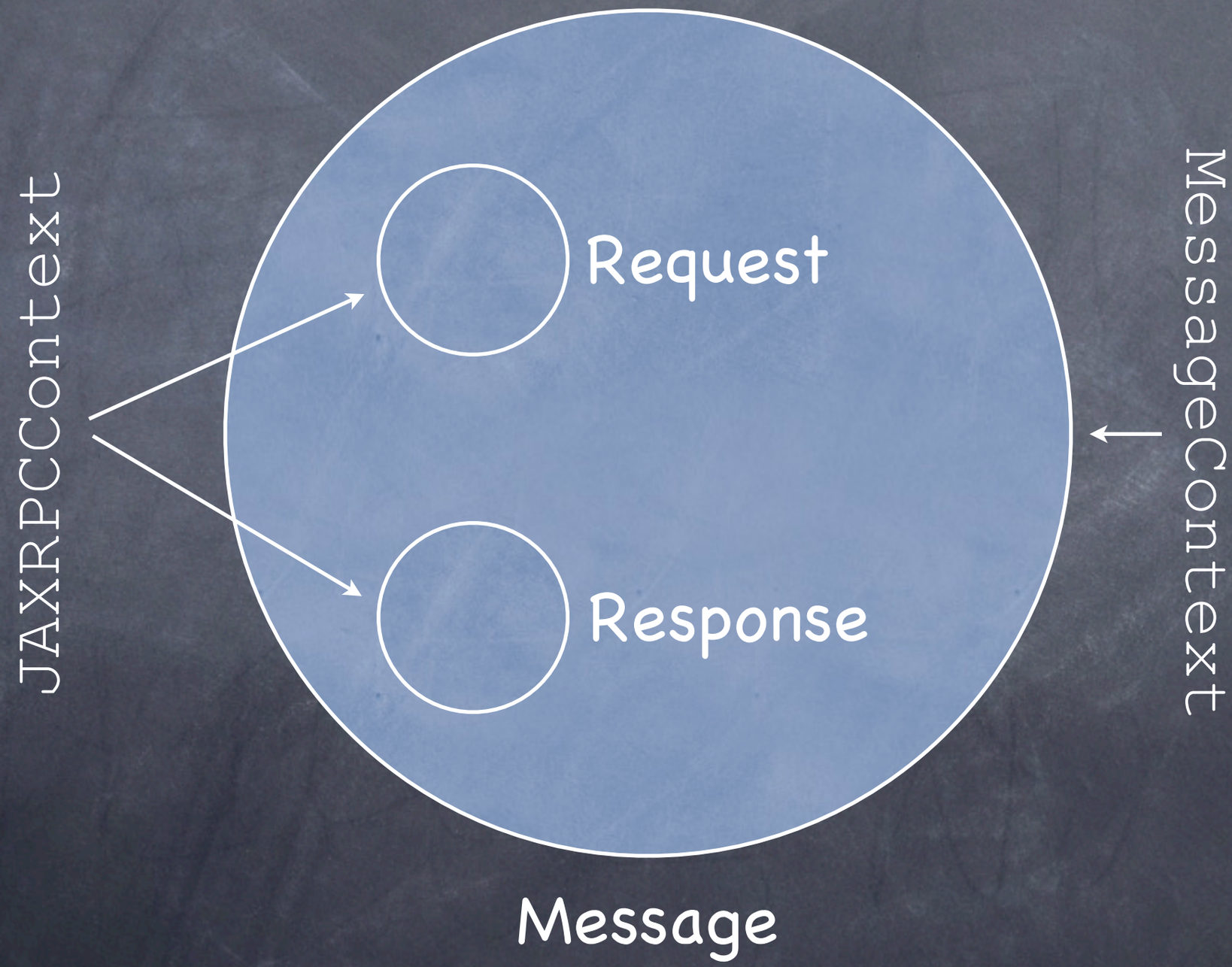
Refactoring Context

Marc Hadley
Sun Microsystems, Inc.

Client-side Context



Use of Context

State	Request	Message	Response
Before	foo=bar	null	Empty
During	foo=bar	foo2=bar2 Request(foo=bar, foo3=bar3) Response(foo4=bar4)	Empty
After	foo=bar	null	foo4=bar4

Advantages

- Request context properties can be set before a sequence of method invocations since the results of each invocation do not affect the request context of subsequent invocations.
- Only a subset of the properties set by handlers are visible to client code.
- Properties that will be made available to client code are clearly demarked from those that are private to handlers.

Disadvantages

- The subdivisions are a client side only construct, writing a handler that works on client and server can be more complex because of this.
- Its not entirely intuitive that changes made to the request context property are not visible to the client code.
- There's no inheritance relationship between MessageContext and JAXRPCContext though they share many similarities.

Proposed Changes

- Make MessageContext extend JAXRPCContext.
- Add the following to MessageContext:
  - `public enum Scope {APPLICATION, HANDLER};`
  - `public void setPropertyScope(String name, Scope scope);`
  - `public Scope getPropertyScope(String name);`

Proposed Changes (cont'd)

- Eliminate the current client-side compartmentalization such that:
  - Properties set in the request context are used to seed the message context for outbound messages.
  - Only properties whose scope is APPLICATION are made available in the response context after a method invocation

Proposed Changes (cont'd)

- Default scope is HANDLER.
- Properties have to be explicitly marked as visible to the application.
- Request context properties are not normally copied to the response context unless their scope is changed to APPLICATION by a handler.

Use of Context

State	Request	Message	Response
Before	foo=bar	null	Empty
During	foo=bar	foo(HANDLER)=bar foo2(HANDLER)=bar2 foo3(APPLICATION)=bar3	Empty
After	foo=bar	null	foo3=bar3

Impact of Issue 12

- If resolved to define classes of endpoint
  - Legacy - can obtain MessageContext, required for backwards compatibility
  - Protocol Neutral - can only obtain LogicalMessageContext (or derivative)
  - Possible to extend property scoping rules to server side