NCI AGENCY

NATO Communications and Information Agency
Agence OTAN d'information et de communication

AGENCY INSTRUCTION

INSTR TECH 06.02.13

SERVICE INTERFACE PROFILE FOR CORE AND ADVANCED INSTANT MESSAGING COLLABORATION SERVICES

Effective date:

Revision No: Original

Issued by: Chief, Core Enterprise Services

Approved by: Director Service Strategy

Table of Amendments

Amendment No	Date issued	Remarks

Author Details

Organization	Name	Contact Email/Phone
NCI Agency	A. Ross	
NCI Agency	M. Laukner	michael.laukner@ncia.nato.int
NCI Agency	L. Schenkels	leon.schenkels@ncia.nato.int

Table of Contents

List of Annexes

AGENCY INSTRUCTION 06.02.13

SERVICE INTERFACE PROFILE FOR CORE AND ADVANCED INSTANT MESSAGING COLLABORATION SERVICES

0 PRELIMINARY INFORMATION

0.1 References

A. NCIA/GM/2012/235; Directive 1 Revision 1; dated 3 May 2013
B. NCIA/RECCEN-4-22852 DIRECTIVE 01.01 Agency Policy on Management and Control of Directives, Notices, Processes, Procedures and Instructions, dated 20 May 2014
C. NCIA/RECCEN-4-23297, Directive 06.00.01, Management and Control of Directives, Processes, Procedures and Instructions on Service Management, dated 03 June 2014

0.2 Purpose

This Technical Instruction (TI) provides detailed information, guidance, instructions, standards and criteria to be used when planning, programming, and designing Agency products and services. In this specific case the TI defines a Service Interface Profile (SIP) for one of NATO's Core Enterprise Services.

TIs are living documents and will be periodically reviewed, updated, and made available to Agency staff as part of the Service Strategy responsibility as Design Authority. Technical content of these instructions is the shared responsibility of SStrat/Service Engineering and Architecture Branch and the Service Line of the discipline involved.

TIs are primarily disseminated electronically[1], and will be announced through Agency Routine Orders. Hard copies or local electronic copies should be checked against the current electronic version prior to use to assure that the latest instructions are used.

0.3 Applicability

This TI applies to all elements of the Agency, in particular to all NCI Agency staff involved in development of IT services or software products. It is the responsibility of all NCI Agency Programme, Service, Product and Project Managers to ensure the implementation of this technical instruction and to incorporate its content into relevant contractual documentation for external suppliers.

1 INTRODUCTION

At a fundamental level all collaboration is a human-based activity. The users can communicate and collaborate using a large variety of synchronous (e.g. instant messaging, white-boarding, voice and video conferences) and asynchronous (e.g. email, portals, shared workspaces) collaboration tools.

Instant messaging, a form of collaboration, supports:

- The capability to initiate synchronous communication instantly
- The capability to discover entities including real-time presence information
- Ad hoc collaboration
- Participation in a number of concurrent sessions
- Chat participation on user devices through thick and thin clients.

[1] https://servicestrategy.nr.ncia/SitePages/Agency%20Directives%20(Technical).aspx

The instant messaging services, specified in this document, are based on the extensible messaging and presence protocol (XMPP). XMPP has been selected as a basis for the Collaboration Service, as it is a mature protocol with widespread adoption and many commercial and open source implementations. XMPP is also gaining growing acceptance in NATO and among the nations. For that reason it is important to standardize its implementations to maintain interoperability in NATO enterprise and federated scenarios.

1.1 Purpose of this Document

This document specifies the Service Interface Profile (SIP) for a number of instant messaging services that can be implemented and used by any XMPP entity (*XMPP Client* or *XMPP Server*) on the XMPP network. These instant messaging services are categorized into core and advanced service interfaces, whereby core relates to a set of service interfaces that are REQUIRED to be implemented by a compliant Collaboration Service and advanced represents a suite of service interfaces that MAY be implemented by one or more XMPP entities on the XMPP network for providing a compliant Collaboration Service.

This document also distinguishes those service interfaces that can be exposed, whereby using XMPP as a data transport, to facilitate exchange of structured messages, such as alerts and incident reports. XMPP offers an extensible mechanism to transfer any type of payload capable of supporting the following functionality:

- Sharing location details amongst XMPP entities
- Discovering and distributing forms
- Signalling alerts for real-time emergency notifications.

Table A.1 lists the core and advanced instant messaging service interface specifications, that are applicable for an *XMPP Client* and an *XMPP Server*.

1.2 Audience

The target audience for this specification is the broad community of NATO Network Enabled Capability (NNEC) stakeholders, who are delivering capability in an NNEC environment, or anticipate that their services may be used in this environment.

These may include (but are not limited to):

- Project Managers procuring Bi-Strategic Command (Bi-SC) or NNEC related systems
- The architects and developers of service consumers and providers
- Coalition partners whose services may need to interact with NNEC services
- Systems integrators delivering systems into the NATO environment.

1.3 Notational Conventions

The following notational conventions apply to this document:

- The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [IETF RFC 2119, 1997].
- Words in *italics* indicate terms referenced in Section 1.4, Terminology.
- `Courier font` indicates syntax derived from the different open standards, such as [IETF RFC 6120, 2011], [IETF RFC 6121, 2011] and [IETF RFC 6122, 2011].

1.4 Terminology

Table 1
Terminology

XMPP Server	Provides basic messaging, presence, and XML routing features.
XMPP Client	An application that enables you to connect to an *XMPP Server* for instant messaging with other users over a network.
Service	A feature or function that can be used by any XMPP application. Note: The term *Service* is used throughout this document to refer to XMPP enabling services.
Roster	A user's contact list that is stored by the user's *XMPP Server*.
XML Stream	A container for the exchange of XML elements between any two XMPP entities over a network.
Stream Feature	The set of XMPP protocol interactions that the initiating XMPP entity has to complete before the receiving XMPP entity will accept *XML Stanzas* from the initiating XMPP entity.
XML Stanza	A first-level XML element (at depth=1 of the stream) which typically contains one or more child XML elements (with accompanying attributes, elements, and XML character data) in order to convey the desired information. There are three kinds of *XML Stanzas: Message, Presence*, and *IQ* (short for "Info/Query").
Message	XML structure to support a "fire-and-forget" mechanism (basic "push" method) for getting information from one place to another.
Presence	Describes the status of an XMPP entity, and their availability for communication over a network.
IQ	Provides a structure for request-response interactions that is optimized for a more reliable exchange of data.
JabberID (JID)	A string, structured as an ordered sequence of localpart, domainpart, and resourcepart (where the first two parts are demarcated by the '@' character used as a separator, and the last two parts are similarly demarcated by the '/' character).
Bare JID	A *JabberID* that is of the form localpart@domainpart.
Full JID	A *JabberID* that is of the form localpart@domainpart/resourcepart.

1.5 Namespaces

The following namespaces, including namespaces referenced in [NCIA TR/2013/SPW008423/36, 2014], are used in this document.

Table 2
Namespaces

Namespace	Reference
http://etherx.jabber.org/streams	[IETF RFC 6120, 2011]
http://jabber.org/features/compress	[XSF XEP-0138, 2009]
http://jabber.org/protocol/address	[XSF XEP-0033, 2004]
http://jabber.org/protocol/amp	[XSF XEP-0079, 2005]
http://jabber.org/protocol/amp#errors	[XSF XEP-0079, 2005]
http://jabber.org/protocol/amp?drop	[XSF XEP-0079, 2005]
http://jabber.org/protocol/amp?error	[XSF XEP-0079, 2005]
http://jabber.org/protocol/amp?notify	[XSF XEP-0079, 2005]
http://jabber.org/protocol/amp condition=expire-at	[XSF XEP-0079, 2005]
http://jabber.org/protocol/compress	[XSF XEP-0138, 2009]
http://jabber.org/protocol/disco#info	[XSF XEP-0030, 2008]
http://jabber.org/protocol/disco#items	[XSF XEP-0030, 2008]
http://jabber.org/protocol/geoloc	[XSF XEP-0080, 2009]
http://jabber.org/protocol/muc	[XSF XEP-0045, 2008]
http://jabber.org/protocol/muc#roominfo	[XSF XEP-0045, 2008]
http://jabber.org/protocol/muc#user	[XSF XEP-0045, 2008]
http://jabber.org/protocol/pubsub	[XSF XEP-0060, 2010]
http://jabber.org/protocol/pubsub#errors	[XSF XEP-0060, 2010]
http://jabber.org/protocol/pubsub#event	[XSF XEP-0060, 2010]
http://jabber.org/protocols/xdata-validate	[XSF XEP-0122, 2004]
http://jabber.org/protocols/xdata-layout	[XSF XEP-0141, 2005]
http://peoc3t.us.army.mil/abcs	(See Chapter 11)
http://tridsys.com/forms	(See Chapter 11)

Namespace	Reference
http://www.incident.com/cap/1.0	[XSF XEP-0127, 2004]
http://www.nato.int/2012/12/nxl/xcl#machine	[NC3A TN-1456 REV-1, 2013]
jabber:client	[IETF RFC 6121, 2011]
jabber:iq:roster	[IETF RFC 6121, 2011]
jabber:iq:search	[XSF XEP-0055, 2009]
jabber:server	[IETF RFC 6121, 2011]
jabber:x:data	[XSF XEP-0004, 2007]
muc_hidden	[XSF XEP-0045, 2008]
muc_membersonly	[XSF XEP-0045, 2008]
muc_moderated	[XSF XEP-0045, 2008]
muc_nonanonymous	[XSF XEP-0045, 2008]
muc_open	[XSF XEP-0045, 2008]
muc_passwordprotected	[XSF XEP-0045, 2008]
muc_persistent	[XSF XEP-0045, 2008]
muc_public	[XSF XEP-0045, 2008]
muc_rooms	[XSF XEP-0045, 2008]
muc_semianonymous	[XSF XEP-0045, 2008]
muc_temporary	[XSF XEP-0045, 2008]
muc_unmoderated	[XSF XEP-0045, 2008]
muc_unsecured	[XSF XEP-0045, 2008]
urn:ietf:params:xml:ns:xmpp-bind	[IETF RFC 6120, 2011]
urn:ietf:params:xml:ns:xmpp-sasl	[IETF RFC 6120, 2011]
urn:ietf:params:xml:ns:xmpp-stanzas	[IETF RFC 6120, 2011]
urn:ietf:params:xml:ns:xmpp-streams	[IETF RFC 6120, 2011]
urn:ietf:params:xml:ns:xmpp-tls	[IETF RFC 6120, 2011]

Namespace	Reference
urn:int:nato:nc3a:xmpp:geowhiteboard	(See Chapter 10)
urn:us:gov:ic:ism:v2	[ODNI IC-ISM, 2008]
urn:xmpp:bidi	[XSF XEP-0288, 2012]
urn:xmpp:delay	[XSF XEP-0203, 2009]
urn:xmpp:features:bidi	[XSF XEP-0288, 2012]
urn:xmpp:ping	[XSF XEP-0199, 2009]
urn:xmpp:sec-label:0	[XSF XEP-0258, 2011]
urn:xmpp:sec-label:catalog:2	[XSF XEP-0258, 2011]
urn:xmpp:sec-label:ess:0	[XSF XEP-0258, 2011]
urn:xmpp:sm:3	[XSF XEP-0198, 2011]
urn:xmpp:time	[XSF XEP-0202, 2009]
vcard-temp	[XSF XEP-0054, 2008]

1.6 Goals

This document consists of a set of information that, along with the clarifications, refinements and interpretations provided, can facilitate interoperability, based on the use of core and advanced instant messaging service interfaces and data structures to be implemented by XMPP entities (*XMPP Clients* and *XMPP Servers*). It also identifies the instant messaging service interfaces that can be leveraged to provide data transport capabilities for exchanging structured data over XMPP.

1.7 Non-goals

The following topics are outside the scope of this profile:

- Recommendations for the use of products or platforms
- Network configuration and parameters required for text-based instant messaging
- Configuration details of a particular server or client implementation.

1.8 Relationships to other Profiles and Specifications

Relationship with other CES Service Interface Profile (SIP) is:

- Enterprise Directory Service (see [NC3A RD-3153, 2011]) – Provides identity attributes used for authenticating XMPP entities to the Instant Messaging Collaboration Service.

All XMPP entities are addressable on the XMPP network. The XMPP based Collaboration Service is reliant on the domain name system (DNS) services to provide the network addressing structure for enabling XMPP entities to discover and communicate with each other over the XMPP network.

1.8.1 Normative References

The following documents, including the documents referenced in [NCIA TR/2013/SPW008423/36, 2014], have fed into this specification, and are incorporated as normative references:

- [IETF RFC 6120, 2011] Extensible Messaging and Presence Protocol (XMPP): Core
- [IETF RFC 6121, 2011] Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence
- [IETF RFC 6122, 2011] Extensible Messaging and Presence Protocol (XMPP): Address Format
- [XSF XEP-0004, 2007] Data Forms
- [XSF XEP-0030, 2008] Service Discovery
- [XSF XEP-0033, 2004] Extended Stanza Addressing
- [XSF XEP-0045, 2008] Multi-User Chat
- [XSF XEP-0048, 2007] Bookmarks
- [XSF XEP-0053, 2008] XMPP Registrar Function
- [XSF XEP-0054, 2008] VCard-Temp
- [XSF XEP-0055, 2009] XMPP Jabber Search
- [XSF XEP-0060, 2010] Publish-Subscribe
- [XSF XEP-0068, 2011] Field Standardization for Data Forms
- [XSF XEP-0079, 2005] Advanced Message Processing
- [XSF XEP-0080, 2009] User Location
- [XSF XEP-0082, 2003] XMPP Date and Time Profiles
- [XSF XEP-0122, 2004] Data Forms Validation
- [XSF XEP-0127, 2004] Common Alerting Protocol (CAP) Over XMPP
- [XSF XEP-0138, 2009] Stream Compression
- [XSF XEP-0141, 2005] Data Forms Layout
- [XSF XEP-0198, 2011] Stream Management
- [XSF XEP-0199, 2009] XMPP Ping
- [XSF XEP-0203, 2009] Delayed Delivery
- [XSF XEP-0202, 2009] Entity Time
- [XSF XEP-0220, 2011] Server Dialback
- [XSF XEP-0256, 2009] Last Activity in Presence
- [XSF XEP-0258, 2011] Security Labels in XMPP
- [XSF XEP-0288, 2012] Bidirectional Server-to-Server Connections.

2 SIP DEFINITION

2.1 Subject

This document focuses on the interface profile of core and advanced instant messaging services, as part of the Collaboration Service. The Collaboration Service is part of the *Interaction* services as defined in [NC3A CES Framework, 2009].

It is impossible to completely guarantee the interoperability of a particular service. However, this SIP aims to increase the level of interoperability based on implementation experience to date.

2.2 Services

XMPP entities can offer a number of *Service* interfaces for the instant messaging services. [XSF XEP-0302, 2011] introduces a core and advanced concept, whereby core and advanced are categories for

XMPP Clients and *XMPP Servers* indicating the listed REQUIRED specifications for compliance purposes for both categories. This document reuses this concept for categorizing XMPP *Services* as core and advanced instant messaging services.

An XMPP entity presenting core and advanced instant messaging service interfaces SHALL support the fundamental features listed below, which are specified in Section 2.3 of [NCIA TR/2013/SPW008423/36, 2014]:

- Global addresses
  - JabberID
  - Domainpart
  - Localpart
  - Resourcepart
- Streaming XML
  - TCP (transmission control protocol) bindings
  - Open XML (extensible markup language) stream
  - Stream negotiation
  - Resource-binding
  - Address determination
  - XML Stanzas
  - Close XML stream
  - Directionality
  - Management of XML streams
  - Error-handling
- Communication primitives
  - Message Stanza
  - Presence Stanza
  - IQ Stanza.

An XMPP entity presenting core and advanced instant messaging service interfaces SHALL support the security mechanisms specified in Section 2.4 of [NCIA TR/2013/SPW008423/36, 2014].

An XMPP entity offering core instant messaging services is responsible for authentication with another XMPP entity (within its own XMPP domain), message submission and delivery to local XMPP entities (including roster management), and maintaining and sharing presence information. Federation across XMPP domains allows XMPP entities to be able to communicate, share presence information and collaborate in a cross-domain sharing environment. The core instant messaging services presented by a compliant XMPP entity are:

1) Presence Service
2) Roster Service
3) One-To-One Messaging Service.

The XMPP architecture is a federated architecture, which allows for XMPP entities to share and host different *Services*. As such, a single XMPP entity providing advanced instant messaging services is NOT REQUIRED to provide all of the advanced instant messaging services. An XMPP entity SHALL present all core instant messaging service interfaces and MAY optionally present one or more *Service* interfaces from the following advanced instant messaging services:

1) XMPP Services Discovery Service
2) Multi-Party Messaging Service
3) Notification Service
4) Labelling Service
5) Whiteboarding Service
6) Structured Data Form Service
7) Time-Sensitive Messaging Service.

Figure 1 illustrates the instant messaging services and their relationship with the fundamental features and security highlighting the components that are mandatory and optional to implement.

XMPP is an extensible architecture and provides for the capability to exchange structured data messages, therefore XMPP can be leveraged as a data transport mechanism. The Multi-Party Messaging *Service* and the Notification *Service* provide an XMPP data transport mechanism for transmitting: structured data forms; user location data; and, signalling alerts.

This document contains a set of operations, messages (input and output) and errors for the *Service* interfaces presented for each instant messaging service listed.

3 PRESENCE SERVICE

3.1 Service Interface

The *Service* interface is not a web service and therefore a web service description language (WSDL) is not appropriate to describe the operations of the service.

The *Service* interface is defined by the IETF XMPP Standard specified in Section 2 of [IETF RFC 6121, 2011].

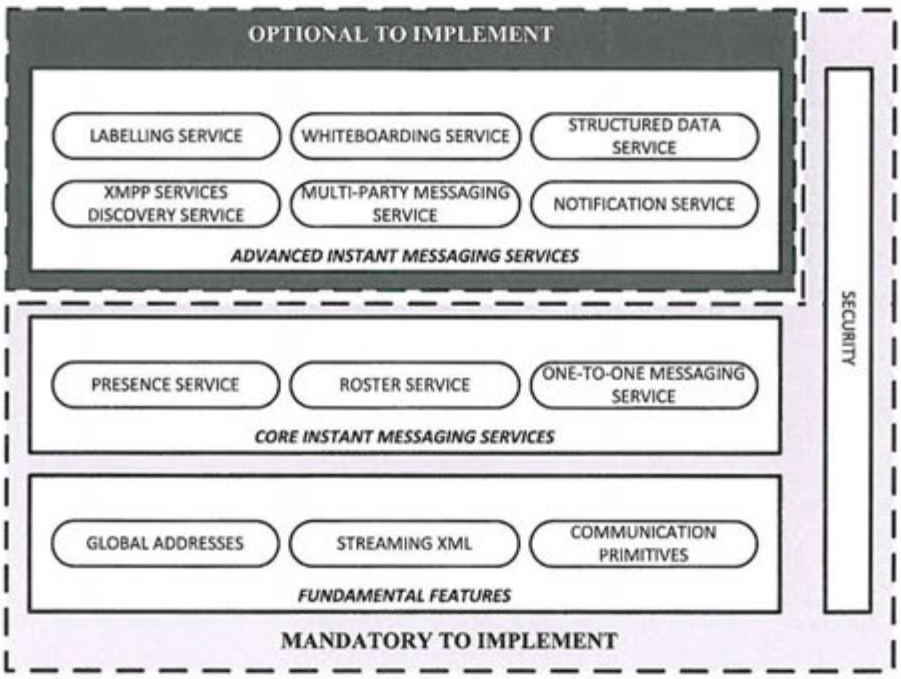

Figure 1 Logical relationship between instant messaging services, fundamental features and security

3.2 Operations

The Presence Service operations follow a:

- "Publish-subscribe" model, whereby an XMPP entity publishes its presence information to an *XMPP Server*, and the *XMPP Server* broadcasts the information to other XMPP entities that have subscribed to the publishing XMPP entity's presence.
- "Request-response" model, whereby XMPP entities can subscribe or unsubscribe from other XMPP entities presence available on the XMPP network.

The Presence Service operations can be categorized in the following groups:

- Subscription management – subscribe, unsubscribe, cancel
- Publish presence – publish
- Broadcast presence – notify.

Table 3 shows the list of operations as specified in [IETF RFC 6121, 2011].

Table 3
List of Presence Service operations

Operation	Reference
`Subscribe`	[IETF RFC 6121, 2011] Section 3.1
`Unsubscribe`	[IETF RFC 6121, 2011] Section 3.3
`Cancel`	[IETF RFC 6121, 2011] Section 3.2
`Publish`	[IETF RFC 6121, 2011] Section 4.2.1, 4.4.1 and 4.5.1
`Notify`	[IETF RFC 6121, 2011] Section 4.2.2, 4.2.3, 4.4.2, 4.4.3, 4.5.2 and 4.5.3

3.2.1 Subscription management operations

3.2.1.1 Subscribe

The Subscribe operation is initiated by an XMPP entity where a *Presence* stanza is sent to the XMPP entity's *XMPP Server* for distribution to the receiving XMPP entity (local or remote).

The '`type`' attribute of the *Presence* Stanza SHALL be set as '`subscribe`'.

The response is initiated by the receiving XMPP entity and is a *Presence* stanza sent to the initiating XMPP entity with a '`type`' attribute that SHALL be set as '`subscribed`'. The response is an approval (or rejection) that the receiving XMPP entity authorizes (denies) the initiating XMPP entity access to its presence information.

An XMPP entity SHALL implement this interface as defined in Section 3.1 of [IETF RFC 6121, 2011].

3.2.1.1.1 Data types

Reference is made to Section 4.7 of [IETF RFC 6121, 2011] for the *Presence* Stanza syntax that SHALL be implemented.

3.2.1.1.2 Input

The input is a *Presence* Stanza specified in Section 3.1 of [IETF RFC 6121, 2011], whereby:

- The '`type`' attribute of the *Presence* Stanza SHALL be set as '`subscribe`'.
- The '`to`' attribute SHALL be the *Bare JID* of the receiving XMPP entity.
- The '`from`' attribute SHALL be the *Bare JID* of the initiating XMPP entity.

3.2.1.1.3 Output

In the case where the receiving entity approves the request, the output is a *Presence* Stanza specified in Section 3.1 of [IETF RFC 6121, 2011], whereby:

- The '`type`' attribute of the *Presence* Stanza SHALL be set as '`subscribed`'.
- The '`to`' attribute SHALL be the *Bare JID* of the initiating XMPP entity.
- The '`from`' attribute SHALL be the *Bare JID* of the receiving XMPP entity.

In the case where the receiving entity denies the request, the output is a *Presence* Stanza specified in Section 3.1 of [IETF RFC 6121, 2011], whereby:

- The '`type`' attribute of the *Presence* Stanza SHALL be set as '`unsubscribed`'.
- The '`to`' attribute SHALL be the *Bare JID* of the initiating XMPP entity.
- The '`from`' attribute SHALL be the *Bare JID* of the receiving XMPP entity.

The semantics for this output to the Subscribe operation SHALL be compliant with Section 3.2 of [IETF RFC 6121, 2011].

3.2.1.2 Unsubscribe

The Unsubscribe operation is initiated by an XMPP entity whereby a *Presence* stanza is sent to the XMPP entity's *XMPP Server* for distribution to the *XMPP Server* managing the XMPP entity's account (local or remote). The response is a *Presence* Stanza reflecting that the unsubscribed XMPP entity is '`unavailable`'.

An XMPP entity SHALL implement this interface as defined in Section 3.3 of [IETF RFC 6121, 2011].

3.2.1.2.1 Data types

Reference is made to Section 4.7 of [IETF RFC 6121, 2011] for the *Presence* Stanza syntax that SHALL be implemented.

3.2.1.2.2 Input

The input is a *Presence* Stanza specified in Section 3.3 of [IETF RFC 6121, 2011], whereby:

- The '`type`' attribute of the *Presence* Stanza SHALL be set as '`unsubscribe`'.
- The '`to`' attribute SHALL be the *Bare JID* of the initiating XMPP entity.
- The '`from`' attribute SHALL be the *Bare JID* of the receiving XMPP entity.

3.2.1.2.3 Output

A *Presence* Stanza specified in Section 3.3 of [IETF RFC 6121, 2011], from each of the 'unsubscribe' XMPP entity's interested resources, whereby:

- The '`type`' attribute of the *Presence* Stanza SHALL be set as '`unavailable`'.
- The '`to`' attribute SHALL be the *Bare JID* of the initiating XMPP entity.
- The '`from`' attribute SHALL be the *Full JID* of the receiving XMPP entity.

3.2.1.3 Cancel

This operation supports the cancelling of a previously approved subscription. This operation is initiated by an XMPP entity whereby a *Presence* stanza is sent to the XMPP entity's *XMPP Server* for distribution to the *XMPP Server* managing the XMPP entity's account (local or remote). The response is a *Presence* Stanza reflecting that the unsubscribed XMPP entity is '`unavailable`'.

An XMPP entity SHALL implement this interface as defined in Section 3.2 of [IETF RFC 6121, 2011].

3.2.1.3.1 Data types

Reference is made to Section 4.7 of [IETF RFC 6121, 2011] for the *Presence* Stanza syntax that SHALL be implemented.

3.2.1.3.2 Input

A *Presence* Stanza specified in Section 3.2 of [IETF RFC 6121, 2011], whereby:

- The '`type`' attribute of the *Presence* Stanza SHALL be set as '*`unsubscribed`*'.
- The '`to`' attribute SHALL be the *Bare JID* of the initiating XMPP entity.
- The '`from`' attribute SHALL be the *Bare JID* of the receiving XMPP entity.

3.2.1.3.3 Output

A *Presence* Stanza specified in Section 3.3 of [IETF RFC 6121, 2011], from each of the 'unsubscribed' XMPP entity's interested resources, whereby:

- The '`type`' attribute of the *Presence* Stanza SHALL be set as '*`unavailable`*'.
- The '`to`' attribute SHALL be the *Bare JID* of the initiating XMPP entity.
- The '`from`' attribute SHALL be the *Full JID* of the receiving XMPP entity.

3.2.2 Publish and Notify operations

The Publish and Notify operations support the updating of an XMPP entity's availability on the XMPP network to all other XMPP entities that have subscribed to that XMPP entity's presence. The XMPP entity publishes its presence information to the *XMPP Server*, and the *XMPP Server* then broadcasts that presence information to the subscribed XMPP entities.

An XMPP entity SHALL implement the Publish interface as defined in Sections 4.2.1, 4.4.1 and 4.5.1 of [IETF RFC 6121, 2011].

An XMPP entity SHALL implement the Notify interface as defined in Sections 4.2.2, 4.2.3, 4.4.2, 4.4.3, 4.5.2 and 4.5.3 of [IETF RFC 6121, 2011].

3.2.2.1 Data types

Reference is made to Section 4.7 of [IETF RFC 6121, 2011] for the *Presence* Stanza syntax that SHALL be implemented.

3.2.2.2 Publish input

The Publish operation inputs can be categorized into three groups:

1) Initial presence (see the specification Section 4.2.1 of [IETF RFC 6121, 2011])
2) Subsequent presence (see the specification Section 4.4.1 of [IETF RFC 6121, 2011])
3) Unavailable presence (see the specification Section 4.5.1 of [IETF RFC 6121, 2011]).

3.2.2.3 Publish output

No output is specified by for this operation.

3.2.2.4 Notify input

The Notify operation inputs can be categorized into three groups:

1) Initial Presence (see the [IETF RFC 6121, 2011] specification Sections 4.2.2 and 4.2.3)
2) Subsequent Presence (see the [IETF RFC 6121, 2011] specification Sections 4.4.2 and 4.4.3)
3) Unavailable Presence (see the [IETF RFC 6121, 2011] specification Sections 4.5.2 and 4.5.3).

For each *Presence* Stanza (categorized above):

- The '`to`' attribute SHALL be the *Bare JID* of the initiating XMPP entity.
- The '`from`' attribute SHALL be the *Full JID* of the receiving XMPP entity.

3.2.2.5 Notify output

No output is specified by for this operation.

3.3 Additional Messages

For each of the subscription management operations there is an additional message that SHOULD be sent from the *XMPP Server* to the XMPP entity that it is hosting, whereby a Push operation, as described in Section 4.3.1.1, is sent to each connected resource for that XMPP entity.

An *XMPP Server* SHOULD be compliant with [IETF RFC 6121, 2011] specification Appendix A, for *Presence* processing related to these Push operation messages.

3.4 Optimized Messages

For the subscription management and notify operations an XMPP entity can be required to send the same *Presence* Stanza to multiple XMPP entities. The number of *Presence* Stanzas sent from an initiating XMPP entity to receiving XMPP entities can be optimized by sending an optimized *Presence* Stanza that includes all the intended receiving XMPP entities.

An XMPP entity MAY support optimized *Presence* Stanzas for the subscription management and notify operations.

An XMPP entity that supports optimized *Presence* Stanzas SHALL be compliant with [XSF XEP-0033, 2004].

3.5 Errors

When errors occur during the processing of *Presence* Stanzas, a *Presence* Stanza of type '`error`' SHALL be sent to the originating XMPP entity of the *Presence* Stanza.

Error types and error conditions that SHALL be supported are specified in Section 8.3.2 of [IETF RFC 6120, 2011]] and Section 8.9.3 of [IETF RFC 6120, 2011], respectively.

4 ROSTER SERVICE

4.1 Service Interface

The *Service* interface is not a web service and therefore a WSDL is not appropriate to describe the operations of the service.

The *Service* interface is defined by the IETF XMPP Standard specified in Section 3 of [IETF RFC 6121, 2011] and Contact Information specified in [XSF XEP-0054, 2008].

4.2 Operations

The Roster Service operations consist of a series of *IQ* Stanza requests and responses that are predominantly client-to-server oriented. The Roster Service operations can be categorized into the following groups:

- Retrieve – Get Roster
- Modification – Add, Update, Delete
- Client Update – Push
- Contact Info – Get, Set.

Table 4 shows the list of operations as specified in [IETF RFC 6121, 2011] and [XSF XEP-0054, 2008].

Table 4
List of Roster Service Operations

Operation	Reference
`Get Roster`	[IETF RFC 6121, 2011] Section 2.2
`Add`	[IETF RFC 6121, 2011] Section 2.3
`Update`	[IETF RFC 6121, 2011] Section 2.4
`Delete`	[IETF RFC 6121, 2011] Section 2.5
`Push`	[IETF RFC 6121, 2011] Section 2.1.6
`Get`	[XSF XEP-0054, 2008] Section 3.1
`Set`	[XSF XEP-0054, 2008] Section 3.1

4.3 Messages

The Roster Service operations are a structured request-response mechanism.

4.3.1 Retrieve, modification and client update requests (input)

All operation requests SHALL be wrapped in *IQ* Stanzas whereby:

- The '`type`' attribute SHALL be '`get`' or '`set`'.
- Containing a `<query/>` child element that SHALL be qualified by a '`jabber:iq:roster`' namespace.

The operation requests SHALL adhere to the detailed syntax specified in Section 2.1.2 of [IETF RFC 6121, 2011].

An operation request of '`type`' attribute '`get`' SHALL be compliant with the detailed semantics specified in Section 2.1.3 of [IETF RFC 6121, 2011].

An operation request of '`type`' attribute '`set`' SHALL be compliant with the detailed semantics specified in Section 2.1.5 of [IETF RFC 6121, 2011].

4.3.1.1 Push operation message request

The execution of the operations categorized in the Modification group, SHOULD result in a Push operation being executed.

A Push operation is the only server-to-client request, whereby any modifications to the roster SHOULD be sent to each available resource that the XMPP entity initiating the modification is bound to.

The semantics for a Push operation SHALL adhere to the specifications detailed in Section 2.1.6 of [IETF RFC 6121, 2011].

4.3.2 Contact info requests (input)

All operation requests SHALL be wrapped in *IQ* Stanzas whereby:

- The 'type' attribute SHALL be 'get' or 'set'.
- Containing a <vcard/> child element that SHALL be qualified by a 'vcard-temp' namespace.

4.3.2.1 Responses (output)

The syntax and semantics of the operation responses depend on the groups that the operations are categorized into.

4.3.2.2 Retrieve response

The retrieve operation response SHALL be wrapped in a *IQ* Stanza, whereby the 'type' attribute SHALL be 'result', and contains a <query/> child element that SHALL be qualified by a 'jabber:iq:roster' namespace.

The retrieve operation response SHALL adhere to the detailed syntax specified in Section 2.1.2 of [IETF RFC 6121, 2011].

The retrieve operation response SHALL be compliant with the detailed semantics specified in Sections 2.1.4 and 2.2 of [IETF RFC 6121, 2011].

4.3.2.3 Modification response

The modification operation response SHALL be wrapped in an *IQ* Stanza, whereby the 'type' attribute SHALL be 'result'.

The modification operation response SHALL be compliant with the specifications detailed in Sections 2.3.2, 2.4.2 and 2.5.2 of [IETF RFC 6121, 2011].

4.3.2.4 Client update response

Section 8.2.3 of [IETF RFC 6120, 2011] specifies that an *IQ* Stanza with 'type' attributes of 'get' and 'set' SHALL be replied to with a an *IQ* Stanza with 'type' attributes of 'result' and 'error'. This SIP recognizes an unnecessary bandwidth usage and a potential security risk (presence leaks[2]) with regards to returning such a response to the server; therefore, it is RECOMMENDED that no client update response SHOULD be sent to the server based on a client update request.

Any client update response with a 'type' attribute of 'error' SHALL be ignored.

---

[2] See Section 13.10.2 of [IETF RFC 6120, 2011].

4.3.2.5 Contact info response

This operation response SHALL be wrapped in a *IQ* Stanza, whereby the 'type' attribute SHALL be 'result', and contains a <vcard/> child element that SHALL be qualified by a 'vcard-temp' namespace.

This operation response SHALL adhere to the syntax and semantics specified in Section 3.1 of [XSF XEP-0054, 2008].

4.4 Errors

An *IQ* Stanza with a 'type' attribute of 'error' SHALL be sent in the case of errors associated with any of the operation requests (see Section 4.3.2.4 for the exception).

The list of errors that SHALL be supported by compliant *XMPP Servers* is presented in Table 5 (for reference purposes only).

A detailed explanation of these error conditions can be found in Section 2 of [IETF RFC 6121, 2011]. Other errors that are listed in Section 8.9.3 of [IETF RFC 6120, 2011] MAY be returned.

Table 5
Mandatory error list to be supported by the roster service

Error condition
Forbidden
Bad-request
Not-acceptable
Item-not-found
Service-unavailable
Internal-server-error

5 ONE-TO-ONE MESSAGING SERVICE

5.1 Service Interface

The *Service* interface is not a web service and therefore a WSDL is not appropriate to describe the operations of the service.

The *Service* interface is defined by the IETF XMPP Standard specified in Section 5 of [IETF RFC 6121, 2011].

5.2 Operations

No operations are specified for this service interface.

5.3 Messages

5.3.1 Data types

Reference is made to Section 5.2 of [IETF RFC 6121, 2011] for the *Message* Stanza syntax that SHOULD be implemented.

5.3.2 Input

Once an XMPP entity has authenticated with an *XMPP Server* and bound a resource to an *XML Stream* as described in Section 2.3.2.4 of [NCIA TR/2013/SPW008423/36, 2014], that XMPP entity can send *Message* Stanzas.

An XMPP entity SHALL be conformant to the *Message* Stanza specification as described in Section 2.3.3.1 of [NCIA TR/2013/SPW008423/36, 2014].

An XMPP entity SHALL support the following '`type`' attribute values for a *Message* Stanza:

- Normal
- Headline
- Chat
- Error.

5.3.3 Output

No output is specified for this service interface.

5.3.4 Optimization

A XMPP entity can be required to send the same *Message* Stanza to multiple XMPP entities. The number of *Message* Stanzas sent from an initiating XMPP entity to receiving XMPP entities can be optimized by sending an optimized *Message* Stanza that includes all the intended receiving XMPP entities.

An XMPP entity MAY support optimized *Message* Stanzas.

An XMPP entity that supports optimized *Message* Stanzas SHALL be compliant with [XSF XEP-0033, 2004].

5.3.5 Time-sensitive messages

An XMPP entity MAY request that the *Message* Stanza is delivered before a certain absolute point in time.

XMPP entities that support time-sensitive messaging SHALL support the Time-Sensitive Messaging Service specified in Chapter 12.

The *Message* Stanza SHALL contain an `<amp/>` element qualified by the '`http://jabber.org/protocol/amp`' namespace with a `<rule/>` child n.

Table 6 details the REQUIRED attributes and values associated with the `<rule/>` element.

Table 6
REQUIRED Rule Attributes and Values

Attribute	Notes
`action`	`alert; drop; error; or notify`
`condition`	`expire-at`
`value`	Time when the Message Stanza expires in UTC

5.4 Errors

When errors occur during the processing of *Message* Stanzas, a *Message* Stanza of type 'error' SHALL be sent to the originating XMPP entity of the *Messaging* Stanza.

Error types and error conditions that SHALL be supported are specified in Section 8.3.2 of [IETF RFC 6120, 2011]] and Section 8.9.3 of [IETF RFC 6120, 2011], respectively.

5.5 Additional Rules

An XMPP entity SHOULD follow the rules for processing XMPP *Message* Stanzas as specified in Section 8 of [IETF RFC 6121, 2011].

6 XMPP SERVICE DISCOVERY SERVICE

6.1 Service Interface

The *Service* interface is not a web service and therefore a WSDL is not appropriate to describe the operations of the service.

The *Service* interface is defined by the XMPP Standards Foundation (XSF) XEPs specified in [XSF XEP-0030, 2008], [XSF XEP-0055, 2009] and [XSF XEP-0060, 2010]. This service offers the functionality for a requesting XMPP entity to be able to discover target XMPP entities and discover information about the target XMPP entities on the XMPP network, such as:

- Protocols and features that a target XMPP entity can offer.
- The type and categorization of the target XMPP entity.

The service interface also provides the capability to search for other XMPP entities within the XMPP network.

6.2 Operations

The Service Discovery Service operations consist of a series of *IQ* Stanza requests and responses. The Service Discovery Service operations can be categorized into the following groups:

- Inquiry
- Publication
- Search.

Table 7 explains the list of operations.

Table 7
Service Discovery Service operations

Operation	Reference
`Inquiry`	This operation provides three types of discovery methods: Discovery of a target XMPP entity whereby the request message is qualified by the `'http://jabber.org/protocol/disco#items'` namespace. Discovery of a non-network target XMPP entity, known as an XMPP node, whereby the request message is qualified by the `'http://jabber.org/protocol/disco#items'` namespace. Discovery of information about a discovered target XMPP entity whereby the request message is qualified by the `'http://jabber.org/protocol/disco#info'`

	namespace.
`Publication`	Publication is supported if the requesting and target XMPP entities conform to XEP-0060: Publish-Subscribe as specified in [XSF XEP-0060, 2010]. This allows for information at the target XMPP entity to be updated and deleted. The message input, output and errors are further specified in Chapter 8 of this document.
`Search`	The operation, specified in [XSF XEP-0055, 2009], provides the capability to perform searching of XMPP entities within the XMPP network. A compliant XMPP entity SHALL publicize support for searching by returning '`jabber:iq:search`' in the Service Discovery Service Request (specified in Section 6.3.2.2) `var` attribute of the `feature` element.

Different use cases can be identified to leverage this capability dependent on the type of XMPP entity requesting information and the level of information that the XMPP entity needs returned. It is deemed outside of the scope of this document to identify the potential use cases.

6.3 Inquiry Messages

6.3.1 Input

6.3.1.1 Discover an addressable XMPP entity

In order to be able to discover information about an XMPP entity, a requesting XMPP entity must first be able to discover a target XMPP entity. A requesting XMPP entity, such as an *XMPP Client* will always know at least one target XMPP entity, i.e. the *XMPP Server* it is connected to.

A requesting XMPP entity SHALL first discover the network location of a target XMPP entity by following the specification (TCP Bindings and Domain Name System Services) described in Section 2.3.2.1 of [NCIA TR/2013/SPW008423/36, 2014].

An operation request SHALL be wrapped in an *IQ* Stanza whereby:

- The '`type`' attribute SHALL be '`get`'.
- The '`to`' attribute SHALL be the *JabberID* of the target XMPP entity.

The operation request SHALL contain an empty <query/> element that is qualified by the namespace '`http://jabber.org/protocol/disco#items`'.

6.3.1.2 Discover non-addressable XMPP entities

An XMPP node is an entity, associated with a target XMPP entity, and is not addressable on an XMPP network.

A requesting XMPP entity SHALL follow the semantics specified in Section 4.2 of [XSF XEP-0030, 2008] for discovering a XMPP node.

An operation request SHALL be wrapped in an *IQ* Stanza whereby:

- The '`type`' attribute SHALL be '`get`'.
- The '`to`' attribute SHALL be the *JabberID* of the target XMPP entity associated with the XMPP node.

The operation request SHALL contain a <query/> element that is qualified by the namespace '`http://jabber.org/protocol/disco#items`' and a '`node`' attribute containing the value for the XMPP node that is being discovered.

6.3.1.3 Discover information about a discovered XMPP entity

To determine the identity of the discovered XMPP entity and the features and protocols supported by that XMPP entity, the requesting XMPP entity is REQUIRED to send an Inquiry request to the discovered XMPP entity.

An operation request SHALL be wrapped in an *IQ* Stanza whereby:

- The 'type' attribute SHALL be 'get'.
- The 'to' attribute SHALL be the *JabberID* of the target XMPP entity.

The operation request SHALL contain an empty <query/> element that is qualified by the namespace 'http://jabber.org/protocol/disco#info'.

In the case where the discovery of information is for an XMPP node, the <query/> element SHALL also contain a 'node' attribute containing the value for the XMPP node, and the 'to' attribute SHALL be the *JabberID* of the target XMPP entity associated with the XMPP node.

6.3.2 Output

6.3.2.1 Response to request for discovering an XMPP entity

The operation response SHALL be wrapped in an *IQ* Stanza, whereby:

- The 'to' attribute SHALL be the *JabberID* of the requesting XMPP entity.
- The 'from' attribute SHALL be the *JabberID* of the target XMPP entity.
- The 'type' attribute SHALL be 'result'.
- Contains a <query/> child element that SHALL be qualified by an 'http://jabber.org/protocol/disco#items' namespace.

If there are no discoverable XMPP entities the <query/> child element SHALL be empty.

For each discoverable XMPP entity the <query/> child element SHALL contain an <item/> child element for each XMPP entity.

If the response message is for an Inquiry to discover an XMPP node, then the <query/> child element SHALL contain a 'node' attribute containing the value for the XMPP node that is being discovered.

A target XMPP entity SHALL construct an <item/> child element, for each discoverable XMPP entity hosted by the target XMPP entity, with the attributes specified in Table 8.

Table 8
Attributes contained in an item element

Attributes	Notes
jid	REQUIRED attribute that represents the *JabberID* of a discoverable XMPP entity.
name	OPTIONAL attribute.
node	REQUIRED attribute if the discovered XMPP entity is not addressable in the XMPP network (XMPP node). In this case, the 'jid' attribute SHALL be the *JabberID* of the XMPP entity hosting the XMPP node. Otherwise, this attribute SHOULD NOT be present.

This operation response SHALL adhere to the semantics and syntax specified in Sections 4.1 and 4.2 of [XSF XEP-0060, 2010].

6.3.2.2 Response to request for discovering information about an XMPP entity

The operation response SHALL be wrapped in a *IQ* Stanza, whereby:

- The '`to`' attribute SHALL be the *JabberID* of the requesting XMPP entity.
- The '`from`' attribute SHALL be the *JabberID* of the target XMPP entity.
- The '`type`' attribute SHALL be '`result`'.
- Contains a `<query/>` child element that SHALL be qualified by a '`http://jabber.org/protocol/disco#info`' namespace (the `<query/>` SHALL contain a '`node`' attribute if the request was for discovering information about an XMPP node).

Included as child elements of the `<query/>` element there SHALL be at least one `<identity/>` element and at least one `<feature/>` element (every compliant target XMPP entity SHALL support at least the '`http://jabber.org/protocol/disco#info`' '`feature`').

This operation response SHALL comply with the semantics and syntax specified in Section 3.1 of [XSF XEP-0030, 2008].

Tables 9 and 10 detail the attributes for the `<identity/>` and `<feature/>` elements, respectively.

Table 9
Attributes contained in an identity element

Attribute	Values	Comments
`category`	Refer to [XSF Service Discovery Identities, 2011] for the values that MAY be contained in the `category` attribute.	REQUIRED.
`name`	String	OPTIONAL.
`type`	Refer to [XSF Service Discovery Identities, 2011] for the values that MAY be contained in the `type` attribute associated with each `category`.	REQUIRED.

Table 10
Attributes contained in a features element

Attribute	Values	Comments
`var`	Refer to [XSF Namespaces, 2011] and [XSF Service Discovery Features, 2010] for the `feature` values that MAY be a value of the `var` attribute.	REQUIRED.

The value of a 'node' attribute MAY contain values listed at [XSF Nodes, 2004] for well-known service discovery nodes.

6.3.3 Errors

An *IQ* Stanza with a 'type' attribute of 'error' SHALL be sent in the case of errors associated with any of the operation requests.

The list of errors that SHALL be supported is specified in Section 7 of [XSF XEP-0030, 2011].

6.4 Search

6.4.1 Input

An operation request SHALL be wrapped in an *IQ* Stanza whereby:

- The 'type' attribute SHALL be 'set'.
- The 'to' attribute SHALL be the *JabberID* of the XMPP entity providing the search capability.

The operation request SHALL contain a <query/> element that is qualified by the namespace 'jabber:iq:search'.

The <query/> element SHALL contain a child element(s), representing search fields, supported by the XMPP entity providing the search capability.

The search fields that SHALL be supported by the 'jabber:iq:search' are specified in Section 7.1 of [XSF XEP-0055, 2009].

6.4.2 Output

An operation response SHALL be wrapped in an *IQ* Stanza whereby the 'type' attribute SHALL be 'result'.

The operation response SHALL contain a <query/> element qualified by the namespace 'jabber:iq:search'.

If there are no matching entries, the response SHALL return an empty <query/> element.

A response that contains matching entries SHALL contain a <query/> element qualified by the namespace 'jabber:iq:search', and containing at least one <item/> element.

The <item/> element SHALL contain a 'jid' attribute representing the *JabberID* of the matched XMPP entity.

The <item/> element MAY contain child element(s), representing search fields, supported by the XMPP entity providing the search capability.

The search fields that SHALL be supported by the 'jabber:iq:search' are specified in Section 7.1 of [XSF XEP-0055, 2009].

7 MULTI-PARTY MESSAGING SERVICE

7.1 Service Interface

The *Service* interface is not a web service and therefore a WSDL is not appropriate to describe the operations of the service.

The *Service* interface is defined by the XMPP Standards Foundation (XSF) XMPP Extension Protocols (XEPs) specified in [XSF XEP-0045, 2011] and further augmented with support for Data Forms [XSF XEP-0004, 2007], Geolocation specified in [XSF XEP-0080, 2009], Conference Bookmarks specified in [XSF XEP-0048, 2007] and Date Time Group (DTG) display format specified in [XSF XEP-0203, 2009].

The Multi-Party Messaging Service offers a real-time multi-party interaction, known as group chat or text-conferencing, whereby XMPP entities can join chat rooms and exchange messages and presence information.

The Multi-Party Messaging Service can be leveraged to transport customized and structured data among multiple XMPP entities on the XMPP network.

7.2 Operations

The Multi-Party Message Service operations support the functionality for an XMPP entity:

- To discover chat rooms hosted by any XMPP entity
- To join and leave chat rooms
- To participate in chat rooms by exchanging messages and presence information with other members within those chat rooms
- Invite other XMPP entities on the XMPP network to a chat room
- Transport any payload within XMPP *Message* stanzas for processing by XMPP entities participating in the chat room.

The Multi-Party Messaging Service operations can be categorized as detailed in Table 11.

7.3 DATA Structure

7.3.1 Discover operation

The Discover operation SHALL follow the specifications detailed in Chapter 6 of this document for requesting information and items from XMPP entities and chat rooms that support the Multi-Party Messaging Service.

7.3.1.1 Input

The structure of the *IQ* Stanza for requesting information or items of an XMPP entity supporting this *Service* SHALL be conformant with the specifications detailed in Sections 6.3.1.1 and 6.3.1.3 of this document.

Table 11
Multi-party messaging service operations

Operation	Reference
Discover	The operation provides a capability to query an XMPP entity or a chat room to determine the features that are available. The operation is specified in [XSF XEP-0045, 2011] Section 6.
Join	The operation provides the capability for an XMPP entity to request to join a chat room. The operation is specified in [XSF XEP-0045, 2011] Section 7.1.2. The operation is further augmented by allowing an XMPP entity to auto-join a chat room specified in [XSF XEP-0048, 2007].

Notify	The operation occurs after an XMPP entity has requested to join a chat room. The notify operation can be categorized as: Inform existing members of the XMPP entity that joined the chat room (specified in [XSF XEP-0045, 2011] Section 7.1.3) Share the members of the chat room with the new XMPP entity (specified in [XSF XEP-0045, 2011] Section 7.1.3) Share a history of the messages exchanged in the chat room with the new XMPP entity (specified in [XSF XEP-0045, 2011] Sections 7.1.15 and 7.1.16).
Chat	The operation supports the exchange of messages between the members of the chat room (specified in [XSF XEP-0045, 2011] Sections 7.8 and 7.9). The operation supports the exchange of structured data supporting the exchange of: User Location Data as specified in [XSF XEP-0080, 2009] qualified by XML namespace 'http://jabber.org/protocol/geoloc'; Alert Data as specified in [XSF XEP-0127, 2004] qualified by XML namespace 'http://www.incident.com/cap/1.0'; and, Dynamic Data Forms as specified in Chapter 11.
Invite	The operation allows a member of a chat room to invite another XMPP entity to the chat room.
Part	The operation notifies the chat room that an XMPP entity has left the chat room (specified in [XSF XEP-0045, 2011] Section 7.2).

7.3.1.2 Output

The structure of the *IQ* Stanza for the responses SHALL be conformant with the specifications detailed in Sections 6.3.2.1 and 6.3.2.2 of this document.

A compliant XMPP entity SHALL publicize support for the Multi-Party Messaging Service by returning 'http://jabber.org/protocol/muc' in the Service Discovery Service Request (specified in Section 6.3.2.2) var attribute of the feature element.

A Service Discovery Service request for information relating to a chat room MAY contain any of the following features in the var attribute of the feature element of the response (specified in [XSF XEP-0045, 2011] Section 15.3):

- muc_hidden
- muc_membersonly
- muc_moderated
- muc_nonanonymous
- muc_open
- muc_passwordprotected
- muc_persistent
- muc_public
- muc_rooms

- muc_semianonymous
- muc_temporary
- muc_unmoderated
- muc_unsecured.

The response SHOULD only contain the features that are supported by the chat room.

The response MAY contain more detailed information related to the chat room.

A response that provides more detailed information SHALL include a `FORM_TYPE` with the attribute 'type' set to `hidden` and qualified by the XML namespace 'http://jabber.org/protocol/muc#roominfo' (specified in Section 15.5.4 of [XSF XEP-0045, 2011]) as a child element of an <x/> element that is qualified by the 'jabber:x:data' namespace with a 'type' attribute of `result` (specified in [XSF XEP-0004, 2007]).

7.3.2 Join operation

An XMPP entity sends a request to join a chat room by submitting a *Presence* Stanza to the XMPP entity hosting the chat room. If the XMPP entity hosting the chat room can add the XMPP entity to the chat room, the notify operation specified in Section 7.3.3 SHALL be invoked.

The join operation can be further augmented by supporting the ability to bookmark chat rooms, whereby providing the ability to auto-join chat rooms.

An XMPP entity that provides this capability SHALL be compliant with the specifications detailed in [XSF XEP-0048, 2007].

7.3.2.1 Input

The operation request SHALL be wrapped in a *Presence* Stanza (described in Section 2.3.3.2 of [NCIA TR/2013/SPW008423/36, 2014]) whereby the 'to' attribute is the *JabberID* of the room and includes an empty <x/> element qualified by the 'http://jabber.org/protocol/muc' namespace as specified in Section 7.1.2 of [XSF XEP-0045, 2011].

7.3.2.2 Output

No output is specified by for this service interface.

7.3.3 Notify operation

The XMPP hosting the room, if it can add the XMPP entity requesting to join the chat room, SHALL:

- Notify all the members of the room by sending a *Presence* Stanza to each member of the chat room (See Section 7.3.3.1)
- Notify the XMPP entity requesting to join the chat room of the other members in the chat room by sending a *Presence* Stanza from each of the members (see Section 7.3.3.2)
- Notify the XMPP entity requesting to join the chat room of its status within the room (see Section 7.3.3.3)
- Notify the XMPP entity requesting to join the chat room of a history of all the messages (see Section 7.3.3.4).

7.3.3.1 Notify new member input

The operation request SHALL be wrapped in a *Presence* Stanza (described in [NCIA TR/2013/SPW008423/36, 2014] Section 2.3.3.2) whereby:

- The 'to' attribute is the *Full JID* of an existing member.
- The 'from' attribute is the chat room *JabberID* of the new member.

- Including an <x/> element qualified by the namespace 'http://jabber.org/protocol/muc#user' with an <item/> child element containing affiliation and role attributes (values specified in Section 7.1.4 of [XSF XEP-0045, 2011]) for the new member, as specified in Section 7.1.3 of [XSF XEP-0045, 2011].

This operation request SHALL be invoked for each member of the chat room.

7.3.3.2 Notify all members input

The operation request SHALL be wrapped in a *Presence* Stanza (described in Section 2.3.3.2 of [NCIA TR/2013/SPW008423/36, 2014]) whereby:

- The 'to' attribute is the *Full JID* of the new member.
- The 'from' attribute is the chat room *JabberID* of an exisiting member.
- Including an <x/> element qualified by the namespace 'http://jabber.org/protocol/muc#user' with an <item/> child element containing affiliation and role attributes (values specified in Section 7.1.4 of [XSF XEP-0045, 2011]) for the existing member, as specified in Section 7.1.3 of [XSF XEP-0045, 2011].

This operation request SHALL be invoked for each member of the chat room.

7.3.3.3 Notify status input

This operation request SHALL be invoked after the operation request specified in Section 7.3.3.2 has been executed. This assists the XMPP entity that requested to join the chat room to understand that all the members of the chat room have been sent and received to the new member.

The operation request SHALL be wrapped in a *Presence* Stanza (described in Section 2.3.3.2 of [NCIA TR/2013/SPW008423/36, 2014]) whereby:

- The 'to' attribute is the *Full JID* of the new member.
- The 'from' attribute is the room *JabberID* of the new member.
- Including an <x/> element qualified by the namespace 'http://jabber.org/protocol/muc#user' with an <item/> child element containing affiliation and role attributes (values specified in Section 7.1.4 of [XSF XEP-0045, 2011]) for the new member, as specified in Section 7.1.3 of [XSF XEP-0045, 2011].

The <x/> element MAY also contain one or more <status/> elements containing a code attribute. The values of the code attribute SHALL be compliant with the values specified in Section 15.6.2 of [XSF XEP-0045, 2011].

7.3.3.4 Notify messages input

The operation request SHALL be wrapped in a *Message* Stanza as specified in Section 2.3.3.1 of [NC3A RD-3186, 2011].

The *Message* Stanza SHALL have a 'type' attribute with value 'groupchat' and include a <delay/> element qualified by the 'urn:xmpp:delay' namespace indicating that the message is delayed and the date and time the *Message* Stanza was originally sent (as specified in [XSF XEP-0203, 2009] conformant with XMPP date and time profiles specified in [XSF XEP-0082, 2003]).

The number of historical messages that can be sent to the new member of the chat room is outside the scope of this document. This is accomplished through local configuration settings on the XMPP entity hosting the Multi-Party Messaging Service.

7.3.3.5 Output

No output is specified by for this service interface.

7.3.3.6 Chat operation

The Chat operation supports the ability of an XMPP entity to participate in text-conferencing by exchanging messages and exchanging presence with each member of a chat room. This operation also supports the ability for one XMPP entity to direct one-to-one chat with another XMPP entity in the chat room. This operation can be categorized into the following groups:

4) Group Chat – Exchange messages with all members in a chat room
5) One-to-One Chat – Direct messages to a single XMPP entity in a chat room
6) Presence – Share presence with all members in a chat room.

7.3.3.7 Group chat input

The operation request SHALL be wrapped in a *Message* Stanza whereby:

- The '`from`' attribute is the *Full JID* of the sending XMPP entity member.
- The '`to`' attribute is the *JabberID* of the chat room.
- The '`type`' attribute has a value of '`groupchat`'.

An XMPP entity MAY request that the *Message* Stanza is delivered before a certain absolute point in time.

XMPP entities that support time-sensitive messaging SHALL support the Time Sensitive Messaging Service specified in Chapter 12.

The *Message* Stanza SHALL contain an `<amp/>` element qualified by the '`http://jabber.org/protocol/amp`' namespace with a `<rule/>` child element.

Table 12 details the REQUIRED attributes and values associated with the `<rule/>` element.

Table 12
REQUIRED rule attributes and values

Attribute	Notes
`action`	`alert; drop; error; or notify`
`condition`	`expire-at`
`value`	Time when the Message Stanza expires in UTC

7.3.3.8 One-to-one chat input

The operation request SHALL be wrapped in a *Message* Stanza whereby:

- The '`from`' attribute is the *Full JID* of the sending XMPP entity member.
- The '`to`' attribute is the chat room *JabberID* of the receiving XMPP entity.
- The '`type`' attribute has a value of '`chat`'.

An XMPP entity MAY request that the *Message* Stanza is delivered before a certain absolute point in time.

XMPP entities that support time-sensitive messaging SHALL support the Time-Sensitive Messaging Service specified in Chapter 12.

The *Message* Stanza SHALL contain an `<amp/>` element qualified by the 'http://jabber.org/protocol/amp' namespace with a `<rule/>` child element.

Table 13 details the REQUIRED attributes and values associated with the `<rule/>` element.

Table 13
REQUIRED rule attributes and values

Attribute	Notes
action	alert; drop; error; or notify
condition	expire-at
value	Time when the Message Stanza expires in UTC

7.3.3.9 Presence input

The operation request SHALL be wrapped in a *Presence* Stanza whereby:

- The 'from' attribute is the *Full JID* of the sending XMPP entity member.
- The 'to' attribute is the *JabberID* of the chat room.

7.3.3.10 Group chat output

The operation output SHALL be wrapped in a *Message* Stanza whereby:

- The 'from' attribute is the chat room *JabberID* of the sending XMPP entity member.
- The 'to' attribute is the *Full JID* of a member of the chat room.
- The 'type' attribute has a value of 'groupchat'.

This operation request SHALL be invoked for every member of the chat room.

7.3.3.11 One-to-one chat output

The operation output SHALL be wrapped in a *Message* Stanza whereby:

- The 'from' attribute is the chat room *JabberID* of the sending XMPP entity member.
- The 'to' attribute is the *Full JID* of the receiving XMPP entity.
- The 'type' attribute has a value of 'chat'.

7.3.3.12 Presence output

The operation output SHALL be wrapped in a *Presence* Stanza whereby:

- The 'from' attribute is the chat room *JabberID* of the sending XMPP entity member.
- The 'to' attribute is the *Full JID* of a member of the chat room.

This operation request SHALL be invoked for every member of the chat room.

7.3.4 Invite operation

This operation provides a chat room mediated approach for an XMPP entity, already a member of a chat room, with the capability to invite another XMPP entity to that chat room.

7.3.4.1 Input

The operation request SHALL be wrapped in a *Message* Stanza whereby:

- The 'from' attribute is the *Full JID* of the sending XMPP entity member.
- The 'to' attribute is the *JabberID* of the chat room.

The *Message* Stanza SHALL contain an <x/> element qualified by the 'http://jabber.org/protocol/muc#user' namespace with an <invite/> child element.

Table 14 details the attributes and elements associated with the <invite/> element.

Table 14
Invite attributes and elements

Attribute/element	Notes
to	REQUIRED element that represents the *Bare JID* of the XMPP entity to be invited.
from	NOT REQUIRED.
reason	OPTIONAL child element that provides a string value containing a reason for the invitation.

7.3.4.2 Output

The operation output SHALL be wrapped in a *Message* Stanza whereby:

- The 'from' attribute is the *JabberID* of the chat room.
- The 'to' attribute is the *Bare JID* of the invited XMPP entity.

The *Message* Stanza SHALL contain an <x/> element qualified by the 'http://jabber.org/protocol/muc#user' namespace with an <invite/> child n with a 'from' attribute containing the *Bare JID* of the XMPP entity to be invited.

A compliant XMPP entity supporting this operation SHALL follow semantics and syntax as specified in Section 7.5.2 of [XSF XEP-0045, 2011].

7.3.5 Part operation

Exiting a chat room is specified in Section 7.2 of [XSF XEP-0045, 2011].

7.3.5.1 Input

The operation request SHALL be wrapped in a *Presence* Stanza whereby:

- The 'to' attribute is the chat room *JabberID* of the XMPP entity requesting to leave.
- The 'from' attribute is the *Full JID* of the XMPP entity requesting to leave.
- The 'type' attribute SHALL be set to 'unavailable'.

7.3.5.2 Output

The operation request SHALL be wrapped in a *Presence* Stanza whereby:

- The 'to' attribute is the *Full JID* of an existing member.
- The 'from' attribute is the room *JabberID* of the new member.
- The 'type' attribute SHALL be set to 'unavailable'.

- Including an <x/> element qualified by the namespace 'http://jabber.org/protocol/muc#user' with an <item/> child element containing affiliation and role attributes as specified in Section 7.2 of [XSF XEP-0045, 2011].

This operation request SHALL be invoked for every member of the chat room.

7.4 Errors

A *Message* Stanza or *Presence* Stanza with a 'type' attribute of 'error' SHALL be sent in the case of errors associated with any of the operation requests.

Error types and error conditions that SHALL be supported are specified in Tables 13 and 14 of [NCIA TR/2013/SPW008423/36, 2014].

The Multi-Party Messaging Service MAY support error conditions as specified in Section 11 of [XSF XEP-0045, 2011] that are associated with the namespace 'http://jabber.org/protocol/muc#user'.

A compliant XMPP entity supporting the Multi-Party Messaging Service SHALL follow semantics specified in [XSF XEP-0045, 2011] for error-handling.

8 NOTIFICATION SERVICE

8.1 Service Interface

The *Service* interface is not a web service and therefore a WSDL is not appropriate to describe the operations of the service.

The *Service* interface is defined by the XSF XEPs specified in [XSF XEP-0060, 2010], [XSF XEP-0127, 2004].

The Notification Service provides a publish/subscribe (PubSub) capability, whereby XMPP entities that are interested in specific information are notified and updated with new published information as opposed to continually polling for new information. The Notification Service also supports a Push model for sending alerts as specified in [XSF XEP-0127, 2004].

The Notification Service can be leveraged to transport customized and structured data among multiple XMPP entities on the XMPP network.

8.2 Operations

A compliant XMPP entity SHALL publicize support for the Notification Service by returning 'http://jabber.org/protocol/pubsub' in the Service Discovery Service Request (specified in Section 6.3.2.2) var attribute of the feature element.

The Notification Service operations can be categorized in the following groups:

1) Subscription – Subscribe, Unsubscribe
2) Update Information – Publish, Delete, Alert
3) Broadcast Information – Notify.

Table 15 shows the list of operations as specified in [XSF XEP-0060, 2010.

Table 15
List of notification service operations

Operation	Reference
Subscribe	[XSF XEP-0060, 2010] Section 6.1
Unsubscribe	[XSF XEP-0060, 2010] Section 6.2
Publish	[XSF XEP-0060, 2010] Section 7.1
Delete	[XSF XEP-0060, 2010] Section 7.2
Alert	[XSF XEP-0127, 2004]
Notify	[XSF XEP-0060, 2010] Section 7.1.2.1, 7.1.2.2 and 7.2.2.1

8.3 Subscription Operations

8.3.1 Subscribe

An XMPP entity subscribes to a PubSub node by sending a subscription request to the XMPP entity hosting the PubSub node.

The subscription request SHALL be acknowledged by the XMPP entity hosting the PubSub node.

8.3.1.1 Input

A Subscribe operation request SHALL be wrapped in an *IQ* Stanza whereby:

- The '`type`' attribute SHALL be '`set`'.
- The '`from`' attribute is the *Full JID* of the requesting XMPP entity.
- The '`to`' attribute SHALL be the *JabberID* of the target XMPP entity hosting the PubSub node.

The operation request SHALL contain a `<pubsub/>` element that is qualified by the namespace '`http://jabber.org/protocol/pubsub`'.

The `<pubsub/>` element SHALL contain a `<subscribe/>` element that is specified in Table 16.

Table 16
Attributes contained in a subscribe element

Attribute	Comments
`node`	REQUIRED attribute that contains the name of PubSub node that the XMPP entity is requesting subscription for.
`jid`	REQUIRED attribute. The value SHALL represent the subscribed *JabberID* for the requesting XMPP entity.

8.3.1.2 Output

A Subscribe operation response SHALL be wrapped in an *IQ* Stanza whereby:

- The 'type' attribute SHALL be 'result'.
- The 'from' attribute is the *JabberID* of the target XMPP entity hosting the PubSub node.
- The 'to' attribute SHALL be the *Full JID* of the requesting XMPP entity.

The operation request SHALL contain a <pubsub/> element that is qualified by the namespace 'http://jabber.org/protocol/pubsub'.

The <pubsub/> element SHALL contain a <subscription/> element that is specified in Table 17.

Table 17
Attributes contained in a subscription element

Attribute	Comments
node	REQUIRED attribute that contains the name of PubSub node that the XMPP entity has requested subscription to.
jid	REQUIRED attribute. The value SHALL represent the subscribed *JabberID* for the requesting XMPP entity.
subid	OPTIONAL attribute. The XMPP entity hosting the PubSub node MAY include this attribute.
subscription	OPTIONAL attribute. If the attribute is set, the value SHOULD be 'subscribed'.

8.3.1.3 Errors

An *IQ* Stanza with a 'type' attribute of 'error' SHALL be sent in the case of errors associated with a Subscribe operation request.

The XMPP entity originating an error SHALL be compliant with the specifications described in Section 6.1.3 of [XSF XEP-0060, 2010].

If an XMPP entity specifies an error condition that is specific to this operation, as specified in Section 6.1.3 of [XSF XEP-0060, 2010], the XMPP entity SHALL qualify the specific error with a 'http://jabber.org/protocol/pubsub#errors' XML namespace.

8.3.2 Unsubscribe

The Notification Service offers the capability to unsubscribe from a PubSub node. The unsubscribe operation supports this capability, whereby a request to unsubscribe is acknowledged with a response.

8.3.2.1 Input

An Unsubscribe operation request SHALL be wrapped in an *IQ* Stanza whereby:

- The 'type' attribute SHALL be 'set'.
- The 'from' attribute is the *Full JID* of the requesting XMPP entity.
- The 'to' attribute SHALL be the *JabberID* of the target XMPP entity hosting the PubSub node.

The operation request SHALL contain a <pubsub/> element that is qualified by the namespace 'http://jabber.org/protocol/pubsub'.

The <pubsub/> element SHALL contain an <unsubscribe/> element that is specified in Table 18.

Table 18
Attributes contained in a unsubscribe element

Attribute	Comments
node	REQUIRED attribute that contains the name of PubSub node that the XMPP entity is unsubscribing from.
jid	REQUIRED attribute. The value SHALL represent the subscribed *JabberID* for the XMPP entity requesting to be unsubscribed.

8.3.2.2 Output

An Unsubscribe operation response SHALL be wrapped in an *IQ* Stanza whereby:

- The 'type' attribute SHALL be 'result'.
- The 'from' attribute is the *JabberID* of the target XMPP entity hosting the PubSub node.
- The 'to' attribute SHALL be the *Full JID* of the XMPP entity that requested to be unsubscribed.

8.3.2.3 Errors

An *IQ* Stanza with a 'type' attribute of 'error' SHALL be sent in the case of errors associated with a Subscribe operation request.

The XMPP entity originating an error SHALL be compliant with the specifications described in Section 6.2.3 of [XSF XEP-0060, 2010].

8.4 Update Information Operations

8.4.1 Publish

The Publish operation provides the capability for an XMPP entity to publish information to a PubSub node.

8.4.1.1 Input

A Publish operation request SHALL be wrapped in an *IQ* Stanza (described in [NCIA TR/2013/SPW008423/36, 2014]) whereby:

- The 'type' attribute SHALL be 'set'.
- The 'from' attribute is the *Full JID* of the requesting XMPP entity.
- The 'to' attribute SHALL be the *JabberID* of the target XMPP entity hosting the PubSub node.

The operation request SHALL contain a <pubsub/> element that is qualified by the namespace 'http://jabber.org/protocol/pubsub'.

The <pubsub/> elements are described in Table 19.

Table 19
Elements contained within the PubSub element for a publish request

Element	Comments
/publish	REQUIRED element that indicates that the XMPP entity wishes to publish an item.
/publish:node	REQUIRED attribute. The value identifies the PubSub node that the XMPP entity wishes to publish to.
/publish/item	REQUIRED element that is used to wrap the payload </entry> element that is to be published to the PubSub node.
/publish/item:id	REQUIRED attribute and SHALL be unique for the PubSub node that the item is being published.
/publish/item/<entry>	REQUIRED element that contains the actual payload. The payload SHALL be qualified by an XML namespace. An example payload MAY be the CAP payload specified in [XSF XEP-0127, 2004], and specifically Section 3.2 for a PubSub node.

8.4.1.2 Output

A Publish operation response SHALL be wrapped in an *IQ* Stanza whereby:

- The 'type' attribute SHALL be 'result'.
- The 'from' attribute is the *JabberID* of the ta.get XMPP entity hosting the PubSub node.
- The 'to' attribute SHALL be the *Full JID* of the XMPP entity that published the item.

The operation response SHALL contain a <pubsub/> element that is qualified by the namespace 'http://jabber.org/protocol/pubsub'.

The <pubsub/> elements specific to the response are described in Table 20.

Table 20
Elements contained within the PubSub element for a Publish response

Element	Comments
/publish	REQUIRED element.
/publish:node	REQUIRED attribute that identifies the PubSub node that the XMPP entity published to.
/publish/item	REQUIRED element.
/publish/item:id	REQUIRED attribute that SHALL be the unique identifier specified in the publish request.

8.4.1.3 Errors

An *IQ Stanza* with a 'type' attribute of 'error' SHALL be sent in the case of errors associated with a Publish operation request.

The XMPP entity originating an error SHALL be compliant with the specifications described in Section 7.1.3 of [XSF XEP-0060, 2010].

8.4.2 Delete

8.4.2.1 Input

A Delete operation request SHALL be wrapped in an *IQ* Stanza (described in [NCIA TR/2013/SPW008423/36, 2014]) whereby:

- The 'type' attribute SHALL be 'set'.
- The 'from' attribute is the *Full JID* of the requesting XMPP entity.
- The 'to' attribute SHALL be the *JabberID* of the target XMPP entity hosting the PubSub node.

The operation request SHALL contain a <pubsub/> element that is qualified by the namespace 'http://jabber.org/protocol/pubsub'.

The <pubsub/> elements are described in Table 21.

Table 21
Elements contained within the PubSub Element for a delete request

Element	Comments
/retract	REQUIRED element that indicates that the XMPP entity wishes to delete an item.
/retract:node	REQUIRED attribute that identifies the PubSub node that the XMPP entity wishes to delete an item from.
/retract:notify	OPTIONAL attribute. This indicates, if the value is set to 'true' or '1', that the XMPP entity hosting the PubSub node SHALL notify the other XMPP entities that have subscribed to the PubSub node of the delete operation.
/retract/item	REQUIRED element and there SHALL only be one item.
/retract/item:id	REQUIRED attribute that SHALL be the unique identifier that represents the item to be deleted.

8.4.2.2 Output

A Delete operation response SHALL be wrapped in an *IQ* Stanza whereby:

- The 'type' attribute SHALL be 'result'.
- The 'from' attribute is the *JabberID* of the target XMPP entity hosting the PubSub node.
- The 'to' attribute SHALL be the *Full JID* of the XMPP entity that deleted the item.

8.4.2.3 Errors

An *IQ Stanza* with a 'type' attribute of 'error' SHALL be sent in the case of errors associated with a Publish operation request.

The XMPP entity originating an error SHALL be compliant with the specifications described in Section 7.2.3 of [XSF XEP-0060, 2010].

8.4.3 Alert

8.4.3.1 Input

An alert operation message SHALL be wrapped in a *Message* Stanza whereby:

- The 'from' attribute is the *JabberID* of the sending XMPP entity.
- The 'to' attribute SHALL be the *JabberID* of the target XMPP entity.
- The 'type' attribute MAY be present, however, if the Alert message is directed at a Multi-Party Messaging Service, the attribute value SHALL be set to 'groupchat' (as specified in Section 7.3.3.7).

The operation message SHALL contain an <alert/> element that is qualified by the namespace 'http://www.incident.com/cap/1.0'.

The <event/> elements that SHALL be supported are specified in Section 3.1 of [XSF XEP-0127, 2004].

8.4.3.2 Output

No output is specified by for this service interface.

8.5 Notifications Operations

Notifying XMPP entities that have subscribed to a PubSub node occurs when an XMPP entity publishes an item to a PubSub node or deletes an item from a PubSub node and the notify attribute of the <retract/> element is set as specified in Table 21. [XSF XEP-0060, 2010] specifies that a notification can be sent with or without a payload for published items.

This service interface only RECOMMENDS the case where an item is published with a payload.

Further revisions of this service interface MAY specify the case whereby a notification is sent without a payload and the XMPP entity subscribed to the PubSub node has to request the newly published item.

8.5.1 Notify

For each Notify operation (Notify operation on publish and Notify operation on delete) a notification is sent to each XMPP entity that has subscribed to the PubSub node.

8.5.1.1 Notify on publish input

The operation request SHALL be wrapped in a *Message* Stanza whereby:

- The 'from' attribute is the *JabberID* of the XMPP entity hosting the PubSub node.
- The 'to' attribute is the subscribed *JabberID* (see Section 8.3.1.1) of the subscribed XMPP entity.

The *Message* Stanza SHOULD contain a unique 'id' attribute for correlation-tracking of notification messages.

The operation request SHALL contain an <event/> element that is qualified by the namespace 'http://jabber.org/protocol/pubsub#event'. The <event/> elements are described in Table 22.

Table 22
Elements contained within the event element for a Notify on publish request

Element	Comments
/items	REQUIRED element.
/items:node	REQUIRED attribute that identifies the PubSub node to the subscribed XMPP entity that is receiving the published item.
/items/item	REQUIRED element and there SHALL be one or more item.
/event/item:id	REQUIRED attribute that SHALL be the unique identifier that represents the published item.
/event/item:node	OPTIONAL attribute and, if present, SHALL be the unique identifier that represents the PubSub node.
/event/item:publisher	OPTIONAL attribute. If present, the value SHALL be the subscribed *JabberID* of the XMPP entity that published the item.
/event/item/<entry>	REQUIRED element that contains the published payload. The payload SHALL be qualified by an XML namespace. An example payload MAY be the CAP payload specified in [XSF XEP-0127, 2004], and specifically Section 3.2 for a PubSub node.

8.5.1.2 Notify on publish output

No output is specified for this service interface.

8.5.1.3 Notify on delete input

The operation request SHALL be wrapped in a *Message* Stanza whereby:

- The '`from`' attribute is the *JabberID* of the XMPP entity hosting the PubSub node.
- The '`to`' attribute is the subscribed *JabberID* (see Section 8.3.1.1) of the subscribed XMPP entity.

The *Message Stanza* SHOULD contain a unique '`id`' attribute for correlation-tracking of notification messages.

The operation request SHALL contain an `<event/>` element that is qualified by the namespace '`http://jabber.org/protocol/pubsub#event`'. The `<event/>` elements are described in Table 23.

Table 23
Elements contained within the event element for a Notify on delete request

Element	Comments
/items	REQUIRED element.

/items:node	REQUIRED element that identifies the PubSub node to the subscribed XMPP entity that is receiving the published item.
/items/retract	REQUIRED element and there SHALL be one or more present.
/event/retract:id	REQUIRED attribute that SHALL be the unique identifier that represents the item to be deleted.

8.5.1.4 Notify on delete output

No output is specified for this service interface.

8.5.1.5 Errors

An *IQ* Stanza with a 'type' attribute of 'error' SHALL be sent in the case of errors associated with a Publish operation request.

The XMPP entity originating an error SHALL be compliant with the specifications described in Sections 7.1.3 and 7.2.3 of [XSF XEP-0060, 2010].

8.5.2 Additional considerations

In the event that a subscribed XMPP entity has been off-line, the XMPP entity hosting the PubSub nodes SHOULD support Last Activity in Presence as specified in [XSF XEP-0256, 2009]. The subscribed XMPP entity should receive all notifications, supported by the operations specified in Section 8.5.1, since the time that the XMPP entity was last available on the XMPP network.

9 LABELLING SERVICE

9.1 Service Interface

The *Service* interface is not a web service and therefore a WSDL is not appropriate to describe the operations of the service.

The *Service* interface is defined by the XSF XEP specified in [XSF XEP-0258, 2011].

The *Service* interface supports the concept of providing a consistent and coherent labelling service, for the Instant Messaging Collaboration Service across the NATO enterprise, whereby an XMPP entity can request a catalogue of security labels and apply a security label without needing to understand the complexity of:

- The security label – the syntax, semantics and validity against the security policy.
- The access control mechanisms that may constrain the distribution or release of the message, based on the clearance of the identities involved in the message transaction, whereby an identity may be a NATO role, coalition role, multi-user chat room, service, foreign domain or boundary interface.

The *Service* interface offers a lightweight approach for obtaining a set of valid security labels, whereby an XMPP operator can select one of the returned security labels from the set to be applied to the XMPP message. This *Service* interface supports the collaboration requirement for applying a confidentiality label to XMPP messages for cross-domain collaboration.

9.2 Operations

A compliant XMPP entity SHALL publicize support for the labelling service by returning 'urn:xmpp:sec-label:catalog:2' in the Service Discovery Service Request (specified in Section 6.3.2.2) var attribute of the feature element.

The Labelling Service interface offers the following operations:

- Get
- Set.

9.2.1 Get operation

The Get operation consists of an *IQ* Stanza request and response, whereby the target XMPP entity is requested to return a catalogue containing valid security labels. The catalogue returned depends on the associated information assurance (IA) attributes of the requesting XMPP entity and intended target XMPP entity identities.

9.2.2 Set operation

The Set operation consists of a *Message* Stanza containing the selected security label element.

9.3 Get Messages

This operation request and response SHALL adhere to the semantics and syntax specified in Section 4 of [XSF XEP-0258, 2011].

9.3.1 Input

An operation request SHALL be wrapped in an *IQ* Stanza (described in [NCIA TR/2013/SPW008423/36, 2014]) whereby:

- The 'type' attribute SHALL be 'get'.
- The 'to' attribute SHALL be the *JabberID* of the target XMPP entity hosting the labelling service.

The operation request SHALL contain a <catalog/> element that is qualified by the namespace 'urn:xmpp:sec-label:catalog:2' and contains a 'to' attribute of value pertaining to the target XMPP entity for the XMPP message.

9.3.2 Output

The operation response SHALL be wrapped in an *IQ* Stanza, whereby:

- The 'type' attribute SHALL be 'result'.
- The 'to' attribute SHALL be the *JabberID* of the requesting XMPP entity.
- Contains a <catalog/> child element that SHALL be qualified by a 'urn:xmpp:sec-label:catalog:2' namespace.

If there are no valid security labels for the XMPP entities, the <catalog/> child element SHALL be empty.

For each valid security label (including no label – if supported by the security policy[3]) the <catalog/> child element SHALL contain an <item/> child element.

---

[3] The governing Security Policy for a security domain MAY not require that an XMPP message contains a

In the case where no label is supported, the <catalog/> element SHALL contain only one empty <item/> elements.

Table 24 specifies the attributes that are defined for the <catalog/> element.

An <item/> element SHALL contain a <securitylabel/> item, and SHALL be qualified with the 'urn:xmpp:sec-label:0' namespace, for each valid security label to be returned in the operation response.

If a <securitylabel/> item is to be used as a default selector the <item/> element attribute 'default' SHALL be set to true.

The element metadata is detailed in Table 25.

Table 24
Attributes contained in a catalogue element

Attribute	Comments
/catalog:to	REQUIRED attribute that provides the identity of the target XMPP entity, whereby the IA attributes can be retrieved from an attribute store (Enterprise Directory Service) to be used as part of the access control information for an access control decision function.
/catalog:from	OPTIONAL attribute.
/catalog:name	OPTIONAL attribute.
/catalog:desc	OPTIONAL attribute.
/catalog:id	REQUIRED attribute for correlation and audit requirements.
/catalog:size	OPTIONAL attribute.
/catalog:restrict	REQUIRED attribute and the value must be set to true. This ensures that the XMPP operator is expected to make a choice, even if the choice is no label (if supported by the security policy).

Table 25
Elements contained in a securitylabel element

Element	Comments
/securitylabel/displaymarking	REQUIRED element for providing a human-readable representation of the security label for the XMPP operator. The fgcolor and bgcolor attributes SHOULD be present and be conformant with [XMPP.org "Security Label Schema", 2011] as specified in [XSF XEP-0258, 2011].

security label.

/securitylabel/label	REQUIRED element for providing the machine readable (encoded) security label. The <label/> element SHALL contain an encoded label child element that SHOULD be defined by an XML namespace. The Labelling Service SHALL support the following XML namespaces: 'urn:xmpp:sec-label:ess:0' – Represents an RFC 2634 ESS Security Label ([IETF RFC 2634, 1999]) as specified in [XSF XEP-0258, 2011] 'urn:us:gov:ic:ism:v2' – Represents the Intelligence Community Information Security Marking Version 2.1 as defined at [ODNI IC-ISM, 2008] 'http://www.nato.int/2012/12/nxl/xcl#human' – Defined in [NC3A TN-1456 REV-1, 2013]. A <label/> element MAY be empty, if the Security Policy supports a default policy. Such a security label SHALL represent the default label for the Security Policy.
/securitylabel/equivalentlabel	OPTIONAL element and its presence, representation and usage depend on the governing security policy that is being enforced.

9.3.3 Output

No output is specified by for this operation.

9.3.4 Errors

When during the processing of *Message* Stanzas errors occur that contain a <securitylabel/> element, a *Message* Stanza of type 'error' SHALL be sent to the originating XMPP entity of the *Messaging* Stanza, but SHOULD NOT contain any content from the violating stanza.

Error-handling, as specified, in [XSF XEP-0258, 2011] SHALL be followed by the XMPP entity that determined the policy violation.

9.4 Future Considerations

This service interface provides the capability for a requesting XMPP entity to obtain a set of valid security labels (Get) from a target XMPP entity that supports this capability. Further research is being undertaken within the NATO Communications and Information Agency (NCI Agency) for providing a trusted labelling architecture covered in [NC3A TN-1480, 2011], [NC3A RD-3320], whereby a NATO metadata-binding service will also provide labelling service interfaces to Get, Set and Verify IA metadata.

It is RECOMMENDED that this service interface be replaced in alignment with the trusted labelling architecture, whereby the NATO metadata-binding service will provide a coherent and consistent labelling service for all services across NATO.

10 WHITEBOARDING SERVICE

10.1 Service Interface

The service interface is not a web service and therefore a WSDL is not appropriate to describe the operations of the service.

The service interface is a NATO-specified service and is defined in [NC3A JCHAT ICD].

10.2 Operations

The Whiteboarding Service operations consist of pushing *Message* Stanzas and using the multi-party messaging service to distribute to the other XMPP entities sharing the whiteboard session. The Whiteboarding Service operations can be categorized as detailed in Table 26.

Table 26
List of Whiteboarding Service operations

Operation	Reference
`Create`	[NC3A JCHAT ICD]
`Update`	[NC3A JCHAT ICD]
`Delete`	[NC3A JCHAT ICD]
`Select`	[NC3A JCHAT ICD]
`Unselect`	[NC3A JCHAT ICD]
`Synchronize`	[NC3A JCHAT ICD]

10.3 Messages

10.3.1 Data types

All Whiteboard Service operation messages SHALL be wrapped in a *Message* Stanza whereby:

- The '`from`' attribute is the *JabberID* of the sending XMPP entity.
- The '`to`' attribute SHALL be the *JabberID* of the target XMPP entity.
- The '`type`' attribute SHALL be set to '`groupchat`' (as specified in Section 7.3.4.1).

10.3.2 Create input

The operation message SHALL contain a child `<x/>` element that is qualified by the namespace '`urn:int:nato:nc3a:xmpp:geowhiteboard`'.

The `<x/>` element specification for this operation is described in Table 27.

Table 27
X elements specification for create operation message input

Element	Notes
`/x/create`	REQUIRED element that informs the target XMPP entity that a new item is to be added to the whiteboard.
`/x/create:id`	REQUIRED attribute that represents the unique identifier of the item that is to be created on the whiteboard. The format of the unique identifier SHALL be: *JabberID* of XMPP entity hosting the whiteboard, followed by a '/', followed by the

	unique identifier representing the whiteboard, followed by a '#', followed by a unique identifier representing the item to be created. An example is: whiteboard.example.com/whiteboard#4.
/x/create/nvg	REQUIRED element. The element SHALL be conformant with the [TIDE NVG, 2008] specification for representing the item that is being created.

10.3.3 Update input

The operation message SHALL contain a child <x/> element that is qualified by the namespace 'urn:int:nato:nc3a:xmpp:geowhiteboard'.

The <x/> element specification for this operation is described in Table 28.

Table 28
X elements specification for update operation message input

Element	Notes
/x/update	REQUIRED element that informs the target XMPP entity that an item is to be modified on the whiteboard.
/x/update:id	REQUIRED attribute that represents the unique identifier of the item that is to be modified on the whiteboard. The format of the unique identifier SHALL be: *JabberID* of XMPP entity hosting the whiteboard, followed by a '/', followed by the unique identifier representing the whiteboard, followed by a '#', followed by a unique identifier representing the item to be modified. An example is: whiteboard.example.com/whiteboard#4.
/x/update/nvg	REQUIRED element. The element SHALL be conformant with the [TIDE NVG, 2008] specification for representing the item that is being modified.

10.3.4 Delete input

The operation message SHALL contain a child <x/> element that is qualified by the namespace 'urn:int:nato:nc3a:xmpp:geowhiteboard'.

The <x/> element specification for this operation is described in Table 29.

Table 29
X elements specification for delete operation message input

Element	Notes
/x/delete	REQUIRED element that informs the target XMPP entity that an item is to be deleted from the whiteboard.
/x/delete:id	REQUIRED attribute that represents the unique identifier of the item that is to be deleted from the whiteboard. The format of the unique identifier SHALL be: *JabberID* of XMPP entity hosting the whiteboard, followed by a '/', followed by the

	unique identifier representing the whiteboard, followed by a '#', followed by a unique identifier representing the item to be deleted. An example is: whiteboard.example.com/whiteboard#4.

10.3.5 Select input

The operation message SHALL contain a child <x/> element that is qualified by the namespace 'urn:int:nato:nc3a:xmpp:geowhiteboard'.

The <x/> element specification for this operation is described in Table 30.

Table 30
X elements specification for select operation message input

Element	Notes
/x/select	REQUIRED element that informs the target XMPP entity that an item is to be selected from the whiteboard.
/x/select:id	REQUIRED attribute that represents the unique identifier of the item that is to be selected from the whiteboard. The format of the unique identifier SHALL be: *JabberID* of XMPP entity hosting the whiteboard, followed by a '/', followed by the unique identifier representing the whiteboard, followed by a '#', followed by a unique identifier representing the item to be selected. An example is: whiteboard.example.com/whiteboard#4.

10.3.6 Unselect input

The operation message SHALL contain a child <x/> element that is qualified by the namespace 'urn:int:nato:nc3a:xmpp:geowhiteboard'.

The <x/> element specification for this operation is described in Table 31.

10.3.7 Synchronize input

The operation message SHALL contain a child <x/> element that is qualified by the namespace 'urn:int:nato:nc3a:xmpp:geowhiteboard'.

The <x/> element specification for this operation is described in Table 32.

This operation results in a series of Create operations, whereby the current state of the whiteboard is reflected at the XMPP entity's local whiteboard that generated the Synchronize operation.

Table 31
X elements specification for unselect operation message input

Element	Notes
/x/unselect	REQUIRED element that informs the target XMPP entity that an item is to be unselected from the whiteboard.

/x/unselect:id	REQUIRED attribute that represents the unique identifier of the item that is to be unselected from the whiteboard. The format of the unique identifier SHALL be: *JabberID* of XMPP entity hosting the whiteboard, followed by a '/', followed by the unique identifier representing the whiteboard, followed by a '#', followed by a unique identifier representing the item to be unselected. An example is: whiteboard.example.com/whiteboard#4.

Table 32
X element specification for synchronize operation message input

Element	Notes
/x/synchronize	REQUIRED element that SHALL be empty.

10.3.8 Output

No output is specified for this service interface.

10.4 Errors

When errors occur during the processing of *Message* Stanzas, a *Message* Stanza of type 'error' SHALL be sent to the originating XMPP entity of the *Message* Stanza.

Error types and error conditions that SHALL be supported are specified in Tables 13 and 14 of [NCIA TR/2013/SPW008423/36, 2014].

11 STRUCTURED DATA FORM SERVICES

11.1 Service Interface

The *Service* interface is not a web service and therefore a WSDL is not appropriate to describe the operations of the service.

The *Service* interface is defined by the XSF XMPP XEPs specified in [XSF XEP-0004, 2007], [XSF XEP-0030, 2008], [XSF XEP-0068, 2011], [XSF XEP-0080, 2009], [XSF XEP-0122, 2004] and [XSF XEP-0141, 2005].

The Structured Data Form Service provides a capability for:

- An XMPP entity to retrieve a form template
- An XMPP entity to submit and validate a form.

The submission and distribution of a form is provided for by the data transport capabilities specified by the XMPP Multi-Party Messaging Service (see Chapter 7) or specified by the Notification Service (see Chapter 8).

11.2 Operations

A compliant XMPP entity SHALL publicize support for the Structured Data Form Service by returning, in the Service Discovery Service Request (specified in Section 6.3.2.2) `var` attribute of the `feature` element, support for the following features:

- 'http://tridsys.com/forms'

- 'jabber:x:data'.

The Structured Data Form Service operations are categorized and described in Table 33.

11.3 Discover Operation

The Discover operation SHALL follow the specifications detailed in Chapter 6 for discovering the form templates hosted by an XMPP entity.

11.3.1 Input

The Discover operation inputs can be categorized into two groups:

1) Discover Service Location Request – *IQ* Stanza specified in Section 6.3.1.1
2) Discover Service Information Request – *IQ* Stanza specified in Section 6.3.1.3.

11.3.2 Output

The Discover operation outputs can be categorized into two groups:

1) Discover Service Location Response – *IQ* Stanza specified in Section 6.3.2.1
2) Discover Service Information Response – *IQ* Stanza specified in Section 6.3.2.2.

Each `<item/>` element that is included as a child element of the *IQ* Stanza represents a form template. The structure of the `<item/>` element is explained in Table 34.

Table 33
List of Structured Data Form Service operations

Operation	Reference
Discover	The operation provides a capability to query an XMPP entity that provides this service to determine the forms that are available. The operation is specified in [XSF XEP-0030, 2008].
Get	The operation provides the capability for an XMPP entity to request a form. The form template SHALL be compliant with [XSF XEP-0004, 2007] and in conformance with [XSF XEP-0068, 2011]. The operation MAY be compliant with [XSF XEP-0141, 2005] for providing hints to an XMPP entity rendering the form for data layout.
Submit	The operation for submitting a form depends on the XMPP transport mechanism that the form must be submitted over. The operation SHALL support one of the following XMPP transport Mechanisms: Multi-Party Messaging Service as specified in Section 7.3.4.1 or Notification Service as specified in Section 8.4.1.1. The operation MAY support the One-to-One Messaging Service (as specified in Chapter 5) as an XMPP transport mechanism. Prior to the form being submitted this operation SHOULD validate the form compliant with [XSF XEP-0122, 2004].

Table 34
Attributes contained in item element for the discover operation response

Element	Comments
/item	REQUIRED element for each form template that is hosted by the XMPP entity.
/item:jid	REQUIRED attribute that identifies the *JabberID* for the form template. The *JabberID* will contain the form name, followed by the @, followed by the XMPP entity's *JabberID* hosting the Structured Data Form Service.
/item:name	REQUIRED attribute containing the form name to be displayed to the XMPP operator.
/item:node	REQUIRED attribute that SHALL represent the version number of the form.

11.4 GET Operation

The operation provides the capability for an XMPP entity to request a form.

The Structured Data Form Service SHOULD support data forms as specified in [XSF XEP-0004, 2007] as a generic data description format that can be used for dynamic forms generation and data-modeling in a variety of circumstances.

The namespace for data forms SHALL be 'jabber:x:data' , where the root element is an <x/> element.

The Structured Data Form Service SHOULD support field standardization for data forms as specified in [XSF XEP-0068, 2011] to standardize field variables used in the context of 'jabber:x:data' forms.

The Structured Data Form Service MAY support data forms layout as specified in [XSF XEP-0141, 2005], which is an extension to the XMPP data forms protocol [XSF XEP-0004, 2007].

The namespace for data forms layout SHALL be 'http://jabber.org/protocol/xdata-layout'.

The Structured Data Form Service SHOULD support data forms validation as specified in [XSF XEP-0122, 2004] to specify additional validation guidelines related to a form, such as validation of standard XML data types, application-specific data types, value ranges and regular expressions.

The namespace for data forms layout SHALL be 'http://jabber.org/protocols/xdata-validate'.

11.4.1 Input

A Get operation request SHALL be wrapped in an *IQ* Stanza whereby:

- The 'type' attribute SHALL be 'get'.
- The 'from' attribute is the *Full JID* of the requesting XMPP entity.
- The 'to' attribute SHALL be the *JabberID* of the Structured Data Form Service XMPP entity.

The operation request SHALL contain an empty <query/> element that is qualified by the namespace 'http://tridsys.com/forms'.

11.4.2 Output

A Get operation response SHALL be wrapped in an *IQ* Stanza whereby:

- The 'type' attribute SHALL be 'result'.
- The 'from' attribute is the *JabberID* of the Structured Data Form Service XMPP entity.
- The 'to' attribute SHALL be the *Full JID* of the requesting XMPP entity.

The operation response SHALL contain a <query/> element, that MAY be qualified by the namespace 'http://tridsys.com/forms', and SHALL contain a <x/> child element that SHALL be qualified by the namespace 'jabber:x:data'.

The <x/> elements specific to the response are described in Table 35.

11.4.3 Errors

If a form cannot be found an *IQ* Stanza with type 'error' SHALL be returned with item-not-found error specified in the <error/> element.

11.5 SUBMIT Operation

The operation provides the capability for an XMPP entity to submit a form.

The Structured Data Form Service SHALL support submitting data forms using one of the XMPP data transport mechanisms specified in Sections 7.3.4.1 or 8.4.1.1.

Table 35
Elements and Attributes Data Forms specification for the Get operation

Element	Comments
/x:type	REQUIRED attribute and the value SHALL be 'form'.
/x/instructions	OPTIONAL element that specifies natural-language instructions to be followed by the form-submitting entity.
/x/title	OPTIONAL element that specifies the label for the form.
/x/field	A data form of type 'form' SHOULD contain at least one <field/> element.
/x/field:var	Uniquely identifies the field in the context of the form. If the <field/> element type is anything other than 'fixed' (see /x/field:type below), it SHALL possess a 'var' attribute; if it is 'fixed', it MAY possess a 'var' attribute.
/x/field:label	OPTIONAL attribute that defines a human-readable name for the field.
/x/field:type	Defines the data type of the field data. Each <field/> element SHOULD possess a 'type' attribute that defines the data type of the field data. The value for this attribute SHALL be one of the following: boolean fixed hidden jid-multi

	jid-single list-multi list-single text-multi text-private text-single. (Reference is made to [XSF XEP-0004, 2007] for an explanation of these values)
/x/field/desc	OPTIONAL element that provides an explanatory description of the field.
/x/field/required	OPTIONAL element. If the element is present it SHALL be an empty element.
/x/field/value	Defines the default value for the field in a data form of type 'form'. Fields of type 'list-multi', 'jid-multi', 'text-multi', and 'hidden' MAY contain more than one <value/> element; all other field types SHALL NOT contain more than one <value/> element.
/x/field/option	This is required if the field is of type 'list-single' or 'list-multi'. If the type is not those values, there SHALL NOT be an <option/> element present.
/x/field/option:label	OPTIONAL attribute that defines a human-readable name.
/x/field/option:value	REQUIRED attribute and there SHALL be only one <value/> element present.

A Structured Data Form Service MAY support submitting data forms using the One-to-One Messaging Service as specified in Chapter 5.

11.5.1 Input

A Submit operation request SHALL be wrapped:

- In an *IQ* Stanza whereby the 'type' attribute SHALL be 'set'.
- In a *Message* Stanza.

The *XML Stanza* is dependent on the data transport mechanism being leveraged to disseminate the submitted data.

The operation request SHOULD contain an <x/> child element that is qualified by the namespace 'jabber:x:data'. The <x/> elements specific to the response are described in Table 35, with the exception that the form 'type' SHALL be 'result'.

An operation request MAY contain an <x/> child element that is qualified by a namespace notherthan 'jabber:x:data'. The list of namespaces that are currently supported by this service interface for the <x/> child element, are:

- 'http://peoc3t.us.army.mil/abcs'.

11.5.2 Output

No output is specified for this service interface, however, the XMPP entity processing the data SHOULD perform Data Forms Validation as specified in [XSF XEP-0122, 2004] and MAY provide an `error` Stanza to the submitting XMPP entity.

12 TIME-SENSITIVE MESSAGING SERVICE

12.1 Service Interface

The *Service* interface is not a web service and therefore a WSDL is not appropriate to describe the operations of the service.

The *Service* interface is defined by the XSF XEPs specified in [XSF XEP-0079, 2005], [XSF XEP-0202, 2009].

The Time-Sensitive Messaging Service provides the capability to ensure delivery of a *Message* Stanza before an absolute point in time. An *XMPP Client* sets the time to live (ttl) for a *Message* Stanza and the *XMPP Server* ensures that the *Message* Stanza is delivered before the *Message* Stanza expires. In the case where the *Message* Stanza cannot be delivered before it expires, the *XMPP Server* issues a notification to the initiating XMPP entity informing the initiating XMPP entity that the *Message* Stanza cannot be delivered as a result of exceeding its ttl.Operations.

A compliant XMPP entity SHALL publicize support for the Time-Sensitive Messaging Service by returning, in the Service Discovery Service Request (specified in Section 6.3.2.2) `var` attribute of the `feature` element, support for the following features:

- 'http://jabber.org/protocol/amp'
- 'urn:xmpp:time'.

The Time-Sensitive Messaging Service operations are categorized and described in Table 36.

Table 36
List of Time-Sensitive Messaging Service operations

Operation	Reference
`Discover`	The operation provides a capability to query an XMPP entity that provides this service to determine the actions and conditions that are available. The operation is specified in [XSF XEP-0030, 2008].
`Get`	The operation provides the capability for an XMPP entity to request the local time of an entity. The operation is specified in [XSF XEP-0202, 2009].
`Expire`	The operation provides a capability for expiring a *Message* Stanza if that *Message* Stanza has expired. The operation is specified in [XSF XEP-0079, 2005].

12.2 Discover Operation

The Discover operation SHALL follow the specifications detailed in Chapter 6 for requesting information and items from XMPP entities that support the Time-Sensitive Messaging Service.

12.2.1 Input

The structure of the *IQ* Stanza for requesting information or items of an XMPP entity supporting this *Service* SHALL be conformant with the specifications detailed in Section 6.3.1.3.

The 'node' attribute SHALL contain the value 'http://jabber.org/protocol/amp'.

12.2.2 Output

The structure of the *IQ* Stanza for the responses SHALL be conformant with the specifications detailed in Section 6.3.2.2.

A Service Discovery Service request for actions and conditions relating to this service SHALL contain any of the following features in the var attribute of the feature element of the response (specified in [XSF XEP-0079, 2005]):

- http://jabber.org/protocol/amp?condition=expire-at.

A Service Discovery Service request for actions and conditions relating to this service SHALL contain one or more of the following features in the var attribute of the feature element of the response (specified in [XSF XEP-0079, 2005]):

- http://jabber.org/protocol/amp?action=drop
- http://jabber.org/protocol/amp?action=error
- http://jabber.org/protocol/amp?action=notify.

12.3 Get Operation

The Get operation SHALL follow the specifications detailed in [XSF XEP-0202, 2009] for communicating the local time from XMPP entities that support the Time-Sensitive Messaging Service. This operation supports environments whereby all XMPP entities within the XMPP network are not guaranteed to be synchronized with established time authorities.

12.3.1 Input

An operation request SHALL be wrapped in an *IQ* Stanza whereby:

- The 'type' attribute SHALL be 'get'.
- The 'to' attribute SHALL be the *JabberID* of the XMPP entity that the local time is being requested from.

The operation request SHALL contain a <time/> element that is qualified by the namespace 'urn:xmpp:time'.

12.3.2 Output

An operation response SHALL be wrapped in an *IQ* Stanza whereby the 'type' attribute SHALL be 'result'.

The operation response SHALL contain a <time/> element qualified by the namespace 'urn:xmpp:time'.

The <time/> elements specific to the response are described in Table 37.

Table 37
REQUIRED elements for Get operation

Element	Comments
/time/tzo	REQUIRED element representing the XMPP entity's numeric time zone offset from UTC.
/time/utc	REQUIRED element that represents the UTC time according to the XMPP entity.

12.4 Expire Operation

The Expire operation SHALL follow the specifications detailed in [XSF XEP-0079, 2005] for expiring *Message* Stanzas that have exceeded their time-to-live value.

12.4.1 Input

The input to this operation is a Message Stanza as specified in:

- Section 5.3.5
- Section 7.3.4.1
- Section 7.3.3.8.

12.4.2 Output

In the case where the *Message* Stanza has expired, the output of this operation is dependent on the 'action' value that has been specified in the *Message* Stanza. Table 38 specifies the output to this operation as a result of the 'action' value if the *Message* Stanza has been determined to have expired.

Table 38
Expire operation output as a result of the value for the 'action' attribute

'action' attribute value	Operation output
alert	Specified in [XSF XEP-0079, 2005] Section 3.4.1
drop	Specified in [XSF XEP-0079, 2005] Section 3.4.2
error	Specified in [XSF XEP-0079, 2005] Section 3.4.3
notify	Specified in [XSF XEP-0079, 2005] Section 3.4.4

In the case where the Message Stanza has not expired, no output is specified for this service interface.

13 REFERENCES

[IETF RFC 1035, 1987]:
Internet Engineering Task Force (on-line) http://www.ietf.org Request for Comments 1035, "DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION", P. Mockapetris, at http://tools.ietf.org/html/rfc1035, November 1987, viewed 5 March 2012.

[IETF RFC 2119, 1997]:
Internet Engineering Task Force (on-line) http://www.ietf.org Request for Comments 2119, "Key words for use in RFCs to Indicate Requirement Levels", S. Bradner, at http://tools.ietf.org/html/rfc2119, March 1997, viewed 5 March 2012.

[IETF RFC 2782, 2000]:
Internet Engineering Task Force (on-line) http://www.ietf.org Request for Comments 2782, "A DNS RR for specifying the location of services (DNS SRV)", A. Gulbrandsen et al., at http://tools.ietf.org/html/rfc2782, February 2000, viewed 5 March 2012.

[IETF RFC 3629, 2003]:
Internet Engineering Task Force (on-line) http://www.ietf.org Request for Comments 3629, "UTF-8, a transformation format of ISO 10646", F. Yergeau, at http://www.ietf.org/rfc/rfc3629, November 2003, viewed 5 March 2012.

[IETF RFC 3749, 2004]:
Internet Engineering Task Force (on-line) http://www.ietf.org Request for Comments 3749, "Transport Layer Security Protocol Compression Methods", T. Dierks, at http://www.ietf.org/rfc/rfc3749, May 2004, viewed 5 March 2012.

[IETF RFC 3920, 2004]:
Internet Engineering Task Force (on-line) http://www.ietf.org Request for Comments 3920, "Extensible Messaging and Presence Protocol (XMPP): Core", P. Saint-Andre, at http://www.ietf.org/rfc/rfc3920, October 2004, viewed 5 March 2012.

[IETF RFC 3921, 2004]:
Internet Engineering Task Force (on-line) http://www.ietf.org Request for Comments 3921, "Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence, P. Saint-Andre, at http://www.ietf.org/rfc/rfc3921, October 2004, viewed 5 March 2012.

[IETF RFC 4121, 2005]:
Internet Engineering Task Force (on-line) http://www.ietf.org Request for Comments 4121, "The Kerberos Version 5 Generic Security Service Application Programming Interface (GSS-API) Mechanism: Version 2", L. Zhu et al., at http://www.ietf.org/rfc/rfc4121, July 2005, viewed 5 March 2012.

[IETF RFC 4422, 2006]:
Internet Engineering Task Force (on-line) http://www.ietf.org Request for Comments 4422, "Simple Authentication and Security Layer (SASL)", A. Melnikov et al., at http://www.ietf.org/rfc/rfc4422, June 2006, viewed 5 March 2012.

[IETF RFC 4505, 2006]:
Internet Engineering Task Force (on-line) http://www.ietf.org Request for Comments 4505, "Anonymous Simple Authentication and Security Layer (SASL) Mechanism", K. Zeilenga, at http://www.ietf.org/rfc/rfc4505, June 2006, viewed 5 March 2012.

[IETF RFC 4616, 2006]:
Internet Engineering Task Force (on-line) http://www.ietf.org Request for Comments 4616, "The PLAIN Simple Authentication and Security Layer (SASL) Mechanism", K. Zeilenga, at http://www.ietf.org/rfc/rfc4616, June 2006, viewed 5 March 2012.

[IETF RFC 4752, 2006]:
Internet Engineering Task Force (on-line) http://www.ietf.org Request for Comments 4752, "The Kerberos V5 ("GSSAPI") Simple Authentication and Security Layer (SASL) Mechanism", A. Melinokv, at http://www.ietf.org/rfc/rfc4752, November 2006, viewed 5 March 2012.

[IETF RFC 5246, 2008]:
Internet Engineering Task Force (on-line) http://www.ietf.org Request for Comments 5246, "The Transport Layer Security (TLS) Protocol Version 1.2", T. Dierks, at http://www.ietf.org/rfc/rfc5246, August 2008, viewed 5 March 2012.

[IETF RFC 6120, 2011]:
Internet Engineering Task Force (on-line) http://www.ietf.org Request for Comments 6120, "Extensible Messaging and Presence Protocol (XMPP): Core", P. Saint-Andre, at http://www.ietf.org/rfc/rfc6120, March 2011, viewed 5 March 2012.

[IETF RFC 6121, 2011]:
Internet Engineering Task Force (on-line) http://www.ietf.org Request for Comments 6121, "Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence", P. Saint-Andre, at http://www.ietf.org/rfc/rfc6121, March 2011, viewed 5 March 2012.

[IETF RFC 6122, 2011]:
Internet Engineering Task Force (on-line) http://www.ietf.org Request for Comments 6122, "Extensible Messaging and Presence Protocol (XMPP): Address Format", P. Saint-Andre, at http://www.ietf.org/rfc/rfc6122, March 2011, viewed 5 March 2012.

[NC3A CES Framework, 2009]:
NATO Consultation, Command and Control Agency ISSC Core Enterprise Services Working Group Document, "NATO Core Enterprise Services Framework v1.2", NC3A, The Hague, Netherlands, April 2009 (NATO/EAPC Unclassified).

[NC3A RD-3153]:
NATO Consultation, Command and Control Agency, Reference Document 3153,"Enterprise Directory Services Service Interface Profile Proposal" (Provisional Title), NC3A Core Enterprise Services Team, NC3A, The Hague, Netherlands, unpublished document dated September 2011 (NATO Unclassified).

[NC3A TN-1456 REV-1, in prep.]:
NATO Consultation, Command and Control Agency, Technical Note 1456 REV-1, "NATO Profile for the XML Confidentiality Label Syntax" (Provisional Title), S. Oudkerk, NC3A, The Hague, Netherlands, in preparation (NATO Unclassified).

[NC3A TN-1480, 2011]:
NATO Consultation, Command and Control Agency Technical Note 1480, "An Incremental Approach to Trusted Labelling in Support of Cross-Domain Information Sharing", S. Oudkerk, G. Lunt, NC3A, The Hague, Netherlands, April 2012 (NATO Unclassified).

[NCIA TR/2013/SPW008423/36, 2014]:
NATO Communications and Information Agency Technical Report 2013/SPW008423/36, "Basic Collaboration Services Service Interface Profile Proposal", A. Ross, M. Laukner, L. Schenkels, NCI Agency, The Hague, Netherlands, January 2014 (NATO Unclassified).

[Unicode Consortium Unicode Version 3.2.0, 2002]:
The Unicode Consortium (on-line) http://www.unicode.org Unicode Version 3.2.0, "Components of the Unicode Standard Version 3.2.0", at http://www.unicode.org/versions/components-3.2.0.html, March 2002, viewed 5 March 2012.

[XSF XEP-0138, 2009]:
XMPP Standards Foundation (on-line) http://xmpp.org , "XEP-0138: Stream Compression", XMPP Extensions, Joe Hildebrand, Peter Saint-Andre, at http://xmpp.org/extensions/xep-0138.html, 27 May 2009, viewed 5 March 2012.

[XSF XEP-0198, 2011]:
XMPP Standards Foundation (on-line) http://xmpp.org , "XEP-0198: Stream Management", XMPP Extensions, Justin Karneges et al., at http://xmpp.org/extensions/xep-0198.html, 29 June 2011, viewed 12 July 2013.

[XSF XEP-0199, 2009]:
XMPP Standards Foundation (on-line) http://xmpp.org , "XEP-0199: XMPP Ping", XMPP Extensions, Peter Saint-Andre, at http://xmpp.org/extensions/xep-0199.html, 03 June 2009, viewed 12 July 2013.

[XSF XEP-0220, 2011]:
XMPP Standards Foundation (on-line) http://xmpp.org , "XEP-0220: Server Dialback", XMPP Extensions, Jeremie Miller et al., at http://xmpp.org/extensions/xep-0220.html, 19 September 2011, viewed 5 March 2012.

[XSF XEP-0288, 2012]:
XMPP Standards Foundation (on-line) http://xmpp.org , "XEP-0288: Bidirectional Server-to-Server Connections", XMPP Extensions, Philipp Hancke et al., at http://xmpp.org/extensions/xep-0288.html, 21 August 2012, viewed 10 July 2013.

[XSF XEP-0302, 2011]:
XMPP Standards Foundation (on-line), http://xmpp.org, "XEP-0302: Compliance Suites 2012", XMPP Extensions, Peter Saint-Andre, at http://xmpp.org/extensions/xep-0302.html, 21 July 2011, viewed 5 March 2012.

14 ABBREVIATIONS

Bi-SC	Bi-Strategic Command
C3	Consultation, command and control
DNS	Domain name system
IA	Information assurance
NISP	NATO Interoperability Standards and Profile
NNEC	NATO Network Enabled Capability
SIP	Service Interface Profile
TCP	Transmission control protocol
UTC	Coordinated universal time
XEP	XMPP extension protocol
XML	Extensible markup language
XMPP	Extensible messaging and presence protocol
XSF	XMPP Standards Foundation

ANNEX 1 – SERVICE INTERFACES FOR XMPP CLIENT AND XMPP SERVER

The extensible messaging and presence protocol (XMPP) architecture is a decentralized client-server architecture that enables the separation between client and server capabilities. It is necessary, therefore, to differentiate which specifications are optional or mandatory to be implemented by *XMPP Clients* and *XMPP Servers* depending on the fundamental features and security detailed in [NCIA TR/2013/SPW008423/36, 2014] and the core and advanced instant messaging service interfaces detailed in this document. Note that a core instant messaging service and an advanced instant messaging service is predicated on the fundamental features and security specified in [NCIA TR/2013/SPW008423/36, 2014].

Table A.1 lists the specifications that are REQUIRED for compliance purposes for an *XMPP Server* and an *XMPP Client* dependent on the categorization of presenting a core or advanced instant messaging service interface. Table A.1 also provides a list of the necessary standards related to text-based collaboration to support C3 interoperability according to the NATO Interoperability Standards and Profiles (NISP).

Table A.1
List *of XMPP Client*-compliant specifications categorized as core and advanced

Specification	*Service* Interface	NISP	XMPP Server			XMPP Client		
			Core	Adv	Support	Core	Adv	Support
[IETF RFC2634, 1999] Enhanced Security Services	Labelling Service			X	O		X	O
[IETF RFC 3749, 2004] Transport Layer Security Protocol Compression Methods	Fundamental Features specified in [NCIA TR/2013/SPW008423/36, 2014]		X	X	O	X	X	O
[IETF RFC4121, 2005] The Kerberos Version 5 Generic Security Service Application Programming Interface (GSS-API) Mechanism: Version 2	Security specified in [NCIA TR/2013/SPW008423/36, 2014]		n/a	n/a	O	X	X	O
[IETF RFC4422, 2006] Simple Authentication and Security Layer (SASL)	Security specified in [NCIA TR/2013/SPW008423/36, 2014]	X	X	X	O	X	X	M
[IETF RFC4505, 2006] Anonymous Simple Authentication and Security Layer (SASL) Mechanism	Security specified in [NCIA TR/2013/SPW008423/36, 2014]		n/a	n/a		X	X	M
[IETF RFC4616, 2006] The PLAIN Simple Authentication and Security Layer (SASL) Mechanism	Security specified in [NCIA TR/2013/SPW008423/36, 2014]		n/a	n/a		X	X	M
[IETF RFC4752, 2006] The Kerberos V5 ("GSSAPI") Simple Authentication and Security Layer (SASL) Mechanism	Security specified in [NCIA TR/2013/SPW008423/36, 2014]	X	n/a	n/a		X	X	O
[IETF RFC5246, 2008] The Transport Layer Security (TLS) Protocol Version 1.2	Security specified in [NCIA TR/2013/SPW008423/36, 2014]	X	X	X	O	X	X	O

Specification	*Service* Interface	NISP	XMPP Server			XMPP Client		
			Core	Adv	Support	Core	Adv	Support
[IETF RFC6120, 2011] Extensible Messaging and Presence Protocol (XMPP): Core	Fundamental Features specified in [NCIA TR/2013/SPW008423/36, 2014]		X	X	M	X	X	M
[IETF RFC6121, 2011] Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence	Presence Service Roster Service One-to-One Messaging Service		X		M	X		M
[IETF RFC6122, 2011] Extensible Messaging and Presence Protocol (XMPP): Address Format	Fundamental Features specified in [NCIA TR/2013/SPW008423/36, 2014]		X	X	M	X	X	M
[XSF XEP-0004, 2007] Data Forms	Structured Data Form Service	X	n/a	n/a			X	O
[XSF XEP-0030, 2008] Service Discovery	XMPP Service Discovery Service	X		X	M		X	O
[XSF XEP-0033, 2004] Extended Stanza Addressing	Presence Service One-to-One Messaging Service	X	X		O	X		O
[XSF XEP-0045, 2008] Multi-User Chat	Multi-Party Messaging Service	X		X	M		X	O
[XSF XEP-0048, 2007] Bookmarks	Multi-Party Messaging Service			X	O		X	O
[XSF XEP-0053, 2008] XMPP Registrar Function			n/a	n/a		n/a	n/a	
[XSF XEP-0054, 2008] VCard-Temp	Roster Service		X		O	X		O
[XSF XEP-0055, 2009] XMPP Jabber Search	XMPP Service Discovery Service			X	O		X	O

Specification	*Service* Interface	NISP	XMPP Server			XMPP Client		
			Core	Adv	Support	Core	Adv	Support
[XSF XEP-0060, 2010] Publish-Subscribe	Notification Service	X		X	O		X	O
[XSF XEP-0068, 2011] Field Standardization for Data Forms	Structured Data Form Service		n/a	n/a			X	O
[XSF XEP-0079, 2005] Advanced Message Processing	One-to-One Messaging Service Multi-Party Messaging Service Time-Sensitive Messaging Service	X	X	X	O	X	X	O
[XSF XEP-0080, 2009] User Location	Multi-Party Messaging Service Notification Service		n/a	n/a			X	O
[XSF XEP-0082, 2003] XMPP Date and Time Profiles	Multi-Party Messaging Service Notification Service	X		X	O		X	O
[XSF XEP-0122, 2004] Data Forms Validation	Structured Data Form Service	Xe[4]	n/a	n/a			X	O
[XSF XEP-0127, 2004] Common Alerting Protocol (CAP) Over XMPP	Multi-Party Messaging Service Notification Service		n/a	n/a			X	O
[XSF XEP-0138, 2009] Stream Compression	Fundamental Features specified in [NCIA TR/2013/SPW008423/36, 2014]	X	X	X	O	X	X	O

[4] Identified in the NISP as an emerging standard.

Specification	*Service* Interface	NISP	XMPP Server			XMPP Client		
			Core	Adv	Support	Core	Adv	Support
[XSF XEP-0141, 2005] Data Forms Layout	Structured Data Form Service		n/a	n/a			X	O
[XSF XEP-0198, 2011] Stream Management	Fundamental Features specified in [NCIA TR/2013/SPW008423/36, 2014]	X	X	X	O	X	X	O
[XSF XEP-0199, 2009] XMPP Ping	Fundamental Features specified in [NCIA TR/2013/SPW008423/36, 2014]	Xe[5]	X	X	O	X	X	O
[XSF XEP-0202, 2009] Entity Time	Time-Sensitive Messaging Service	X		X	O		X	O
[XSF XEP-0203, 2009] Delayed Delivery	Multi-Party Messaging Service Notification Service			X	O		X	O
[XSF XEP-0220, 2011] Server Dialback			X	X	M	n/a	n/a	
[XSF XEP-0256, 2009] Last Activity in Presence	Multi-Party Messaging Service			X	O		X	O
[XSF XEP-0258, 2011] Security Labels in XMPP	Labelling Service			X	O		X	O

[5] Identified in the NISP as an emerging standard.