REST & SOAP

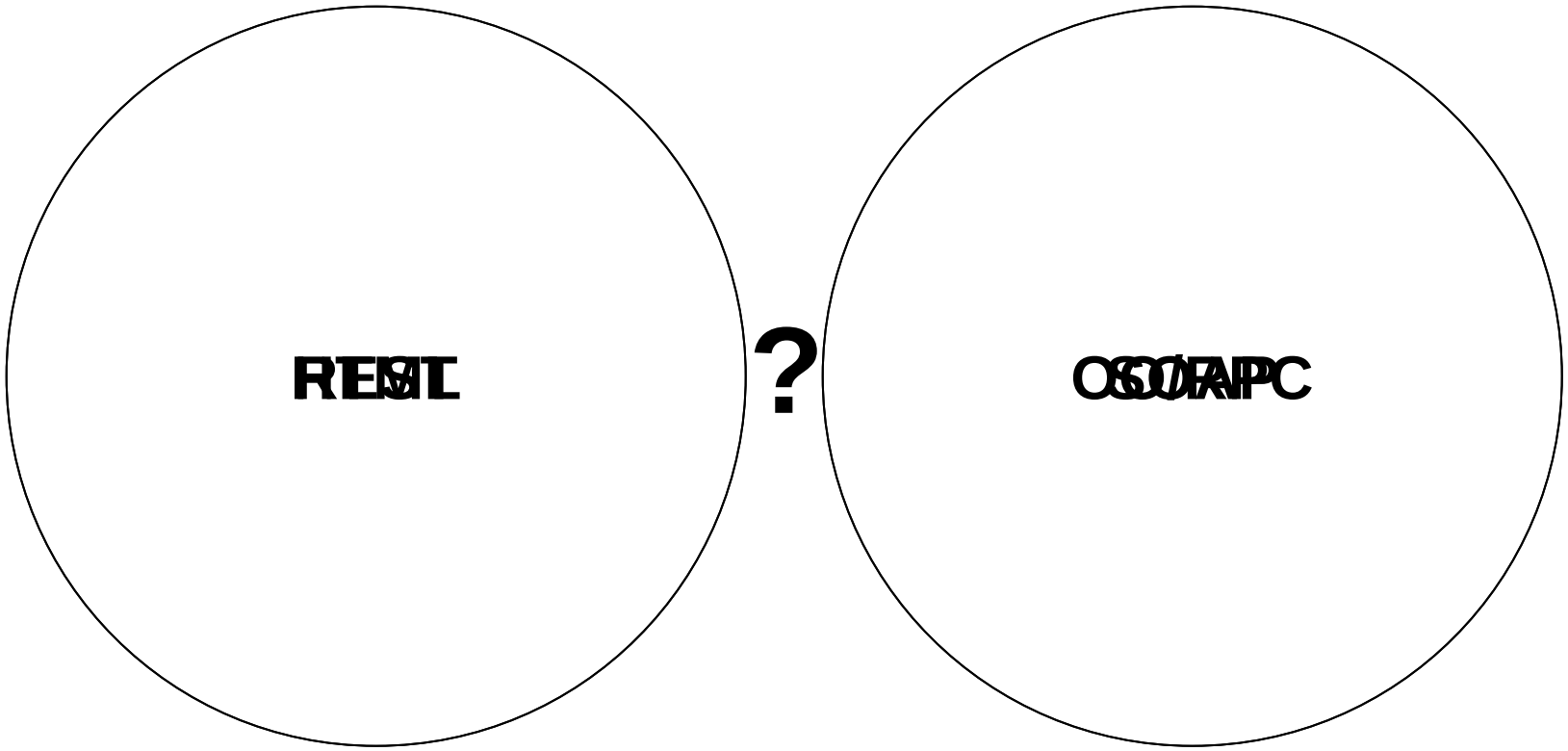
Peter Drayton

peter@razorsoft.com

Agenda

- What is REST?
- How does REST relate to SOAP?
- Can SOAP & REST be combined?

How did we get here?



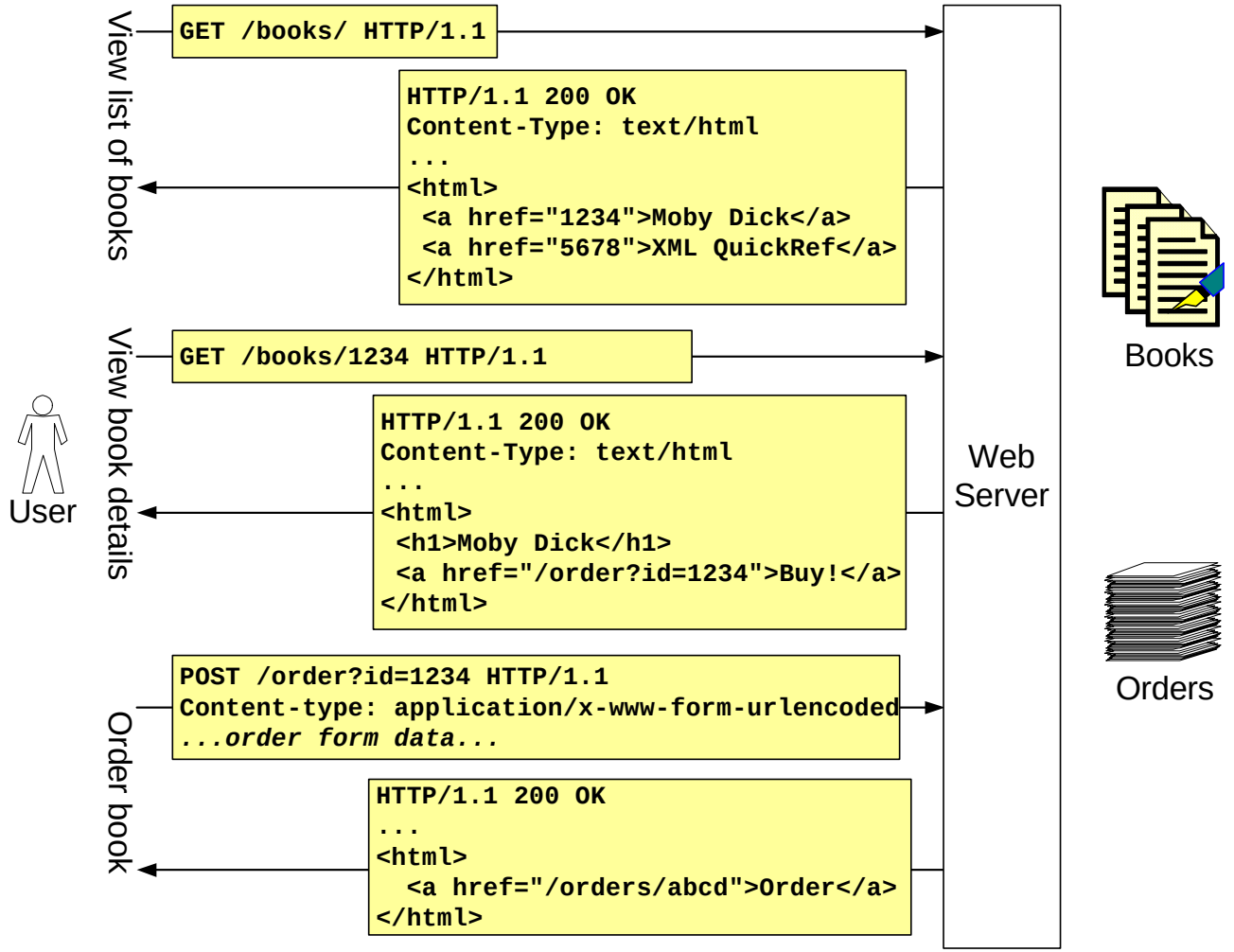
REST == Representational State Transfer

- **REST is an architectural style, not a standard**
  - *"Representational State Transfer is intended to evoke an image of how a well-designed Web application behaves: a network of web pages (a virtual state-machine), where the user progresses through an application by selecting links (state transitions), resulting in the next page (representing the next state of the application) being transferred to the user and rendered for their use." - Dr. Roy T. Fielding*
- **REST emphasizes**
  - Scalability of component interactions
  - Generality of interfaces
  - Independent deployment of components
  - Support for intermediaries

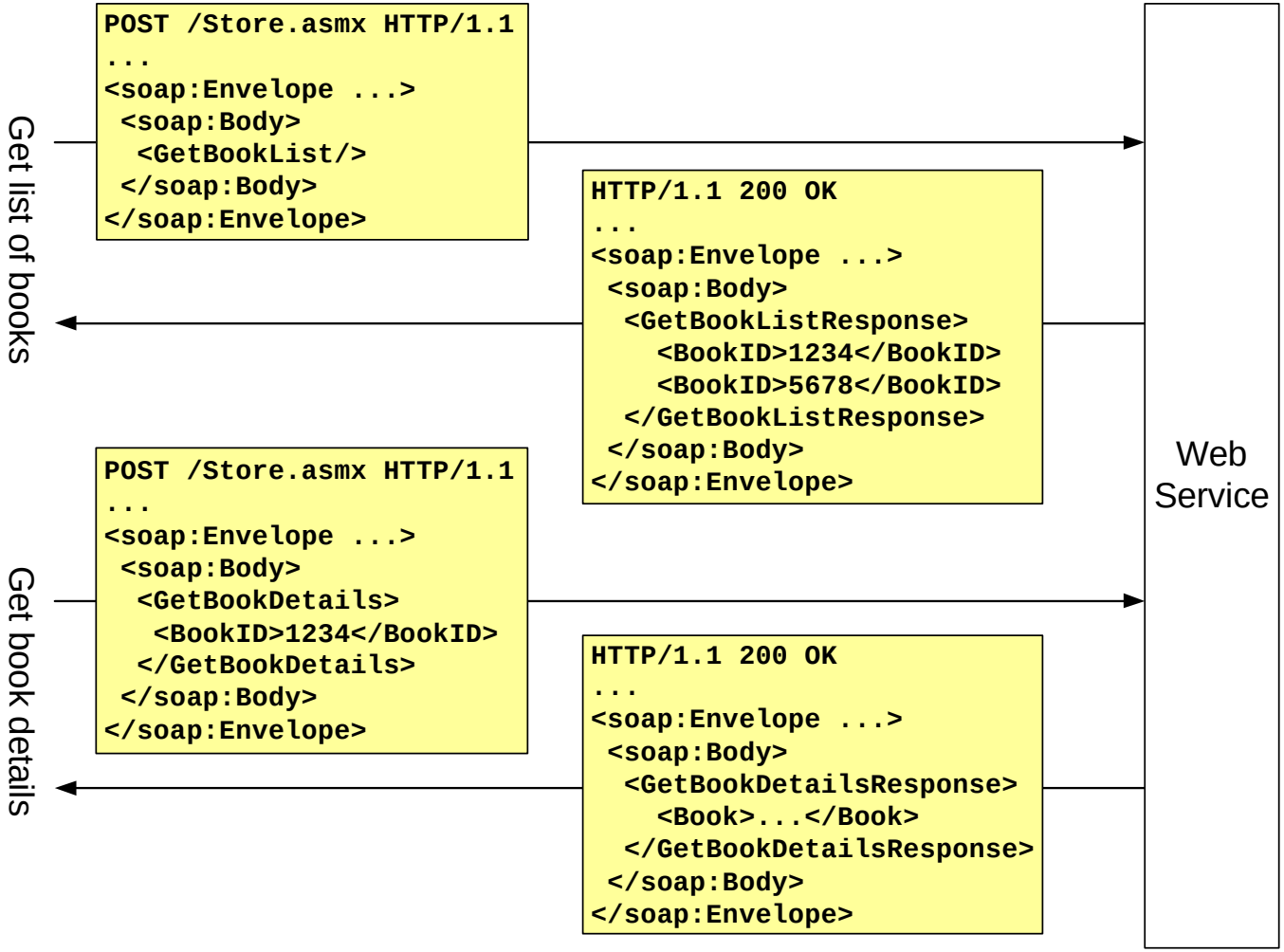
Key Principles of REST

- ***"Identification of resources"***
  - Resources are anything that can be named
  - Naming authority assigns an identifier to a resource
- ***"Manipulation of resources through representations"***
  - Representations capture current/intended state of resources
  - Representations are transferred between components
  - Representations often contain links to related resources
- ***"Self-descriptive messages"***
  - Resource identifiers, representation data formats, control data
- ***"Hypermedia as the engine of application state"***
  - Servers are stateless, messages are independent
  - Clients maintain state (representations) & traverse links
- **Induces scalability, generality, evolvability, extensiblity**

REST & the HTML Web



Common SOAP 1.1



Common SOAP 1.1 (2)

SOAP from a REST viewpoint: Addressing

- **REST architectures utilize the existing web addressing model**
  - Standardized URI schemes subsume protocols (http, ftp, etc.)
  - Standardized distributed naming authorities (DNS)
  - Standardized way of discovering, referring to resources (URIs)
- **SOAP applications define their own addressing schemes**
  - Web service entrypoints have URIs
  - Resources have custom, service-specific addresses
  - No standardized way of discovering, referring to resources
- **Issues**
  - Intermediaries (proxies, caching) cannot operate solely on URI
  - Simple URI-based technologies (XSLT, XInclude) hampered
  - Integrating disparate applications requires custom logic
  - "Deep linking" into applications not generally possible

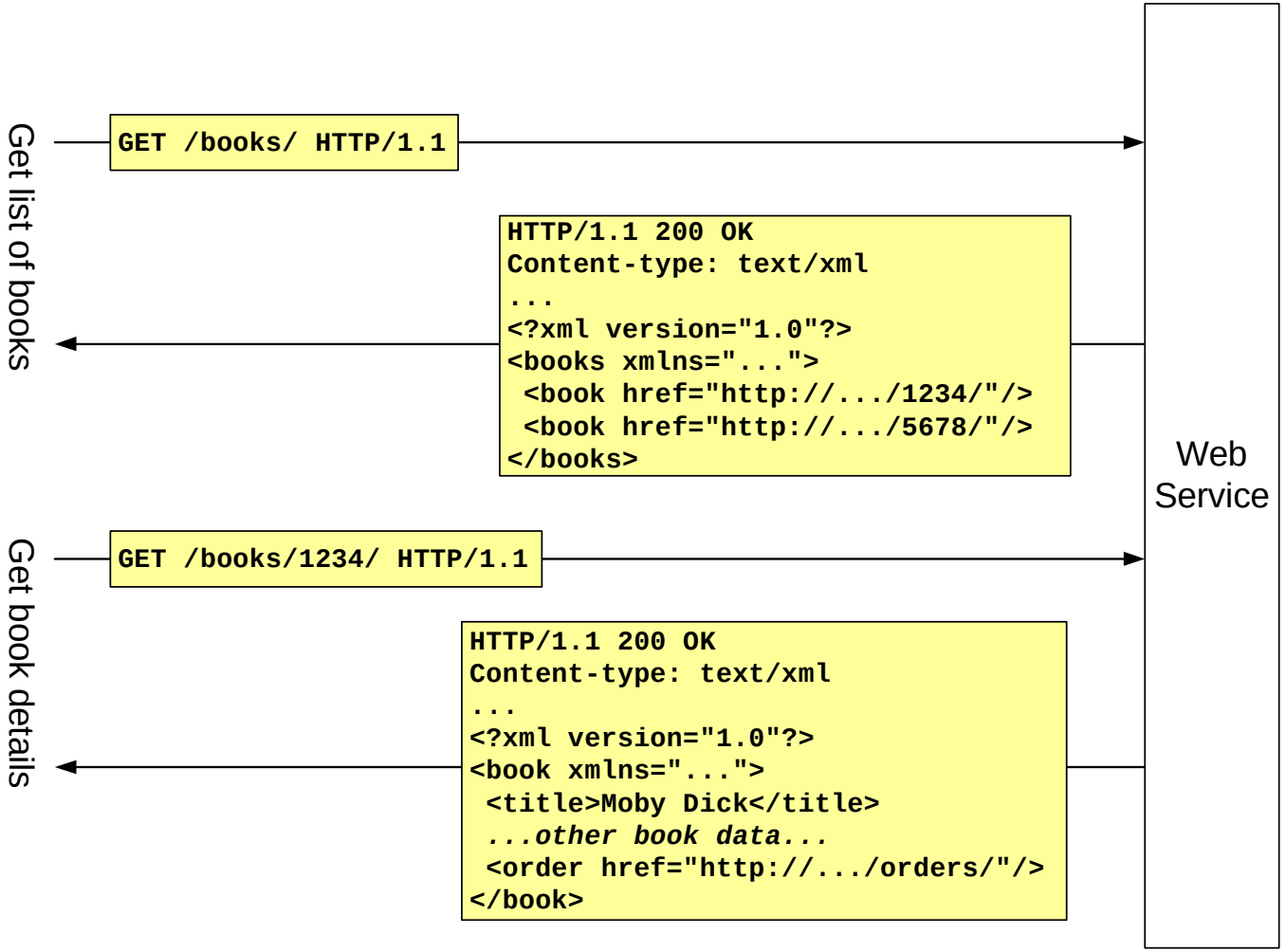
SOAP from a REST viewpoint: Generic Interfaces

- **REST emphasizes standardized, generic operations**
  - Technique widely used (SQL, file system, registry, Hailstorm)
  - HTTP provides CRUD-like PUT, GET, POST, DELETE
  - Allows for uniform manipulation of URI-identified resources
- **SOAP does not provide for generic operations**
  - Each application defines it's own set of operations
  - Creates need for description, discovery mechanisms
  - Knowledge of semantics of operation is out-of-band
- **Issues**
  - Clients need knowledge of description, discovery mechanisms
  - Clients need foreknowledge of specific service semantics
  - Generic clients not universally feasible (local standardization)
  - "Deep linking" into applications not generally possible

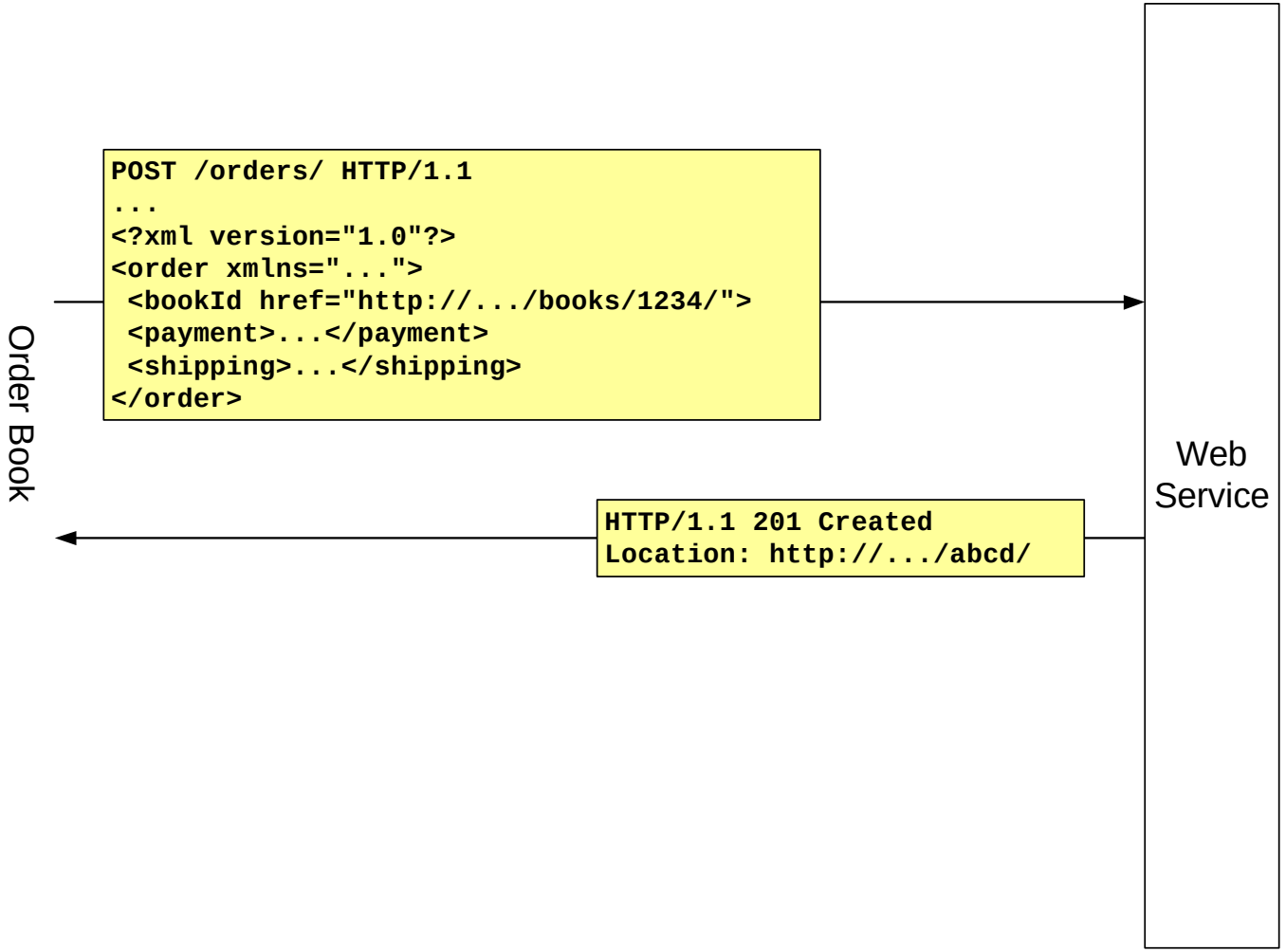
SOAP from a REST viewpoint: State Management

- **REST applications have explicit state transitions**
  - Servers & intermediaries are inherently stateless
  - Resources contain data, links to valid state transitions
  - Clients maintain state, traverse links in generic manner
- **SOAP applications have implicit state transitions**
  - Servers & intermediaries may (should!) be stateless
  - Messages contain only data (not valid state transitions)
  - Clients maintain state, require knowledge of state machine
- **Issues**
  - Clients need foreknowledge of service's state machine
  - Generic clients not universally feasible (local standardization)
  - Limits independent evolution of client/server state machine
  - State machine description needed for automated discovery

REST & the XML Web



REST & the XML Web (2)



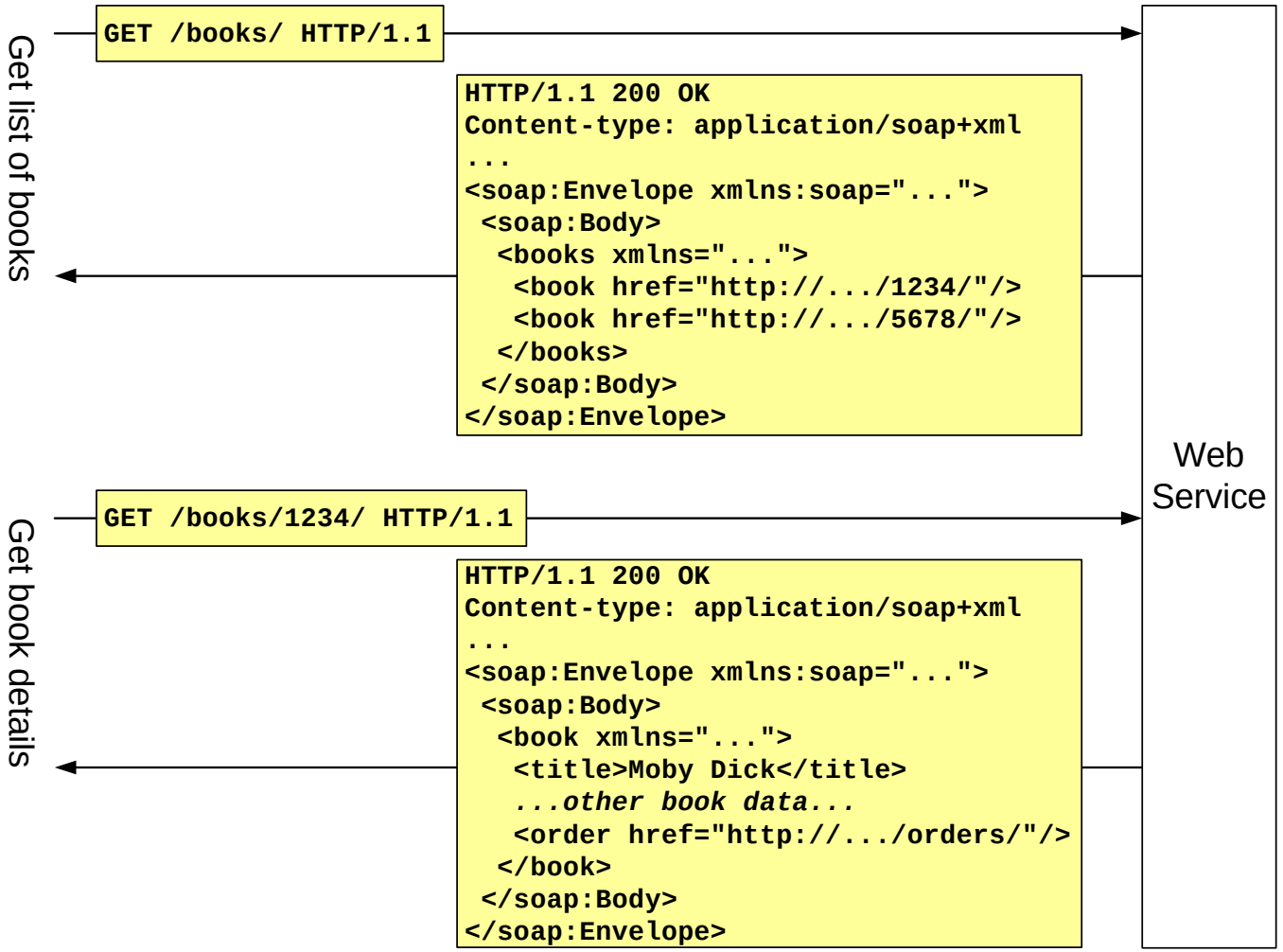
REST from a SOAP viewpoint: Counterpoint

- **SOAP & related technologies have broad industry support**
  - While HTTP is entrenched, WS hype machine is at fever pitch
- **SOAP client & server toolkits are widely deployed**
  - Tool support on client & server matters
- **SOAP headers provide a widely adopted extensibility model**
  - Despite presence of HTTP extension mechanisms
- **WS specifications provide end-to-end protocols/features**
  - HTTP/REST only provides point-to-point solutions
- **SOAP can be bound to other, non-HTTP transports**
  - Important for richer XML messaging in the future
- **SOAP 1.2 can be used in a RESTful manner**
  - *"Can't we all just get along?"*

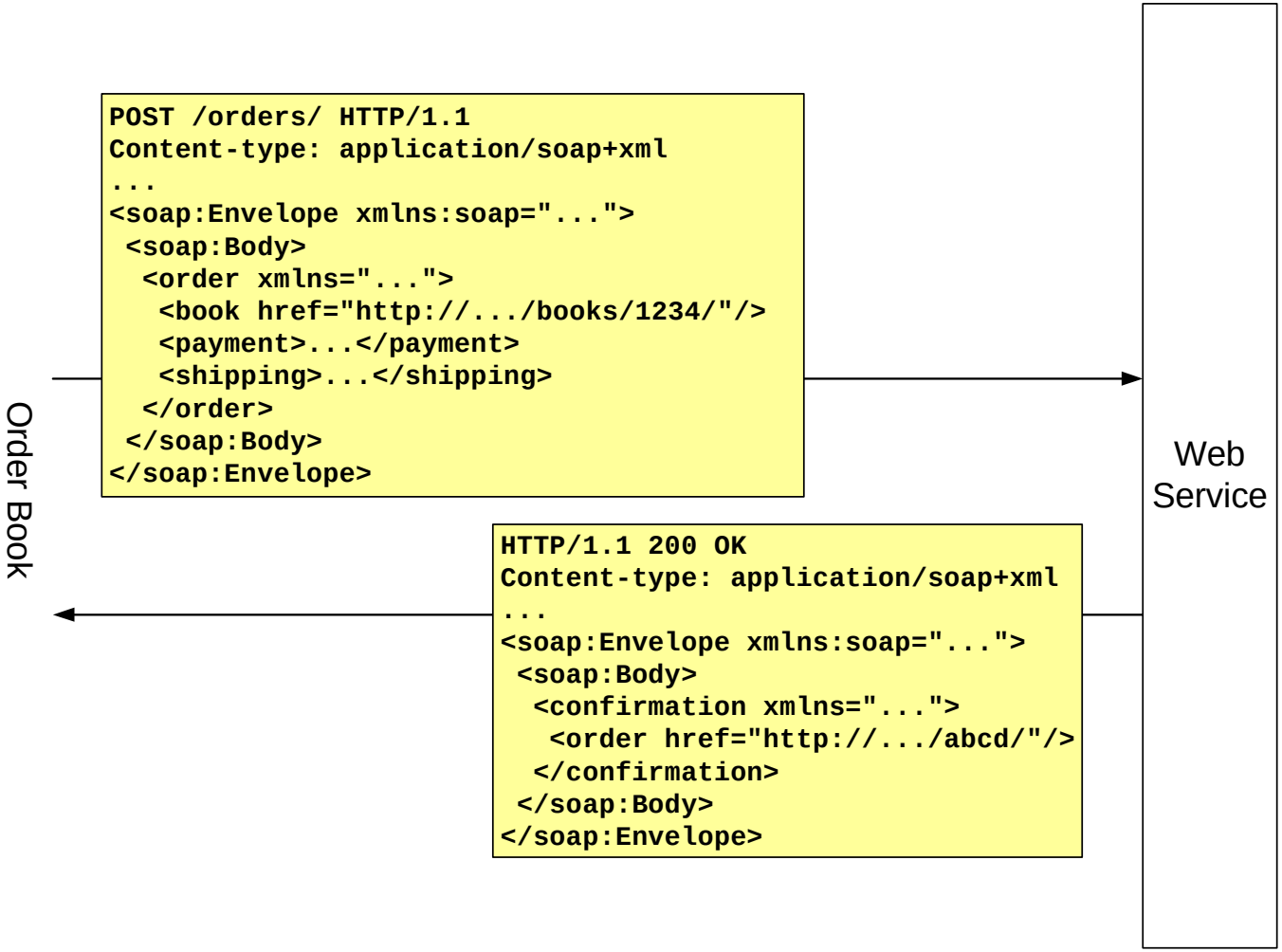
RESTful SOAP 1.2 Guidelines

- **Model your system as a set of resources**
  - Consider both concrete & conceptual resources
- **Assign logical URIs to resources**
  - Nouns, not verbs; Model whole/part relationships
- **Define schemas for resource representations**
  - Plan for future evolution in schema & instance documents
  - Use WXS for interoperability & reuse in WSDL definitions
- **Enable discoverability of resources**
  - Representations include links to contained & related resources
- **Provide appropriate resource manipulation operations**
  - GET for idempotent operations (read)
  - POST for state-modifying operations (create/update/delete/etc)
  - Consider using a combination of POST & GET for queries

RESTful SOAP 1.2



RESTful SOAP 1.2 (2)



Issues

- **Client & Server toolkits**
  - Support for SOAP 1.2 & SOAP Response MEP
  - Large-scale URI-based message dispatch
  - Dynamic modification of client proxy URIs
- **WSDL issues**
  - SOAP 1.2 support
  - Typed service references (R085)
  - HTTP GET-in/SOAP-out (a-la **`<http:urlReplacement/>`**)
- **Tradeoffs**
  - SOAP extensibility model
  - End-to-end properties
  - Alternate protocol bindings

Summary

- REST describes the current Web architecture
- SOAP enjoys broad industry support, extensibility
- Classic SOAP has some limitations compared to REST
- REST principles can be applied to XML Web Services
- Combining SOAP 1.2 and REST is both possible & powerful
- Unfortunately, there are still lots of issues to be resolved

In a web services world, what matters most:
***Web* or *Services*?**

Resources

- **Roy T. Fielding:** ***Architectural Styles and the Design of Network-based Software Architectures***
- **Paul Prescod:** ***REST resources***
- **Sam Ruby:** ***REST + SOAP***
- **Roger Costello:** ***REST Tutorial***
- **W3C: Architectural Principles of the WWW WD, SOAP 1.2 WD**
- **Mailing lists: rest-discuss, rest-explore, www-tag, www-ws-desc, xml-dist-app, soapbuilders**
- **Web sites: RESTwiki, Xml.com**
- **Bloggers: Paul Prescod, Mark Baker, Sam Ruby**