Considerations for Versioning SOA Resources

Ken Laskey klaskey@mitre.org
The MITRE Corporation
AFCEA SOLUTIONS Series
GMU Critical Issues in C4I
19 May 2009

Agenda

- From the OASIS SOA RM & RA
- The Aspects of SOA Versioning
- Final Advertisement
A Review of SOA Reference Model & Architecture Work

SOA Reference Model (SOA-RM)
- OASIS Standard, 12 October 2006
- Abstract framework for understanding basic concepts and significant relationships
- Independent of specific standards, technologies, implementations, or other concrete details
- http://www.oasis-open.org/specs/index.php#soa-rmv1.0

Instantiation of UML model of service description in SOA Reference Architecture (SOA-RA)
- Follow-on work to SOA-RM utilizing UML modeling
- Abstract realization focusing on the elements and their relationships needed to enable SOA-based systems to be used, realized and owned
- Independent of specific standards, technologies, implementations, or other concrete details
- http://docs.oasis-open.org/soa-rm/soa-ra/v1.0/soa-ra-pr-01.html

Definition of SOA and SOA Service

Service Oriented Architecture (SOA) is a paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains.

A service is a mechanism to enable access to one or more capabilities, where the access is provided using a prescribed interface and is exercised consistent with constraints and policies as specified by the service description.

Note:
- Definition of SOA does not include undefined service
- Definition of service implies capabilities at least notionally exist to solve recognized problem
What This Means for Resources under SOA

- Resources accessed as part of SOA interactions are
  - Independently owned and evolved
  - must be unambiguously identifiable
- For evolving resources
  - Consumer must be able to evaluate how changes to the resource affect appropriateness for use
  - For a service, includes changes to the underlying capabilities, the service access, or the service description

Motivation for This Work

- Lot of discussion on controlling what appears as services
- Too little discussion on managing evolution, tracking change
  - of resources
  - of use of resources
- Ideas presented begin to lay framework relevant to SOA
What is Versioning?

- Assumes simultaneous existence
  - of multiple (different) implementations of a given resource
  - with every implementation distinguishable and individually addressable
- Recognizes process of systematically cataloguing the changes to a resource
- Implies
  - an identifiable resource,
  - a specific set of revisions to that resource,
  - a modified resource that is the result of applying the revisions to the original

The Version Identifier

- Unique label that indicates a specific configuration of a resource
- Commonly
  - an immutable name (e.g. example.txt) and
  - a varying string of nonnegative integers separated by decimal points (e.g. 3.2.1)
- No convention universally used or used consistently
  - Windows 95, Windows 2000, Windows XP, Windows Vista; SP1, SP2, SP3
  - Apple OS X 10.1, 10.2, 10.3, 10.4, 10.5 but this is not consistent with prior to OS X

Conclusion: not only should a version be specified through use of a version identifier, but an explanation should be available on how the identifier is to be interpreted
Versioning and Compatibility (1)

- Typically speak of backward or forward compatibility
- More generally
  - how a command set or information set designed for a predecessor is to be used by the current version
  - how a command set or information set for the current version is to be processed by previous versions
- Consumer of the resource
  - Typically prepared to engage a certain version of the resource
  - How well can a consumer accustomed to one version deal with a predecessor or future version of the resource

Versioning and Compatibility (2)

- Compatibility is determined with respect to
  - a revision
  - the resource or consumer reflecting (providing or using) that revision
- Compatibility depends on context
- General statement that something is backward or forward compatible is meaningless unless it is stated against what compatibility is being assessed

**Conclusion:** description of version must provide basis for consumer making compatibility decision
Versioning and Sufficiency (1)

- If “compatible”
  - older resource can receive message constructed for newer consumer, can process message in terms of its understanding, perform functions consistent with its older context
  - older consumer may receive a response from newer resource that contains unexpected information, just make use of the content it finds consistent with its older context
- Question not addressed by compatibility: are results of interactions across versions sufficient for the intent of the consumer?

Versioning and Sufficiency (2)

- Consider
  - A new resource has new functionality that can be invoked through new terms in the schema used by the message payload
  - The new terms are added through an extensibility mechanism built into the original schema
  - The original schema can validate the new payload without understanding that new functionality is desired
  - A new message sent to the old resource can generate and return reasonable results but not necessarily the results required by the consumer

Conclusion: versioning policies on the part of the consumer may be needed to define when interacting across compatible versions is appropriate for generating desired effects
Note on Policies

- Policies for SOA are typically addressed in terms of the service policies
- The consumer may also have policies
- The two policy sets must be reconciled if interaction is to proceed

Note about Service Endpoints

- Every version of the resource could be reachable through a different endpoint
  => the consumer explicitly chooses the endpoint and thus the version to be used
- A resource provider may simply want to reuse the endpoint for new versions
  - No externally available version for Google but its ranking algorithms are often altered
  - Single, stable endpoint at www.google.com
  - Consumers use whatever version is currently accessible from that endpoint

**Conclusion:** the unstated context “use whatever is at the endpoint” often becomes the default versioning policy
Versioning for SOA Services – The SOA-RA Resource

- Resources have descriptions
- Descriptions reference one or more identifiers by which the identity of the resource is established
- SOA-RA models the general concept of description, service description as extension of general description model
- Both services AND service descriptions are resources

Versioning of SOA Services – Connection with Description

- Service as resource has identifier; version designator should be part of identifier
- Service description provides information on
  - What a service does: business functions, real world effects
  - How to communicate: message semantics, structure; actions resulting from messages
  - Conditions for use: policies
  - Metrics indicating performance
  - Details for reaching service: endpoints, protocols
- Changes could effect any service information

Conclusion: change in version designator must in some manner be able to reflect changes of interest to possible consumers
Versioning of SOA Services – What Changes to Reflect

- Change could derive from underlying capability or service as access to that capability
- SOA principle of opacity
  - Consumer cares only about what affects and what results from the interaction
  - Specifics of where any changes occurred in the implementation are generally irrelevant
- Components, other resources
  - likely have their own configuration management and versioning conventions
  - but these are generally not of interest to service consumer

Conclusion: not necessary to explicitly capture version information about component capabilities or component services from which the service of interest is constructed

Versioning of Service Description

- Possible changes in the description that may not directly derive from changes to the service
- Consider
  - correcting errors that do not significantly change the description, e.g. a simple typo
  - correcting errors that do significantly change description, e.g. the word NOT was missing from the functionality description
  - adding information, e.g. an additional real world effect that was previously considered inconsequential
  - removing information that was previously required or thought useful, e.g. the number of times the service has been used
  - consolidating specifics and substituting link, e.g. version history
- Importance likely depends on context of use

Conclusion: any change in the service description should be reflected in a new version for the description
Notes on Description

- from SOA-RM: description is inherently incomplete – one can never describe every aspect of a resource
- Possible that different aspects of description will be captured by different description sets
  - Description for consumer indicates version and characteristics of version
  - Description for CM identifies components and component versions
- Emphasis for current discussion: description needed to enable and support service interaction

Service Description for Multiple Service Versions

- Description must unambiguously identify its subject resource
  - Identity of resource should indicate its version
  - Explanation of the resource versioning scheme should be readily available
- New service version may change nothing intended for previous version, e.g. implementation bug fixed
- Can have multiple service versions concurrently available, some older versions retired

Conclusion: each version of the service should have a unique description, even if the only thing to change is the version designator
Multiple Versions of Service Description

- Possible multiple versions of service description for same service version
- New version of description would supersede any previous one for that resource
  - Description update would take precedence
  - Single version of description for given service at given time
- Distinct versioning of description allows review of past descriptions, even if service no longer available

Challenges of Service Opacity

- Guiding principle of SOA:
  - Consumer should be able to use a service without concern or interest in implementation details
  - “loose coupling”
- Stable interface is of obvious importance; naive to expect implementation changes are of no interest
  - ex: Service accesses new data source to respond to a query
  - Consumer interested if returned values start showing a different pattern from past experience
  - Especially true if new pattern emerged without any obvious action on the part of consumer
- Challenge: balancing value of opacity of implementation against ability to consider implications of underlying change
Conclusions

- Presented considerations, not necessarily solutions
- Thorough understanding of SOA versioning among current topics for SOA-RA work
- Important points
  - Defining and applying a well-documented versioning strategy
  - Related versioning strategy for corresponding service description
  - Understanding and guiding decisions on compatibility and sufficiency
  - Trade-off between "complete enough" description and opacity