Mind the Opportunities

Towards a Smart Grid Culture of Architecture

William Cox
Cox Software Architects LLC
wtcox@CoxSoftwareArchitects.com

Agenda

- Introduction
- Sound Architectural Approach
- What Makes a Good Spec?
- Design Goals and Techniques
- An Example: Open ADR
- Summary
- For More Information
- Parting Thought

Introduction

- Who am I?
  - Consulting enterprise software architect
  - Elected to OASIS Technical Advisory Board
    - OASIS is the leading eBusiness, Web services, and XML vocabulary standards venue
  - Skilled at building standards and products from ideas to adoption
  - Business, marketing, and technical background
  - http://www.CoxSoftwareArchitects.com/energy

Sound Architectural Approach

- Think through the
  - Requirements
  - Security
  - Things that might happen
  - Things that you think will happen
- You can't always revise standards quickly
  - With capital investments have to last years
  - Firmware updates can extend product life
- Conformance clauses and interop testing
  - Best trialed as specification is built

What Makes a Good Spec?

- "You know it when you see it"
- Apply sound architectural principles while building and evolving
- Clear and easy to understand
  - At least as easy as the business domain…
- Clear what to implement and how to test interoperability and conformance
  - Not the same thing!

Can't Anticipate All Uses

- You can't tell what some bright person may (try to) do with your architecture
- Still, try to anticipate
  - Who foresaw the internet and eCommerce explosions?
  - Hindsight is excellent, foresight isn't
- Mitigate problems by following design principles
- Don't try to solve *only* today's problems
  - Know where you want to be in 5 years

Encourage Next Steps

- Plan where you can for
  - Change
  - Compatible evolution
- Consider versioning
  - Data structures
  - XML schemas
  - Protocols
  - Interfaces
- Consider monitoring of use

Design Goals and Techniques

- Service Orientation Good
- Composition Good
- Policy Separation Good
- Business Models Hard
- Allow for Symmetry
- Strive for Consistency
- Reuse Where You Can

Service Orientation Good

- Enterprise software state of the art
- Reuse, Repurpose, Realign
- Service definitions
  - Large chunks often have better reuse
- Tie business value and actions to technical
- Growing base of tooling and governance
- Architectural level of services difficult to get right
  - May take experimentation, apply experience

Composition Good

- Small specifications can be composed
- Smaller specifications are more easily understood
- Permits architecture with similar interaction in differing environments
  - Accommodate differences by composing (e.g.) security
  - WS-SecureConversation from WS-Security and WS-ReliableMessaging

Policy Separation Good

- Separating policy from mechanism
  - Mechanism is the same, policies can differ
  - Allows reuse with differing policies
- See OASIS Standards
  - WS-SecurityPolicy
  - Other policy frameworks
- Consider along with composition
  - Policy assertions allow variable application for differing circumstances

Business Models Hard

- Understanding (and explaining) the business model is hard to do well
- Appropriate architectural level for the components and the interfaces/interactions
- Architecture must support the business model

Allow for Symmetry

- The producer may be the consumer
  - Today, tomorrow, with a different commodity
  - Micro Grids, rooftop solar are producers
  - Micro Grids are consumers
- Avoid un-needed asymmetry
  - REST is stateless; need for scalability
  - But requiring an open connection adds state
  - Data models can be symmetric; can protocols?

Strive for Consistency

- Reusing an architecture (or piece) makes your solution easier to use
  - Don't do things differently just because you can
  - Easier implementation of new interfaces
  - Don't diverge without excellent reasons
- Design patterns drawn from
  - Other specifications in the domain
  - Common uses

Reuse Where You Can

- Don't reinvent, reference
- If you're not expert in an area, consult those who are
  - May find opportunities for reuse, extensibility
  - Open standards work benefits from wider audience, wider input
- Don't be embarrassed to use architectural and interaction models from elsewhere
  - IPR considerations
  - Better to be the same than 90% the same!

An Example: OpenADR (1)

- Questions to ask…
- Extensible or not? And at what cost?
  - Firmware updates? Network updates? Consistency of partially updated environment?
- Can the communications mechanism…
  - Be used both ways?
  - How symmetric is the communication?
    - Data rates, data payloads,
  - What uses can we think of for the communications?
  - What uses might someone else think of?

An Example: OpenADR (2)

- Architectural level? Is this for a building, an enterprise, or something else?
- Who are the actors/principals?
- Is it symmetric? Can producer/consumer be reversed?
  - They will be in the future, if not today
  - Certainly within the lifetime of the specification
- Versioning of schemas and interfaces
- Conformance and interoperability

Summary

- By taking an architectural approach we
  - Plan for extensibility
  - Improve usability
  - Broaden the application domain
  - Allow for varying levels of complexity and service
    - Composition, policy
  - Consider issues of
    - Symmetry
    - Compatibility
  - Enable reuse of services and architecture
  - Can better evaluate a specification or standard

For More Information

- Email me wtcox@CoxSoftwareArchitects.com
- Talks and tutorials on architecture, free newsletter signup at http://www.CoxSoftwareArchitects.com/
- Energy Focus Area information at http://www.CoxSoftwareArchitects.com/energy
- Other talks in the Architecture track at this conference
- OpenADR http://drrc.lbl.gov/openadr/

Parting Thought

A doctor can bury his mistakes but an architect can only advise his clients to plant vines.

Frank Lloyd Wright

Let's do the best we can to avoid vine planting.