This reading sample is Chapter 5, “Introduction to OData Service Creation.” It gives an overview of both service development and service generation, and is the basis for the detailed instructions offered in Chapter 6 and Chapter 7.

"Introduction to OData Service Creation"

Contents

Index

The Authors

Bönner, Drees, Fischer, Heinz, and Strothmann

OData and SAP NetWeaver Gateway

672 Pages, 2014, $69.95/€69.95

www.sap-press.com/3431
This chapter explains the end-to-end cycle and the specific tools for creating SAP NetWeaver Gateway services, both for service development and for service generation.

5 Introduction to OData Service Creation

As you'll recall from Chapter 2, OData services are what implement the OData protocol and expose an endpoint that allows access to data. The number of OData services shipped with SAP NetWeaver Gateway is limited and will likely remain rather low because, by nature, OData services are granular and mostly tailored to individual use cases. More commonly, services are shipped as part of products such as SAP Fiori or SAP mobile solutions. A large amount of development time can go into building the right OData service, so understanding this process is essential.

The central interface that is used to define and implement services within SAP NetWeaver Gateway is the Service Builder (Transaction SEGW). After you've created a service in the Service Builder, it can be used directly in any interface. The Service Builder is a one-stop shop with respect to SAP NetWeaver Gateway service development and is supplemented by additional support tools. In certain cases, it even allows you to perform selected steps in third-party tools and then import the results (e.g., usage of an OData modeler for the model definition).

The main objective of this chapter is to give you an overview of the process of service creation, which we then discuss in more detail in Chapter 6 and Chapter 7. To achieve this, in Section 5.1, we give you a brief overview of the steps in the process for both types of service creation (service development and service generation). In Section 5.2, we look at the main tool involved in service creation: the SAP NetWeaver Gateway
Service Builder. We then complement this first look at the Service Builder with a quick look at some of SAP NetWeaver Gateway’s other tools that support service creation and maintenance. This section will give you an idea of the tools that are available to assist with tasks during the service creation process. In Section 5.3, we then dig deeper into service creation and look in more detail at the three main steps in service creation: data model definition, service implementation, and service maintenance. Also, we look at additional service creation-related topics such as service redefinition and the reuse of existing SAP NetWeaver Gateway services in mashups to create OData services. Finally, we give you an introduction to the development paradigm used for service development: the OData channel (Section 5.4).

5.1 Service Creation Process Overview

In this section, we introduce you to the general steps in service creation. This explanation of the service creation process is somewhat simplified in an effort to explain it with distinct and sequential steps (a waterfall approach). In reality, some of the steps can also be performed out of order (an incremental approach). We’ll go into a bit more detail about this at the end of this section, after presenting the simplified process.

There are two ways of creating OData services with SAP NetWeaver Gateway:

- **Service development**
  The classic option is the code-based development of SAP NetWeaver Gateway services. This ABAP-based option is extremely flexible and allows you to develop highly efficient and specialized services, but it also requires some significant technical know-how.

- **Service generation**
  The second way is the generation of SAP NetWeaver Gateway services. There are three main methods of service generation:
  - RFC generation: Allows you to generate a service using a tool called the RFC/BOR Generator.
  - Redefinition: Allows you to define a service based on an existing data source or an existing SAP NetWeaver Gateway service.
  - Model composition: Allows for mashing up multiple existing services. The result is a new service that can be created without the need to change the existing services.

Of these two approaches, service generation is the quicker approach and requires a lot less effort. On the other hand, it’s more limited, and thus is primarily recommended for developing very straightforward services. Service generation doesn’t give you much optimization potential because, without custom coding, you are restricted to what the service generators offer. In most real-world situations, you’ll want to opt for service development because the advantages are well worth the effort. Still, if you have a GenIL or Service Provider Interface (SPI) objects, analytical queries such as SAP NetWeaver BW Easy Queries, or a suitable RFC function module or Business Application Programming Interface (BAPI) and are aiming for a quick result, this might be an option for you. (We’ll go into more detail about these specific options in Chapter 7, where we discuss service generation in detail.)

Whether you’re using service development or service generation, you create an OData service by following the SAP NetWeaver Gateway service creation process. This process consists of three main phases: model definition, service implementation, and service maintenance. Depending on whether you go for development or generation, the individual phases of the service creation process can have different flavors. These flavors result in different tracks you take during the actual process.

Before you can start with this process, you have to complete the process of service definition as a prerequisite. This is the process of identifying what service to create and specifying its details. Ideally, you’ve done all of this together with the client developers so that you know exactly what data they require and how this works with the artifacts in the SAP Business Suite that will be the basis for your SAP NetWeaver Gateway service. After you have the service definition, you can start with the three development phases of the service creation process.

During the starting phase, data model definition, you define the model your service is based on. That is, you define the required artifacts such as entity types, entity sets, associations, and other components that your service will use (refer back to Chapter 2 for explanations of these.
components). After data model definition, you must generate the repository objects and register them in the SAP Business Suite system so that you can proceed with the next main phase, which is the service implementation.

In the service implementation phase, the operations that are supported by the service are implemented. Here the different tracks for service development and service generation come into play:

- For service development, operations that are supported by the service are implemented using ABAP coding.
- For service generation, there are three paths depending on the type of generation chosen:
  - If you use RFC/BOR generation, service implementation takes place by mapping the OData model to the methods of the RFC function module or a Business Object Repository (BOR) object.
  - If you use redefinition, there is no service implementation step. You only have to perform the model definition step, because the implementation of the service is generated based on the customizing that has been performed in the model definition step.
  - If you use model composition, there is again no service implementation step. Instead, you include one or more existing services into a new model.

The third phase of the service creation process, service maintenance, publishes the service so that it becomes visible in the service catalog of the SAP NetWeaver Gateway system. In effect, this means that the created OData service can then be consumed.

Although we clearly delineate the two methods of service creation (service generation and service development), it’s actually possible to mix these in a way that suits you best. For example, you can create an OData service where one entity set is implemented using the RFC/BOR Generator (service generation), while a second entity set is implemented using code-based implementation (service development).

As we mentioned before, we’ve presented the service creation process in a very structured and clearly sequential way. This waterfall approach allowed you to easily understand what the different phases are for. In real-world projects, after you’ve understood how it works, you can adjust the sequence to what fits you best (within certain boundaries). The one exception to this rule is the service maintenance phase—this is almost always a one-time activity. As soon as a service is registered and activated (published), you don’t have to touch these settings anymore, even if the implementation and/or model definition changes.
The service publication is a one-time activity as long as you don’t perform major changes. Registering the service for additional SAP Business Suite systems, for example, is such an activity in which you would have to go back to the service maintenance phase. Again, though, changes in the implementation of an already published service or in the data model can be used in the already published service without any further activities.

For all other phases, you’ll typically always follow an incremental approach: you build a service—or part of it—execute and test it, and then go back and refine that same service until it fits all of your needs. During the creation of an OData service, you may change the model and/or the service implementation multiple times.

Furthermore, an approach often used in real-world projects is to perform the service implementation and the service maintenance in a different order. Performing the service maintenance with a service implementation stub before the actual service implementation allows you to browse metadata (service document and service metadata document), even if the service itself doesn’t yet have any functionality. You’ve basically started with a service stub and can then fill this stub in an incremental way.

Figure 5.2 depicts the incremental service creation process. It’s based on Figure 5.1 and adds incremental steps to the original process. These incremental steps are displayed by the solid line arrows that depict potential transitions among the three phases of data model definition, service implementation, and service maintenance. These phases are symbolized by the horizontal boxes. The dotted line stands for the one-time activity of service publication as part of the service maintenance phase.

5.2 SAP NetWeaver Gateway Toolset

SAP NetWeaver Gateway provides a set of tools to address all needs from development to testing to operations. For now, we’ll skip tools targeted at operating SAP NetWeaver Gateway and focus specifically on service creation-related tools. In this section, we’ll take a look at SAP NetWeaver Gateway Service Builder—the central, one-stop development tool for SAP NetWeaver Gateway services—and the additional, well-integrated tools that support you during the SAP NetWeaver Gateway service creation process.
5.2.1 SAP NetWeaver Gateway Service Builder

The Service Builder contains all relevant functions for modeling and development of OData services in SAP NetWeaver Gateway. This includes both code-based development of services and the generation of OData services. Also, it provides direct access to additional development-related functions such as service registration/activation and service validation. The Service Builder supports the entire development lifecycle of an OData service in SAP NetWeaver Gateway, and you can start it using Transaction SEGW (Figure 5.3).

Overall, the Service Builder addresses the needs of both experienced and less experienced developers, as well as nondevelopers. Whereas experienced developers can develop their own source code with maximum flexibility in their service implementation, they still can use the built-in OData modeler and other tools to simplify the development process. Less experienced developers will appreciate the possibility to use tools that allow generating OData services without having to write a single line of code.

The SAP NetWeaver Gateway Service Builder allows for centrally displaying and creating the definition of an OData service. This includes runtime artifacts (model provider class [MPC], data provider class [DPC], model, and service), OData artifacts (entity set, entity type, and properties), as well as used data sources and models.

The modeling environment follows a project-based approach, and all relevant data is consolidated in these projects. Development using the Service Builder is therefore organized in projects, and creating a project is the starting point of every service development using the Service Builder. Projects are used to bundle all artifacts that are needed for service development in one central place, thereby providing a means to organize the development process. The Service Builder allows the developer to open several projects at the same time as shown in Figure 5.4 (in this example, ZGWSAMPLE and ZPRODUCT).

---

Project-based development

---

Note

From a technical system perspective, the Service Builder is used in a system where the Business Enablement Provisioning (BEP) component is installed,
which is typically an SAP Business Suite system (refer to Chapter 4 for a discussion of the different deployment options for SAP NetWeaver Gateway). The BEP component is delivered as add-on IW_BEP until SAP NetWeaver release 7.31. As of SAP NetWeaver release 7.40 SP02, the BEP component is included in SAP NetWeaver itself as part of component SAP_GWFND. As a result, it’s possible to perform development of OData services using the Service Builder without additional effort in all systems after they run on top of SAP NetWeaver 7.40 SP02 or later.

Because the Service Builder is part of the BEP component that is typically (but not necessarily) installed on the SAP Business Suite system, you define the service model (model provider class, MPC) as well as the service logic (data provider class, DPC) on the same system where the BEP component is deployed. This is important to understand if it comes to referencing other ABAP Repository objects such as Data Dictionary (DDIC) elements (e.g., structures or data elements) that are required when calling, for example, an RFC or BAPI.

The objective of the Service Builder is to provide comprehensive support for building OData services in a declarative way or by reusing existing business objects in the SAP Business Suite system. However, there are restrictions in what can be declared or generated. Advanced OData features may need to be implemented manually, and certain operations aren’t available in a refined business object. The result of what you do in the Service Builder will always be ABAP classes, which are based on the OData channel programming model of SAP NetWeaver Gateway (covered in Section 5.4). You can always drill down to understand what is going on during service execution or tweak the code.

5.2.2 Beyond the Service Builder: Supporting Tools during the Service Creation Process

As stated, the main tool during the service creation process is the SAP NetWeaver Gateway Service Builder. At the same time, SAP NetWeaver Gateway provides a set of additional tools that are very useful during the development of SAP NetWeaver Gateway services. These tools allow, for example, for early testing of services or tracing what is happening when calling a service. As such, this section aims to briefly introduce you to some of the functionalities. For a more comprehensive description of the development support and administration toolset of SAP NetWeaver Gateway, see Chapter 13.

Integrated Test Environment

The SAP NetWeaver Gateway client can be used for both testing and troubleshooting and is a REST client built into SAP NetWeaver Gateway. It can be started from within SAPGUI using Transaction /IWFWND/GW_CLIENT. After you’ve created a service, you can use this tool for a first test, as shown in Figure 5.5.

First select an HTTP method such as GET, POST, DELETE, HEAD, or OPTIONS, as shown in 1. Then enter the URI of your request into the Request URI input field, as shown in 2. It’s also possible to set a certain HTTP header if needed. The body of an HTTP request can be entered either manually or uploaded from a file, as shown in 3. In addition, it’s possible to use the REQUEST function to create, for example, an update request based on
the response (shown in 3) of a read request that has been issued against the URI before. Finally, perform the HTTP request by choosing EXECUTE, as shown in 1.

**Test cases**

A very useful feature of the SAP NetWeaver Gateway client is that test cases can be stored in a database. The test case shown in Figure 5.5 is one of more than 70 sample test cases that are delivered in test group CORE_SAMPLES for the standard test services TEA_TEST_APPLICATION and RMTESTSAMPLEFLIGHT. Note that the test cases of the CORE_SAMPLES test group have to be manually created from within the SAP NetWeaver Gateway client as shown in Figure 5.6.

![Create Core Samples from within the SAP NetWeaver Gateway Client](image)

**Figure 5.6 Create Core Samples from within the SAP NetWeaver Gateway Client**

If you've saved a request as a test case, you can afterwards add or change the expected HTTP return code. A request can return multiple HTTP return codes that are valid (for example, 200, 401, 402, and 403). Therefore multiple statuses, including status ranges separated by a space, can be entered (for example: 201 401-403).

One or more test cases can then be run using the SAP NetWeaver Gateway client. The results are displayed in a table indicated by a traffic light icon together with the expected and actual HTTP return code.

**Error Log**

The error log is the second tool the developer will find very useful when it comes to troubleshooting. The error log can be called using Transaction /IW_FND/ERROR_LOG in the SAP Business Suite system.

The error log is tightly integrated with the SAP NetWeaver Gateway client, so it's possible to rerun a request sent by a consumer that led to errors. Do this by selecting REPLAY • GATEWAY CLIENT as shown in Figure 5.7.

![Transaction /IW_FND/ERROR_LOG](image)

**Figure 5.7 Transaction /IW_FND/ERROR_LOG**

As another way to dig into potential problems, monitoring log entries can be generated for the system log and the application log of SAP NetWeaver Gateway. The system logs can be accessed using Transaction SM21. For the application log, use Transaction /IW_FND/APPS_LOG.

**Catalog Service**

Each SAP NetWeaver Gateway system provides a CATALOG service that can be used to retrieve a list of all available services on SAP NetWeaver Gateway (Figure 5.8). The CATALOG service is an OData service, and the list of available services can be accessed via the following URL:

http://<server>/<ports>/sap/opu/odata/iwfnd/CATALOGSERVICE/Catalog Collection
OpenSearch

The Catalog service supports OpenSearch. Developers or development tools are thus able to use a free text search allowing them to find services based on the service description that can be retrieved using the following URL:

http://<server>:<port>/sap/opu/odata/iwfnd/CATALOGSERVICE/ServiceCollection/OpenSearchDescription.xml

5.3 Steps in the Service Creation Process

In the opening parts of this chapter, we've introduced the SAP NetWeaver Gateway service creation process. To recap, this process consists of three phases: data model definition, service implementation, and service maintenance. You can take different tracks for creating your services depending on whether you go for service development or service generation. Now let's take a closer, more technical look at the different tracks and the individual steps in these tracks. Due to the various options for creating SAP NetWeaver Gateway services, you'll find it useful to refer back to Figure 5.1 throughout this section.

5.3.1 Data Model Definition in the Service Builder

The first phase of the service creation process is the data model definition phase. The goal of this phase is to use the Service Builder to create a data model that contains all information about the OData model of a service, such as entity types, complex types, properties, and associations. So, when developing an SAP NetWeaver Gateway service (service development) or when generating an SAP NetWeaver Gateway service using the RFC/BOR Generator (one specific type of service generation), the first main process step is to create a data model.

Note

When using the second method of service generation, which is to redefine an existing service, the data model isn't defined but rather redefined based on the existing business objects. For information about that kind of data model building, see Section 5.3.5.

There are several ways of defining a data model with the Service Builder, each of which addresses a specific use case.

The first option is the manual creation of the various components of an OData model, which is called a declarative model definition. Entity types, associations, and association sets in this approach are created manually.

The second option is the import of data models in the EDMX format that have either been defined by the OData Model Editor of the SAP NetWeaver Gateway Productivity Accelerator (GWPA) or the entity data modeler provided by Microsoft Visual Studio. In addition, it's possible to import the service metadata document of an existing OData service.

The third option is to create entity types by reusing data models that already exist in the SAP Business Suite System. This can be done by the import of DDIC structures/tables or, alternatively, by the generation of new entity types based on an RFC/BOR interface.

Next, we'll discuss all four options in a bit more detail.

Declarative Data Model

A declarative data model is created manually using the Service Builder. This method is mainly used to create entity types based on manually created properties, which can be based on existing DDIC types. (To model
Introduction to OData Service Creation

an OData service from scratch in WYSIWYG style, alternative OData modeling tools, such as SAP NetWeaver Gateway Productivity Accelerator [see Chapter 8] and Microsoft Visual Studio, are better. However, in these cases, the model has to then be imported into the Service Builder.

Import Data Model via EDMX

Using the import model option, the developer can import a complete OData model stored in an EDMX file, or a metadata document of an existing OData service, into the Service Builder. This includes the definition of entity types, entity sets, associations, and other components. You can import data model files that have been defined by graphical OData modeling tools, or service metadata files of an existing OData service.

Note
If you perform an import of a service metadata document or an EDMX file for an existing project into the Service Builder prior to SAP NetWeaver Gateway 2.0 SP07, the existing data model will be overwritten. With SP07, SAP provides the possibility to reimport data model files.

Import Data Model via DDIC

DDIC type support
To reduce the time required to create entity types and complex types in your data model and to leverage existing data structures in your SAP Business Suite system, you can import the following DDIC types into the Service Builder:
- Views
- Database tables
- Structures

Beautification
When creating an entity type from a DDIC type, the name of the entity type and the names of the properties of the entity type suggested by the Service Builder are derived from the original names of the DDIC type and its fields by removing the underscores and generating a name with camel case notation instead. For example, when using a structure such as `BAPI_EPM_PRODUCT_HEADER`, the Service Builder will propose the name `BapiEpmProductHeader`

The name of the entity set and its properties should be easy to understand. This is because it’s the entity set and the names of its properties that are visible to the consumer, and the names of the properties of an entity set are derived from the property names of the underlying entity type.

During the process of importing a DDIC structure or even afterward, the developer can start a process called beautification. Through this process, it’s possible to reduce the number of properties of an entity type by simply removing single properties from it. In addition, it’s possible to maintain the names of the properties of an entity type.

Reducing the number of properties to those that are absolutely necessary and maintaining the names that are visible to the outside world are important for creating services that are easy to consume. Publishing existing DDIC structures as is to the outside world is usually not very beneficial.

Beautification is discussed in more detail in Chapter 7, Section 7.2.1.

Import Data Model via RFC/BOR

Finally, the Service Builder also enables you to create entity types from function module parameters and BAPI parameters. A wizard is provided to guide you through the process. Using the interface of an RFC function module or a BOR interface is beneficial if they are being used to access the data in the SAP Business Suite system. Both code-based implementation and using the RFC/BOR Generator are possible with this approach.

5.3.2 Service Registration in the SAP Business Suite System

After the data model is defined, it must then be registered. Service registration in the SAP Business Suite manifests the data model definition phase’s results. This means that the runtime objects required for an SAP NetWeaver Gateway service are generated using the Service Builder. For the convenience of the developer, the Service Builder also performs the necessary tasks to register the service in the SAP Business Suite.
Service Registration versus Service Maintenance

As you may recall from Section 5.1, the service maintenance phase of service creation involves activating and registering the service on the SAP NetWeaver Gateway server. This isn’t to be confused with service registration in the SAP Business Suite system, which is a process that occurs after the data model definition. In this section, we’re focusing on service registration in the SAP Business Suite system. In Section 5.3.4, we’ll discuss service maintenance.

The difference between service registration and service maintenance is as follows:

- **Service registration** is an activity during service development that results in the creation of artifacts needed for development.
- **Service maintenance** is an activity during the deployment/operation of an SAP NetWeaver Gateway service. It activates the service for consumption.

**Stub class creation**

Based on the data model that has been created, the Service Builder generates a corresponding MPC and DPC, as well as extension classes. The MPC contains the coding that programatically declares the data model being used by your service. The implementation of the service operations is performed in the DPC. The extension classes that have been generated by the Service Builder can be used to redefine methods of the generated base classes by custom code because the base classes are always regenerated when the model has been changed. (For more information on MPC and DPC, see Section 5.4.)

**Service registration**

To be used as a service, some configuration steps have to be performed; these steps are supported by the Service Builder (Figure 5.9). When generating a project for the first time, the developer has to specify the names of the MPC and its extension class and the DPC and its extension class. In addition, the developer has to specify the TECHNICAL MODEL NAME and the TECHNICAL SERVICE NAME. The latter becomes the external service name that is later used for publishing the service on the SAP NetWeaver Gateway.
5.3.3 Service Implementation

During the service implementation phase of the service creation process, operations that are to be supported by the SAP NetWeaver Gateway services are implemented via ABAP code or via mapping of the methods of an RFC function module or BAPI on the properties of an OData model. Operations are executed on the defined data model during runtime and encompass CREATE, READ, UPDATE, DELETE, and QUERY methods (CRUD-Q methods).

It’s important to note that the service implementation phase applies only to service development and to one of the service generation options: RFC/BOR generation. For service generation using redefinition, the service implementation step isn’t necessary. This is because the implementation of the service will be generated based on the customizing that has been performed in the model definition step.

Note
We provide an introduction to service generation using redefinition in Section 5.3.5.

Next, we’ll give you a brief overview of the service implementation phase for both scenarios where the phase is relevant: service development and service generation via mapping RFC/BOR interfaces.

Implementation for Service Development

Remember that during the service registration of the data model definition phase, a data provider extension class was created. Also during the service implementation phase, operations that are to be supported by the SAP NetWeaver Gateway services are being implemented.

To implement the supported SAP NetWeaver Gateway services using ABAP coding, you have to manually redefine the respective methods of the data provider extension class, which should remind you of the CRUD-Q operations:

- `<ENTITY_SET_NAME>_CREATE_ENTITY`
- `<ENTITY_SET_NAME>_UPDATE_ENTITY`
- `<ENTITY_SET_NAME>_DELETE_ENTITY`
- `<ENTITY_SET_NAME>_GET_ENTITYSET`

Access to these methods is offered in a very convenient way by the Service Builder. This takes place by expanding the service implementation node as depicted in Figure 5.11. From there, you can navigate to the respective entry of an entity set, expanding all CRUD-Q methods of an entity set. Selecting Go to ABAP Workbench allows the developer to switch seamlessly to the class builder (Transaction SE24) to implement an operation.

![Figure 5.11 Code-Based Implementation](image)

In addition, it might be necessary to redefine additional methods in the data provider extension class that aren’t specific to an entity set such as the CRUD-Q methods mentioned earlier (if, for example, deep insert should be supported by the OData service).

Implementation for Mapping RFC/BOR interfaces

The process of implementation for mapping is different from that of service development. To start the mapping process, you have to select Map to DATASOURCE in the context menu of a CRUD-Q method of an entity.
set in the Service Implementation folder (Figure 5.12). The built-in mapping tool of the Service Builder then allows defining relations between the interface parameters of a function module or BAPI and the properties of an entity set.

**Figure 5.12  Mapping the Methods of an Entity Set to a Data Source**

**CRUD-Q** You can map the CREATE, READ, UPDATE, DELETE, and QUERY (CRUD-Q) methods of each entity set separately. The actual service implementation, that is, the coding in the CRUD-Q methods mentioned earlier, will be generated by the Service Builder based on the mapping you’ve performed. The Service Builder supports the developer by providing mapping proposals if the entity type has been created by importing a BOR or RFC interface. For example, as shown in Figure 5.13, the Service Builder suggested a mapping between the property **SoId** in the entity set **SalesOrderSet** and the property **SO_ID** of the export parameter **SOHEADERDATA** of the **BAPI BAPI_EPM_SO_GET_LIST**. This mapping can automatically be suggested because the entity type on which the entity set **SalesOrderSet** is based has been created by importing the interface parameter **SOHEADERDATA**.

If additional methods for the entity sets are mapped, the Service Builder checks the already existing mappings and derives proposals for them. If you, for example, started to map the Query operation (GET_ENTITYSET) of your entity set and now want to map the Read operation (GET_ENTITY), the Service Builder is able to provide a proposal for those properties that have already been mapped in the GET_ENTITYSET method.

**Figure 5.13  Mapping Proposals**

### 5.3.4 Service Maintenance

The service maintenance phase primarily consists of the service activation and service registration step in the SAP NetWeaver Gateway system. For SAP NetWeaver Gateway to consume a service using an OData client, this service has to be activated. This activation takes place in the SAP NetWeaver Gateway server and makes the service ready for consumption.

The registration and activation of services in the hub is performed using Transaction /IWFND/MAINT_SERVICE (Activate and Maintain Service). Transaction /IWFND/MAINT_SERVICE is also used to maintain all activated services on the SAP NetWeaver Gateway server. Services have to be changed if they have been registered in several/additional connected SAP Business Suite systems, or they can simply be deactivated.

Because the Service Builder is the one-stop-shop for service development, functionality has been added to the Service Builder that allows the developer to directly call the transaction for service maintenance from within the Service Builder. This is even possible for remote systems.

The developer can either select the list of SAP NetWeaver Gateway systems in the SERVICE MAINTENANCE node (Figure 5.14) or can click on the **Register** button.
5.3.5 Service Generation via Redefinition

As explained in Section 5.1, redefinition is the process of generating a service based on an existing data source. This is done using a wizard and combines both the data model definition phase and the service implementation phase into the single phase of **redefinition**. The resulting generated service has to be registered and activated in the SAP NetWeaver Gateway server system (the service maintenance phase), and can then be consumed. The goal of redefinition is to allow for service creation with less effort.

**Figure 5.14** Registering a Service in the Hub from the SAP Business Suite

**Steps in the Service Creation Process**

1. Select the business object.
2. Select artifacts of the data source (data model definition).
3. Generate runtime artifacts and service registration in the backend (service implementation).

In other words, the wizard starts with the data model definition part but automatically performs the steps that belong to the service implementation phase. After the service has been registered and implemented in the SAP Business Suite, it has to be activated in the SAP NetWeaver Gateway server.

The different integration scenarios described in this section are partly based on specific add-ons listed in Table 5.1. If these add-ons have been deployed to the SAP Business Suite system, the related context menu options in the Service Builder are visible as shown in Figure 5.15.

Most of the scenarios are also remote-enabled, which means that the business object that is to be consumed (for example, an SPI object) doesn’t have to exist in the same system in which the BEP component is deployed. As a result, these scenarios can be implemented in the SAP NetWeaver Gateway server (assuming you are using hub deployment with development on the hub).

### Table 5.1 Add-Ons for Generating a Service Based on an Existing Data Source

<table>
<thead>
<tr>
<th>Name of Add-On</th>
<th>Integration Scenario</th>
<th>Remote-Enabled</th>
</tr>
</thead>
<tbody>
<tr>
<td>IW_GIL</td>
<td>Generic Interaction Layer (GenIL)</td>
<td></td>
</tr>
<tr>
<td>IW_SPI</td>
<td>Service Provider Interface (SPI)</td>
<td>X</td>
</tr>
<tr>
<td>IW_BEP</td>
<td>Analytical Queries</td>
<td>X</td>
</tr>
<tr>
<td>IW_HDB</td>
<td>SAP HANA</td>
<td>X</td>
</tr>
<tr>
<td>IW_BEP and IW_FND</td>
<td>OData Service (External)</td>
<td>X</td>
</tr>
<tr>
<td>IW_BEP</td>
<td>OData Service (SAP NetWeaver Gateway)</td>
<td>X</td>
</tr>
</tbody>
</table>

**Note**

On top of integrating existing SAP Business Suite business objects, it’s also possible to integrate third-party OData services. This integration scenario is explained in Section 7.2.3 of Chapter 7, where the generation of services is explained in more technical detail.
Next, let’s look at the different possible sources for suitable business objects in detail.

**Generic Interaction Layer (GenIL)**

Integration of GenIL with SAP NetWeaver Gateway offers the possibility of generating OData services based on existing GenIL components. GenIL is meant to be a wrapper around existing business logic. It provides access to all business objects via a unified interface for consuming application logic in the UI layer by using the BOL API. The BOL consists of two pieces:

- **GenIL**
  - The lower layer is a “dispatcher” that manages GenIL components and their models at runtime and distributes requests from above to the respective components implementing the requested objects.

- **BOL**
  - The stateful layer provides optimized performance by avoiding expensive repetitive access to the APIs and thus acts as a buffer for the UI.

While BOL was built for SAP CRM Web Client, the role of GenIL is different because it can be used for other integration scenarios as well. The consumption of SOAP-based web services using the Web Service tool that directly consumes GenIL is an example of such additional integration.

Similarly, SAP NetWeaver Gateway also allows you to generate OData services leveraging GenIL (as shown in Figure 5.16). The nodes, relations, and queries in the GenIL model are transformed to the corresponding entities in an OData model, as shown in Figure 5.17.

![Diagram](image)

**Figure 5.15** Context Menu Options to Create a Data Model Using Redefinition

**Figure 5.16** Integration of GenIL with SAP NetWeaver Gateway

Although BOL (and thus GenIL) are frequently used for SAP CRM Web Client, it has also been used in other SAP Business Suite applications such as SAP ERP Financials and SAP ERP Human Capital Management (HCM). The integration is contained in the IW_GIL add-on. This must be deployed locally on the SAP Business Suite System (for example, SAP CRM) on top of the BEP component.

**Note**

The GenIL integration scenario isn’t remote enabled. To use services that are generated based on GenIL objects, the add-on IW_BEP component (SAP_GWFND starting from SAP NetWeaver release 7.40) has to be deployed on the SAP Business Suite system.
Introduction to OData Service Creation

Service Provider Interface (SPI)
The Service Provider Interface (SPI) was originally developed for SAP PLM. SPI is a framework generated within the application layer that has different consumers. The framework is currently not only used by the applications for which it was originally developed, but also for various other applications within the SAP Business Suite.

SPI objects can be called remotely. As a result, it isn’t mandatory to deploy the SAP NetWeaver Gateway IW_SPI add-on for SPI on the SAP Business Suite system. Because the add-on calls the RFC interface of the SPI layer, it can be deployed on the SAP NetWeaver Gateway server system. The IW_GIL add-on instead must be deployed locally on the SAP Business Suite system (for example, SAP CRM). The integration of SPI with SAP NetWeaver Gateway allows SPI application building blocks to be provisioned as OData services.

Further Resources
For more information about this topic, we recommend the following:
- SPI Wiki on SCN: http://wiki.sdn.sap.com/wiki/display/SP/SP/Home
- SAP Online Help: http://help.sap.com/saphelp_crm70/helpdata/en/7c/0f77e9f297402aacb48ca7110c7f2a/frameset.htm

Analytic Queries
Analytic queries are the main tools for consuming analytical data that is embedded in business applications such as the SAP Business Suite and in data warehouses such as SAP NetWeaver BW. While analytic queries in SAP Business Suite provide access to consistent operational data, analytic queries in the SAP NetWeaver BW hub offer access to consistent, highly aggregated data across the enterprise.

SAP NetWeaver Gateway and SAP NetWeaver BW integration allows you to publish SAP NetWeaver BW content as an OData service that has been defined using multidimensional expressions (MDX) or SAP NetWeaver BW Easy Queries. While the MDX approach can also be used for SAP NetWeaver BW systems starting with 7.0, the SAP NetWeaver BW Easy Query approach is only supported for release 7.30 and higher. SAP NetWeaver BW Easy Queries are, however, easier to understand and to handle.

SAP NetWeaver BW Easy Queries are analytic queries that meet certain criteria. For a given SAP NetWeaver BW Easy Query, an RFC module is created in the system. This is done automatically by the system, based on the available SAP NetWeaver BW query definition. Using this RFC, an SAP NetWeaver BW Easy Query interface can be defined as an OData service.

To release an analytical query as an SAP NetWeaver BW Easy Query, you have to mark the corresponding checkbox in the query properties in BEx Query Designer (see Figure 5.18).

After this has been done and the query is saved, the generation of the RFC is triggered. General rules that apply for SAP NetWeaver BW Easy Queries are that characteristics are on the rows, key figures are on the columns, and free characteristics aren’t mapped to OData.

Further Resources
More information about SAP NetWeaver BW Easy Queries can be found in SAP Online Help. Specifically, we recommend the following:
Analytical annotations

Dimensions, dimension attributes, and measures are represented as properties of an entity type. The entity type representing the results of an MDX or an SAP NetWeaver BW Easy Query is annotated as sap:semantics=aggregate. Table 5.2 shows how SAP NetWeaver BW objects such as dimensions, dimension attributes, and measures are represented in OData. The table shows only the main annotations.

<table>
<thead>
<tr>
<th>SAP NetWeaver BW Objects</th>
<th>OData Representation</th>
<th>SAP Annotation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cube of Type Query</td>
<td>Entity Type</td>
<td>sap:semantics=aggregate</td>
</tr>
<tr>
<td>Dimension</td>
<td>Property</td>
<td>sap:aggregation-role=dimension</td>
</tr>
<tr>
<td>Dimension Attribute</td>
<td>Property</td>
<td>sap:attribute-for=(dimension name)</td>
</tr>
<tr>
<td>Measure</td>
<td>Property</td>
<td>sap:aggregation-role=measure</td>
</tr>
</tbody>
</table>

Table 5.2 Analytical Annotations

SAP HANA

SAP HANA is a high-performance, in-memory database. There are different integration scenarios available depending on the release of the SAP Business Suite system or the SAP NetWeaver BW system. With newer releases, these systems can run on top of SAP HANA; this means that SAP HANA replaces the relational database that was previously being used. These scenarios are called SAP ERP on HANA, SAP CRM on HANA, and SAP NetWeaver BW on HANA.

Older releases can also leverage the power of the SAP HANA database. In a side by side scenario, it’s possible to copy data from these systems to SAP HANA in real time using the SAP Landscape Transformation replication service. This scenario is called SAP NetWeaver Gateway with SAP HANA.

Both the SAP on SAP HANA approach as well as the side-by-side approach store data in SAP HANA artifacts that can be published via SAP NetWeaver Gateway as an OData service. We’ve depicted both scenarios in Figure 5.19.
SAP HANA views

With the help of the integration framework, it’s possible to publish data from SAP HANA information models that are stored in a separate SAP HANA database as an OData service through SAP NetWeaver Gateway. The integration framework of SAP NetWeaver Gateway allows implementing read-only scenarios for the following SAP HANA objects:

- Attribute views
- Analytic views
- Calculation views

The connection between SAP NetWeaver Gateway and SAP HANA in an SAP NetWeaver Gateway with SAP HANA scenario is based on a secondary database connection that is maintained in Table DBCONN. The integration scenario isn’t integrated into the Service Builder and requires code-based implementation of an MPC. This MPC must then be manually registered as a service together with a generic DPC.

ADBC interface

Because the ABAP Database Connectivity (ADBC) interface uses only one defined database user in Table DBCONN and single sign-on (SSO) isn’t possible, the SAP HANA analytical privileges concept can’t be used. As a workaround, additional authorization checks on the ABAP side have been implemented. The integration framework for SAP HANA is contained in the IW_BEP add-on for earlier versions of SAP NetWeaver, and starting from SAP NetWeaver 7.40 SP02, it’s included in the SAP_GWFND component.

A more native integration can be achieved if the SAP Business Suite system runs on top of SAP NetWeaver AS ABAP 7.40, because this release has been optimized for SAP HANA. With SAP NetWeaver 7.40, new features have been added that allow easy consumption of existing SAP HANA artifacts in your ABAP code. Specifically, SAP HANA attribute views can now be accessed natively in ABAP applications using the new DDIC entity called External View using Open SQL. In addition, the modeling information from these new objects can be leveraged in the Service Builder. To do so, an external DDIC view can be imported as a data model with DDIC structures (as described earlier in the “Import Data Model via DDIC” section in Section 5.3.1).

External OData Service

OData Services Consumption and Integration (OSCI) is an additional integration scenario that aims at enabling consumption and integration of any OData service. With SP07 of SAP NetWeaver Gateway 2.0, this functionality is fully integrated with the Service Builder. In SP06, it has to be started using Transaction /IWBEP/OCI_SRV_GEN. The integration has to be implemented on the SAP NetWeaver Gateway server system, where the IW_BEP add-on also has to be deployed. The reason for this requirement is that, for the consumption of an OData service, you need the OData library—and this only resides on the SAP NetWeaver Gateway server. In addition, you also need IW_BEP for service development on the SAP NetWeaver Gateway server.

As of SAP NetWeaver ABAP 7.40 SP02, this prerequisite will be fulfilled by any SAP NetWeaver ABAP system, because the software component SAP_GWFND comprises the required functionality described above.

OData Service (SAP NetWeaver Gateway)

The Service Builder allows you to generate a service based on an existing OData service in SAP NetWeaver Gateway. This integration scenario can be used to create a new service with the same interface as the original service, but with a changed behavior, which is accomplished by redefining methods in the new DPC extension class.

Service Generation via Model Composition

As discussed, it’s also possible to mash up multiple existing services in SAP NetWeaver Gateway, which is called model composition. The result is a new service that can be created without the need to change the existing services. It’s possible to redefine any SAP NetWeaver Gateway service...
irrespective of how it was created, which means that model composition can be used for older services that have been handcrafted, or any other service that has been built using the Service Builder. For example, it's possible to create a service in the Service Builder that includes the hand-crafted sample services GW_DEMO or RMTSAMPLEFLIGHT, even though these services were not built using the Service Builder.

A typical scenario for the use of model composition is to enhance services that have been generated from SAP NetWeaver BW, GenIL, SPI, SAP HANA, or external OData services. Think, for example, about a service that allows for creating a new purchase order and taking into account the purchase order history of your customer. The purchase order history can be retrieved using an appropriate SAP NetWeaver BW Easy Query, while a second service that creates new purchase orders can be created as well. A second scenario is the integration of data that is retrieved from SAP HANA or a third-party OData service with another SAP NetWeaver Gateway service.

5.4 The OData Channel Development Paradigm

Now that we've discussed the basics of the different tracks of the SAP NetWeaver Gateway service creation process, let's look a little closer at the OData channel development paradigm, which is a specific approach for service development. This introduction lays the theoretical foundation for Chapter 6, which goes into great detail about service development. The OData channel is part of the SAP NetWeaver Gateway basics if you plan on using service development.

The OData channel for SAP NetWeaver Gateway allows you to develop content by defining object models and registering a corresponding runtime DPC. The advantage of the OData channel paradigm is a certain freedom with respect to development; entire DDIC definitions and local interfaces of the SAP Business Suite can be used to develop SAP NetWeaver Gateway services. In addition, OData query options can be leveraged in the SAP Business Suite systems. This means that only data that has been requested by the client is selected from the SAP Business Suite system and sent back over the wire, which results in highly optimized services and major performance improvements due to a lower transferred data size.

SAP NetWeaver Gateway services with respect to the OData programming model consist of four components:

- The implementation of a model provider class (MPC) that provides the runtime representation of your model definition
- The implementation of a data provider class (DPC) that is called at runtime to perform data requests
- The technical service name that is used to register the service in the SAP Business Suite system
- The technical model name that is used to register the service in the SAP Business Suite system

The technical service name and technical model name are automatically generated with the MPC and DPC classes when generating a project using the Service Builder.

5.4.1 Model Provider Class

The MPC is an ABAP class that provides the runtime representation of your model definition; that is, the MPC defines the entity data model of a service. As such, all model information that you've defined in your project is generated into the MPC. As a consequence, you have to regenerate the MPC every time you change the model definition in your project. The MPC is important because everything you find in the
service metadata document of an OData service published via SAP NetWeaver Gateway has programmatically been defined in the MPC.

Technically, the model definition is actually generated into two classes:

- **The base class** (with the suffix _MPC). Technically, the base class is derived from the /IWBEP/CL_MGW_PUSH_ABS_MODEL super class.
- **The extension class** (with the suffix _MPC_EXT). The extension class has the base class as the super class. The extension class is the class that will be registered via the technical model name. In the extension class, you can choose which methods to redefine and which methods to inherit from the base class.

In most cases, there is no need for a developer to touch the MPC that has been generated by the Service Builder. The exception to that rule is, for example, if you want to build SAP NetWeaver Gateway services with features that can't (yet) be modeled using SAP NetWeaver Gateway tools. In this case, the developer can redefine methods in the model provider extension class (see Figure 5.20).

Usually there's no need for a developer to tap into the coding of the MPC being generated by the Service Builder. Let's still take a closer look at the methods being generated to get a better understanding of the underlying framework.

The **DEFINE** method in the MPC generated by the Service Builder contains calls to the entity type-specific define_<entity_type> methods and in addition a call to the define_Association method that creates the associations, association sets, referential constraints, and navigation properties.

The method **GET_LAST_MODIFIED** is the basis for a handshake between the SAP Business Suite and SAP NetWeaver Gateway to start a refresh of the cached metadata of the service on the SAP NetWeaver Gateway backend and the SAP NetWeaver Gateway server after the class has been changed. This method should not be changed manually.

In the entity type-specific DEFINE methods, the Service Builder generates the coding that creates the parts of the OData model that define the entity types and the entity sets that are based on entity type. The properties are created, and those properties that have been marked as a key field in the Service Builder are set as key fields in the coding:

```abap
lo_property = lo_entity_type->create_property( iv_property_name = 'ProductID' iv_abap_fieldname = 'PRODUCT_ID' ).
lo_property->set_is_key( ).
```

Finally, the entity type is bound to a DDIC structure, and one or more entity sets are created. Note that an entity type that is bound to an existing DDIC structure can leverage conversion exits as well as the labels of the data elements from the DDIC. The medium field label of a data element is used as sap:label by default:

```abap
lo_entity_type->bind_structure( iv_structure_name = 'BAPI_EPM_PRODUCT_HEADER' iv_bind_conversions = 'X' ).
```

In the method DEFINE_ASSOCIATION, you can find the generated code that defines associations, association sets, referential constraints, and navigation properties of an OData model.
5.4.2 Data Provider Class and Data Provider Extension Class

The DPC is an ABAP class that provides all methods that are required to handle OData requests. It’s called at runtime to perform these requests; essentially, we’re talking about the runtime representation of your service implementation. For instance, a DPC executes **CREATE**, **READ**, **UPDATE**, **DELETE**, **QUERY**, and many more operations.

Again you can find an extension class (suffix _DPC_EXT) and a base class (suffix _DPC). The data provider extension class inherits from the DPC base class (see Figure 5.21). The DPC extension class is the class that is registered via the technical service name. So the extension class is the class that is executed in your OData service.

It’s important to note that in the DPC, there are methods that are and are not specific to an entity set.

<table>
<thead>
<tr>
<th>Entity Set-Specific methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>DPC Method Name</td>
</tr>
<tr>
<td>&lt;ENTITYSET&gt;_CREATE_ENTITY</td>
</tr>
<tr>
<td>&lt;ENTITYSET&gt;_DELETE_ENTITY</td>
</tr>
<tr>
<td>&lt;ENTITYSET&gt;_GET_ENTITY</td>
</tr>
<tr>
<td>&lt;ENTITYSET&gt;_GET_ENTITYSET</td>
</tr>
<tr>
<td>&lt;ENTITYSET&gt;_UPDATE_ENTITY</td>
</tr>
</tbody>
</table>

| Table 5.3 | Entity Set-Specific CRUD Method Implementation in the DPC |

There are additional methods available that apply not only for a single entity set but for all of them (nonentity set-specific methods). Examples of these methods are the methods handling $expand statements, deep insert statements, or those that are called when a function import is performed. Let’s take a closer look at these examples.

- **GET_EXPANDED_ENTITY, GET_EXPANDED_ENTITYSET**
  Handling of $expand statements is offered by the SAP NetWeaver Gateway framework out of the box in a generic way after you’ve modeled the appropriate navigation property and implemented the handling of navigation properties. There might be situations where you would instead handle $expand requests by a specific application implementation. Examples are certain BAPIs such as BAPI_EPM_SO_GET_LIST that, along with the header data, also retrieve line items. In this case, when retrieving the sales order header data for a certain sales order, the corresponding sales order items are also read. If the entity set is also called to expand the line items alongside the sales order header, this results in unnecessary database requests.

- **CREATE_DEEP_ENTITY**
  The counterpart of the $expand statement is the deep insert statement, which calls the CREATE_DEEP_ENTITY method. A typical example is the case where a sales order can only be created alongside at least one sales order item. In contrast to the $expand statement, there is no generic handling of a deep insert request. The developer has to implement this method.

- **EXECUTE_ACTION**
  The EXECUTE_ACTION method is a nonentity set-specific method as well. It’s rather service semantic and is called if a function import into an OData service is called. Function imports allow you to execute functions that can
read and/or write data. Function imports are suitable whenever the business scenario requires data to be read or changed that can’t be modeled into an entity where you can use the CRUD-Q methods.

5.4.3 Technical Considerations with Respect to OData Channel Development

OData channel development can either take place on the SAP Business Suite system or on the SAP NetWeaver Gateway server, as shown in Figure 5.22. Both options are suited for certain use cases and have their advantages. Wherever you develop, the BEP component has to be installed there.

![Figure 5.22 OData Channel Development on the Hub or on SAP Business Suite](image)

5.5 Summary

Building OData services with SAP NetWeaver Gateway is done by following the SAP NetWeaver Gateway service creation process. This process is strongly supported and facilitated by the central SAP NetWeaver Gateway service creation tool: the SAP NetWeaver Gateway Service Builder. In this chapter, we introduced you to the tool and the process to establish a base of knowledge for the more technical step-by-step instructions in Chapter 6 and Chapter 7, which focus in detail on the processes of service development and service generation. In Chapter 6, you’ll also be able to take advantage of the OData channel programming paradigm that you’ve learned about here.
Contents

Foreword by Dr. Vishal Sikka ............................................................ 17
Foreword by Michael Reh ................................................................. 19
Foreword by Stephan Herbert .......................................................... 21
Introduction ..................................................................................... 23
Acknowledgments ............................................................................ 27

Part I: Getting Started

| 1 | Introduction to SAP NetWeaver Gateway ................... | 33 |
| 1.1 | Modern Business Applications | 34 |
| 1.1.1 | User Interfaces | 35 |
| 1.1.2 | Infrastructures | 42 |
| 1.2 | SAP NetWeaver Gateway for Modern Business Applications | 45 |
| 1.3 | Installation and Deployment | 51 |
| 1.3.1 | Installation | 52 |
| 1.3.2 | Deployment | 54 |
| 1.4 | SAP NetWeaver Gateway and Related Products | 57 |
| 1.4.1 | Duet Enterprise | 58 |
| 1.4.2 | SAP NetWeaver Portal | 60 |
| 1.4.3 | SAP Mobile Platform | 61 |
| 1.4.4 | SAP HANA | 63 |
| 1.4.5 | SAP NetWeaver Process Integration (PI) | 64 |
| 1.4.6 | SAP NetWeaver Business Warehouse (BW) | 64 |
| 1.5 | Summary | 65 |

| 2 | Introduction to OData .................................. | 67 |
| 2.1 | OData and REST | 67 |
| 2.1.1 | What Is REST? | 67 |
| 2.1.2 | What Is OData? | 71 |
| 2.2 | Structure of an OData Service | 76 |
| 2.2.1 | Service Document | 79 |
| 2.2.2 | Service Metadata Document | 83 |
2.3 OData Operations ........................................................... 86
2.3.1 Create ............................................................... 86
2.3.2 Read ................................................................. 87
2.3.3 Update ............................................................... 88
2.3.4 Delete ............................................................... 89
2.4 OData Query Options ................................................... 89
2.4.1 Filtering and Projecting ($filter and $select) ................. 91
2.4.2 Sorting ($orderby) .............................................. 95
2.4.3 Client-Side Paging ($top, $skip, and $inlinecount) ......... 96
2.4.4 Counting ($count) ............................................. 100
2.4.5 Inlining ($expand) ............................................. 100
2.4.6 Formatting ($format) .......................................... 104
2.5 OData in SAP Solutions ................................................. 106
2.5.1 Mobile Productivity Applications .............................. 112
2.5.2 SAP Fiori .......................................................... 112
2.5.3 SAP Jam .......................................................... 113
2.5.4 SAP NetWeaver Portal ....................................... 113
2.5.5 Duet Enterprise ................................................ 113
2.5.6 SAP Solution Manager ....................................... 114
2.5.7 SAP HANA ....................................................... 114
2.5.8 SAP-Certified Partner Solutions .............................. 117
2.6 Summary ............................................................... 118

3 Architecture and Integration ............................................ 119

3.1 Gateway Principles ..................................................... 119
3.2 Architecture ............................................................ 121
3.2.1 Consumer Tier .................................................. 124
3.2.2 SAP NetWeaver Gateway Tier .............................. 125
3.2.3 SAP Business Suite Tier ..................................... 126
3.2.4 Add-On Structure .............................................. 128
3.3 Integration with Other SAP Interfaces ......................... 130
3.3.1 Remote Function Call (RFC) ................................ 131
3.3.2 Business Object Repository (BOR) ....................... 131
3.3.3 Service Provider Infrastructure (SPI) ..................... 131
3.3.4 SAP NetWeaver Business Warehouse (BW) InfoCubes ... 132
3.3.5 Multidimensional Expressions (MDX) ..................... 132

4 Deployment Options, Installation, and Configuration .................. 137
4.1 Introduction to SAP NetWeaver Gateway Deployment .......... 137
4.1.1 Hub Deployment with Development in the SAP Business Suite System ........................................ 140
4.1.2 Hub Deployment with Development on the Hub ........ 141
4.1.3 Embedded Deployment ...................................... 144
4.1.4 Comparison of Deployment Options ....................... 146
4.1.5 Mixed Deployment Options ................................... 148
4.2 Preparing for Installation and Configuration .................... 150
4.3 Quick Start Guide .................................................... 153
4.3.1 Step 1: Deployment of the SAP NetWeaver Gateway Add-Ons .................................................... 154
4.3.2 Step 2: Activate SAP NetWeaver Gateway .......... 155
4.3.3 Step 3: Create an SAP System Alias .................... 155
4.3.4 Step 4: Create an SAP NetWeaver Gateway Alias .... 157
4.3.5 Step 5: Activate OPU Node ................................ 158
4.3.6 Step 6: Test Your Settings .................................. 159
4.4 Installation and Configuration in Detail ......................... 161
4.4.1 Installing the SAP NetWeaver Gateway Add-Ons .......... 162
4.4.2 Basic Configuration Settings ................................ 162
4.4.3 OData Channel Configuration .............................. 165
4.4.4 Business Enablement Provisioning (BEP) Configuration ................................................................. 172
4.4.5 Smoke Testing .................................................. 173
4.5 Summary ............................................................... 175
Part II: Service Creation

5 Introduction to OData Service Creation ............................. 179

5.1 Service Creation Process Overview .................................. 180
5.2 SAP NetWeaver Gateway Toolset ................................. 185
5.2.1 SAP NetWeaver Gateway Service Builder ............... 186
5.2.2 Beyond the Service Builder: Supporting Tools during the Service Creation Process .................... 188
5.3 Steps in the Service Creation Process .............................. 192
5.3.1 Data Model Definition in the Service Builder .......... 192
5.3.2 Service Registration in the SAP Business Suite System .......................................................... 195
5.3.3 Service Implementation .......................................... 198
5.3.4 Service Maintenance ............................................. 201
5.3.5 Service Generation via Redefinition ..................... 202
5.3.6 Service Generation via Model Composition ........... 211
5.4 The OData Channel Development Paradigm ................. 212
5.4.1 Model Provider Class .......................................... 213
5.4.2 Data Provider Class and Data Provider Extension Class ....................................................... 216
5.4.3 Technical Considerations with Respect to OData Channel Development ....................... 218
5.5 Summary ........................................................................ 218

6 Service Development ..................................................... 221

6.1 Data Model Definition .................................................... 222
6.1.1 Creating a Project ............................................... 223
6.1.2 Creating the Data Model ...................................... 226
6.2 Service Registration in the SAP Business Suite System ...... 248
6.3 Service Stub Generation ............................................. 254
6.4 Service Maintenance .................................................... 257
6.5 Incremental Service Implementation and Model Enhancement ......................................................... 262
6.5.1 Feed (GET_ENTITYSET) ...................................... 264
6.5.2 Single Read (GET_ENTITY) ................................... 269
6.5.3 Query Options .................................................... 271

Part III: Application Development

8 SAP NetWeaver Gateway Productivity Accelerator .... 403

8.1 Consumption Toolkits ............................................... 405
8.2 Installation .............................................................. 406
8.3 OData Sample Services ............................................. 409
8.4 SAP Gateway Productivity Accelerator OData Model Editor ....................................................... 411
8.5 Summary ........................................................................ 415
Index

A

ABAP Class Builder, 255
ABAP Database Connectivity interface, 210
Adobe Forms, 132
Agentry SAP Framework, 650, 651
Analytic views, 134
Android development, 428
  basic application, 430
  example code, 432
  model the application pages, 430
  SDE, 429
  templates, 430
  workflow application, 430
API management, 621
Application log, 560
Application Log Viewer, 563
Application requirements, 34
  infrastructure, 42
  user interfaces, 35
Architecture
  consumer tier, 121, 122, 124
  SAP Business Suite tier, 121, 122
  SAP NetWeaver Gateway tier, 121, 122
Ariba Business Network, 612
Association, 77, 85
  define, 340
  set, 341
Atom, 33, 104
AtomPub, 71, 74
Attribute views, 134
Authentication, 580, 583
  anonymous access, 601
  basic, 583
  for R2C, 600
  for cloud, 598
  for desktop applications, 593
  for mobile applications, 594
  for SAP Mobile Platform, 596
  for web applications, 591
  for web server, 599
Authentication (Cont.)
  Kerberos, 590
  OAuth, 588
  recommendations, 590
  SAML, 603
  SAML 2.0 browser protocol, 586
  SAP Logon Tickets and SAP NetWeaver Portal, 584
  user self-service, 601
  X.509 client certificate, 584
Authorization, 579

B

BAPI, 55, 127, 331
Base class, 250
Batch, 325
BEAM server, 437
Beautification, 380
BEX Query Designer, 64
BlackBerry development, 436
  code sample, 437
BOL, 204
BOR, 127, 130, 131
Breakpoints, 642
Bring your own device (BYOD), 38
Browser protocol, 141
Business Application Programming Interfaces → BAPI
Business Enablement Provisioning (BEP), 172, 622, 623
Business Enablement Provisioning templates, 166
Business Object Layer → BOL
Business Object Repository → BOR

C

C#, 37, 440
Calculation views, 134
Cascading Style Sheets Level 3 (CSS3), 457
Hybrid application, 438
Hybrid container apps, 419
Hybrid development, 420, 443
create new project, 444
package content, 447
sample code, 444
steps, 443
Hybrid Web Container, 442

I
Identity Provider (IdP), 581
Identity trends
gamification, 615
Internet of Things, 617
social media, 617
UIs, 614
InfoCubes, 130, 132
Infrastructure as a Service (IaaS), 44
Infrastructures
point-to-point solutions, 42, 46
Input mapping, 348
Installation, 52
activation, 155
add-ons, 154
information gathering, 153
node OPU, 158
Quick Start Guide, 137, 153
SAP NetWeaver Gateway alias, 157
SAP system alias, 155
test settings, 159
Installation and configuration
activate services, 164
add-ons, 162
Business Enablement Provisioning, 172
OData channel, 165
profile parameters, 163
settings, 162
smoke testing, 173
steps, 161
trust relationship, 168
Integrated development environment (IDE), 124
Internet Communication Framework (ICF), 163, 583

Internet Communications Manager (ICM), 583
Internet of Things, 617
iOS, 422
iOS development
create project, 425
minimum requirements, 423
view types, 426
IsEntity column, 247
IW_BEP, 129, 138, 154
IW_FND, 129, 138, 154
IW_FNDGC, 129
IW_GIL, 126, 129
IW_HDB, 129
IW_PGW, 129
IW_SPL, 126, 129

J
Java, 37, 440
JavaScript, 440, 446, 457
Jetty, 467
JSON, 74, 104, 124
K
Kerberos, 141, 590
Key fields, 270
M
Machine to Machine (M2M)
communication, 618
MDX, 65
Media resources, 303
Microsoft, 619
Microsoft ASP.NET, 535
Microsoft Excel, 518, 525
Microsoft Excel development, 525
Power Pivot, 525
Microsoft LightSwitch, 518, 529
development, 529
minimum requirements, 529
Microsoft LightSwitch Designer, 533
Microsoft Office 365, 518, 523, 524
minimum requirements, 519
Microsoft SharePoint, 58, 518
Microsoft SharePoint development
specify source, 522
Microsoft Windows 8, 432
MIME type, 304
Mixed deployment
use cases, 148
Mobile application development, 417
introduction, 418
Mobile Business Objects (MBOs), 442
Mobile Enterprise Application Platform (MEAP), 63
Model composition, 331, 332, 391
activate service, 392
create association, 395
define data model, 392
implement methods, 393
include service, 394
Model provider class (MPC), 197, 250
base class, 252
Model-View-Controller (MVC), 458
Monitoring, 559
Multidimensional expressions (MDX), 130, 132, 207
Multiple origin composition (MOC), 629, 631
MySQL, 491

N
Native application, 419, 420
Native development
pros and cons, 421
Navigation property, 281, 315
create, 340
definition, 285
.NET, 529, 535
Notifications, 635
pull, 636
push, 636

O
OASIS, 72
OAuth, 502, 588
2.0, 588
Objective-C, 440
OData, 33, 45, 47, 48, 57, 63
and Duet Enterprise, 113
and mobile applications, 112
and SAP, 106
and SAP Fiori, 112
and SAP HANA, 114
and SAP Jam, 113
and SAP NetWeaver Portal, 113
and SAP Solution Manager, 114
and SAP solutions, 111
as a REST-based protocol, 67
building blocks, 75
client libraries, 75
design principles, 74
introduction, 67
obsolete plugins, 490
protocol, 75
service document, 79
service metadata document, 83
OData channel, 134
activate SAP NetWeaver Gateway, 170
activate services, 170
authorization configuration, 166
configuration, 165
RFC destination, 169, 171
SAP system alias, 169
settings for service development, 170
OData channel templates, 166
OData consumer, 73
OData Mobile Client SDK, 424
OData Mobile SDK, 423
OData operations, 86
create, 86
delete, 89
query operation, 87
single read, 88
update, 88
OData producer, 73
OData protocol, 124
GW_CORE, 129
OData query operations
client-side paging, 96
counting, 100
filtering and projecting, 91
formatting, 104
OData query operations (Cont.)
inlining, 100
sorting, 95
OData query options, 89
OData sample services, 409
OData service, 75, 133
service document, 76
service metadata document, 76
structure, 76
ZGWSAMPLE_SRV, 79
OData software development kit (SDK)
for PHP, 491
Offline scenarios, 648
OLAP, 132
Open standards, 44, 46
Operations, 558
application log, 559
daily jobs, 558
periodic tasks, 558
troubleshooting tips, 568

Q
QR codes, 618
Query options, 271
Sfilter, 272
Sinlinecount, 276
Sorderby, 280
Sselect, 272
SSkip, 276
Step, 276
Quick Start Guide installation, 153
steps, 154

R
Redefinition, 331, 369
beautification, 380
Easy Query, 372
external OData service, 388
from existing Gateway services, 370
from external OData services, 370
from SAP NetWeaver BW, 370
GenIL, 370
SAP HANA, 370
SPI, 370, 382, 383
three steps, 371
Referential constraints, 315, 340
Remote function call → RFC
Repository objects, 335, 553
transport, 550, 552
REST, 33, 45, 57, 67, 124
architecture, 69
commands, 68
links, 70
multiple representations of a resource, 71
stateless communication, 71
uniform interface, 70

SAML (Security Assertion Markup Language), 141, 586
SAML 2.0 browser protocol, 586
SAP annotations, 339
SAP App Designer, 455
SAP Business Suite, 55, 62, 138, 140
SAP Business Suite tier, 126
SAP Business Workflow, 134
SAP Community Network (SCN), 616
SAP Developer Center, 51
SAP Fiori, 112, 179, 449, 450, 451, 481, 483, 614
application layers, 455
architecture, 453
available applications, 452
enhance UI layer, 480
home page, 452
introduction, 451
SAP App Designer, 455
SAP HANA, 63, 114, 133, 209, 370
analytic view, 210
SAP HANA Cloud Platform, 598
SAP HANA cloud solutions, 611, 616
SAP HANA Extended Application Services, 117
SAP Jam, 113
SAP logon ticket, 584
SAP Mobile Platform, 61, 62, 122, 124, 419, 442
SAP Mobile Secure, 62
SAP NetWeaver 7.0, 165
SAP NetWeaver 7.01, 165
SAP NetWeaver 7.02, 165
SAP NetWeaver 7.31, 141, 165
SAP NetWeaver 7.40, 54, 57, 129, 138, 140, 165, 622
SAP NetWeaver Business Process Management (BPM), 134
SAP NetWeaver BW, 64, 130, 132
MDX, 65
SAP NetWeaver Gateway, 33
add-on structure, 128
add-ons, 138, 622, 626
advanced topics, 629
and Atom, 33
and general industry trends, 610
and OData, 33
and REST, 33
and SAP HANA, 115
application creation workflow, 47
application requirements, 49
architecture, 119, 121
configuration, 137
deployment, 54, 137
for modern business applications, 45
future developments, 609
hardware requirements, 52
installation, 52, 137
installation and configuration, 150, 161
integration with other SAP interfaces, 130
introduction, 23, 33
lifecycle management, 539
monitoring, 559
SAP NetWeaver Gateway (Cont.)
open standards, 46
operations, 539, 558
prerequisites, 150
related products, 57
SAP NetWeaver 7.40, 54
security, 569
service deployment, 539
service enablement, 157
services, 126
software requirements, 54, 151
solution lifecycle, 540
testing, 539
versioning, 556
SAP NetWeaver Gateway 2.0, 138, 139, 175
SAP NetWeaver Gateway client, 541, 543
SAP NetWeaver Gateway components
IW_CORE, 52
IW_BEP, 53
IW_FND, 52, 53
IW_FNDGC, 53
IW_GIL, 53, 54
IW_HDR, 53
IW_PGW, 53
IW_SCS, 53
IW_SPI, 53
SAP_GWFND, 52
SAP NetWeaver Gateway Developer Center, 78
SAP NetWeaver Gateway developer tool for iOS, 422
SAP NetWeaver Gateway Productivity Accelerator → GWPA
SAP NetWeaver Gateway Service Builder → Service Builder
SAP NetWeaver PI, 64
SAP NetWeaver Portal, 60, 61, 113, 584
Views, 61
SAP Quick Sizer, 174
SAP Service Marketplace, 51, 152, 443
SAP Solution Manager, 64, 114, 560, 564, 573
SAP_GWFND, 129, 155
SAPGUI, 38, 40
SAPUI5 → HTML5/SAPUI5
Screen scraping component, 622, 625
SDK, 62
mobile client, 62
Security, 569
cross-site request forgery, 574
cross-site scripting, 574
input validation, 570, 573
network and communication, 569
transport protection, 570
virus scan interface, 578
Service Builder, 78, 127, 132, 157, 171, 179, 186, 331
Create Project screen, 225
functionality, 186
project, 187, 223
project tree, 228
service maintenance, 201
service registration, 196
supporting tools, 188
Service Builder tools
catalog service, 191
error log, 190
integrated test environment, 189
Service creation, 198, 221
Service design, 181
Service deployment, 198, 221
for RFC generation, 199
in service development, 198
Service maintenance, 201, 221, 257
Service Maintenance node, 160
Service Provider (SP), 582
Service Provider Interface (SPI), 131, 206, 311, 382
activate service, 386
create new project, 383
Service redefinition, 180, 202
analytic queries, 207
GenIL, 204
main process steps, 203
possible data sources, 204
SAP HANA, 209
Service Provider Interface (SPI), 206
Service registration, 248, 253
Service validation tool, 544
Sina Weibo, 489, 504
Sina Weibo development, 489, 504
Twitter development, 500
create developer account, 500
ODATA tweet, 504
SDKs, 500
TwitterOAuth-library, 501
SAP_GWFND, 129, 155
SAP населения, 39
example, 222
navigation property, 281
service implementation, 262
service maintenance, 257
service registration, 248, 258
stub generation, 254
Service document, 378
Service generation, 179, 180, 331, 333
map interface, 332
model composition, 332, 391
redefinition, 202, 331, 369
RFC/BOR, 333
Service implementation, 198, 221
for RFC generation, 199
in service development, 198
Service maintenance, 201, 221, 257
Service Provider (SP), 582
Service Provider Interface (SPI), 131, 206, 311, 382
activate service, 386
create new project, 383
Service redefinition, 180, 202
analytic queries, 207
GenIL, 204
main process steps, 203
possible data sources, 204
SAP HANA, 209
Service Provider Interface (SPI), 206
Service registration, 248, 253
Service validation tool, 544
Sina Weibo, 489, 504
Sina Weibo development, 489, 504
create application, 504
create user account, 505
PHP SDK, 512
Single read, 269, 271
Single sign-on, 580
SAP NetWeaver, 585
Social media, 489, 617
strategy, 490
Social media development, 489
Facebook, 494
PHP, 490
Sina Weibo, 504
Twitter, 500
Software as a Service (SaaS), 44
Source URI, 303
SuccessFactors, 113
SupplerCollection, 282
Sybase Unwired Platform Mobile SDK, 443
System alias
transport, 554, 555
Technical model name, 196
Technical service name, 196
Testing, 540
best practices, 547
client applications, 545
services, 541
Throttling, 633
Transaction /IW/FND/GW_CLIENT, 261
Transaction /IW/FND/MAINT_SERVICE, 257
Transaction SE24, 222
Transaction SEGW, 127, 157, 171, 179
Transaction SICF, 158, 174
Twitter, 489, 500
Twitter development, 500
create developer account, 500
ODATA tweet, 504
SDKs, 500
TwitterOAuth-library, 501
UI requirements
agility, 39
availability, 38
business orientation, 37
innovation, 37
integration, 39
maintainability, 40
nondisruptiveness, 41
reduced TCO, 41
security, 40
URLs, 69
User interface, 34
technology, 34
User management, 579
Versioning, 556
Visual Basic, 529
Visual C#, 529
Visual Studio, 520
Microsoft LightSwitch, 529
Index

Visual Studio 2010, 535
Visual Studio 2012, 529
Visual Studio C#, 620

W
Web services, 43
Windows
  OData plugin, 432
Windows CE, 432
Windows NT, 432
Windows Phone 8, 432, 619
  development, 432

Windows Phone 8 development
  add service, 433
  minimum requirements, 433
  OData client tools, 433
  sample code, 435
Windows Phone SDK, 433

X
X.509 client certificate, 580, 584
XAMPP, 491
Xcode, 37, 422, 423, 424, 427
Xcode Emulator, 428
XML, 124, 562
We hope you have enjoyed this reading sample. You may recommend or pass it on to others, but only in its entirety, including all pages. This reading sample and all its parts are protected by copyright law. All usage and exploitation rights are reserved by the author and the publisher.