



NextGRID SLA Template Repository Use Cases

Editors:	Roland Kübert	HLRS

Date	Author	Comments	Version	Status
26.06.2007	RK	First draft	0.1	Draft
11.10.2007	RK	Incorporation of comments from the internal review	0.2	Draft
19.10.2007	RK	Response to second review	0.3	Stable draft
29.01.2008	RK	Updated use cases	0.4	Stable draft
05.02.2008	DS	Final QA	1.0	Final



1 INTRODUCTION.....3

2 USE CASES.....3

2.1 Publish Template4

2.2 Revoke Template5

2.3 Query by Name5

2.4 Query Random Template for a Bid.....5

2.5 Query all Templates for a Given Bid5

2.6 Query Templates.....6

3 REFERENCES.....6

1 Introduction

This document describes the basic set of use cases for a NextGRID SLA Template Repository, hereafter known as the “template repository”. Along with the NextGRID SLA Template Repository Profile document, this document forms part of the NextGRID Generalised Specification for an SLA Template Repository in a NextGRID compliant Grid.

The NextGRID Generalized Specifications aim at capturing NextGRID architectural concepts in a set of profiles that may be composed together. These profiles are specified in such a way that they could be implemented in terms of other well known specifications. While overall consistency is achieved at the conceptual level, and captured through the motivating Use Cases accompanying each Generalized Specification, system implementations, which may be based on other specifications, may not be consistent with this profile. Thus, each profile defines a realisation of the underlying concept that can be implemented. However, implementers of the NextGRID architecture may need to support multiple underlying specifications.

A template repository is a component that allows the storage and retrieval of SLA template documents. Therefore, it acts as a means of publishing SLA templates by a service provider, while at the same time acting as a means for a service consumer to obtain SLA templates.

The use cases introduced in this document describe interactions with a template repository.

2 Use Cases

The NextGRID SLA Template Repository use cases show the usage of a template repository by two actors, a *publisher* and a *requester*.

The *publisher* will normally be a service provider who wishes to publish SLA templates corresponding to the services that they offer. The *requester* can be represented by various components and users of services provided by service providers.

It can be envisioned that the template repository is queried by the Universal Dynamic Activity Package (UDAP) [1] [2] on behalf of a service consumer.

We do not provide a fine-grained distinction of the different entities which can act as a *requester* as this does not seem necessary. For example,

- a negotiator component, acting on behalf of a service consumer;
- an SLA broker component;
- a UDAP instance; or
- a service consumer itself;

could each interact with the template repository.

Figure 1 shows an overview of the SLA management use-cases.

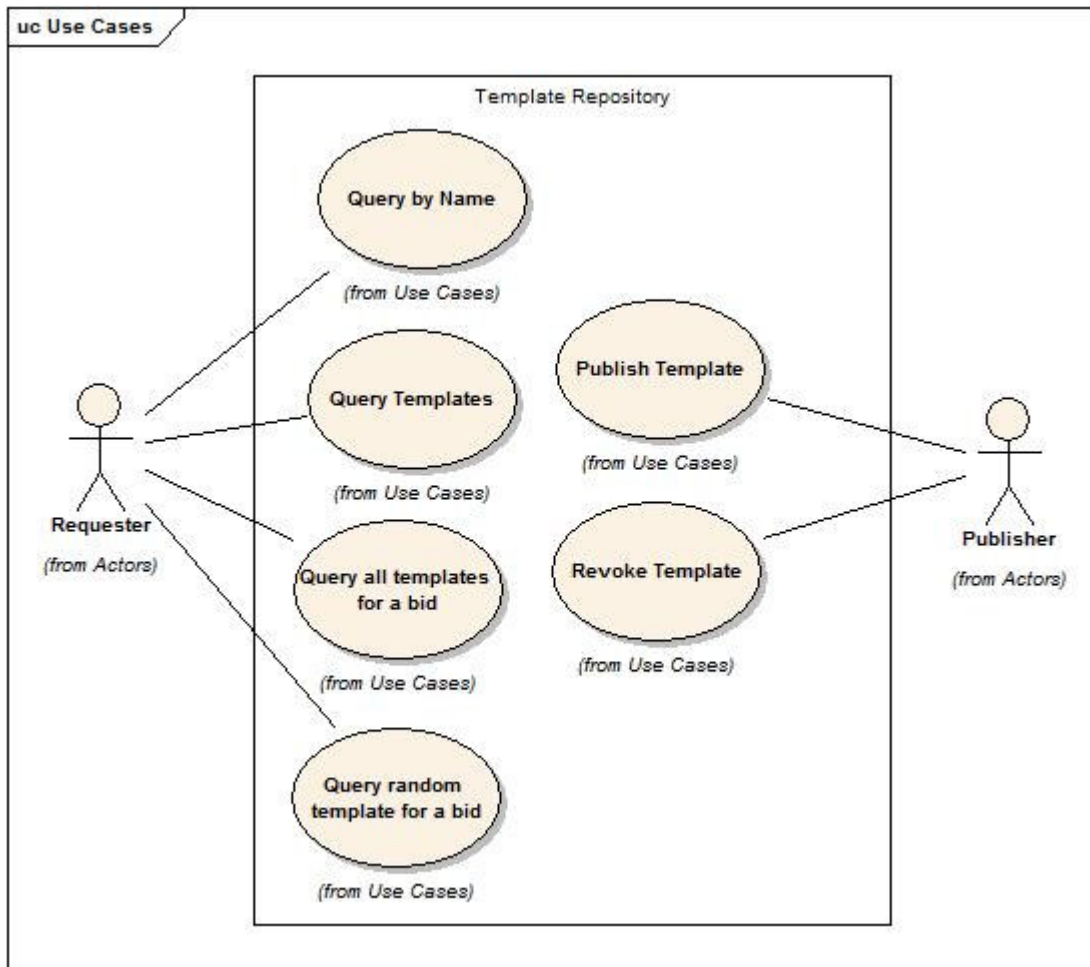


Figure 1: SLA Template Repository Use Cases.

2.1 Publish Template

A publisher (service provider) wants to publish SLA templates for a specific service that he provides, so that potential requesters (service consumers) can obtain those templates and bid to use that service.

Instead of providing a proprietary interface for obtaining SLA templates, the publisher can publish templates in a NextGRID Template Repository. To publish a template, the publisher first chooses a name, which must be of the XML Schema data type 'string' [4], by which the template can be identified and accessed, and then publishes the chosen template to the template repository for storage under the selected name.

The name is a unique identifier, that is, there exists a one-to-one mapping between a name and an SLA template. Every name is related to exactly one SLA template and every SLA template has exactly one name assigned to it.

The Template Repository restricts access to the method for publishing SLA templates to the publisher. If an SLA template with the same name as an existing SLA template is published, the stored SLA template is overwritten. It is the responsibility of the publisher to ensure that he is not overwriting a useful template. If a template with the chosen name is already published, the template is replaced by the new one being published for the same name.

2.2 Revoke Template

If a publisher does not want to offer a specific SLA template any more, he can revoke the template, which removes the template from the template repository. The publisher is the only person allowed to revoke templates.

To successfully revoke a template, the template must be unambiguously identified by its name. If a template with the specified name cannot be found, an error message is returned to the publisher; if revoking is successful, a success message is returned.

2.3 Query by Name

A requester looking for a specific template can query the repository for a template using that template's name. This will return at most one template, if a template stored with the given name has been found. Otherwise, no template will be returned.

2.4 Query Random Template for a Bid

A requester looking for a template corresponding to a bid may obtain it by sending the bid to a template repository. The template repository will try to match the bid and return a corresponding template if one is found. The term "random" specifies that if more than one matching template is found, a template may be selected by the template repository at random.

Exactly when a template matches a bid is implementation specific and depends on the concrete template repository.

2.5 Query all Templates for a Given Bid

A requester may be interested in obtaining more than one template for a given bid. Therefore, he may submit a bid and will obtain all templates that correspond to the given bid.

Exactly when a template matches a bid is implementation specific and depends on the concrete template repository.

2.6 Query Templates

A requester, who is interested in obtaining templates that match a specified set of terms, may query the template repository using a dialect specified by him and a query specified in the given dialect. The template repository must support at least XPath [3] queries; the support of other dialects is optional. If a dialect specified by the requester is not supported, the template repository must provide an error message which clearly states all supported dialects.

Querying templates with queries specified by the requester allows a precise specification of properties that the template must fulfil. If any query results in only a portion of a template being obtained, an empty result set will be returned by the template repository.

As well as the ability to return a single matching template, it must be possible to obtain all templates published in the template repository which match a given query.

3 References

- [1] UDAP Schema, NextGRID, September, 2007, http://www.nextgrid.org/GS/management_systems/UDAP_framework/NextGRID_UDAP_framework_schema.pdf.
- [2] UDAP Use Cases and Requirements, NextGRID, September, 2007, http://www.nextgrid.org/GS/management_systems/UDAP_framework/NextGRID_UDAP_framework_usecases.pdf.
- [3] XML Path Language (XPath) Version 1.0, W3C Recommendation, November, 1999; <http://www.w3.org/TR/xpath>.
- [4] XML Schema Part 2: Datatypes Second Edition, W3C Recommendation, October, 2004; <http://www.w3.org/TR/xmlschema-2/>