**Network Service Interface Architecture**

Status of This Document

This document provides information to the Grid community on the service interface between a requesting software agent and the provider software agent that offers and delivers a Network Service. It is intended to describe the processes and environment in which software agents interact to deliver the service(s). Representing applications or other networks, these agents may request certain services of other network agents. Distribution is unlimited.

Copyright Notice

Copyright © Open Grid Forum (2008-2010). All Rights Reserved.

Trademark

OGSA is a registered trademark and service mark of the Open Grid Forum.

Abstract

The Network Service Interface (NSI) is defined to be the set of protocols and parameters that are used between a software agent requesting a network service and the software agent providing that network service. The Network Service Interface Architecture describes a Service Plane Network topology model, and associated processes and concepts that occur among the Network Service Agents in order to satisfy service requests. This document and its partner document the NSI Protocol recommendation (GWD-R) form the NSI definition.

Contents

[1. Context and Overview 2](#_Toc264032691)

[2. NSI architecture 3](#_Toc264032692)

[2.1 The Network Service Interface 3](#_Toc264032693)

[2.2 The Network Service Agent 3](#_Toc264032694)

[2.3 The Network Resource Manager 4](#_Toc264032695)

[2.4 NSI Services 4](#_Toc264032696)

[2.5 NSI service extensibility 5](#_Toc264032697)

[2.6 The NSI Service Plane 5](#_Toc264032698)

[2.7 Hierarchical communications model and federation 6](#_Toc264032699)

[3. The NSI Protocol 8](#_Toc264032700)

[3.1 NSI Protocol Sessions 8](#_Toc264032701)

[3.2 NSI messages 8](#_Toc264032702)

[3.3 NSI Service Instances and Primitives 9](#_Toc264032703)

[3.4 NSI Service Definitions 9](#_Toc264032704)

[3.5 Temporal aspects of NSI services 10](#_Toc264032705)

[3.6 Trust and authentication in NSI 10](#_Toc264032706)

[3.7 Error handling NSI 12](#_Toc264032707)

[3.8 Transport failure awareness 14](#_Toc264032708)

[4. Representing network resources 14](#_Toc264032709)

[4.1 Describing network topologies 14](#_Toc264032710)

[4.2 Service Termination Points 15](#_Toc264032711)

[5. The NSI Services 17](#_Toc264032712)

[5.1 NSI Connection Service 17](#_Toc264032713)

[5.1.1 Connection service concepts 17](#_Toc264032714)

[5.1.2 Service definitions for Connection Services 18](#_Toc264032715)

[5.1.3 The Connection Service States 19](#_Toc264032716)

[5.1.4 Connection reservation messages 20](#_Toc264032717)

[5.1.5 Connection reservation and timing parameters 21](#_Toc264032718)

[5.1.6 The Path Object 22](#_Toc264032719)

[5.1.7 Tree and Chain Connection modes for inter-domain pathfinding 22](#_Toc264032720)

[5.1.8 The Connection Path Algebra 24](#_Toc264032721)

[5.2 Future Services 24](#_Toc264032722)

[6. Contributors 25](#_Toc264032723)

[7. Glossary 25](#_Toc264032724)

[8. Intellectual Property Statement 26](#_Toc264032725)

[9. Disclaimer 26](#_Toc264032726)

[10. Full Copyright Notice 26](#_Toc264032727)

[11. References 27](#_Toc264032728)

# Context and Overview

Over the last decade, global networks have begun delivering high performance transport services directly to applications that require performance levels or capabilities unavailable in conventional best-effort IP networks. The ability to create connections between a fixed set of ports worldwide, with specific, predictable, and often demanding performance characteristics, enables emerging global collaborations to establish well-defined and highly customized network environments to support the end users and their applications. This is particularly true within the Research and Higher Education space and the growing Grid community.

Connections across these transport networks have been historically reserved and provisioned in a variety of ways. The most common approach is manual provisioning – typically performed by a network engineer. More recently, some networking communities have developed tools and protocols to automate the process of network resource allocation and to allow the user or application to participate directly in the path creation process. These new approaches to automating transport connection provisioning are the basis for the standardization effort being described in this recommendation.

Automated connection-oriented transport provisioning capabilities are currently being deployed by Research & Education (R&E) providers as well as by commercial providers, and could eventually be implemented in home/ retail networks as deployment progresses. These automated provisioning systems, while being developed independently by different groups, all have common elements. They have developed software based control and/or management agents to regulate access to these resources, to schedule and reserve resources, to trigger or control timely provisioning of the network resources, and to monitor and release resources. These controllers are deployed in two different contexts. One context is application (or Grid) centric, where a network provides a resource to an application or middleware. The other context is network centric, where network resources are collaboratively shared among networks to expand or improve network performance or reach. In the former context, a user or application agent is requesting the service of a network provider. In the latter context, one network is interacting with other network(s) to manage these resources and deliver a comprehensive and well integrated service portfolio to the user community. This informational document defines an architecture for the NSI protocol which works in either of these contexts.

The NSI defines several key architectural elements: the Network Service Interface (NSI), the Network Service Agent (NSA), the NSI Protocol, the supporting topology model, and a set of basic services. These concepts are assigned to a conceptual Service Plane. The Network Service Interface (NSI) architecture describes an environment within which network capabilities and components are abstracted and manipulated as “network resources.” Within the NSI architecture, network resources can be selected, allocated, interrogated, and manipulated by software agents on behalf of networked applications. Network resources and capabilities are presented to the consumer through a set of “network services.” The Network Service Interface presents a simple “one-stop shopping” model for interacting with these services.

Version 1.0 of these NSI supports only one Network Service – the Connection Service.

Network Services are delivered by the capabilities of participating operators. They will make use of a range of network related functional capabilities such as topology sharing, path finding, resource reservation, hardware provisioning, and other ancillary services and functions.

This NSI Architecture document describes the broad concepts, models, and key objects necessary to realize the delivery of these services. The NSI Protocol recommendation (GWD-R-XXX) forms the partner document to this architecture document and describes the specific and detailed messages, associated parameters, transactions, and state transitions that occur to request and deliver network services.

# NSI architecture

## The Network Service Interface

The Network Service Interface (NSI) provides secure and reliable sessions for service related communication between NSAs. An instance of the NSI exists at the boundary between two communicating software agents: the Requester NSA and the Provider NSA. These agents interact to realize the delivery of a Network Service intrinsic to the network infrastructure. In this model, the Requester NSA requests some service, and the Provider NSA attempts to deliver it (see Figure 1).

Figure 1: NSI interface

## The Network Service Agent

The NSA is central to the NSI architecture since all NSI processes are invested in the Network Service Agent (NSA). The role of the NSI is to provide a platform for delivering Network Services by exchanging information between NSAs. Each NSA can also interact with its Transport Plane to deliver the local part of the Network Service and with other infrastructures such as those for security and monitoring.

The NSA assumes two possible roles - Requester and Provider. As a Requester, the NSA requests network resources and as a Provider it delivers these network resources to create a service. The Network Service Agent may at times act as a requester over one NSI interface while acting as a provider at a different NSI interface.

The NSA incorporates a number of functional components; some of which may be defined NSI Network Services others may be NSA internal functions; this is depicted in Figure 2. An example of the former might be a Connection Service or a Topology Service. An example of the latter may be a path-finding function.

Figure 2: Roles of Network Service Agent

## The Network Resource Manager

The provider NSA may incorporate a Network Resource Manager (NRM). The NRM manages the part the Network Service implemented locally; this is shown in Figure 3.

The NSI architecture allows many Network Service Agents (NSAs) to be interconnected with NSI interfaces; details relating to such a federation of networks are described in paragraph 2.7.

Figure : NRM associated with Provider NSA

## NSI Services

The NSI protocol deals with an abstracted model of transport services. This abstraction reduces or hides many of the real-world complexities of delivering a particular transport service. For instance, the NSI Connection Service takes a rather complex data transport circuit and presents it as if it were a simple pipe of a certain size between two locations. The user only needs to specify the ends, and the diameter of the pipe. The user places data in one end of the pipe and it emerges some time later from the other end in the same manner. This abstract concept is a simplified and convenient means of presenting the key functional aspects of the service object while hiding most or all of technical details that are in general not relevant to the application.

In practice, the abstraction of a service presents a basic set of service primitives and a bounded set of parameters on those primitives that fully define what will be delivered as result of the service. To make this point, the NSI Connection Service provides primitives such as ReserveRequest and CancelRequest that perform very clear operations on a Connection, and these primitives are carefully bounded by the parameters associated with each – such as capacity, or end points, or start time. The detail of coordinating schedules and the exchange of circuit provisioning information across possibly many networks is hidden.

In the NSI Architecture, these Services exist in the form of service specific agents known to, or incorporated within, the NSA. The NSI Protocol provides an extensible framework for the definition and incorporation of network services. The NSI has defined only one initial service: the Connection Service to meet the emerging need for automated creation and management of network connections.

## NSI service extensibility

The NSI Interface must provide a common framework in which Network Services can be delivered. To achieve this aim, the NSI Architecture is extensible; it inherently supports the ability to add new Network Services as they emerge. Examples of anticipated services include a Topology Service to distribute topology information and a Connection Monitoring service. The Network Service Agents must support these services and functions in order to provide the integrated service envisioned.

Version 1.0 of the NSI supports only a Connection Service, this service allows connection-oriented circuits to be requested and managed.

## The NSI Service Plane

For the purposes of defining layering, this architecture assigns the NSI to a conceptual Service Plane. Here we define the service plane as incorporating participating NSAs and the associated NSI sessions between these NSAs. This is depicted in Figure 4.

In general, the NSI Service Plane relies on the capabilities of the control plane and/or management plane (not shown) to effect changes in the Transport Plane, where the control and management planes follow conventional definitions. The transport resources and the physical instance of the Connection reside on the Transport Plane.

Figure : Transport Plane and Service Plane

The NSI Service Plane operates on an abstracted representation of the transport plane resources. In this abstracted view, transport resources within a particular Network are treated as a single opaque object under the control of a Provider NSA. Representation of transport resources is described further in the paragraph 4.

Considering the example of the NSI Connection Service, the service managed on the NSI Service Plane is the ability to create a Connection, with a set of simple, well defined manipulation rules that operate within a network and are defined by the NSI. These abstractions are exchanged and processed in the NSI Service Plane. The NSI Service Plane contains information about owners, permissions, performance parameters, schedule, and Path information.

Connection should be thought of as an NSI Service Plane construct that binds high level service planning results (such as path selection, authorization, reservation, and scheduling) to the low level configuration and management information necessary to provision the transport physical instances that make up path in the transport plane.

## Hierarchical communications model and federation

The NSI protocol is intended to allow services to be delivered across a chain of multiple participating Networks. To facilitate this, the forwarding of NSI messages is designed to be flexible. This section describes the communications models supported for NSI message handling.

The NSI message handling model is designed to support both the tree and chain model of path-finding as described later in the section on the Connection Service. It is important to note that this means that the NSA communications path is independent of any routing on of connections on the transport plane.

No assumptions are made about the reachability of participating NSAs, an NSA may be directly reachable or reachable only via a gateway NSA. For instance, an arbitrary set of Networks may band together under NSI rules and peer exclusively with a single parent “Federating NSA”. The parent federation NSA may have no transport resources of its own – just those resources under management of the children NSAs. Service requests will flow along the trusted sessions hierarchically among NSAs, even if the transport connection itself takes a different route through the transport infrastructure.

Figure : NSA variants

Figure 6a: Hierarchical communications model, shows an example of the hierarchical model of communications. In the case of a federation of NSA, the federating NSA becomes a communications parent for its child NSAs. An example of this is shown where NSA A communicates with NSA D via NSA B.

Figure a: Hierarchical communications model

It should be noted that in the case of highly meshed NSAs, a destination NSA may be reachable by more than one path. An example of this is shown in the next figure. v1.0 of the NSI places no constraints on how to forward NSI messages. For example NSI A wishing to control a resource at E may choose to do this via intermediate NSAs B or C.

\*\*\* This example will lead to a resource conflict it applied to a single service instance. I suggest that this architecture should be avoided by operators to prevent ambiguity. This is best done by sticking to a hierarchical architecture, i.e no meshing of NSI interfaces.\*\*\* Guy

Figure b: Complex communications model

# The NSI Protocol

## NSI Protocol Sessions

Network Services are delivered with the use of the NSI Protocol. This protocol defines the constructs, state machines, messages, and parameters associated with the NSI services model. An NSA, by definition, is an agent that implements the NSI Protocol. The details of this protocol are out of scope of this document; the NSI Protocol recommendation (GWD-R-XXX) provides a detailed definition of the NSI protocol.

The NSI Protocol requires a “trust relationship” between NSAs. These trust relationships mean that each NSA believes the other to be whom they claim to be (authentication) and that both NSAs are willing to accept service requests from the other and act to satisfy them (authorization). Further, there must be a comfort level that the messages have not been tampered with, and optionally that they have not been exposed to unauthorized/untrusted third parties.

## NSI messages

The NSI protocol is made up of datagram style NSI messages. Each message envelope contains sufficient information to route the message to the proper network service agent, followed by sufficient service specific information to associate the request to an appropriate service instance and to identify the service primitive. The message canbe broken into smaller sizes by the underlying layers for transport of the message. These NSI messages are handled completely by the provider and should return at a minimum either a confirmation or rejection of the request in its entirety.

An NSI Message must allow identification of the Network Service type. (eg Connection Service, Topology Service, etc). An NSI Message must allow identification of the Connection service version (v1.0, v2.0 etc). NSI Messages mustinclude a mechanism to ensure that ordering is maintained in a NSI Message Thread.  Note that the NSI message ordering just does not prevent asynchronous operation of the service protocol, it just ensures that the message sequence is preserved across the secure session.

An NSI Message mustinclude a mechanism to associate it with an NSI Message Thread to allow differentiation of message streams associated with simultaneous and asynchronous service functions occurring between pairs of NSAs

Authentication of the source of all messages may be done on a message by message basis or with an authenticated session.

## NSI Service Instances and Primitives

The base NSI protocol handler recognizes NSI messages between NSAs. The protocol examines each message received for its Service Identifier and forwards that message to the appropriate service specific agent.

While the service defines the full set of capabilities that are offered to requesters, the service instance defines one specific instance of the service.

Each NSI service defines a service instance and a set of service primitives that operate within the context of a service instance. This service instance is an independent, uniquely identifiable deliverable unit of the service. The NSI Connection Service refers to a particular connection as a service instance. A topology distribution service may define an instance to be a particular topology graph, or a topology transaction such as a full dump or incremental update.

In general, a service specific state machine allocated and associated with each service instance, and the service primitives drive the transitions of that state machine. A service primitive may require a sequence of messages or even its own state machine to affect an exchange of messages.

Service Instances are processed asynchronously with respect to other service instances. For example, one connection may transition from reserving, to scheduled, to in-service, to release at a vastly different speed than another connection established by the same service agent.

The network service instanced must be locally unique.  Should have a field large enough to hold policy based ids such as GLIF.

## NSI Service Definitions

The concept of Service Definitions is introduced to allow operators to formally identify and define the characteristics associated with each service offering.

The Service Definition consists of a set of attributes that formally and explicitly define the complete scope of a service offering. In particular, the NSI Connection Service uses the Service Definition as a baseline set of parameters to bound the scope of the service that will be offered to requesters.

The Service Definition specifies the set of service parameters that completely specify a service instance. For example, the Service Definition might identify “capacity”, “mtu\_size”, and “maximum\_frame\_loss\_rate” as three aspects of the service. The Service Definition also describes the *range* of allowed values for each service parameter, and a default value can be specified. In the context of the previous example, the range of allowed values for the “capacity” parameter may be 50 mbps to 10 gbps in increments of 150 mbps. Or the “mtu\_size” may be defined to be 1500 Bytes to 9000 Bytes with a default of 1500. The parameters in the Service Definition form a kind of template that the service request must fill in. I.e. A service request must fill in the template with an explicit value - or a default value taken from the service definition - for each parameter of the service it is requesting. A service request is fully specified when all parameters associated with that service have been determined either by explicit user specification or by implicit default values found in the Service Definition. This fuly specified request is then processed by the NSA and, if all service specifications can be satisfied, a service instance is created (and reserved).

The Service Definition is an integral component of the NSI architecture in that it is key to vetting service requests against the multi-domain service offerings encountered along a candidate path.

The Service Definition is a public document that can serve as a both a human readable guide to available service capabilities and a machine readable file that can be processed by automated agents in the NSI Architecture.

## Temporal aspects of NSI services

Services, in which resources are dynamically requested, reserved and provisioned, require temporal aspects to be understood and controlled. Each NSA must maintain its own real-time clock, and it is necessary for these clocks to be synchronized. NSAs may implement NTP or radio time and frequency services to achieve this synchronization.

When resources are sought by a requester NSA from a provider NSA, a service instance is created and an identifier is assigned to that service instance. Then, according to the parameters of the request (i.e. its Service Definition), the provider NSA identifies and reserves a set of available resources which satisfy the request and associates them to the instance. The resources are provisioned and released at some point on the temporal axis. The time information and signaling are used to specify the time boundary of the requested connection in-service period. It is the responsibility of the Provider NSA to attempt to deliver the connection as close to the start and end times as it is able. It should be noted that this may have some uncertainty as typically the duration of the provisioning phase cannot be precisely predicted.

In the case of the Connection Service, advance reservation requests will specify the required resources and the provisioning start and end time. The request is processed by a scheduler, and the scheduler finds a set of resources available for the requested duration and allocates them to the request to create a reservation. If the scheduler cannot find an available set of resources which satisfies the request, the request is denied. This scheduling process is part of path finding in the connection service. A reservation database (i.e. calendar) should be maintained by the scheduler or resource managers, and referred and updated by the scheduler. The detail of advance reservation is covered in detail in paragraph 5.1.4.

## Trust and authentication in NSI

This section describes the approach taken to trust and authentication in the NSI protocol. The detailed mechanisms for providing trust and authentication are not defined here; the protocol definition describes these implementation details.

These trust relationships can exist in one of two modes: First, for high volume and/or persistent peering relationships, an authenticated, authorized, secure (encrypted) and reliable session can be established between the NSAs. Traffic passed across such a session is known to be trusted and can proceed directly to the service handler. The second mode is to employ a more message based trust framework such as Web Services. This message based form is more appropriate for occasional messaging as might occur between an application agent and various provider NSAs.

Two types of trust are identified, these are NSA-NSA and service-to-service, these are shown in Figure 8. In the NSA-NSA case, the NSI requestor and provider have a trust relationship. Using that trust relationship they provide integrity and privacy for communications between the each other. In the service-to-service case, the requestor and provider services share message sequences with each other. These have trust, based on trust between NSAs, which is used to provide message integrity between services.

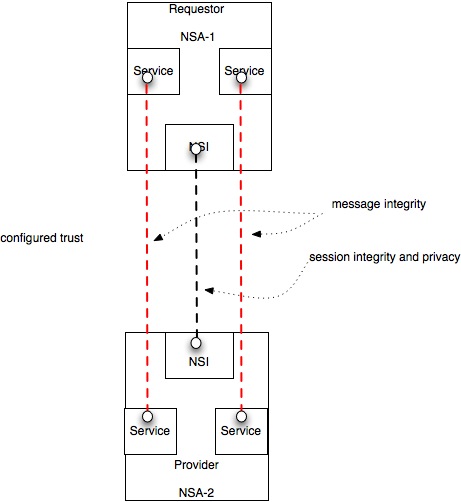


Figure : Agent and service trust

A service request may include attributes a request that identify a requestor or a department that is responsible for the request. Figure 9 shows that the authorizer of an attribute must be known to the service that uses an attribute to determine policy. An example of this case is where a user requests is authorized and assigned an individual id and a group id. These id’s are used by the resource provider to determine which resource, if any, should be allocated to a request.

In most current cases attributes are assumed to be from the adjacent NSA so trust between NSAs can be used to ensure integrity of attributes. This is supported in version 1 of NSA.

To support this in all cases requires preconfigured trust between provider and user of attributes and a mechanism to carry secure attributes through a chain of NSAs

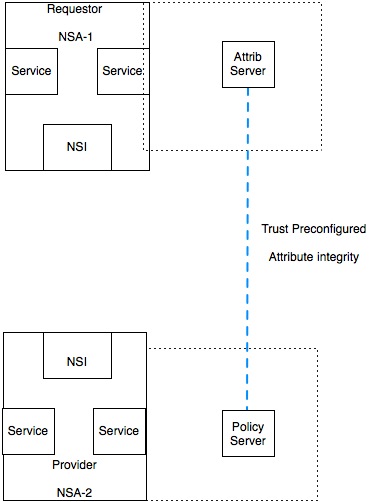


Figure : Attribute trust

## Error handling NSI

The NSI architecture is a distributed, multi-agent architecture which is designed to handle error cases in such a way as to ensure predictable and deterministic behavior. This section describes the basis of error handling for this protocol.

Network errors can be broadly categorized as soft or hard failures. A soft failure occurs when two NSA agents lose communication with each other or the NRM loses communications with the transport resources. A hard failure occurs if the NSA software crashes or the underlying server hardware fails. Such failures may cause a loss of information about state information in transit or not committed.

The NSI protocol incorporates mechanisms to recover to a consistent and predictable state after detecting an anomaly. The following architectural principles guide error handling and recovery:

Handling of failures should result in deterministic behavior that is user centric and oriented towards the service model, for example: A failure in the Service Plane should not affect connections that are provisioned and active in the Transport Plane. A failure in the Service Plane should not result in an incomplete service.

Recovery of Transport and/or Service Plane should not be reliant on external entities or mechanisms, for example: An NSA recovering from a hard failure error condition will not depend on peer NSAs to reconstruct its state.

It is important to note that explicit user negotiations may take precedence over these default behaviors. For example, a user may request that if any NSA fails, all the NSAs handling the same service instance should tear down the Connection Service in the Transport Plane.

Failures in the Service Plane can result in NSA state faults. Examples of Service Plane errors include: losing communication with an NSA, losing communication with the transport network, corruption/crash in the platform etc. These errors may result in service disruptions until these states can be synchronized, hence the NSI protocol and state machine design should account for such scenarios.

Figure : Service Plane Failure Sensitive Sections

Failures in the Service Plane during Reservation, Provisioning, Teardown, and Release phases can cause problems for the operation of the NSI. Since a reliable communications is assumed between NSAs, failure in communication with a remote Provider NSA or Requester NSA can be assumed to be a remote NSA failure. The following diagram illustrates local and remote failures.

Figure : Local/Remote Failures

Regardless of where the error originates, it is important that the NSA recover to a deterministic state. This means that both the user service state and the resource state should be consistent between NSAs.

The distributed model of servicing user requests using tree/chain model allows each NSA to assume the role of a requester or provider. When Service Plane failures occur, it is possible that an NSA will become entirely disconnected from other NSAs involved in a service instance. This scenario imposes a requirement on the NSA to have a linkage between its Requester and Provider Agent state machines to understand the impact of the failure on the service tree and recover from it. The state machines of Requester and Provider NSAs should be designed so the outcome of a distributed failure ends each state machine in a deterministic state.

## Transport failure awareness

The NSI needs to be aware of transport layer failures. …. How is this done?

# Representing network resources

## Describing network topologies

A Network topology is an object-oriented representation of Network resources based on a network description language. The Network topology may be used to perform functions such as path-finding and resource reservation. Many languages or models have been proposed to describe networks, such as OGF NML or ITU-T G.805. These are designed to fully describe the components of a network.

The NSI defines a new topology description: the inter-Network topology this is concerned with describing the global interconnection of Networks, and the intra-Network topology concerned with the transport resources within the Network. The formal representation of traditional intra-Network topology is out-of-scope for the NSI. While the choice of representation of intra-Network topology is up to the operator, the inter-Network topology is described here.

The method by which the inter-Network topology is assembled out-of-scope for version 1.0 of the NSI. It is assumed that each NSA has access to this topology information, but no assumptions are made as to how this has been gathered. Later version of the NSI are planned to a topology exchange service.

Figure 12: Inter-Domain Topology

Figure 12 depicts an inter-Network abstracted topology with Networks interconnected at Service Termination Points (STPs).

NSI inter-Network topology supports the administrative grouping of Transport Plane resources into a single topology object called a Network. Networks interconnect with other Networks via interconnected physical ports. These are modeled as STPs in the service plane, it is the responsibility of the NSAs to define a valid mapping to relate STPs with physical ports.

By aggregating detailed transport topology into a single Network, or by grouping several Networks together to form a federated Network object, the global network topology may be reduced substantially. How such a federation is implemented and the resulting simplified inter- Network topology map is out of scope for at least this version of NSI. Successful implementation for a particular deployment will allow Pathfinders to inexpensively compute coarse grained path(s) between any pair of networks. Each NSA along the candidate path is then consulted to reserve and confirm the resources. (For most of the NSI discussion (certainly for V1.0 inter-domain discussion) we assume each Network has a transfer function that is a simple non-blocking any-port-to-any-port crossbar switching function. Any other path constraints are managed by the Resource Manager.)

It is important to note here that the arrangement of NSI interfaces between NSAs is *not* necessarily congruent with the connectivity between Networks.

From a global perspective, hiding detailed transport topology within an opaque Network object substantially reduces the size and complexity of the topology information base. This has positive implications for coherence and convergence, for dynamic topology distribution, path finding efficiency, and for scalability in the global environment. It has the less desirable effect of reducing optimality – it becomes increasingly difficult to choose a resource efficient path. The trade off is an issue of pragmatism, and will be steered by best practices as the experience base improves.

## Service Termination Points

The NSI Architecture adopts a generalized notion of a Service Termination Point (STP). An STP names a topological location that acts as a concatenation point connecting two Connections. This is the junction between the ingress of a Connection in one Network and the egress of a Connection in the next Network.

A prerequisite for an STP is the existence of a physical connection between Networks. This preexisting capability (typically made up of a physical port on each network) can be advertised by each network to the NSA. Note that the choice about which resources to advertize is subject to local policy. Once advertised, these capabilities may be used by the pathfinding function of the NSA.

An STP capability can be a list of possible STP instances, or a more flexible representation like wildcard and constraints. For example, if there are 10 links (1, 2, 3, …, 10) and any two of these links can be aggregated, there are 90 possible STP instances (1-2, 1-3, …, 9-10). A wildcard like representation may be better than listing all of such STPs individually.

An STP can function as both an ingress point on one side and an egress point on the other. Two such connections that share a single STP in this way are said to be concatenated. These two concatenated connections then appear to the user payload as a single end-to-end transport plane data-path. In this way an STP becomes intermediate transit-point of a path or connection, i.e a routing point through which the connection must pass.

STPs may be uni or bi-directional and are typically associated with Ports in the topology. In the uni-directional case, the STP functions either as an ingress point or an egress point, this is defined by the flow polarity of the associated Port, and which side of the junction is the user side, and which side is the network side.

Two adjacent networks agree on the STP capability between the two networks in advance. The STP capability is the information which describes all possible STPs which can be instantiated between them. Constraints are also included in the information.

Once the STP capabilities are advertised, the STP needs to be dynamically generated for an inter-network connection over the STP to be instantiated.

Once allocated, an STP may have properties such as a framing, bandwidth and a VLAN id. Some of these properties may reflect the requirements specified in the service definition. Labeling (cf. fiber id, wavelength, VLAN id) and aggregation (cf. an STP may be provisioned by aggregating multiple switch ports) can be modeled as a property of STP.

To instantiate an inter-network connection, the requester NSA requests STPs with the same representation to the adjacent networks. Each network then keeps a calendar of reservations.

Figure : STPs

Assume there are two networks, X and Y, and possible STP instances between X and Y are: v1, v2, v3 and I1, I5, I9. This information is advertised by X and Y in advance. To request an inter-network connection, the requester NSA requests:

• To network X: connection between somewhere to to\_Y:v1

• To network Y: connection between to\_X:v1 to somewhere.

Each network looks up its own calendar and check availability of v1. The availability may be different by the networks.

When the above intra-network connections are instantiated, they are inherently connected.

An STP is a symbolic reference, i.e. it is comprised of a parsable alphanumeric string containing two components: 1) a Network identifier string in the higher order portion, and 2) a local STP identifier in the lower order portion. An STP must always resolve to a specific topological Port object as defined in the NSI topology. It is an implementation decision as to how to map that NSI Port identification to the corresponding resource in the physical transport plane.

It is important to note that the NSI inter-network topology model is composed of Networks interconnected by STPs. It should be noted that this topology is neither a standard nor does it imply that an NSI implementation must adopt specifically any particular schema for its database in the code.

# The NSI Services

The NSI Protocol is designed to mediate trust sessions and deliver messages between many NSA based services. NSI v1.0 stipulates a single NSI Service: The NSI Connection Service, this is described below.

## NSI Connection Service

### Connection service concepts

The NSI Connection Service is the Network Service that manages Connections.

There is a growing requirement to integrate customized networks resources into existing grid resources pools and applications. The ability to manage network connections effectively and easily by the grid community is perhaps the most pressing driver for the OGF NSI specification effort. This particular requirement has had a significant influence on design decisions and prioritization of capabilities incorporated into the NSI Architecture.

The NSI Connection Service reserves, schedules, and instantiates Connection instances. Single channel, point-to-point connections are supported in v1.0. These may be flagged as either uni-directional or bidirectional connections.

The NSI Connection Service is designed to create Connections that support a high capacity, highly asymmetric data flow such as occur in large file transfers or real-time streaming of digital media content.

Figure : Anatomy of a Connection

As illustrated in Figure 14, the Connection consists of three basic components: an ingress point where user data enters the connection, a transport section that carries the data across the network, and an egress point where user data exits the connection. The network components that present the user data to the ingress point or carry the user data away from the egress point are the access sections. The network infrastructure that carries user data from the ingress point through the network to the egress point is the transport section. The end of the transport section or the junction between the transport section and the access section is called the Service Termination Point (STP).

The user data (the “payload data”) is carried across each section of the network inside a “framing protocol”. The framing protocol, provides the necessary timing, control, and data integrity functions required to move the payload from node to node through the network. It is important to distinguish between a) the access framing protocols, b) the transport framing protocols, and c) the user payload data carried inside each of these protocols.

* *It is the user payload data stream that must be preserved from ingress to egress in a Connection.*

The transport framing can be any framing protocol as long as the end-to-end preservation requirement is honored. In fact, the only constraints on the transport framing is that the transport section be able to adapt the ingress payload data to each of the successive transport protocols that may be used along the path and ultimately be able to adapt the user payload to the egress framing at the egress point. While specification of the connection end points, access framing, and other parameters associated with a connection are defined by the connection requester (or implicitly by the service definition), the choice of the transport protocol and associated transport path parameters are explicitly delegated to the network service provider in order to allow the provider the greatest latitude in finding a valid, available, and optimal path for the connection request. This is another example of how abstraction separates the user perception of a “connection” from that of the provider. While this abstraction simplifies the service concept, the NSI Architecture allows the omniscient requester to participate in these connection planning decisions.

### Service Definitions for Connection Services

The Service Definition formally describes each aspect of a service. Indeed, a ”service” only exists if it is formally defined in some manner. Within the NSI Architecture, each network presents one or more transport services at its inter-domain edge points. Each service is defined in a document called the Service Definition (SD). With respect to the NSI Architecture, this document is a machine readable format that allows the NSAs to access and validate service requests against the services offered by the associated network. In practice, the service definition should also be available in a human readable form so that users and applications developers have guidance as to what network transport capabilities are available.

The Service Definition has its roots and most immediate application in the definition of the NSI Connection Service offering(s), and for NSI v1.0 that is the sole purpose for which the Service Definition is adopted. (Note that the notion of formal service specifications is still a widely researched topic with new application to emerging network services from Connections to Topology to Monitoring. Further exploration and refinement of this helpful concept within the NSI Architecture will be a continuing effort in NSI futures.)

Each service offering has a service definition. The SD consists of a list of attributes or parameters that identify each characteristic of the service. For each service parameter the SD specifies the range of valid settings for that parameter. For instance, an “Ethernet Transport Service” might define a service parameter called “Capacity” that defines a range of allowable service capacities between 1 Mbps and 10 Gbps. Another Parameter, say “Access\_Framing”, may specify a set of framing protocols that the user may request for ingress or egress. In this case, the Access\_Framing might be “802.1”, “802.1Q”, and “802.1ad”, default = “802.1”, indicating that ethernet frames will be carried that conform to one of three IEEE standards. A default may also be specified in order to fully specify a service request where the user does not specify a value for a particular parameter, or where the requester may wish to allow greater degree of freedom to the NSA in selecting a path

If a service request describes a service instance that lies within the bounds of the set of defined service parameters, then it forms a “valid” request. Each provider NSA along a candidate path must compare the service request to the local Service Definition in order to insure each specified parameter lies within the range of valid settings that the service offering can support. If a service parameter is not present in the service request, then the provider NSA should “fill in the blanks” from default values in the Service Definition. As the request is processed down the NSA service tree, default values adopted in one transit network may implicitly constrain the request in downstream networks. Therefore, in general, each NSA should use default values that provide the greatest leeway to the pathfinder in satisfying the request both within the local network and in external downstream networks. Ultimately, the parameters explicitly specified by a requester agent must be honored. All other parameters must simply be compatible.

When the NSAs complete the reservation process for a service request, the reservation confirmation indicates that the network (*all* networks along the path) have agreed to provide the requested level of service. This constitutes a defacto service level agreement upon which the requesting agent should be able to depend.

A key architectural aspect of the Service Definition paradigm is that each service instance has a clearly defined profile that constitutes when that service instance is performing to specifications, and when it is not. This performance can be measured end to end by user agents. If a service instance does not provide the service level specified in the request, and confirmed by the network, then it is in violation of the service agreement. It is beyond the scope of NSA Architecture to discuss what the implications should be for a SLA violation. However, when a confirmed service instance does not meet its agreed upon performance levels, it should be detectable by both the Requesting Agent and the Provider Agent in some manner. The NSI architecture does not require such monitoring by either agent – the architecture will function without it. But Best Common Practice would council that a violation is indicative of a failure somewhere in the chain, and should raise flags to notify operations personnel or the end applications so that remedial action can be initiated.

The service definition provides a publicly available description of the service, and should be made available in a native language document that the users can reference in developing or configuring their applications. The users (application developers) should consult this service definition in order to understand what service capabilities are available to them within a given service offering.

The Service Definition also plays an important role long before a service request is received. The service definition can be used as design objectives during the engineering phase of deploying a new service. The network engineering team can look at the SD in order to select hardware and software than can meet the technical and administrative requirements of the service. Further, two networks with similar services may compare and negotiate a common, or at least interoperable, service definitions. For instance, if network A offers capacity in 50 mbps increments, and network B offers it in 1 mbps increments, these networks will still be compatible – though certain requests may require more resources than are actually necessary.

It is important to stress one more aspect of a service definition; if a parameter is not identified within the service definition document, then the user can make no inference about its presence, absence, or value in the service. For instance, if a service definition has no jitter specifications, the user can make no predictions or assumptions about the jitter characterisitcs. And the network has made no commitments regarding jitter. Indeed, a request satisfied on Monday might have excellent jitter characteristics, and the exact same request submitted and satisfied on Tuesday might have horrid jitter characteristics. As long as the service constraints presented on both requests were met, these are – from a formal service perspective – properly performing and identical service instances.

The converse is also true. The network should be very careful about how it defines service parameters. For instance, an Ethernet service may define connection capacities in “bits per second” (bps). On its face, one might construe that a 1 Gbps connection would accept bits at 1 billion bits each second measured over any one second period. However, if this 1 Gbps connection is provisioned over a 10 Gbps network link, this interpretation would allow a 100 millisecond burst at 10 Gbps followed by a 900 millisecond quiescent period. Such a burst of 125 megaBytes can easily induce buffer overruns and packet discards on interfaces along the connection path. …And yet the user would have been perfectly within their performance profile. The implication here is that simple fixed capacity connections in asynchronous packet transport networks requires sophisticated and detailed planning in order to guarantee service capabilities. The service definition allows the networks to specify burst characteristics they can support for connections reserved across their infrastructure. These burst charateristics can be pro-actively requested by the user, or implicitly applied during pathfinding.

When a service request is returned to the requesting agent, the full template of service parameters should be returned containing the values assigned during pathfinding and reservation. This allows the requesting agent to adapt local processes appropriately.

### The Connection Service States

The states of a connection relate to the life cycle of the connection. In the NSI, a connection goes through five phases: Reserving, Scheduled, Provisioning, In-Service, Releasing.

First, a request is submitted to a provider thus beginning the life cycle of the connection within that provider Agent’s network. This first phase is called the “Reserving” phase. It includes path selection and resource reservation. In the NSI, path selection includes future scheduling as well as the performance and authorization checking. Once the scheduling phase is complete, the Requester NSA is notified and connection goes into a “Scheduled” state.

For automatically provisioned connections, when the service start time arrives, the connection goes into a “Provisioning” phase. For signaled connections, the “Provisioning” phase is initiated by a signal from the Requester NSA. Provisioning is the process where the connection is physically instantiated by configuring each device along the path to reflect the path plan developed and reserved in the Reservation phase.

Once provisioning is complete, the connection then moves into an “In-Service” state and the user are notified that the connection is ready for use. The In-Service phase is where user data is allowed to transit the connection.

When the connection is no longer needed (or the scheduled time expires) the connection is “Releasing”. The Releasing phase is where each network along the path is informed of the Release event and resources associated with the connection are released back to the available pool. Upon entering the releasing phase, the connection will no longer pass traffic. When the Release has completed, the connection object is deleted from the Service Plane.

Figure : Connection Lifecycle

### Connection reservation messages

Advance vs. immediate reservation

The connection request can have either a start-time and end-time, or a start-time and duration.

If the connection request includes a valid start-time and an end-time then the request is considered to be an advance reservation request. If the connection request has the start-time set to ‘asap’ and has a duration field rather than an end time field, the request is considered to be an immediate reservation request.

Automatic vs. explicit provisioning mode

Provisioning of a connection is achieved using a 1 phase commit mode. The transition from Scheduled state to Provisioning state can be either explicit (i.e. signaled by the requester NSA) or automatic (i.e initiated by the provider NSA). Each connection request will include a *flag* to indicate which of these two provisioning modes is to be used. It is not possible to mix these modes in a single connection.

When operating in explicit mode, it is the responsibility of the requestor NSA to signal the reservation to begin provisioning and to begin de-provisioning of the connection. These signals are known as the ProvisionRequest and CancelRequest.

Responses to connection requests

The provider NSA will send a response back to the Requester NSA once it has completed processing a connection request. This response will send back a pass/fail response and will confirm the state of the provider NSA.

When provisioning and de-provisioning have been completed it is necessary for the provider NSA to send a notification back to the requester NSA.

As shown in Figure 4, a requester NSA first sends a ReserveRequest message to a provider NSA. The provider NSA schedules resources, and notifies the requester whether it can deliver the request. If explicit provisioning is used, the requester must send a ProvisonRequest message to the provider NSA, and the NSA then provisions a service instance. If the automatic provisioning is used, the resources are provisioned at the start time. The requester may send a CancelRequest message. A CancelRequest message received before the provisioning has begun removes the reservation from the reservation database. A CancelRequest message received after the provisioning is completed releases the provisioned resources.



Figure : Service instance lifecycle.

The phases of the connection are as follows:

P1 Reserving phase.

P2 Scheduled phase.

P3 Provisioning phase.

P4 In-service phase.

P5 Releasing Phase.

### Connection reservation and timing parameters

For advance reservation with *automatic* provisioning, the start-time refers to the time at which the connection moves from provisioning state to in-service state. It is the responsibility of the provider NSA to make sure that this in-service start time is met. This may require beginning the provisioning process in advance of the start-time and will require some knowledge of the expected provisioning time.

For advance reservation with *explicit* provisioning, the start-time refers to the time at which the provider is able to accept a provision signal. It is now the requestor’s responsibility to advance the explicit signal to ensure good in-service time. The reservation end-time refers to the time at which the reservation is removed. (If the user has not yet sent a CancelRequest signal the connection is de-provisioned first)

“Infinite” can be used as an end time. In this case, resources are reserved forever (i.e. until a release request is received or may be overwritten by policy limits). Note that the resource reserved forever cannot be used for other requests of later time.

It takes some time to process a request. Possible maximum time required to process a request and make resources ready for provisioning is called “guard time”. Each provider NSA must define its guard time and provide it to requester NSAs. A requester NSA should not request a reservation which start time is smaller (earlier) than (current time + guard time). Time required for message delivery should also be taken into account.

A flag is included to indicate whether a start-time later than specified guard is allowed. When allowed, these start times are treated as ‘now’ and warning may be sent back to the Requester NSA. Otherwise, if a provider NSA receives a reservation request which start time is before (current time + guard time), it simply denies the request (the start time is a constraint in the path finding process). Note also, that if explicit provisioning is used, the processing of a ProvisionRequest message will take some time to complete.

This system is designed to be compatible with systems based on 2PC. In a 2PC system, an additional phase exists between P2 and P3. This phase is the ‘commit’ phase. The commit phase allows the originating requestor NSA to collect reservation confirmations from child NSAs. The originating requester then sends out a commit request once all confirmations have been received. The purpose is to prevent provisioning beginning on any networks before all participating networks have confirmed their reservation. This prevents partially provisioned connections being created.

This operation can be replicated in the 1PC system defined here with the use of explicit provisioning mode. In this case the original requester may wait for all child NSAs to confirm their reservation before issuing a ProvisionRequest message. This in effect combines the commit and provision requests of the 2PC method into a single message.

### The Path Object

The “Path Object” (or Path) describes a route through the topology. When present in a Connection Request, the Path specifies an ordered set of Service Termination Points (STPs) that the connection must transit, and in the order the connection must transit them. Within a Connection Request, the Path Object, at a minimum, must specify the ingress and egress STPs for the Connection. Additional intermediate transit points may be included in the Path, and when present, they are considered a required constraint on the Connection’s route and must be honored.

A Path Object associated with a confirmed Connection contains, or references, a significant amount of information regarding the user, the source or destination of flows, the global topology, and internal detail of specific networks, etc. This Path information may pose a security or privacy issue to the user or the involved networks, or may just be considered proprietary information. Within the NSI, access to such information is considered a policy decision of each agent involved. Therefore, Path information is available to external agents via an authorized Query() primitive to the Connection Service.

The provider NSA is responsible for maintaining, among other things, a Path describing the fully specified path for any Connection reserved across its network. In order to protect the PO, the provider NSA must store the Path locally and return a redacted Path containing a list of STPs, and/or Named Path, specified in order, according to its internal authorization policies.

Since Connection Requests submitted to other NSAs may return a Path identifier rather than a Path Object itself, there must be means for distinguishing the two and a clear understanding of how a path object fits into the path algebra. Since the Connection Request segmentation processing is tree-like, it follows that the reserved Path Objects will also be tree-like. So a Path Object must be able to contain not just directly referenced STP Names, but must be able to contained Named POs as well. A Path Object then consists of a list of objects that either directly or indirectly resolve to topological points. For named POs, the NSA that owns the Named PO must also maintain authorization association(s) for the PO.

### Tree and Chain Connection modes for inter-domain pathfinding

There are two levels of pathfinding related to the NSI architecture: the inter-domain pathfinding which is concerned with choosing a path across the global topology of Networks, and the intra-domain pathfinding concerned with the transport resources within the Network. NSI is concerned only with inter-Network pathfinding.

Inter-Network Connections extend across multiple networks; they are constructed by concatenating connections built across the individual networks. This is done by choosing appropriate STPs such that the egress STP of one connection corresponds directly with the ingress STP of the successive connection.

The choice of which sequence of networks a path should follow is a pathfinding function. Two modes are described, tree and chain.

In the tree mode of pathfinding, once a set of STPs is chosen, the connection requests corresponding to each segment can be issued simultaneously and directly to the NSAs responsible for each of the corresponding networks. The process can be recursively implemented in for creating multi-level trees in the Service Plane. That is, several NSAs without direct control over the NRM/networks can be deployed in a hierarchical architecture with one or more levels.

The Tree model processing computes a course grained inter-domain path first. It uses that network path vector to decompose the connection request into several concatenated connection segments. A benefit of the tree model is that it enables the NSA to reserve the segments in parallel via direct interaction with the respective networks. The disadvantage of this approach is that it is not known in advance whether each Network has the resources available internally to reach the next Network chosen by the inter-Network pathfinder.

Figure : Tree pathfinding mode.

Alternatively, if the local NSA does not have sufficient topology information or authorization credentials to identify and interact directly with all the downstream networks, the local NSA can simply choose a neighbor network as the next hop, and using the interconnect STP as the ingress point, forward a request to that next hop NSA for handling. This conventional hop-by-hop approach is called the Chain model.

Chain style processing reserves resources sequentially beginning at the source STP and working hop by hop successively through each downstream network to the destination. The path computation requires only a simple next hop reachability calculation (though more sophisticated path finders can be implemented), and no downstream resources are reserved until the upstream prefix path has been confirmed. It is highly distributed, scales well and is robust. But it does hide or delegates much network provisioning decision to [unknown] downstream agents.

Figure 16: Chain pathfinding mode.

In both the tree style processing and the chain style processing, the end-to-end connection cannot be confirmed until all of the constituent connection segments have been successfully reserved and confirmed. Which model will be more effective is unclear at this time and will likely be directly related to complexity of topology distribution and path analysis, robustness, authorization schemes, request volume, network diameter, utilization density, cost, ease of use, and reach (to name just a few aspects.)

Both the Tree and Chain model reduce pathfinding to a constraint-based search over a topology to build a k-preferred path tree. Both can accept requester guidance in path selection through the inclusion of intermediate transit points in the connection request (discussed further under Path Objects). The method, tree or chain, used to process a request is made exclusively in the requester NSA. The requesting agent implements a Tree model process by submitting individual requests for each connection segment. These individual segment requests are processed asynchronously and in parallel. The requesting agent implements a Chain model by allocating a path through the local network, and then forwarding the request to a neighbor domain to resolve the remaining downstream portion of the connection.

## Future Services

The NSI defines a framework that will allow future Network Services to be added. The framework is defined in such a way that allows each Network Service to be independent.

# Contributors

Jerry Sobieski, NORDUNET

John Vollbrecht, Internet2

Guy Roberts, DANTE

Inder Monga, ESnet

Tomohiro Kudoh, AIST

Joan A. García-Espín

# Glossary

Connection

A Connection is a conduit that transparently moves user information across a Network from an ingress point to an egress point. A Connection has a set of properties (for instance, capacity, or authorization, or start time).  These properties, and their allowed range of values, are defined by a service definition. A Connection instance is a particular Connection, identified by a Connection Identifier.

Connection Identifier

A Connection Identifier is a label unique to an NSI interface which can be used to identify a Connection for the purposes of request, instantiation and management.

Connection Service

A Connection Service is a service that allows a Requester NSA to request and manage a Connection from a Provider NSA

Control and Management Planes

The Control Plane and/or Management Plane are not defined in this document, but follow common usage.

Network  
A Network includes all of the transport resources that are managed by a single NSA.

Network Resource Manager (NRM)

The Network Resource Manager owns a particular set of transport resources and has ultimate responsibility for authorizing and managing the use of these resources.

Network Service

A Network Service is an abstract notion that must be implemented by a concrete network service agent (NSA). The Network Service is the service characterized by the set of functionality that is provided in an NSA.

Network Service Agent (NSA)

The Network Service Agent is a concrete piece of software that sends and receives NSI Messages.  The NSA includes a set of capabilities that allow Network Services to be delivered.

Network Service Interface (NSI)

The NSI is the interface between Requester NSAs and Provider NSAs.  The NSI defines a set of interactions or transactions between these NSAs to realize a Network Service.

NSI Message

A NSI Message is a structured unit of data sent between a Requester NSA and a Provider NSA.

Path

A Path is an ordered list of Routing Objects.

Requester/ Provider NSA

An NSA acts in one of two possible roles relative to a particular instance of an NSI.  When an NSA requests a service, it is called a Requester NSA. When an NSA realizes a service, it is called a Provider NSA.

Routing Object

A Routing Object may include the following transport resources.

Service Definition

The Service Definition is the set of attributes associated with connection services (for instance, capacity, or authorization, or start time) and a range of allowed values for these attributes. Each Connection has an associated Service Definition instance.

Service Plane

The Service Plane is a plane in which services are requested and managed; these services include the Network Service. The Service Plane contains set Network Service Agents communicating using Network Service Interfaces.

Topology Rename

The Topology resides in the Service Plane.  The Topology describes both the physical resources and their interconnection as well as the non-physical groupings of various components

Transport Plane

The Transport Plane contains is the set of physical resources that transport user data through the network.  The Transport Plane forms the substrate over which Connections are allocated and provisioned.

# Intellectual Property Statement

The OGF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the OGF Secretariat.

The OGF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this recommendation. Please address the information to the OGF Executive Director.

# Disclaimer

This document and the information contained herein is provided on an “As Is” basis and the OGF disclaims all warranties, express or implied, including but not limited to any warranty that the use of the information herein will not infringe any rights or any implied warranties of merchantability or fitness for a particular purpose.

# Full Copyright Notice

Copyright (C) Open Grid Forum (2008-2010). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the OGF or other organizations, except as needed for the purpose of developing Grid Recommendations in which case the procedures for copyrights defined in the OGF Document process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the OGF or its successors or assignees.

# References