Change Log for JSR-212 SAMS Messaging API

Proposed Specification Changes for Maintenance Review 2 November 8, 2004

Feedback

Please use the following address to provide feedback on the specification changes described here.

DG.JSR-212-Maintenance@nokia.com

I Executive Summary

1 Specification bug fixes and clarifications

None

2 Specification changes for existing features

2.3 Make it easier to implement the Resource API according to the Java Connector Architecture (JCA) by allowing selected JCA interfaces to be implemented instead of the similar JSR-212 specific interfaces.

3 Specification changes for new features

3.3 Enable the explicit closing of the connections.
3.4 Specify Segment, UdhElement and MessageId as Serializable.

II Detailed Change Descriptions

2.3 Make it easier to implement the Resource API according to the Java Connector Architecture (JCA) by allowing selected JCA interfaces to be implemented instead of the similar JSR-212 specific interfaces.

Overview:

The intention is that Resource API may also be implemented according to the JCA. Currently, it is difficult since the Resource API and JCA have conflicting interfaces, i.e. they have methods with the same signature.

Justification:

All currently specified Resource API interfaces are required to support the implementations that are not based on the JCA. However, Resource API implementations that are based on the JCA are easier or more natural to develop when they implement certain JCA interfaces instead of the equivalent Resource API interfaces.

Proposed change to the specification:

The method getConnection() is removed from javax.sams.spi.mms.MmsConnectionFactory and javax.sams.spi.sms.SmsConnectionFactory to eliminate the conflict with javax.resource.cci. ConnectionFactory. The method validate() is removed from javax.sams.spi.ActivationSpec to eliminate the conflict with javax.resource.spi.ActivationSpec. In the JCA-based implementations the JCA interfaces javax.resource.spi.ResourceAdapter, javax.resource.spi.endpoint.MessageEndpointFactory and javax.resource.spi.endpoint.MessageEndpoint are implemented instead of javax.sams.spi.ResourceAdapter, javax.sams.spi.MessageEndpointFactory and javax.sams.spi.MessageEndpoint respectively. In the JCA-based implementations the javax.sams.spi.Manager class is not used, but the standard mechanisms used with the JCA, such as the JNDI lookup, are used instead to create and access the connection factories and resource adapters.

3.3 Enable the explicit closing of the connections.

Overview:

Currently, the Resource API does not allow closing explicitly connections.

Justification:

It should be possible to explicitly close connections when they are not needed any more since open connections consume resources.

Proposed change to the specification:

The interface javax.sams.spi.Connection is added. It is extended by all specific Resource API connections and it has the method closeConnection for closing the connection.

3.4 Specify Segment, UdhElement and MessageId as Serializable.

Overview:

Currently, the interface javax.sams.messaging.sms.Segment and the classes javax.sams.messaging.sms.UdhElement and javax.sams.spi.sms.MessageId are not specified as Serializable.

Justification:

javax.sams.messaging.sms.SegementedMessage is derived from java.io.Serializable. javax.sams.messaging.sms.Segment and javax.sams.messaging.sms.UdhElement must also be specified as Serializable to enable the easy serialization of SegementedMessage.

javax.sams.spi.sms.MessageId is a container class and it may need to be stored persistently. Therefore, it is justifiable to specifiy it as Serializable.

Proposed change to the specification:

Segment extends java.io.Serializable. UdhElement and MessageId implement java.io.Serializable.

Maintenance Review 17 Sep 2004 - 18 Oct 2004

Accepted Changes

I Executive Summary

1 Specification bug fixes and clarifications

1.1 Clarify when parameters to Service.openSession are invalid.
1.2 Specify that the default for the destination and originator port addresses of a short message is zero.

2 Specification changes for existing features

2.1 Change the attributes message sender address and message handling time stamp of DeliveryInfo from mandatory to optional.
2.2 Change the attribute content id of ContentPart from mandatory to optional.

3 Specification changes for new features

3.1 The getter is added for the maximum number of segments in a short message.
3.2 Enterprise Java session and message-driven beans are specified as alternatives for the current plain Java Session and MessageListener classes.

II Detailed Change Descriptions

1.1 Clarify when parameters to Service.openSession are invalid

Overview:

Depending on the services, there may be some implementation-specific *mandatory* session specific properties needed as arguments to the method `javax.sams.Service.openSession`. Currently, it is specified that `InvalidArgumentException` is thrown if any of the property values is *invalid*. The intent has been that the given property values are invalid also if any of the mandatory properties have not been given, but it has not been clearly written in the specification.

Justification:

A session cannot be opened if any of the mandatory properties have not been given.

Proposed change to the specification:

Unknown properties can be ignored, but if there are invalid values ***or mandatory properties have not been given***, the `InvalidArgumentException` must be thrown.

1.2 Specify that the default for the destination and originator port addresses of a short message is zero.

Overview:

Currently, the specification does not specify a default for the optional destination and originator port addresses of a short message.

Justification:

This change is needed so that the specification is unambiguous from the implementer point of view.

Proposed change to the specification:

The default for the destination and originator port addresses of a short message is zero. It is used if the application does not set the port explicitly. However, if neither of them is set, the port addresses are not sent at all.

2.1 Change the attributes message sender address and message handling time stamp of DeliveryInfo from mandatory to optional.

Overview:

Currently, the attributes message sender address and message handling time stamp of `javax.sams.messaging.DeliveryInfo` have been specified as mandatory, but they should be changed as optional.

Justification:

These attributes should be optional in the API, since the underlying implementation could also add them. The message sender address may be added based on the message id. The message handling time may be set as the current time.

Proposed change to the specification:

In the constructors of `javax.sams.messaging.DeliveryInfo` the arguments `sender` and `date` may be `null`.

2.2 Change the attribute content id of ContentPart from mandatory to optional.

Overview:

Currently, the attribute content id of ContentPart is specified as mandatory, but it should be changed as optional.

Justification:

In a Multimedia Message either content identifier or content location is required only if the type of the message is Multipart/Related. In that case either one is needed for SMIL references and it must be unique over all the content parts of the message.

Proposed change to the specification:

In the constructor of `javax.sams.messaging.mms.ContentPart` the argument `id` may be `null`. Additionally, the methods `addContent` and `setContents` of `javax.sams.messaging.mms.MmsMessage` must throw `InvalidArgumentException` if the type of the message is Multipart/Related and the content identifier or content location has not been set in the `ContentPart` or it is not unique over all the content parts of the message.

3.1 The getter is added for the maximum number of segments in a short message.

Overview:

Currently, the maximum number of segments in a short message can only be set and not queried.

Justification:

It would be useful for an application also to know the maximum number of segments in the short message.

Proposed change to the specification:

The method `javax.sams.messaging.sms.SmsMessage.getMaxSegmentCount` is added for querying the maximum number of segments in the short message.

3.2 Enterprise Java session and message-driven beans are specified as alternatives for the current plain Java Session and MessageListener classes.

Overview:

Currently, only plain Java Session and MessageListener classes are specified. It should be possible also to *package* them as Enterprise Java session and message-driven beans on the J2EE platforms.

Justification:

The JSR 212 is targeted to both J2SE as well as J2EE environments. This change is needed to be able properly and equally support also J2EE environments in addition to J2SE environments, since the enterprise Java session and message-driven beans are the standard and natural way to implement sessions and asynchronous receivers on top of the J2EE platforms.

Proposed change to the specification:

1. EJB home and remote interfaces are added for the JSR 212 sessions so that they can also be alternatively packaged as session beans in addition to the plain Java classes.

2. The methods of `Session` and `MessagingSession` interfaces throw `java.rmi.RemoteException` so that these interfaces can also be implemented by the EJB session beans or used with the RMI.

3. MDBs can be used in the J2EE instead of MessageListeners to receive asynchronous messages.

Deferred Changes

None