 3 exits defined. A correct definition looks like this:

    Accepted → 4, 5  
    Rejected → 4  
    Expert → 6

    The definition above states that the process should be continued with
    - step 4, 5 if the account manager accepts the complaint
    - step 4 if the account manager rejects the complaint
    - step 6 if the account manager wants to involve an expert to get his or her recommendation

As an example of a completed form please see Figure 17:
### Form (Example): Step Data (to be filled out for each process)

#### Steps for Process: Customer Service Care

<table>
<thead>
<tr>
<th>Step No</th>
<th>Step Title</th>
<th>Step Description</th>
<th>Interactive (I) or Background (B)</th>
<th>Mandatory</th>
<th>Needed Services (numbers)</th>
<th>Due date handling required</th>
<th>Role</th>
<th>Exits Next Step</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Enter Complaint</td>
<td>Customer enters a lengthy description of the complaint</td>
<td>I</td>
<td>Y</td>
<td>2, 4</td>
<td>N</td>
<td>Customer</td>
<td>1. Default → 2</td>
</tr>
<tr>
<td>2</td>
<td>Create Complaint</td>
<td>Creates a new complaint</td>
<td>B</td>
<td>Y</td>
<td>1</td>
<td>N</td>
<td>Customer</td>
<td>1. Default → 3</td>
</tr>
<tr>
<td>3</td>
<td>Decide On Complaint</td>
<td>Manager decides on complaint</td>
<td>I</td>
<td>Y</td>
<td>-</td>
<td>N</td>
<td>Account Manager</td>
<td>1. Accepted → 4, 5, 2. Rejected → 4, 3. Expert → 6</td>
</tr>
<tr>
<td>4</td>
<td>Receive Reason</td>
<td>Customer receives decision</td>
<td>I</td>
<td>Y</td>
<td>-</td>
<td>N</td>
<td>Customer</td>
<td>1. Default → end</td>
</tr>
<tr>
<td>5</td>
<td>Message to clerk to update order</td>
<td>Manager creates message for clerk</td>
<td>I</td>
<td>Y</td>
<td>-</td>
<td>N</td>
<td>Account Manager</td>
<td>1. Default → 7</td>
</tr>
<tr>
<td>6</td>
<td>Evaluate Complaint</td>
<td>Expert gives recommendation</td>
<td>I</td>
<td>Y</td>
<td>-</td>
<td>Y (within 24 hours)</td>
<td>Expert</td>
<td>1. Default → 3</td>
</tr>
<tr>
<td>7</td>
<td>Update Order</td>
<td>Clerk updates original order</td>
<td>I</td>
<td>Y</td>
<td>3</td>
<td>Y (within 24 hours)</td>
<td>Clerk</td>
<td>1. Default → end</td>
</tr>
</tbody>
</table>

**Figure 17**

**Note:** In Figure 17 you will recognize due date definitions for steps 3, 6, and 7. Please make sure that these definitions harmonize with the ones defined at process level (see Figure 12 for details). Be aware of the distinction between the definition of due dates at process and at step level.
4.7 Exception Handling

**Form (Template): Exception Handling (to be considered for service calls)**

<table>
<thead>
<tr>
<th>Exception No</th>
<th>Exception</th>
<th># of step</th>
<th>Action</th>
<th>Continue/Retry/Terminate</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>&lt;enter type of business exception here&gt;</td>
<td>&lt;enter # of step here&gt;</td>
<td>&lt;describe appropriate reactions to the exception here; in addition refer to the UI (by the UI's name) that should be shown or to the process (by the process's name) that should be started if the exception occurs&gt;</td>
<td>&lt;C,R,or T&gt;</td>
</tr>
</tbody>
</table>

**Figure 18**

Within the table “Exception Handling” you have to define all business events that might occur during the execution of a process. Typically these will be events such as the erroneous call of backend systems (e.g. the creation/deletion of an order was not successful due to some business reason). You have to distinguish whether the exception occurs during a background step or during an interactive step. If the exception is thrown during a background step you have to define an appropriate action. Such an action can be a special web page telling the user what went wrong or you can refer to another process in the composite that has to be started if the exception occurs to recover from that situation. Additionally it is important to know how the process should continue once the exception handling step/process has been finished. Either the background step causing the exception should be executed again or the process should continue with the step following the background step or the process should be terminated.

For interactive steps you should at least list possible exceptions in this form here as well, simply to make clear that the UI has to cater for them. This means that the program working on the UI’s inputs has to react to these exceptions and it will not leave the process step until it is satisfactorily resolved. Note that the information on whether the exception is thrown within an interactive or background step can be derived from the step number it is assigned to (column 3).

Technical exceptions such as the breakdown of a backend system or network errors will not be documented in this table.

**Fields**

1. For each business exception provide a number for referencing purposes
2. Describe the business exception here. Provide details about the background to this exception, e.g. why it is thrown, a particular constellation that should be avoided, for instance. Consider all the circumstances that might cause a failure of a service call. For services provided by backend systems the list of exceptional situations should be part of the service’s documentation. You should not expect to have this list completed immediately. As you fill out the forms described in this guide you will identify more exceptional circumstances which will require you to come back to this form and adjust it accordingly.

**Example:** For the background step “createComplaint” the exception “User is not authorized to create the complaint” can be thrown. For the interactive step “Update Order” the exception “Conflicts with data change in backend system” can be thrown. This indicates that the data in the backend system...
has been changed in parallel to the execution of the composite. Due to this mismatch the changed data can not be saved.

3. Indicate which step this exception is valid for. Simply enter the number of the step listed in the “Step Data” form (see chapter 4.6 for details)

4. You need to specify what happens when the exception is thrown. Describe in prose what to do in order to react to the exception. If the exception is thrown in a background step specify either a UI that should be displayed or a (sub-) process that should be kicked off. For interactive steps describe in prose the possible reaction options.

**Example:** Offer a screen which allows a new user to be selected and the task to be delegated to a colleague. Once the colleague has confirmed the complaint the background call for creating the complaint has to be repeated.

5.

a. **Exception thrown in background step**

   After executing the UI/sub-process you described in the previous field, you have to decide what should happen when the UI/sub-process is finished. For this you have 3 options:

   - **Continue (C):** continue the interrupted process with the step following the background step causing the exception
   - **Retry (R):** continue the interrupted process with the background step causing the exception; due to the correction made during the exception handling UI/sub-process the call should work now
   - **Terminate (T):** it makes no sense to continue the interrupted process. So terminate the whole execution.

b. **Exception thrown in interactive step**

   For interactive steps the proper exception handling will be part of the associated UI program for which you have already listed possible reactions. However, by setting this field you indicate how the process is to proceed for interactive steps as well. Use the same abbreviations introduced in the previous bullet point.

**Note:** if you refer to a sub-process for solving an exception it might be a good idea to visualize the complete process flow (that is the basic business process plus the exception handling sub-process) in one swimlane diagram to get a complete overview of the process landscape for this particular business process. It helps the architect to identify process conflicts (deadlocks, racing conditions etc.) and it helps him or her to find reusable process sequences that can be used in each of the processes.

As an example of a completed form please see Figure 19:
## Exception Handling

<table>
<thead>
<tr>
<th>Exception No</th>
<th>Exception</th>
<th># of step</th>
<th>Action</th>
<th>Continue/Retry/Terminate</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>User is not authorized to create the complaint</td>
<td>2</td>
<td>Offer a screen which allows a new user to be selected and delegate the task to a colleague, when the colleague has confirmed the complaint, the background call for creating the complaint has to be repeated.</td>
<td>R</td>
</tr>
<tr>
<td>2</td>
<td>When checking the availability of an order for a certain customer, it was detected that the customer doesn’t exist.</td>
<td>1</td>
<td>Should not happen as the customer id is derived from the customer’s logon. Therefore no action needed.</td>
<td>R</td>
</tr>
<tr>
<td>3</td>
<td>Order cannot be updated due to changes to the order in the backend system (conflict with parallel actions on the backend system itself)</td>
<td>7</td>
<td>Display an error message explaining the mismatch of the data, ask the user to check the new data, and redo the transaction.</td>
<td>T</td>
</tr>
<tr>
<td>4</td>
<td>When reading details about a customer it was detected that the customer doesn’t exist.</td>
<td>1</td>
<td>Should not happen as the customer id is derived from the customer’s logon. Therefore no action needed.</td>
<td>R</td>
</tr>
</tbody>
</table>

Figure 19
4.8 Business Objects Overview

This form is intended to give an overview of all the business objects the composite application works on. The architect should get a first impression of the number of business objects and their role within the composite. So list all business objects with the information described in detail below.

**Fields**

1. Provide a meaningful name for your business object. This name will be referenced from other forms, so make sure it is not too long.

   **Example:** Complaint or Order (please remember: in our example this means the sales order!)

2. Provide a short but precise description of what the business object is for and what happens with the BO during the execution of the application. Mention briefly the BO’s most relevant fields and relationships to other business objects.

   A good source for the description of a business object is SAP’s Enterprise Service Workplace (ES Workplace). We recommend that you have a look at this before you provide your own definition of a BO. In this way, you make sure that you follow (and benefit from) approved business standards and that you are not ‘reinventing the wheel’. The ES Workplace is located on SAP’s Developer Network (SDN) and you can gain access to it as follows:
   - Call the SDN homepage: [http://www.sdn.sap.com](http://www.sdn.sap.com)
   - In the left-hand navigation expand the “Enterprise SOA” node
   - Click on the “ES Workplace” node

   Within this page you will identify another “ES Workplace” link which guides you to the contents of ES Workplace. There you can browse the available enterprise services. You can also directly call this page by entering the URL [http://esapreview.sap.com/socoview](http://esapreview.sap.com/socoview). In Figure 21 for example, the definition of the Customer BO is taken from the ES Workplace and enhanced by a short remark about the role of the BO in this particular scenario.

As an example of a completed form please see Figure 21:
## Form (Example): Business Objects Overview

<table>
<thead>
<tr>
<th>Business Object’s Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Complaint</td>
<td>Holds data about the complaint entered by the customer (e.g. complaint text, reference to the customer entering the data, reference to the order this complaint is associated to etc.)</td>
</tr>
<tr>
<td>Order</td>
<td>Every complaint must be associated to an order. The order object contains several line items. One of the line items must contain the product the complaint refers to.</td>
</tr>
<tr>
<td>Product</td>
<td>Each line item of an order refers to exactly one product business object. It contains detailed information about the product itself (e.g. name, product number etc.).</td>
</tr>
<tr>
<td>Customer</td>
<td>A business partner to whom materials or services are offered or provided. The business object stores detailed information about the customer entering the complaint.</td>
</tr>
</tbody>
</table>

**Figure 21**
### 4.9 Business Object

#### Form (Template): Business Object Data (to be filled out for every Business Object)

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Length/Number Range</th>
<th>Mask - Pattern</th>
<th>New</th>
<th>Used in Step</th>
<th>Sample Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
</tr>
</tbody>
</table>

**Business Object:** <enter name of BO here>

**Relationships for Business Object:** <enter name of BO here>

<table>
<thead>
<tr>
<th>To Business Object</th>
<th>Relationship Type (Composition / Association)</th>
<th>Cardinality</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;enter name of related BO here&gt;</td>
<td>&lt;Association or Composition&gt;</td>
<td>&lt;0..1, 1..1, 0..n, 1..n&gt;</td>
</tr>
</tbody>
</table>

---

**Figure 22**

Each composite application works on a well-defined set of data fields and it is a challenge for the product manager or business expert to define them as precisely as possible. To achieve this goal a series of tables have been developed to support the PM/BE in collecting the essential information about the data fields the composite actually works on. It is important to understand that only the data fields relevant to the composite application have to be defined and assigned to business objects. Although there might be existing business objects in backend systems with any number of fields, it is crucial to concentrate on the relevant ones only.

By walking through the business process, the PM/BE should be able to identify the data fields each step works on. Once this list of data fields is available, the fields have to be sorted and assigned to business objects. For example, an order entry program needs to work with concepts such as orders, invoices, and so on. Each of these may be represented by a business object. For our Customer Service Care example the composite works on business objects such as order, customer, and complaint. The objects together make up the domain the composite works in and each domain has its own particular set of business objects also called domain objects. It is the task of the PM/BE to identify the domain objects for his composite and assign the fields to them. Note that for this guide we are talking about business objects from a business perspective, which has nothing to do with the subsequent technical implementation of that business object. The implementation (that is, the “technical business object”) the architect is responsible for can be much more fine granular than the business object you specify here. A typical example that explains this quite nicely is the “Employee Times” business object. From a business perspective it collects all times relevant in an employee’s business life (e.g. absent time). For the technical implementation it will very probably be necessary to create more fine granular business objects like “Leave Request”. Yours should always be the business perspective. The tailoring into technical objects is then part of the architect’s job.

After identifying the objects and assigning the fields to them this information has to be entered in the forms we are going to discuss now.
Table “Business Object” (see Figure 22, upper table for details)

Fields

1. The name of the business object you are specifying detailed information for. Reuse the business object's name already provided on the “Business Objects Overview” form (see chapter 4.8 for details)

   **Examples:** Complaint or Order

2. Provide a meaningful name for the BO field

   **Example:** complaintText or orderDate or quantity

3. Provide the data type of the BO field. Choose one of the following: Integer, Float, Boolean, Date, String

4. If length of the BO field needs to be restricted (text fields of type String, for example) state this here. For numeric fields (Integer, Float) it is sometimes useful to define a number range. If this is the case, describe it in this field. If neither a length restriction nor a range definition is applicable simply enter Not relevant or leave it blank.

5. If the BO field requires the input of predefined values or the input has to adhere to particular formatting rules, specify these details in this field. For predefined values simply list them. For BO fields which require a particular format well established patterns can be reused such as DD.MM.YYYY HH:MM for date/time or N for a numeric placeholder and C for a placeholder representing a character. If a fixed character has to appear at a particular position, put this letter or digit or set of letters/digits in inverted commas. In all other cases add in prose what the input should look like.

   Be aware that this information is especially useful for displaying and data entry purposes. Do not expect that the data will be saved this way in the database or in any other external systems or that the data will be handed over to services this way. For this, standardized means must be used which are completely neutral to UI relevant aspects. In order to harmonize the business information for Enterprise Services (and hereby for Business Objects) SAP's Global Data Types (GDT) are based on international standards such as the Core Component Technical Specifications (CCTS). More details about GDT and CCTS can be found in an excellent SDN article you can access here: https://www.sdn.sap.com/irj/servlet/prt/portal/prtroot/docs/library/uuid/b602d790-0201-0010-e3a8-9e4ddfc45d17.

   **Example:** the internal customer ID has a particular pattern
   - the length of the field should be 10
   - it has to start with the letter ‘x’ or ‘y’ (case insensitive)
   - followed by 4 digits
   - followed by the character ‘H’ (case sensitive)
   - followed by 3 characters
   - end with the digit 9.

   In this case the pattern is as follows: ‘XxYy’NNNN’H’CCC’9’

6. When you assemble BO’s fields that are relevant for the composite you are likely to find out that you can distinguish the data between different storage locations: data that has already a representation in a backend system and other that is completely new. Even within one business object it can happen that some fields already exist and others do not. Enter the appropriate information in this field:

   - enter **Yes** if the field is not mapped against any backend system
   - enter **No** if the field is mapped against backend systems and a mapping exists for each of the systems you want to support
   - enter **Mixed** if the field is mapped against backend systems but there is at least one backend system you want to support for which a mapping doesn’t exist

   Remember that you have to provide additional information in the mapping table (to be discussed later) for every field that is stored in a backend system.
7. It is important to know in which steps this BO field is used. List the numbers of all those steps which rely on this particular BO field (the step numbers refer to the relevant step of the Step Data form; see chapter 4.6 for details).

8. Finally give an example of possible contents for this BO field.

Once you have listed the business objects together with their respective fields, you have to define the relationships between the business objects. With relationship the natural relationship of business objects in the business world is meant. For example an order might have no, one, or many complaints associated to it. So there is a relationship between the order business object and the complaint business object. On the other hand the complaint is associated to exactly one order. Another example would be the relationship between order and customer: each order is assigned to exactly one customer.

These kinds of relationships have to be defined for each business object.

**Table “Relationships for Business Objects” (see Figure 22, lower table for details)**

**Fields**

9. Enter the name of the business object to which you want to specify a relationship.

10. We differentiate between two relationship types: an **association** and a **composition**. To distinguish between the two ask yourself what should happen if the business object you are currently defining with this form will be deleted. Should the dependent business object (that is the business object you want to define the relationship to) be deleted as well or should it remain untouched? If it should be deleted as well, enter **Composition** in this field, otherwise enter **Association**.

   **Example:** The relationship between an order and a customer is of type association as you obviously don’t want to remove the customer if an order is deleted.

11. Define the cardinality between the two objects. You have four options to choose from:
   - 0..1: there can be a relationship to exactly one related object
   - 1..1: there must be a relationship to exactly one related object
   - 0..n: there can be a relationship to many of the related objects
   - 1..n: there must be at least one relationship to the related object

   **Examples:**
   - 1..1: Order to Customer: every order must be assigned to exactly one customer. An order without a customer does not make sense.
   - 1..1: Complaint to Order: every complaint must refer to the appropriate order.
   - 0..n: Order to Complaint: an order can be referenced by many complaints but it is also possible for an order not to be referenced at all (and this should be the normal case ;-)
### Business Object: Complaint

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Length/ Number Range</th>
<th>Mask - Pattern</th>
<th>New</th>
<th>Used in Step</th>
<th>Sample Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Complaint Text</td>
<td>String</td>
<td>1024</td>
<td>Not relevant</td>
<td>Y</td>
<td>1,2,3,4,5,6,7</td>
<td>5 of the new monitors are damaged!</td>
</tr>
<tr>
<td>ComplaintId</td>
<td>String</td>
<td>36</td>
<td>Not relevant</td>
<td>Y</td>
<td>2</td>
<td>123456789012345678901234567890123456</td>
</tr>
<tr>
<td>Complaint Date</td>
<td>Date</td>
<td>0</td>
<td>DD’.MM.’YYYY</td>
<td>Y</td>
<td>1,2,3,4,5,6,7</td>
<td>25.08.2006</td>
</tr>
</tbody>
</table>

### Relationships for Business Object: Complaint

<table>
<thead>
<tr>
<th>To Business Object</th>
<th>Relationship Type (Composition / Association)</th>
<th>Cardinality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Order Item</td>
<td>Association</td>
<td>1 : 1</td>
</tr>
<tr>
<td>Customer</td>
<td>Association</td>
<td>1 : 1</td>
</tr>
</tbody>
</table>
As we have already explained, (see discussion of field 6 on page 37) several business object fields may already exist in various backend systems. For the development of a composite application the information about the backend system as well as details about the appropriate business object and fieldname in the backend system are essential. For this fill out the fields in the following table (please make sure you provide the appropriate information for every field you marked with **No** or **Mixed** in the **New** column of the Business Object form (see Figure 22 for details).

Table “Mapping Information” (see Figure 24, upper table for details)

**Fields**

12. Repeat the name of the business object you are currently working on (same as field 1)
13. Enter the name of the BO’s field you want to specify a relation for
14. Provide details about the system the BO resides in. Essential are the **system type** (ERP, CRM, SCM, SRM, PLM, etc), the **vendor**, and the **release**.
15. Enter the name of the business object as it is defined in the backend system.
16. Enter the name of the BO’s field as it is defined in the backend system.

**Note:** For backend systems that are already service enabled, the BO itself is hidden by service calls. The BO’s fields that have to be mapped against the backend systems are therefore nothing other than parameters of appropriate service calls. To apply the right mapping for those backend systems provide in field 15 the service’s name and in field 16 the name of the associated service parameter.
Once you have completed the mapping information, you will need to answer a few final questions about the expected number of business objects and their usage.

Table “Additional Information” (see Figure 24, lower table for details)

Fields

17. Estimate the number of business objects you expect over time. If it is master data (such as customer or product) you will probably have a figure at hand and it will probably not change significantly. For transactional data estimate the rate of its creation, update and deletion.

18. Regarding the usage of a business object we basically distinguish between the lifecycle methods create, read, update, and delete. Please provide information on how the current business object will be used in your composite. If it is read only, enter the capital letter R, if it is created and updated, enter the letters C and U and so on. Any combination is possible and please consider the whole composite not only one business process.

19. Sometimes business objects are used as temporary storage during the lifecycle of a business process. Once the business process reaches its end the BO’s data is, for example, stored in a backend system. In this field state how you want your business object to be used. Simply answer with Yes or No depending on your particular use case.

20. This question has to be answered for BOs with backend representation only: can the business object be modified in the backend system (e.g. via another application running on the backend) while it is being used in the composite? Simply answer with Yes or No.

As an example of a completed form please see Figure 25:

Form (Example): Business Object Data cont’d

Mapping Information for Business Object: Customer (only applicable if fields already exist in backend systems):

<table>
<thead>
<tr>
<th>Field</th>
<th>System/SAP Component</th>
<th>System-SAP-specific Business Object or Service</th>
<th>Field or Service Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>CustomerName</td>
<td>mySAP-ERP 2005</td>
<td>CUSTOMER</td>
<td>NAME1</td>
</tr>
</tbody>
</table>

Additional Information:

<table>
<thead>
<tr>
<th>Information</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>How many BO’s of this type do you expect for the composite?</td>
<td>approx. 1,000,000</td>
</tr>
<tr>
<td>How will the BO be used in the composite (Create, Read, Update, Delete)?</td>
<td>R</td>
</tr>
<tr>
<td>Will the BO survive the business process?</td>
<td>Yes</td>
</tr>
<tr>
<td>If the BO exists in a backend system: can the BO be changed in the backend while it is being used in the composite?</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Figure 25
4.10 UI Mockup

The next step in specifying a composite application is the description of all the interactive steps. This is separated into two phases: first you have to create a mockup of the screen layout as depicted in Figure 26 or Figure 27. For creating a mockup take a tool of your choice. We recommend using Microsoft PowerPoint for its broad distribution and ease of use for mockup creation.

The second step is a detailed definition of each UI field. This is explained in the next chapter 4.11 (UI Description).

**Fields**

1. Enter the number of the step the UI is assigned to (the step number refers to the relevant step in the Step Data form; see chapter 4.6 for details).

2. Provide a meaningful name for the UI. Ideally the name should describe what happens on the screen.

**Examples:**
- Enter Complaint
- Decide on Complaint
- Update Order

3. For the mockup itself, place all the necessary fields along with their UI components on the screen and explain by adding notes what you want to achieve (see Figure 27 for an example). It is important to understand that you should not make the UI mockup too detailed. Remember that it is more important that the UI fits in the overall process flow (are all important data fields available?) than the design of the UI or whether you have selected the right UI components. Concentrate more on the
fields and their business process relevant contents than on the layout!
The following table should be regarded as a suggestion of UI components you can choose from for creating a mockup but it is not the intention of this guide to explain the exact functionality covered by each component in detail (nor should you rely on a particular functionality).
In chapter 4.5 (Visualization of Process Flow) we introduced guided data entry screens as screenflows to gather relevant data for one transaction. Although you should work out a UI mockup, guided data entries are NOT relevant here either. The decision to spread UI fields over more than one screen should be taken by the architect.

Here is the table of UI components:

<table>
<thead>
<tr>
<th>Calendar</th>
<th>July 2006</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SU</td>
</tr>
<tr>
<td></td>
<td>26</td>
</tr>
<tr>
<td></td>
<td>29</td>
</tr>
<tr>
<td></td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>29</td>
</tr>
<tr>
<td></td>
<td>5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Checkbox</th>
<th>Bool1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Date Picker</th>
<th>Date2: 07/21/2006</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Drop-down List</th>
<th>Entry 1</th>
<th>Entry 2</th>
<th>Entry 3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Input Field</th>
<th>Str3:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>List box</th>
<th>Entry 1</th>
<th>Entry 2</th>
<th>Entry 3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Plain Text</th>
<th>Plain Text Label</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Pushbutton</th>
<th>Btn1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Radio Group</th>
<th>Option 1</th>
<th>Option 2</th>
<th>Option 3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Str5:</th>
<th>Option 3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table

<table>
<thead>
<tr>
<th>Str1</th>
<th>Date1</th>
<th>Str2</th>
<th>Str3</th>
<th>Lbl1</th>
<th>Btn1</th>
<th>Bool1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure 27**
4.11 UI Description

Now that you have defined your UI visually it is time to explain each of the fields in more detail. For this fill out the form shown in Figure 28 for every component you placed in the UI.

### Fields

1. Enter the number of the step the UI is assigned to (the step number refers to the relevant step in the Step Data form and must be identical to the one provided for the mockup screen; see chapter 4.6 for details). If the UI is reused in other steps you can enter more than one step here.

2. Enter the name of the UI in this field. Make sure that it is identical to the one provided for the mockup (see previous chapter).

3. Forms can be handled online and/or offline. In most cases the user of a composite will fill out screens and submit their contents immediately to the application. These are typical online scenarios everyone knows from using SAP or any other ERP software. But with the advent of new technologies such as interactive forms based on Adobe technology it is possible to download a PDF form, save the form locally, e.g. on a laptop, and fill it out later. Once the user is online again he/she can submit the data from the PDF form to the composite. As this is a new approach to handling forms it makes sense to have a more detailed discussion about it. In the appendix a special chapter about interactive forms has been added giving you more insight about this technology: Chapter 6.2 (Interactive Forms) on page 65 discusses

- what interactive forms are all about
- how to use interactive forms
- when to use interactive forms.
As a result of this discussion you should now be able to decide how your UI is to be implemented. Simply enter **Yes** or **No** for the usage type (online or offline or both).

4. If the UI field is associated with a business object you have already defined, enter the business object's name here. For calculated or derived fields, leave this column empty.

5. For UI fields associated with a business object, enter its fieldname here. Both names must be identical to the ones provided when defining the business object (see chapter 4.8 Business Objects Overview, for details). For calculated or derived fields enter an appropriate and meaningful fieldname.

6. Is the UI field part of a table – i.e. does the field represent a column name? If so enter **Yes** here, otherwise **No**.

7. Is the UI field a read-only field or can it be modified? Enter **R** for a read-only, **M** for a modifiable field. It is particularly important to know how read only fields should be prefilled. Typically this will be either by a service call (if so, list the number of the service here), from the field of an already existing business object (if so name the business object and its field here), or calculated somehow from existing fields (if so describe how the calculation is made and which fields are involved). For modifiable fields it is sometimes useful to provide default values (e.g. the complaint date field can be prefilled with the current date as default). If this is the case describe how this should be done as you would do it for read-only fields.

8. If your UI mockup does not allow a conclusion to be drawn on the UI component to be used for this field, specify this in this column. To identify possible UI components, have a look at the components listed in chapter 4.10 (UI Mockup) for field 3. You can also provide additional information in prose if required.

**Example:** The Complaint Date field should be represented on the UI as a Date Picker UI component.

9. Does the user have to fill out the field? If so it is a mandatory field and you have to enter **Yes** in this column, otherwise **No**.

10. For certain UI fields a value help must be provided to define the set of valid entries for the particular field. Typically these values are retrieved by a service. Indicate whether the current field needs a value help (**Yes** or **No**), and if so add the number of the service which computes the values. This number refers to form 4.12 (Services) in which all the required services for the composite are listed.

11. For field validation you have three options:
   a. You do not want any field validation at all: simply leave the column field empty
   b. A rule based validation is to be applied: describe in prose the rule that should be used (several rules are possible as well)
      **Example:** for a date field: date must not be in the future
   c. The validation has been provided by a backend service: add the number of the service which implements the validation (more than one service is also allowed – just list the numbers of the services). This number refers to form 4.12 (Services) in which all required services for the composite are listed.

12. Give an example of a valid entry for this UI field.

As an example of a completed form please see Figure 29:
Form (Example): UI data for mockup (to be filled out for every interactive step)

UI Description for step: 1 (Enter Complaint)

Online Form: Yes
Offline Form: No

<table>
<thead>
<tr>
<th>Business Object</th>
<th>Field</th>
<th>Table &lt;Y/N&gt;</th>
<th>Read / Modify</th>
<th>Additional Info</th>
<th>Mandatory &lt;Y/N&gt;</th>
<th>Value Help (if yes, which service?)</th>
<th>Field Validation (only applicable for input fields)</th>
<th>Sample Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer</td>
<td>Customer Name</td>
<td>N</td>
<td>R (filled by service 4)</td>
<td>Text Input Field</td>
<td>N</td>
<td>Y</td>
<td>Only digits allowed</td>
<td></td>
</tr>
<tr>
<td>Order</td>
<td>OrderId</td>
<td>N</td>
<td>M</td>
<td>Text Input Field</td>
<td>Y</td>
<td>N</td>
<td>Only digits allowed</td>
<td></td>
</tr>
<tr>
<td>Complaint</td>
<td>Complaint Text</td>
<td>N</td>
<td>M</td>
<td>Text Editor</td>
<td>Y</td>
<td>N</td>
<td>6 of the new monitors are damaged</td>
<td></td>
</tr>
<tr>
<td>Complaint</td>
<td>Complaint Date</td>
<td>N</td>
<td>M</td>
<td>Date Picker</td>
<td>Y</td>
<td>Y, intrinsic by date picker</td>
<td>Value must not be in the future</td>
<td>12.07.2006</td>
</tr>
</tbody>
</table>

Figure 29
### 4.12 Services

**Form (Template): Service (to be filled out for every service)**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service Name</td>
<td><code>&lt;enter service name here&gt;</code></td>
</tr>
<tr>
<td>Input Parameters</td>
<td><code>&lt;list all input parameters here; each parameter name must follow the pattern „BO_Name.BO_Field“, e.g. Customer.FirstName&gt;</code></td>
</tr>
<tr>
<td>Output Parameters</td>
<td><code>&lt;list all output parameters here; each parameter name must follow the pattern „BO_Name.BO_Field“, e.g. Customer.FirstName&gt;</code></td>
</tr>
<tr>
<td>Exceptions (enter number of exceptions from the „Exception Handling“ form)</td>
<td><code>&lt;list numbers of all business exceptions here (must be listed on the „exception handling“ form)&gt;</code></td>
</tr>
<tr>
<td>Business Logic (Prose)</td>
<td><code>&lt;describe what happens during the service call&gt;</code></td>
</tr>
<tr>
<td>Relying on which Services</td>
<td><code>&lt;list all service numbers this service needs to fulfill its functionality&gt;</code></td>
</tr>
<tr>
<td>Mapped Against External Service</td>
<td><code>&lt;if available, name the external system’s service name&gt;</code></td>
</tr>
<tr>
<td>External Service Running On (System Information, Release etc.)</td>
<td><code>&lt;if available specify details about the system the external service is running on&gt;</code></td>
</tr>
</tbody>
</table>

**Figure 30**

Key building blocks of a composite application are the services on which a composite works. In a service-oriented world they represent the Lego blocks of a composite. As we have made clear throughout this guide, services play a central role for almost every step of the composite’s business processes. Therefore this form is introduced to describe the required services as precisely as possible.

Please ensure that you have checked SAP’s Enterprise Services Workplace web page (ES Workplace, http://esapreview.sap.com/socoview) for available enterprise services before filling out this form!

**Fields**

1. For each service provide a number for referencing purposes.
2. Provide a meaningful name for the service. Ideally the name should express what the service does. When providing names for services which modify data you can help the architect who will be responsible for the technical implementation of the service by following a certain naming conventions. For modifying services in particular, it is important for the architect to know how the modifications need to be saved. Imagine that two users of a composite are independently working on the same business object data. What is to happen if both write back the modified data of that same business object? Should the second user simply overwrite the changes his/her colleague made or should he/she receive an error message telling him/her that the data has been changed in the meantime? You can distinguish between these situations by providing different names for the relevant service. By convention use **change** as the prefix for the service if you want to express that the data should be overwritten without notification. Use **update** as prefix if the service should first check for changes and apply the modifications only if no data changes have been made in the meantime. Be aware of this distinction and help the architect by providing the appropriate prefix.
Example: createComplaint, changeComplaint (simply overwrite data), updateComplaint (first check data and if the data has been modified, return error message otherwise update database)

3. As the business objects with their fields have already been defined, simply list the fields the service needs as input parameters. In order to identify the fields more easily, apply the following naming convention for each field: BO_name.BO_Field → the name of the business object followed by a dot followed by the field name. The information (BO_Name, BO_Field) can be retrieved from the form Business Object (see chapter 4.9 for details).

Example: Complaint.complaintText; Order.orderDate

4. Apply the same rule from the previous step for the output parameters.

5. Services will certainly identify business situations for which the execution of the service no longer makes sense. They will throw an exception. These exceptions have already been declared and you only have to list all references to these exceptions which can occur for the current service. In the chapter about exception handling we mentioned that potential exceptions can be derived by reading the service descriptions from the providing backend systems. If necessary the table containing all exceptions will need to be updated. More about exceptions and their numbers can be found in chapter 4.7 (Exception Handling) on page 31.

6. Describe in a few words the business logic provided by the service. Explain which business objects participate, how they interact, and which business goal is achieved once the service call returns. Do not forget to describe error situations and how they are expressed by the output parameters.

7. Sometimes a service relies on other services to accomplish its task (another term for this kind of service usage is “service composition”). Simply list the numbers of those services in this field.

8. If the service has already been implemented in another system, enter detailed information about the service and the system it is running on (see also next field). In the “Mapped against external service” field enter the name of the service in the external service.

9. In this field enter details about the system the external service is running on. Essential are the system type (ERP, CRM, SCM, SRM, PLM, etc), the vendor, and the release.

As an example of a completed form please see Figure 31:
## Form (Example): Service (to be filled out for every service)

### Service Number: 1

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service Name</td>
<td>Create Complaint</td>
</tr>
<tr>
<td>Input Parameters</td>
<td>Customer.CustomerId Order.OrderId Order.OrderItemId</td>
</tr>
<tr>
<td></td>
<td>Complaint.ComplaintText</td>
</tr>
<tr>
<td></td>
<td>Complaint.ComplaintDate</td>
</tr>
<tr>
<td>Output Parameters</td>
<td>Complaint.ComplaintId</td>
</tr>
<tr>
<td>Exceptions (enter number of exceptions from the &quot;Exception Handling&quot; form)</td>
<td>1</td>
</tr>
<tr>
<td>Business Logic (Prose – describe what happens)</td>
<td>A new complaint business object has to be created</td>
</tr>
<tr>
<td>Relying on which Services (list all services this service needs to fulfill its functionality)</td>
<td>-</td>
</tr>
<tr>
<td>Mapped Against External Service</td>
<td>- (composite internal)</td>
</tr>
<tr>
<td>External Service Running On (System Information, Release etc.)</td>
<td>- (composite internal)</td>
</tr>
</tbody>
</table>
5 Handover of Specification to Architect

Now that the last form has been completed, it is time to handover the specification to the system architect. The forms will allow him or her to make sound decisions as they provide the following information:

- Details of the business concept behind the composite; the problem that is addressed and what improvements can be expected
- Overview of the process flow with the participating process roles
- Detailed explanation of each process step for both interactive and background steps
- Summary of all business exceptions that can occur including a description of the reactions to them
- A complete list of all business objects participating in the composite, their fields, and relationships to other business objects
- Mockups for interactive steps
- A list of all required services with their parameters and exceptions
- An overview of the backend systems intended to be supported by the composite

All these details together make up a comprehensive specification of the composite which will help the architect to find the right solutions for the different aspects of the composite’s development process.

We recommend that an official meeting is arranged for the product manager / business expert and architect to discuss the specification and clarify any potential misunderstandings arising from the descriptions. At the end of that meeting the architect should understand the composite’s intention so that he or she can begin designing the architecture by following the Architecture Guide. The Architecture Guide is a handbook closely related to this guide and written to help architects consider the options they have for the different layers of a composite. The basic idea of the Architecture Guide is to give technology recommendations for each layer. The architect is then in charge of creating a specification which is subsequently handed over to the developer.
6 Appendix

6.1 Short Reference Guide

The short reference guide lists all forms that have to be filled out by the product manager / business expert in order to specify a composite application. Additionally a reference to the detailed description of each form within this guide will be given.

6.1.1 Form: General Application Information

Form (Template): General Application Information
(Executive Summary)

<table>
<thead>
<tr>
<th>Describe the business problem the new solution addresses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Missing process or deficiencies in the existing process</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Describe the idea behind the solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>How do you propose improving the situation described above?</td>
</tr>
<tr>
<td>Goal: the external reader must be able to grasp the fundamental idea behind the solution</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Describe the market and the value of the solution (attach business case)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Values such as: improve efficiency, improve flexibility, cost related values (reduce costs, better cost control), reduce error rate, increase throughput, increase accuracy, increase customer satisfaction</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Backend integration: solution should support several releases of the backend systems accessed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes (if yes add information about systems supported)</td>
</tr>
<tr>
<td>No</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Basic numbers (e.g. # users; # business objects; access frequency)</th>
</tr>
</thead>
</table>

Figure 32
Details: Chapter 4.1, General Application Information, page 15
### 6.1.2 Form: Process Overview

#### Form (Template): Processes Overview

<table>
<thead>
<tr>
<th>Process title</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Enter the title of the process here&gt;</td>
<td>&lt;description comes here&gt;</td>
</tr>
</tbody>
</table>

![Figure 33](Image)

Details: Chapter 4.2, Processes Overview, page 18
6.1.3 Form: General Process Information

**Form (Template): General Process Information (to be filled out for each process)**

**Process title: <Enter title of process here>**

<table>
<thead>
<tr>
<th>Information</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>When and how is the process integrated/started (consider any pre-conditions that have to be fulfilled before the process can start)?</td>
<td>&lt;description comes here&gt;</td>
</tr>
<tr>
<td>Due Date handling</td>
<td>&lt;description comes here&gt;</td>
</tr>
<tr>
<td>Number of parallel instances of this process (≥ number of concurrent users)</td>
<td>&lt;enter number here&gt;</td>
</tr>
</tbody>
</table>

Figure 34
Details: Chapter 4.3, General Process Information, page 20
6.1.4 Form: Roles

**Form (Template): Roles (to be filled out for each process)**

<table>
<thead>
<tr>
<th>Role</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;enter a process role name here&gt;</td>
<td>&lt;enter description here&gt;</td>
</tr>
</tbody>
</table>

**Figure 35**
Details: Chapter 4.4, Roles, page 23
6.1.5 Form: Visualization of Process Flow

<table>
<thead>
<tr>
<th>Process Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Role 1</td>
</tr>
<tr>
<td>Role 2</td>
</tr>
<tr>
<td>Role 3</td>
</tr>
<tr>
<td>Role 4</td>
</tr>
<tr>
<td>Role 5</td>
</tr>
</tbody>
</table>

Figure 36
Details: Chapter 4.5, Visualization of Process Flow, page 25
### 6.1.6 Form: Step Data

**Form (Template): Step Data (to be filled out for each process)**

#### Steps for Process: <Enter title of process here>

<table>
<thead>
<tr>
<th>Step No</th>
<th>Step Title</th>
<th>Step Description</th>
<th>Interactive (I) or Background (B)</th>
<th>Mandatory</th>
<th>Needed Services (numbers)</th>
<th>Due date handling required</th>
<th>Role</th>
<th>Exits Next Step</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;# of step &gt;</td>
<td>&lt;enter step title here&gt;</td>
<td>&lt;enter description of business logic here&gt;</td>
<td>&lt;I or B&gt;</td>
<td>&lt;Y or N&gt;</td>
<td>&lt;reference number to a service from the service list&gt;</td>
<td>&lt;Y -if so define the role or N&gt;</td>
<td>&lt;enter responsible role here &gt;</td>
<td>&lt;list all possible exits of the step here and for each exit assign the number of the next step&gt;</td>
</tr>
</tbody>
</table>

---

**Figure 37**

Details: Chapter 4.6, Step Data, page 28
### 6.1.7 Form: Exception Handling

**Figure 38**
Details: Chapter 4.7, Exception Handling, page 31

#### Exception Handling

<table>
<thead>
<tr>
<th>Exception No</th>
<th>Exception</th>
<th># of step</th>
<th>Action</th>
<th>Continue/Retry/Terminate</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;enter # of exception here&gt;</td>
<td>&lt;enter type of business exception here&gt;</td>
<td>&lt;enter # of step here&gt;</td>
<td>&lt;describe appropriate reactions to the exception here; in addition refer to the UI (by the UI's name) that should be shown or to the process (by the process's name) that should be started if the exception occurs&gt;</td>
<td>&lt;C, R, or T&gt;</td>
</tr>
</tbody>
</table>

---

*The Best Run Businesses Run SAP*
### 6.1.8 Form: Business Objects Overview

#### Form (Template): Business Objects Overview

<table>
<thead>
<tr>
<th>Business Object’s Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;Enter name of business object here&gt;</td>
<td>&lt;description comes here&gt;</td>
</tr>
</tbody>
</table>

**Figure 39**

Details: Chapter 4.8, Business Objects Overview, page 34
6.1.9 Forms: Business Object

Form (Template): Business Object Data (to be filled out for every Business Object)

Business Object: <enter name of BO here>

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Length/Number Range</th>
<th>Mask - Pattern</th>
<th>New</th>
<th>Used in Step</th>
<th>Sample Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;field name&gt;</td>
<td>&lt;type&gt;</td>
<td>&lt;length/ range&gt;</td>
<td>&lt;pattern&gt;</td>
<td>&lt;Y or N or M&gt;</td>
<td></td>
<td>&lt;enter some sample data here&gt;</td>
</tr>
</tbody>
</table>

Relationships for Business Object: <enter name of BO here>

<table>
<thead>
<tr>
<th>To Business Object</th>
<th>Relationship Type (Composition / Association)</th>
<th>Cardinality</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;enter name of related BO here&gt;</td>
<td>&lt;Association or Composition&gt;</td>
<td>&lt;0, 1, 1, 0..n, 1..n&gt;</td>
</tr>
</tbody>
</table>
Form (Template): Business Object Data cont’d

Mapping Information for Business Object: <enter name of BO here> (only applicable if fields already exist in backend systems):

<table>
<thead>
<tr>
<th>Field</th>
<th>System/ SAP Component</th>
<th>System/ SAP-specific Business Object or Service</th>
<th>Field or Service Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;enter field name here&gt;</td>
<td>&lt;enter external system info here (system type, vendor, release, etc.)&gt;</td>
<td>&lt;enter remote BO name here&gt;</td>
<td>&lt;enter remote field here&gt;</td>
</tr>
</tbody>
</table>

Additional Information:

<table>
<thead>
<tr>
<th>Information</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>How many BO’s of this type do you expect for the composite?</td>
<td>&lt;description comes here&gt;</td>
</tr>
<tr>
<td>How will the BO be used in the composite (Create, Read, Update, Delete)?</td>
<td>&lt;description comes here&gt;</td>
</tr>
<tr>
<td>Will the BO survive the business process?</td>
<td>&lt;description comes here&gt;</td>
</tr>
<tr>
<td>If the BO exists in a backend system: can the BO be changed in the backend system while it is being used in the composite?</td>
<td>&lt;description comes here&gt;</td>
</tr>
</tbody>
</table>

Figure 41
Details: Chapter 4.9, Business Object, page 36
6.1.10 Form: UI Mockup

Create UI Mockup (e.g. PPT...) (to be created for every interactive step)

UI Mockup for step: <# of step> (<enter name of UI here>)
UI Mockup for step: 1 (Enter Complaint)

<table>
<thead>
<tr>
<th>Enter Complaint</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer Name:</td>
</tr>
<tr>
<td>Panorama Studios, P. Miller</td>
</tr>
<tr>
<td>Order Number:</td>
</tr>
<tr>
<td>5001658</td>
</tr>
<tr>
<td>Complaint Text:</td>
</tr>
<tr>
<td>15 Flat Future monitors do not work. Please replace them. P. Miller</td>
</tr>
<tr>
<td>Complaint Date:</td>
</tr>
<tr>
<td>07.08.2006</td>
</tr>
</tbody>
</table>

Submit

Figure 42
Details: Chapter 4.10, UI Mockup, page 42
6.1.11 Form: UI Description

Form (Template): UI data for mockup (to be filled out for every interactive step)

**UI Description for step: <# of step> (<enter name of UI here>)**

Online Form: <Yes/No>
Offline Form: <Yes/No>

<table>
<thead>
<tr>
<th>Business Object</th>
<th>Field</th>
<th>Table &lt;Y/N&gt;</th>
<th>Read / Modify</th>
<th>Additional Info</th>
<th>Mandatory &lt;Y/N&gt;</th>
<th>Value Help (if yes, which service?)</th>
<th>Field Validation (only applicable for input fields)</th>
<th>Sample Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;enter BO name here&gt;</td>
<td>&lt;enter field name here&gt;</td>
<td>&lt;Y or N&gt;</td>
<td>&lt;R or M&gt;</td>
<td>&lt;R or M&gt; if &quot;R&quot; is chosen specify the source of the data (typically BO or service)</td>
<td>&lt;Y or N&gt;</td>
<td>&lt;Y or N&gt;</td>
<td>&lt;Y with number of service that delivers the data for the value help or N&gt;</td>
<td>&lt;empty or field validation rule that has to be applied or number of service that fulfills the validation&gt;</td>
</tr>
</tbody>
</table>

Figure 43
Details: Chapter 4.11, UI Description, page 45
### 6.1.12 Form: Services

#### Form (Template): Service (to be filled out for every service)

**Service Number:** <# of service>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service Name</td>
<td>&lt;enter service name here&gt;</td>
</tr>
<tr>
<td>Input Parameters</td>
<td>&lt;list all input parameters here; each parameter name must follow the pattern „BO_Name.Bo_Field“, e.g. Customer.Firstname&gt;</td>
</tr>
<tr>
<td>Output Parameters</td>
<td>&lt;list all output parameters here; each parameter name must follow the pattern „BO_Name.Bo_Field“, e.g. Customer.Firstname&gt;</td>
</tr>
<tr>
<td>Exceptions</td>
<td>&lt;list numbers of all business exceptions here (must be listed on the „exception handling“ form)&gt;</td>
</tr>
<tr>
<td>Business Logic (Prosa)</td>
<td>&lt;describe what happens during the service call&gt;</td>
</tr>
<tr>
<td>Relying on which Services</td>
<td>&lt;list all service numbers this service needs to fulfill its functionality&gt;</td>
</tr>
<tr>
<td>Mapped Against External Service</td>
<td>&lt;if available, name the external system’s service name&gt;</td>
</tr>
<tr>
<td>External Service Running On (System Information, Release etc.)</td>
<td>&lt;if available specify details about the system the external service is running on&gt;</td>
</tr>
</tbody>
</table>

**Figure 44**

Details: Chapter 4.12, Services, page 48
6.2 Interactive Forms

What is it all about?
Interactive Forms based on Adobe software (or Interactive Forms) is SAP's forms technology, which is based on Adobe technology components integrated into SAP NetWeaver Application Server. The solution is therefore available to any application based on SAP NetWeaver.

Interactive Forms allows you to integrate PDF forms generated by your SAP system into your business processes. Such forms can be used for interactive data capture, i.e. they can contain fields - such as text fields, check boxes, or radio buttons - that can be edited by a user in the free Adobe Reader. After filling in the form, the data entered by the user is automatically submitted back to the system, where the process continues.

Interactive Forms is not intended to be an alternative UI to the SAP backend, but there are suitable use cases for including such a form into a Guided Procedure process.

How can I use Interactive Forms?
In general, Interactive Forms can be used in three different scenarios:

- **Online interactive scenario:**
  While filling in the PDF form, the user maintains an active connection to a backend. This allows for, among others, the provision of context-sensitive value help, and immediate entry validation, i.e. all the tried and tested features of SAP software. In an online interactive scenario with Guided Procedures, the PDF form is integrated into a Web Dynpro application (for Java or ABAP).

- **Offline interactive scenario:**
  In this case, the user does not need access to the backend system. He/she completes the PDF form, which can contain prefilled data from the backend, in the stand-alone Adobe Reader, and submits the form (and data entered therein) to backend by e-mail or through an upload. Given the lack of accessible backend logic, this scenario offers fewer help and check options than the online one, but allows for, for example, static value help, static (local) checks inside the form, and simple arithmetic calculations.
  **Note:** Support for offline processes is the standard way in which Guided Procedures integrates Interactive Forms.

- **Form printing scenario:**
  This classic forms processing scenario is not interactive, and serves to generate output documents that are printed, e-mailed, archived or faxed. PDF forms can be used for frontend printing on the client, or high-volume backend printing purposes.

When Should I Use an Interactive Form?
It may make sense to include a PDF form in your Guided Procedures application, if any of these criteria applies:

- **Need for offline processing:** You want to include people in your process who provide system-relevant data, but have no system access, for example, external suppliers. Prefill a PDF form with backend information, e-mail the form to the external contact who uses the stand-alone Adobe Reader to fill it in, and get back the data you require for your processes.

- **Consistency in presentation of online/offline processing:** Use the same form for both scenarios to ensure that the end user always sees the same information in the same position on the screen. Different interfaces for the same task confuse end users.

- **Maintain familiar look and feel of paper forms:** Casual system users in particular may not warm up to standard user interfaces for the interaction with the system. If you offer them the form they have known for years from the paper-based world on screen, their acceptance of system-based processes will grow.

- **PDF import and layouting capabilities for business end user:** Let each expert focus on their speciality: Programmers take care of the code, business experts want to design their form interface...
themselves. Adobe LiveCycle Designer allows them to do this through easy-to-use drag & drop features.

- WYSIWIG printing of filled forms: We may be able to transfer the vast majority of our processes into a system, but human beings still like to have paper copies of their work as a statement of record. Being able to print out a filled interactive form is a key requirement in many scenarios. Only the PDF format guarantees an exact print-out.

- Possibility to save local electronic copy of filled form: Keep a copy of the filled form in an offline scenario, or save an electronic copy of an online form. PDF makes it possible.

- Digital signatures and certification of PDF forms: To ensure the authenticity of the form, its content, and the sender, use end-user digital signatures and certification (server-side signatures) with PDF forms.