# GRA-UML

# Scope & Approach

# Design Document

4/16/2014

# Introduction

It is established that GRA-UML will provide for the production of valid GRA service specifications from a UML model. However, exactly what that means is open to quite a bit of interpretation. This document is intended to help arrive at consensus for that scope and approach. Once the general direction for scope and approach is set we will be able to document more detail.

There are 4 primary issues to address:

* That GRA is intentionally loosely specified and open ended in what it can contain. Thus it is unclear where a UML representation stops.
* There is a lot of redundancy and complexity in GRA that could be reduced by abstracting out that redundancy and complexity, but doing so will limit the set of possible GRA specifications that can be represented
* There are few if any completely compliant GRA specifications on which to base “best practice”. The team has provided 2 specifications to use as reference which are now being refined.
* The SDD SIP specifications have business requirements and standards for a service reliability and security (e.g. nonrepudiation). There are multiple options as to how those requirements are met and variation as to how they would be implemented.

How to best address these issues will largely be driven by or approach, which we can summarize as one of the following:

* Bottom up – use existing specifications as a guide and make sure we can represent all variations in UML
* Top down – decide what must be said about a service in a UML friendly way and generate a valid GRA specification

The bottom-up approach tends to expose more of the complexity and technology implications. The top-down approach will not allow anything that could possibly be said in GRA, and it may have information loss when reverse engineering existing specifications.

# The issues in more detail

## The open ended nature and WSDL Extensions

Prior documents have discussed the WSDL extensions capability in GRA. If, for example, a WSDL extension is added for a specific data tagging technology – does that mean that data tagging and it’s mapping to that technology are part of GRA-UML? If so, there could be hundreds of similar options and variations to specify. If, on the other hand, that policy is represented in a technology specific way – then GRA-UML has not really raised the level of abstraction, just put a pretty picture on what exists. To be practical there must be some limit on the set of extensions to scope that will be in the model.

Within the SIP Specifications, there are a dozen or so references to independent web service extensions, each associated with a specification, namespace, and schema. The SSP reference template contains about 10 different WSDL extensions, of which 5 are not mentioned in the SIP Specifications. The set of published SSPs contain 9 different WSDL extensions, of which 6 are not mentioned in either the SIP Specification or the SSP reference template. An isomorphic representation of these extensions in UML would likely require dozens of stereotypes, awkward modeling patterns, and perhaps too much coupling between technology binding and business requirement.

A general purpose representation for WSDL Extensions would be unwieldy for a modeler. However, it has been observed that there is, in practice, little or no variance between the policies applied across a spectrum of SSPs. If the key patterns of policy application could be identified, it would substantially simplify the modeling process and improve consistency across provisioned SSPs.

The objective of the SSP provisioning process is to produce an SSP conformant with the GRA architectural requirements, including the Reliable Secure Web Services SIP. Compliance with the RS-WS SIP conformance targets can be achieved with minimal modeling effort when the key policy application patterns are identified. The goal would be to cover the vast majority of use cases without requiring explicit technology bindings for each wsdl extension point.

Note that in NIEM-UML we faced the same issues and chose to support both a top-down and bottom-up profile. However NIEM is much more structured and has less variation that GRA and the same approach may not be practical. In addition, feedback indicates that even more abstraction away from the technology details would benefit NIEM-UML.

## Redundancy

Consider that some of the same information and summaries could be in metadata, the SDD, WSDL, XSD and a SIP. There is nothing that says, for example, that a summary of purpose or description need be the same in each. So to represent the potential variation each must be a separate model element. On the other hand if we say the model has one summary of purpose it can be provisioned out to multiple documents – but it must then be the same. There is quite a bit of this redundancy.

The existence of redundant sets of schemas for an IEPD and the SIP will be mitigated by the new GRA Service Specification Guidelines. However, there are many other artifacts in an SSP which relate a set of common concepts across different viewpoints. In the case of many of the current SSPs, the identity, description, and relationships of those concepts are not consistent across different viewpoints. This is to be expected in an environment where artifact production is manually implemented, at different points of time, by different providers, and with no means of reliably synchronizing the various forms of artifacts. An objective of the UML model is to ensure that those concepts are semantically consistent, non-redundant, and at the suitable level of abstraction to enable provisioning of a consistent set of required artifacts.

One of the problems with the suite of GRA specifications and templates is lack of precision, clarity, and consistency of concept scope, purpose, name and description across required artifacts. An objective of the UML model is to ensure that each unique concept reflects the composite definition of the GRA concept currently distributed across multiple specifications/templates.

## Best practice and examples

We now have 2 “reference examples” of GRA specifications. Do these represent best practice and the scope of what we need to model? Are they fully correct and complete? Certainly different practitioners will have different opinions.

There have been a number of problems encountered with these reference examples. The problems have generally been fixed in a timely fashion. Problems included existence of spurious or otherwise unreferenced schemas, references to non-existent policies, invalid references from the SSP catalog to metadata, invalid references from the IEPD catalog to schemas, and invalid SSP catalog.

We do not have a “reference example” based on the upcoming GRA Service Specification Guidelines. Those new guidelines eliminate a major portion of redundancy within an SSP, the duplication of schemas between IEPD and SIP. Such an example would provide essential information related to how a SOAP technology binding wraps an IEPD exchange message.

## Business requirements on service implementations

The “double edge sword” of standards is that if they are under specified interoperability is not obtained. If they are over specified it restricts and complicates implementation. For example, within a secure network the same web service may not need the same encryption it would “in the wild”. For many of the business requirements there are several options listed as to the standards to use. In some cases there is insufficient detail as to how those specifications are to be used to enable interoperability. There are permutations and interactions among the various options.

# The questions to be answered

To successfully complete GRA-UML we must know the following:

* What, exactly, should be specifiable in the UML PIM and metadata specification (the metadata specification drives the provisioning of the GRA artifacts)
* What, exactly, are the GRA artifacts to be produced
* How do the models impact what is to be generated
* What are the options and variations at both levels

# Strawman position

Our position is as follows: Our goal should be to make it as simple as possible to produce a valid GRA specification that meets the needs of most users (of course “most” is subjective). We should focus on a user-centric top-down SOA profile that has sufficient information to generate a valid and useful set of GRA artifacts. Being able to consume every possible option of every existing GRA specification is secondary. (note that this does not preclude reverse engineering, it just doesn’t make it the driver)

However, we should be as flexible as possible in allowing various “technology profiles”, where a technology profile is a specific set of GRA standards and details about how they are to be used. The specification should define at least one such profile but allow for others. The “one” we will call the “reference profile”. The choice of the technology profile should be configurable.

The reference profile should allow for the specification of all the business requirements found in the reliable secure web services SIP as well as others identified (the need for logging and audit has already been identified). The profile standards should define how those options provision the reference profile but not exclude others. While there is mention of EbXML, this does not seem to be used and would not be defined in the specification (which would not preclude others from adding it).

The scope of what should be produced as GRA artifacts will be defined by the reference GRA specifications provided (OTNS & SIRS). If a potential capability is not used in a reference specification it is not a requirement for the profile to be able to produce it. This does not preclude other references being introduced in our process, but any such reference should be available no later than 5/1/2014.

While we wish to limit the scope and variability of this specification to come to a conclusion, as much as possible we should strive for allowing configurability and “plugins” of different capabilities.