This Presentation Courtesy of the
International SOA Symposium
October 7-8, 2008 Amsterdam Arena
www.soasymposium.com
info@soasymposium.com

Founding Sponsors

Platinum Sponsors

Gold Sponsors

Silver Sponsors

Design Patterns for Web Service Contract Versioning

SOA Systems Inc.

About the Book Series

Five titles currently in development for release in 2009.

The Prentice Hall Service-Oriented Computing Series is the top-selling SOA book series in the world.

www.soabooks.com

About the SOA Certified Professional Program

SOASchool.com™
SOA CERTIFIED
Professional

Industry-recognized certification program for the following designations:

- Certified SOA Architect
- Certified SOA Analyst
- Certified SOA Consultant

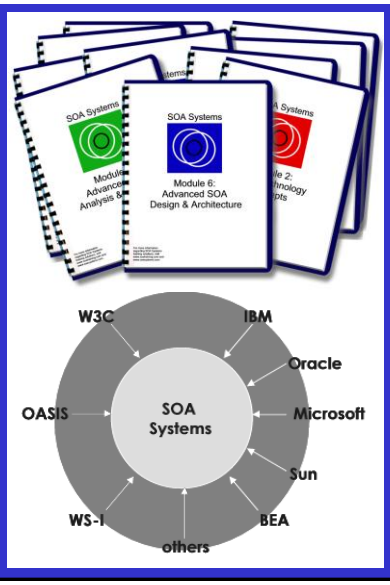
For more information:

- www.soacp.com
- www.soaschool.com

SOASchool.com™ SOA CERTIFIED *Architect*

SOASchool.com™ SOA CERTIFIED *Analyst*

SOASchool.com™ SOA CERTIFIED *Consultant*



Introduction

- A series of versioning patterns covering ways to communicate and support change within services.
- Patterns provide proven and standardized approaches in the governance of change within services contracts and logic.
- Collaborative effort with Thomas Erl and David Orchard as part of SOA Design Patterns, Prentice Hall, 2008.

Service Versioning Introduction

- Service versioning focuses on changes that occur in WSDL, XML Schema and WS-Policy.
- The fundamental starting point is the WSDL definition.
- Version Identification tells consumers whether changes are large or small and extent of compatibility.
- Compatibility identifies the amount of impact changes will have on service consumers.

Overview

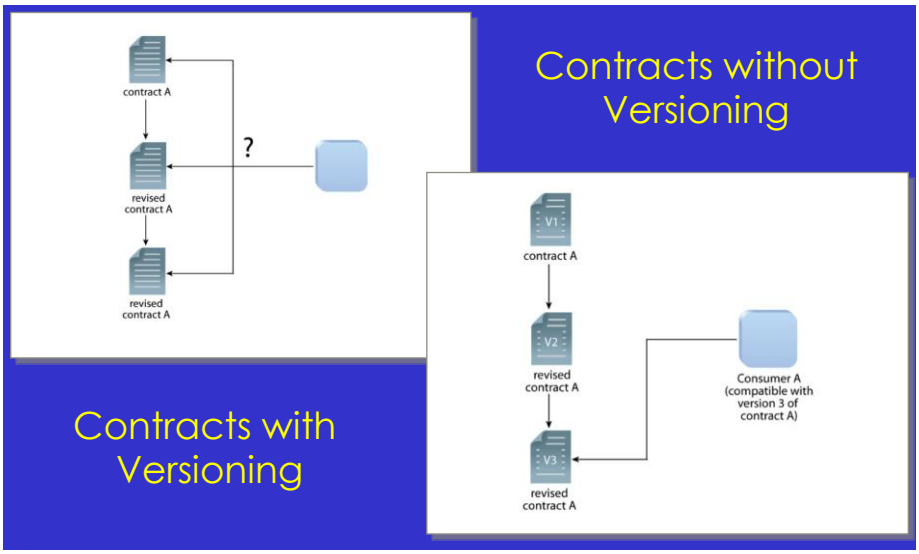
Five Service Versioning Design Patterns:

- Version Identification
- Compatible Change
- Partial Validation
- Termination Notification
- Canonical Versioning

Version Identification

- Problem: Service contracts are changed and there is no mechanism employed to notify consumers.
- Solution: The version of a service contract is important for communication and enforcement. Version identifiers can exist in namespaces and annotations.
- Impact: No standard approach towards version identification and communication with service consumers.

Version Identification



Version Identification

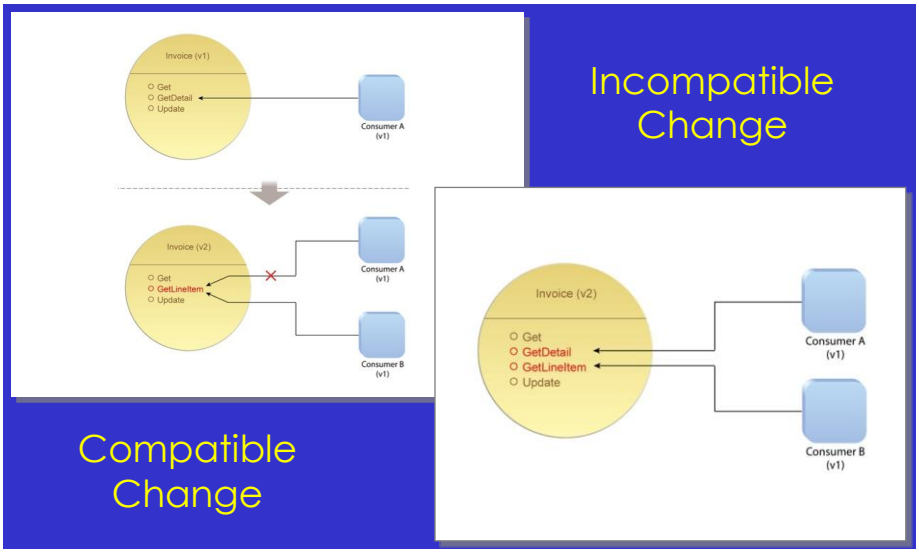
Two approaches for communicating versioning information via identifiers:

- Amount of Work - Major/Minor numbers indicate the level of effort in the change.
- Compatibility Guarantee - Major numbers indicate no backward compatibility. Minor numbers indicate backward compatibility.

Compatible Change

- Problem: Changes to service contracts can break service consumers if applied in an inappropriate way.
- Solution: Compatible changes provides approaches to maintain support for service consumers of different versions.
- Impact: Compatible change can introduce service contracts that are too generic and will still require the need for service governance.

Compatible Change



Compatible Change

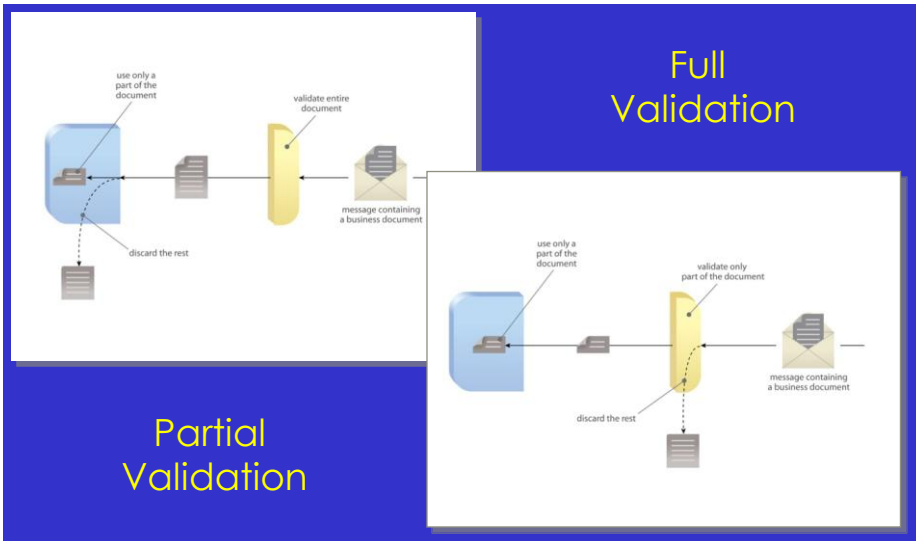
Some examples of backward compatible changes are:

- Adding new operations to existing WSDL
- Adding new optional elements/attributes
- Altering constraint of existing message via coarser validation.
- Addition of policy alternatives

Partial Validation

- Problem: Coarse/generic capabilities in services may provide consumers with unnecessary information.
- Solution: Avoid unnecessary processing of message allowing for loose coupling of consumer from service
- Impact: Increases design and run-time filtering efforts which may reduce the overall benefit.

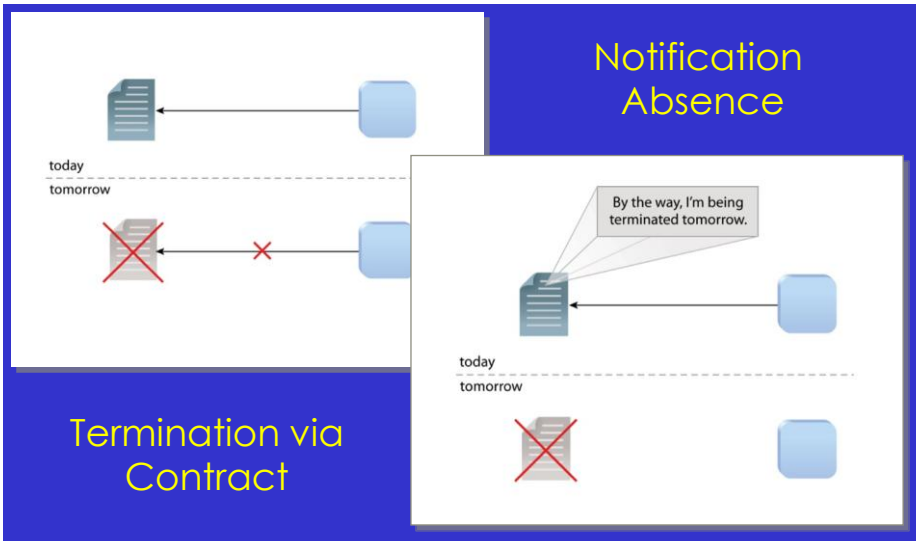
Partial Validation



Termination Notification

- Problem: Service consumers are unaware of service contract changes or service termination.
- Solution: Service contracts can describe service termination information for human/programmatic consumption.
- Impact: Service consumers must be aware of service termination information syntax in order to be supportive of changes.

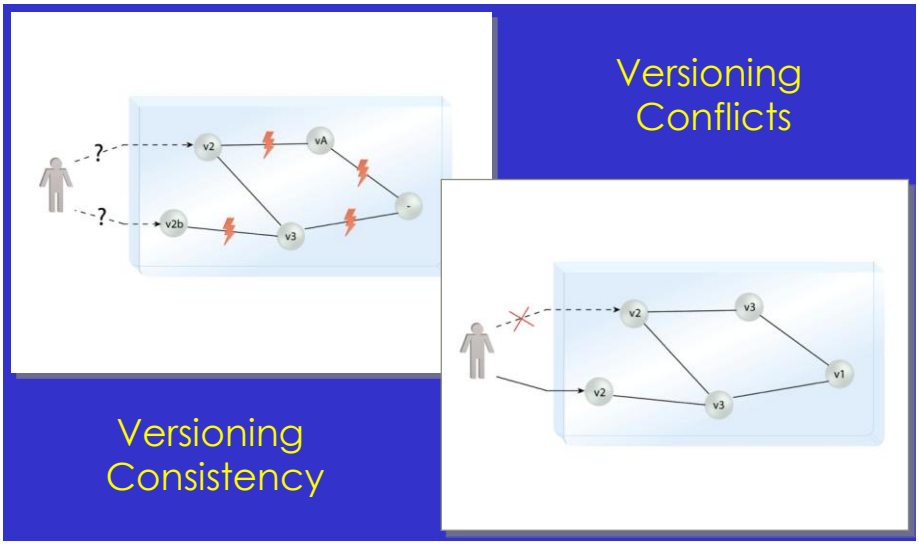
Termination Notification



Canonical Versioning

- Problem: Services within the same Inventory may utilize different versioning approaches which can impact native Interoperability between services.
- Solution: Standardization within the inventory boundary of version identification and version rules.
- Impact: New governance requirements will be introduced to support creation and enforcement of versioning standards.

Canonical Versioning



Canonical Versioning

Three common strategies include:

- Strict - All changes in contracts require a new version. No support for backward or forward compatibility
- Flexible - Focus on backward compatible changes only and any incompatible changes requires new version.
- Loose - Incompatible changes results in new version of contract which uses both backward and forward compatibility.

Q&A

SOA Systems Inc.	www.soasystems.com
SOA Training	www.soaschool.com
SOA Certification	www.soacp.com
SOA Books	www.soabooks.com
SOA Magazine	www.soamag.com
SOA Patterns	www.soapatterns.org
Updates	notify@soasystems.com
Contact	info@soasystems.com