**OpenO&M**

**Information Service Bus Model (ISBM) Specification**

**Version 1.0 C4**

This document defines the OpenO&M Information Service Bus Model (ISBM). It defines a underlying logical data model, the web services for the registry, and a normative XML Schema/WSDL specification for the web services.

OpenO&M Information Service Bus Model (ISBM)

Copyright 2011. MIMOSA

All Rights Reserved. http://www.mimosa.org

Parts derived from WBF B2MML-V0401

"The Business To Manufacturing Markup Language (B2MML) is used courtesy of WBF."

*Copyright © MIMOSA 2011. All Rights Reserved.*

*All capitalized terms in the following text have the meanings assigned to them in the MIMOSA Intellectual Property Rights Policy (the "MIMOSA IPR Policy"). The full Policy may be found at the MIMOSA website.*

*This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published, and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this section are included on all such copies and derivative works. However, this document itself may not be modified in any way, including by removing the copyright notice or references to MIMOSA, except as needed for the purpose of developing any document or deliverable produced by a MIMOSA Technical Committee (in which case the rules applicable to copyrights, as set forth in the MIMOSA IPR Policy, must be followed) or as required to translate it into languages other than English.*

*The limited permissions granted above are perpetual and will not be revoked by MIMOSA or its successors or assigns.*

*This document and the information contained herein is provided on an "AS IS" basis and MIMOSA DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY OWNERSHIP RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.*

*MIMOSA requests that any MIMOSA Party or any other party that believes it has patent claims that would necessarily be infringed by implementations of this MIMOSA Final Deliverable, to notify MIMOSA TC Administrator and provide an indication of its willingness to grant patent licenses to such patent claims in a manner consistent with the IPR Mode of the MIMOSA Technical Committee that produced this deliverable.*

*MIMOSA invites any party to contact the MIMOSA TC Administrator if it is aware of a claim of ownership of any patent claims that would necessarily be infringed by implementations of this MIMOSA Final Deliverable by a patent holder that is not willing to provide a license to such patent claims in a manner consistent with the IPR Mode of the MIMOSA Technical Committee that produced this MIMOSA Final Deliverable. MIMOSA may include such claims on its website, but disclaims any obligation to do so.*

*MIMOSA takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this MIMOSA Final Deliverable or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on MIMOSA' procedures with respect to rights in any document or deliverable produced by a MIMOSA Technical Committee can be found on the MIMOSA website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this MIMOSA Final Deliverable, can be obtained from the MIMOSA TC Administrator. MIMOSA makes no representation that any information or list of intellectual property rights will at any time be complete, or that any claims in such list are, in fact, Essential Claims.*

This specification defines an OpenO&M Information Service Bus Model (ISBM) for exchanging the information defined in the ISA 95 Enterprise/Control System Integration standards, OpenO&M Common Interoperability Registry (CIR), MIMOSA OSA-EAI, the WBF Business to Manufacturing Markup Language (B2MML), ISO 15926 information, and OPC UA address space objects that have been converted to standardized OPC UA XML payloads.

The ISBM defines a minimal interface subset to Enterprise Service Buses (ESB) and other XML message exchange middleware, using a standard interface consisting of CHANNELS and TOPICS. The benefit from this approach is to allow applications to expose a single, standardized interface rather than having to be custom built for every version and format of ESB or message exchange system.

The knowledge requirements to interface to just one ESB can be immense, and is usually not transferable to a different ESB. The ISBM defines a single interface, independent of the underlying services, for Level 3-3 and Level 4-3 communications. This removes the need for vendors to build custom interface after custom interface, and for end users to get locked into a single vendor because their investment prevents them from reusing any of the integration efforts.

**Table of Contents**

[1 References 7](#_Toc296515995)

[2 OpenO&M Information Service Bus Model 8](#_Toc296515996)

[2.1 Interface Model 8](#_Toc296515997)

[2.2 Application to Application Data Exchange 8](#_Toc296515998)

[2.3 Transaction Model 10](#_Toc296515999)

[2.4 Communicating Applications 10](#_Toc296516000)

[2.5 Managed Communication Channels 11](#_Toc296516001)

[2.6 ISBM Channel Services 12](#_Toc296516002)

[2.7 ISBM Publication Channel Services 13](#_Toc296516003)

[2.8 ISBM Request and Response Channel Services 16](#_Toc296516004)

[3 Methods of Operation of ISBM Channels 19](#_Toc296516005)

[3.1 ISBM Root 19](#_Toc296516006)

[3.2 Channel Scope 19](#_Toc296516007)

[3.3 Information Scope 20](#_Toc296516008)

[3.4 Channel Use 20](#_Toc296516009)

[3.5 Topics 20](#_Toc296516010)

[3.6 Standard Topics 20](#_Toc296516011)

[3.7 Security 22](#_Toc296516012)

[3.7.1 Security Tokens on Channels 22](#_Toc296516013)

[3.7.2 Security Token Format 23](#_Toc296516014)

[3.7.3 ISBM Service Provider Implementations 26](#_Toc296516015)

[3.7.4 ISBM Application Implementation Considerations 26](#_Toc296516016)

[3.7.5 ISBM Channel Security Considerations 27](#_Toc296516017)

[3.7.6 Notification 27](#_Toc296516018)

[3.8 ISBM Service Provider Considerations 27](#_Toc296516019)

[3.8.1 Security Considerations 27](#_Toc296516020)

[3.8.2 Data Format Checking 27](#_Toc296516021)

[3.8.3 Allowed Application Checking 27](#_Toc296516022)

[3.8.4 Data Exchange Logging 27](#_Toc296516023)

[3.8.5 Common Error Handling 28](#_Toc296516024)

[3.8.6 Data Transformation Services 28](#_Toc296516025)

[3.8.7 Cross Company Bridge 29](#_Toc296516026)

[3.9 W3C Standards 30](#_Toc296516027)

[3.9.1 WDSL 30](#_Toc296516028)

[3.9.2 UDDI 30](#_Toc296516029)

[3.9.3 ebXML Registry 31](#_Toc296516030)

[3.9.4 WS-\* Standards 31](#_Toc296516031)

[4 Service Definitions 32](#_Toc296516032)

[4.1 Type Definitions 32](#_Toc296516033)

[4.1.1 Success and Error Criteria 32](#_Toc296516034)

[4.1.2 ISBM Channel Type 32](#_Toc296516035)

[4.1.3 Channel Name 32](#_Toc296516036)

[4.1.4 Channel ID 32](#_Toc296516037)

[4.1.5 Channel Session ID 32](#_Toc296516038)

[4.1.6 Topic ID 32](#_Toc296516039)

[4.2 ISBM Channel Management Services 32](#_Toc296516040)

[4.2.1 Assign Security Token 32](#_Toc296516041)

[4.2.2 Create Channel 33](#_Toc296516042)

[4.2.3 Create Channel Namespace 33](#_Toc296516043)

[4.2.4 Create Topic 33](#_Toc296516044)

[4.2.5 Delete Channel 34](#_Toc296516045)

[4.2.6 Delete Channel Namespace 34](#_Toc296516046)

[4.2.7 Delete Topic 34](#_Toc296516047)

[4.2.8 Get All Channels 35](#_Toc296516048)

[4.2.9 Get All Sessions 35](#_Toc296516049)

[4.2.10 Get All Topics 35](#_Toc296516050)

[4.2.11 Get All Users 35](#_Toc296516051)

[4.2.12 Get Channel Info 36](#_Toc296516052)

[4.2.13 Get Session Info 36](#_Toc296516053)

[4.2.14 Get Topic Info 36](#_Toc296516054)

[4.2.15 Get User Info 37](#_Toc296516055)

[4.2.16 Remove Security Token 37](#_Toc296516056)

[4.3 ISBM Notification Channel Services 37](#_Toc296516057)

[4.3.1 Notify Listener 37](#_Toc296516058)

[4.4 ISBM Consumer Publication Channel Services 38](#_Toc296516059)

[4.4.1 Subscribe Publication Channel 38](#_Toc296516060)

[4.4.2 Read Publication 38](#_Toc296516061)

[4.4.3 Remove Publication 38](#_Toc296516062)

[4.4.4 Unsubscribe Publication Channel 39](#_Toc296516063)

[4.5 ISBM Consumer Request Channel Services 39](#_Toc296516064)

[4.5.1 Open Request Channel 39](#_Toc296516065)

[4.5.2 Subscribe Response Channel 39](#_Toc296516066)

[4.5.3 Post Request 40](#_Toc296516067)

[4.5.4 Read Response 40](#_Toc296516068)

[4.5.5 Remove Response 40](#_Toc296516069)

[4.5.6 Unsubscribe Response Channel 40](#_Toc296516070)

[4.5.7 Close Request Channel 41](#_Toc296516071)

[4.6 ISBM Provider Publication Channel Services 41](#_Toc296516072)

[4.6.1 Open Publication Channel 41](#_Toc296516073)

[4.6.2 Post Publication 41](#_Toc296516074)

[4.6.3 Close Publication Channel 42](#_Toc296516075)

[4.7 ISBM Provider Request Channel Services 42](#_Toc296516076)

[4.7.1 Subscribe Request Channel 42](#_Toc296516077)

[4.7.2 Read Request 42](#_Toc296516078)

[4.7.3 Remove Request 42](#_Toc296516079)

[4.7.4 Open Response Channel 43](#_Toc296516080)

[4.7.5 Post Response 43](#_Toc296516081)

[4.7.6 Close Response Channel 43](#_Toc296516082)

[4.7.7 Unsubscribe Request Channel 44](#_Toc296516083)

[4.8 ISBM WSDL Interface Specifications 44](#_Toc296516084)

[5 Browsing ISBM Service Providers (Optional) 45](#_Toc296516085)

[5.1 Browse Channels, Applications and Topics 45](#_Toc296516086)

[5.2 ISBM Information Verbs 45](#_Toc296516087)

[Annex A: The 7 Layers of the OSI Model 48](#_Toc296516088)

[Annex B: Enterprise Service Buses 49](#_Toc296516089)

# References

* ANSI/ISA 95.00.01, Enterprise-Control System Integration – Part 1: Models and Terminology
* ANSI/ISA 95.00.02, Enterprise-Control System Integration – Part 2: Object Model Attributes
* ANSI/ISA 95.00.05, Enterprise-Control System Integration – Part 5: Business to Manufacturing Transactions
* IEC 62264-1, Enterprise-Control System Integration – Part 1: Models and Terminology
* IEC 62264-2, Enterprise-Control Systems Integration – Part 2: Object Model Attributes
* IEC 62264-5, Enterprise-Control Systems Integration – Part 5: Business to Manufacturing Transactions
* IEC 62541, OPC Unified Architecture, Parts 1-12, [www.opcfoundation.org](http://www.opcfoundation.org)
* WBF B2MML Schemas, [www.wbf.org](http://www.wbf.org), V0400 and later schemas
* MIMOSA Open Systems Architecture for Enterprise Application Integration (OSA-EAI) [www.mimosa.org](http://www.mimosa.org)
* OpenO&M Common Interoperability Registry (CIR), [www.mimosa.org](http://www.mimosa.org)
* ISO 15926, [www.iso.org](http://www.iso.org)
* [X509] X.509 Public Key Certificate Infrastructure, <http://www.itu.int/rec/T-REC-X.509-200003-I/e>
* [IS Glossary] Internet Security Glossary, <http://www.ietf.org/rfc/rfc2828.txt>
* [NIST 800-12] Introduction to Computer Security, <http://csrc.nist.gov/publications/nistpubs/800-12/>
* ISA 14750, Information Technology – Open Distributed Processing – Interface Definition Language

# OpenO&M Information Service Bus Model

## Interface Model

The ISBM defines a standard set of services that would be provided by an application or network service. The services provide a method for multiple applications to communicate using the transaction models defined in the ANSI/ISA 95.05 and IEC 62264-5 standards. The ISBM:

* specifies the definition of services but does not define how the services are implemented
* specifies a general architecture for an ISBM implementation but does not define the architecture of any supporting applications or network services
* specifies the underlying external communication method but does not define any specific underlying internal communication methods

Multiple different implementations are envisioned. The network service will have to include some method for storage or caching of exchanged information, and must guarantee message delivery. However, the ISBM interface is designed to be independent of the underlying message transfer mechanism.

The ISBM essentially provides a standard interface to an ESB (Enterprise Service Bus) system[[1]](#footnote-1) or to any other message or file exchange system that offers guaranteed message and storage or caching of exchanged messages.



Figure 1 - ISBM Interface to ESB and Other Service Providers

Certain services are not defined by this specification, for example, quality of service, message validation, and transformation capability, but can be provided by the ISBM Service Provider to offer differentiation between suppliers and solutions.

## Application to Application Data Exchange

Application to application data exchange is represented in the OSI communication model as a single “Application” layer. However, with the development of data object standards and data representation messages (such as CIR, B2MML, MIMOSA CCOM-ML, ISO 15926, OPC UA address space Objects, and OAGIS Nouns), a simple single layer is insufficient to describe the complexity of object based, loosely coupled application-to-application transactional communication.

Three sublayers can be defined for the application layer for application-to-application communication: a data object layer, a transaction layer, and an exchange service layer, as shown in Figure 2. ISBM is a minimal interface subset that can reside on any exchange service layer and that is based on well-defined and structured data objects and transaction messages.



Figure 2 – Loosely Coupled Application Communication Stack

Each of these “Application” sublayers address a specific element of application data exchange, as shown in Figure 3:

1. The Data Object layer defines the meaning, format, and structure of the basic elements of exchanged information. This layer contains application space specific definitions, such as the ISA 95.02 object definitions, WBF B2MML, MIMOSA CCOM objects, OpenO&M CIR objects, ISO 15926 objects, OPC UA address space objects, and “Nouns” defined in OAGIS.
2. The Transaction layer defines the meaning, format, and structure of actions to be taken on the data objects. For the ISBM, this layer contains IEC 62264-5 transaction style specific definitions.
3. The ISBM Service Interface defines a minimal interface to the Exchange Service Layer.
4. The Exchange Services layer defines the meaning, format, and structure for coordination, buffering, and exchange of messages or files. This layer contains transfer or exchange style specific definitions, such as Enterprise Service Buses, Enterprise Message Delivery Systems, RSS, FTP, or Named Pipes.



Figure 3 – Defined Standards at Each Application Sublevel

The OpenO&M Information Service Bus Model (ISBM) defines a set of transaction services that are suitable for use of exchange of OpenO&M information objects, using IEC 62264-5 transactions. In a sense, ISBM defines the standard “on-ramp” and "off-ramp" to a set of communication services. It defines how data is placed into exchange methods and how it is retrieved from the exchange methods.

## Transaction Model

The ISA 95.05 and IEC 62264-5 standards define three models for business transactions: a publish model, a push model, and a pull model[[2]](#footnote-2).

The ISBM defines a standard interface for applications to exchange data following any of the ISA 95.05 transaction models using OpenO&M XML schemas to represent data.

The transactions supported by the ISBM support:

1. A publish-subscribe model with multiple subscribers and multiple publishers, where the publishers and subscribers have not direct knowledge of other applications.
2. A push and pull model, also called a request-response model, where an application sends unsolicited requests for a service and has no direct knowledge of the receiving application that will process the request.

## Communicating Applications

ISA 95 and IEC 62264 define four roles:

1. Information Provider (to receive GET messages and send SYNC messages)
2. Information Receiver (to receive PROCESS, CHANGE, and CANCEL messages)
3. Information Users (to send GET messages and receive SYNC messages)
4. Information Sender (to send PROCESS, CHANGE, and CANCEL messages).

In the OpenO&M ISBM model these are simplified to Provider Application (Information Provider and Information Receiver) and Consumer Application (Information User and Information Sender), as shown in Figure 4.

An application can be a provider application, consumer application or both. If an application is both, then it should be a consumer of different data than it is provides.



Figure 4 – OpenO&M Information Service Bus Model Names

While the ISA 95.05 verbs (e.g. GET, SYNC and CANCEL) can be used with both transaction model types (i.e. publish-subscribe and request-response), the semantics of the GET verb only aligns with the request-response model.

## Managed Communication Channels

The OpenO&M ISBM is based on the concept of managed communication channels. A “channel” is a software object that represents a specific many-to-many communication conduit between applications. Think of a channel as a channel in a CB radio, some channels are for requests and responses, some channels are for general information distribution. Channels have topics, think of a topic as a conversion topic within a CB channel, you can chose to listen to some topics on the channel but ignore others.

The assumption of the standard is that the ISBM services are provided by a communication application, applications, middleware, or services. The implementation method for the ISBM internal services are not defined here and multiple architectures are possible.

The ISBM provides a definition of the standard interfaces to the services (not how they are implemented).

* A managed communication channel is called an ISBM *Channel*.
* The services provided for each ISBM Channel are the *ISBM Channel Services*.
* An ISBM *Channel Name* is identified using a URI. A URI allows a hierarchy of channel definitions that match a company’s physical or application structures, such as channels identified by plant site or major application suite name.
* An ISBM *Service Provider* is the application or network service that exposes and implements the *ISBM Channel Services*.
* A recommended structure for the *ISBM Channel* hierarchy is defined in this document.

Each *ISBM Channel* supports three general types of information exchange:

1. Publications – Information that may be sent to multiple consumer applications.
2. Requests – Information that may be sent to one or more provider applications.
3. Responses – Information returned from a request to a consumer application.

Each *ISBM Channel* supports two way communications between provider applications and consumer applications.

1. An ISBM *Channel* may be created to support either publication services, request services, or response services.
2. A *Provider Application* may post publications to an ISBM *Publication Channel*.
3. *Consumer Applications* may subscribe to publication notifications (if supported by the specific *ISBM Publication Channel Service*) and may read publications. If notifications are not supported, then the *Consumer Application* may poll the *ISBM Publication Channel* using the read publication service.
4. A *Consumer Application* may post requests to an ISBM *Request Channel*.
5. A *Provider Application* may subscribe to request notifications (if supported by the specific *ISBM Request Channel Service*) and may read requests. If notifications are not supported, then the *Provider Application* may poll the *ISBM Request Channel* using the read request service.
6. *ISBM Channels* have associated *Topics*. Topics are identified when subscribing to a channel, when posting a publication, and when posting a request.

## ISBM Channel Services

The ISBM Channel Services are shown in Figure 5. These services would usually be called used by a provider application, or by a dedicated channel management application.



Figure 5 – ISBM Channel Management Services

The ISBM Channel Management Services are used to create and delete channels and to control the security token specification for channels.

## ISBM Notification Services

The single Notify Listener service allows the ISBM to indicate to a provider or consumer application that a message that meets their read criteria is waiting to be read. It provides an asynchronous callback alternative to the provider/consumer application polling the ISBM. See Section 3.7 for details on message queues and notification.

## ISBM Publication Channel Services

The ISBM Publication Channel Services are shown in Figure 6. The services allow multiple Provider Applications to post publications to the same channel or different channels. Consumer Applications may subscribe to callback notifications of a new message in their session queue or can poll to read publications. Topics provide a subscription filtering mechanism.



Figure 6 – ISBM Publication Channel Services

A publish-subscribe scenario with a single provider application, notification services available, and a consumer application able to use notification services is shown in Figure 7. (Note, there will usually be multiple consumer applications receiving publications, but only one is shown in this example for simplicity.)

In this scenario, the provider application opens an ISBM publication channel with a channel URI and security token. When the provider application has determined that data should be published it posts publications with a message topic.

A consumer application subscribes to the ISBM publication channel using a channel URI, security token, and list of topics. When a new message with the right topic is posted, the consumer application is notified of the posting and then reads the new publication message from the ISBM channel. When the consumer application no longer needs data, or is exiting, it unsubscribes from the ISBM channel.

NOTE: See Section 3 Methods of Operation of ISBM Channels for a recommendation of the structure of channels for a more robust actual implementation.



Figure 7 – Publish-subscribe scenario with notification

A publish-subscribe scenario with a single provider application, where notification services are notavailable orthe consumer application is notable to use notification services is shown in Figure 8. In this scenario, there is no change for the actions of the provider application as in the previous scenario.

In this scenario the consumer application would poll the ISBM channel for publications either periodically or based on some local event. The returned information from the Read Publication indicates if a new publication was returned.



Figure 8 – Publish-subscribe scenario without notification

More than one provider application may use the same publication channel. The scenario shown in Figure 9 has two provider applications. For example, one application could publish changes to Material Definitions while another may publish changes to Material Lots.



Figure 9 – Publish-subscribe scenario with multiple provider applications

## ISBM Request and Response Channel Services

The ISBM Request-Response Channel Services are shown in Figure 10. The services allow one or more Consumer Applications to post requests to Provider Applications, allow one or more Provider Applications to read requests and post responses, and for the Consumer Application to read the response. Topics allow Provider Applications to determine if it should process the request and post a response to the requestor.



Figure 10 – Services for Request/Response

A request-response scenario with a single provider application, notification services available, and a consumer application able to use notification services is shown in Figure 11.



Figure 11 – Request-Response scenario with notification

In this scenario, the provider application subscribes to the request channel. A consumer application opens the request channel and posts a request. The provider is notified and reads the request. The provider application performs its appropriate function (in this case to get data) and sends the response message. The consumer application is notified of the posting and reads the request.

If the applications or ISBM services do not support notification, then the provider and consumer applications may poll for a request or response. Figure 12 illustrates a request-response scenario where the consumer application must poll for a response.



Figure 12 – Request-Response scenario without notification

Figure 13 illustrates a scenario with multiple provider applications. In this case two provider applications have subscribed to requests on the same ISBM channel. The consumer application posts a request with a specific topic (such as Personnel Information).

Provider Application 1 is notified of a request that matches a topic that it subscribed to. Provider Application 1 reads the message and generates a response. Provider Application 2 is not notified of the request, because the topic does not match a subscribed topic.

In this scenario, the consumer application is not able to handle notifications, so it polls the ISBM services for a response message.



Figure 13 – Request-response scenario with multiple provider applications

Note: A full system should not have multiple providers for the same topic on the same request channel. If this occurs then there is a possibility of an indeterminate number of response messages that would be returned to the consumer application. This consideration requires careful design of a system of applications to remove dual responsibility for request topic provider applications.

# Methods of Operation of ISBM Channels

There is no restriction on the use of Channels and Topics. There are two main elements that should be used for channels and topics, scope of information and type of information.

This section defines a recommended ISBM method for identifying channels IDs and topics that can be used in order to ensure maximum interoperability.

Channel names should be defined as a name hierarchy determined by the company or the application suites. Channel names should follow the syntax:

\<ISBM root>\<channel scope>\<information scope>\<channel use>

For example:

\AJAXEnterprises\Company\Material\Checkpoint

\AJAXEnterprises\Company\Material\Request

\SystemTest\Final\OurMaterialManager\Inventory\Changes

\AJAXEnterprises\France\Personnel\Checkpoint

## ISBM Root

The ISBM Root is the root of a hierarchy defined when the ISBM services are installed or initialized. Depending on the ISBM Service implementation there may be one or more roots allowed. The ISBM is used to define the top level of the channel hierarchy when browsing the hierarchy.

The ISBM Service Provider may require specific values for ISBM Root.

For example:

1. AN ISBM root may be the name of the company.

Such as: “*AJAX*” or “*AJAXEnterprises\SpecialToolCo*”.

1. AN ISBM root may be a related set of services, with sets for testing, deployment, and operations.

Such as: “*SystemTest\Beta*”, “*SystemTest\Final*”, “*SpecialToolCo\Operations*”.

NOTE: The “<ISBM root>\ISBM\Request”, “<ISBM root>\ISBM\Checkpoint”, and “<ISBM root>\ISBM\Changes” channels are reserved for browsing applications channels and topics. See Section 5.

## Channel Scope

The channel scope contains a hierarchy that may correspond to a physical, geographical, or logical grouping determined by the enterprise, application or project. It may be used to limit the scope of the exchanged information, such as information only exchanged within a one division of a company. The hierarchy may include site, region, division, area, software system or any other enterprise defined element.

For example:

1. A channel scope may include a site or region name to limit the number of distributed messages, such as: “*AsiaPacific*”, “*SouthAfrica*”, or “*France*”.
2. A channel scope may be a software system, because the information is provided by a well-known system name, such as “*OurMaterialManager*”, “*PersonnelTracker*”, “*InventoryDatabase*”.
3. A channel scope may be companywide because the information is intended for any application in the company. In this case the channel scope should indicate the entire enterprise or company, such as “*Enterprise*” or “*Company*”, or it may be null.

## Information Scope

The information scope defines the range or general type of information exchanged. The information scope may be related to transaction nouns, to other collections of objects, or to business or control processes that deal with a collection of objects.

For example:

1. An application that handles all forms of material information may define a channel with an information scope of “*Material*”.
2. An application that only handles Material Lot and Sublot inventories may define a channel with an information scope of “*Inventory*”.

## Channel Use

The channel use qualifies the information scope to indicate how the information is being used. The channel use may be related to transaction verbs or other business or control process that deal with how the information on the channel is to be used.

For example:

1. An application that sends material requests may define a channel with a channel use of “*Request*”.
2. An application that indicates changes handles Material Lot and Sublot inventories may define a channel with a channel use of “*Changes*”.

## Topics

Topics are used in application services to limit or filter the type of information that is obtained from read and notify requests for Provider Applications and Consumer Applications.

Topics are also used by Provider Applications to specify the type of information that they will be publishing or posting on an ISBM *Channel*.

Topics allow a single channel to handle a collection of different data, yet still provide a method for the receiver of the data to limit the types of data that it is required to handle.

## Standard Topics

To support interoperability, *topics* will be defined as XPath V1.0 expressions, with a namespace prefix which was previously registered when the topic was defined to the ISBM Provider Application. The standardized namespace prefixes are:

* B2MML
* CCOM
* CIR
* ISO15926
* OPCUA

The same topic may be defined on multiple channels. For example:

1. There may be a *ProductionSchedule* topic defined for *CheckPoint* and *Changes* channels with a site channel scope, and a *ProductionSchedule* topic defined for *Checkpoint* and *Changes* channels for an area channel scope.
2. There may be a *QualificationTest* topic defined for a *Request* channel at the enterprise channel scope, and a *QualificationTest* topic defined for a *Request* channel at the country channel scope.

## Message Queuing

The ISBM must maintain a queue for each channel session for a consumer application (i.e. subscriber sessions, request sessions, and response sessions). The queue stores messages routed to the consuming application, from which the application will read and delete the top most message. The message order in the queue is determined by when a provider application *invokes* the call to put the message on the ISBM (and not by the completion of the call). In a notification scenario, the ISBM must delay notifying any consumer applications of the pending message until notifications of earlier invoked messages have been sent.

In the below example, Provider Application 2 starts the PostPublication call before and Provider Application 1 starts its PostPublication call. While Provider Application 1 completes the call first (due to a smaller message size), the Consumer Application will not receive a notification (or in a polling scenario, will not be able to read the message) until Provider Application 2’s PostPublication call is complete. At this point, notifications are sent consecutively (in order of invocation) to the Consumer Application.



Figure – Message Queuing and Delayed Notification

All messages in a channel session queue can be safely discarded once the consumer application has closed the session (i.e. unsubscribe publication, request or response channels).

## Security

Security in the ISBM services is of paramount importance. When using the ISBM, the communication applications have no knowledge of their communication partners, and do not know if there are none (for a publisher with no subscriptions), one, or many. Therefore, security cannot be defined as communication with trusted partners, instead security is defined as communication through secure channels.

### Security Tokens on Channels

Security is managed through security tokens. Security tokens are assigned to channels by the provider applications. The security tokens are used by applications when opening or subscribing to a channel. If the application provided security token does not match a security token assigned to the channel, then no channel information is returned.

Security tokens are exchanged in an out-of-band communication channel, such as manual exchange of tokens, or electronic exchange through a secure point-to-point channel.



Figure 15 – Security of Channels

### Security Token Format

Security tokens must follow the WS-Security standard. There are a large number of ways to validate a user using WS-Security. This specification defines three formats for security tokens.

1. Username/Password
2. PKI through X.509 Certificates
3. Kerberos

*ISBM Service Provides* may provide additional formats for security tokens. In this case the *ISBM Service Provider* must supply or make available an appropriate Security Token Service to create and acquire security tokens.

The security method is based on the availability of a Security Token Service that can return tokens based upon a request, as shown in Figure 15. The interface to the Security Token Service is not defined as part of this specification.

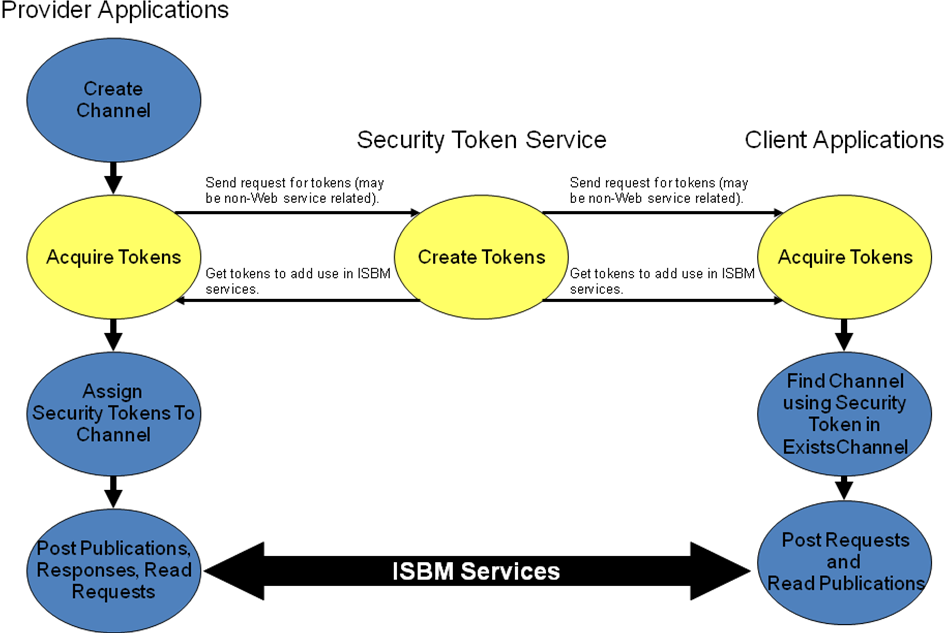


Figure 16 - Security Token Service

Tokens are XML documents that follow the WS-Security definition for a SOAP Header element to carry security-related data.

The specific security token specification followed by ISBM is defined in:

<http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf>

The XML schema that defines the format for token representation is:

<http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd>

This specification defines the <wsse:Security> header as a mechanism for conveying security information with and about a SOAP message. This header is, by design, extensible to support many types of security information.

For security tokens based on XML, the extensibility of the <wsse:Security> header allows for these security tokens to be directly inserted into the header.

#### Username/Password

A common way to identify security is through the use of a username and password combination. WS-Security has defined the UsernameToken element to pass user credentials in this manner. The schema definition for this element is:

<xs:element name="UsernameToken">

<xs:complexType>

<xs:sequence>

<xs:element ref="Username"/>

<xs:element ref="Password" minOccurs="0"/>

</xs:sequence>

<xs:attribute name="Id" type="xs:ID"/>

<xs:anyAttribute namespace="##other"/>

</xs:complexType>

</xs:element>

This schema element references the Username and Password types. These two types are strings that contain extra attributes as needed.

The Password element contains an attribute named Type that indicates how the password is being passed around. A password can be passed as plain text or in digest format. An example UsernameToken with a clear text password is:

<UsernameToken>

<Username>Bob Smith</Username>

<Password Type="PasswordText">hardtobreak</Password>

</UsernameToken>

An example UsernameToken with a password that is encrypted is:

<UsernameToken>

<Username>Bob Smith</Username>

<Password Type="PasswordDigest">

KE6QugOpkPyT3Eo0SEgT30W4Keg=

</Password>

<Nonce>5uW4ABku/m6/S5rnE+L7vg==</Nonce>

<Created xmlns:wsu="http://schemas.xmlsoap.org/ws/2002/07/utility">

2002-08-19T00:44:02Z

</Created>

</UsernameToken>

#### PKI Through X.509 Certificates

Security tokens may specify an X.509 certificate. is an ITU-T standard for a public key infrastructure (PKI) for single sign-on (SSO) and Privilege Management Infrastructure (PMI).

When a message sends an X.509 certificate, it passes the public version of the certificate in a WS-Security token named BinarySecurityToken. The certificate is sent as base64 encoded data. The BinarySecurityToken has the following schema:

<xs:element name="BinarySecurityToken">

<xs:complexType>

<xs:simpleContent>

<xs:extension base="xs:string">

<xs:attribute name="Id" type="xs:ID"/>

<xs:attribute name="ValueType" type="xs:QName"/>

<xs:attribute name="EncodingType" type="xs:QName"/>

<xs:anyAttribute namespace="##other"

processContents="strict"/>

</xs:extension>

</xs:simpleContent>

</xs:complexType>

</xs:element>

#### Kerberos

Kerberos is a computer network authentication protocol, which allows nodes communicating over a non-secure network to prove their identity to one another in a secure manner. It is also a suite of free software published by Massachusetts Institute of Technology (MIT) that implements this protocol. Its designers aimed primarily at a client-server model, and it provides mutual authentication — both the user and the server verify each other's identity.

The Kerberos specification used in ISBM is defined in:

<http://www.ws-i.org/Profiles/KerberosTokenProfile-1.0.html>

Message layer security with the Kerberos protocol in WSE 3.0 involves the following participants:

**Client**: The client accesses the Web service. The client provides the credentials for authentication during the request to the Web service.

**Service:** The service is the Web service that requires authentication of a client prior to authorizing the client.

**Key Distribution Center (KDC)**: The KDC is the broker that authenticates clients and issues service tickets.

The main steps in the client side of a Kerberos system is:

1. Request a service ticket from the KDC.
2. Retrieve the service ticket from the KDC.
3. Send the message to the service using the service ticket as the security token.

The service authenticates the client using information found in the security token. The main service side steps are:

1. Validate the token.
2. Verify the XML signature.
3. Perform the specified service.
4. Initialize and send a response to the client (optional).

A Kerberos token may contain the schema used to validate the token, must defined a value type of the token (as defined by the KDC), the encoding type of the token, and the token. Some examples of Kerberos tokens are:

<wsse:BinarySecurityToken

xmlns:wsse="http://schemas.xmlsoap.org/ws/2002/xx/secext"

wsu:Id="myToken"

ValueType="wsse:Kerberosv5ST"

EncodingType="wsse:Base64Binary">

MIIEZzCCA9CgAwIBAgIQEmtJZc0...

</wsse:BinarySecurityToken>

<wsse:BinarySecurityToken

wsu:Id="myKerberosToken"

ValueType="http://docs.oasis-open.org/wss/2005/xx/oasis-2005xx-wss-kerberos-token-profile-1.1#GSS\_Kerberosv5\_AP\_REQ"

EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0#Base64Binary">

YIIEZzCCA9CgAwIBAgIQEmtJZc0...

</wsse:BinarySecurityToken>

### ISBM Service Provider Implementations

1. All *ISBM Service Providers* **must** implement security tokens.
2. The form, format, and out-of-band token exchange **must** be defined by the *ISBM Service Provider*.
3. *ISBM Service Providers* may choose to limit the ability to use the ISBM Channel Management services to approved applications, servers, or domains in order to increase security.
4. Provider applications may chose not to apply security tokens to channels. While there is a requirement that the services provide security services, there is no requirement that a specific implementation use the services.

For example:

* + A system may share information across companies through open Internet channels. In this case an ISBM *Service Provider* implementation should provide a strong security token system through a public key mechanism with specific security token assigned to specific communicating companies.
  + A system may be entirely contained within a secure environment behind both corporate and operations firewalls. In this case the user may decide to not assign security tokens to channels.

### ISBM Application Implementation Considerations

AN ISBM application implementation should take the following concerns and issues into account:

1. Security tokens will usually be stored in the provider and client applications so they can be used on startup or restart of the application. The tokens should be saved in a secure manner to prevent unauthorized discovery of the tokens.
2. In high security environments there may be a unique security token assigned for each possible communication path and security tokens may be changed on a regular basis, so mechanism should be in setup to tokens on a regular basis.
3. The ISBM services will not validate XML messages. The receiving applications should validate any received messages against the agreed to schema sets, such as B2MML.

### ISBM Channel Security Considerations

Some implementations may require additional levels of security than defined in Clause 3.4. For example an implementation may require separate security for GET/SHOW messages than for PROCESS, CHANGE, CANCEL messages. Separate request channels may be setup for the query (GET/SHOW) and process (PROCESS, CHANGE, CANCEL) messages.

## ISBM Service Provider Considerations

The following sections define ESB type services that **can** be provided by ISBM *Service Providers*. The services are **not** part of the ISBM specification, but provide some of the areas in which vendors and others can provide differentiated service.

### Security Considerations

AN ISBM *Service Provider* should take the following concerns and issues into account:

1. The *ISBM Service Provider* may store messages in a persistent data store. If this is the case and there is security on the channel, then the stored messages may need to be encrypted to prevent unauthorized access to the stored messages.
2. Requests for access with invalid security tokens should be logged. They either indicate a problem with configuration information or a possible attack of the system.
3. Messages exchanged within the ISBM Service implementation may require encryption or connection through secure channels. The method used may be dependent on the transport services used and is not defined in the ISBM interface.
4. Channel Session IDs should be globally unique and use restricted to a specific provider or consumer in order to prevent access to a channel without going through token security.

### Notification

*ISBM Service Providers* are encouraged to implement notification capability utilizing the provided notification service. This specification also allows light weight *ISBM Service Provider* implementations, where polling is an acceptable method for synchronization of applications.

### Data Format Validation

*ISBM Service Providers* could provide data format validation services for messages. If the message are to follow a predefined and well specified format, such as B2MML or BatchML, then the service provider could provide a service to check the syntax correctness of posted messages. This would provide a governance check on messages. This could be implemented by the ISBM Service Provider maintaining a map between topic namespaces and XML Schema files.

### Allowed Application Checking

*ISBM Service Providers* could provide a governance check that applications creating and subscribing to channels are allowed applications. This check would provide an additional level of security, which may be important if the ISBM Services go outside the company.

### Data Exchange Logging

*ISBM Service Providers* could provide services to log all or selected messages for purposes of governance, compliance, and auditing. Because all messages are in an XML format, and the posting application is know, this could provide an audit or error tracing log that captures all in-band communications.

### Common Error Handling

*ISBM Service Providers* could provide services for a consistent method for handling errors detected by provider and consumer applications. An error handling service, provided as a dedicated channel, could be used to determine the response to the error. Depending on the error, such as; invalid message received, lost message, incorrect data in message, or failure in ISBM services, the error handling service could notify the appropriate person or entity with responsibility.

### Data Transformation Services

*ISBM Service Providers* could provide transformation services for messages. Typically this would be from a provider or consumer application specific format into a common format (such as B2MML or BatchML), and from a standard format to an application specific format.

A possible method to handle the transformation interfaces is through topics. Topics may be defined that match the application specific format for the messages. The *ISBM Service Provider* could provide a method for associating a topic to a transformation mapping. When a message is received with a transformation topic, then the *ISBM Service Provider* would transform the message to a standard format. When a read request is received with a transformation topic, then the *ISBM Service Provider* would transform the standard format into the application specific topic format.

The *ISBM Service Provider* would maintain the relationship between the application specific topics, the transformation rules to a standard, and a “standard” topic definition. There are no *ISBM Channel Services* for transformation. The assumption is that the transformation is not handled by the applications, and that creating and maintaining the transformation rules and associations is handled by the *ISBM Service Provider*.



Figure 17 – Transformation Services with the ISBM Service Provider

### Cross Company Bridge

*ISBM Service Providers* could provide cross company communication and authentication services for messages.

A method to provide chain of custody for published messages is shown in Figure 17. In this scenario a proxy application (or part of the ISBM) in Company A’s environment would listen for publications from the ISBM. The proxy would forward the publications using a authenticated or secure method to a proxy application in Company B’s environment. The receiving proxy would publish the message in Company B’s ISBM environment. The bridge may also convert Channel and Topics from Company A’s namespace to Company B’s namespace.

The specification of secure or authenticated communication channels is outside the scope of this specification.



Figure 18 – Cross Company Bridge between multiple ISBMs

# Service Definitions

This section defines the detailed format for the *ISBM Service* definitions.

## Type Definitions

### Success and Error Criteria

0 = Success

1 = Error

### ISBM Channel Type

1 = Publication Channel Type

2 = Request Channel Type

3 = Response Channel Type

### Channel Name

The Channel name is a string allowing channel names with international character sets.

### Channel ID

The channel ID is a string allowing IDs to be encrypted or made non-obvious and not easily guessable.

### Channel Session ID

The Channel Session ID is a string used to identify and manage individual channel consumer/provider sessions and corresponding session message queues.

### Topic ID

The topic ID is a string, with a single topic allowed for a message, but multiple topics allowed for a channel.

## ISBM Channel Management Services

### Assign Security Token

|  |  |
| --- | --- |
| **Name** | Assign Security Token |
| **Description** | Assigns an additional security token to a channel for a valid user for a specific Token Class (Manager, Consumer, or Provider).  Return an error if the channel ID does not match an existing channel. If the provided manager security token is not valid, then an error is returned, otherwise the new security token is added to the channel. |
| **Input Parameters** | * Channel ID * Manager-level Security token * Token class (Manager, Consumer, Provider) * Additional Valid User Security token |
| **Returns** | * Success or error criteria * Status message |

### Create Channel

|  |  |
| --- | --- |
| **Name** | Create Channel |
| **Description** | Creates a new ISBM channel if authorized by the ISBM system based on the submitted manager security token.  If the channel name does not exist, then the channel is created.  If the channel name already exists, then an error is returned. |
| **Input Parameters** | * Channel Name * Channel Type (Publication, Request, or Response) * Manager-level Security token |
| **Returns** | * Channel ID * Success or error criteria * Status message |

### Create Channel Namespace

|  |  |
| --- | --- |
| **Name** | Create Channel Namespace |
| **Description** | Creates a new namespace prefix assignment for a channel that can be referenced when creating topic XPath definitions  If the namespace prefix does not exist, then the namespace prefix is created.  If the namespace prefix is already defined for the channel, then an error is returned. |
| **Input Parameters** | * Channel ID * Namespace Prefix * Namespace URI |
| **Returns** | * Success or error criteria * Status message |

### Create Topic

|  |  |
| --- | --- |
| **Name** | Create a topic on a channel if authorized by the ISBM system based on the submitted manager security token |
| **Description** | Creates a new topic on a channel  If the topic ID does not exist on the channel, then the topic is created.  If the topic ID already exists on the channel, then an error is returned.  If the topic XPath definition is specified and the prefix of a namespace has not been created for this channel, then an error is returned. |
| **Input Parameters** | * Channel ID * Topic ID * Topic Description * Topic Xpath Definition (XPath V1.0) * Content Validation Schema URI * Manager-level Security token |
| **Returns** | * Success or error criteria * Status message |

### Delete Channel

|  |  |
| --- | --- |
| **Name** | Delete Channel |
| **Description** | Deletes a channel if authorized by the ISBM system based on the submitted manager security token. All topics and sessions for this channel will be deleted also.  Returns an error if the channel ID does not match an existing ISBM channel or security token is invalid.  Any cached or stored messages are no longer available. |
| **Input Parameters** | * Channel ID * Manager-level Security token |
| **Returns** | * Success or error criteria * Status message |

### Delete Channel Namespace

|  |  |
| --- | --- |
| **Name** | Delete Channel Namespace |
| **Description** | Deletes a namespace prefix assignment for a channel  If the namespace prefix does not exist, then an error is returned.  If the namespace prefix is in use in an XPath definition of a topic for this channel, then an error is returned. These topics must first be deleted before this namespace prefix can be deleted. |
| **Input Parameters** | * Channel ID * Prefix of Namespace |
| **Returns** | * Success or error criteria * Status message |

### Delete Topic

|  |  |
| --- | --- |
| **Name** | Delete Topic if authorized by the ISBM system based on the submitted manager security token |
| **Description** | Deletes a topic on a channel.  Returns an error if the channel ID and/or topic ID does not match an existing ISBM channel ID or topic ID.  Any cached or stored messages for the topic are no longer available. |
| **Input Parameters** | * Channel ID * Topic ID * Manager-level Security token |
| **Returns** | * Success or error criteria * Status message |

### Get All Channels

|  |  |
| --- | --- |
| **Name** | Get All Channels |
| **Description** | Returns a list of all accessible channels based on the security token provided to the ISBM system. |
| **Input Parameters** | * Security token |
| **Returns** | * List of Channel IDs * Success or error criteria * Status message |

### Get All Sessions

|  |  |
| --- | --- |
| **Name** | Get All Sessions on a Channel |
| **Description** | Returns a list of all accessible channel sessions for a channel. |
| **Input Parameters** | * Channel ID |
| **Returns** | * List of Channel Session IDs * Success or error criteria * Status message |

### Get All Topics

|  |  |
| --- | --- |
| **Name** | Get All Topics on a Channel |
| **Description** | Returns a list of all topics on an channel |
| **Input Parameters** | * Channel ID |
| **Returns** | * List of Topic ID’s * Success or error criteria * Status message |

### Get All Users

|  |  |
| --- | --- |
| **Name** | Get All Users Authorized on a Channel |
| **Description** | Returns a list of all authorized users based on the security tokens assigned to the channel. |
| **Input Parameters** | * Channel ID * Manager-level Security token |
| **Returns** | * List of User IDs * Success or error criteria * Status message |

### Get Channel Info

|  |  |
| --- | --- |
| **Name** | Get Channel Information |
| **Description** | Returns information about a channel. |
| **Input Parameters** | * Channel ID |
| **Returns** | * Channel Name * Channel Type (Publication, Request, or Response) * Creator User ID * List of Topic ID’s * Success or error criteria * Status message |

### Get Session Info

|  |  |
| --- | --- |
| **Name** | Get Session Information |
| **Description** | Returns information about a channel session. |
| **Input Parameters** | * Channel Session ID |
| **Returns** | * Channel Session Type (Consumer, Provider) * Creator User ID * List of Topic ID’s (Optional: for consumer publication sessions only) * URI of Publication Listener (Optional: for consumer publication sessions only) |

### Get Topic Info

|  |  |
| --- | --- |
| **Name** | Get Topic Information |
| **Description** | Returns information about a topic on a channel. |
| **Input Parameters** | * Channel ID * Topic ID |
| **Returns** | * Topic Description * Topic XPath Definition (XPath V1.0) * Content Validation Schema URI * Creator User ID * Success or error criteria * Status message |

### Get User Info

|  |  |
| --- | --- |
| **Name** | Get User Information |
| **Description** | Returns information about an authorized user on a channel. |
| **Input Parameters** | * Channel ID * User ID |
| **Returns** | * User Class (Manager, Consumer, Provider) * Creator User ID * Success or error criteria * Status message |

### Remove Security Token

|  |  |
| --- | --- |
| **Name** | Remove Security Token |
| **Description** | Removes a security token from a channel.  Return an error if the channel ID does not match an existing channel. If the provided manager security token is not valid or the user security token is not valid, then an error is returned, otherwise the security token is removed to the channel. |
| **Input Parameters** | * Channel ID * Manager Security token * User Security token to remove |
| **Returns** | * Success or error criteria * Status message |

## ISBM Notification Channel Services

### Notify Listener

|  |  |
| --- | --- |
| **Name** | Notify Listener |
| **Description** | Asynchronous notification of a new message on a channel session based on a previous subscription.  This is a callback function invoked by the notifying application based on the Listener URI provided to the notifying application when the application desiring asynchronous notifications subscribed to a channel. |
| **Input Parameters** | * Channel Session ID * Topic ID * Message ID * Originating Request Message ID (only applicable to Consumer Response Channels) |
| **Returns** | * Success or error criteria * Status message |

## ISBM Consumer Publication Channel Services

### Subscribe Publication Channel

|  |  |
| --- | --- |
| **Name** | Subscribe Publication Channel |
| **Description** | Subscribes to one or more topics on a publication channel and receives a channel session ID for reading subscriptions synchronously or by providing an asynchronous listener URI |
| **Input Parameters** | * Channel ID * User Security token * URI of Publication Listener (optional) * List of Topic IDs |
| **Returns** | * Channel Session ID * Success or error criteria * Status message |

### Read Publication

|  |  |
| --- | --- |
| **Name** | Read Publication |
| **Description** | For synchronous polling consumers, reads the earliest publication message from a publication channel session message queue based on previous publication channel subscription. An application must have previously subscribed to the publication channel and received a channel session ID in order to read publications. NOTE: This service does not remove the message from the session queue (see “Remove Publication”). |
| **Input Parameters** | * Channel Session ID |
| **Returns** | * Topic ID of publication message * Message ID of publication message * Publication message * Success or error criteria * Status message |

### Remove Publication

|  |  |
| --- | --- |
| **Name** | Remove Publication |
| **Description** | Indicates that a publication message has been read successfully by a consumer and that this specific message can be removed from the publication channel session queue. NOTE: Some implementations may require a Read Publication service which returns this specific message be invoked prior to invoking this service. |
| **Input Parameters** | * Channel Session ID * Message ID |
| **Returns** | * Success or error criteria * Status message. |

### Unsubscribe Publication Channel

|  |  |
| --- | --- |
| **Name** | Unsubscribe Publication Channel |
| **Description** | Unsubscribes from the publication channel session to stop all messages on all topics and close the session. |
| **Input Parameters** | * Channel Session ID |
| **Returns** | * Success or error criteria * Status message. |

## ISBM Consumer Request Channel Services

### Open Request Channel

|  |  |
| --- | --- |
| **Name** | Open Request Channel |
| **Description** | Opens a request channel for posting of request messages.  If the channel ID does not exist, or the security token does not match the channel’s security token, then no channel session ID is returned. |
| **Input Parameters** | * Channel ID * User Security Token |
| **Returns** | * Channel Session ID * Message ID of the last posted request message, if available * Success or error criteria * Status message |

### Subscribe Response Channel

|  |  |
| --- | --- |
| **Name** | Subscribe Response Channel |
| **Description** | Subscribes to a response channel and receives a channel session ID for reading responses synchronously or by providing an asynchronous listener URI. |
| **Input Parameters** | * Channel ID * User Security token * URI of Response Listener (optional) * List of Topics ID’s * Message ID of the last read response message from channel, if known |
| **Returns** | * Channel Session ID * Success or error criteria * Status message |

### Post Request

|  |  |
| --- | --- |
| **Name** | Post Request |
| **Description** | Posts a request message to a request channel session |
| **Input Parameters** | * Channel Session ID * Request message * Topic ID |
| **Returns** | * Message ID of the request message * Success or error criteria * Status message |

### Read Response

|  |  |
| --- | --- |
| **Name** | Read Response |
| **Description** | For synchronous polling consumers, reads the earliest response message from a response channel session queue based on previous response channel subscription. An application must have previously subscribed to the response channel and received a channel session ID in order to read responses. NOTE: This service does not remove the message from the session queue (see “Remove Response”). |
| **Input Parameters** | * Channel Session ID |
| **Returns** | * Message ID of originating request * Topic ID of response message * Message ID of response message * Response message * Success or error criteria * Status message |

### Remove Response

|  |  |
| --- | --- |
| **Name** | Remove Response |
| **Description** | Indicates that a response message has been read successfully by a consumer and that this specific message can be removed from the response channel session queue. NOTE: Some implementations may require a Read Response service which returns this specific message be invoked prior to invoking this service. |
| **Input Parameters** | * Channel Session ID * Message ID |
| **Returns** | * Success or error criteria * Status message. |

### Unsubscribe Response Channel

|  |  |
| --- | --- |
| **Name** | Unsubscribe Response Channel |
| **Description** | Unsubscribes from a response channel session to stop all messages and close the session. |
| **Input Parameters** | * Channel Session ID |
| **Returns** | * Success or error criteria * Status message |

### Close Request Channel

|  |  |
| --- | --- |
| **Name** | Close Request Channel |
| **Description** | Closes a request channel from posting of messages. |
| **Input Parameters** | * Channel Session ID |
| **Returns** | * Success or error criteria * Status message |

## ISBM Provider Publication Channel Services

### Open Publication Channel

|  |  |
| --- | --- |
| **Name** | Open Publication Channel |
| **Description** | Opens the publication channel for the subsequent posting of publication messages and returns a publication session ID  If the channel ID does not exist or the security token does not match the channel’s security token, then no channel session ID is returned. |
| **Input Parameters** | * Channel ID * User Security Token |
| **Returns** | * Channel Session ID * Message ID of the last posted publication message, if available * Success or error criteria * Status message |

### Post Publication

|  |  |
| --- | --- |
| **Name** | Post Publication |
| **Description** | Posts a publication message to a channel session. |
| **Input Parameters** | * Channel Session ID * Topic ID * Publication message |
| **Returns** | * Message ID of posted message * Success or error criteria * Status message |

### Close Publication Channel

|  |  |
| --- | --- |
| **Name** | Close Publication Channel |
| **Description** | Closes a publication channel session. |
| **Input Parameters** | * Channel Session ID |
| **Returns** | * Success or error criteria * Status message |

## ISBM Provider Request Channel Services

### Subscribe Request Channel

|  |  |
| --- | --- |
| **Name** | Subscribe Request Channel |
| **Description** | Subscribes to a request channel and receives a channel session ID for reading requests synchronously or by providing an asynchronous listener URI. |
| **Input Parameters** | * Channel ID * User Security token * URI of Request Listener (optional) * List of Topic ID’s * Message ID of the last read request message from channel, if known |
| **Returns** | * Channel Session ID * Success or error criteria * Status message |

### Read Request

|  |  |
| --- | --- |
| **Name** | Read Request |
| **Description** | For synchronous polling providers, reads the earliest request message from a request channel session queue based on a previous request channel subscription. An application must have previously subscribed to the request channel and received a channel session ID in order to read requests. NOTE: This service does not remove the message from the session queue (see “Remove Request”). |
| **Input Parameters** | * Channel Session ID |
| **Returns** | * Message ID of request message * Request message * Success or error criteria * Status message |

### Remove Request

|  |  |
| --- | --- |
| **Name** | Remove Request |
| **Description** | Indicates that a request message has been processed successfully by a provider and that this specific message can be removed from the request channel session queue. NOTE: Some implementations may require a Read Request service which returns this specific message be invoked prior to invoking this service. |
| **Input Parameters** | * Channel Session ID * Message ID |
| **Returns** | * Success or error criteria * Status message. |

### Open Response Channel

|  |  |
| --- | --- |
| **Name** | Open Response Channel |
| **Description** | Opens a response channel for posting of response messages.  If the channel ID does not exist, or the security token does not match the channel’s security token, then no channel session ID is returned. |
| **Input Parameters** | * Channel ID * User Security Token |
| **Returns** | * Channel Session ID * Message ID of the last posted request message, if available * Success or error criteria * Status message |

### Post Response

|  |  |
| --- | --- |
| **Name** | Post Response |
| **Description** | Posts a response message to a response channel session. |
| **Input Parameters** | * Channel Session ID * Message ID of originating request * Topic ID of response message * Response message |
| **Returns** | * Message ID of response message * Success or error criteria * Status message |

### Close Response Channel

|  |  |
| --- | --- |
| **Name** | Close Response Channel |
| **Description** | Closes a response channel from posting of response messages. |
| **Input Parameters** | * Channel Session ID |
| **Returns** | * Success or error criteria * Status message |

### Unsubscribe Request Channel

|  |  |
| --- | --- |
| **Name** | Unsubscribe Request Channel |
| **Description** | Unsubscribes from a request channel session to stop all messages and close the session. |
| **Input Parameters** | * Channel Session ID |
| **Returns** | * Success or error criteria * Status message |

1. See Annex B for a brief discussion on ESBs. [↑](#footnote-ref-1)
2. See the ISA 95 standards and WBF B2MML documentation for a complete description of the types, format and structure for transactions. [↑](#footnote-ref-2)