**Architectural Changes**

* **Arch-1:** This issue introduces a significant change by adding support for associating a separate HttpState object with each client in Axis2. This architectural change modifies how the HttpClient manages session and credential data, potentially affecting multiple components (kernel and transports) (Axis2-4288).
* **Arch-2:** This issue implements a consistent version of MTOM (Message Transmission Optimization Mechanism) in alignment with MTOM policy assertions (versions 1.0 and 1.1). This impacts how Axis2 handles message optimization and communicates capabilities to clients via WSDL (Axis2-4220).
* **Arch-3:** This issue introduces a new MessageBuilder capable of parsing and handling various types of email messages, including attachments of any content type, and mapping them to the Axis2 MessageContext. This enhancement impacts the core message handling capabilities in Axis2, qualifying as an architectural change (Axis2-4153).
* **Arch-4:** This issue adds a capability in Axis2’s schema generator to handle "wrapped" array types, modifying the XML schema generation process and impacting the structure of data representation, which is a significant architectural change (Axis2-4073).
* **Arch-5:** This issue introduces support for the WS-AddressingAndIdentity specification, allowing endpoint references to include identity information such as public keys, which is particularly useful in security contexts (Axis2-4066).
* **Arch-6:** This change impacts the internal data handling and unmarshalling processes in the JAXWS layer, adding a structured approach to manage payload data using JAXB more efficiently (Axis2-3546).

**Existence Changes**

* **Exi-1:** The enhancement introduces a new interface generation feature in the codegen component, allowing abstract client interfaces separate from stub implementations. This is an existence decision that expands Axis2’s flexibility in client-side code generation (Axis2-4018).
* **Exi-2:** The lifecycle listener is a new feature that provides additional functionality to Axis2, specifically managing deployment-related events. This qualifies as an existence decision as it introduces a new component that handles specific lifecycle actions (Axis2-3957).
* **Exi-3:** This issue introduces new functionality for binding-level policy attachments, enabling service authors to specify policies in services.xml (Axis2-3677).
* **Exi-4:** The fix optimizes binding creation by introducing a shared binding model, a new approach to managing Axis Bindings across multiple transports (Axis2-3675).
* **Exi-5:** Introduces a ServiceInstanceFactory interface, moving instance creation logic from EndpointController to a factory-based approach, enhancing modularity and extensibility (Axis2-3269).

**Executive Changes**

**Property Changes**

* **Prop-1:** The primary focus is on enhancing performance by using a more efficient caching mechanism for wrapper beans, leading to a more robust and performant marshalling process (Axis2-4775).
* **Prop-2:** The purpose of this change is to improve performance by deferring the initialization of logging until it is necessary (Axis2-4620).
* **Prop-3:** This change improves usability and correctness, ensuring the system behaves as expected when a SOAPAction header is set (Axis2-4392).
* **Prop-4:** Enhances the ServiceBuilder to consistently create binding hierarchies, improving consistency, maintainability, and extensibility of service descriptions (Axis2-3523).
* **Prop-5:** Adds a plugpoint to the JAX-WS server-side code, allowing for custom InvocationListener implementations that enhance extensibility and modularity (Axis2-3505).
* **Prop-6:** Enhances the JAX-WS layer to support standard MTOM configuration options, improving interoperability, performance, and usability (Axis2-3448).
* **Prop-7:** Modifies AbstractContext to lazily initialize PropertyDifferences, optimizing resource usage and performance for non-clustered environments (Axis2-3442).
* **Prop-8:** Enhances the JAX-WS client to allow selective metadata overrides, improving flexibility and usability without changing the architecture’s structure (Axis2-3439).
* **Prop-9:** Adds configurability for thread switching, corrects processing errors, and ensures proper resource management, enhancing configurability and reliability (Axis2-3298).
* **Prop-10:** Modifies WSDL2Java to fully adhere to the XSD by including both elements and attributes, ensuring correct and compliant SOAP message generation (Axis2-3291).
* **Prop-11:** Adds support for file-based configuration to MetadataFactoryRegistry, improving configurability and flexibility for user-defined overrides (Axis2-3264).

**Non-Architectural Changes**

* **Non-Arch-1:** This issue focuses on an enhancement allowing JMS credentials to be set directly via properties instead of relying solely on JNDI entries. This is a straightforward configuration enhancement for flexibility in credential specification and does not involve new components or complex architectural changes (Axis2-4164).
* **Non-Arch-2:** This is a non-architectural change as it does not modify the system’s core structure or introduce significant new components (Axis2-4039).
* **Non-Arch-3:** This issue focuses on fixing a bug in the JAXWS engine to correctly detect and handle SOAPFaults in outbound messages when using a Provider (Axis2-4031).
* **Non-Arch-4:** This change does not involve new components or structural changes; it merely adds a safeguard to release resources appropriately, falling under resource management improvements (Axis2-3966).
* **Non-Arch-5:** This qualifies as a non-architectural change since it’s a small fix within the namespace processing workflow (Axis2-3960).
* **Non-Arch-6:** This is a corrective measure that improves the message processing workflow without adding new components or affecting the core architecture (Axis2-3925).
* **Non-Arch-7:** This bug fix corrects request routing behavior but does not add new components or alter the system’s core architecture (Axis2-3905).
* **Non-Arch-8:** This fix improves file management during AAR generation but does not introduce new components or change the core architecture (Axis2-3881).
* **Non-Arch-9:** This is a bug fix that ensures the correctness of user input by checking XML structure and validity, but it does not introduce new components or affect the system’s architecture (Axis2-3880).
* **Non-Arch-10:** The enhancement focuses on refining how SOAP headers are managed within existing processes (Axis2-3708).
* **Non-Arch-11:** This issue refines attachment handling by integrating with the lifecycle manager for deletion (Axis2-3644).
* **Non-Arch-12:** This issue adds client-side support for checking, improving usability and compliance without altering the system’s structure (Axis2-3620).
* **Non-Arch-13:** This qualifies as a non-architectural change as it refines an existing validation process without altering the system structure (Axis2-3567).
* **Non-Arch-14:** This issue adds an option for asynchronous message receivers in WSDL2Java, enhancing flexibility and performance without impacting the system’s structure (Axis2-3557).
* **Non-Arch-15:** The change is limited to test structure and organization, making it a non-architectural improvement (Axis2-3550).