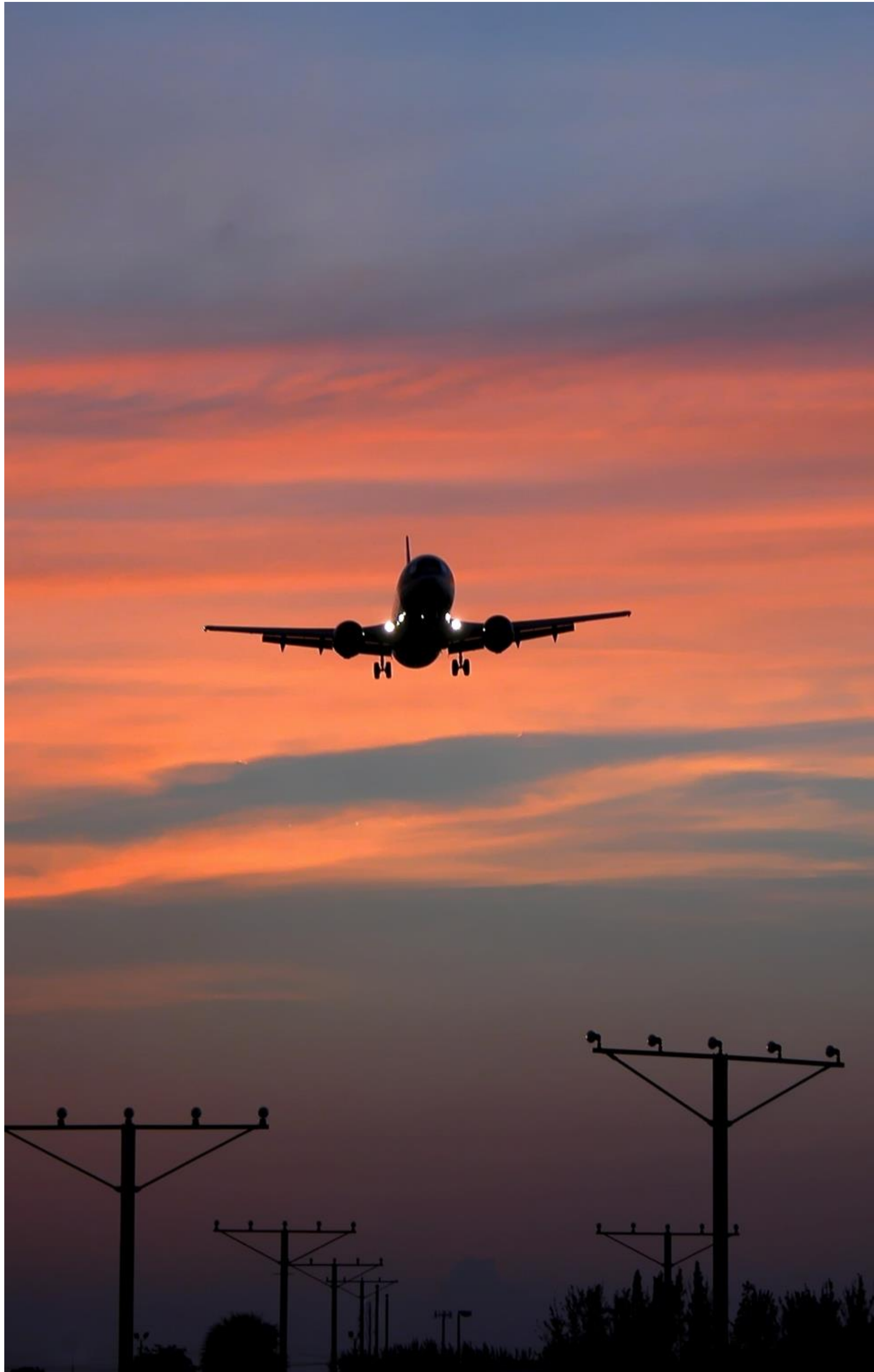
SWIM Discovery Service (SDS) Implementation Specification v.1.0.0

Presented to: OGC Technical Review Workshop

By: **Mark Kaplun (FAA)**
Wen Zhu (NIRA, INC)

Date: **September 9, 2020**



SDS Implementation Specification

- This specification describes the enabling technologies and practices that support federated service discovery among independently developed and autonomously managed discovery mechanisms.
  - It establishes guidelines and general technical principles for the development of a discovery service.
  - And defines patterns of interactions among discovery components to enable service discovery across independently developed and autonomously operated SWIM initiatives.

Background

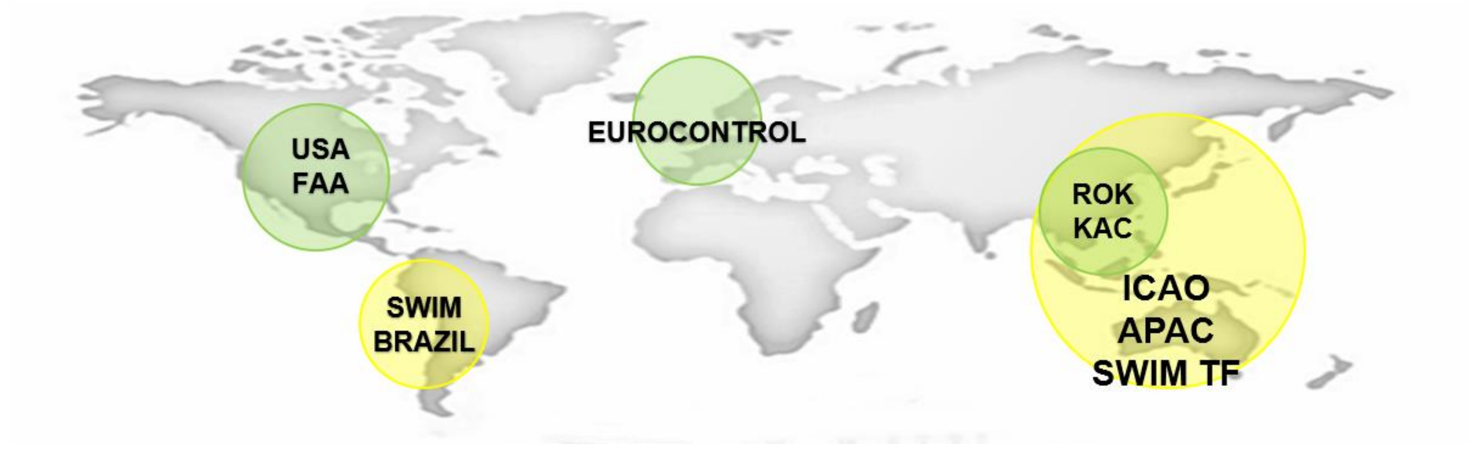
2018	SWIM Inter-Registry Framework (SIRF) CONOPS v1.0
2017	Registry Integration Module (RIM) v1.0
2016	Service Description Conceptual Model (SDCM) v2.0
2015	Service Description Conceptual Model (SDCM) v1.0

Background

- System Wide Information Management (SWIM) is an architectural solution that defines information sharing in the context of civil aviation; it is realized through consistent application of principles of Service-Oriented Architecture(SOA).
- A realization of SWIM usually includes a communications infrastructure, architectural solutions, and governance for identifying, developing, provisioning, and operating a network of highly-distributed and reusable services.

Background (cont.)

- In recent years, SWIM implementations have increased in numbers and complexity.
- The ability of SWIM stakeholders to find (discover) services across geographical and organizational boundaries is a precursor for achieving global information exchange.



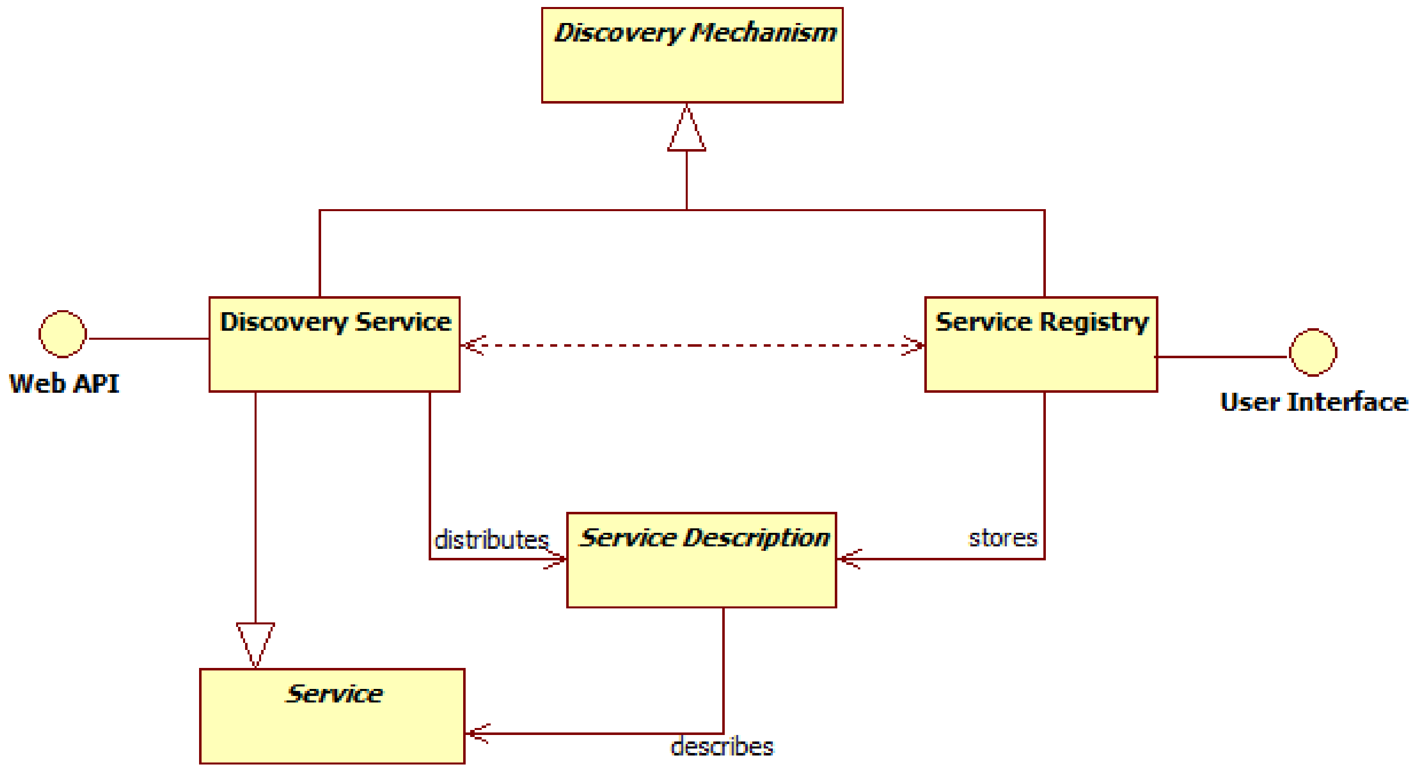
Background (cont.)

- If a service cannot be found by potential consumers, the service -- and SWIM in general -- fails to achieve its objectives.
- The process of locating information about services that meet service consumer business needs is known as *service discovery*.
- Every service-centric environment such as SWIM supports the ability of services to be discovered through some form of retrieval mechanism (e.g., a service registry, a discovery service).

Discovery Service vs. Registry

- A *service registry* is an authoritative, centrally controlled ***store of information***. It is commonly realized as a repository that allows a user to catalog and manage service-relevant information.
- A *discovery service* is a ***service*** that provides service meta-information. It is loosely-coupled, interoperable, autonomous, reusable, discoverable; it is identified by a URI, and its interface and binding are defined, described, and discovered in a formal language (e.g., XML, JSON).

Discovery Service vs. Registry (cont.)



Objectives

- To enable service discovery among *independently developed* and *autonomously managed* SWIM domains.
- To allow a user to obtain service metadata from multiple sources simultaneously in a single, consolidated, semantically consistent result.
- To avoid reliance on a single centralized discovery mechanism by allowing a discovery service to describe itself and other discovery services.

Design Principles

- To conform to SOA architectural principles.
- To be based on World Wide Web architecture and standards.
- To use a formal standard language for information exchange.
- To be self-describing and self-advertising.
- To be composable but not coupled.

Architectural Vision

- Representational State Transfer (REST)
  - REST defines an approach for implementing services (RESTful Web services) that provide and access a representation of information resources by using a uniform and predefined set of stateless operations.
- Peer-to-Peer (P2P) Discovery
  - A P2P distributed application architecture defines a network where every node ("peer") is an equally privileged, equipotent participant with the same capabilities and responsibilities.

The SDS architecture defines a network of RESTful Web discovery services where each service functions as both a "client" and a "server" to the other peers.

SDS Specification Content

- Behavior Model
  - Describes how a discovery service interacts with a user and other services.
- Information Model
  - Defines common structures for information exchanged among discovery services.
- Resource Model
  - Defines a collection of interlinked resources.
- Interface Requirements
  - Prescribes the operations and messages supported by a discovery service.

Behavior Model

- The *Behavior Model* presents a collection of use cases, which together describe the discovery service's behavior, i.e., how the service interacts with a user and other services.
- The SDS specification is designed to meet the use cases presented below.

Actors

- ***user*** - A person who deploys a *user agent* to initiate a request to and receives a response from a discovery service.
  - Note: a user may represent a group of users (organization), but for the purpose of this model the *user* is always a singular entity.
- ***user agent*** - A software program, such as a browser, whose purpose is to mediate interactions with services on behalf of the user under the user's preferences.
- ***peer*** - A *discovery service* that is accessed by or responds to other discovery services.

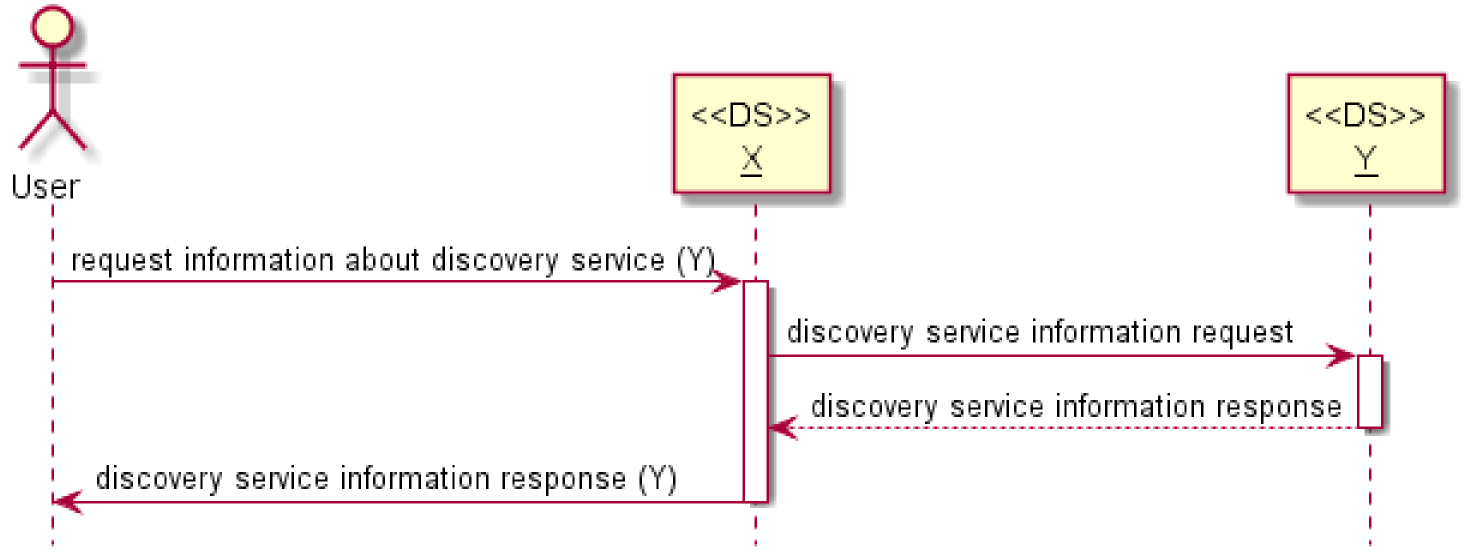
Assumptions

For **all** use cases, the model assumes the following:

- A *user* has access to an affiliated *discovery service* (*DS "X"* in this model) and has the necessary security credentials to perform all operations offered by the service's interface.
- The *user* always utilizes *X* to send a request to other *discovery services (peers)* and consolidate the responses.
- All discovery services are compliant with this specification; however, the extent of conformance and access control policy may vary.

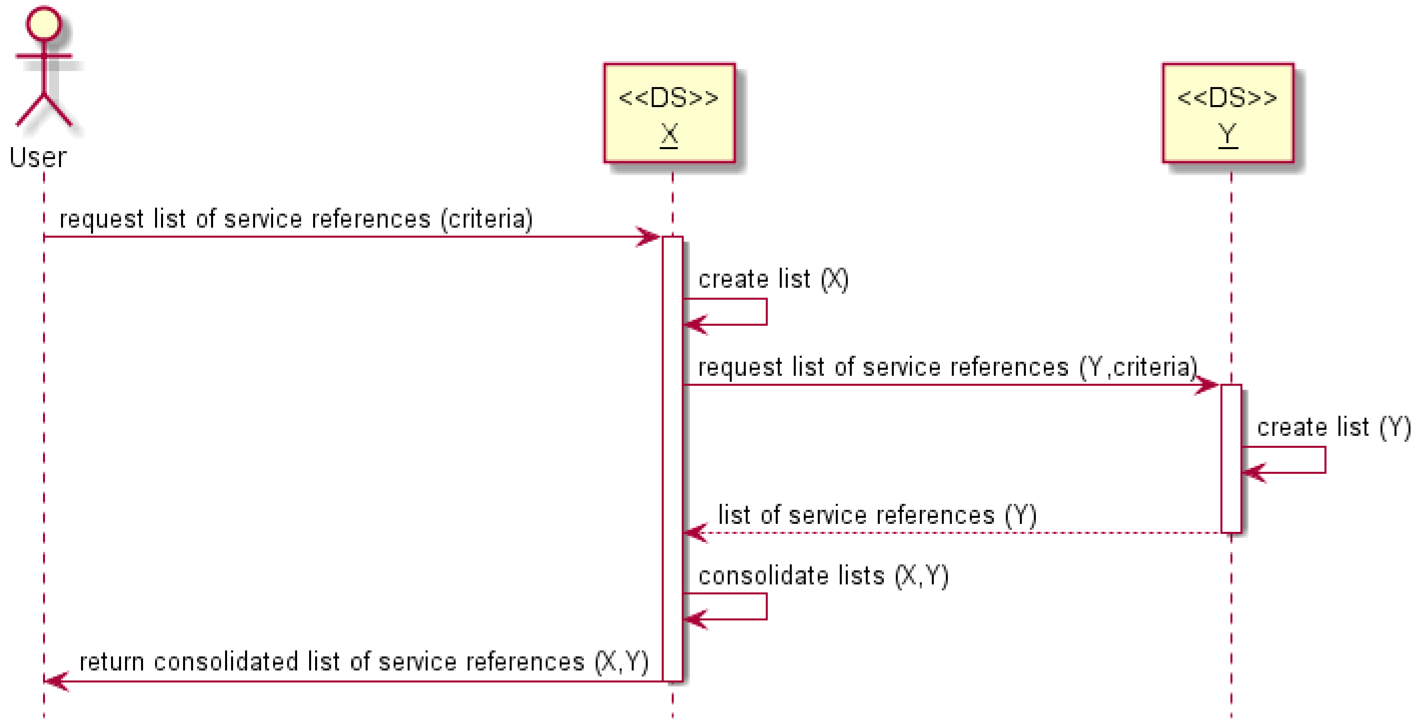
UC 01. Obtaining information about a discovery service

- **Precondition:** The user is aware of the existence (i.e., knows the network address) of a discovery service Y.


User
<<DS>>
X
<<DS>>
Y
request information about discovery service (Y)
discovery service information request
discovery service information response
discovery service information response (Y)

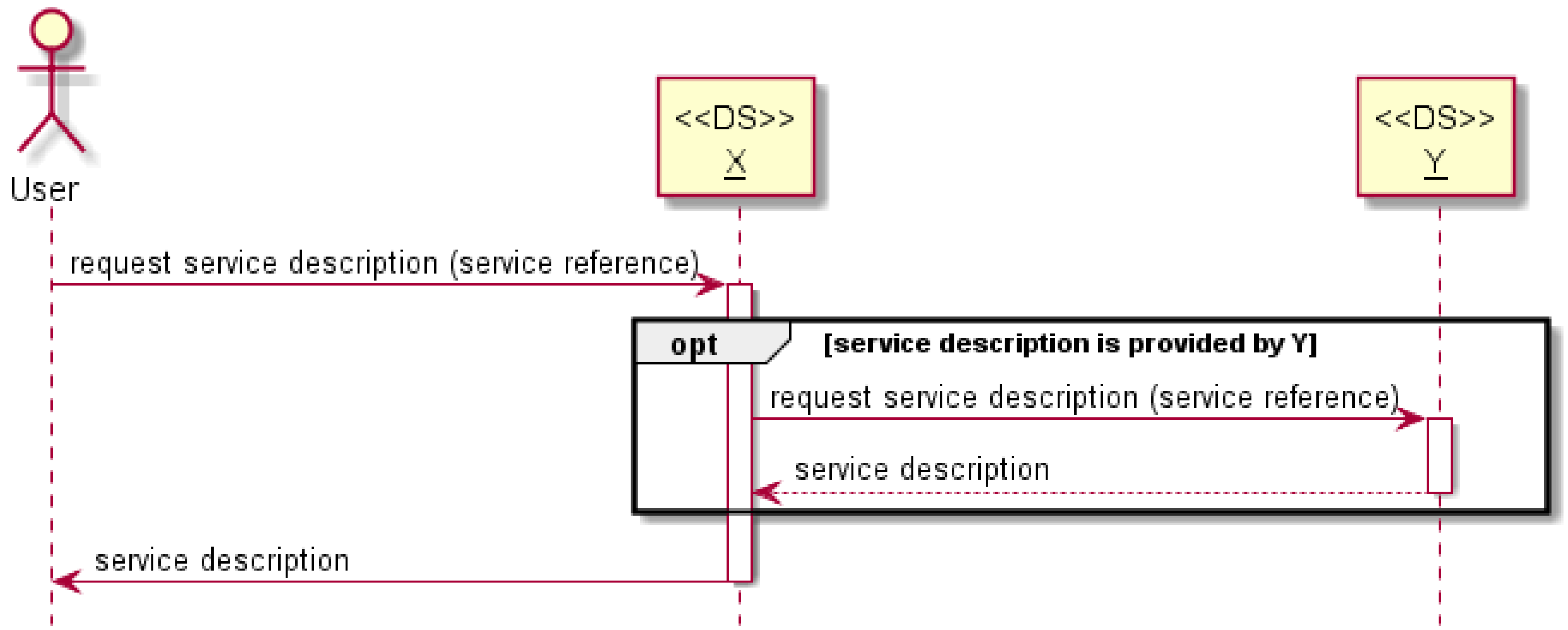
UC 02. Obtaining a list of services

- **Precondition:** After conducting UC01, the user knows that Y is capable of supporting service discovery operations.


User
<<DS>>
X
<<DS>>
Y
request list of service references (criteria)
create list (X)
request list of service references (Y,criteria)
create list (Y)
list of service references (Y)
consolidate lists (X,Y)
return consolidated list of service references (X,Y)

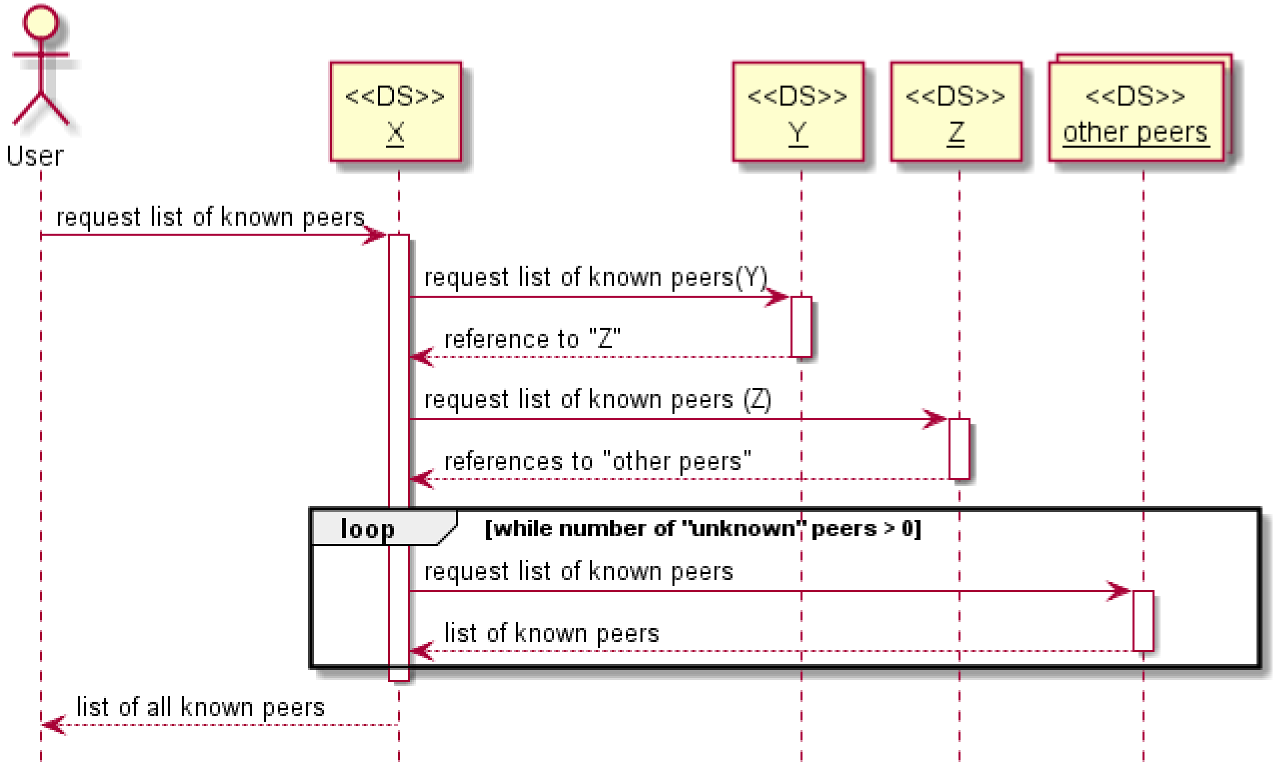
UC 03. Obtaining a description of a service

- **Precondition:** The user has a list of references to descriptions of all services provided by both X and Y.


User
<<DS>>
X
<<DS>>
Y
request service description (service reference)
opt
[service description is provided by Y]
request service description (service reference)
service description
service description

UC 04. Discovering previously unknown peers

- **Precondition:** Y is aware of the address of the discovery service Z not known to X.


User
<<DS>> X
<<DS>> Y
<<DS>> Z
<<DS>> other peers
request list of known peers
request list of known peers(Y)
reference to "Z"
request list of known peers (Z)
references to "other peers"
loop
[while number of "unknown" peers > 0]
request list of known peers
list of known peers
list of all known peers

Information Model

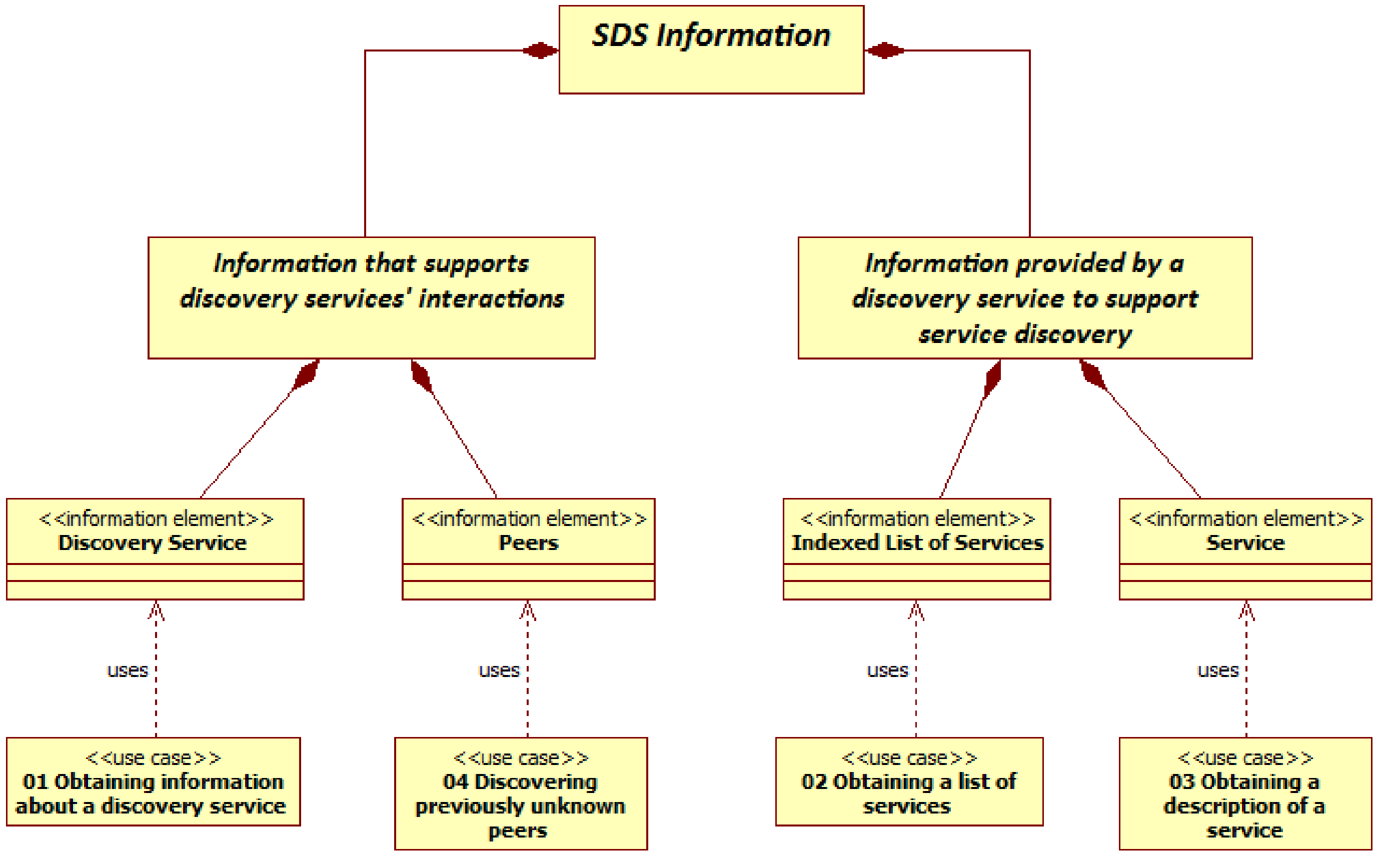
- The SDS Information Model defines common structures for information exchanged among discovery services and/or a user.

Information Model: Structure

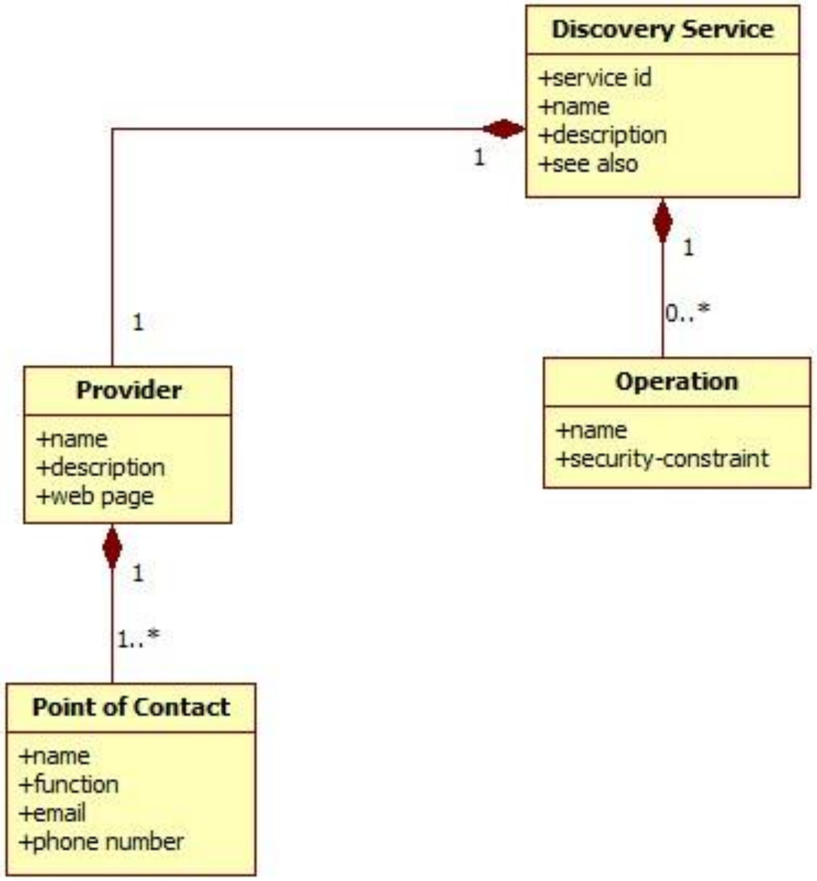
The Information Model distinguishes two kinds of information:

- Information that supports discovery services' interactions, and which may include identification of a discovery service, functionalities provided by the service, access policies, and references to other discovery services (peers).
- Information provided by a discovery service to support service discovery, such as lists of services or detailed descriptions of these services.

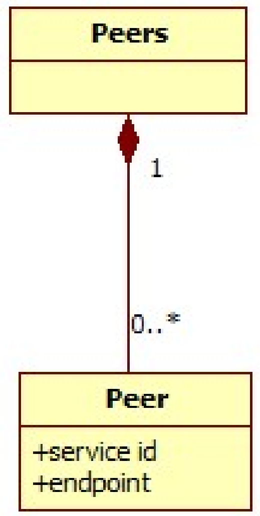
Information Model: Structure (cont.)


SDS Information
Information that supports discovery services' interactions
Information provided by a discovery service to support service discovery
<<information element>> Discovery Service
<<information element>> Peers
<<information element>> Indexed List of Services
<<information element>> Service
uses
uses
uses
uses
<<use case>> 01 Obtaining information about a discovery service
<<use case>> 04 Discovering previously unknown peers
<<use case>> 02 Obtaining a list of services
<<use case>> 03 Obtaining a description of a service

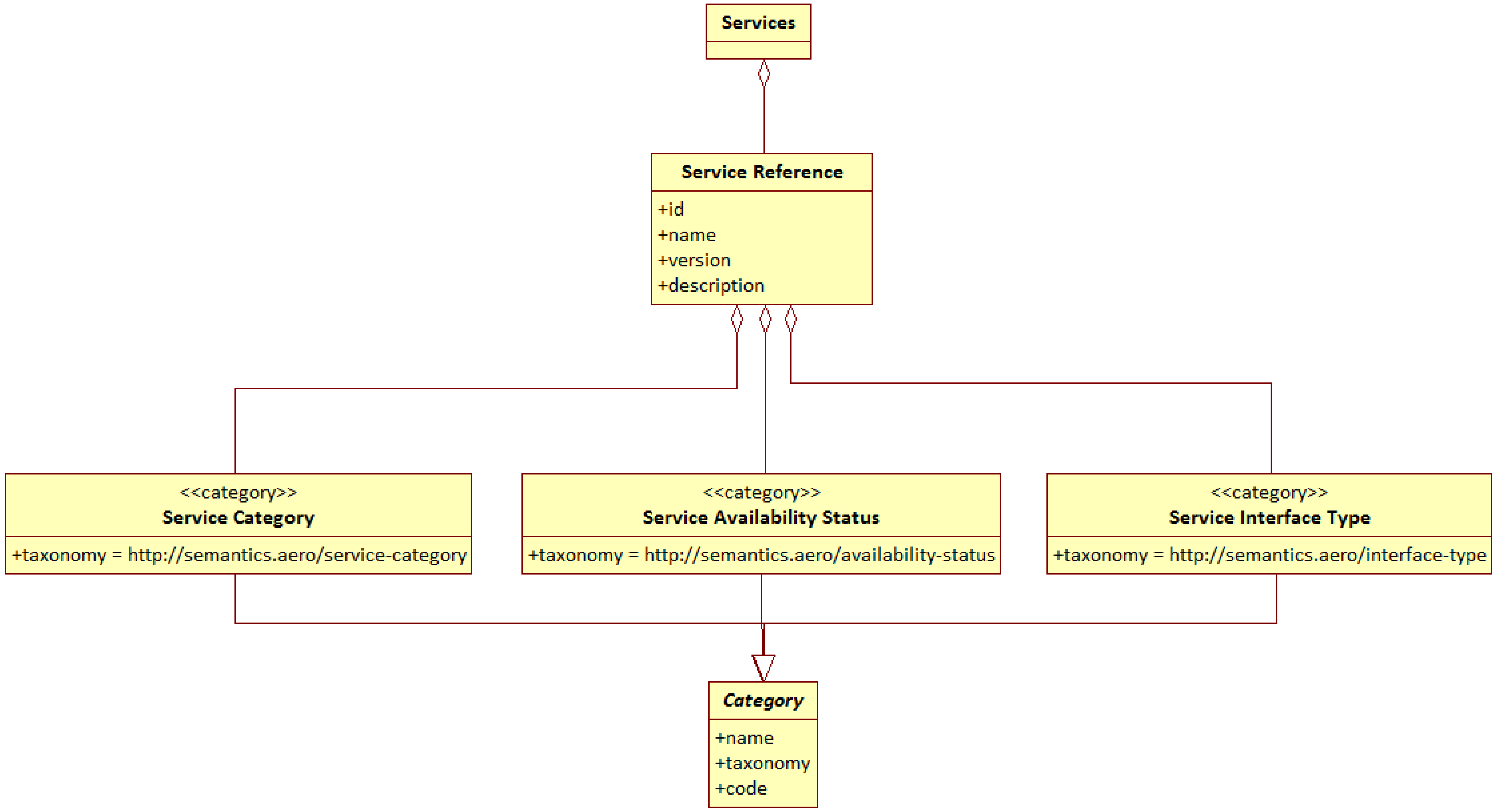
Discovery Service Information



Peers Information



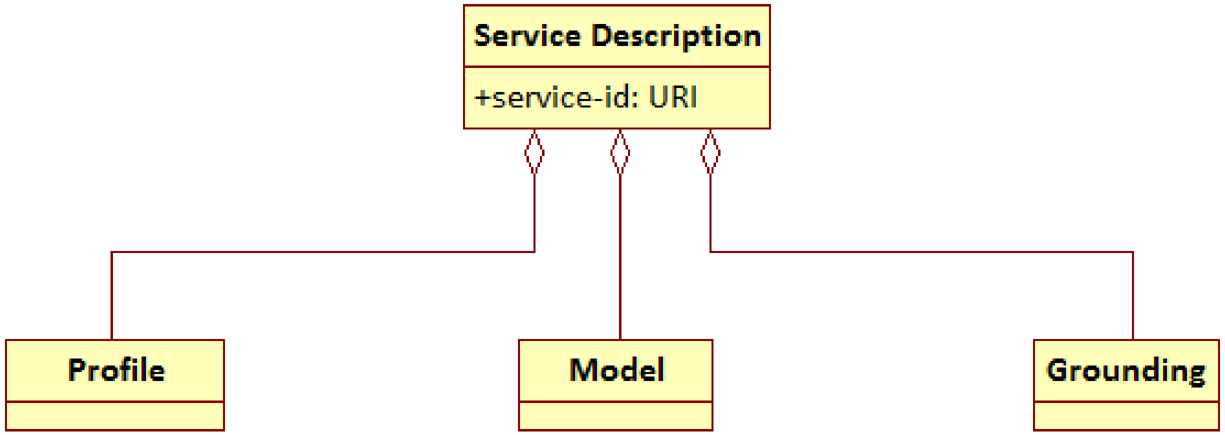
Indexed List of Services



Services
Service Reference
+id
+name
+version
+description
<<category>>
Service Category
+taxonomy = http://semantics.aero/service-category
<<category>>
Service Availability Status
+taxonomy = http://semantics.aero/availability-status
<<category>>
Service Interface Type
+taxonomy = http://semantics.aero/interface-type
Category
+name
+taxonomy
+code

Service (Service Description) Information

- The Service Information element follows Service Description Conceptual Model (SDCM) v. 2.0 (which in turn follows W3C OWL-S).
- For the complete model, see either SDCM 2.0 or its JSON representation, SDM-J, at https://semantics.aero/service-description/sdm-j/sdm-j-1.0.0/.

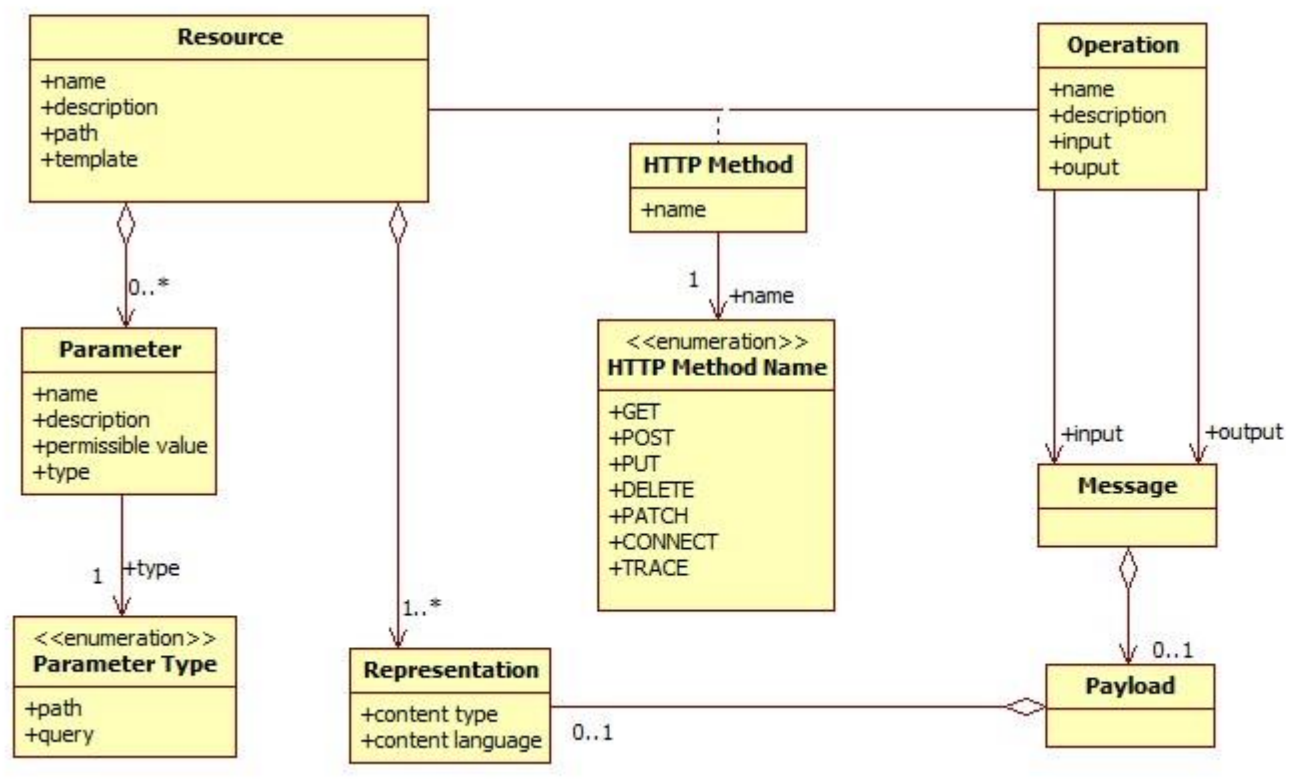


Resource Model

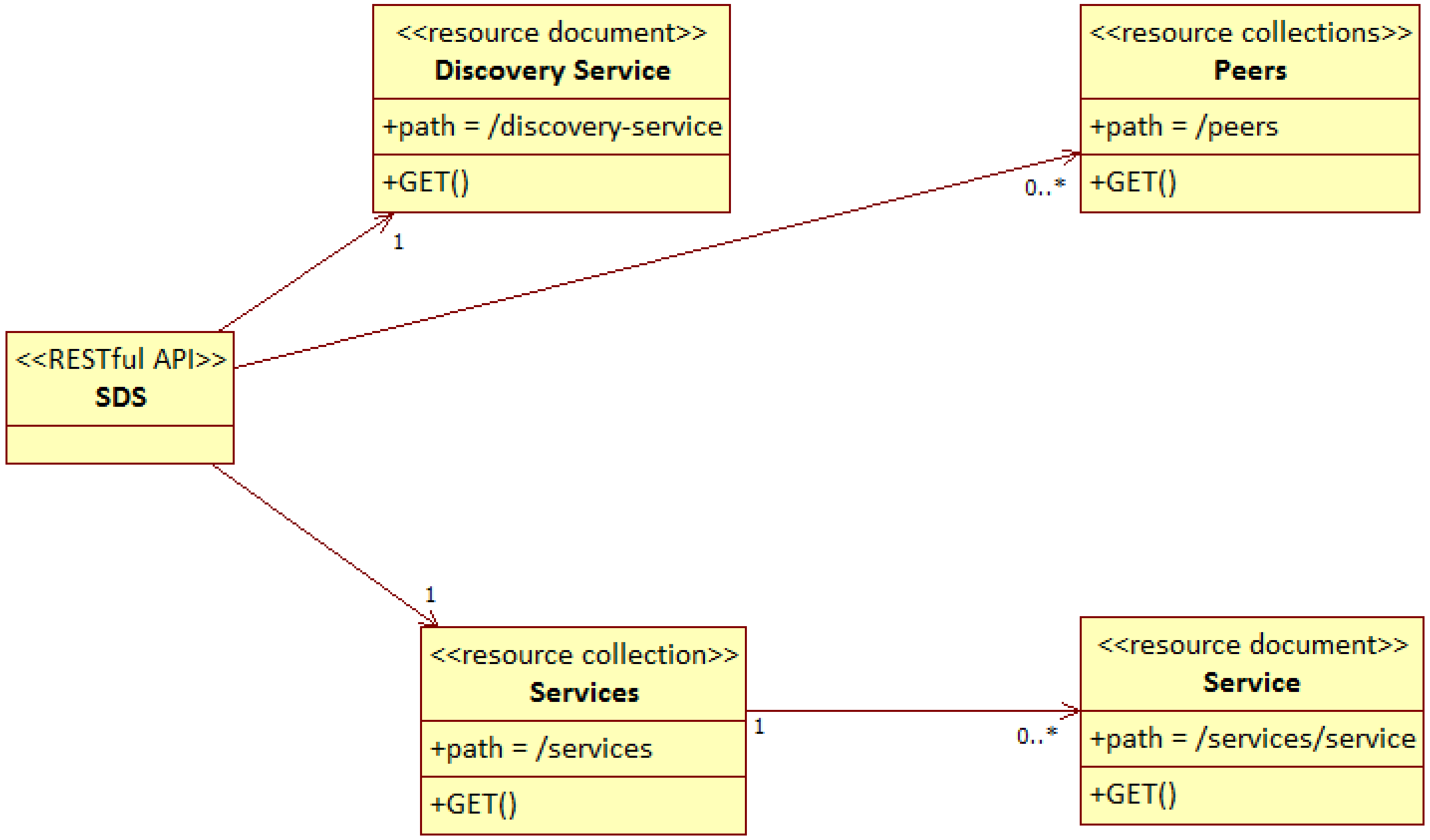
- The SDS Resource Model is a collections of interlinked resources.
- Terms employed in the SDS Resource Model:
  - ***resource:*** An information object identified by a Uniform Resource Identifier (URI).
  - ***resource id:*** A URI by which the resource is uniquely referenced.
  - ***resource path:*** A relative path that represents a resource node within a hierarchical resource model.

    Note: the *path* is appended to the service URL; no relative path resolution is assumed.
  - ***resource template:*** A *resource id* syntax that includes variables that must be substituted before the resource id's resolution.

Resource: Architectural Vision



SDS Resource Model



SDS Resource Model (cont.)

name	***discovery service***
description	A resource that allows a requester to retrieve a description of a discovery service.
path	/discovery-service
http method	GET
template	/discovery-service
representation	JSON, XML (opt.)

name	***peers***
description	A resource that allows a requester to retrieve a collection of references to other discovery services (peers).
path	/peers
http method	GET
template	/peers
representation	JSON, XML (opt.)

name	***services***
description	A resource that allows a requester to retrieve a collection of references to services.
path	/services
http method	GET
template	/services
parameters	service-category, availability-status, interface-type
representation	JSON, XML (opt.)

name	***service***
description	A resource that allows a requester to retrieve information about a specific service.
path	/services/service
http method	GET
template	/ services/{service-id}
representation	JSON, XML (opt.)

Interface Requirements

- The SDS Specification uses OpenAPI v.3.0.0 for defining and prescribing the interface requirements.
- All instances of SDS SHALL be valid with the OpenAPI schema provided in Section 3.a (Figure 14) of this specification.

Operations

- All operations SHALL support the GET HTTP Method as defined in RFC 7231 [1] section 4.3.1.
- Each operation SHALL be read-only, i.e., a requester does not request, and does not expect, any state changes on the invoking service as a result of applying the operation to a target resource.
- Each operation SHALL be idempotent, i.e., the intended effect of multiple identical requests on the invoked service is the same as the effect for a single such request.
- Each operation MAY be cacheable, that is, the received response can be saved for a future use.

Operations (cont.)

name	***GetDiscoveryService***
description	Allows a client to retrieve a *discovery service* resource
obligation	required
example	`http://nsrr.faa.gov/smxs/discovery-service`

name	***GetPeers***
description	Allows a client to retrieve a *peers* resource
obligation	optional
example	`http://nsrr.faa.gov/smxs/peers`

name	***GetServices***
description	Allows a client to retrieve a *services* resource
obligation	required
example	`http://nsrr.faa.gov/smxs/services?service-category=discovery&availability-status=prospective&interface-type=resource-oriented`

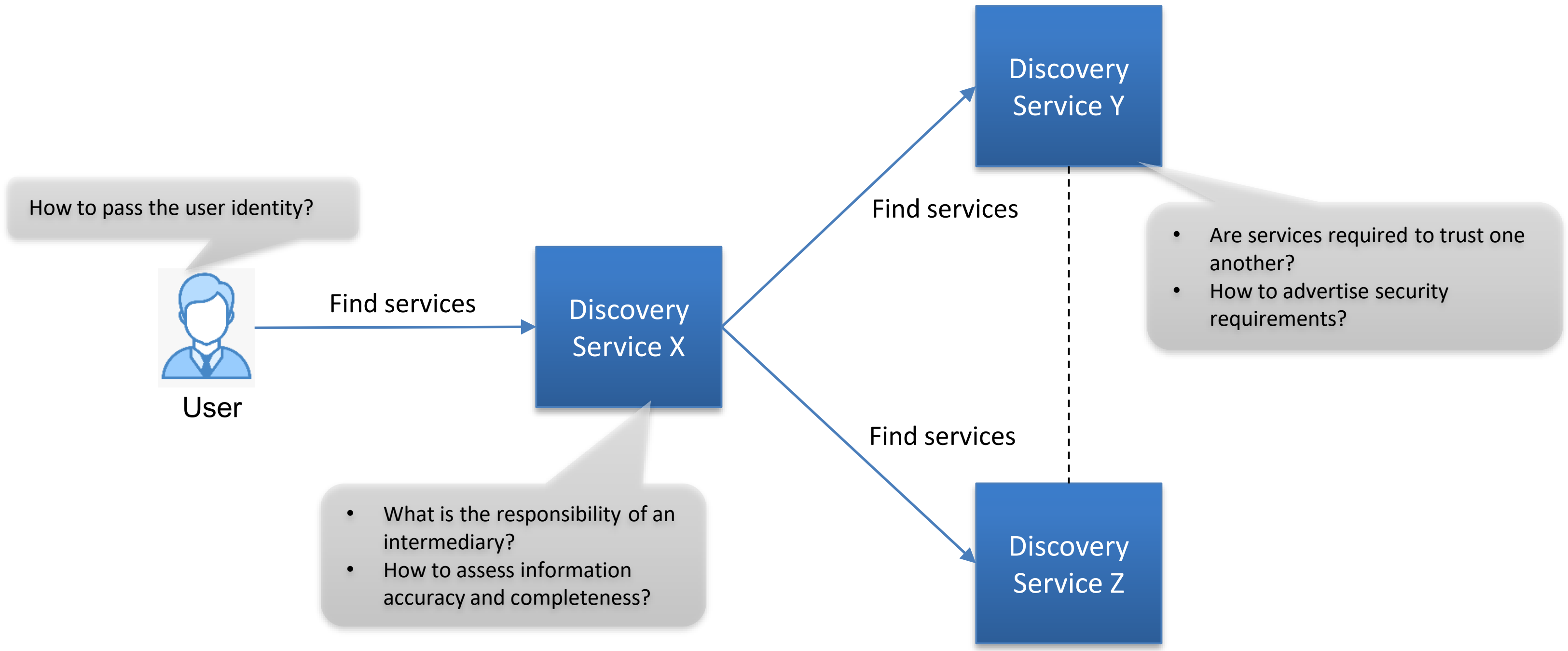
name	***GetService***
description	Allows a client to retrieve a description of the service identified by the resource parameter value
obligation	required
example	`http://swim.org/ds/services/”http:swim.org/fps”`

Messages

- All messages SHALL comply with the syntax and semantics prescribed by RFC 7231.
- All request messages, that is, the messages sent by a user agent or a discovery service, SHALL include a header field “Accept”.
- The value of the field “Accept” SHOULD be “application/json”.
- The value of the field “Accept” MAY be “application/xml”.
- All response messages SHALL include a Content-Type header field to indicate the formal language used by the associated representation.
- The value “application/json” SHALL be a default value for the Content-Type header field.
- The value “application/xml” MAY be included in a request message header.
- All response messages SHALL include a status code as described in RFC 7231 [1] section 6.

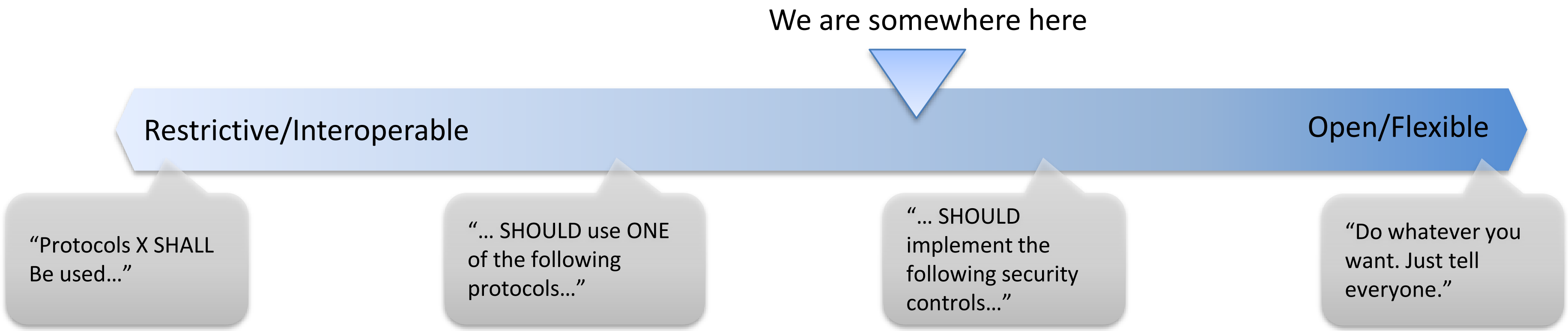
Security Context

- Security and trust require governance and technology considerations.
  - Relationships/agreements among peers.
  - Standards and protocols to secure information exchanges.
- A single solution is <u>unlikely</u> to satisfy the needs of every service.


How to pass the user identity?
User
Find services
Discovery Service X
Find services
Discovery Service Y
Find services
Discovery Service Z
Are services required to trust one another?
How to advertise security requirements?
What is the responsibility of an intermediary?
How to assess information accuracy and completeness?

Approach to Security Requirements

- Current Approach
  - Requiring following best practices for Web services security.
  - Not specifying a particular protocol but calling out examples.

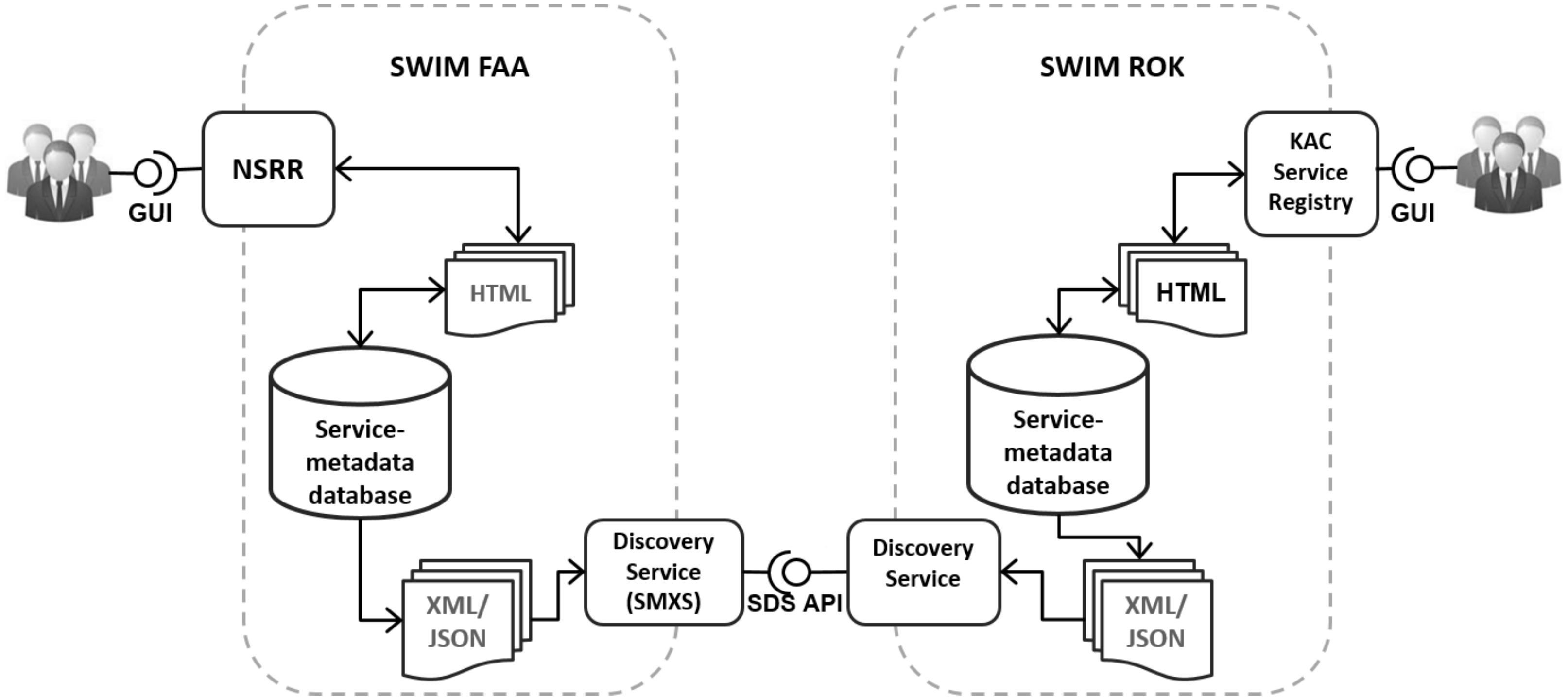

We are somewhere here
Restrictive/Interoperable
Open/Flexible
"Protocols X SHALL Be used…"
"… SHOULD use ONE of the following protocols…"
"… SHOULD implement the following security controls…"
"Do whatever you want. Just tell everyone."

SDS Requirements

- Transport Security
  - Requests SHOULD be made over a communication channel secured by the Transport Security Layer/Secure Socket Layer (TLS/SSL) protocol.
- Access Control
  - MAY restrict access to certain information it maintains.
    - SHOULD require a client to authenticate in these cases.
  - MAY restrict certain operations to a subset of authenticated users.
- Authentication
  - SHOULD use an authentication method that is compatible with HTTP protocol.
    - E.g., HTTP basic or digest authentication, OAuth 2.0

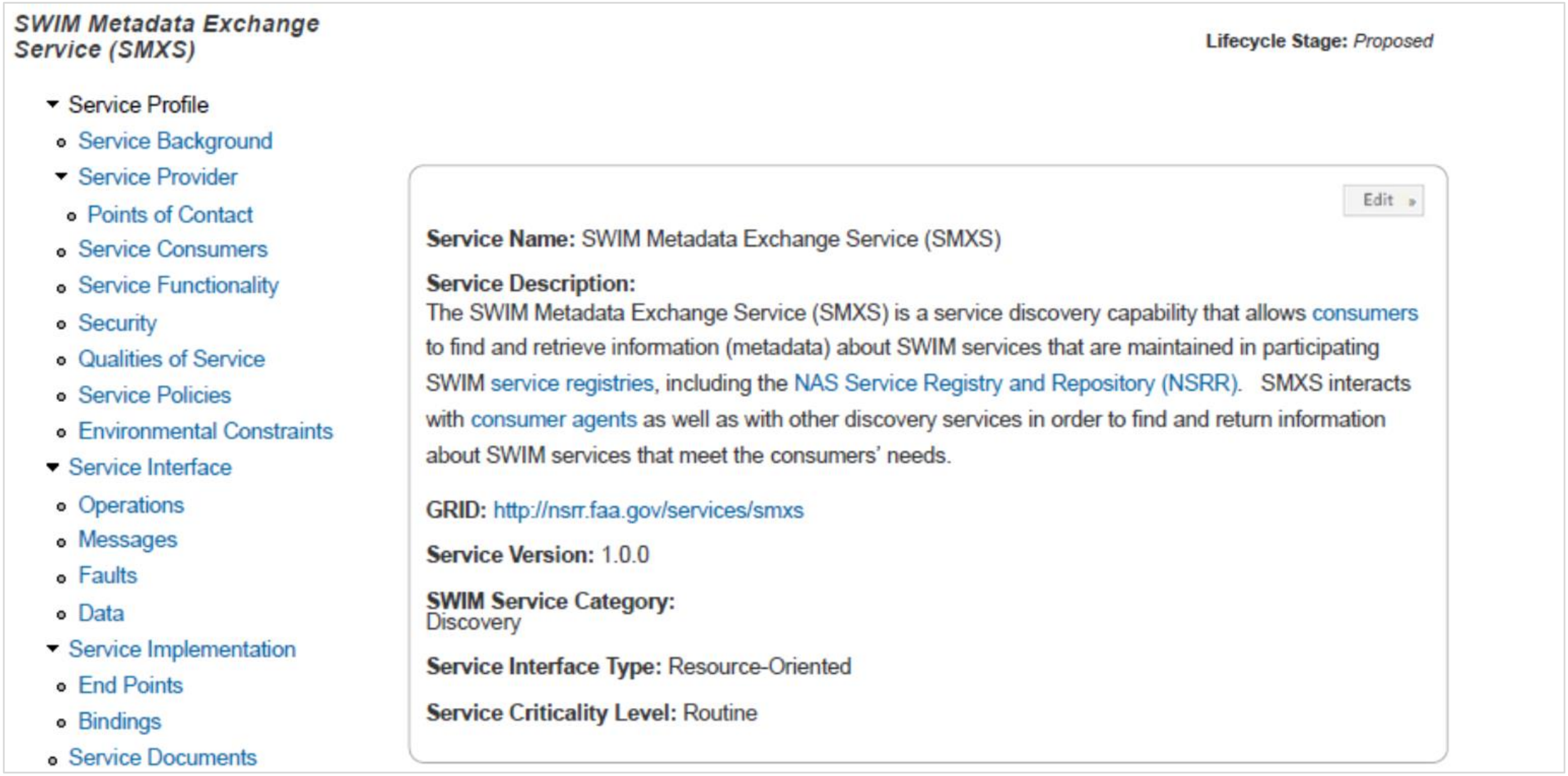
SDS Implementations

- FAA SWIM and Korean Airport Corporation (KAC), Republic of Korea (ROK) are working on development of SDS-compliant services.



FAA's SDS Instance

SWIM Metadata Exchange Service (SMXS), FAA's instance of SDS, is registered in the FAA SWIM service registry, NSRR.



SWIM Metadata Exchange Service (SMXS)

Lifecycle Stage: *Proposed*

- Service Profile
  - Service Background
  - Service Provider
    - Points of Contact
  - Service Consumers
  - Service Functionality
  - Security
  - Qualities of Service
  - Service Policies
  - Environmental Constraints
- Service Interface
  - Operations
  - Messages
  - Faults
  - Data
- Service Implementation
  - End Points
  - Bindings
- Service Documents

Service Name: SWIM Metadata Exchange Service (SMXS)

Service Description:
The SWIM Metadata Exchange Service (SMXS) is a service discovery capability that allows consumers to find and retrieve information (metadata) about SWIM services that are maintained in participating SWIM service registries, including the NAS Service Registry and Repository (NSRR). SMXS interacts with consumer agents as well as with other discovery services in order to find and return information about SWIM services that meet the consumers' needs.

GRID: http://nsrr.faa.gov/services/smxs

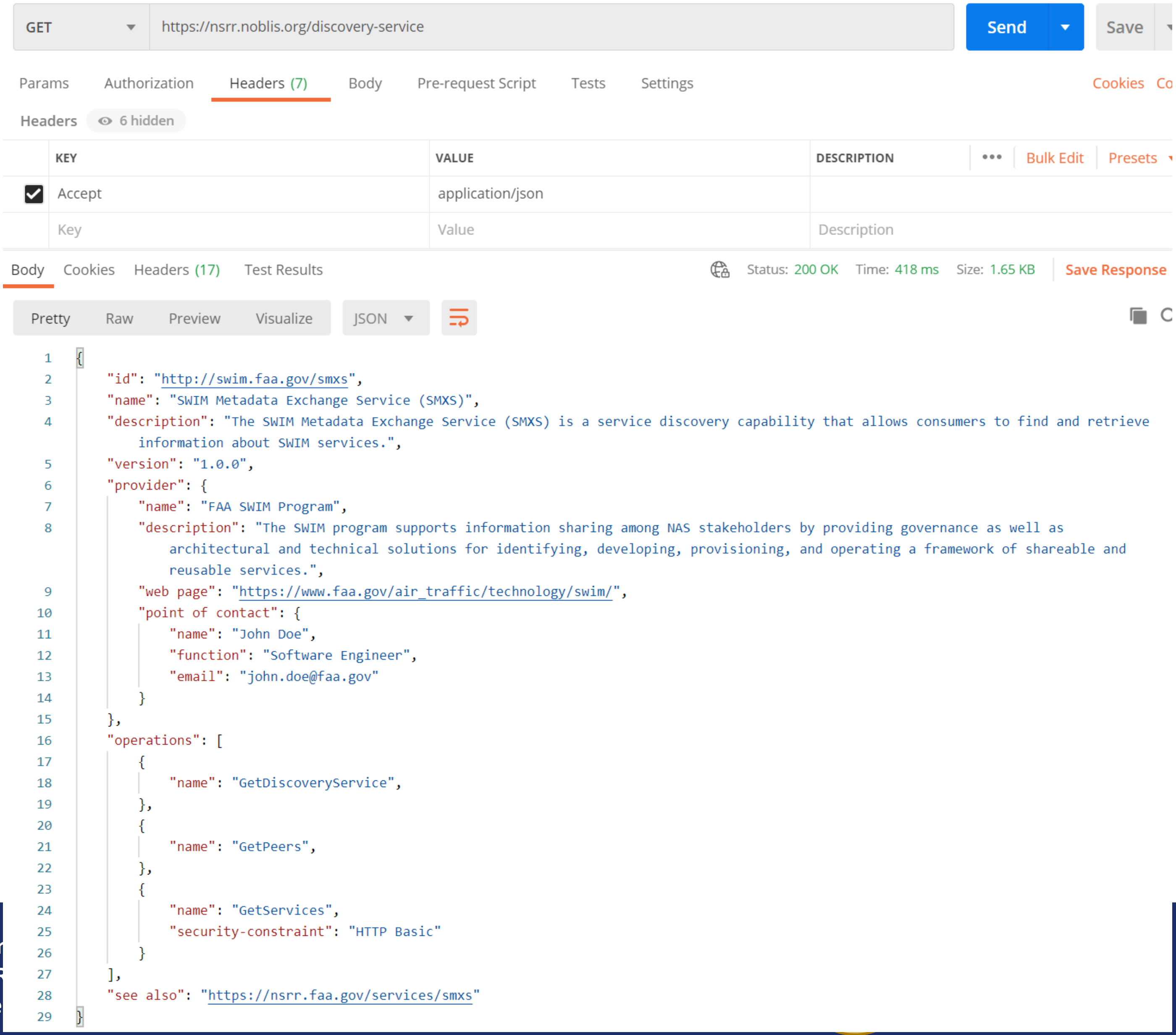
Service Version: 1.0.0

SWIM Service Category:
Discovery

Service Interface Type: Resource-Oriented

Service Criticality Level: Routine

FAA SMXS (test version)



https://nsrr.noblis.org/discovery-service

	KEY	VALUE	DESCRIPTION
✓	Accept	application/json	
	Key	Value	Description

Status: 200 OK Time: 418 ms Size: 1.65 KB

```
{
    "id": "http://swim.faa.gov/smxs",
    "name": "SWIM Metadata Exchange Service (SMXS)",
    "description": "The SWIM Metadata Exchange Service (SMXS) is a service discovery capability that allows consumers to find and retrieve
        information about SWIM services.",
    "version": "1.0.0",
    "provider": {
        "name": "FAA SWIM Program",
        "description": "The SWIM program supports information sharing among NAS stakeholders by providing governance as well as
            architectural and technical solutions for identifying, developing, provisioning, and operating a framework of shareable and
            reusable services.",
        "web page": "https://www.faa.gov/air_traffic/technology/swim/",
        "point of contact": {
            "name": "John Doe",
            "function": "Software Engineer",
            "email": "john.doe@faa.gov"
        }
    },
    "operations": [
        {
            "name": "GetDiscoveryService",
        },
        {
            "name": "GetPeers",
        },
        {
            "name": "GetServices",
            "security-constraint": "HTTP Basic"
        }
    ],
    "see also": "https://nsrr.faa.gov/services/smxs"
}
```

Contacts:

Mark Kaplun (FAA) <mark.kaplun@faa.gov>
Wen Zhu (NIRA) <wzhu@nira-inc.com>