

# Learning Registry Technical Specification V 0.10.0

V 0.10.0 Archive -- Stable.

[Learning Registry Technical Specification V 0.10.0](#)

[Technical Specification Introduction](#)

[Learning Registry Overview](#)

[Specification License](#)

[Notation](#)

[Conformance](#)

[Technical Specification Overview](#)

[Design Principles](#)

[Resource Distribution Network Model](#)

[Network Nodes and Node Services](#)

[Network Topology](#)

[Network Data Models](#)

[Network Node Description Data Model](#)

[Network Node Service Description Data Model](#)

[Network Node Connectivity Description Data Model](#)

[Resource Distribution Network Description Data Model](#)

[Resource Distribution Network Policy Data Model](#)

[Network Community Description Data Model](#)

[Network Description](#)

[Resource Data Data Models](#)

[Resource Data Description Data Model](#)

[Metadata Formats](#)

[Paradata Formats](#)

[Resource Data](#)

[Trust and Identity](#)

[Security and Information Assurance](#)

[Services and APIs](#)

[Resource Data Distribution Service](#)

[Resource Data Publish Services](#)

[Basic Publish Service](#)

[SWORD Publish Service](#)

[Resource Data Access Services](#)

[Basic Obtain Service](#)

[Basic Harvest Service](#)

[Basic Query Service](#)

[SRU Service](#)

[SiteMap Service](#)

[Atom Pub Service](#)

- [Administrative Services](#)
  - [Network Node Status Service](#)
  - [Network Node Description Service](#)
  - [Network Node Services Service](#)
  - [Resource Distribution Network Policy Service](#)
- [Broker Services](#)
- [Common Data Model and API Attributes and Behaviors](#)
  - [Data Model Attributes](#)
    - [Identifiers](#)
    - [Strings and Internationalization](#)
    - [Time and Date](#)
  - [API Attributes and Behaviors](#)
- [Operations](#)
  - [Networks and Communities](#)
    - [Building the Network Description](#)
    - [Maintaining Networks and Communities](#)
    - [Network Discovery](#)
  - [Network Design](#)
  - [Resource Data Publication, Distribution and Access](#)
  - [Network Administration](#)
- [Glossary](#)
- [References](#)
- [Change Log](#)
- [Working Notes and Placeholder Text](#)

## Technical Specification Introduction

This document provides the technical specification for the Learning Registry. It specifies the network model, data models and APIs. It does not specify policy-based operational procedures or the instantiation of the Learning Registry. While targeted at a document-oriented infrastructure, the specification itself is independent of any particular toolset. The document is currently a work in progress; its structure, organization and content are subject to change. All information is in this single monolithic document.

## Learning Registry Overview

The Learning Registry [<http://learningregistry.org>] aims to make “learning resources easier to **find**, easier to **access** and easier to **integrate** into learning environments *wherever* they are stored -- around the country and the world.” It defines a learning resource distribution network model and a set of open APIs and open interoperability standards to provide three fundamental, enabling capabilities:

1. a lightweight mechanism to publish (push) learning resources (or metadata or paradata describing the resources) into a learning resource distribution network, independent of format or data type (e.g., resource, metadata or paradata);
2. the ability for anyone to consume the published data and then, in turn, to publish additional

feedback about the resources' use into the network (e.g., additional paradata), amplifying the overall knowledge about the resources;

3. a high-latency, loosely connected network of master-master synchronizing brokers distributing resources, metadata and paradata.

There is no central control, central registries or central repositories in the core resource distribution network. Published data can eventually flow to all nodes in the network. The network aims to be self assembling. Edge services can connect to any distribution node to find out what resources (and resource sources) are in the network, what's changed, what's being used, etc. Organizations may build consumer-facing, value-added services at the edge nodes to enable using, finding, sharing, and amplifying the resources, metadata and paradata for user communities. The Learning Registry provides *social networking for metadata* (trusted social collaboration around learning resources), enabling a *learning layer* on the social web.

## Specification License

This specification is being developed under the Open Web Foundation [Contributor License Agreement - Contributor Copyright Grant \(CLA 0.9\)](#). The intent is that the final specification will be released under the [Open Web Foundation Agreement \(OWFa 0.9\)](#). Later versions may apply.

Your use of this Specification may be subject to other third party rights. THIS SPECIFICATION IS PROVIDED "AS IS." The contributors expressly disclaim any warranties (express, implied, or otherwise), including implied warranties of merchantability, non-infringement, fitness for a particular purpose, or title, related to the Specification. The entire risk as to implementing or otherwise using the Specification is assumed by the Specification implementer and user. IN NO EVENT WILL ANY PARTY BE LIABLE TO ANY OTHER PARTY FOR LOST PROFITS OR ANY FORM OF INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY CHARACTER FROM ANY CAUSES OF ACTION OF ANY KIND WITH RESPECT TO THIS SPECIFICATION OR ITS GOVERNING AGREEMENT, WHETHER BASED ON BREACH OF CONTRACT, TORT (INCLUDING NEGLIGENCE), OR OTHERWISE, AND WHETHER OR NOT THE OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

## Notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC 2119](#).

The vocabulary of terms used in describing the Learning Registry are listed in the [Glossary](#). Specific terms are set in **bold** when introduced in context.

Data models are described in a JSON-like notation. It follows JSON syntax, but instead of providing the value for a name, the data model defines the JavaScript data type of the named element. A description of the element, further restrictions on the value space (e.g., if a string is a URL) and if the element is

optional or required is described in a comment. This model is used for convenience in early prototyping. A future version of the specification may describe the data models and their implementation binding independently.

## Conformance

There is no overall conformance statement for this specification. The Learning Registry Test Suite (link TBD) MAY be used to test an implementation. However, passing the Test Suite does not imply conformance to this specification. There is no defined “reference implementation” (by definition when there is a conflict between this specification and the reference implementation, the reference implementation is considered to be authoritative).

An organization MAY place tighter requirements on an implementation than those stated, e.g., it MAY constrain a MAY, SHOULD or OPTIONAL clause to MUST, SHALL or REQUIRED. It MAY NOT relax any constraint.

## Technical Specification Overview

The specification is split into several parts:

- **Network:** The description of the resource distribution network and its parts. A fixed multi-level structure of network parts is used to support distributing content and to provide policy-based security and operations.
- **Data Models:** The data models that are used to describe the network and learning resources data. Data models are document oriented.
- **APIs:** The APIs used to publish and consume data and those used to operate the network. The APIs are designed to abstract the logical behaviors of the Learning Registry from any particular implementation tools.
- **Operations:** Operational procedures that apply to any implementation.

## Design Principles

The learning registry design and technical specification is based on several key principles:

- **Decentralized:** There are no centralized registries or repositories or central data stores. Thus all core data is replicated across the network.
- **Redundant:** There is no single point of failure in the design (an implementation may have single points of failure).
- **Abstracted:** Abstraction is critical to layering capabilities, e.g., network content replication is content type agnostic.
- **Minimal:** Specify only what is required. Features that are community specific or can be layered on top of the core design are excluded from the specification although essential elements needed to support such modeling are included.
- **Generic:** Prefer approaches, models, standards, etc., that have wide uptake beyond just the learning technology and digital repository space.
- **Secure:** Security is by design, e.g., default values lock down an implementation and must be explicitly overridden even to do common operations.

- **Trusted:** Data and operations need to be authentic and trusted
- **Document Oriented:** The design is targeted at a document-oriented system for implementation using document-oriented databases.
- **RESTful:** APIs are RESTful, and use [CoolURIs](#) to return different data representations.
- **Scalable:** The design needs to seamlessly scale and perform at scale.
- **Extensible and Enabling:** The design is meant to enable new capabilities. Unless explicitly restricted (usually to satisfy security requirements) anything in the design is extensible.
- **Web 2.0 Friendly:** The design is based on current, widely implemented Web 2.0 technologies.

## Resource Distribution Network Model

The core of the Learning Registry is the network of loosely connected master-master synchronizing broker nodes distributing resources, metadata and paradata. Consumers and producers (edge node consumer and producer agents) connect to network nodes to inject information into the network or to extract information for external processing.

The network model is defined in terms of nodes, their services, the assembly of nodes into resource distribution networks, and the structuring of networks into communities. This two-tiered model of networks and communities supports security requirements for partitioning of resource data between different communities.

### Network Nodes and Node Services

A **node** is a server process with network connectivity to either other nodes or to edge services. Nodes process **resource data** (e.g., network messages about resources, metadata, paradata, etc.).

A node **SHALL** be described using the [network node data model](#). Only the owner of a node description **MAY** change the description. Certain attributes of the node description are immutable. *NB:* These are security constraints.

A node **MAY** provide five (5) different classes of services ([APIs, services protocols](#) and [data models](#) are detailed below) :

- **Publish Services:** Publish services are used by external agents to push (publish) resource data from the external agent into the distribution network. The data model for publication data is specified below. A node that provides publish services **MAY** support different publishing APIs, but all **SHALL** use the publication data model. The definitions of several common publishing APIs are specified [below](#).
- **Access Services:** Access services are used by external agents to discover, access and obtain (pull) resource data from the distribution network. A node that provides access services **MAY** support different access APIs. The definitions of several common access APIs are specified [below](#).
- **Distribution Services:** Distribution services are used to transfer, replicate and synchronize resource data from node X to node Y. X is the source node for distribution, Y is the destination node. To support security requirements, distribution is directed from X to Y; independent paired

services  $[X \rightarrow Y + Y \rightarrow X]$  are used for bi-directional synchronization. The distribution services SHALL use the distribution data model specified [below](#).

- **Broker Services:** Broker services operate at a node to augment, transform or process resource data held at that node to produce new or updated resource data for access or distribution. A node that provides broker services MAY support different broker processes. The definitions of several common broker processes are specified [below](#).
- **Administrative Services:** Administrative services are used to query a node to obtain its status or to trigger node administrative actions. The definitions of several common administrative services and APIs are specified [below](#).

## Network Topology

A **resource distribution network** is a group of one or more connected nodes, with each node providing node services. All nodes in a resource distribution network operate under the same policies. Multiple resource distribution networks MAY be established.

A resource distribution network SHALL be described using the [resource distribution network data model](#). Only the owner of a network description MAY change the description. Certain attributes of the resource distribution network description are immutable. *NB:* These are security constraints.

Two types of network nodes and connectivity within a network are defined:

- **Common Node:** A common node MAY provide any of the node service classes listed. If provided, the distribution services of a common node SHALL be limited to connecting to other nodes in the same network (the distribution service MAY connect to multiple destination nodes). A common node is denoted CN herein.
- **Gateway Node:** A gateway node SHALL provide a distribution service. A gateway node MAY connect to one or more common nodes within the same network. A gateway node SHALL connect to and provide resource distribution to a gateway node in another network. A gateway node MAY provide administrative services. A gateway node SHALL NOT provide publish, access or broker services. A gateway node is denoted GN herein. *NB:* As defined, a gateway is a 1:1 interconnect between two networks. 1:1 is used to simplify topology in support of security requirements; it is not a technical constraint. *NB:* Multiple gateway nodes between two networks are permitted.

A node SHALL participate in, and be subject to the policies of, only one resource distribution network.

A node SHALL not transition or be moved from one resource distribution network to another. A node MAY only be added to or removed from a distribution network. *NB:* This is a security constraint.

A gateway node X that participates in some network N1 SHALL connect to a gateway node Y that participates in some other network N2. A gateway node SHALL NOT connect to any other nodes in network N2 or to any node in any other network.

A **network community** is a collection of interconnected resource distribution networks. A community

MAY contain one or more resource distribution networks. A resource network SHALL be a member of only one community. Gateway nodes provide the connectivity between resources networks within a network community and MAY provide connectivity between networks in different communities. *NB*: A gateway node that provides an intra-community network gateway is undifferentiated from one that provides an inter-community network gateway.

A network community SHALL be described using the [network community data model](#). Only the owner of a network community description MAY change the description. Certain attributes of the network community description are immutable. *NB*: These are security constraints.

Two types of network communities are defined:

- **Social Community**: A social community provides connectivity to other social communities. A network within a social community MAY connect to another network within the same social community or with a network that belongs to a different social community.
- **Closed Community**: A closed community provides no connectivity outside of the community. A network within a closed community SHALL NOT connect with another network within a different community.

For example, the Learning Registry is a social community; other social communities may connect to the Learning Registry community. For security and testing, the Learning Registry Testbed is a closed community, i.e., it consists of different networks (multiple networks to enable testing gateway protocols) but the testbed cannot be connected to the social production community.

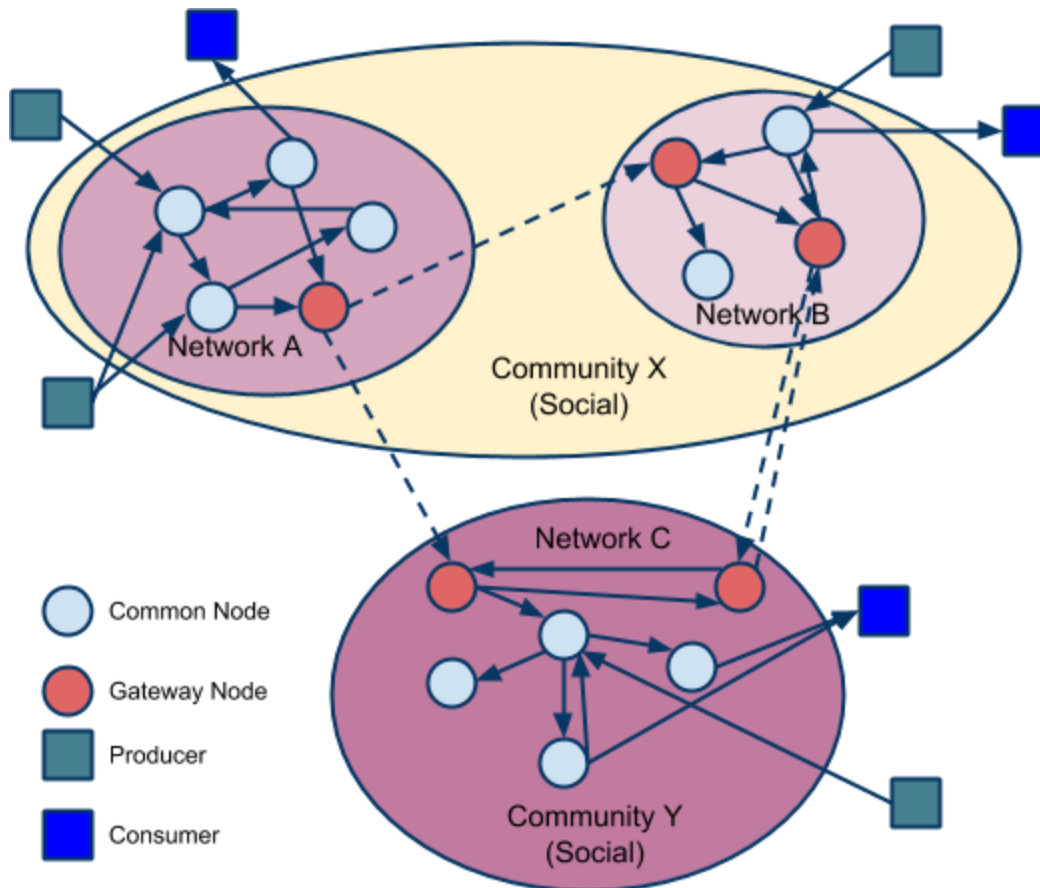
The Learning Registry community might consist of multiple networks and gateways. One network might be for uncurated OERs. A second network might be for curated OERs. And several others networks could be established for commercial resources (e.g., one per publisher). If the uncurated OER network has a gateway to the curated OER network, and there are gateways to each commercial networks, resource data can flow in only one direction, e.g., resource data for OERs into the commercial networks, but not the reverse.

A network SHALL not transition or be moved from one network community to another. A network MAY only be added to or removed from a network community. *NB*: This is a security constraint.

The resource network model provides nodes, collections of connected nodes within a network and the interconnection of networks in communities. The network model has this fixed hierarchy of components. Network communities connect to other communities using the same mechanism as networks that connect within a community.

Other network topology restrictions MAY be established by the policies of a particular network or community. This specification is intentionally minimal and does not define or limit other topologies, including degenerate topologies. *NB*: The model lets one design a network topology that might violate the policy and security constraints of a participating organization.

The diagram illustrates the network model. In the diagram there are three resource distribution networks (A, B, C) and two network communities (X and Y). Resource distribution network A connects to network B; both are part of the same community. Resource distribution network A also connects to network C and network C connects to network B. Resource distribution network C is in a different network community from A and B. If either network community X or Y was a closed community, the inter-network connection would not be permitted.



## Network Data Models

The description of a network is maintained in a set of documents that includes:

- Network Node documents:
  - The description of the node.
  - The description of the connectivity of the node within the network (including gateways).
  - The description of the services provided by the node.
- Resource Distribution Network documents:
  - The description of the resource distribution network that the node is a part of.
  - The description of the policies that govern the resource distribution network.
- Network Community documents:
  - The description of the network community that the node is a part of.



All data models MAY be extended with additional elements. The name of any extension element SHALL begin with the characters “X\_” designating an extension element. Any document that includes any element that is not in the defined data model or is not an extension element is non conforming and SHALL be rejected by any service.

All data models have a named attribute that is a “type” element (doc\_type). The data model description specifies the literal value for this element for all instances of each type of document.

All data models have a named attribute that is a “version” element (doc\_version). The data model description specifies the literal value for this element for all document instances.

All data models have a named attribute that indicates if the document instance is in use (active). Network data model document instances are never deleted; they transition from active to not active.

Additional constraints on attributes values are detailed [below](#).

## Network Node Description Data Model

The data model describing a node document. Once the data model has been instantiated for a node, the value of an immutable element SHALL NOT change. Other values MAY be changed only by the owner of the node document.

```
{
  "doc_type":          "node_description",    // the literal "node_description"
                                     // required, immutable
  "doc_version":       "0.10.0",             // the literal for the current version -- "0.10.0"
                                     // required, immutable
  "doc_scope":         "node",               // the literal "node"
                                     // required, immutable
  "active":            boolean,              // is the network node active
                                     // required, mutable from T to F only
  "node_id":           "string",             // id of the node, required
                                     // unique within scope of the LR
                                     // immutable
  "node_name":         "string",             // name of the node, optional
  "node_description":  "string",            // description of the node, optional
  "node_admin_url":    "string",            // URL of node admin, optional
  "network_id":        "string",            // id of the network that this node is a part of
                                     // recommended (required for gateway distribution)
                                     // immutable
  "community_id":      "string",            // id of the community that this node is a part of
                                     // recommended (required for gateway distribution)
                                     // immutable
  "gateway_node":      boolean,             // T if node is a gateway node
                                     // recommended, F if not present, immutable
  "open_connect_source":boolean,           // T if node is willing to be the source to
                                     // connect to any other node
}
```

```

// F if node connectivity is restricted
// recommended; F if not present
"open_connect_dest":  boolean, // T if node is willing to be the destination
                        // to connect to any other node
                        // F if node connectivity is restricted
                        // recommended; F if not present
"sync_frequency":    integer, // target time between synchronizations in minutes
                        // optional
"X_xxx":             ??????   // placeholder for extensibility, optional
}

```

*NB:* Synchronization/replication frequency is maintained on a per node basis. This allows each node to sync on a different frequency (versus a network wide sync frequency), but does not allow each connection to a node to sync on a different frequency, which might complicate scheduling.

## Network Node Service Description Data Model

The data model describing a service description document; one document per service available at a node. Once the data model has been instantiated for a service, the value of an immutable element **SHALL NOT** change. Other values **MAY** be changed only by the owner of the node document. *NB:* Ownership and control of the node description document and of the node service description document are vested in the same identity.

```

{
  "doc_type":          "service_description", // the literal "node_description"
                                     // required, immutable
  "doc_version":       "0.10.0",           // the literal for the current version -- "0.10.0"
                                     // required, immutable
  "doc_scope":         "node",              // the literal "node"
                                     // required, immutable
  "active":            boolean,             // is the service active
                                     // required, mutable from T to F only
  "service_id":        "string",           // id of the service, required
                                     // unique within scope of the LR
                                     // immutable
  "service_type":      "string",           // fixed vocabulary ["publish", "access",
                                     // "distribute", "broker", "administrative"]
                                     // required, immutable
  "service_name":      "string",           // name of the service, optional
  "service_description": "string",         // description of the service, optional
  "service_version":   "string",           // version number of the service description, required
                                     // version is local to the Learning Registry
                                     // not the version of some underlying spec for the service
  "service_endpoint":  "string",           // URL of service, required
  "service_auth":      "string",           // fixed vocabulary ["public, ...]
                                     // required, mutable from public to a stronger auth
  "service_data":      {},                // service-specific name-value pairs
                                     // optional
}

```

```
"X_xxx":          ????? // placeholder for extensibility, optional
}
```

**Open Question:** What is the auth vocabulary?

## Network Node Connectivity Description Data Model

The data model describing a node connectivity document; one document per connection at a node. Once the data model has been instantiated for a connection, the value of an immutable element **SHALL NOT** change. Other values **MAY** be changed only by the owner of the node document. *NB:* Ownership and control of the node description document and of the node connectivity description document are vested in the same identity.

```
{
  "doc_type":          "connection_description", // the literal "connection_description"
                                                    // required, immutable
  "doc_version":       "0.10.0", // the literal for the current version -- "0.10.0"
                                                    // required, immutable
  "doc_scope":         "node", // the literal "node"
                                                    // required, immutable
  "active":            boolean, // is the connection active
                                                    // required, mutable from T to F only
  "connection_id":     "string", // id of the connection, required
                                                    // unique within scope of the LR
                                                    // immutable
  "source_node_url":   "string", // URL of the source of the connection
                                                    // required, immutable
  "destination_node_url": "string", // URL of the destination of the connection
                                                    // required, immutable
  "gateway_connection": boolean, // T if this is a connection to a gateway node
                                                    // F for a common node
                                                    // recommended; F if not present (common node)
                                                    // immutable
  "X_xxx":             ????? // placeholder for extensibility, optional
}
```

*NB:* By policy, there **SHALL** be only one document with a **gateway\_connection** value of T per node.

*NB:* The source URL is not strictly needed. It is present to enable [building a network map](#).

*Working Assumption:* It is assumed that a vocabulary to describe additional types of connections is not needed.

## Resource Distribution Network Description Data Model

The data model describing a resource distribution network document. Once the data model has been instantiated for a network, the value of an immutable element **SHALL NOT** change. Other values **MAY** be changed only by the owner of the network description document.

```
{
  "doc_type":          "network_description", // the literal "network_description"
                                                    // required, immutable
```

```

"doc_version":      "0.10.0",      // the literal for the current version -- "0.10.0"
                                // required, immutable
"doc_scope":        "network",      // the literal "network"
                                // required, immutable
"active":           boolean,        // is the resource distribution network active
                                // required, mutable from T to F only
"network_id":        "string",      // id of the network, required
                                // unique within scope of the LR
                                // immutable
"network_name":      "string",      // name of the network, optional
"network_description": "string",    // description of the network, optional
"network_admin_url": "string",      // URL of network admin, optional
"community_id":      "string",      // id of the community that this node is a part of
                                // recommended
                                // immutable
"X_xxx":            ??????         // placeholder for extensibility, optional
}

```

## Resource Distribution Network Policy Data Model

The data model describing the policies of a resource distribution network document. Once the data model has been instantiated for a network, the value of an immutable element **SHALL NOT** change. Other values **MAY** be changed only by the owner of the network *description* document. *NB*: Ownership and control of the network description document and of the policy description document are vested in the same identity.

```

{
  "doc_type":        "policy_description", // the literal "policy_description"
                                // required, immutable
  "doc_version":      "0.10.0",          // the literal for the current version -- "0.10.0"
                                // required, immutable
  "doc_scope":        "network",          // the literal "network"
                                // required, immutable
  "active":           boolean,            // are the policies active
                                // required, mutable from T to F only
  "network_id":        "string",          // id of the network, required
                                // unique within scope of the LR
                                // immutable
  "policy_id":         "string",          // id of the policy description, required
                                // unique within scope of the LR
                                // immutable
  "policy_version":    "string",          // version identifier for the policy
  "TTL":              integer,            // minimum time to live for resource data in the network
                                // in days, required
  "policy_element_x":  ??????,           // placeholder for more policy elements
  "X_xxx":            ??????             // placeholder for extensibility, optional
}

```

*NB*: The list of policy elements is currently incomplete.

## Network Community Description Data Model

The data model describing a network community document. Once the data model has been instantiated for a community description, the value of an immutable element **SHALL NOT** change. Other values **MAY** be changed only by the owner of the network community description.

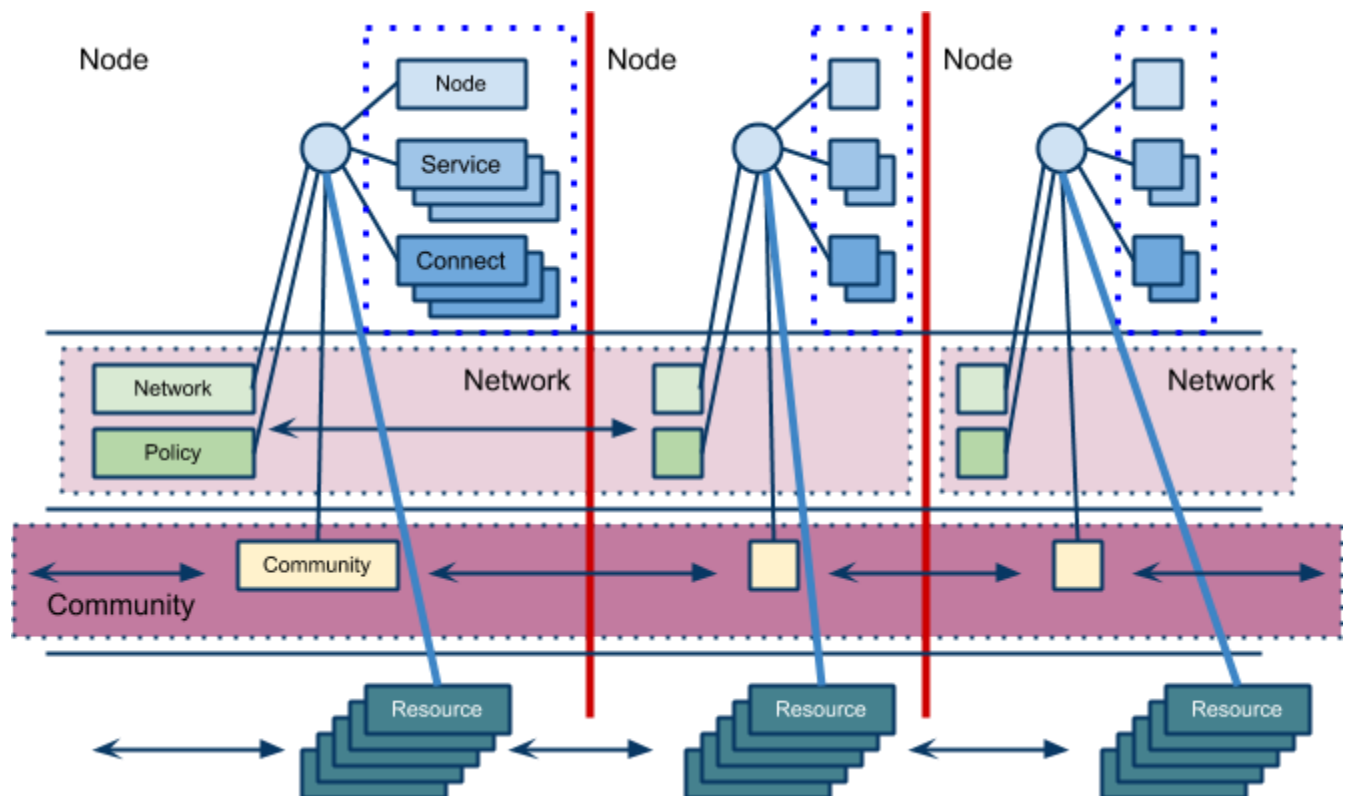
```
{
  "doc_type":          "community_description",      // the literal "community_description"
                                                           // required, immutable
  "doc_version":       "0.10.0",                    // the literal for the current version -- "0.10.0"
                                                           // required, immutable
  "doc_scope":         "community",                 // the literal "community"
                                                           // required, immutable
  "active":            boolean,                     // is the network community active
                                                           // required, mutable from T to F only
  "community_id":      "string",                    // id of the community, required
                                                           // unique within scope of the LR
                                                           // immutable
  "community_name":    "string",                    // name of the community, optional
  "community_description": "string",                // description of the community, optional
  "community_admin_url": "string",                  // URL of community admin, optional
  "social_community":  boolean,                     // T if the community is a social community
                                                           // F if the community is a closed community
                                                           // recommended; F if not present (closed community)
                                                           // immutable
  "X_xxx":             "?????",                    // placeholder for extensibility, optional
}
```

## Network Description

The network **SHALL** be described through a set of documents stored at each node in the network.

- Each node **SHALL** store one instance of the [network node description document](#). A document **SHALL** be unique per node.
- Each node **SHALL** store one instance of the [network node services document](#) for each service that it provides. A document **SHALL** be unique per node.
- Each node **SHALL** store one instance of the [network node connectivity document](#) for each connection to the node. A document **SHALL** be unique per node.
- Each node **SHALL** store one instance of the [resource distribution network description document](#). This document **SHALL** describe the network that the node is a part of. The contents of this document **SHALL** be identical for all nodes in the network.
- Each node **SHALL** store one instance of the [resource distribution network policy document](#). This document **SHALL** describe the policies of the network that the node is a part of. The contents of this document **SHALL** be identical for all nodes in the network.
- The node **SHALL** store one instance of the [network community description document](#). This document **SHALL** describe the community that the network is a part of. The contents of this document **SHALL** be identical for all nodes in the community.

The illustration shows the mapping of documents to nodes and the distribution and synchronization of documents within resource distribution networks and network communities.



## Resource Data Data Models

The resource distribution network and its nodes process and distribute **resource data** (e.g., network messages about resources, metadata, paradata, etc.). Producer edge nodes publish resource data to a node of the network; the resource distribution network moves it to other nodes, and consumer edge nodes pull resource data for external use from nodes.

Resource data is described in a different documents that includes:

- The description of the resource, metadata, paradata, etc.

All data models **MAY** be extended with additional elements. The name of any extension element **SHALL** begin with the characters “X\_” designating an extension element. Any document that includes any element that is not is the defined data model or is not an extension element is non conforming and **SHALL** be rejected by any service.

All data models have a named attribute that is a “type” element (`doc_type`). The data model description specifies the literal value for this element for all instances of each type of document.

All data models have a named attribute that is a “version” element (`doc_version`). The data model

description specifies the literal value for this element for all document instances.

Additional constraints on attributes values are detailed [below](#).

## Resource Data Description Data Model

The data model describing resources, metadata, paradata, etc., that is distributed by the resource distribution network. Once the data model has been instantiated the value of an immutable element SHALL NOT change. Other values MAY be changed only by the owner of the document.

```
{
  "doc_type":      "resource_data",      // the literal "resource_data"
                                     // required, immutable
  "doc_version":   "0.10.0",             // the literal for the current version -- "0.10.0"
                                     // required, immutable
  ...
}
```

## Metadata Formats

How to describe metadata and supported metadata formats within a resource data description will be provided in a future draft of the specification.

## Paradata Formats

How to describe paradata and supported paradata formats within a resource data description will be provided in a future draft of the specification.

## Resource Data

The resource data SHALL be maintained in a set of documents stored at each node in the network.

- Each node MAY store one or more instances of the [resource data description documents](#). All document instances stored at a node SHALL be unique. A document MAY be replicated at many nodes.

## Trust and Identity

The section on trust and identity will be provided in a future version or draft of the specification.

## Security and Information Assurance

The section on security and information assurance will be provided in a future version or draft of the specification.

All services SHOULD maintain a secure log of all service actions. Details of logging requirements will be provided in a future version or draft of the specification.

## Services and APIs

The services and their APIs provide the functionality that edge node producer and consumer agents use to

push resource data into the distribution network and to discover and pull resource data from the network. They also define how to distribute the resource data throughout a network and how to manage and observe resource distribution network behavior.

The defined list of services follows. Any non gateway node **MAY** provide any of these services. A node **MAY** provide additional services not specified herein.

*Open Question:* Do we need to tag other services (e.g., as X\_) to differentiate them from those in this specification. or can anyone provide any service?

Services and APIs are RESTful and bound to a particular node in the resource distribution network. Service descriptions include the API call (HTTP binding), the API arguments, the message payload (using the JSON-like notation), the service results (using JSON-like notation), an informative pseudo code description of a possible implementation, and the network node service description data model.

The [network node service description data model](#) provides a machine and human readable description of the service; an instance of the description document is stored at the node that provides the service.

Additional constraints on API attributes, HTTP bindings (headers, HTTP errors), error processing and behaviors are [described below](#).

## Resource Data Distribution Service

The resource data distribution service is used to distribute (synchronize or replicate) the resource data from one node to its connected nodes (unidirectional). Future drafts or versions of this specification **MAY** define additional resource data distribution services.

A task scheduler **SHOULD** trigger resource data distribution from a source node to all of its destination nodes according to the frequency specified by the `sync_frequency` in the node document.

### API

POST <nodeURL>/distribute

Arguments:

resource\_documents\_database:  
name of database to distribute, required

Request Object:

None

Results Object:

```
{  
  "OK":      boolean,      // results summary and errors  
  "error":   "string"      // T if successful  
}
```



Return Codes:

200	Successful
500	An error

## Resource Distribution

// Distribute a resource data description document collection from one node to its connected nodes

VALIDATE resource\_document\_database.doc\_type = "resource\_data"

// only distributing resource data

GET the *network node description* document for the source node to obtain

source.network\_id

source.community\_id

source.gateway\_node

GET the *network community description* document for the source node to obtain

source.social\_community

GET the *network node connectivity* documents for the source node

IF > 1 connectivity document with gateway\_connection = T

THEN ABORT

// only one gateway is allowed, faulty network description

FOR EACH node connectivity document

GET the *network node description* document for the destination node

(via the destination\_node\_url) to obtain

destination.network\_id

destination.community\_id

destination.gateway\_node

GET the *network community description* document for the destination node to obtain

destination.social\_community

IF source.community\_id <> destination.community\_id

AND ((NOT source.social\_community)

OR (NOT destination.social\_community))

THEN SKIP

// cannot distribute across non social communities

IF (NOT gateway\_connection) AND

source.network\_id <> destination.network\_id

THEN SKIP

// cannot distribute across networks (or communities) unless gateway

IF gateway\_connection AND

source.network\_id = destination.network\_id

THEN SKIP

// gateway must only distribute across different networks

IF gateway\_connection

AND ((NOT source.gateway\_node)

(OR (NOT destination.gateway\_node))

THEN SKIP

// gateways can only distribute to gateways

```
COMMIT all outstanding resource data description database operations
PERFORM distribution service between source and destination nodes
// this is the distribution primitive
```

*NB:* There may be a better way to do the validations via map-reduce.

*NB:* Since all attributes of the network that model its topology are immutable, the replication process should be transitionally safe.

*NB:* The process is only designed to distribute resource data. It encodes specific business rules about gateway processing. It **SHOULD NOT** be used to distribute network descriptions.

*NB:* The process does not return errors when there are bad connection descriptions; they are skipped. A bad connection should never have been accepted; the checks are included to ensure consistency.

*Open Question:* Other services are bound to a specific database. Should this service be bound to a database like the others (no argument), should the other services also have a database argument, ...

### Service Description

```
{
  "doc_type":      "service_description",
  "doc_version":   "0.10.0",
  "doc_scope":     "node",
  "active":        true,
  "service_id":    "<uniqueid>",
  "service_type":  "distribute",
  "service_name":  "Resource Data Distribution",
  "service_version": "0.10.0",
  "service_endpoint": "<nodeURL>",
  "service_auth":  "xxxx"           // a value that is not public access to the service
}
```

## Resource Data Publish Services

Publish services are used to push resource data into the network. They are used by publishing edge nodes. Future drafts or versions of this specification **MAY** define additional resource data publish services.

### Basic Publish Service

The basic publish service pushes an instance of a resource data description document (or a set of documents) directly to a node on a resource distribution network. It is the most basic, direct mechanism to publish resource data.

Each resource data description document in the supplied set is published independently. In addition to the overall service return indicating status, there **SHALL** be one returned object per resource data description document, aligned 1:1 to the documents in the supplied resource data description document array, indicating status of publishing of the resource data description document.

Each resource data description document **SHALL** be published to the node's resource data description document database. Prior to being published, it **SHALL** be validated: e.g., the syntax **MUST** be correct, mandatory values **MUST** be present, all values **MUST** come from the appropriate data space.

If the resource data description document does not have an assigned identifier, the service **SHALL** assign one and return the value.

If the resource data description document has an identifier and a document with the same identifier exists in the resource data description document collection, the new document **SHALL** be an update, replacing the existing document in total. If the resource data description document is being updated, the value of an immutable element **SHALL NOT** be changed.

*NB:* There are no restrictions on the size of a batch publish document set, either in the number of elements or the total size of the HTTP message. An implementation **SHALL** indicate any size limits in the service description.

*NB:* The process currently does not handle attachments.

*Open Question:* Publishing to the node is by the node owner. Do we need more to support trust?

*Open Question:* Does the process need to supply other data -- **fill in once the data model is complete.**

## API

POST <nodeURL>/publish

Arguments:

None

Request Object:

```
{
  "documents":
  [
    {resource_data_description}
  ]
}
```

// resource data to be published  
// array of  
// resource data description documents

Results Object:

```
{
  "OK":          boolean,          // T if successful
  "error":       "string",         // text describing global error
                                     // present only if NOT OK
  "document_results":
  [
    {
      "doc_ID":   "string",        // ID of the document
      "OK":       boolean,         // T if document was published
      "error":    "string",        // text describing error
                                     // present only if NOT OK
    }
  ]
}
```

Return Codes:

200

500

### Basic Publish

```
// Publish each resource data description document in the supplied list
VALIDATE the publish request // apply appropriate business rules
  IF there is an overall error
    THEN // create the global error object
      OK := F
      error := "error msg" // an appropriate error for global condition
      EXIT
  OK := T // global return status
  FOR EACH resource data description document
    VALIDATE the resource data description document // all syntactical and semantic rules
    IF there is an error
      THEN // create an error object array element object for the individual document
        OK := F
        error := "error msg" // an appropriate error for the document
        doc_ID := supplied doc_ID
        SKIP
    IF resource data description document did not have a supplied doc_ID
      THEN generate a new unique doc_ID
    PUBLISH the resource data description document to the node
      by the owner of the node
      to the node's resource data description document database
    IF there is a publishing error
      THEN // create an error object array element object for the individual document
        OK := F
        error := "publish failed" // an appropriate error for the publish failure
        doc_ID := supplied doc_ID
        SKIP
    // create a return object array element object for the individual document
    OK := T
    doc_ID // supplied or generated doc_ID
```

### Service Description

```
{
  "doc_type":      "service_description",
  "doc_version":   "0.10.0",
  "doc_scope":     "node",
  "active":        true,
  "service_id":    "<uniqueid>",
  "service_type":  "publish",
  "service_name":  "Basic Publish",
  "service_version": "0.10.0",
```

```

    "service_endpoint":    "<nodeURL>",
    "service_auth":       "xxxx",          // a value that is not public access to the service
    "service_data":
    {
        "doc_limit":       integer,          // specify the maximum number of documents in
a batch
        "msg_size_limit":  integer          // specify the maximum message size
    }
}

```

## SWORD Publish Service

[SWORD](#) (Simple Web-service Offering Repository Deposit) is a profile of the Atom Publishing Protocol (known as APP or ATOMPUB). The SWORD APP API provides a mechanism for a repository to publish its metadata to a resource distribution network. The API supports SWORD V 1.3.

The complete specification of the SWORD service will be provided in a future draft of the specification.

## API

in a future draft of the specification

## SWORD Publish

// pseudo code in a future draft of the specification

## Service Description

```

{
    "doc_type":            "service_description",
    "doc_version":         "0.10.0",
    "doc_scope":           "node",
    "active":              true,
    "service_id":          "<uniqueid>",
    "service_type":        "publish",
    "service_name":        "SWORD APP Publish V1.3",
    "service_version":     "0.10.0",
    "service_endpoint":    "<nodeURL>",
    "service_auth":       "xxxx", // a value that is not public access to the service
    "service_data":
    {
        "version":         "1.3"
    }
}

```

## Resource Data Access Services

Access services are used to pull resource data from the network. They are used by access edge nodes to obtain one or more resource data description documents for “off network” processing. Future drafts or versions of this specification **MAY** define additional resource data access services.

## Basic Obtain Service

The basic obtain service pulls an instance of a resource data description document (or a set of documents) directly from a node on a resource distribution network. It is the most basic, direct mechanism to access resource data.

For the list of supplied resource data description document IDs, the service **SHALL** return the corresponding resource data description document from the node's resource data description document database if it exists. The results **SHALL** be aligned 1:1 with the IDs in the request. If the ID does not match a resource data description document in the node's resource data description document database, the result object returned **SHALL** be NULL.

*NB:* There are no restrictions on number of requested documents or in the total size of the HTTP message or response. An implementation **SHALL** indicate any size limits in the service description.

*NB:* The process currently does not handle attachments.

## API

POST <nodeURL>/obtain

Arguments:

None

Request Object:

// list of resource data descriptions to obtain

{“request\_IDs”:

[

// array of

“doc\_ID”: ID

// resource data description document ID

// required

}]

Results Object:

// list of resource data description documents

{“documents”:

[

{

“doc\_ID”: ID,

// resource data description document ID

{resource\_data\_description}

// resource data description documents

// present only if ID is valid, otherwise NULL

}

]

}

Return Codes:

200

500

## Basic Obtain

// Obtain the resource data description document for each supplied ID

```

FOR EACH resource data description document ID
  Put the resource data description document ID in the results object
  GET the corresponding resource data description document
  IF Successful
    THEN PUT the resource data description document in the results object
    ELSE PUT NULL in the results object

```

### Service Description

```

{
  "doc_type":      "service_description",
  "doc_version":   "0.10.0",
  "doc_scope":     "node",
  "active":        true,
  "service_id":    "<uniqueid>",
  "service_type":  "access",
  "service_name":  "Basic Obtain",
  "service_version": "0.10.0",
  "service_endpoint": "<nodeURL>",
  "service_auth":   "xxxx"           // a value that is not public access to the service
}

```

### Basic Harvest Service

The basic harvest service can be used by an external agent to connect to a node to harvest (pull) the resource data description documents held by the node. The service is patterned after the OAI-PMH specification. It can harvest the native JSON encoded metadata or paradata formats used in the Learning Registry.

The complete specification of the basic harvest service will be provided in a future draft of the specification.

### API

in a future draft of the specification

### Basic Harvest

// pseudo code in a future draft of the specification

### Service Description

```

{
  "doc_type":      "service_description",
  "doc_version":   "0.10.0",
  "doc_scope":     "node",
  "active":        true,
  "service_id":    "<uniqueid>",
  "service_type":  "access",
  "service_name":  "Basic Harvest",
  "service_version": "0.10.0",
}

```

```

"service_endpoint": "<nodeURL>",
"service_auth": "xxxx" // a value that is not public access to the service
}

```

## OAI-PMH Harvest Service

The OAI-PMH harvest services can be used by an external agent to connect to a node to harvest (pull) the metadata contained in the resource data description documents stored at the node. The service defines how to harvest a variety of metadata formats (DC, LOM), paradata formats, etc., along with full resource data description documents stored at the node. Unless specified, the service **SHALL** support OAI-PMH V2.0

The complete specification of the OAI-PMH harvest service will be provided in a future draft of the specification. The service supports a subset of the functionality defined in the OAI-PMH specification.

### API

in a future draft of the specification

## OAI-PMH Harvest

// pseudo code in a future draft of the specification

### Service Description

```

{
  "doc_type": "service_description",
  "doc_version": "0.10.0",
  "doc_scope": "node",
  "active": true,
  "service_id": "<uniqueid>",
  "service_type": "access",
  "service_name": "OAI-PMH Harvest",
  "service_version": "0.10.0",
  "service_endpoint": "<nodeURL>",
  "service_auth": "xxxx", // a value that is not public access to the service
  "service_data": {
    {
      "version": "OAI-PMH 2.0"
    }
  }
}

```

## Basic Query Service

The complete specification of the Basic Query service will be provided in a future draft of the specification.

### API

in a future draft of the specification



## Basic Query

// pseudo code in a future draft of the specification

### Service Description

```
{
  "doc_type":      "service_description",
  "doc_version":   "0.10.0",
  "doc_scope":     "node",
  "active":        true,
  "service_id":    "<uniqueid>",
  "service_type":  "access",
  "service_name":  "Basic Query",
  "service_version": "0.10.0",
  "service_endpoint": "<nodeURL>",
  "service_auth":  "xxxx"           // a value that is not public access to the service
}
```

## SRU Service

The complete specification of the SRU/CQL query service will be provided in a future draft of the specification.

### API

in a future draft of the specification

## SRU

// pseudo code in a future draft of the specification

### Service Description

```
{
  "doc_type":      "service_description",
  "doc_version":   "0.10.0",
  "doc_scope":     "node",
  "active":        true,
  "service_id":    "<uniqueid>",
  "service_type":  "access",
  "service_name":  "SRU",
  "service_version": "0.10.0",
  "service_endpoint": "<nodeURL>",
  "service_auth":  "xxxx"           // a value that is not public access to the service
}
```

## SiteMap Service

The complete specification of the XML SiteMap service for web crawlers will be provided in a future draft of the specification.

## API

in a future draft of the specification

## SiteMap

// pseudo code in a future draft of the specification

### Service Description

```
{
  "doc_type":      "service_description",
  "doc_version":   "0.10.0",
  "doc_scope":     "node",
  "active":        true,
  "service_id":    "<uniqueid>",
  "service_type":  "access",
  "service_name":  "SiteMap",
  "service_version": "0.10.0",
  "service_endpoint": "<nodeURL>",
  "service_auth":   "xxxx"           // a value that is not public access to the service
}
```

## Atom Pub Service

The complete specification of the Atom Pub service will be provided in a future draft of the specification.

## API

in a future draft of the specification

## Atom Pub

// pseudo code in a future draft of the specification

### Service Description

```
{
  "doc_type":      "service_description",
  "doc_version":   "0.10.0",
  "doc_scope":     "node",
  "active":        true,
  "service_id":    "<uniqueid>",
  "service_type":  "access",
  "service_name":  "Atom Pub",
  "service_version": "0.10.0",
  "service_endpoint": "<nodeURL>",
  "service_auth":   "xxxx"           // a value that is not public access to the service
}
```

## Administrative Services

Administrative services are used to trigger network node administrative operations, to determine node status or to retrieve descriptive information about a network node. Future drafts or versions of this

specification **MAY** define additional administrative services. Future drafts or versions of this specification **MAY** define additional service query arguments that will customize the returned data.

All administrative services **SHALL** support HTTP content negotiation. All administrative services **SHALL** support return of **CONTENT-TYPE: text/plain**. All administrative services **SHOULD** support return of **text/html**, **text/xml**, **application/rdf+xml**.

## Network Node Status Service

The network node status service is used to return information and operational data about a network node. The service **SHALL** return all of the key-value pairs listed that have a valid value. The service **MAY** return additional key-value pairs that indicate status.

A network node **SHALL** maintain all of the data necessary to return the required key-value pairs.

### API

GET <nodeURL>/status

Arguments:

None

Request Object:

None

Results Object:

```
{
  "timestamp",          "string",      // time of report, time/date encoding
  "active":             boolean;    // is the network node active
  "node_id",            "string",    // ID of the network node
  "node_name",           "string",    // name of the network node
  "doc_count",           integer,     // number of resource data documents
                                // held by the node
  "install_time":       "string",    // time/date of node install
  "start_time":         "string",    // server restart time/date
                                // last reboot
  "last_in_sync":       "string",    // time of last inbound sync
                                // omit if node has not sync'ed
  "in_sync_node":       "string",    // id of the node from the last inbound sync
                                // omit if node has not sync'ed
  "last_out_sync":      "string",    // time of last outbound sync
                                // omit if node has not sync'ed
  "out_sync_node"       "string",    // id of the node for the last outbound sync
                                // omit if node has not sync'ed
}
```

### Network Node Status

// Return the operational status of a network node

```

DEFINE VIEW on
    network node description document containing the required fields
    + network node operational data containing the required fields
QUERY
TRANSFORM results to specified CONTENT-TYPE

```

### Service Description

```

{
  "doc_type":          "service_description",
  "doc_version":       "0.10.0",
  "doc_scope":         "node",
  "active":            true,
  "service_id":        "<uniqueid>",
  "service_type":      "access",
  "service_name":      "Network Node Status",
  "service_version":   "0.10.0",
  "service_endpoint":  "<nodeURL>",
  "service_auth":      "xxxx"           // a value that is not public access to the service
}

```

### Network Node Description Service

The network node description service is used to return descriptive information about a network node, the resource distribution network that it is a part of and the network community that it is a part of. The service **SHALL** return all of the key-value pairs listed that have a valid value. The service **MAY** return additional informational values.

### API

GET <nodeURL>/description

Arguments:  
None

Request Object:  
None

Results Object:

```

{
  "timestamp",          "string",          // time of report, time/date encoding
  "active":             boolean;           // is the network node active
  "node_id",            "string",          // ID of the network node
  "node_name",          "string",          // name of the network node
  "node_description":   "string",          // description of the node
  "node_admin_url":     "string",          // URL of node admin
  "network_id":         "string",          // id of the network
  "network_name":       "string",          // name of the network
  "network_description": "string",          // description of the network
}

```

```

"network_admin_url": "string",           // URL of network admin
"community_id":      "string",           // id of the community
"community_name":    "string",           // name of the community, optional
"community_description": "string",       // description of the community
"community_admin_url": "string",         // URL of community admin
"policy_id":         "string",           // id of the policy description
"policy_version":    "string",           // version identifier for the policy
"gateway_node":      boolean,            // node is a gateway node
"open_connect_source": boolean,          // node is willing to be a source
"open_connect_dest": boolean,            // node is willing to be a destination
"social_community":  boolean,            // is community is a social community
}

```

### Network Node Description

```

// Return the description of a network node
DEFINE VIEW on
    network node description document containing the required fields
    + resource distribution network description document containing the required fields
    + resource distribution network policy document containing the required fields
    + network community description document containing the required fields
QUERY
TRANSFORM results to specified CONTENT-TYPE

```

### Service Description

```

{
  "doc_type":      "service_description",
  "doc_version":   "0.10.0",
  "doc_scope":     "node",
  "active":        true,
  "service_id":    "<uniqueid>",
  "service_type":  "access",
  "service_name":  "Network Node Description",
  "service_version": "0.10.0",
  "service_endpoint": "<nodeURL>",
  "service_auth":  "xxxx"           // a value that is not public access to the service
}

```

### Network Node Services Service

The network node services service is used to return the list of services available at a network node. For each service at a node, the service **SHALL** return all of the key-value pairs listed that have a valid value. The service **MAY** return additional key-value pairs for a service.

The service **SHOULD** group and sort the results in some logical form, e.g., by **ACTIVE**, by **TYPE**.

### API

```
GET <nodeURL>/services
```

Arguments:

None

Request Object:

None

Results Object:

```
{
  "timestamp",          "string",          // time of report, time/date encoding
  "active":             boolean;          // is the network node active
  "node_id",            "string",          // ID of the network node
  "node_name",          "string",          // name of the network node
  "services",           // array of service description objects
  [
    {
      "active":          boolean;          // is the service active
      "service_id":      "string",          // id of the service
      "service_type":    "string",          // fixed vocabulary
      "service_name":    "string",          // name of the service
      "service_description": "string",      // description of the service
      "service_version": "string",          // version number of the service description
      "service_endpoint": "string",         // URL of service
      "service_auth":    "string",          // fixed vocabulary
      "service_data":    {}                // service-specific name-value pairs
    }
  ]
}
```

### Network Node Services

// Return the description of network node services

DEFINE VIEW on

*network node description* document containing the required fields

+ ALL *network node service description* documents containing the required fields

GROUPED and ORDERED on service attributes.

QUERY

TRANSFORM results to specified CONTENT-TYPE

### Service Description

```
{
  "doc_type":            "service_description",
  "doc_version":         "0.10.0",
  "doc_scope":           "node",
  "active":              true,
  "service_id":          "<uniqueid>",
  "service_type":        "access",
  "service_name":        "Network Node Services",
  "service_version":     "0.10.0",
}
```

```

    "service_endpoint":    "<nodeURL>",
    "service_auth":        "xxxx"           // a value that is not public access to the service
}

```

## Resource Distribution Network Policy Service

The resource distribution network policies service is used to return information about the policies that apply to the resource distribution network that the network node is a part of. The service **SHALL** return all of the key-value pairs listed that have a valid value. The service **MAY** return additional policy key-value pairs. The service **MAY** be called at any node in the resource distribution network; all network nodes store an identical copy of the policy data.

### API

GET <nodeURL>/policy

Arguments:

None

Request Object:

None

Results Object:

```

{
    "timestamp",           "string",           // time of report, time/date encoding
    "active":              boolean;           // is the network node active
    "node_id",             "string",           // ID of the network node
    "node_name",           "string",           // name of the network node
    "network_id":          "string",           // id of the network
    "network_name":        "string",           // name of the network
    "network_description": "string",           // description of the network
    "policy_id":           "string",           // id of the policy description
    "policy_version":      "string",           // version identifier for the policy
    "TTL":                 integer.           // minimum time to live for resource data
}

```

### Resource Distribution Network Policy

// Return the description of network policies

DEFINE VIEW on

*network node description* document containing the required fields

+ *resource distribution network description* document containing the required fields

+ *resource distribution network policy* document containing the required fields

QUERY

TRANSFORM results to specified CONTENT-TYPE

### Service Description

```

{
    "doc_type":            "service_description",

```

```

"doc_version":      "0.10.0",
"doc_scope":        "node",
"active":           true,
"service_id":        "<uniqueid>",
"service_type":      "access",
"service_name":      "Resource Distribution Network Policy",
"service_version":   "0.10.0",
"service_endpoint":  "<nodeURL>",
"service_auth":      "xxxx"           // a value that is not public access to the service
}

```

## Broker Services

Broker services operate at a network node to process, transform, augment, amplify or otherwise manipulate the resource data held at a node. No broker services are currently defined. Broker services will be defined in future drafts or versions of the specification.

## Common Data Model and API Attributes and Behaviors

The definition of several common attributes shared across all data models and APIs, along with common API behaviors are specified here as a single point of specification. In case of a discrepancy, the definition here SHALL take precedence over the definition elsewhere in this specification.

### Data Model Attributes

#### Identifiers

Most data models include one or more *identifiers*. An identifier SHALL be unique within a defined scope or context. Unless otherwise specified, the scope for all identifiers SHALL be all implementations of the Learning Registry. Unless otherwise specified by an implementation, an identifier SHALL conform to ISO/IEC 11578:1998, ISO/IEC 9834-8, RFC 4122, and SHOULD use Version 5 (SHA-1 Hash). These specifications standardize the generic OSF DCE UUID. As a data type, an identifier is commonly encoded as a string.

*NB:* What is called an identifier in a data model is more precisely just a label. The use of the label to identify an instance of the data model within the scope or context of the Learning Registry makes it an identifier (within that scope).

*NB:* For many items, the scope could be all implementations of the Learning Registry within one network community. Extending the scope to all implementations is an intentional simplification.

*Open Question:* UUID Version 1 (MAC Address) Version 5 (SHA-1 Hash)?

### Strings and Internationalization

All character strings SHALL support full UTF-8 encoding of Unicode character representation.

### Time and Date



The format for all times and dates **SHALL** conform to ISO 8601. All times **SHALL** include a Time Zone designator. **ToDo:** Specify JSON encoding.

## API Attributes and Behaviors

Description here about RESTful APIs, CoolURIs, Context negotiation, application/JSON vs text/plain

HTTP requests **SHALL** use HTTP 1.1. Communications **MAY** use TLS.

HTTP requests **SHOULD** include a CONTENT-TYPE header. Unless noted, the header **SHOULD** be CONTENT-TYPE: application/JSON

HTTP responses **SHOULD** include a CONTENT-TYPE header. Unless noted, the header **SHOULD** be CONTENT-TYPE: text/plain; charset=utf=8

## Operations

This section outlines one approach of how to use the specified network model, data models and APIs to set up and operate a Learning Registry network community.

## Networks and Communities

### Building the Network Description

Node-specific information (i.e., description, services and connectivity) **SHALL** be maintained on a per node basis. Authorized document updates **MAY** be applied only at the node. All node-specific information **MAY** be maintained in a node document collection per node.

Network-specific information (i.e., network description, network policy) **SHALL** be replicated on a per node basis. The initial document **MAY** be stored at any node in the network. Authorized document updates **MAY** be applied at any node in the network. Replication, using the node's connectivity information, **MAY** be used to propagate the documents throughout the network. All network-specific information **MAY** be maintained in a network document collection per node.

Community-specific information (i.e., community description) **SHALL** be replicated on a per node basis. The initial document **MAY** be stored at any node in the network. Authorized document updates **MAY** be applied at any node in the community. Replication, using the node's connectivity information, **MAY** be used to propagate the documents throughout the network, including across network gateways between different communities. All community-specific information **MAY** be maintained in a community document collection per node.

When using document propagation, the node's document collections **MAY** be out of sync, temporarily violating the requirement for identical values at all nodes. *NB:* Need to determine if lack of sync can introduce any security holes.

An implementation **MAY** place the node-specific information in one document database that is not replicated, network-specific information in a second database that is replicated throughout the network, and community-specific information in a third database that is replicated across the community.

**Open Issue:** As defined, gateways permit networks that allow documents to be replicated **into** the network from another network, but do not permit documents to be replicated **out** of the network (unless there is a corresponding bi-directional link). With such a topology, community-specific information cannot reach all nodes if it is published to a node within a network with no outbound flow; the information will not reach other networks. Possible solutions:

- publish the community-specific information to a node in each network that does not have outbound connectivity.
- define a distribution strategy (and possible extensions of node connectivity information) to allow bi-directional flow across network gateways or that limit flow based on document type.
- define node connections that provide only for distribution of network or community information, not resource data.

Likewise, gateways transmit documents across network boundaries. They do not limit distribution to be within a single network. Thus network-specific documents can cross network boundaries. Possible solutions:

- define a distribution strategy that is limited to a network and not across gateways.
- define node connections that provide only for distribution of network information.

*NB:* The procedures described below are currently incomplete and do not implement any of these solutions. They call for “network wide” (meaning network restricted) or “community wide” distribution.

**Open Question:** How to establish a security model so that only an authorized user may update a description at any node. Is it necessary to sign the descriptions or to store a public key in a description?

**Open Question:** How to constrain the network so that if someone finds a connection, they cannot exploit it.

The network model **MAY** be instantiated in a set of procedures. The procedures assume the *resource distribution network description* document, *resource distribution network policy* document and *network community description* document are published before the network is expanded from the base node. If network expansion is done prior to these documents being created, appropriate *Distribution* processes **SHALL** be triggered after the documents are published to distribute them to all nodes in the network or community.

*NB:* The procedures only establish the network structure and descriptive documents. They do not populate the network with resource data. In particular, adding a new node to a network does not load resource data into the node’s resource data description document database.

The procedures include **VALIDATE** steps which are explicit semantic rules that need to be enforced, generally involving values in different documents. The procedures also include **REJECT** steps. These are general rules designed to check if a document is well formed, that all mandatory fields are present, that

values are from constrained vocabularies, that immutable values are not changed on update, etc. The procedure exits without changing the state of any database if validation fails (i.e., the procedure is a transaction).

All process pseudo below code is informative.

### **Establish a Network Node**

```
// Create one new node
FIND a source for the node software, e.g., the Learning Registry GitHub
    // out of band process
INSTALL the node software on a hardware node // physical or virtual
CONFIGURE the node software
CREATE an identity for the node owner
    // used to own node documents, proxy for document owner
CREATE a network node description document for the node
PUBLISH the network node description document to the new node
    by the node owner
    to the node's node document database
    REJECT if the network node description document is not valid
    REJECT if a network node description document exists
FOR EACH service that the node provides:
    CREATE a network node service description document for the service
    PUBLISH the network node services description document to the node
        by the node owner
        to the node's node document database
    REJECT if the network node service description document is not valid
```

### **Establish a Resource Distribution Network**

```
// Create a base, one-node network with a network description
PREREQUISITE: one active node // denoted the base node
    // via the Establish a Node process
CREATE an identity for the network owner
CREATE a resource distribution network description document
PUBLISH the resource distribution network description document to the first node
    by the owner of the resource distribution network description document
    to the node's network document database
VALIDATE // same network
    network_id in the network node description document =
    network_id in the resource distribution network description document
    REJECT if the resource distribution network description document is not valid
    REJECT if a resource distribution network description document exists
CREATE a resource distribution network policy document
PUBLISH the resource distribution network policy document to the base node
    by the owner of the resource distribution network policy document
    to the node's network document database
VALIDATE // same network
    network_id in the network node description document =
```

network\_id in the *resource distribution network policy* document  
REJECT if the *resource distribution network policy* document is not valid  
REJECT if a *resource distribution network policy* document exists

### **Establish a Network Community**

// Create a base, one-network community with a network description  
PREREQUISITE: one active node // denoted the base node  
// via the *Establish a Network Node* process  
PREREQUISITE: one active network  
// via the *Establish a Resource Distribution Network* process  
CREATE an identity for the network community owner  
CREATE a *network community description* document  
PUBLISH the *network community description* document to the base node  
by the owner of the network community  
to the node's *network community document* database  
VALIDATE // same community in network and community descriptions  
community\_id in the *resource distribution network description* document =  
community\_id in the *network community description* document  
VALIDATE // same community in node and community descriptions  
community\_id in the *network node description* document =  
community\_id in the *network community description* document  
REJECT if the *network community description* document is not valid  
REJECT if a *network community description* document exists

### **Add a Network Node to a Resource Distribution Network**

// Add a node to an existing network  
// NB: Nothing in this process loads resource data into the node  
PREREQUISITE: an established active network // one or more active nodes  
PERFORM the *Establish a Network Node* process to create a node // denoted the new node  
FIND another active node in the network to connect to // denoted the existing node  
// discovery and agreement to connect is out of band  
PERFORM the *Adding Connections within a Resource Distribution Network* process  
source node is the existing node  
destination node is the new node  
PERFORM the *Distribution* process to replicate the *network document* database  
from the source node to the destination node  
// propagates network description and policy to only the new node  
// may proceed asynchronously

### **Adding Connections within a Resource Distribution Network**

// Add connectivity between two existing nodes in a network  
// A source node connects to a destination node  
PREREQUISITE: an established network with two or more nodes  
PREREQUISITE: the active source node is known (connecting node)  
PREREQUISITE: the active destination node is known (connected node)  
// discovery and agreement to connect is out of band  
VALIDATE // same network  
network\_id in the source node's *network node description* document =

network\_id in the destination node's *network node description* document  
 VALIDATE // same community  
     community\_id in the source node's *network node description* document =  
     community\_id in the destination node's *network node description* document  
 CREATE a *network node connectivity description* document with  
     source\_node\_url := source node URL  
     destination\_node\_url := destination node URL  
     gateway\_connection := F  
 PUBLISH the *network node connectivity description* document  
     by the source node owner  
     to the source node's *node document* database  
 REJECT if the *network node connectivity description* document is not VALID  
 REJECT if the same source -> destination active connection exists

### Connect Networks within a Community

// Add a gateway connection between two networks  
 // NB: Nothing in this process distributes resource data across the gateway  
 PREREQUISITE: two established networks  
 PREREQUISITE: the active source node is known (connecting node)  
 PREREQUISITE: the active destination node is known (connected node)  
     // discovery and agreement to connect is out of band  
 VALIDATE // different network  
     network\_id in the source node's *network node description* document <>  
     network\_id in the destination node's *network node description* document  
 VALIDATE // same community  
     community\_id in the source node's *network node description* document =  
     community\_id in the destination node's *network node description* document  
 VALIDATE // no gateway  
     FOR EACH source node's *network node connectivity description* document  
         NOT gateway\_connection  
 CREATE a *network node connectivity description* document with  
     source\_node\_url := source node URL  
     destination\_node\_url := destination node URL  
     gateway\_connection := T  
 PUBLISH the *network node connectivity description* document  
     by the source node owner  
     to the source node's *node document* database  
 REJECT if the *network node connectivity description* document is not VALID  
 REJECT if the same source -> destination active connection exists

### Connect Communities

// Add a gateway connection between two communities  
 // NB: Nothing in this process distributes resource data across the gateway  
 PREREQUISITE: two communities networks  
 PREREQUISITE: the active source node is known (connecting node)  
 PREREQUISITE: the active destination node is known (connected node)  
     // discovery and agreement to connect is out of band

```

VALIDATE // social communities
    social_community T in the source node's network community description document
    social_community T in the destination node's network community description document
VALIDATE // different network
    network_id in the source node's network node description document <>
    network_id in the destination node's network node description document
VALIDATE // different community
    community_id in the source node's network node description document <>
    community_id in the destination node's network node description document
VALIDATE // no gateway
    FOR EACH source node's network node connectivity description document
        NOT gateway_connection
CREATE a network node connectivity description document with
    source_node_url := source node URL
    destination_node_url := destination node URL
    gateway_connection := T
PUBLISH the network node connectivity description document
    by the source node owner
    to the source node's node document database
REJECT if the network node connectivity description document is not VALID
REJECT if the same source -> destination active connection exists

```

## Maintaining Networks and Communities

An established network or community **MAY** be maintained by updating descriptions of network nodes, their services, their connectivity, descriptions of resource distribution networks and of network communities. By definition, elements of the network model **SHALL NOT** be deleted; they are transitioned from active to not active. *NB*: The data models contain forward links to other models. Deleting a document would require network-level garbage collection to determine when all links to a document have been deleted.

All process pseudo code below is informative.

### Change Network Node Description

```

// update the description of any node
PUBLISH the updated network node description document to the node
    by the node owner
    to the node's node document database
REJECT if the network node description document is not valid
REJECT if the network node description document is not an update
// node may have only one network node description

```

### Delete a Network Node

```

// remove a node from a network
// but the node remains, inaccessible
// first sync the documents so that others have them
// sync before delete is an operational policy choice; could be modeled in policy

```

COMMIT all outstanding resource data description database operations  
 PERFORM the Distribute Resource process  
 FOR EACH *resource data description* document  
     delete the document from the node's *resource data description document database*  
     // this is an explicit delete  
 PUBLISH the updated *network node description* document to the node  
     by the node owner  
     ACTIVE = F  
     to the node's *node document database*  
     REJECT if the *network node description* document is not valid  
     REJECT if the *network node description* document is not an update  
     // node may have only one network node description  
 FOR EACH *network node services description* document  
     PERFORM the Delete Node Service process  
 FOR EACH *network node connectivity description* document  
     PERFORM the Delete Node Network Connection process

### Change Node Service Description

// update the description of a service at any node  
 PUBLISH the updated *network node services description* document to the node  
     by the node owner  
     to the node's *node document database*  
 REJECT if the *network node service description* document is not valid  
 REJECT if the *network node service description* document is not an update

### Add Node Service

// add a service to any node  
 PUBLISH the new *network node services description* document to the node  
     by the node owner  
     to the node's *node document database*  
 REJECT if the *network node service description* document is not valid  
 REJECT if the *network node service description* document is not an addition

### Delete Node Service

// delete a service from any node  
 PUBLISH the updated *network node services description* document to the node  
     ACTIVE = F  
     by the node owner  
     to the node's *node document database*  
 REJECT if the *network node service description* document is not valid  
 REJECT if the *network node service description* document is not an update

### Change Node Connectivity

// change the connectivity description of a connection from a node  
 // unless there are mutable extension data elements, the process is a NO-OP  
 // all other data elements are immutable  
 PUBLISH the updated *network node connectivity description* document  
     by the source node owner

to the source node's *node document* database  
REJECT if the *network node connectivity description* document is not valid  
REJECT if the *network node connectivity description* document is not an update

#### **Delete Node Network Connectivity**

// remove the connection from a node to another node  
// applies to intra-network or inter-network or inter-community  
PUBLISH the updated *network node connectivity description* document  
    ACTIVE = F  
    by the source node owner  
    to the source node's *node document* database  
REJECT if the *network node connectivity description* document is not valid  
REJECT if the *network node connectivity description* document is not an update

#### **Change Resource Distribution Network Description**

// change the resource distribution network description  
// applied at some node  
PUBLISH the updated *resource distribution network description* document  
    by the *resource distribution network description* document owner  
    to the node's *network document* database  
REJECT if the *resource distribution network description* document is not valid  
REJECT if the *resource distribution network description* document is not an update  
PERFORM a network-wide *Distribution* process to replicate the *network document* database  
    to the other nodes in the network  
    // propagates resource distribution network description to all nodes in the network  
    // may proceed asynchronously

#### **Delete Resource Distribution Network Description**

// delete the resource distribution network description  
// applied at some node  
PUBLISH the updated *resource distribution network description* document  
    ACTIVE = F  
    by the *resource distribution network description* document owner  
    to the node's *network document* database  
REJECT if the *resource distribution network description* document is not valid  
REJECT if the *resource distribution network description* document is not an update  
PERFORM a network-wide *Distribution* process to replicate the *network document* database  
    to the other nodes in the network  
    // propagates resource distribution network description to all nodes in the network  
    // may proceed asynchronously

#### **Change Resource Distribution Network Policy**

// change the resource distribution network policy  
// applied at some node  
PUBLISH the updated *resource distribution network policy* document  
    by the *resource distribution network policy* document owner  
    to the node's *network document* database  
REJECT if the *resource distribution network policy* document is not valid



REJECT if the *resource distribution network policy* document is not an update  
PERFORM a network-wide *Distribution* process to replicate the *network document* database  
to the other nodes in the network  
// propagates resource distribution network policy to all nodes in the network  
// may proceed asynchronously

#### **Delete Resource Distribution Network Policy**

// delete the resource distribution network policy  
// applied at some node  
PUBLISH the updated *resource distribution network policy* document  
ACTIVE = F  
by the *resource distribution network policy* document owner  
to the node's *network document* database  
REJECT if the *resource distribution network policy* document is not valid  
REJECT if the *resource distribution network policy* document is not an update  
PERFORM a network-wide *Distribution* process to replicate the *network document* database  
to the other nodes in the network  
// propagates resource distribution network policy to all nodes in the network  
// may proceed asynchronously

#### **Change Network Community Description**

// change the network community description  
// applied at some node in some network in the community  
// node must have connectivity to reach all other networks  
// otherwise apply to multiple nodes  
PUBLISH the updated *network community description* document  
by the *network community description* document owner  
to the node's *network community document* database  
REJECT if the *network community description* document is not valid  
REJECT if the *network community description* document is not an update  
PERFORM a community-wide *Distribution* process to replicate the  
*community document* database to the other nodes in the community  
// propagates community description to all nodes in the community  
// may proceed asynchronously

#### **Delete Network Community Description**

// delete the network community description  
// applied at some node in some network in the community  
// node must have connectivity to reach all other networks  
// otherwise apply to multiple nodes  
PUBLISH the updated *network community description* document  
ACTIVE = F  
by the *network community description* document owner  
to the node's *network community document* database  
REJECT if the *network community description* document is not valid  
REJECT if the *network community description* document is not an update  
PERFORM a community-wide *Distribution* process to replicate the  
*community document* database to the other nodes in the community

```
// propagates community description to all nodes in the communities
// may proceed asynchronously
```

## Network Discovery

Finding all the nodes in a network or community is a non core service. One approach is to use replication and distribution to build a complete list of all network links at each node (an alternative is to traverse the network and build the link structure). Given a database of node-specific documents that includes a node's connectivity that is not replicated (private to the node), and a second duplicate database that is replicated across the entire network or community, first replicate the connectivity document from the private node database to the network or community database stored at the node. This replication is done at each node in the network. It is a one-way replication from the private node database to the second database, not a full synchronization. Then distribute (synchronize) the second database across all nodes. Distribution (synchronization) across nodes of the network or community database will build a network connectivity link table at each node.

The completeness of the table in showing the entire network or part of the network will depend on the connectivity and gateways, i.e., the connectivity of a network will not be propagated to nodes outside the network unless there is a directional gateway. The tables of networks or communities with only inbound connectivity will include the entries of the external networks.

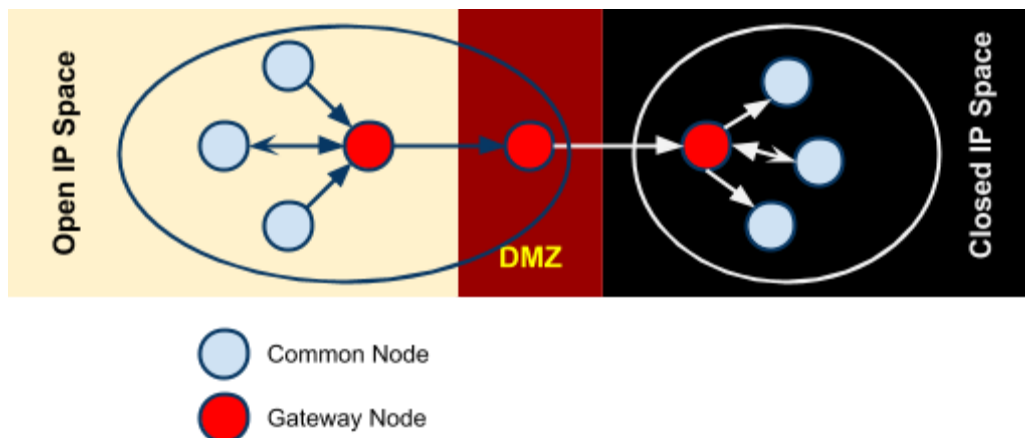
*NB:* This process only builds the network link table at a point in time. It does not provide real-time connectivity status.

## Network Design

Any resource distribution network is technically open; any node **MAY** connect. Managing connectivity and details of the topology and operations of any specific resource distribution network (and its connection to other resource distribution networks and network communities) **SHALL** be determined by the governance and policy rules of a resource distribution network or network node, not via this specification.

Multiple resource distribution networks and network communities **MAY** be established and connected. In addition to external governance and policies, this specification imposes restrictions on connections between network communities and resource distribution networks, i.e., 1:1 connections only using gateway nodes. Because of the technical burden imposed, multiple resource distribution networks and network communities **SHOULD** only be created when these specification-imposed criteria are needed to enable additional security and technical criteria that cannot be adequately satisfied by policy alone.

For example, connecting an open community in an open IP space to a more restricted community might involve establishing a gateway node in the open community in a public IP space, providing a gateway node in for the restricted community in a DMZ, opening limited network ports between the DMZ and the other nodes in restricted community that operate within a closed IP space (possibly using an additional gateway), and locking down the connections between the open community and the DMZ gateway node. The illustration shows an example of such a gateway structure.



## Resource Data Publication, Distribution and Access

Once a resource distribution network has been established, external agents use the publish services at network nodes to publish or update resource data to nodes in the resource distribution network. Network nodes **SHALL** store the valid, published resource data in their resource data description documents database. Publication **MAY** be performed via any of the [supplied publish services](#) or via tools and applications that are built upon these services. Other mechanisms to publish resource data **SHALL NOT** be supported.

Publication **SHALL** satisfy trust and identity conditions. The same resource data may be published to one or more nodes in one or more resource distribution networks subject to their policies and the trust and identity conditions.

Resource distribution network nodes **SHALL** persist the resource data subject to the resource distribution network policies.

Resource distribution network nodes distribute their data to other nodes. If a network node provides the resource data distribution service, it **SHALL** periodically launch the [resource data distribution](#) service to distribute resource data from the node to its connected nodes. Any node that wants to establish a connection with another node, i.e., wants to be the target of distribution, **SHALL** support the necessary core services that allow the source node to distribute data to it.

Resource data is available for access by external agents through the access services at network nodes. Access **MAY** be provided via any of the supplied [access services](#) or via tools and applications that are built using these services. Other mechanisms to access resource data **SHALL NOT** be supported.

Access **SHALL** satisfy trust and identity conditions. When available via distribution, the same resource data may be accessed from one or more nodes in one or more resource distribution networks, subject to their policies and the trust and identity conditions.

Resource distribution network nodes **MAY** provide [broker services](#). Operation of a broker service **SHALL** be determined by the governance and policy rules of a resource distribution network or network node, not via this specification.

Service provisioning (publish, access, distribution, broker) **SHALL** be determined by the governance and policy rules of a resource distribution network or network node, not via this specification.

## Network Administration

Resource distribution network nodes **MAY** provide [administrative services](#). Operation of an administrative service **SHALL** be determined by the governance and policy rules of a resource distribution network or network node, not via this specification.

Provisioning of administrative services **SHALL** be determined by the governance and policy rules of a resource distribution network or network node, not via this specification.

**Open Question:** Do we need to make any admin services mandatory?

## Glossary

The following terms are used in this document as defined.

Additional terms will be provided in a future draft or version of the specification.

*access* (v): to obtain resource data from a network node by an agent that is external to a resource distribution network.

*broker* (n): a server process that provides transformative or data amplification processing of resource data.

*community* (n): see *network community*.

*common node* (n): a network node in a resource distribution network that may provide any service to process resource data and that may connect to any other node in the same resource distribution network for the distribution of resource data within the resource distribution network.

*distribute* (v): to copy or synchronize resource data from one network node to another.

*gateway node* (n): a network node in a resource distribution network that provides an interconnection to a network node in a different resource distribution network (either in the same network community or in a different network community) for the distribution of resource data across the network boundary.

*harvest* (v): to access a network node and obtain sets of resource data; the accessing agent is the harvester; the network node is the harvestee. Harvest is typically based on timestamps used to

identify new resource data held at the harvestee.

*identifier* (n): the name (i.e., a label [e.g., a string] in an authoritative context) associated with a thing (anything that can be given an identifier).

*learning resource* (n): any (digital) resource that is designed for, or has been used, in an educational context.

*metadata* (n): formally authored and curated information describing a learning resource. Also denoted *first party* metadata.

*network* (n): see *resource distribution network*. A network need not correspond to a physical or logical network of computing devices.

*network community* (n): a group of interconnected resource distribution networks.

*network node* (n): a service end point in a resource distribution network that may provide services to process resource data and that may connect to any other nodes to distribute resource data. A network node need not correspond to a physical or logical computing device.

*node* (n): see *network node*.

*paradata* (n): information describing the contextual use of a learning resource. It includes informally authored information and data obtained directly through monitoring the use of a learning resource, its metadata or its paradata. Also denoted *second party* metadata.

*publish* (v): to submit resource data to a network node from a source external to the node's resource distribution network.

*pull* (v): to distribute resource data from A to B, initiated by B.

*push* (v): to distribute resource data from A to B, initiated by A.

*resource* (n): see *learning resource*.

*resource data* (n): any data that describes a learning resource, including, but not limited to metadata and paradata.

*resource distribution network* (n): a group of interconnected network nodes that operate under an agreed set of policies.

*service* (n): a process applied to resource data or system descriptive and operational data operating on a network node.

## References

References below contain both normative and informative references. Unless otherwise noted, this specification references specific versions of other normative standards. More recent versions **SHALL NOT** be used.

Additional references will be provided in a future draft or version of the specification.

- CoolURIs 2008: *Cool URIs for the Semantic Web*, <http://www.w3.org/TR/cooluris/>
- ISO 8601: *Data elements and interchange formats -- Information interchange -- Representation of dates and times*, ISO 8601:2004, [http://www.iso.org/iso/catalogue\\_detail?csnumber=40874](http://www.iso.org/iso/catalogue_detail?csnumber=40874)
- JSON: *The application/json Media Type for JavaScript Object Notation (JSON)*, <http://tools.ietf.org/html/rfc4627>
- OAI-PMH: *The Open Archives Initiative Protocol for Metadata Harvesting*, V2.0, <http://www.openarchives.org/OAI/openarchivesprotocol.html>
- RFC 4122: *A Universally Unique Identifier (UUID) URN Namespace*, RFC 4122, <http://www.ietf.org/rfc/rfc4122.txt>
- SWORD: *SWORD AtomPub Provide V 1.3*, <http://www.swordapp.org/docs/sword-profile-1.3.html>
- Unicode: *The Unicode Consortium. The Unicode Standard, Version 6.0.0*, <http://www.unicode.org/versions/Unicode6.0.0/>
- UTF-8: TBC (where in Unicode 6.0.0 doc?)

## Change Log

*NB:* The change log only lists major updates to the specification.

*NB:* Updates may not result in a version update.

Version	Date	Change
0.10.0	20110117	Public release (lacks resource data model)

## Working Notes and Placeholder Text

Should the spec include a requirements section?

How are APIs secured so that they can only be invoked by trusted agents

Resource data model