



NextGRID Basic Profile Use Cases

Editors:	Vivian Li	FLE
	David Snelling	FLE

Date	Author	Comments	Version	Status
25/07/2006	Vivian Li	Initial Version	V0.1	Draft
24/03/2007	David Snelling	M30 Version	V0.2	Final
13/08/2007	Clive Davenhall	Typos and grammatical tweaks	V0.3	Final
02/01/2008	David Snelling	Updated References and other minor changes.	V1.0	Final



- 1 INTRODUCTION..... 3**

- 2 USE CASES..... 3**
 - 2.1 Security Model 3**
 - 2.1.1 Security Key..... 3
 - 2.1.2 Secure channel 4
 - 2.1.3 Secure Message..... 4

 - 2.2 Service Address..... 4**
 - 2.2.1 Unambiguous Web Services Endpoint..... 4
 - 2.2.2 Web Services Endpoint Name 5
 - 2.2.3 Web Service Endpoint Address Identifier..... 5
 - 2.2.4 Endpoint Reference Resolution..... 5

 - 2.3 Service Properties 5**

 - 2.4 Notification 6**

 - 2.5 Resource Termination 6**

- REFERENCES 6**



1 Introduction

The NextGRID Basic Profile defines certain general properties and corresponding portTypes that apply to all NextGRID service instances. This document describes use cases of the NextGRID Basic Profile (hereafter, “the Profile”) with respect to: obtaining a security key, establishing a secure channel, addressing a service, obtaining certain predefined properties of a service, sending notification messages when a property changes, service instance lifetime management, and dealing with minimal SLA infrastructure.

The NextGRID Basic Profile is part of the NextGRID Generalized Specifications, which aims to capture NextGRID architectural concepts in a set of composable profiles. 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 specification, the implementation in terms of other specifications may not be consistent between different profiles. Thus each profile defines an implementable realisation of the underlying concept, but implementers of the full NextGRID architecture may need to support multiple competing underlying specifications.

2 Use Cases

2.1 Security Model

In order to ensure the secure and interoperable interaction of Web services in the context of distributed resource management and grid computing, three fundamental security capabilities have to be provided by the Profile to ensure a minimal level of security: key information of a service endpoint, communication channel between a client and a service, and message level security. These security elements are defined in conjunction with the OGSA Basic Security Profile. Their use cases are as follows.

2.1.1 Security Key

It is quite often the case that a client wants to send encrypted messages to a service, or to make a policy decision about whether or not a service is qualified to serve its requests. In both cases, the client will have to know the key information of the service before it can proceed further. For example, the public key of the service can be used to encrypt a message that only that service can read. This key is the public part and can be authenticated using the PKI, which created it, so no special care needs to be taken when distributing it. Only a simple distribution mechanism is needed.

The EndpointKeyInfo from the Profile is specially designed for this purpose; it contains all the key information binding, and is embedded inside the Metadata element in the EndpointReference element of a service instance, so that as long as the client has the service EPR, the key information associated with this service is also available before any communication with the service is needed.

2.1.2 Secure channel

To achieve a mutual authentication, better integrity and confidentiality, the Profile also mandates the use of a secure transport layer protocol. In normal Web interactions, anonymous TLS (Transport Layer Security) is adequate as other techniques are used to authenticate the client (if required), e.g. credit card number. For Grid based interactions, a standardized mechanism is needed for authenticating the client to the service. The profile requires a sender to use HTTP over TLS using an identity certificate when establishing an HTTP connection; it requires the receiver to authenticate the sender based on this certificate.

2.1.3 Secure Message

In case more features are required than the secure channel can provide, message-level security is a good alternative or supplement to that offered by basic Web services standards. The need to secure a message across several intermediaries is a common example. Using an X.509 certificate to validate a public key that may be used to authenticate a SOAP message or to identify the public key with which SOAP message was encrypted. Note that for completeness message level security remains part of the basic profile use cases, although the Basic Profile does not profile it. There is a more complete treatment of message level security in the NextGRID Security profile.

2.2 Service Address

As a W3C standard, WS-Addressing is the universally accepted mechanism for identifying or referencing a service endpoint to clients or other service instances. The core element defined by WS-Addressing is the Endpoint Reference type, which comprises a single required *Address* element and a number of optional elements. The mandatory *Address* element is an address URI that can be used to represent a published service endpoint in the format of either a network address or a logical address, e.g. <http://196.20.1.1:8080/services/JobService> or <http://myservice:8080/services/JobService>. From a client's point of view, there are four possible use cases for a service address to guarantee the uniqueness of a resource in time and space; they are described as follows:

2.2.1 Unambiguous Web Services Endpoint

In a Web services environment, it is always possible that after initial contact with a service, the client goes offline, or the service becomes unavailable because of hardware or power failure or some other misadventure. Under these circumstances, the client might want to resume communication with the service from the previous session when the connection resumes. This behaviour raises the need for the client to find out the same resource instance that it used before the disconnection; hence a service endpoint is required to be unambiguous. The Profile tackles this issue by placing restrictions on the creation and use of an EPR necessary to ensure unique delivery in a prescribed context. Specifically it uses the *Address* element to specify only a service location for the same type of services, but takes advantage of the extensibility of WS-Addressing to create a non-normative element to uniquely identify each service instance, e.g. a ResourceDisambiguator of a unique ID, thus guaranteeing the client access to the same resource. To align more correctly with the spirit of the notion of a URI, the profile also recommends that all the necessary resource disambiguation information be contained in the *Address* element of the EPR. Note as a result this profile recommends that WS-Addressing Reference Parameters are not used for resource disambiguation.

2.2.2 Web Services Endpoint Name

In the case where a resource identifier is used in policy expressions, attribute bindings, caching, and client-side audit, the pattern of an *Address* URI plus a unique identifier for identifying a service endpoint is less efficient, because it requires two compulsory parts of information. A *ws-endpoint-name* element defined by the Profile that is embedded in the metadata section of an EPR can serve this purpose, so that a client can obtain a meaningful resource identifier from one single metadata element.

2.2.3 Web Service Endpoint Address Identifier

The *ws-endpoint-name* solution does not address the scenario when audit and policy enforcement are requirements in intermediaries, soap-routers, and server runtimes, since the *ws-endpoint-name* is not included in the SOAP message in a standard WS-Addressing transformation from EPR to SOAP message. The Profile defines a requirement placed on an EPR *wsa:Address* field to ensure its uniqueness in space and time, thus allowing the client to take the advantage of this unique endpoint identifier and also to use it as part of the standard SOAP messages.

2.2.4 Endpoint Reference Resolution

The need for resolution arises when a client needs to access an endpoint of a mobile resource. There are possible two solutions: the client might simply invoke a resolver service that takes an IRI identifier as a WS Endpoint Identifier and returns a new valid EPR, or to use a *RenewableReference* endpoint that can return the current application-EPR.

2.3 Service Properties

Before invoking specific operations on a service, a client might want to know some general information about what services are offered: e.g, what is the *portType* of the service, what are the composed service interfaces, and what are the other resource properties that this service supports, or after initial contact with the service, the client may need to know the service endpoint again. All this information should be made available from resource, which include:

- The WSDL interface of the service,
- The constituent WSDL interfaces composed to create the service interface,
- A list, probably in the form of QNames, of the other properties available from the service, and
- The *EndPointReference* of the service.

If notification capability is required, a client can just subscribe to notifications on certain interested resource properties via the *ResourcePropertyValueChangeNotification* resource property. Some clients may also be interested in the SLA templates which are designed for negotiating an SLA. Values of the above mentioned resource properties could be retrieved by querying the resource property, or by getting the value of a resource property.

2.4 Notification

In a grid system, components quite often need to communicate with each other asynchronously. This interaction and communication among objects or Web services often requires notifications. For instance, the system administrator may want to receive notification messages whenever the load on a resource exceeds its capability, or a client wish to be notified when the job it submitted has finished execution. In these cases, the administrator or the client can subscribe to a NotificationProducer. If successful, the NotificationProducer will monitor the status of a resource, and send notification messages to the administrator or the client who acts as a NotificationConsumer. The NotificationConsumer can receive a notification message, either by the NotificationProducer *pushing* the message to it, or by sending a get message request to the pull point to *pull* the message.

Since NextGRID services operate a properties-based data model, it is preferable that the public representation of this data model be consistent across the various mechanisms for exposing it. In this case the notification pattern and the Service properties patterns should have the same basis. Furthermore, this facilitates moving from existing notification technologies to new ones without the development of a whole new data model.

2.5 Resource Termination

A service instance should support the notion of lifetime for economic reasons. The Profile defines two ways to destroy a service instance: immediate destruction or scheduled self-destruction. In many scenarios, it is appropriate for a client to explicitly invoke immediate destruction. However, it is also common that in a distributed computing environment, the client may become disconnected from the service endpoint and therefore unable to invoke immediate destruction. In such cases, establishing the scheduled termination time can terminate the service instance when a specified time expires. The lifetime of a service instance can also be extended by periodically extending its termination time.

References

- [OGSA WSRF Basic Profile] Foster, I., Maguire, T. and Snelling, D.: OGSA WSRF Basic Profile 1.0. GFD.72, GF 2006.
<http://www.ogf.org/documents/GFD.72.pdf>
- [OGSA Basic Security Profile] T. Mori and F. Siebenlist: OGSA Basic Security Profile 1.0 – Core, GFD.86, OGF 2007, <http://www.ogf.org/documents/GFD.86.pdf>.
- [OGSA Basic Security Profile – Secure Channel] T. Mori and F. Siebenlist: OGSA Basic Security Profile 1.0 – Secure Channel, GFD.99, OGF 2007, <http://www.ogf.org/documents/GFD.99.pdf>.
- [Web Service Endpoint Identification and Resolution] Siebenlist, F., Snelling, D.: Web Service Endpoint Identification and Resolution: Use Cases and Requirements 1.0 GGF OGSA Working Group (OGSA-WG), 2006.