Summary on Exchange Network Node Interoperability

Problem Statement

It is currently possible to build two different Nodes that comply with the Exchange Network Node Protocol and Specification but are incapable of communicating with one another. There are two distinct issues that are causing the interoperability problems.

Issue #1 – Use of Message Transmission Optimization Method (MTOM)

Issue Summary

This interoperability issue relates to the use of a technology called Message Transmission Optimization Method (MTOM). MTOM is a technology used to more efficiently send messages that have large attachments. The narrative of version 2.0 of the Exchange Network Node Specification is somewhat flexible in defining when a Node must encode messages with MTOM. This flexibility is problematic because it has unintentionally created opportunities for different software development tools to introduce incompatibilities that prevent successful Node communication. Without a more explicit set of rules about when to use MTOM encoding, Nodes inherit the default settings used in software development tools. In some cases, these default settings can be incompatible and cause communication failures between Nodes.

Recommended Resolution

Options for addressing this issue are extremely limited. Practically speaking, the only feasible resolution to this problem is to eliminate any flexibility in Node implementations and require the use of MTOM encoding on all Exchange Network messages. The fix to this problem includes two components:

1. Revising the Node 2.0 Specification to state: "All requests and responses for all operations on the Exchange Network MUST be MTOM encoded."
2. All existing Nodes and Node Clients based on the Node 2.0 Specification must be updated to comply with this rule.

Issue #2 – SOAPAction and Web Services Addressing

Issue Summary

This interoperability issue has two components. The first is the manner in which different Nodes make use of something called the 'Action' element. Developer toolkits for various versions of the .NET framework make use of Action in inconsistent ways. In particular, the toolkits used to create OpenNode2 and the EN Node expect the Action element to be populated with a value that matches the name of a given operation, such as "Submit" or "Authenticate". This default behavior conflicts with the behavior that is defined in the Node v2.0 WSDL. The WSDL specifies that the value of the Action element be left empty. This is one source of interoperability problems.

The second issue has to do with the use of an optional specification called Web Services Addressing or WS-A. WS-A is one of several extensions and enhancements that have been built onto the basic web services specification. The Exchange Network Node v2.0 specification does not specify whether or not extensions like WS-A should be supported. It neither endorses nor precludes their usage. This is problematic because when WS-A is included in a message, its rules *require* the aforementioned Action element to include a value and not be left empty as required by the Node v2.0 WSDL. This implies that there is no way to construct a valid Node v2.0 request that includes WS-A. Some

Microsoft .NET software development toolkits are set up to include WS-A elements in the request by default. That behavior must be overridden by the developer to prevent their appearance in the request.

Recommended Resolution

- Revise the narrative of the Node 2.0 Specification to explicitly state that the WSDL requires the Action element in the HTTP Header of a message MUST be left empty. Adjust and redeploy Nodes that do not currently accommodate this behavior (OpenNode2 and EN Node).

- Revise the narrative of the Node 2.0 Specification to explicitly state that Web Services Addressing (WS-A) elements must NOT be included in any Node v2.0 message.