Date: August 2014

UML Profile for the Global Justice Information Sharing Initiative Global Reference Architecture (GRA-UML)

Version 0.50 (Initial submission)

OMG Document Number: gov/2014-08-01

Normative reference: http://www.omg.org/spec/gra-uml/1.0/

Machine readable file(s):

Note: work-in-progress machine-readable files may be found at
https://github.com/GRA-UML/specification/tree/master/GRAUML

Normative:

Non-normative:

USE OF SPECIFICATION - TERMS, CONDITIONS & NOTICES

The material in this document details an Object Management Group specification in accordance with the terms, conditions and notices set forth below. This document does not represent a commitment to implement any portion of this specification in any company's products. The information contained in this document is subject to change without notice.

LICENSES

The companies listed above have granted to the Object Management Group, Inc. (OMG) a nonexclusive, royalty-free, paid up, worldwide license to copy and distribute this document and to modify this document and distribute copies of the modified version. Each of the copyright holders listed above has agreed that no person shall be deemed to have infringed the copyright in the included material of any such copyright holder by reason of having used the specification set forth herein or having conformed any computer software to the specification.

Subject to all of the terms and conditions below, the owners of the copyright in this specification hereby grant you a fully-paid up, non-exclusive, nontransferable, perpetual, worldwide license (without the right to sublicense), to use this specification to create and distribute software and special purpose specifications that are based upon this specification, and to use, copy, and distribute this specification as provided under the Copyright Act; provided that: (1) both the copyright notice identified above and this permission notice appear on any copies of this specification; (2) the use of the specifications is for informational purposes and will not be copied or posted on any network computer or broadcast in any media and will not be otherwise resold or transferred for commercial purposes; and (3) no modifications are made to this specification. This limited permission automatically terminates without notice if you breach any of these terms or conditions. Upon termination, you will destroy immediately any copies of the specifications in your possession or control.

PATENTS

The attention of adopters is directed to the possibility that compliance with or adoption of OMG specifications may require use of an invention covered by patent rights. OMG shall not be responsible for identifying patents for which a license may be required by any OMG specification, or for conducting legal inquiries into the legal validity or scope of those patents that are brought to its attention. OMG specifications are prospective and advisory only. Prospective users are responsible for protecting themselves against liability for infringement of patents.

GENERAL USE RESTRICTIONS

Any unauthorized use of this specification may violate copyright laws, trademark laws, and communications regulations and statutes. This document contains information which is protected by copyright. All Rights Reserved. No part of this work covered by copyright herein may be reproduced or used in any form or by any means--graphic, electronic, or mechanical, including photocopying, recording, taping, or information storage and retrieval systems--without permission of the copyright owner.

DISCLAIMER OF WARRANTY

RESTRICTED RIGHTS LEGEND

TRADEMARKS

COMPLIANCE

OMG's Issue Reporting Procedure

All OMG specifications are subject to continuous review and improvement. As part of this process we encourage readers to report any ambiguities, inconsistencies, or inaccuracies they may find by completing the Issue Reporting Form listed on the main web page http://www.omg.org, under Documents, Report a Bug/Issue (http://www.omg.org/technology/agreement.)

Table of Contents

Figures and Tables

Preface

OMG

Founded in 1989, the Object Management Group, Inc. (OMG) is an open membership, not-for-profit computer industry standards consortium that produces and maintains computer industry specifications for interoperable, portable, and reusable enterprise applications in distributed, heterogeneous environments. Membership includes Information Technology vendors, end users, government agencies, and academia.

OMG member companies write, adopt, and maintain its specifications following a mature, open process. OMG's specifications implement the Model Driven Architecture® (MDA®), maximizing ROI through a full-lifecycle approach to enterprise integration that covers multiple operating systems, programming languages, middleware and networking infrastructures, and software development environments. OMG's specifications include: UML® (Unified Modeling Language™); CORBA® (Common Object Request Broker Architecture); CWM™ (Common Warehouse Metamodel); and industry-specific standards for dozens of vertical markets.

More information on the OMG is available at http://www.omg.org/.

OMG Specifications

As noted, OMG specifications address middleware, modeling and vertical domain frameworks. All OMG Specifications are available from the OMG website at:

http://www.omg.org/spec

Specifications are organized by the following categories:

Business Modeling Specifications

Middleware Specifications

- **CORBA/IIOP**
- **Data Distribution Services**
- **Specialized CORBA**

IDL/Language Mapping Specifications

Modeling and Metadata Specifications

- **UML, MOF, CWM, XMI**
- **UML Profile**

Modernization Specifications

Platform Independent Model (PIM), Platform Specific Model (PSM), Interface Specifications

- **CORBAServices**
- **CORBAFacilities**

OMG Domain Specifications

CORBA Embedded Intelligence Specifications

CORBA Security Specifications

All of OMG's formal specifications may be downloaded without charge from our website. (Products implementing OMG specifications are available from individual suppliers.) Copies of specifications, available in PostScript and PDF format, may be obtained from the Specifications Catalog cited above or by contacting the Object Management Group, Inc. at:

OMG Headquarters
140 Kendrick Street
Building A, Suite 300
Needham, MA 02494
USA
Tel: +1-781-444-0404
Fax: +1-781-444-0320
Email: *pubs@omg.org*

Certain OMG specifications are also available as ISO standards. Please consult http://www.iso.org

Typographical Conventions

The type styles shown below are used in this document to distinguish programming statements from ordinary English. However, these conventions are not used in tables or section headings where no distinction is necessary.

Times/Times New Roman - 10 pt.: Standard body text

Helvetica/Arial - 10 pt. Bold: OMG Interface Definition Language (OMG IDL) and syntax elements.

`Courier - 10 pt. Bold:` Programming language elements.

Helvetica/Arial - 10 pt: Exceptions

NOTE: Terms that appear in italics are defined in the glossary. Italic text also represents the name of a document, specification, or other publication.

Issues

The reader is encouraged to report any technical or editing issues/problems with this specification to http://www.omg.org/report_issue.htm.

0 Submission-related material

0.1 Submission Introduction

The GRA-UML submission team is pleased to present an initial submission to the "UML Profile For GRA (GRA-UML)" Request for Proposal gov/13-09-20.

The IPR mode for this submission is Non-Assert.

Clause 0 of this document contains information specific to the OMG submission process and is not part of the proposed specification. The proposed specification starts with Clause 1. All clauses are normative unless otherwise specified.

0.2 Submission Team

0.2.1 Submitters

Model Driven Solutions, cory-c@modeldriven.com
IJIS Institute, ashwini.jarral@ijis.org

0.2.2 Contributors

- Hidden Symmetry Ltd.
- Open Networks Inc.

0.3 Resolution of Requirements

0.3.1 Mandatory requirements

6.5.1 Submissions shall specify a GRA-UML Logical Profile. The GRA-UML Logical Profile shall be a set of UML stereotypes and properties which support the modeling of SSPs in UML in a technology-independent way. This profile shall support modeling of any content and structure allowed by GRA SSPs, while constraining the modeling of any content and structure disallowed by GRA SSPs, as specified in 6.5.4. The use of the GRA-UML Logical Profile shall result in UML models that are free from dependency on any physical representation (such as XML Schema). In MDA terms, the GRA Logical Profile is a specification of the platform independent model (PIM).	The Logical Services Profile is explained in clause 7 and specified in clause 8.
6.5.2 Submissions shall specify a GRA-UML Profile for SSPs. The GRA-UML Profile for SSPs shall be a set of UML stereotypes and properties which specify the details and metadata of a SSP specification to be produced based on logical GRA model(s). Transformations (discussed in 6.5.3) shall be designed to utilize the Profile for SSPs to parameterize the transformation from the GRA-UML Logical Profile (the PIM) to GRA-conformant SSPs (the PSM), where GRA conformance is as defined in 6.5.4. The SSP profile shall govern the inclusion in a UML model of any information necessary to properly generate GRA-conformant SSP artifacts beyond the information found in the GRA-UML Logical	Instead of a set of stereotypes, the details and metadata for SSP generation are specified by instantiating a model library. This library is explained in clause 7 and specified in clause 9.

profile.	
6.5.3 Submissions shall specify a transformation from UML models using the GRA-UML profiles specified in 6.5.1 and 6.5.2 to the set of artifacts required in a conformant SSP, as defined by 6.5.4. Submissions shall utilize the GRA-UML Profiles to model at least one existing GRA SSP and demonstrate the resulting transformation to an SSP. The SSP produced must be GRA conformant, as defined in section 6.5.4, and the XML Schema set contained within the SSP must validate the same set of exchange documents as the existing SSP IEPD. It is not required that the generated SSP be structurally identical to the existing SSP.	The transformation is done in two phases, as explained in clause 7.2.
6.5.4 The SSPs generated based on models conforming to the GRA-UML profile shall conform to normative GRA specifications referenced in section 6.4.1. These specifications are: • GRA Service Specification Package, v1.0.0 (http://it.ojp.gov/docdownloader.aspx?ddid=1217) • GRA Service Specification Guideline v1.0.0 (http://it.ojp.gov/docdownloader.aspx?ddid=1215) • GRA Web-Services Service Interaction Profile v1.3 (http://it.ojp.gov/docdownloader.aspx?ddid=1173) • GRA ebXML Messaging Service Interaction Profile v1.1 (http://it.ojp.gov/docdownloader.aspx?ddid=1168) • GRA Reliable Secure Web Services, Service Interaction Profile v1.2 (http://it.ojp.gov/docdownloader.aspx?ddid=1134)	The generated SSPs are GRA conformant.
6.5.5 Specifications shall reuse constructs and/or representations from SoaML to express the SSP logical model and will extend SoaML to meet specific requirements of the GRA where required.	SoaML is not used or extended, although some of its general principles are adopted. Instead, normal UML models are used, with just two stereotypes to indicate the providers and consumers of services.
6.5.6 Specifications shall reuse elements and/or representations from UML Profile for BPMN 2 Processes to express the SSP logical model and more specifically the SSP interaction model.	BPMN2 is not used. A process model is not required in order to generate GRA-compliant schemas.
6.5.7 Specifications shall define mappings between GRA, and these industry-standard specifications - OMG SoaML, OASIS SOA-Reference Model, Open Group SOA Ontology.	The revised submission will include such mappings in Annex B.
6.5.8 Specifications shall use OMG-QVT to specify the transformation of the UML logical model to all or part of a SSP.	The Phase-1 transformation uses QVT. Phase-2 uses XSLT and is customizable.

0.3.2 Non-mandatory features

6.6.1 Submissions may specify a "reverse engineering" transformation from a GRA conformant SSP to UML models that conform to the GRA-UML profiles. The reverse engineering shall produce both a "PIM" (A model conforming to the GRA-UML Logical Profile) and a corresponding "PSM" (A model conforming to the GRA-UML Profile for SSPs) such that applying the forward engineering transformations to these models produces a GRA-conformant SSP semantically equivalent to the input.	No such transformation is provided, at least in the initial submission.
6.6.2 The submitters may propose a revision of SoaML which would benefit GRA needs and the community at large.	No such revision is proposed. The GRA logical services profile is similar to a subset of SoaML but simpler and smaller.

0.4 Resolution of Discussion Issues

6.7.1 Submissions shall discuss the relationship of GRA-UML with other ongoing and related GRA standards, the existing GRA specifications and the GRA process.

Defer to revised submission.

6.7.2 Submissions shall discuss their relationship with other relevant standards including but not limited to the Unified Profile for DoDAF/MODAF (UPDM)

Defer to revised submission.

6.7.3 If a GRA SIP for the Representational State Transfer (REST) protocol becomes available during the submission process, submissions may discuss its impact on the submission.

No such SIP exists at present. The two-phase generation approach specified by GRA-UML should be able to incorporate such technology developments.

6.7.4 Submissions shall discuss their relationship with other relevant standards including but not limited to the IEPPV.

There is no substantive relationship with IEPPV. The policy approach specified by GRA-UML is aligned with GRA.

1 Scope

1.1 GRA-UML Background

The Global Reference Architecture (GRA) is an information exchange solution designed to reduce implementation time and costs for state and local justice agencies through reuse of established practices in IT architecture and design. It is developed and maintained under the governance of the Global Standards Council (GSC), part of the Global Justice Information Sharing Initiative (Global) which serves as a Federal Advisory Committee (FAC) and advises the U.S. Attorney General on justice information sharing and integration initiatives.

GRA solutions to information exchange are made up of a combination of the connection method (often Web Services), the exchange language (use of NIEM is encouraged), and the security specifications (encryption at the transport layer, data layer, etc.). These specifications are packaged into a GRA solution that can be customized to meet a community's need for interoperability and information exchange. These packages are called Service Specification Packages (SSPs). A Service Specification Package Template and a number of reference Service Specification Packages are available as part of the GRA and can be found on the GRA website at http://www.it.ojp.gov/gra. The GRA SSPs are used by a number of justice and public safety agencies to define and implement interoperable information sharing services.

Prior to GRA-UML there were no standard based tools which could be used to model and generate GRA SSPs. As a result creating, reusing and implementing GRA SSPs was a very manual process, and requires comprehensive understanding of the GRA.

In GRA-UML, SSPs are generated from models, thus simplifying and improving the consistency of their implementation. The models are created from two viewpoints:

- The Logical Service Model is a platform-independent description of the services automated by the SSP.
- The Annotation Model is a description of the additional metadata needed to interpret the Logical Service Model and generate a complete and conformant SSP.

Each of these models is created using a subset of UML. GRA-UML defines these subsets together with the transformations and workflows used to generate a complete and conformant SSP from the models.

1.2 Intended Users of GRA-UML

GRA-UML enables the construction of an SSP to be distributed amongst architects and developers with different skillsets. The Logical Service Model specifies business requirements for the services, including the various participating individuals, organizations and software systems, the capabilities offered by these participants, and the interactions between them. These are modeled in a technology-agnostic way using a modest and well-defined subset of UML; the modelers need only understand this subset of UML and the business requirements to be modeled. The Annotation Model adds the detail required to generate the complete SSP. This includes metadata, technology and policy choices for the SSP, in cases where these choices cannot be inferred from the Logical Service Model by applying defaults. The annotation modeler needs a good understanding of the structure of SSPs and the various policy and implementation choices available. The final step in the production of an SSP involves the application of XSLT transformations to generate the actual SSP artifacts; these XSLT transformations may be customized by a template developer, expert in XSLT and in the target technology, in cases where the detailed implementation choices provided by the pre-packaged templates require changing.

1.3 GRA-UML Profile, Library and Transformations

A GRA-UML Logical Service Model is created using the Logical Services UML Profile described in Clause 8. This profile defines a subset of UML that includes SOA constructs such as Actors, Use Cases, Components, Ports, Interfaces

etc., and two UML stereotypes, together with modeling conventions about how to use the profile to describe service participants and their interactions.

A GRA-UML Annotation Model is created by using the GRA Annotation Library described in Clause 8. Metadata describing the SSP requirements is created by instantiating types from the Annotation Library as UML InstanceSpecifications, linked together into a UML instance model that portrays the overall structure of the metadata for the SSP.

Given a Logical Services Model and Annotation Model, a *two-phase provisioning* process is used. First, QVT transformations are used to convert the models into a standardized intermediate format that provides all of the metadata needed to construct a complete valid SSP. Secondly, XSLT transformations are applied to create the final SSP artifacts.

Clause 7 is a modeling guide that provides a detailed rationale and description for each of these models, conventions, workflows and transformations.

2 Conformance

All clauses of this specification that are not marked as "informative" are normative, and a conforming implementation shall satisfy all of them.

3 Normative References

The following normative documents contain provisions which, through reference in this text, constitute provisions of this specification. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply.

[MOF] OMG Meta Object Facility (MOF) Core Specification v2.4.2, formal/2014-04-03, http://www.omg.org/spec/MOF/2.4.2/PDF/

[UML] OMG Unified Modeling Language (UML) Superstructure v2.4.1, formal/2011-08-06, http://www.omg.org/spec/UML/2.4.1/Superstructure/PDF/

[OCL] OMG Object Constraint Language (OCL) v2.4, formal/2014-02-03, http://www.omg.org/spec/OCL/2.4/PDF/

[QVT] OMG Meta Object Facility (MOF) Query/View/Transformation v1.2, ptc/2014-03-38, http://www.omg.org/spec/QVT/1.2/Beta/PDF

[GRA] Global Reference Architecture, https://it.ojp.gov/gra

[NIEM] National Information Exchange Model, www.niem.gov

[NIEM-UML] UML Profile for NIEM, http://www.omg.org/spec/NIEM-UML/

4 Terms and Definitions

For the purposes of this specification, the following terms and definitions apply. For terms labelled (GRA), full definitions are found at https://it.ojp.gov/gist/43/The-Global-Reference-Architecture--GRA--Service-Specification-Guideline-V-1-0-0.

IEPD (NIEM)

Information Exchange Package Documentation.

LSM

Logical Service Model.

SOA

Service Oriented Architecture.

SDD (GRA)

Service Description Document.

SIDD (GRA)

Service Interface Description Document.

SSP (GRA)

Service Specification Package.

SIP (GRA)

Service Interaction Profile.

WSDL

Web Services Description Language, a W3C standard XML-based interface definition language used for describing the functionality offered by a web service.

XHTML

Extensible HyperText Markup Language.

5 Symbols

There are no symbols defined in this specification.

6 Additional Information

6.1 Acknowledgements

The following entities have played a significant role in driving the development of this specification:

- Submitters
  - Model Driven Solutions
  - The Integrated Justice Information Systems Institute (IJIS)
- Government Stakeholders
  - US Department of Justice
- Contributors
  - Hidden Symmetry Ltd.
  - Open Networks Inc.

7 GRA-UML Modeling Guide

7.1 GRA SSP Overview

The purpose of a GRA-UML tool is to enable GRA architects and developers to model the business and technological requirements for a GRA SSP using UML, and to provision the complete valid GRA SSP as an output. This section briefly describes the contents and structure of a GRA SSP: it is necessary to understand these in order to understand the process used to create them. Full details of the GRA SSP can be found at https://it.ojp.gov/gist/43/The-Global-Reference-Architecture--GRA--Service-Specification-Guideline-V-1-0-0.

The files that constitute a valid GRA SSP must be arranged in the predefined folder structure shown in Figure 1.

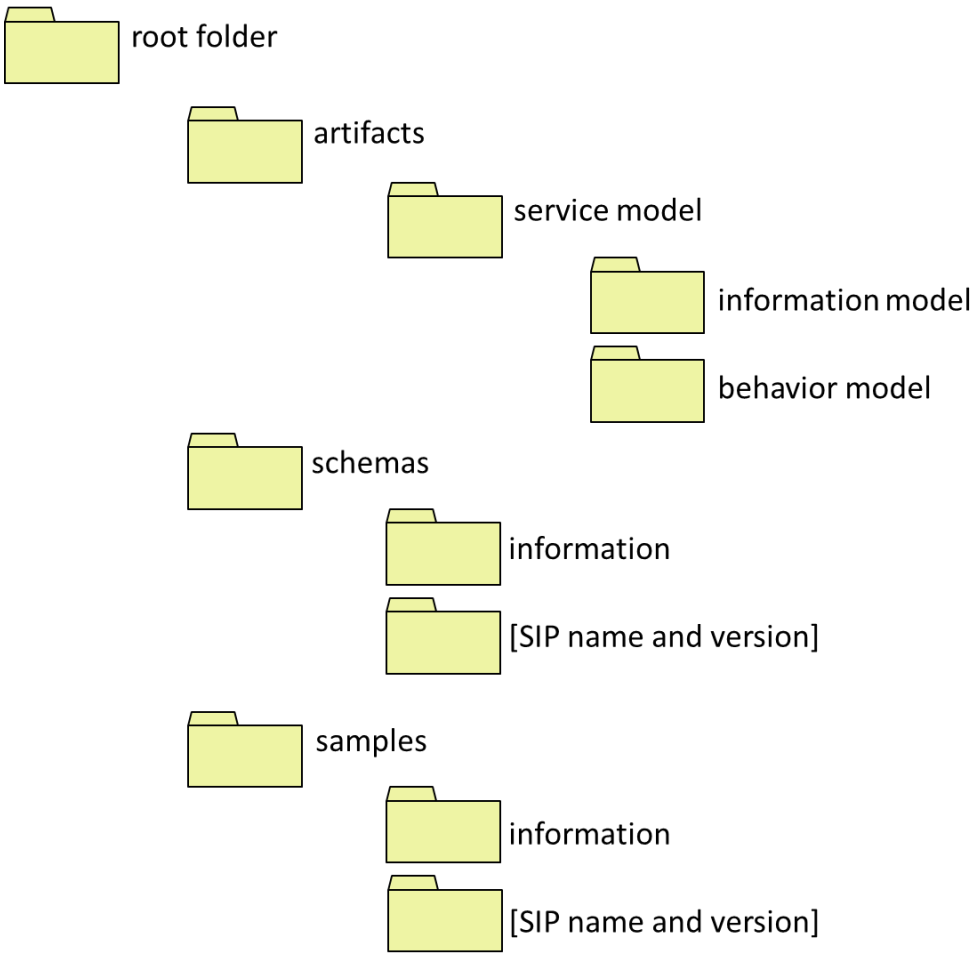

Figure 1 Mandatory SSP folder structure

A complete GRA SSP (Service Specification Package) comprises at least the following files:

- A Service Description, manifested in a Service Description Document (SDD). This is a human-readable file contained in the artifacts folder. For GRA-UML, XHTML format is generated.
- One or more Service Interface Descriptions, manifested in Service Interface Description Documents (SIDDs). Each Service Interface Description must conform to one or more Service Interaction Profiles (SIPs). SIDDs are human-readable files contained in the artifacts folder. For GRA-UML, XHTML format is generated.

- A Service Metadata file. This is called metadata.xml and contained in the root folder.
- A Service Catalog. This consists of two files, a machine-readable catalog.xml and a human-readable catalog.html, both contained in the root folder.
- A Service Specification Information Model, which must be a NIEM-conformant IEPD. This is packaged in a zip file or folder and contained in the artifacts/service model/information model folder.
- A Service Specification Behavior Model. In the case of GRA-UML, this is the Logical Service Model (see clause 7.3). It is contained in the artifacts/service model folder.
- Schemas. As per recent GRA guidance, Schemas not included in an IEPD that are used in the Service Specification are contained in the schemas/information folder. Web Service descriptions in WSDL format that are generated from the annotated GRA-UML model are contained in the schemas/[SIP name and version] folder, where the name of the folder corresponds to the SIP used to generate the WSDL.

Other optional files may be included and the folder structure may be extended according to rules specified by GRA. Such optional files and folders are not consumed or provisioned by GRA-UML and their description is omitted from this specification.

7.2 Two-Phase SSP Provisioning

GRA-UML has two potentially conflicting goals: to generate a fully GRA-conformant SSP, while providing flexibility in how SSPs are produced, what technologies are used and how these technologies are used. The reason for this flexibility as that GRA is essentially open ended with respect to the service interaction profiles (SIPs) used as well as what specific technologies are used within a SIP. In addition, SSP architects frequently have additional requirements or styles with respect to the generated documentation.

To resolve this dilemma the generation and provisioning of the final SSP is divided into two steps, called *Phase-1* and *Phase-2* as shown in Figure 2 below. Phase-1uses a standardized algorithm and is implemented in QVT; Phase-2 is customizable and is implemented using XSLT.

The GRA-UML services architect creates a model of the SSP containing two main aspects that separate concerns between the logical services and the GRA-specific requirements and metadata:

- The Logical Service Model - a platform-independent specification of the services to be automated by an implementation of the SSP, in terms of SOA constructs such as Actors, Use Cases, Components, Ports, Interfaces etc. It contains normal UML structures that are used to define services by applying simple conventions and a couple of GRA-specific stereotypes.
    - *Note: this same UML model could be used and extended for purposes other than producing GRA specifications, such as producing implementations.*
- The Annotation Model – a modeled specification of GRA-specific metadata, including technology and policy choices for the SSP, in cases where these choices cannot be inferred from the Logical Service Model by applying defaults.

Phase-1 uses a standardized algorithm to generate an intermediate format representation of the SSP. It uses the OMG's QVT (Query/View/Transformation) to transform the GRA-UML model into a standardized XML-based format that contains all of the data required to generate the complete SSP. This intermediate format uses XMI (XML Metadata Interchange) corresponding to the OMG's EMOF (Essential MOF) specification. XMI is an XML format with a schema that provides a convenient and standardized intermediate format for consumption by Phase-2.

Phase-2 uses an XSLT transformation template to generate the complete final SSP.

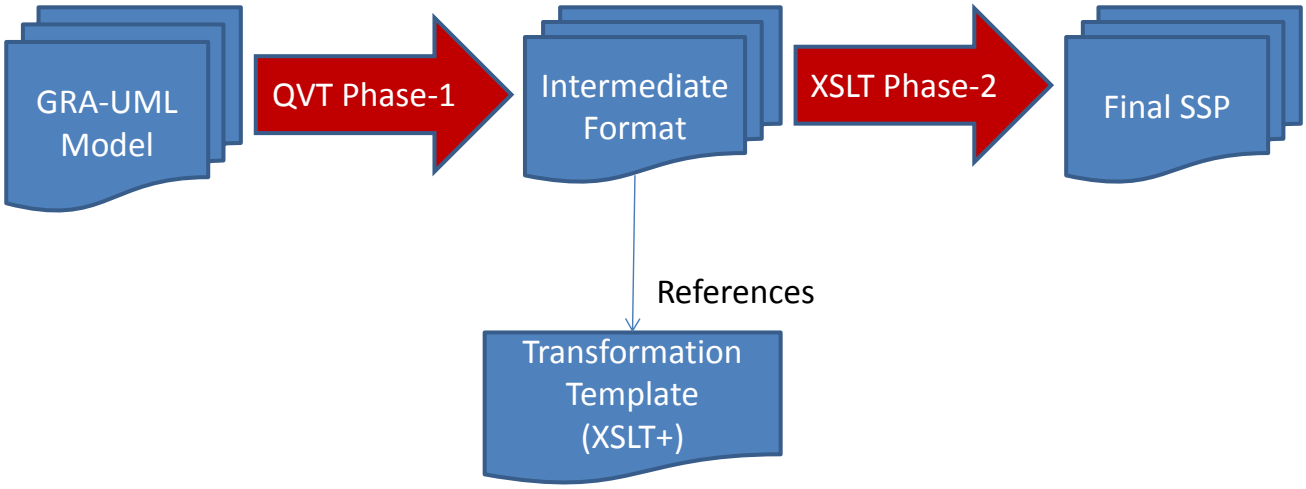

Figure 2 Two-phase provisioning

The envisaged workflow for a user of GRA-UML is as follows:

- User loads GRA-UML, which consists of the Logical Services Profile and the GRA Annotation Library, into a UML tool.
- User uses the Logical Services Profile to create the Logical Services platform-independent model – the PIM.
- User loads a starter annotation file which contains InstanceSpeifications classified by classes in the GRA Annotation Library (a starter file is provided as part of the specification).
- User edits and adds annotation elements and properties, and connects annotation elements to elements in the PIM using Realization and Usage dependencies.
- User invokes an action which causes the tool to, for each ServiceDescription:
  - Generate the filled out intermediate format annotation instance file (annotations.xmi) and the other artifacts produced by stage one (catalog, etc)
  - Invoke stage 2 to produce the final SSP artifacts and to place all artifacts correctly in the folder organization.

7.3 The Logical Service Profile

A UML Profile is a means to customize the general-purpose modeling language UML so that it can be used to create models that relate to a particular domain. A UML Profile has the following characteristics:

- It may define a subset of UML which is applicable to the intended domain.
- It may define a set of Stereotypes. These are labeled bundles of information which may be applied to particular kinds of UML element, carrying additional metadata relevant to the intended domain.
- It may define a library of domain-specific types for use within the model.

GRA-UML does all three of these. It defines:

- A subset of UML, used for modeling services at a logical, platform-independent level of abstraction.
- Two stereotypes, needed for distinguishing between service consumers and providers.
- The GRA Annotation Library, used to create elements that represent the detailed metadata required to provision SSPs. This library is described in clause 7.4 and specified in detail in clause 9.

This sub-clause describes the use of the first two of these constituents, which are together known as the Logical Services Profile.

Logical Service Models (LSMs) may contain any kind of UML element, but only those elements that are consumed by the transformations defined in this specification will influence the machine-generated outputs provisioned to the SSP. Logical Service Models use existing UML concepts in familiar ways and should be easily understood by typical UML users.

The Logical Service Profile is similar to, but considerably simpler than, OMG's existing SoaML specification. It contains just two stereotypes called «Provider» and «Consumer» which are intended to be applied to UML Associations as shown in 7.3.2. No additional metadata is required to model the Logical Services Model using normal UML elements as shown in the following sections.

7.3.1 Actors

Actors are the business participants in the SSP's subject community. They are typically people or organizations. Figure 3 shows Actors in an inheritance hierarchy.

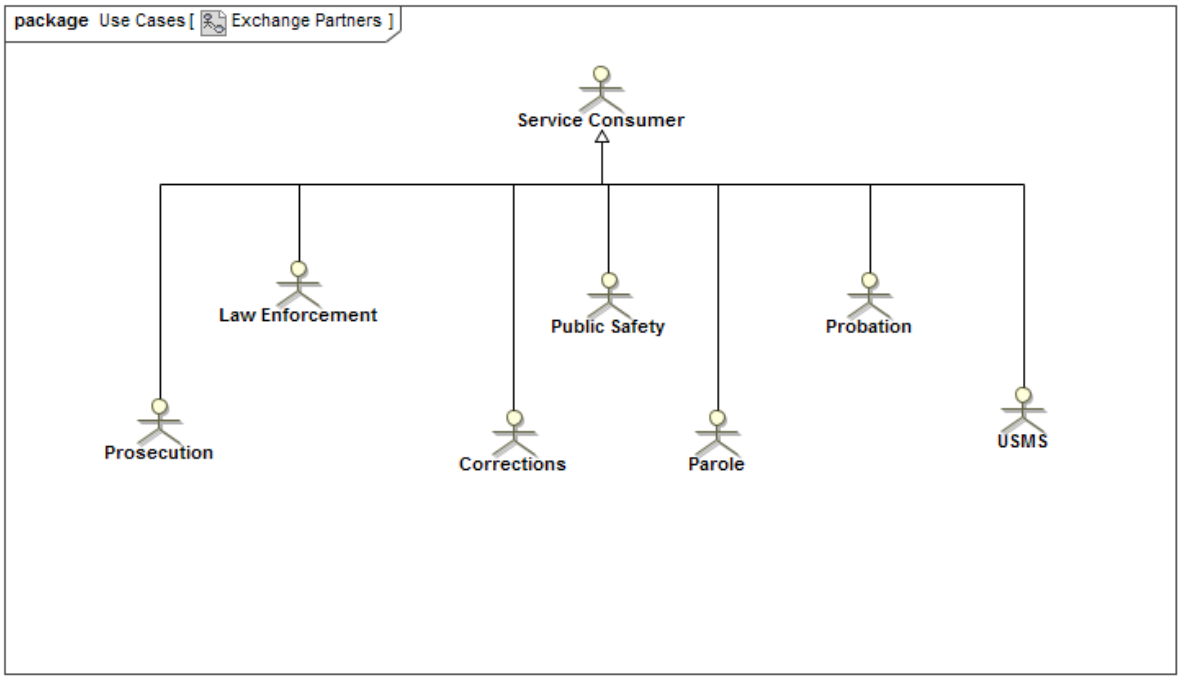

Figure 3 Actors

7.3.2 Use cases

The *real-world effects* of services are modeled as UseCases involving Actors. Each UseCase (oval) is a real-world effect and a business interaction. Each association between a UseCase and an Actor may be marked with a stereotype to indicate whether the Actor is a «Provider» or «Consumer» of the service. Figure 4 shows a simple example.

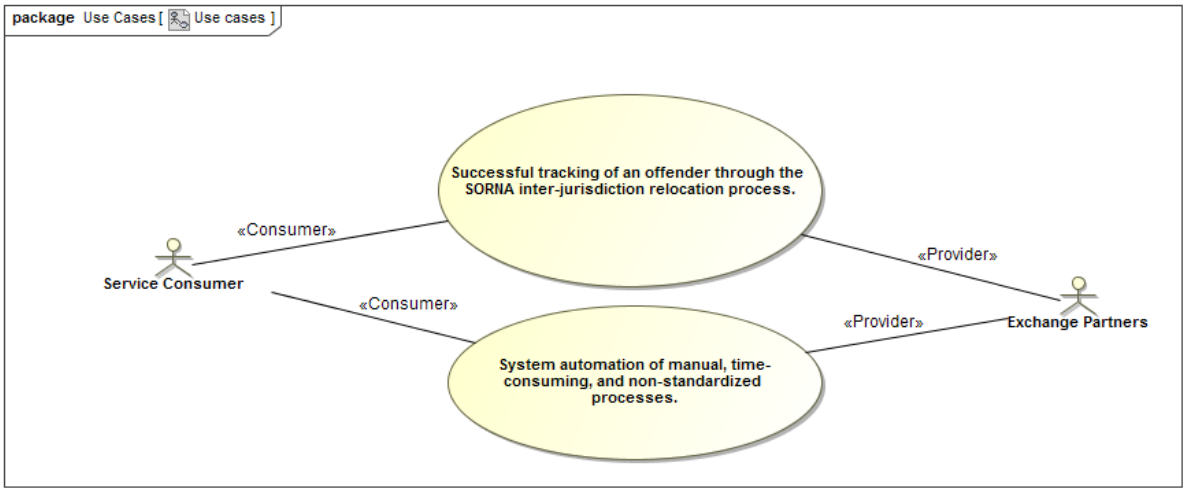

Figure 4 Use Cases

7.3.3 Interfaces

UML Interfaces define the operations and signal receptions that service providers implement to expose a service. The arguments of operations directly reference [NIEM-UML] message types as their content. Figure 5 shows a simple example.

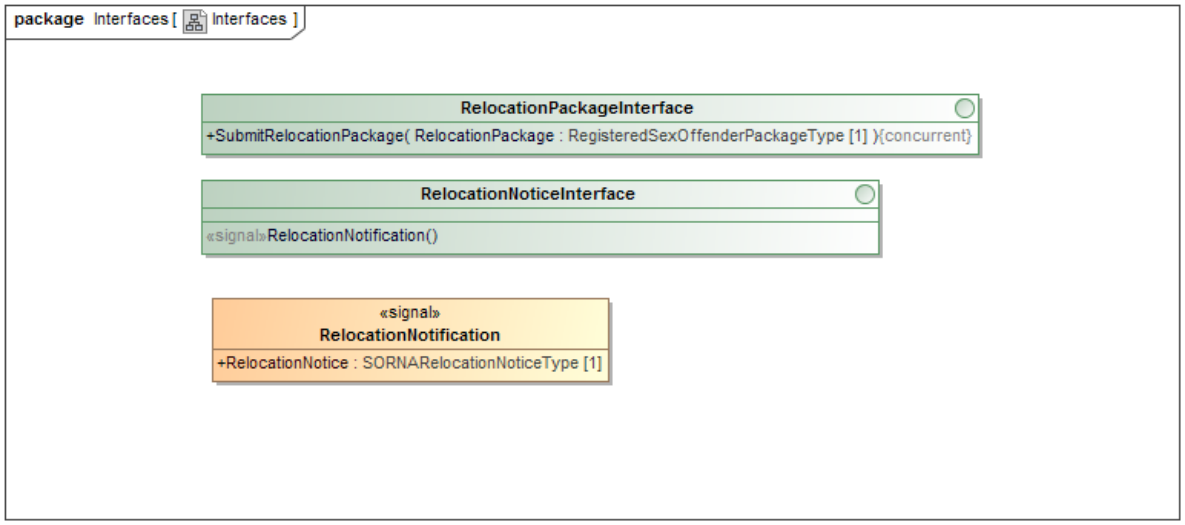

Figure 5 Interfaces

7.3.4 Components and ports

UML Components represent the service provided by an Actor. Each Component has a set of Ports that provide or use UML Interfaces. A Component may realize an Actor, to signify in the model that it represents a service provided by that Actor.

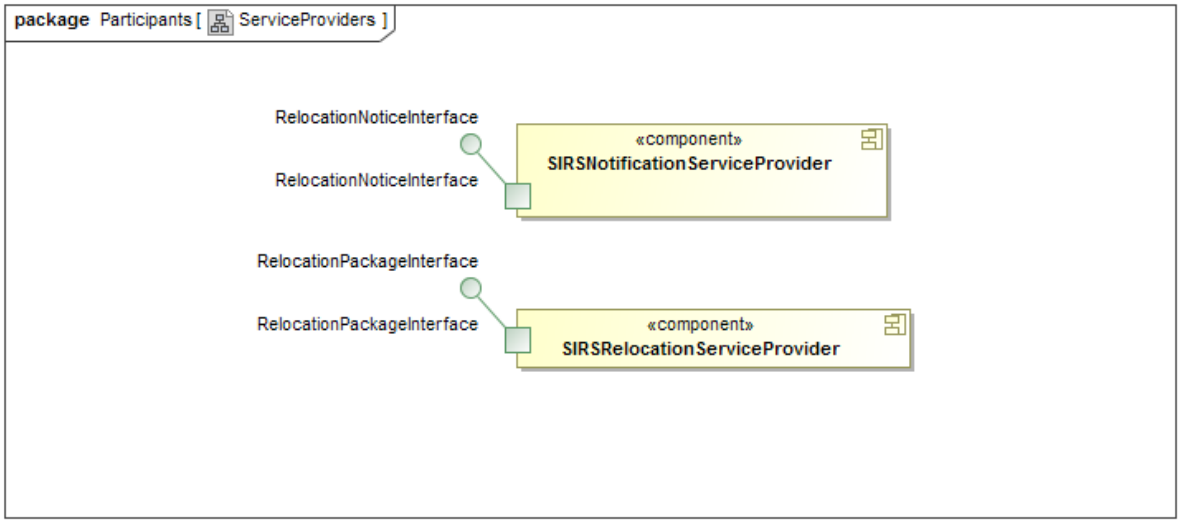

Figure 6 Components and Ports

7.3.5 Collaborations & Interactions

Collaborations are used to define the *community* for which the SSP is intended, providing a property for each Actor and service Component playing a role in that community.

Figure 7 Collaboration

The Collaboration that represents the community owns a set of Interactions that define the choreography of exchanges between the Actors to implement the UseCases. Figure 8 shows examples.

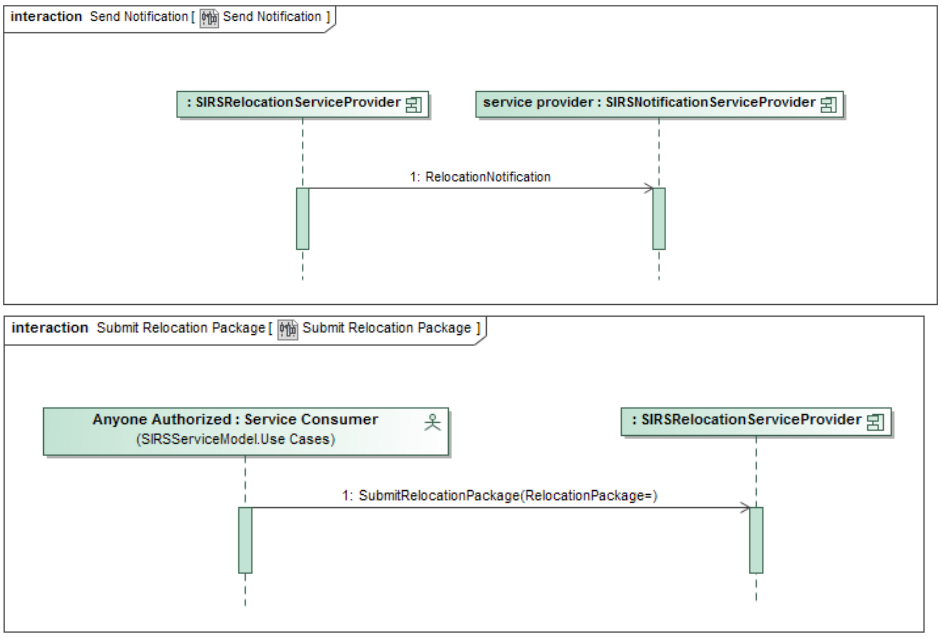

Figure 8 Interactions

7.4 The GRA Annotation Library

7.4.1 Overview

The GRA Annotation Library performs two roles in the overall workflow, and hence exists in two similar forms that differ in their details. The two forms are called the Annotations and the Intermediate Model. Both libraries contain the same UML classes and enumerations, with a few additional classes in the Intermediate Model. The Annotations are designed to simplify the modeling task by avoiding duplication of information and assuming that defaults are applied wherever possible. The Intermediate Model is designed to provide a convenient representation for consumption by Phase-2, which uses XSLT. So, for example, instances of the Intermediate Model are nested and avoid cross-references, whereas no such restriction applies to the Annotations.

The purpose of the Annotations is to provide the ingredients for the end user to model the GRA-specific metadata needed for generating the SSP. The end-user accomplishes this by creating UML InstanceSpecification elements that instantiate classes in the Annotations.

> *Note: For readability in describing the use of the annotation library, the word "instance" will be used to mean UML InstanceSpecification, unless this interpretation would cause confusion. See Figure 9 below for examples.*

Having created instances, the user gives values to properties in those instances, links them together, and provides Realization dependencies between specific instances and elements in the Logical Service model, and Usage dependencies to the Information Model.

The purpose of the Intermediate Model is to act as a metamodel for the intermediate annotations.xmi format. The Phase-1 transformation consumes the end-user's model created using the Annotations, and creates (in memory) additional InstanceSpecifications, links and property values that represent all of the annotations needed to generate the SSP. The resulting model is then serialized as XMI in OMG EMOF format, where each UML InstanceSpecification is serialized as an EMOF instance, and UML links and slots serialized as EMOF properties.

Although it would simplify this specification if these two libraries were identical, this would place an additional burden on the GRA-UML modeler to know about and ignore those elements that are only used in the Intermediate Model. The authors of GRA-UML consider this burden to be excessive, and hence have decided to keep the libraries separate, although for convenience they are documented together in Clause 9 of this specification.

The Annotation Library, in both of its forms, is specified in two packages: GRAAnnotationModel, containing platform-independent annotations, and GRA_WSDL, containing WSDL-specific annotations.

The pivotal class in the Annotations is ServiceDescription. The user creates an instance of this class to represent the overall service description which ultimately leads to the creation of an SSP. This ServiceDescription instance should be related to a Collaboration in the Logical Service Model by a UML Realization, signifying the community that the ServiceDescription relates to. It should also be related to the corresponding information model – one or more NIEM-conformant IEPDs – by means of UML Usages. Linked to the ServiceDescription is a set of instances that represent the metadata describing capabilities, requirements, exchange partners and so on; these instances, together with the properties of the ServiceDescription instance, are used to drive generation of the SDD documentation.

Accompanying the ServiceDescription instance are ServiceInterfaceSpecification instances that correspond to the Service Interfaces of the SPP. Each of these has properties and related instances that carry metadata to drive the generation of documentation. It also has related Service instances and a ServiceInteractionProfile instance that together drive the generation of WSDL schemas. As far as possible, default generation rules are applied, so that the user only needs to model exceptions from the default situation.

Example

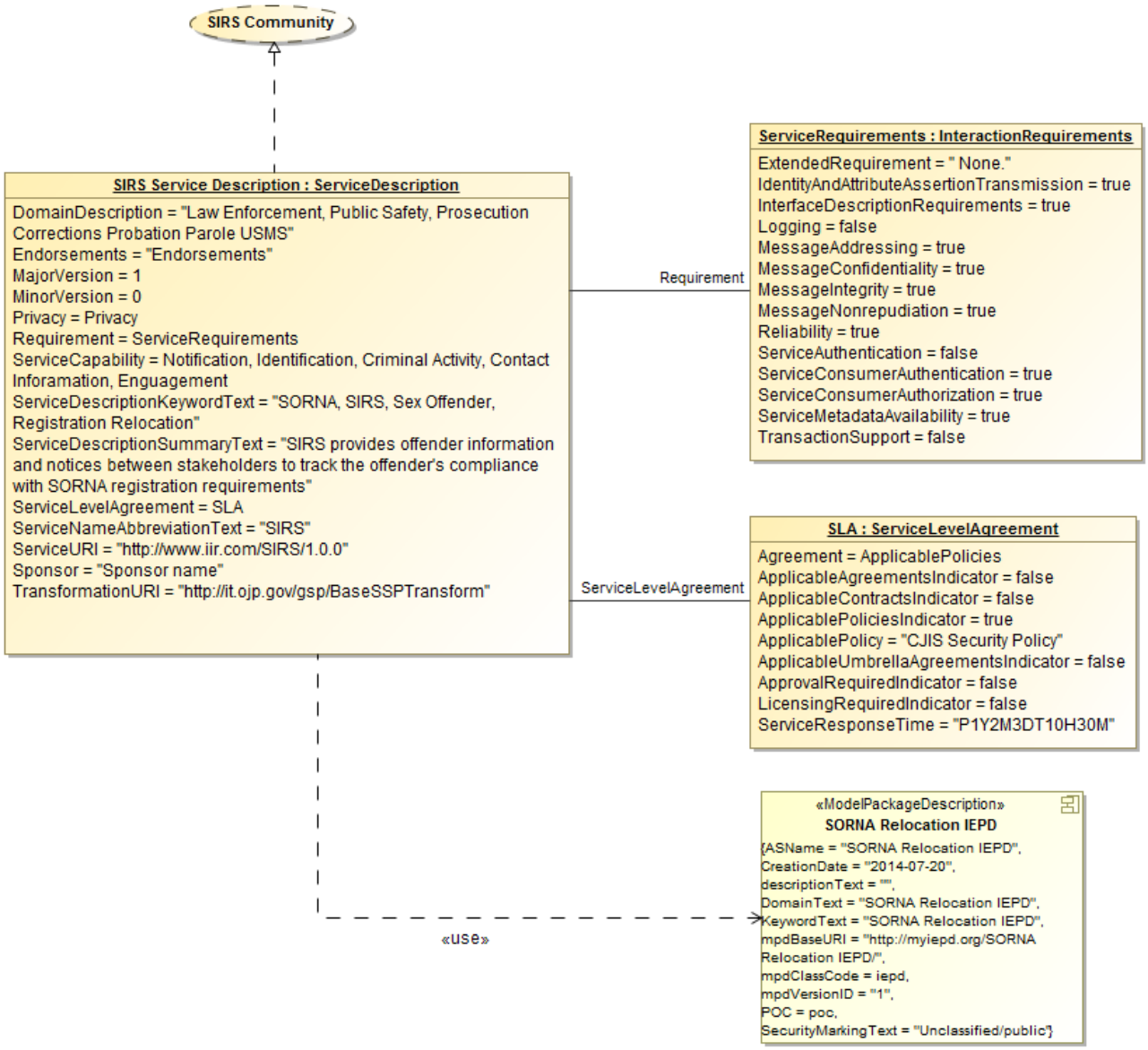

Figure 9 Example annotations

Figure 9 shows a partial example to illustrate some of these principles. A much more complete and comprehensive example appears in Annex A:

In Figure 9, SIRSServiceDescription is an instance of the class ServiceDescription, using the UML notation for InstanceSpecifications. This contains values for the properties defined in the class ServiceDescription, such as ServiceURI = “http://www.iir.com/SIRS/1.0.0”.

Some of the properties are also represented as links. In the example, the property ServiceLevelAgreement refers to the instance named SLA, an instance of the class ServiceLevelAgreement. The property Requirement is similarly shown.

Also shown on the diagram is a Realization from the SIRSServiceDescription instance to the Collaboration called SIRS Community. This is a part of the Logical Service Model which contains elements as described in clause 7.3. The Realization is notated as a dashed line with open triangular arrowhead.

Additionally a Usage dependency connects the SIRSServiceDescription instance to a ModelPackageDescription component from the Information Model. This component is constructed using the UML Profile for NIEM, which defines the «ModelPackageDescription» stereotype shown on the component.

Shown below is a fragment of the annotations.xmi file that corresponds to this example. The top-level element of this fragment corresponds to the SIRSServiceDescription instance. Within that element are attributes that correspond to many of the properties defined by the model. There is one attribute shown – serviceDescriptionURI – that does not appear in Figure 9; this value is derived by Phase-1 and only appears in the Intermediate Model. There is also a ModelReference element, also derived by Phase-1, resulting from the Realization between the SIRSServiceDescription instance and the SIRS Community collaboration. Such derivations are discussed in the next section.

```
<graa:ServiceDescription
       xmlns:xmi=http://www.omg.org/spec/XMI/20110701
       xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
       xmlns:graa=http://ijis.org/GRA/Annotations
       xmi:id="_0"
       name="SIRS Service Description"
       serviceUri=http://www.iir.com/SIRS/1.0.0
       serviceNameAbbreviationText="SIRS"
       serviceDescriptionSummaryText="SIRS provides offender information and notices
between stakeholders to track the offender's compliance with SORNA registration
requirements"
       domainDescription="Law Enforcement, Public Safety, Prosecution Corrections
Probation Parole USMS"
       transformationUri=http://it.ojp.gov/gsp/BaseSSPTransform
       majorVersion="1"
       serviceDescriptionUri="artifacts/SIRS_SDD_v_1.0.0.docx">
       <modelReference
              xmi:id="_0.modelReference"
              name="SIRS Community"
              documentation="Stakeholders"
              elementId="_17_0_5_1_7b3022e_1401746624193_446086_4974">
              <model
                     xmi:id="_0.modelReference.model"
                     modelUri="platform:/resource/Examples/SIRS/SIRS_GRA_EXAMPLE/SIRS-
              Service.uml"
                     label="SIRS-Service"/>

       </modelReference>

       ……

</ graa:ServiceDescription >
```

7.4.2 Derived properties

Many properties in the intermediate model are derived by Phase-1. For example, the property GRAServiceAnnotationBase::ModelReference is derived for any instance that has a Realization to an element in the Logical Service Model. The type of this property is the class ModelReference, which encapsulates a set of metadata used to identify and describe the realized logical service element. This property is derived by Phase-1 and emitted into the intermediate annotations.xmi for consumption by Phase-2. The Annotations contains neither this property nor the type ModelReference: all the modeler needs to do is create the correct Realization.

Another example involves Documentation properties. In the user's model, text may be entered as documentation into a comment attached to the annotation instance: many UML tools distinguish the first comment of a model element through a special documentation field in the user interface. Alternatively the appropriate documentation might be found as a comment attached to a logical service element realized by the annotation instance. Either way, Phase-1 extracts the text from the appropriate comment and places it into a derived Documentation property for the Phase-2 XSLT to consume.

Table 1 specifies all of the properties that are derived by Phase-1, together with specifications of their derivations.

7.4.3 Defaults

To make the modeling process simpler and also to provide for flexibility in the technology and documentation artifacts produced, GRA-UML utilizes defaults at multiple levels. The following sections are intended to clarify the default mechanisms.

Some of these defaults are processed in Phase-1 whereas others are processed in Phase-2. Note that using the "template" mechanism of Phase-2, the Phase-2 defaults many be modified by the GRA-UML architect by providing an alternative template whereas the Phase-1 defaults are specified by the GRA-UML standard and are not modifiable.

Defaults fall into the following categories:

- Service hierarchy defaults, expanded in Phase-1
- Interaction Requirements defaults, expanded in Phase-2
- Property value defaults, provided in Phase-2

7.4.4 Service hierarchy defaults

A service specification and the typical WSDL artifacts produced from it follow a hierarchical pattern of:

- Service interface specification
    - Service
        - Port
            - Interface
                - Operation or signal
                    - Parameter
                        - Message

The *output* from Phase-1will have an annotation instance for *every* service, port, interface, operation, parameter and message defined by the logical service model. These will correspond to the

- Component realized by the service
- Ports on those service components
- Interfaces provided or used by those ports
- Operations and signals contained within those interfaces
- Parameters on the operations
- Messages that are the data types of the parameters

It is not necessary for the user to construct the corresponding hierarchy of annotations corresponding to the complete LSM: Phase-1 does that, and will provide default annotations for every element in the hierarchy. So in the simplest case the user only needs to specify the Service annotation, and everything else will be generated from the structure of the Logical Service Model.

For more control the user may override what Phase-1 does, by providing *explicit* annotations and *defaults*.

The user may create a corresponding explicit hierarchy of annotation instances: Port, Interface, Operation, Parameter, Message, where each annotation instance must connect via a Realization to its corresponding element in the LSM. In this case, Phase-1 will use the explicit annotations wherever they are provided.

Alternatively, the user may create *default* metadata for any of these elements. These are modeled using properties called PortDefault, InterfaceDefault, OperationDefault, ParameterDefault and MessageDefault, which can be set by the user on any higher level of the hierarchy – so for example an OperationDefault can be specified for a Port, but not vice-versa. If

there is just one such default set on an element, and that default is not marked to realize anything, then that default metadata will be expanded for every corresponding subsidiary element in the intermediate file. A default may also be marked to realize a particular LSM element, in which case that default will be expanded as annotations for only that realized element.

None of these default properties appear in the intermediate annotations.xmi, since they would have all been expanded in Phase-1. The actions of Phase-1 also cause some differences in lower multiplicity bounds between the Intermediate Model and Annotations: where zero elements are required from the modeler, at least one element may be required in the intermediate form. All of these differences between the libraries are specified in Clause 9.

Example

The SIRS example has a default for "RelocationPackageInterface" as follows:

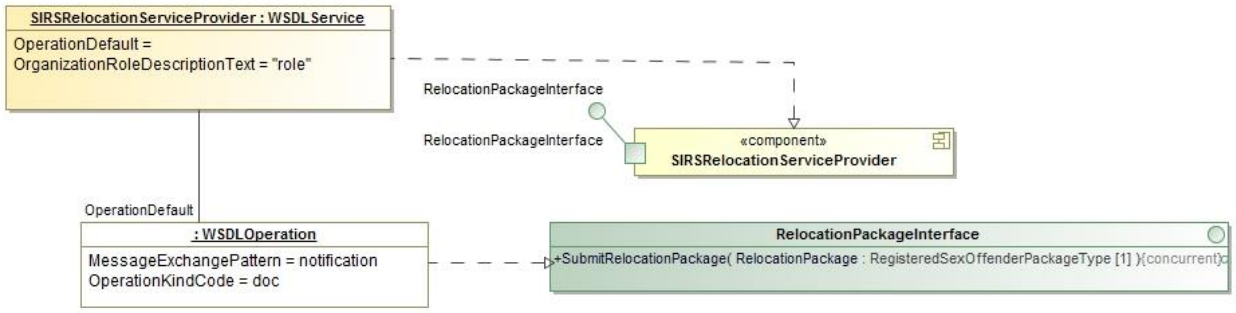

Figure 10 Example operation default

This shows an operation default that realizes the operation SubmitRelocationPackage. This specifies that the SubmitRelocationPackage operation will have a MessageExchangePattern of "notification" and and OperationKindCode of "doc".

If there were no realization from this default, then it would apply to all operations within the hierarchical scope of SIRSRelocationServiceProvider.

The user could also have created an explicit hierarchy of WSDLPort, WSDLInterface and WSDLOperation annotations, explicitly realizing respectively the Port called RelocationPackageInterface, Interface called RelocationPackageInterface, and Operation called SubmitRelocationPackage. This would have the same effect.

7.4.5 Interaction requirements defaults

Interaction requirements specify the business requirements for a service interaction. Interaction requirements are specified using the following class:

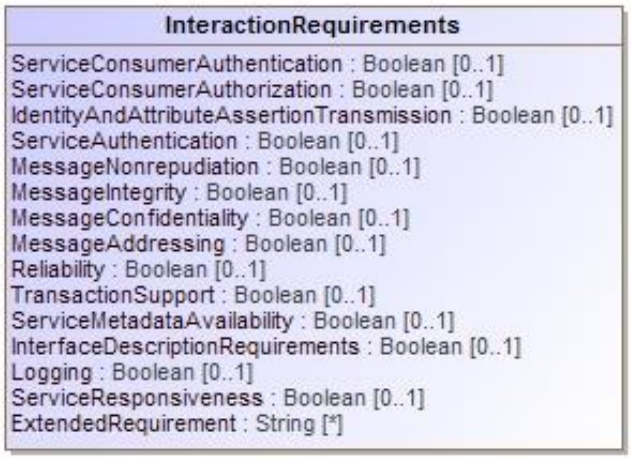

Figure 11 InteractionRequirements

Interaction requirements are meaningful through the full hierarchy of an SSP, starting with the "Service Description" and proceeding down the service hierarchy described above.

The InteractionRequirements specified at the level of the SSP are considered business requirements and are included in the GRA SDD document. These requirements also serve as a default for all the lower levels of the hierarchy, but any level of the hierarchy may override the more general requirements and specify the interaction requirements for that level.

If the interaction requirements are the same for the entire SSP there is only the need for the one InteractionRequirement annotation element. However it is common for specific ports, interfaces or operations to have more specific needs. To define these more specific needs a "Requirement" is linked to the more specific element and the requirement properties filled in.

Any property not specifically set to true or false in any InteractionRequirement element will acquire its value from a higher level, or if there is no such value, a default value for the property as specified by Table 2.

Phase-1 will only output the InteractionRequirements specifically provided by the model; the processing of these defaults is handled by Phase-2.

7.4.6 Property value defaults

There are many elements that are required by the WSDL or GRA specification that are typically filled in with "boilerplate" or other typical values. To simplify the model and also to provide flexibility as to the contents of this boilerplate, certain required properties may be omitted from the service specification. Phase-1 will fill in values that can be determined from the PIM. Phase-2 will then fill in any missing values with "boilerplate". Any value that is specifically provided in the GRA model will take precedence over the boilerplate default. The GRA Phase-2 template contains a specific section for these defaults for easy modification [TBD].

Table 1 details the property values that will be calculated by Phase-1, if no value is provided by the modeler. Table 2 details the property values that will be used by Phase-2, if no value is provided by Phase-1. The end-result should be a valid GRA SSP with all mandatory values included.

Example

The GRA specification requires a "Security" section. Many SSPs use a standard text of the form:

> The service implementation must follow the Service Interface Description Document (SIDD) to meet the Security Requirements.
>
> The service must adhere to the rules of the CJIS Security Policies regarding the Advanced Encryption Standard (AES) level of encryption.

If the “Security” property of a ServiceDescription has no value provided, Phase-2 will insert the above text.

7.4.7 Compositions

XSLT transformations work best on hierarchically-structured documents, and Phase-2 is an XSLT that consumes the output of Phase-1. So the intermediate form is designed to be hierarchical, and this is represented in the Intermediate Model by numerous compositions. From the modeler’s perspective, however, it may be inconvenient to structure the annotations hierarchically: the same annotation may be shared between several elements, and this is most simply represented simply by linking them together. So in the Annotations there are no compositions; one of the tasks of Phase-1 is to duplicate annotation elements shared in the user’s model so that they are hierarchically represented in the intermediate model.

An example of this approach is the property GRAServiceAnnotationBase::InteractionRequirements. This allows the attaching of an instance of InteractionRequirements at any level of the model from the complete ServiceDescription down to the individual Message. This instance provides a bundle of metadata specifying interaction requirements for the element to which it is attached – see the description in 7.4.5, above. The user might share a single instance of InteractionRequirements amongst, say, a set of Operations; Phase-1 processing would replicate that instance so that a copy is hierarchically contained with the metadata for all of those operations in the intermediate annotations.xmi.

Clause 9 specifies where these compositions differ between the libraries.

8 GRA-UML Logical Services Profile

8.1 UML subset

GRA-UML Logical Service Models are normal UML models. Any UML construct can be used within a LSM. However, only certain elements within the LSM are consumed by the SSP provisioning process. The following section documents those elements. Other elements may be used for documentation purposes, or as input to other provisioning processes outside the scope of this specification. The complete Logical Service Model is itself part of the provisioned SSP, appearing in the artifacts/service model folder.

8.2 Logical service classes

This section specifies the UML classes whose instances affect Phase-1 of SSP provisioning. These are classes from the complete merged L3 version of UML 2.4.1, in which each is fully identified by their name, so for example the xmi definition of Actor may be found at http://www.omg.org/spec/UML/20110701/UML.xmi#Actor.

Elements in the model that do not have an effect documented here are ignored by Phase-1.

8.2.1 Actor

A UML Actor represents a participant in the SSP subject community. Actors are used by Phase-1 to populate the property ServiceDescription::ExchangePartner. Actors may be directly realized by Participants in the annotation model, or may be contained in realized Packages (see 8.2.3). Where an Actor plays a role in a Collaboration realized by the ServiceDescription, a corresponding ExchangePartner is generated, if it is not already present.

8.2.2 UseCase

A UML UseCase represents a “real world effect” – the user’s understanding of a service. UseCases are used by Phase-1 to populate the property ServiceDescription::RealWorldEffect. UseCases may be directly realized by UseCases in the annotation model, or may be contained in realized Packages (see 8.2.3).

8.2.3 Package

A UML Package may be used to collect together Actors or UseCases. Where Actors are contained by Packages, those Packages may be realized by Participants in the annotation model, in which case Phase-1 generates an ExchangePartner for each Actor contained in the Package. Where UseCases are contained by Packages, those Packages may be realized by UseCases in the annotation model, in which case Phase-1 generates a RealWorldEffect for each UseCase contained in the Package.

8.2.4 Association

A UML Association between an Actor and a UseCase stereotyped as «Provider» or «Consumer» (see 8.3) specifies whether the ExchangePartner represented by the Actor is a provider or consumer of the RealWorldEffect represented by the UseCase. Such an Association is used by Phase-1 to populate the annotation properties UseCase::Provider and UseCase::Consumer.

8.2.5 Component

A UML Component represents a Service.

A Component may realize an Actor, signifying that it represents a Service provided by the Participant represented by that Actor. Such a Realization is used by Phase-1 to populate the property Service::ServiceProvider.

A Component may be realized by a Service annotation, providing the foundation for a service interface description. Each service interface description must have a Service realizing a Component. Phase-1 uses this Realization to generate the service hierarchy (see 7.4.4).

8.2.6 Port

A UML Port represents a port on the Service represented by the Component that owns the Port. Phase-1 uses Component::ownedPort to generate the service hierarchy (see 7.4.4).

8.2.7 Interface

A UML Interface represents an interface on the port that provides the interface. Phase-1 uses Port::provided to generate the service hierarchy (see 7.4.4).

8.2.8 Operation

A UML Operation represents a synchronous operation on the interface that owns it. Phase-1 uses Interface::ownedOperation to generate the service hierarchy (see 7.4.4), and for each Operation generates an operation annotation with isAsynchronous = false.

8.2.9 Reception

A UML Operation represents an aSynchronous operation on the interface that owns it. Phase-1 uses Interface::ownedReception to generate the service hierarchy (see 7.4.4), and for each Reception generates an operation annotation with isAsynchronous = true.

8.2.10 Realization

UML Realizations are used to relate annotations to the Logical Service Model elements that they annotate.

8.2.11 Usage

A UML Usage is used to relate a ServiceDescription to each IEPD that it uses. Each Usage is used by Phase-1 to generate an IEPDReference in the ServiceDescription.

8.2.12 Collaboration

A UML Collaboration represents the community involved in the SSP. It must be realized by the ServiceDescription that represents the SSP. Phase-1 generates a ModelReference from this realization.

8.2.13 Interaction

A UML Interaction represents a service interaction. It may be realized by a ServiceInteraction whose Participants must realize Actors that participate in the Interaction.

8.3 Stereotypes

8.3.1 «Provider»

The Provider stereotype may be applied to a UML Association between an Actor and UseCase. It signifies that the ExchangePartner represented by the Actor is a provider of the RealWorldEffect represented by the UseCase.

8.3.2 «Consumer»

The Consumer stereotype may be applied to a UML Association between an Actor and UseCase. It signifies that the ExchangePartner represented by the Actor is a consumer of the RealWorldEffect represented by the UseCase.

9 GRA Annotations

9.1 Package GRAUML::GRAAnnotationModel

9.1.1 Class Agreement

A data structure representing an agreement, formal or informal, associated with a service, application or thing, tangible or otherwise.

Name	Agreement
Abstract	false
Base Classifier	
Constraint	

Properties

AgreementURI

A locator referencing an agreement, formal or informal, associated with a service, application or thing, tangible or otherwise.

Type	String
Default Value	
Multiplicity	0..1
Annotation	

Composition	composite
Constraint	

AutomatedAgreementIndicator

Indicates whether the policy or contract represented by the Agreement has an automated implementation.

Type	Boolean
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

[Intermediate Model Only] Documentation

A free text description of an agreement, formal or informal, associated with a service, application or thing, tangible or otherwise.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	[stringValueOf("Documentation") = docComment()]

9.1.2 Class Description

A data structure used for the types of properties that are intended to document an element, with the possibility of a link to external documentation.

Name	Description
Abstract	false
Base Classifier	
Constraint	

Properties

[Intermediate Model Only] Documentation

The documentation for the element. Derived from the UML element's comment.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	none
Constraint	[stringValueOf("Documentation")=docComment()]

ExternalDocumentation

Relative URL of an artifact which documents the element.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	none
Constraint	

9.1.3 Class GRAServiceAnnotationBase

Abstract base class for all GRA Service annotation metadata classes.

Name	GRAServiceAnnotationBase
Abstract	true
Base Classifier	
Constraint	

Properties

[Intermediate Model Only] Documentation

The documentation for this element. Derived from a realized element or, if not provided, the element's documentation.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	none
Constraint	[stringValue("Documentation") = if realizesElement() then realizedElement().docComment() else docComment() endif]

Flag

Flag provides an arbitrary extension mechanism whereby any business or technology choice may be notated on the containing element. The intent is that a user supplied phase-2 template will then respect these flags and produce the desired result. Unknown flags are ignored.

Type	String
Default Value	
Multiplicity	0..*
Annotation	
Composition	none
Constraint	

[Intermediate Model Only] ModelReference

Reference to a model element annotated by this element. Modeled as a Realization.

Type	ModelReference
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

[Intermediate Model Only] Name

The element name. Derived from the UML InstanceSpecification's name, or if not set will default to the realized element's name.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	none
Constraint	[stringValueOf("Name") = (if name->notEmpty and name <> "" then name else if realizesElement() then realizedElement.name else "" endif endif)]

Requirement

The service interaction requirements.

Type	InteractionRequirements
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite [intermediate model only]
Constraint	

Template

Template is the name of an XSLT template that will be called to process the containing element thus overriding the default mapping.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	none
Constraint	

9.1.4 [Intermediate Model Only] Class IEPDReference

A data structure describing the name and location of an IEPD. An annotation element may be related to an IEPD by means of a Usage dependency, in which case the properties of this structure are derived from the used IEPD.

Name	IEPDReference
Abstract	false
Base Classifier	
Constraint	[TODO: define derivation from uses relationship]

Properties

[Intermediate Model Only] IEPDURL

A URL where the IEPD is posted and available.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

[Intermediate Model Only] Name

A human readable identification of the IEPD

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

9.1.5 Class InteractionRequirements

The Service Interaction Requirements.
This may be specified at multiple levels and it assumed to default to the level above. If not specified in a ServiceDescription it defaults to GRA values (or false if not specified).

Name	InteractionRequirements
Abstract	false
Base Classifier	
Constraint	

Properties

ExtendedRequirement

ExtendedRequirement provides an arbitrary extension mechanism whereby InteractionRequirement may be notated on the containing element. The intent is that the phase-2 transform may then respect these flags and produce the desired result. Unknown extensions are ignored.

Type	String
Default Value	
Multiplicity	*
Annotation	
Composition	none
Constraint	

IdentityAndAttributeAssertionTransmission

True if the service has identity and attribute assertion transmissions, false otherwise.

Type	Boolean
Default Value	
Multiplicity	0..1
Annotation	
Composition	none
Constraint	

InterfaceDescriptionRequirements

True if the service has interface description requirements, false otherwise.

Type	Boolean
Default Value	
Multiplicity	0..1
Annotation	
Composition	none
Constraint	

Logging

True if the service has logging, false otherwise.

Type	Boolean
Default Value	
Multiplicity	0..1
Annotation	
Composition	none
Constraint	

MessageAddressing

True if the service has message addressing, false otherwise.

Type	Boolean
Default Value	
Multiplicity	0..1
Annotation	
Composition	none
Constraint	

MessageConfidentiality

True if the service has message confidentiality, false otherwise.

Type	Boolean
Default Value	
Multiplicity	0..1
Annotation	
Composition	none
Constraint	

MessageIntegrity

True if the service has message integrity, false otherwise.

Type	Boolean
Default Value	
Multiplicity	0..1
Annotation	
Composition	none
Constraint	

MessageNonrepudiation

True if the service has message non-repudiation, false otherwise.

Type	Boolean
Default Value	
Multiplicity	0..1
Annotation	
Composition	none
Constraint	

Reliability

True if the service has reliability, false otherwise.

Type	Boolean

Default Value	
Multiplicity	0..1
Annotation	
Composition	none
Constraint	

ServiceAuthentication

True if the service has service authentications, false otherwise.

Type	Boolean
Default Value	
Multiplicity	0..1
Annotation	
Composition	none
Constraint	

ServiceConsumerAuthentication

True if the service has service consumer authentications, false otherwise.

Type	Boolean
Default Value	
Multiplicity	0..1
Annotation	
Composition	none
Constraint	

ServiceConsumerAuthorization

True if the service has service consumer authorizations, false otherwise.

Type	Boolean
Default Value	
Multiplicity	0..1
Annotation	
Composition	none
Constraint	

ServiceMetadataAvailability

True if the service has metadata availability, false otherwise.

Type	Boolean
Default Value	
Multiplicity	0..1
Annotation	
Composition	none
Constraint	

ServiceResponsiveness

True if the service has service responsiveness, false otherwise.

Type	Boolean
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

TransactionSupport

True if the service has transaction support, false otherwise.

Type	Boolean
Default Value	
Multiplicity	0..1
Annotation	
Composition	none
Constraint	

9.1.6 Class Interface

Represents an abstract service interface.

Name	Interface
Abstract	false
Base Classifier	GRAServiceAnnotationBase
Constraint	

Properties

[Annotations Only] MessageDefault

A set of default message specifications that apply for the interface. At most one does not realize an element. Where there are more than one, each realizes the element that it applies to.

Type	Message
Default Value	
Multiplicity	*
Annotation	
Composition	none
Constraint	

Operation

The operations contained by the interface.

Type	Operation
Default Value	
Multiplicity	0..* [Annotations Only]
Multiplicity	1..* [Intermediate Model Only]
Annotation	
Composition	composite [Intermediate Model Only]
Constraint	

[Annotations Only] ParameterDefault

A set of default parameter specifications that apply for the interface. At most one does not realize an element. Where there are more than one, each realizes the element that it applies to.

Type	Parameter
Default Value	
Multiplicity	*
Annotation	
Composition	none
Constraint	

9.1.7 Class Message

Represents the data of an operation parameter or a signal.

Name	Message
Abstract	false
Base Classifier	GRAServiceAnnotationBase
Constraint	

Properties

[Intermediate Model Only] Prefix

Prefix used to reference Schema Namespace.

Type	String
Default Value	
Multiplicity	1
Annotation	
Composition	composite
Constraint	

9.1.8 [Intermediate Model Only] Class Model

A representation of the model resource associated with a ModelReference and generated as part of the generation of the ModelReference.

Name	Model
Abstract	false
Base Classifier	

Properties

[Intermediate Model Only] Label

The name of the model.

Type	String
Default Value	
Multiplicity	1
Annotation	
Composition	none
Constraint	

[Intermediate Model Only] ModelURI

URI for the model resource.

Type	String
Default Value	
Multiplicity	1
Annotation	
Composition	none
Constraint	

9.1.9 [Intermediate Model Only] Class ModelReference

Model references are generated from Realizations during phase-1.

Name	ModelReference
Abstract	false
Base Classifier	
Constraint	[TODO: define derivation from realization]

Properties

[Intermediate Model Only] DiagramLink

URIs of diagrams associated with the realized element.

Type	String
Default Value	
Multiplicity	0..*
Annotation	
Composition	none
Constraint	

[Intermediate Model Only] Documentation

The documentation of the realized element

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	none
Constraint	

[Intermediate Model Only] ElementID

The id of the realized element.

Type	String
Default Value	
Multiplicity	1
Annotation	
Composition	none
Constraint	

[Intermediate Model Only] Model

The model resource containing the realized element.

Type	Model
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

[Intermediate Model Only] Name

The name of the realized element

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	none
Constraint	

9.1.10 Class Operation

A data structure representing a service action.

Name	Operation

Abstract	false
Base Classifier	GRAServiceAnnotationBase
Constraint	[realizesElement() and (realizedElement().oclIsKindOf(Operation) or realizedElement().oclIsKindOf(Reception))]

Properties

[Intermediate Model Only] ActionName

The name of the action.

Type	String
Default Value	
Multiplicity	1
Annotation	
Composition	composite
Constraint	[stringValueOf("ActionName") = realizedElement().oclAsType(NamedElement).name]

ActionProvenance

A description of provenance information for this action.
Used to populate Action Model section of SDD document.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

[Annotations Only] MessageDefault

A set of default message definitions that apply for the operation. At most one does not realize an element. Where there are more than one, each realizes the element that it applies to.

Type	Message
Default Value	
Multiplicity	*
Annotation	
Composition	none
Constraint	

[Intermediate Model Only] ActionPurpose

A description of the purpose performed by this service action.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite

Constraint	[stringValueOf("ActionPurpose") = realizedElement().docComment()]

[Intermediate Model Only] IsAsynchronous

True if the operation is asynchronous, false otherwise.

Type	Boolean
Default Value	
Multiplicity	1
Annotation	
Composition	none
Constraint	

MessageExchangePattern

Represents the GRA message exchange pattern associated with the operation.

Type	ExchangePattern
Default Value	
Multiplicity	0..1
Annotation	
Composition	none
Constraint	

Parameter

The parameters of an operation

Type	Parameter
Default Value	
Multiplicity	1..* [Intermediate Model Only]
Multiplicity	0..* [Annotations Only]
Annotation	ordered
Composition	composite [Intermediate Model Only]
Constraint	

9.1.11 Class Organization

A data structure describing information about an organization.

Name	Organization
Abstract	false
Base Classifier	GRAServiceAnnotationBase
Constraint	

Properties

OrganizationAcronym

An acronym for the organization.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

OrganizationFullAddressText

A physical address of an organization in full text form.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

OrganizationPointOfContact

A person designated as the point of contact for an organization.

Type	Person
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

OrganizationRoleDescriptionText

An organization's role defined in free form text. Examples could be creator, provider, owner, maintainer, authorities source, etc.

Type	String
Default Value	
Multiplicity	1
Annotation	
Composition	composite
Constraint	

OrganizationRoleDetailedDescriptionText

A very detailed textual explanation of the role and responsibilities of an organization.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

OrganizationWebSiteURL

An internet address of the organization's web site.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

9.1.12 Class Parameter

Represents the information content of parameters defined in an operation.

Name	Parameter
Abstract	false
Base Classifier	GRAServiceAnnotationBase
Constraint	[TODO: define derivation of Parameter]

Properties

Message

The information transfered in an operation

Type	Message
Default Value	
Multiplicity	0..1 [Annotations Only]
Multiplicity	1 [Intermediate Model Only]
Annotation	
Composition	composite [Intermediate Model Only]
Constraint	

[Intermediate Model Only] Use

The "direction" of the message.

Type	ParameterUse
Default Value	
Multiplicity	1

Annotation	
Composition	none
Constraint	[TODO: Define derivation of Use]

9.1.13 Class Participant

A participant in a service interaction.

Name	Participant
Abstract	false
Base Classifier	GRAServiceAnnotationBase
Constraint	[realizesElement() and realizedElement().oclIsKindOf(Actor)]

Properties

ParticipatingOrganization

An organization acting as a participant.

Type	Organization
Default Value	
Multiplicity	0..1
Annotation	
Composition	none
Constraint	

9.1.14 Class Person

A data structure describing a person's name and the means to contact that person.

Name	Person
Abstract	false
Base Classifier	GRAServiceAnnotationBase
Constraint	

Properties

ContactPersonAddress

A physical address of a person in full text form.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

ContactPersonEmailID

An email address of a person.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

ContactPersonPhoneNumberID

A phone number of the person.

Type	String
Default Value	
Multiplicity	1
Annotation	
Composition	composite
Constraint	

9.1.15 Class Port

Represents a service port, realizes a UML Port on the Service Component.

Name	Port
Abstract	false
Base Classifier	GRAServiceAnnotationBase
Constraint	[realizesElement() and realizedElement().oclIsKindOf(Port) -- TODO: the port must be a port of the Service Component]

Properties

AddressURI

The <soap:address> location of a <wsdl:port>. Derived by default in phase-2 from ServiceInterfaceURI+"/"+"Name" but MAY be changed in the model

Type	String
Default Value	
Multiplicity	0..1 [Annotations Only]
Multiplicity	1 [Intermediate Model Only]
Annotation	
Composition	none
Constraint	

Interface

Interfaces provided by the Port.

Type	Interface
Default Value	
Multiplicity	0..* [Annotations Only]
Multiplicity	1..* [Intermediate Model Only]
Annotation	
Composition	composite [Intermediate Model Only]
Constraint	

[Annotations Only] MessageDefault

A set of default message specifications that apply within interfaces owned by the Port. At most one does not realize an element. Where there are more than one, each realizes the element that it applies to.

Type	Message
Default Value	
Multiplicity	*
Annotation	
Composition	none
Constraint	

[Annotations Only] OperationDefault

A set of default operation specifications that apply within interfaces owned by the port. At most one does not realize an element. Where there are more than one, each realizes the element that it applies to.

Type	Operation
Default Value	
Multiplicity	*
Annotation	
Composition	none
Constraint	

[Annotations Only] ParameterDefault

A set of default parameter specifications that apply within interfaces owned by the Port. At most one does not realize an element. Where there are more than one, each realizes the element that it applies to.

Type	Parameter
DefaultValue	
Multiplicity	*
Annotation	
Composition	none
Constraint	

9.1.16 Class SampleData

Represents the information associated with a service test: the input and expected output associated with an operation under test.
[This is a placeholder - requires more development]

Name	SampleData
Abstract	false
Base Classifier	
Constraint	

Properties

ExpectedOutput

The expected output of a service operation test.

Type	String
Default Value	
Multiplicity	1
Annotation	
Composition	composite
Constraint	

Input

The value of input parameters for a service operation test. Input must be in the same order as parameters.

Type	String
Default Value	
Multiplicity	*
Annotation	ordered
Composition	composite
Constraint	

9.1.17 [Intermediate Model Only] Class SchemaReference

Elements describing a schema used by this service interface

Name	SchemaReference
Abstract	false
Base Classifier	
Constraint	[TODO: derivation]

Properties

[Intermediate Model Only] Namespace

Namespace of schema.

Type	String
Default Value	
Multiplicity	1
Annotation	
Composition	composite

Constraint	

[Intermediate Model Only] Prefix

Prefix used to reference Schema Namespace.

Type	String
Default Value	
Multiplicity	1
Annotation	
Composition	composite
Constraint	

[Intermediate Model Only] SchemaLocation

URI of Schema, relative to SPP Catalog

Type	String
Default Value	
Multiplicity	1
Annotation	
Composition	composite
Constraint	

9.1.18 Class SecurityClassification

Any applicable classification of the security level of the information exchanged by the service, such as SBU, Secret, etc. If there is no strict classification this field can contain a brief statement regarding the security of the data.

Name	SecurityClassification
Abstract	false
Base Classifier	
Constraint	

Properties

[Intermediate Model Only] Name

Any applicable classification of the security level of the information exchanged by the service, such as SBU, Secret, etc. If there is no strict classification this field can contain a brief statement regarding the security of the data.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	[stringValue("Name") = name]

9.1.19 Class Service

Describes a service

Name	Service
Abstract	false
Base Classifier	
Constraint	[realizesElement() and realizedElement().oclIsKindOf(Component)]

Properties

[Annotations Only] InterfaceDefault

A set of default interface specifications that apply for ports owned by the service. At most one does not realize an element. Where there are more than one, each realizes the element that it applies to.

Type	Interface
Default Value	
Multiplicity	*
Annotation	
Composition	none
Constraint	

[Annotations Only] MessageDefault

A set of default message specifications that apply for parameters of operations of interfaces provided by ports owned by the service. At most one does not realize an element. Where there are more than one, each realizes the element that it applies to.

Type	Message
Default Value	
Multiplicity	*
Annotation	
Composition	none
Constraint	

[Annotations Only] OperationDefault

A set of default operation specifications that apply for operations of interfaces provided by ports owned by the service. At most one does not realize an element. Where there are more than one, each realizes the element that it applies to.

Type	Operation
Default Value	
Multiplicity	*
Annotation	
Composition	none
Constraint	

[Annotations Only] ParameterDefault

A set of default parameter specifications that apply for operations of interfaces provided by ports owned by the service. At most one does not realize an element. Where there are more than one, each realizes the element that it applies to.

Type	Parameter
Default Value	
Multiplicity	*
Annotation	
Composition	none
Constraint	

Port

The port specification for the ports owned by the service component realizing this Service.

Type	Port
Default Value	
Multiplicity	0..*
Annotation	
Composition	composite [Intermediate Model Only]
Constraint	

SampleData

Sample data associated with service testing.

Type	SampleData
Default Value	
Multiplicity	0..*
Annotation	
Composition	composite [Intermediate Model Only]
Constraint	

ServiceProvider

The provider of the service.

Type	Participant
Default Value	
Multiplicity	0..1
Annotation	
Composition	none
Constraint	

9.1.20 Class ServiceCapability

A free text format description of the capability provided by a service. Per GRA, A capability represents the provider's view of a service.

Name	ServiceCapability
Abstract	false
Base Classifier	
Constraint	

Properties

Documentation

A free text format description of the capability provided by a service.
[Derived from] Element Documentation

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	[stringValueOf("Documentation") = docComment()]

9.1.21 Class ServiceDescription

A data structure representing the details describing a service.
[Realizes] Collaboration
[Uses] IEPD

Name	ServiceDescription
Abstract	false
Base Classifier	ServiceIdentification
Constraint	[realizesElement() and realizedElement().oclIsKindOf(Collaboration) and usesElement() and usedElement().isIEPD()]

Properties

ActivationDate

A date when a service was or will be first available in production.Not to be confused with the date this service was submitted to a registry. The format is YYYY-MM.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

AdditionalInformation

This property contains any additional information pertinent to the service which should be included in this description but does not belong elsewhere. This could be information about future capabilities the service could provide, information regarding specific conditions which govern the use of the service, information regarding specific domain capabilities the service fulfills, etc. If required, subsections can be created to further structure the information provided in this section.

Additional artifacts related to this section's content can be provided in the artifacts folder of the service package.

[Service Abbreviation] SSP [Service Version]\artifacts

If such artifacts are provided, they should be referenced here. A description of the artifact and a link to it should be provided as part of the reference.

This property is used to populate the Additional Information section of the SDD document.

Type	Description
Default Value	
Multiplicity	0..*
Annotation	ordered
Composition	composite [Intermediate Model Only]
Constraint	

AlertAndNotificationURI

A URL to sign up for alerts and notifications for a specific service.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

Classification

A collection of classifications defining the relationship between a service and an applicable enterprise architecture and business reference model.

Type	String
Default Value	
Multiplicity	*
Annotation	ordered
Composition	composite
Constraint	

CreationDate

A date designation when a service was first created. Not to be confused with the date a service is submitted to a registry. The format is YYYY-MM. Set by default to today by Phase-2.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

DataProvenance

Type	Description
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite [Intermediate Model Only]
Constraint	

DomainDescription

A primary domain or line of business (LoB) that a service covers.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

Endorsements

A collection of names and acronyms of professional or governmental organizations or people that endorse this service as an official business exchange.

Type	String
Default Value	
Multiplicity	0..*
Annotation	ordered
Composition	composite
Constraint	

ExchangePartner

The participants in a service.In the case of a package, each contained actor shall produce a use Participant element in Phase-1.

Type	Participant
Default Value	
Multiplicity	0..*
Annotation	
Composition	composite [Intermediate Model Only]
Constraint	[TODO: derived from Role in Collaboration]

ExecutionContext

Service descriptions should include all information pertinent to the production or consumption of the service, including expected infrastructure functions and other dependencies. No information directly pertaining to implementation platform or technology should be included in the service description. Conversely, platform capabilities which are technology-independent should be included. For example, stating in the service description that services that are encrypted are being provided by the infrastructure is preferred compared to stating that the Public Key Infrastructure (PKI) infrastructure is expected. It is expected that the services defined using this document will minimize the dependence on specific technical infrastructure to provide the greatest flexibility and interoperability for service providers and service consumers.

It is also important to note that the information in this property will be applicable to more than one service, and these required capabilities will be provisioned as part of the infrastructure layer of the architecture. For instance, if information is to be exchanged securely within the execution context, enabling this functionality at the infrastructure level and not per a specific SSP or SIP is the strongly preferred direction for enabling the GRA. In other words, these commonly used technical functions would be most effectively achieved by an infrastructure solution which supports the GRA.

Type	Description
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite [Intermediate Model Only]
Constraint	

ExpirationDate

A year and month (YYYY-MM) this service is expected to be no longer available (if applicable).

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

[Intermediate Model Only] IEPDReference

Reference information identifying an Information Exchange Package Document which the service uses in its data model
[Derived from] <Uses> relation to IEPD

Type	IEPDReference
Default Value	
Multiplicity	1..*
Annotation	
Composition	composite
Constraint	

LastRevisionDate

A date with the year and month specifying when this service information was last revised. The format is YYYY-MM. Set by default by Phase-2.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

LifecycleStatus

A word indicating the current stage of the service within the development lifecycle. Valid values are: In Design, In Development, Release Candidate, Operational/Production, Deprecated.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

MajorVersion

A value identifying the primary version number. Defaults to 1.

Type	Integer
Default Value	
Multiplicity	1
Annotation	
Composition	composite
Constraint	

MinorVersion

A value designating a minor version number. Defaults to 0.

Type	Integer
Default Value	
Multiplicity	1

Annotation	
Composition	composite
Constraint	

NextRevisionDate

A year and month (YYYY-MM) a service is expected to be revised.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

OtherRequirement

[This section lists any other requirements which have to be met and on which the service depends to deliver its capabilities.]

[Additional artifacts related to specific subsections of this section's content can be provided under the artifacts folder of the service package. It is suggested there be a large number of artifacts related to this section. These artifacts are included as documents or folders under the various artifacts subfolder of the artifacts folder.

Type	Description
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite [Intermediate Model Only]
Constraint	

Privacy

While service descriptions are technology-agnostic and do not detail the physical model of a service, certain privacy requirements are applicable to the service and need to be carried through all its implementations. This property outlines those requirements.

Value is used to populate the Privacy section of the SDD Document.

Type	Description
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite [Intermediate Model Only]
Constraint	

RealWorldEffect

A collection of the Real World Effects caused by the Service.

Type	UseCase
Default Value	
Multiplicity	*
Annotation	
Composition	composite [Intermediate Model Only]
Constraint	

RelatedOrganization

A collection of organizations that are somehow related to the service.

Type	Organization
Default Value	
Multiplicity	0..*
Annotation	
Composition	[Intermediate Model Only]
Constraint	

RevisionVersion

A value designating a minor version revision number.

Type	Integer
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

Security

While service descriptions are technology-agnostic and do not detail the physical model of a service, certain security requirements are applicable to the service and need to be carried through all its implementations. This property outlines those requirements.

The value is used to populate the Security section of the SDD document.

Type	Description
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite [Intermediate Model Only]
Constraint	

ServiceAssumption

This property lists all assumptions on which the service depends to deliver its real-world effects.

Type	Description
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite [Intermediate Model Only]
Constraint	

ServiceCapability

An enumeration of the capabilities provided by a service.

Type	ServiceCapability
Default Value	
Multiplicity	0..*
Annotation	ordered
Composition	composite [Intermediate Model Only]
Constraint	

ServiceDependency

Services upon which a service directly depends to deliver its real world effects.

Type	ServiceIdentification
Default Value	
Multiplicity	0..*
Annotation	
Composition	composite [Intermediate Model Only]
Constraint	

ServiceDescriptionKeywordText

Search terms that would not otherwise be in other metadata attributes (e.g., Child Support Warrant, Domestic Relations Warrant, Domestics).

Type	String
Default Value	
Multiplicity	0..*
Annotation	ordered
Composition	composite
Constraint	

ServiceDescriptionSummaryText

A brief summary of this service for short display purposes.

Type	String
Default Value	
Multiplicity	0..1
Annotation	

Composition	composite
Constraint	

[Intermediate Model Only] ServiceDescriptionURI

URI of the Service Description Document, relative to the SSP Catalog.

Type	String
Default Value	
Multiplicity	1
Annotation	
Composition	composite
Constraint	

ServiceInteraction

A Documented Component

Type	ServiceInteraction
Default Value	
Multiplicity	*
Annotation	
Composition	composite [Intermediate Model Only]
Constraint	

ServiceInterface

A set of service interface specifications.

Type	ServiceInterfaceSpecification
Default Value	
Multiplicity	0..*
Annotation	
Composition	composite [Intermediate Model Only]
Constraint	

ServiceLevelAgreement

A set of service level agreements.

Type	ServiceLevelAgreement
Default Value	
Multiplicity	0..*
Annotation	ordered
Composition	composite [Intermediate Model Only]
Constraint	

ServicePurpose

A purpose which the service intends or resolves to perform or accomplish.

Type	Description
Default Value	
Multiplicity	1
Annotation	
Composition	composite [Intermediate Model Only]
Constraint	

ServiceScopeDescription

A description of the scope of a service.

Type	Description
Default Value	
Multiplicity	1
Annotation	
Composition	composite [Intermediate Model Only]
Constraint	

ServiceSecurityClassification

The service's security classification.

Type	SecurityClassification
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite [Intermediate Model Only]
Constraint	

Sponsor

A collection of professional or governmental organization(s) or person that sponsored, contributed, or participated in the development of a service.

Type	String
Default Value	
Multiplicity	*
Annotation	ordered
Composition	composite
Constraint	

TransformationURI

URI of an XSLT that transforms the skeleton SSP into a final one as the "phase-2" transform.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	none
Constraint	

9.1.22 Class ServiceIdentification

A data structure representing the means of uniquely identifying a service.

Name	ServiceIdentification
Abstract	false
Base Classifier	GRAServiceAnnotationBase
Constraint	

Properties

ServiceID

An identification of the service in a service registry and/or repository.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

ServiceNameAbbreviationText

A human readable abbreviation of the Service Name.

Type	String
Default Value	
Multiplicity	1
Annotation	
Composition	composite
Constraint	

ServiceURI

A fully qualified locator of the service interface potentially including version and environment.

Type	String
Default Value	
Multiplicity	1
Annotation	
Composition	composite

Constraint	

9.1.23 Class ServiceInteraction

Represents exchanges between parties. Realizes an Interaction.

Name	ServiceInteraction
Abstract	false
Base Classifier	GRAServiceAnnotationBase
Constraint	[realizesElement() and realizedElement().oclIsKindOf(Interaction) -- TODO: the partipants must realize Actors involved in the interaction]

Properties

Participant

The participants in the interaction, each of which realizes an Actor that is involved in the interaction.

Type	Participant
Default Value	
Multiplicity	0..*
Annotation	
Composition	composite
Constraint	

9.1.24 Class ServiceInteractionProfile

A data structure containing information about a Service Interaction Profile.

Name	ServiceInteractionProfile
Abstract	false
Base Classifier	
Constraint	

Properties

SIPName

The name of the Service Interaction Profile.
[Derived from] SIP Name

Type	String
Default Value	
Multiplicity	1
Annotation	
Composition	composite
Constraint	

SIPVersion

Version of the Service Interaction Profile.

Type	String
Default Value	
Multiplicity	1
Annotation	
Composition	composite
Constraint	

9.1.25 Class ServiceInterfaceSpecification

A data type representing details relating to a service interface.

Name	ServiceInterfaceSpecification
Abstract	false
Base Classifier	GRAServiceAnnotationBase
Constraint	

Properties

[Annotations Only] InterfaceDefault

A set of default interface specifications that apply for provided interfaces of ports of owned services. At most one does not realize an element. Where there are more than one, each realizes the element that it applies to.

Type	Interface
Default Value	
Multiplicity	*
Annotation	
Composition	none
Constraint	

[Annotations Only] MessageDefault

A set of default message specifications that apply for parameters of operations in provided interfaces of ports of owned services. At most one does not realize an element. Where there are more than one, each realizes the element that it applies to.

Type	Message
Default Value	
Multiplicity	*
Annotation	
Composition	none
Constraint	

MessageDefinitionMechanism

This property includes information about the message definition mechanism utilized by the service actions.

Per the GRA, message definition mechanisms are closely related to the interface description requirements. Unlike interface description requirements, message definition mechanisms establish a standard way of defining the structure and contents of a message.

Additional artifacts related to this section's content can be provided in the artifacts folder of the service package.

[Service Abbreviation] SSP [Service Version]\artifacts

If such artifacts are provided, they should be referenced here. A description of the artifact and a link to it should be provided as part of the reference.

Type	Description
Default Value	
Multiplicity	0..*
Annotation	ordered
Composition	composite
Constraint	

[Annotations Only] OperationDefault

A set of default operation specifications that apply for operations in provided interfaces of ports of owned services. At most one does not realize an element. Where there are more than one, each realizes the element that it applies to.

Type	Operation
Default Value	
Multiplicity	*
Annotation	
Composition	none
Constraint	

[Intermediate Model Only] Prefix

Prefix to be used for target namespace of (WSDL) Service Specification. Derived from (the same as) ServiceInterfaceNameAbbeviationText

Type	String
Default Value	
Multiplicity	1
Annotation	
Composition	composite
Constraint	

[Annotations Only] ParameterDefault

A set of default parameter specifications that apply for operations in provided interfaces of ports of owned services. At most one does not realize an element. Where there are more than one, each realizes the element that it applies to.

Type	Parameter
Default Value	
Multiplicity	*

Annotation	
Composition	none
Constraint	

[Intermediate Model Only] SchemaReference

Schema used
TODO: ** Question - do we need to allow entry of schema not in model?

Type	SchemaReference
Default Value	
Multiplicity	*
Annotation	
Composition	composite
Constraint	

[Annotations Only] PortDefault

A set of default port specifications that apply within owned services. At most one does not realize an element. Where there are more than one, each realizes the element that it applies to.

Type	Port
Default Value	
Multiplicity	*
Annotation	
Composition	none
Constraint	

SecurityDescriptionText

A text description that identifies the security which was implemented to protect this service interface (GFIPM, Trusted Broker, etc)

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

SecurityImplementedIndicator

True when security has been implemented to access this service; otherwise false.

Type	Boolean
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite

Constraint	

Service

The services of the service interface. note that in GRA it is best practice to have exactly one service per service interface specification.

Type	Service
Default Value	
Multiplicity	1..*
Annotation	
Composition	composite [Intermediate Model Only]
Constraint	

ServiceInteractionProfile

Information about the implemented Service Interaction Profile.

Type	ServiceInteractionProfile
Default Value	
Multiplicity	1
Annotation	
Composition	composite
Constraint	

[Intermediate Model Only] ServiceInterfaceDescriptionURI

URI of the Service Interface Description document, potentially including version and environment. The URI is relative to the SSP Catalog.

TODO: Really? ->[derived from] Element name.

Type	String
Default Value	
Multiplicity	1
Annotation	
Composition	composite
Constraint	

ServiceInterfaceNameAbbreviationText

A human readable abbreviation of the Service Name that is also appropriate as a WSDL prefix.

Type	String
Default Value	
Multiplicity	1
Annotation	
Composition	none

Constraint	

ServiceTesting

TODO: Missing documentation

Type	Description
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

TargetNamespace

Target namespace of the (WSDL) Service Interface.

Type	String
Default Value	
Multiplicity	1
Annotation	
Composition	composite
Constraint	

[Intermediate Model Only] URIAddress

URI of the (WSDL) service interface potentially including version and environment. The URI is relative to the SSP Catalog.

TODO: really? -> [derived from] Element name.

Type	String
Default Value	
Multiplicity	1
Annotation	
Composition	composite
Constraint	

9.1.26 Class ServiceLevelAgreement

A collection of policies, agreements, licensing and any other governance or performance documentation specifying constraints and any other details regarding the realization of a service.

Name	ServiceLevelAgreement
Abstract	false
Base Classifier	
Constraint	

Properties

Agreement

Applicable Agreements or MOUs governing the use, administration, or implementation of a service.

Type	Agreement
Default Value	
Multiplicity	0..*
Annotation	
Composition	composite [Intermediate Model Only]
Constraint	

ApplicableAgreementsIndicator

True when there are any applicable agreements or Memoranda Of Understanding (MOUs) relating to the use, administration, or implementation of a service; otherwise false.

Type	Boolean
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	[applicableAgreementsIndicator = (Agreement->notEmpty())]

ApplicableContract

A set of formal contract associated with a service, application, process, transaction, or thing, tangible or otherwise.

Type	Agreement
Default Value	
Multiplicity	0..*
Annotation	
Composition	composite [Intermediate Model Only]
Constraint	

ApplicableContractsIndicator

True when there are any applicable contracts relating to the use, administration, or implementation of a service; otherwise false.

Type	Boolean
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	[applicableContractsIndicator = (applicableContract->notEmpty())]

ApplicablePoliciesIndicator

True when there are any applicable policies governing the use, administration, or implementation of a service; otherwise false.

Type	Boolean
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

ApplicablePolicy

A collection of all policies that in some way constrain, govern, or control the usage of a service, application, process. etc.

Type	Description
Default Value	
Multiplicity	0..*
Annotation	
Composition	composite [Intermediate Model Only]
Constraint	

ApplicableUmbrellaAgreementsIndicator

True when there are any applicable umbrella agreements relating to the use, administration, or implementation of a service; otherwise false.

Type	Boolean
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	[applicableUmbrellaAgreementsIndicator = (UmbrellaAgreement->notEmpty())]

ApprovalRequiredIndicator

True when a permission must first be obtained prior to using a service or performing some action in a business process; otherwise false.

Type	Boolean
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

CreationCostAmount

A cost create a thing, such as an application or service This includes the full cost to design, manage, develop, test, implement and maintain. Currency text may precede or follow amount.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

LicensingAgreement

Agreement licensing a service or application. Descriptive values could be In House, No License, Open Source, Purchase License, etc.

Type	Agreement
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite [Intermediate Model Only]
Constraint	

LicensingRequiredIndicator

True when a license is required to use a service or application; otherwise false.

Type	Boolean
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

ServiceAvailability

A description or measurement of the expected availability that a service is usable.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

ServiceAverageThroughput

A description of how often a service is expected to be, or actually used, averaged over a period of time.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

ServiceMaximumThroughput

A description of the limit of how often a service is able to be accessed or used at, over a period of time during peak capacity.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

ServiceResponseTime

A description of the average response time for a service. The response time is calculated as the time input is provided to the service until the service completes its process or provides output for the consumer.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

UmbrellaAgreement

A set of umbrella agreements that in some way constrain, govern, or control the usage of a service, application, process. etc.

Type	Agreement
Default Value	
Multiplicity	0..*
Annotation	
Composition	composite [Intermediate Model Only]
Constraint	

UsageCostAmount

A total cost to use something, such as a service, etc. Currency text may precede or follow amount.

Type	String
Default Value	

Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

UsageUnitCostAmount

A cost associated with a service(e.g. transaction, unlimited transactions, minutes of use). Currency text may precede or follow amount.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	composite
Constraint	

9.1.27 Class UseCase

[Derived from] Use case or package with use cases. In the case of a package, each contained use case shall produce a use case element in Phase-1.

Name	UseCase
Abstract	false
Base Classifier	GRAServiceAnnotationBase
Constraint	[TODO: define derivation.]

Properties

Consumer

Participants that consume services.

Type	Participant
Default Value	
Multiplicity	*
Annotation	
Composition	composite
Constraint	

Provider

Participants that provide services.

Type	Participant
Default Value	
Multiplicity	*
Annotation	
Composition	composite

Constraint	

9.1.28 Enumeration ExchangePattern

Represents a GRA Exchange Pattern.
This type should define the message exchange patterns leveraged by the service actions.
The GRA recognizes the following message exchange patterns:
The FIRE-AND-FORGET pattern calls for the sender of a message (which could be the service consumer or service) to send the message and not expect a reply message back from the recipient. This pattern is useful for one-way transmission of information, such as notification that an event has occurred.
The REQUEST-REPLY pattern calls for the sender of a message to send the message and expect a reply from the recipient.
These two patterns are considered “primitive” patterns, in that they are the fundamental building blocks of more complex information exchange scenarios. For instance, the complex PUBLISH-SUBSCRIBE pattern involves an initial request-reply exchange in which the subscriber subscribes to a service, followed by the service using the fire-and-forget pattern to notify subscribers of an event.
Within the service interface description, the behavioral model should describe message exchanges in terms of these two primitive exchange patterns. Each action can specify a different message exchange pattern.

Literals

unknown
This is a temporary workaround for some EMF issues.

enquiry

subscription

notification

update

message

9.1.29 Enumeration ParameterUse

Defines the direction of a message associated with the parameters and exceptions of an operation.

Literals

unknown
This is a temporary workaround for some EMF issues.

in
Specifies that the corresponding operation parameter has an in direction.

out
Specifies that the corresponding operation parameter has an out direction.

inout
Specifies that the corresponding operation parameter has an in and out direction.

exception
Specifies that the usage for the message is as a fault, or exception, on the operation.

9.2 Package GRAUML::GRA_WSDL

9.2.1 Class WSDLInterface

Represents a <wsdl:portType>.

Name	WSDLInterface
Abstract	false
Base Classifier	Interface
Constraint	

Properties

BindingCode

Specifies technology implementation for a <wsdl:binding>.

Type	BindingType
Default Value	
Multiplicity	0..1
Annotation	
Composition	none
Constraint	

9.2.2 Class WSDLMessage

Represents a <wsdl:message>, its contained <part>, and its usage from a <wsdl:input>, <wsdl:output>, or a <wsdl:fault> within a <wsdl:portType> <wsdl:operation> .

Name	WSDLMessage
Abstract	false
Base Classifier	Message
Constraint	

Properties

Encoding

Represents the encoding style of a wsdl binding operation parameter.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	none
Constraint	

MessageLocationCode

When used in conjunction with a soap binding, indicates the location of the part within the message.

Type	MessageLocation
Default Value	
Multiplicity	0..1
Annotation	
Composition	none
Constraint	

SoapAction

Represents the content of a soapAction within a binding operation having a <soap:binding>.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	none
Constraint	

9.2.3 Class WSDLOperation

Represents a <wsdl:Operation> within a <wsdl:binding> or <wsdl:portType>.

Name	WSDLOperation
Abstract	false
Base Classifier	Operation
Constraint	

Properties

OperationKindCode

Represents the style of a <soap:operation> within a <wsdl:binding><wsdl:operation> which has a <soap:binding>.

Type	OperationKind
Default Value	
Multiplicity	0..1
Annotation	
Composition	none
Constraint	

9.2.4 Class WSDLParameter

Name	WSDLParameter

Abstract	false
Base Classifier	Parameter
Constraint	

Properties

MimeType

Represents the <mime:content> type within a <wsdl:binding> <wsdl:operation> parameter.

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	none
Constraint	

9.2.5 Class WSDLPort

Represents a <wsdl:port> and <wsdl:binding>.

Name	WSDLPort
Abstract	false
Base Classifier	Port
Constraint	

Properties

Certificate

Data for a certificate in support of security

Type	String
Default Value	
Multiplicity	0..1
Annotation	
Composition	none
Constraint	

9.2.6 Class WSDLService

Represents a <wsdl:service>.

Name	WSDLService
Abstract	false
Base Classifier	Service
Constraint	

9.2.7 Class WSDLServiceInterface

Represents a <wsdl:definitions>.

Name	WSDLServiceInterface
Abstract	false
Base Classifier	ServiceInterfaceSpecification
Constraint	

9.2.8 Enumeration BindingType

Literals

soap
Represents a soap binding.

soap12
Represents a wsdl soap 1.2 binding.

http_get
Represents a wsdl http get binding.

http_put
Represents a wsdl http post binding.

9.2.9 Enumeration MessageLocation

Literals

body
For a soap binding, indicates that the part is in the body of the message.

header
For a soap binding, indicates that the part is in the header of the message.

url

9.2.10 Enumeration OperationKind

Literals

doc
Represents a soap:operation style of "document".

rpc
Represents a soap:operation style of "rpc".

9.3 Phase-1 Derived Properties

Table 1 below summarizes all of the properties in the intermediate model (serialized as annotations.xmi) whose values are calculated by Phase-1.

Table 1 Phase-1 Derived Properties

Class	Name	Type	Phase 1 Derivation
Agreement	Documentation	String	Instance documentation

Description	Documentation	String	Instance documentation
GRAServiceAnnotationBase	Documentation	String	Realized element's documentation or instance documentation
GRAServiceAnnotationBase	ModelReference	ModelReference	Realized element
GRAServiceAnnotationBase	Name	String	Instance name or realized element's name
IEPDReference	IEPDURL	String	Usage target
IEPDReference	Name	String	Usage target's name
Interface	Operation	Operation	Derived from Operations / Receptions in realized Interface
Message	Prefix	String	Derived, but I'm not sure where from
Model	Label	String	Created from realization. The name of the realization target's model.
Model	ModelURI	String	Created from realization. The URI of the realization target's model resource.
ModelReference	DiagramLink	String	Created from realization. URIs of diagrams associated with the realized element.
ModelReference	Documentation	String	Created from realization.Documentation of the realized element.
ModelReference	ElementID	String	Created from realization. Id of the realized element.
ModelReference	Model	Model	Created from realization. Model containing the realized element.
ModelReference	Name	String	Created from realization. Name of the realized element.
Operation	ActionName	String	Realized element's name
Operation	ActionPurpose	String	Realized element's documentation
Operation	IsAsynchronous	Boolean	True if realized element is Reception; false if realized element is Operation
Operation	Message Exchange Pattern	ExchangePattern	Based on realized element's pattern (how?)
Operation	Parameter	Parameter	Derived from parameters of realized Operation / Reception.

Parameter	Message	Message	Derived from parameter's type
Parameter	Use	ParameterUse	Derived from parameter direction
Port	Interface	Interface	Derived from provided interfaces of realized Port
SchemaReference	Namespace	String	Derived how?
SchemaReference	Prefix	String	Derived how?
SchemaReference	SchemaLocation	String	Derived how?
SecurityClassification	Name	String	Instance name
Service	Port	Port	Derived from Ports on realized Component
Service	ServiceProvider	Participant	"Provider realized by service component". TODO: I don't understand how this is derived. In current examples it isn't.
ServiceCapability	Documentation	String	Instance documentation
ServiceDescription	Exchange Partner	Participant	Actors defined in service interactions in realized Collaboration, or where an actor or package containing actors is explicitly realized by an ExchangePartner annotation instance.
ServiceDescription	IEPDReference	IEPDReference	Derived from usage to IEPD.
ServiceDescription	Last RevisionDate	String	Derived from today.
ServiceDescription	RealWorld Effect	UseCase	If the modeled real world effect realizes a package, then Phase 1 expands it to give a real world effect for every use case in that package.
ServiceDescription	RevisionVersion	Integer	If nothing is modeled, 1 is inserted.
ServiceDescription	Service DescriptionURI	String	artifacts/[Service Abbreviation]_SDD_v_[major].[minor].[revision]
ServiceDescription	Service Interaction	ServiceInteraction	Derived from Interactions owned by the realized Collaboration
ServiceDescription	Transformation URI	String	http://it.ojp.gov/gsp/BaseSSPTransform
ServiceInteraction	Participant	Participant	Each participant realizes an Actor involved in the Interaction realized by the ServiceInteraction

ServiceInterfaceSpecification	Prefix	String	Derived from (the same as) ServiceInterfaceNameAbbeviationText
ServiceInterfaceSpecification	Schema Reference	SchemaReference	Derived - how?
ServiceInterfaceSpecification	ServiceInterface DescriptionURI	String	artifacts/[Service Abbreviation]_SIDD_[Service Interface Abbreviation] v_[major].[minor].[revision]
ServiceInterfaceSpecification	URIAddress	String	Derived - how?

9.4 Phase-2 Default Property Values

Table 2 below summarizes values that should be assumed by Phase-2 if no values are set in the intermediate file by Phase-1.

Table 2 Phase-2 Default Property Values

Class	Name	Type	Default value
Agreement	Automated Agreement Indicator	Boolean	FALSE
GRAServiceAnnotationBase	Requirement	Interaction Requirements	Default requirements
InteractionRequirements	IdentityAnd Attribute Assertion Transmission	Boolean	FALSE
InteractionRequirements	Interface Description Requirements	Boolean	TRUE
InteractionRequirements	Logging	Boolean	FALSE
InteractionRequirements	Message Addressing	Boolean	TRUE
InteractionRequirements	Message Confidentiality	Boolean	FALSE
InteractionRequirements	Message Integrity	Boolean	FALSE
InteractionRequirements	Message Nonrepudiation	Boolean	FALSE
InteractionRequirements	Reliability	Boolean	FALSE

InteractionRequirements	Service Authentication	Boolean	FALSE
InteractionRequirements	Service Consumer Authentication	Boolean	FALSE
InteractionRequirements	Service Consumer Authorization	Boolean	FALSE
InteractionRequirements	Service Metadata Availability	Boolean	FALSE
InteractionRequirements	Service Responsiveness	Boolean	FALSE
InteractionRequirements	Transaction Support	Boolean	FALSE
Port	AddressURI	String	ServiceInterfaceURI+"/"+"Name"
ServiceDescription	Additional Information	Description	"N/A"
ServiceDescription	CreationDate	String	[today]
ServiceDescription	DataProvenance	Description	“Provenance is defined as the agency, office, or person of origin of records, i.e., the entity that created, received, or accumulated and used the records in the conduct of their business activities. All applicable provenance information or restrictions are provided below. Also, additional artifacts related to this sections content can be provided in the service model folders of the service package. [Service Abbreviation] SSP [Service Version]\artifacts\service model\information model”
ServiceDescription	Domain Description	String	"N/A"
ServiceDescription	Execution Context	Description	“Service descriptions should include all information pertinent to the production or consumption of the service, including expected infrastructure functions and other dependencies. No information directly pertaining to implementation platform or technology should be included in the service description.”
ServiceDescription	LifecycleStatus	String	Boilerplate TBD
ServiceDescription	Other Requirement	Description	"N/A"

ServiceDescription	Privacy	Description	“The MOUs between participating entities will further define specific privacy requirements. Global has developed a document, “Implementing Privacy Policy in Justice Information Sharing: A Technical Framework” This document is intended to provide guidelines for supporting the electronic expression of privacy policy and how to convert privacy policy so that it is understandable to computers and software.”
ServiceDescription	Security	Description	“The service must adhere to the rules of the CJIS Security Policies regarding the Advanced Encryption Standard (AES) level of encryption. The use of Virtual Private Networks (VPNs) or Secure Sockets Layer (SSL) for transport-level security in addition to these requirements is optional.”
ServiceDescription	Service Assumption	Description	“N/A”
ServiceDescription	Service Dependency	Service Identification	"N/A"
ServiceDescription	ServiceLevel Agreement	ServiceLevel Agreement	None
ServiceDescription	ServiceSecurity Classification	Security Classification	“The highest level of security classification for the information exchanged by this service is Sensitive but Unclassified (SBU). As a result the service can be assigned a security classification of SBU.”
ServiceInterfaceSpecification	Message Definition Mechanism	Description	“The service will comply with the message definition mechanisms identified in the GRA Reliable Secure Web Services Service Interaction Profile, Version 1.2 (GRA RS WS-SIP 1.2).”
ServiceInterfaceSpecification	Security Description Text	String	“The service must adhere to the rules of the CJIS Security Policies regarding the Advanced Encryption Standard (AES) level of encryption. The use of Virtual Private Networks (VPNs) or Secure Sockets Layer (SSL) for transport-level security in addition to these requirements is optional.”
ServiceInterfaceSpecification	Security Implemented Indicator	Boolean	TRUE if the technology implements security
ServiceInterfaceSpecification	ServiceTesting	Description	“The service validation and testing will leverage the Springboard specification to validate the interoperable aspects of the service interface specification in order to assert that a participating system conforms to the underlying specification. The conformance specification and the associated test cases define a series of tests designed to exercise each interoperability aspect of the specification at least once. The validation testing is not intended to address functional aspects of the

			systems/services nor is the test suite designed to verify the robustness or performance of the interface software.”
ServiceLevelAgreement	Applicable Agreements Indicator	Boolean	TRUE if there are agreements
ServiceLevelAgreement	Applicable Contracts Indicator	Boolean	TRUE if contracts specified
ServiceLevelAgreement	Applicable Policies Indicator	Boolean	TRUE if policies specified
ServiceLevelAgreement	Applicable Umbrella Agreements Indicator	Boolean	TRUE if umbrellas specified
ServiceLevelAgreement	Approval Required Indicator	Boolean	FALSE
ServiceLevelAgreement	Licensing Required Indicator	Boolean	TRUE if licensing specified
WSDLInterface	BindingCode	BindingType	soap12
WSDLMessage	Encoding	String	Document / Literal
WSDLMessage	Message LocationCode	Message Location	Body
WSDLMessage	SoapAction	String	* Derived
WSDLOperation	Operation KindCode	OperationKind	doc
WSDLParameter	MimeType	String	text/xml

10 GRA-UML Transformations

This clause will be completed in the revised submission.

10.1 SSP workflow and structure

Explain how all the transformations fit together in an overall workflow and resulting folder structure.

10.2 Catalog.xml

Specify catalog schema and metamodel. Note: metamodel includes tool-specific extensions annotating the metamodel to enable XML document generation and representation of the XML type library.

Specify QVT. Describe the action of the QVT in a readable way (Tom?)

10.3 Metadata.xml

Specify metadata schema and metamodel. Note: metamodel includes tool-specific extensions annotating the metamodel to enable XML document generation and representation of the XML type library.

Specify QVT. Describe the action of the QVT in a readable way (Tom?)

10.4 Annotations.xmi

Explain the rules for populating the annotation model from the PIM + the modeled annotations. (Tom)

Explain how XMI is generated from the annotation model as an EMOF instance.

Specify QVT. Describe the action of the QVT in a readable way (Tom?)

10.5 SDD

Using XSLT to generate SDD XHTML from the annotations.xmi. Describe the action of the XSLT in a readable way.

10.6 SIDD

Using XSLT to generate SIDD XHTML from the annotations.xmi. Describe the action of the XSLT in a readable way.

10.7 WSDL

Using XSLT to generate WSDL from the annotations.xmi. Describe the action of the XSLT in a readable way.

Annex A: GRA-UML Example

(informative)

This annex contains a simple example to illustrate the concepts. The example describes a Pet Adoption Center: a fictitious agency whose business is the adoption of (presumably unwanted) pets. This is an extension of the example elaborated in the NIEM-UML profile [NIEM-UML], emphasizing that the information transferred by GRA services is specified using NIEM. The aim is to generate a Service Specification Package (SSP) that defines the services provided and consumed in the pet adoption community.

We start with the Platform Independent Model (PIM). This is a UML model, built using the Logical Service Profile under GRA-UML conventions, which describes services provided by the Pet Adoption Centre.

The PIM contains Actors, which depict the entities that interact when services are offered or consumed. Actors are often people or classes of people, organizations or classes of organizations, or even machines or classes of machines. In this example there are two Actors: the PetAdoptionCenter, and Anyone.

In the PIM, a UseCase represents the user's view of a service, which is known in GRA as the "real world effect". In this example there is one service: "Provide information on a particular pet adoption based on the ID of the adopting individual". The Actors and the UseCase are shown together on a single diagram in Figure 12, which also shows by associations stereotyped «Provider» and «Consumer» that the PetAdoptionCenter provides this service, and Anyone may consume it. These Actors and UseCase are contained in a Package called Use Cases: this fact will be important later.

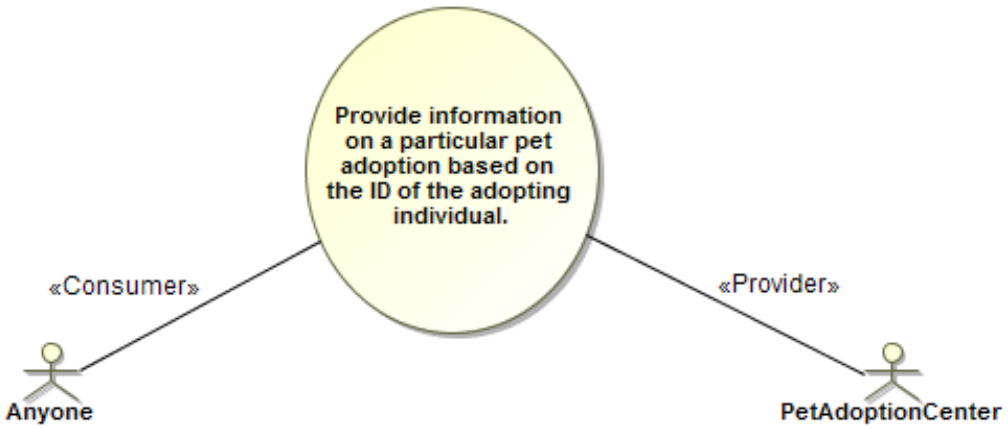

Figure 12 Pet Adoption Actors and UseCase

Interfaces in the PIM define the operations and signal receptions that service providers implement to offer a service. For the PetAdoptionCenter there is a single Interface, shown in Figure 13.

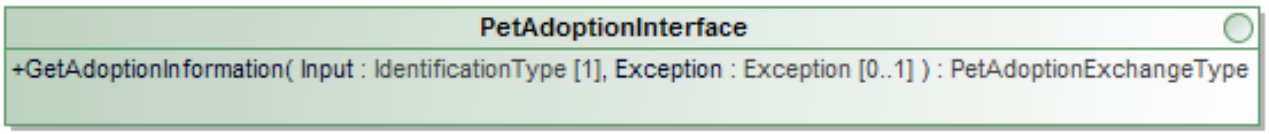

Figure 13 PetAdoptionInterface

The arguments of operations directly reference [NIEM-UML] message types as their content. These types come from the NIEM IEPD (Information Exchange Package Documentation) depicted in Figure 14, which is constructed according to the rules and conventions of NIEM-UML. Full explanation of this IEPD can be found in the [NIEM-UML] specification.

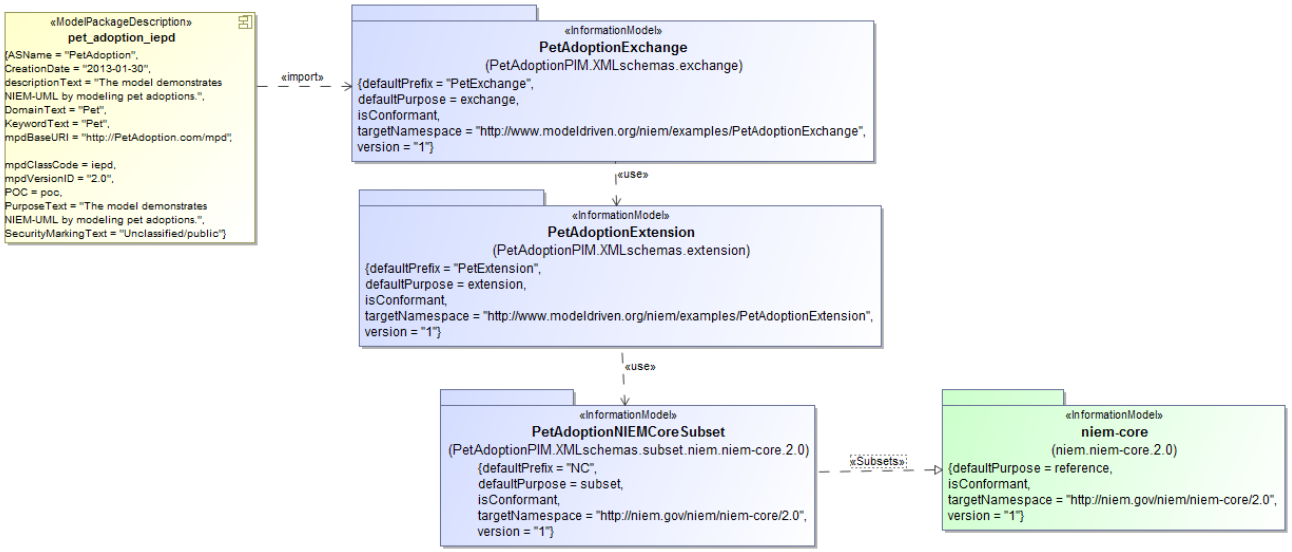

Figure 14 Pet Adoption IEPD

In a GRA-UML PIM, Interfaces are provided by Ports on Components. In this example there is a single Component with a single Port – called ServicePort - which provides the PetAdoptionInterface, as shown in Figure 15.

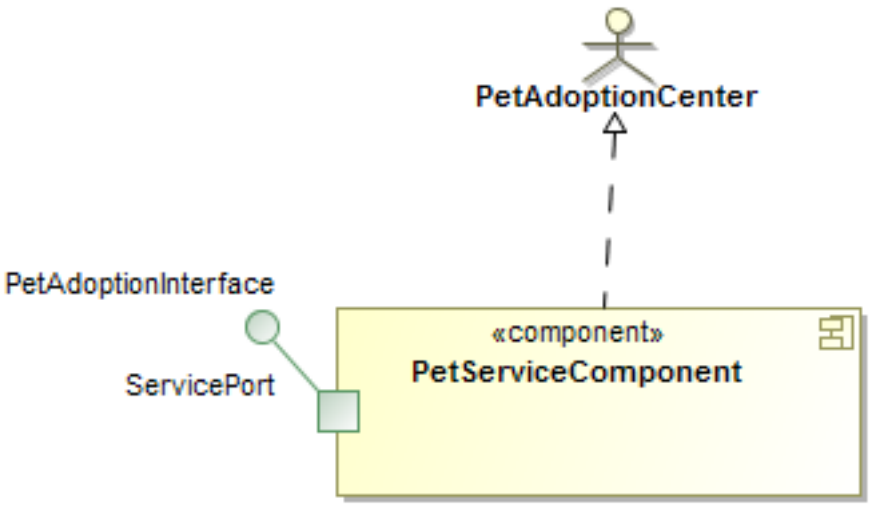

Figure 15 PetServiceComponent

Figure 15 also shows PetServiceComponent realizing the PetAdoptionCenter actor: the dashed arrow with the open triangle arrowhead signifies a UML Realization. This means that the Component represents a service offered by that Actor.

The PetServiceComponent realizing the PetAdoptionCenter may also be shown on the UseCase diagram, as illustrated by Figure 16.

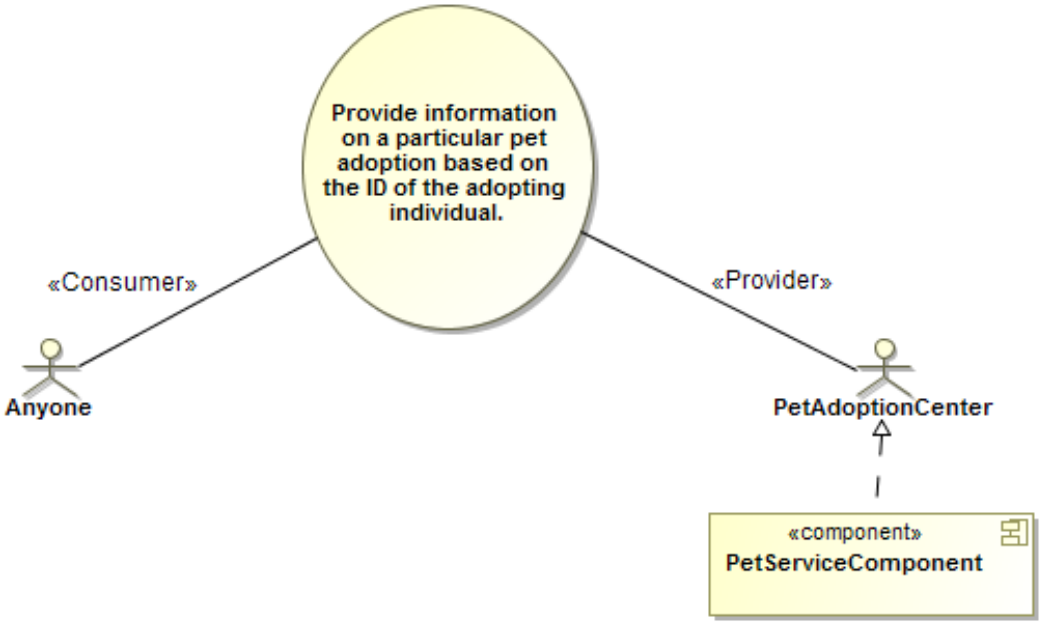

Figure 16 UseCase diagram showing Component realization

The central element of the PIM is a single UML Collaboration that represents the community involved in the provision and consumption of pet adoption services, and pulls together the participants by representing them as roles in the Collaboration.

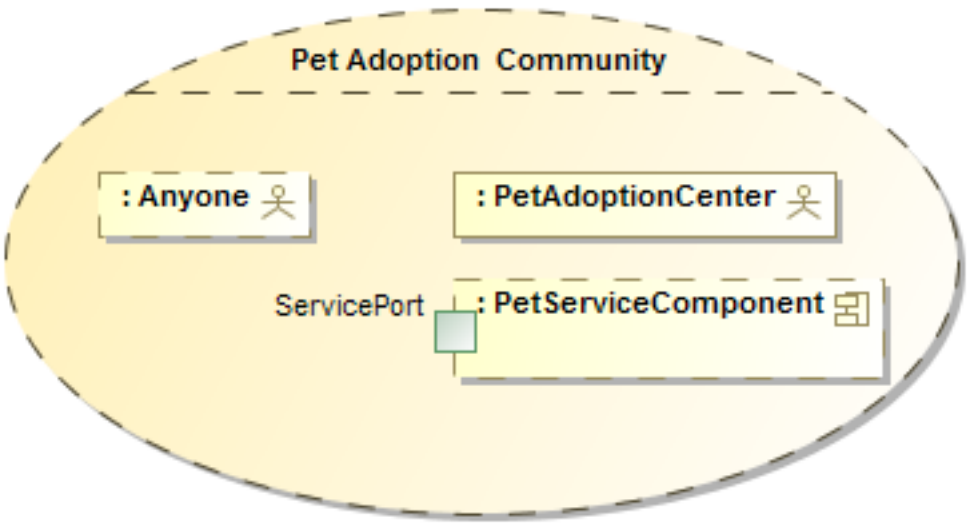

Figure 17 Pet Adoption Community Collaboration

The final aspect of the Pet Adoption PIM is shown in the Interaction in Figure 18, in which Anyone synchronously invokes the GetAdoptionInformation operation on the PetAdoptionInterface offered by a PetServiceComponent.

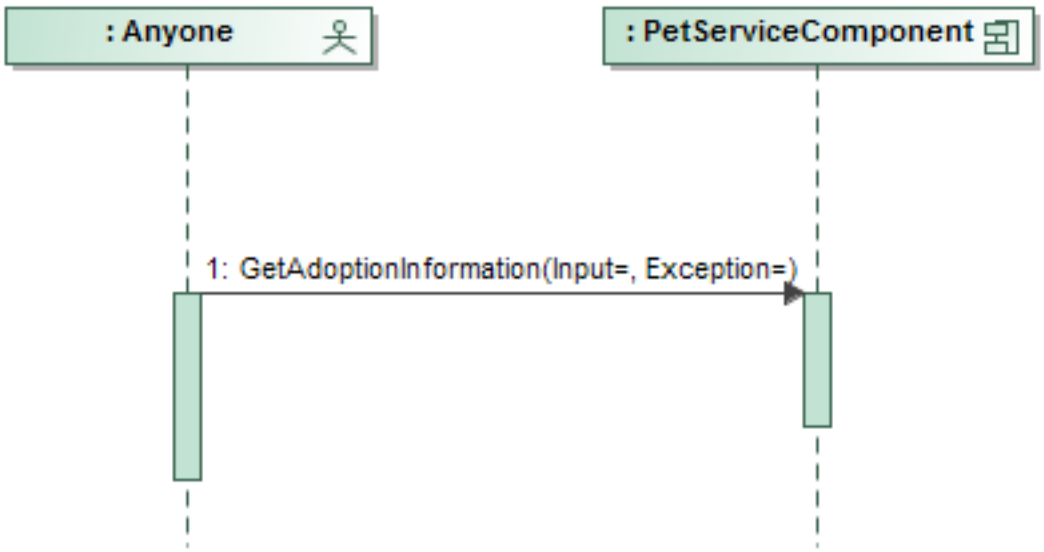

Figure 18 Pet Adoption Interaction

In the UML model, such Interactions are owned by the Collaboration that represents the subject community, and their Lifelines represent roles that the actors and components play in that Collaboration.

Together, these elements – the Collaboration, UseCases and Actors, Components, Ports, Interfaces and Interactions comprise the complete PIM for the example. This, on its own, is insufficient to generate the complete SSP. To complete the story it is necessary to provide annotations that add metadata sufficient for the generation task. These annotations are carried by UML InstanceSpecifications (which for readability we will call instances) whose types are classes in the GRA Annotations model specified in clause 9.

Figure 19 shows the annotations that primarily relate to the generation of the Service Description Document (SDD). The focus of this model is the Pet Adoption Service Description instance. The UML notation for an instance is a rectangle whose label contains the name of the instance, followed by a colon, followed by the type of the instance. In this case the label is Pet Adoption Service Description : ServiceDescription, indicating that the name is Per Adoption Service Description and the type is ServiceDescription.

The Pet Adoption Service Description realizes the Pet Adoption Community collaboration and uses the pet_adoption_iepd. In any GRA-UML PIM it is required that each ServiceDescription instance realizes a collaboration that represents its community, and uses an IEPD that encapsulates its information model. The two-phase generation process uses the ServiceDescription instance to initiate the generation of all of the artifacts that will ultimately constitute the SSP.

In the main body of the Pet Adoption Service Description are a number of properties, in alphabetical order, together with their values in the form propertyname = value. Many of these properties have simple types: String, Integer, Boolean. For example, MajorVersion = 1 and ServiceNameAbbreviationText = “PET”. These property values flow through the generation process and find their place in the generated SDD.

Linked to Pet Adoption Service Description are several other instances. In most cases, the link is also redundantly represented in the main body of Pet Adoption Service Description. For example the third property in the body appears as DataProvenance = Data Provenance. This is a redundant representation of the link depicted as a line between Pet Adoption Service Description and the Data Provenance instance, which also carries the property name DataProvenance.

Note: Depending on the tool being used, depicting such properties redundantly in the body of the instance may be a display option, which a user might choose differently.

Eight of these linked instances have the type Description: Additional Information, Purpose, Scope, Execution Context, Security, Privacy, Data Provenance and ServiceAssumption.

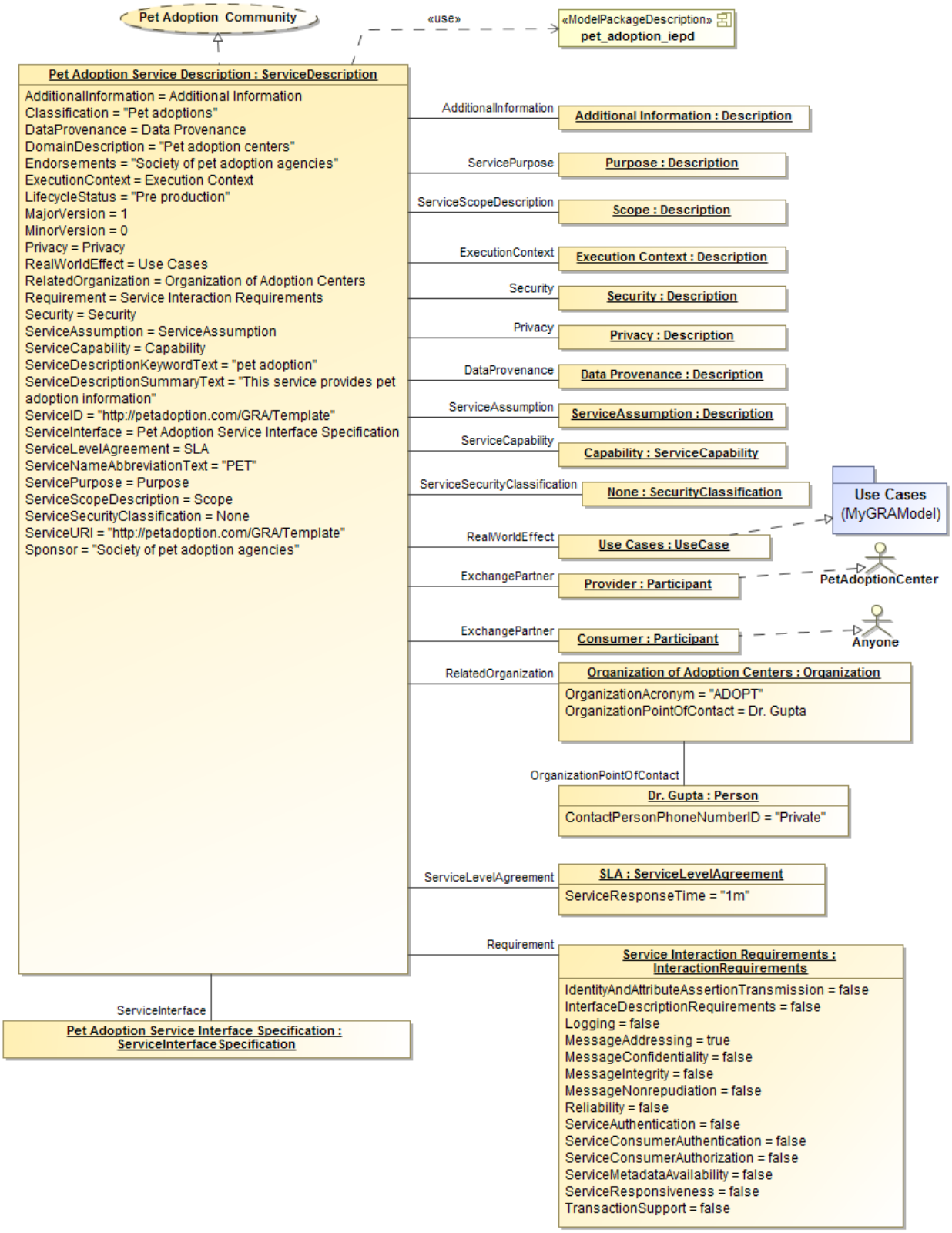

Figure 19 Service Specification annotation model

Although it is not shown on the diagram, each instance carries its own documentation. In UML terms, this is a single comment owned by the instance. Many UML tools have a special user interface to display this comment outside of the diagram. In the case of Data Provenance, the documentation comment is shown in Figure 20: this will flow through the generation process and ends up in the SDD's Data Provenance section.

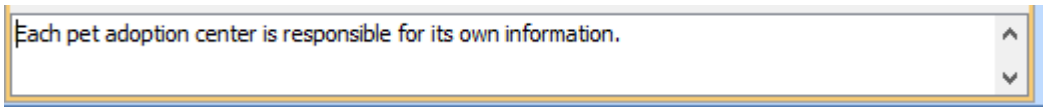

Figure 20 Data Provenance documentation

The class Description also provides the possibility for a property containing a URL of external documentation, which would be incorporated into the generated SDD. None of the instances in this example take advantage of this capability.

Other instances in the diagram have different types: Capability : ServiceCapability; None : SecurityClassification; Organization of Adoption Centers : Organization; Dr. Gupta : Person; SLA : ServiceLevelAgreement. Each of these provides metadata that flows through the generation process into the SDD documentation.

Some of the instances play especially important roles in the generation process. Service Interaction Requirements : InteractionRequirements provide top-level defaults for the interaction requirements, as described in section 7.4.5. They may be – and in this example will be – overridden at low levels of the hierarchy.

The RealWorldEffect property refers to the instance UseCases : UseCase. Notice that UseCases realizes the Use Cases Package in the PIM. The effect of this is that the Phase-1 generation will create a RealWorldEffect for every UseCase in that package. In fact there is only one - "Provide information on a particular pet adoption based on the ID of the adopting individual" – but if there were several UseCases in the package, then a RealWorldEffect would be generated for each of them. Don't confuse the GRA Annotation class UseCase with the UML metaclass with the same name: the GRA Annotation UseCase instance is used to tell the generation process which UML UseCases in the PIM should be included in the output. For example, a model containing one annotation UseCase element that realizes a package containing three UML UseCases will cause the generation of three annotation UseCase elements in the intermediate annotations.xmi file, each referring to the corresponding PIM UseCase.

The ExchangePartner property refers to two instances of Participant called Provider and Consumer, each of which realizes an Actor: the Provider realizes PetAdoptionCenter, and the Consumer realizes Anyone. These ExchangePartners will appear in the intermediate annotations.xmi file. Just as for UseCases, an equivalent effect could have been produced by a single ExchangePartner realizing the Use Cases package. In that case, the Phase-1 generation would have looked for Actors in the package and generated an ExchangePartner for each Actor.

The final instance shown in Figure 19 is Pet Adoption Service Interface Specification : ServiceInterfaceSpecification. This is an elided representation of the ServiceInterfaceSpecification instance that is shown more completely in Figure 21. Remembering from section 7.1 that an SSP contains one SDD and one or more SIDDs, the annotation model must similarly contain one ServiceDescription and one or more ServiceInterfaceSpecifications.

Pet Adoption Service Interface Specification contains some properties with simple types, such as SecurityDescriptionText = "NONE". It is linked to instances representing the ServiceInteractionProfile, the Service, and an overriding InteractionRequirements instance, in which the property ServiceAuthentication = true instead of the false value which would otherwise have been inherited from the top-level requirements.

Service instances are required to realize Components in the PIM, in order that the generation process can create the corresponding service hierarchy described in section 7.4.4. In this case, the single Service instance realizes the PetServiceComponent.

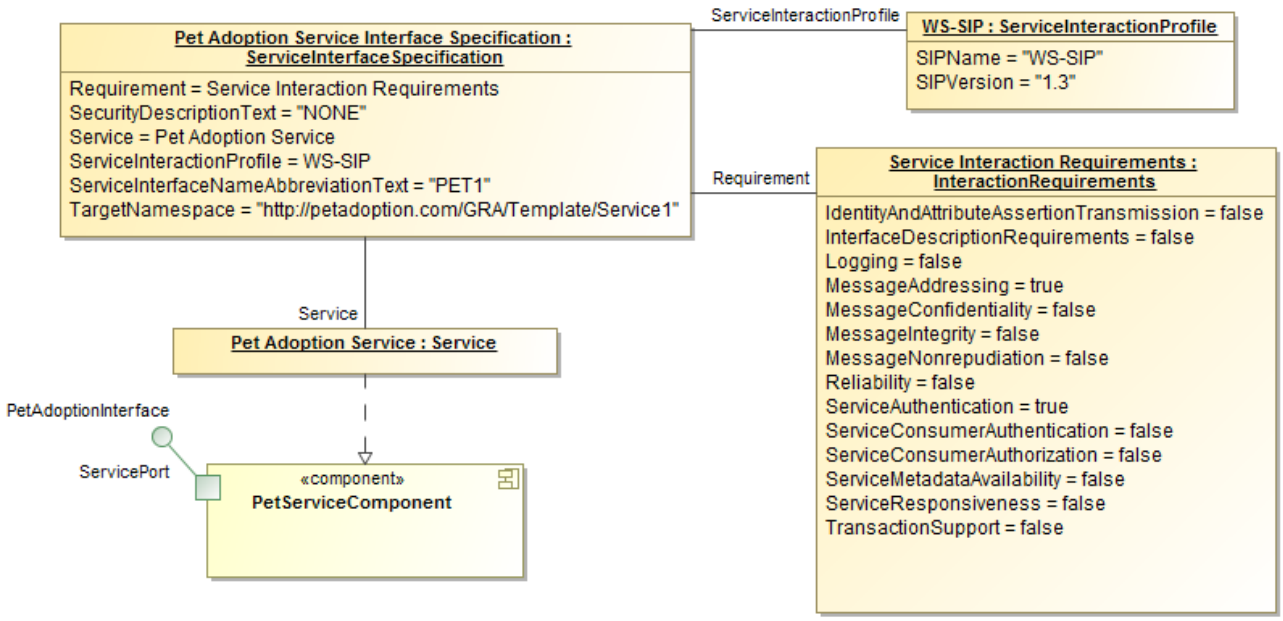

Figure 21 Service Interface Specification annotation model

Annex B: Mappings to other specifications

(informative)

Will be provided in revised submission.

Annex C: Machine Readable Files

Work-in-progress machine-readable files may be found at https://github.com/GRA-UML/specification/tree/master/GRAUML. This includes models, profile definition, examples, templates, schemas and transformations.

The revised submission will contain a complete inventory of normative, non-normative and ancillary machine-readable files.