Business Process Digitalization and Cloud Computing

8. Design Service Implementations

Andrea Morichetta, Phd

November 29, 2017

Computer Science Division

Table of contents

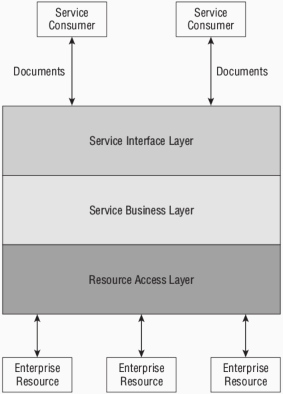
1. Basic service architecture

2. Business layer

Basic service architecture

Service architecture

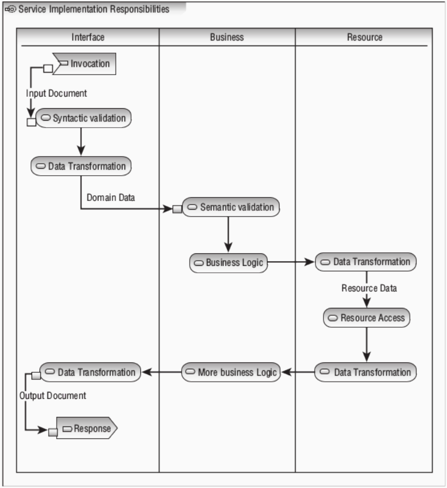
- A service is **not a simple class** that implement an interface, but should be looked as an **application** with all the characteristics thereof.
- **Service implementation model:**



Service implementation model

- Service interface layer implements:
  - the **service contract**
  - the **operations** provided by the service
  - the **document** associated with each operation
  - **data types** associated to the document
- Service business layer implements:
  - the **business logic rules**
  - the **state transitions**
- Resource access layer permit the access to:
  - the **enterprise resources** (database)
  - to existing **enterprise applications**
  - to other **business domain** and **utility services**

Layer responsibilities

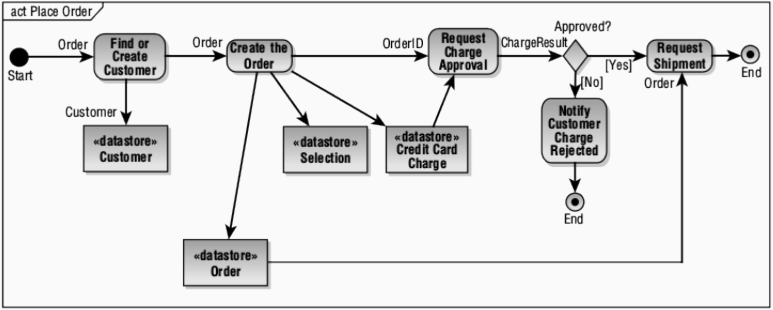


- **Interface** receive the input documents and make any required translation required to invoke business entity.
- **Business layer** duty is to processing business required to implement service contract
- **Resources** is used to access resources.

Operational logic

- **Activity diagrams** can be used for defining the services implementation.
- **Operational logic** can be implemented using Java, C# so can be useful describe the operational flow using the UML activity diagram.
- **Activity diagram** is composed by:
    - input and output parameters
    - actions (rounded rectangles)
    - datastores contains domain data (squared-off rectangle)

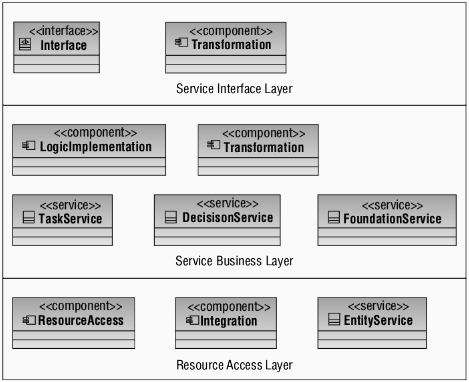
Activity diagram



1. Receives an Order document.
2. Finds the Customer of the Order. If not exists, it creates a new Customer.
3. Creates the Order with Selections and a Credit Card Charge.
4. Sends the credit card charge for the approval.
5. If the charge is approved, notifies the customer and sends the order to be packed and shipped.

Implementation of Service Layer Components

- **Interface:** exposes the operations of the service to the rest of the SOA.
- **Transformation:** change data from one format, schema, or semantic model to another.



Implementation of Business Layer Components

- **Logic implementation:** The business logic components implement the specific logical functions of the service operation.
- **Task Service:** implement business or logical tasks, such as common domain functions or business utilities.
- **Decision Service:** implement (complex) business rules and provide an external mechanism for specifying and maintaining the rules.
- **Foundation Service:** provide traditional platform distributed services such as authentication, logging and configuration.

Implementation of Resource Layer Components

1. **Integration Component:** exposes integration services and make their legacy functions and data available to the service operation.
2. **Resource Access Component:** encapsulate the access to resources, such as datastores.
3. **Entity Service:** provide common service-based access to common business entities.

Document receipt

- The implementation of a service interface starts with the **receiving of input document**.
- to be sure that the values are **meaningful and correct for the operation** is possible to validate the documents:
  - **Syntactic validation:** determines if the parameters are correctly formed.
  - **Semantic validation:** determines if the value are correct and meaningful.
- The parameter structures can be specified in the **schema files**.

  Syntactic validators can be **implemented** either in a interface layer or in the business layer.
- **Data transformation** from semantic data to business entity is straightforward and can be implemented in any general-purpose programming language.

Business layer

Implementing the business layer

The business layer contains the implementation of **business logic and business entities**.

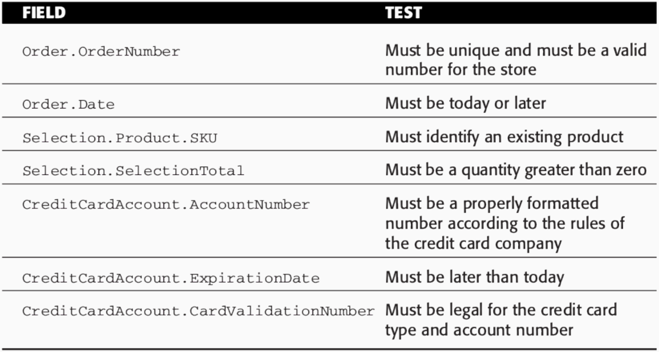
The business implementation consists on:

- Semantic validation of the input parameters
- Performing the business logic of the operation
- Returning a result

Semantic input validation

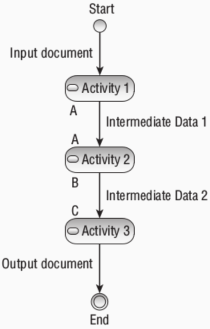
Semantic validations determine the **correctness of specific input values** according to business rule.

It require to examining the data in the context of the service's overall environment, ensuring that all of the **information** for the service invocation is **complete**.



FIELD	TEST
`Order.OrderNumber`	Must be unique and must be a valid number for the store
`Order.Date`	Must be today or later
`Selection.Product.SKU`	Must identify an existing product
`Selection.SelectionTotal`	Must be a quantity greater than zero
`CreditCardAccount.AccountNumber`	Must be a properly formatted number according to the rules of the credit card company
`CreditCardAccount.ExpirationDate`	Must be later than today
`CreditCardAccount.CardValidationNumber`	Must be legal for the credit card type and account number

Business logic of the operations



- The system **starts** with the input of data
- **Data transformation** is required between start and activity 1
- Activity 1 should **return the correct data** for activity 2 or transform the data
- Activity 3 is an utility service is necessary to decide **how to retrieve information**, what information is required for call the service and if you need **more information in input**.

Computing and returning results

- The **returning results** depend by the type of computation:
  - custom code
  - components
  - existing services
- Returning results not require validation
- The **activities** involved are:
  - Collecting data
  - Calculation
  - Translation
  - Formatting the return document

Implementing the resource layer

- Data access logic: is responsible for **retrieve business entity in a database** or other entity inside a business layer
- Resource access layer: contain **invocation to other services** to retrieve resources. This layer include the code for interaction with external resources.

Possible Web Service Engines:

Apache Axis2 Link Link2

GlassFish Metro

CXF

Questions?