Service Specification

This document presents an outline for a Service Specification document.

**Table of Contents**

[1 Service Overview 3](#_Toc316046946)

[1.1 Service One-Line Description 3](#_Toc316046947)

[1.2 Service Abstract 3](#_Toc316046948)

[2 Service Context 3](#_Toc316046949)

[3 Intended Utilization Scenarios 3](#_Toc316046950)

[4 Interface Definitions 3](#_Toc316046951)

[5 Referenced Components 4](#_Toc316046952)

[6 Observable State 4](#_Toc316046953)

[7 Triggered Behaviors 4](#_Toc316046954)

[8 Coordination 4](#_Toc316046955)

[9 Constraints 4](#_Toc316046956)

[10 Non-Functional Behavior 4](#_Toc316046957)

[11 Deployment 5](#_Toc316046958)

[Appendix A: Service Interface Specifications 5](#_Toc316046959)

[Referenced Interface Specifications 5](#_Toc316046960)

# Service Overview

## Service One-Line Description

This description is the most concise possible statement of the service’s purpose and is intended to help the naïve potential user quickly identify services of potential interest.

## Service Abstract

The abstract is a short one-paragraph description of the service’s purpose, functionality, and intended usage. Its purpose is to help the naïve potential user further refine their identification of potentially interesting services.

# Service Context

The context describes the architecture pattern into which the service is intended to fit. It shows both the consumer(s) of the service and the observable components whose supporting behavior is required in order for the service to execute properly. It summarizes the observable dependencies.

The context can be shown at different levels of detail. At the highest level of abstraction, the context shows all of the components and identifies the interfaces involved in their interaction. Some additional context information is required to complete the specification, specifically the details of the interface operations and data structures along with the communications channels over which the interactions occur.

This additional information may be shown in the context or included in other sections of the specification. In the accompanying example, the interface details are given in the triggered behavior descriptions, and the communications channels are given in the deployment specifics section. Placing the interface details in the triggered behavior descriptions makes the context overview concise and easy to understand. Placing the communications channels in the deployment specifics allows different communications channels to be specified for different environments.

# Intended Utilization Scenarios

The utilization scenarios provide a behavioral overview of how the service is intended to fit into the larger business processes. In simple situations in which the service’s participation in the business process amounts to the invocation of a single service operation, these scenarios represent individual use cases for the individual operations. With more complex services, on the other hand, a single business process execution may involve multiple interactions with the service. Such is the case with the Payment Manager used in the accompanying example. In these cases, the utilization scenarios spell out the possible sequences of interactions. In all cases, the utilization scenarios should clearly indicate how the work of the service is coordinated with other work in the business process.

# Interface Definitions

This section defines the interfaces provided by the service. It details the operations of each interface and the data structures used by those operations. While the details for some interfaces can be provided by SOAP WSDL documents, for readability it is prudent to augment such detailed textual representations with graphical representations using UML class diagrams. Place the diagrams here, and the text of the WSDL in an appendix.

# Referenced Components

This section defines the interfaces of other external components that are used by the service. This information is generally taken from the specifications for those components, but it is useful to replicate those interface definitions in the present specification.

# Observable State

Many services are stateful, with some operations of the service altering the state and other operations making that state visible to external components. This section identifies this information.

Services often statefully maintain replicas of information that originates in other components. This section identifies this information and the triggered behavior (if any) involved in maintaining its consistency.

# Triggered Behaviors

The triggered behaviors define the service’s behavior in response to triggering events along with its dependencies on the behaviors of external components. This section enumerates those triggering events and provides details of the ensuing observable behaviors. These behaviors can include interactions with external components and interactions with the internal observable state maintained by the service.

Three broad categories of triggers should be included: operation invocations, notifications, and time-based events.

# Coordination

The manner in which the service’s activity can be coordinated with that of other components must be clearly understood by the service consumer. To the extent that the intended utilization scenarios document this coordination, simple reference to the scenarios suffices for documentation. Otherwise, this section should document the possible coordination as well as the behavior when coordination fails (e.g. observable behavior upon lack of response in a request-reply interaction).

# Constraints

If there are constraints on the use of service operations, these must be documented. In some cases such as allowed sequences of operations, the utilization scenarios may capture this in sufficient detail. In other cases, particularly involving business rule constraints, these need to be explicitly identified. Service consumers need to understand these restrictions.

# Non-Functional Behavior

This section documents the non-functional capabilities of the service. Throughput and corresponding response times for individual service operations should be documented. Required security constraints (authentication, authorization, encryption, non-repudiation) are specified on a per-operation or per-interface basis, as appropriate. Availability, outage constraints, and recovery times may be specified on a per-service, per-interface, or per-operation basis. Whether or not the service guarantees that requests are processed in the order received is another important non-functional behavior.

In some cases, the mechanisms used to achieve fault tolerance, high availability, or load distribution may require specific actions on the part of service consumers or the use of external components. To the extent that the service consumer must be aware of these mechanisms, they must be documented as part of the service specification. In these situations, it is important to clarify whether the performance-related specifications apply to a single instance of the service or to the aggregate capabilities of the multiple instances.

# Deployment

Some observable characteristics of services vary depending upon the deployment environment. SOAP endpoints, for example, will likely be different in development, test, and production environments. Exposed fault-tolerance, high-availability, and load distribution mechanisms may also vary between environments, and these differences need to be documented.

Appendix A: Service Interface Specifications

Appendix B: Place the textual definitions of service interfaces (e.g. WSDLs) here.

Referenced Interface Specifications

Place the textual definitions of referenced service interfaces (e.g. WSDLs) here.