# XEP-0512: XMPP as Interpretive Dance

Guus der Kinderen
mailto:guus.der.kinderen@gmail.com
xmpp:guus.der.kinderen@igniterealtime.org

2026-04-01
Version 5.6.7.8

| Status | Type | Short Name |
| --- | --- | --- |
| Active | Humorous | dance |

This document defines a method for representing XMPP communications through the medium of interpretive dance. By mapping core protocol elements to specific physical movements, it allows expressive, low-bandwidth, and audience-friendly implementations of XMPP suitable for artistic performances, theatrical demonstrations, and deeply confusing hackathons.

# Legal

## Copyright

This XMPP Extension Protocol is copyright © 1999 – 2024 by the XMPP Standards Foundation (XSF).

## Permissions

Permission is hereby granted, free of charge, to any person obtaining a copy of this specification (the ”Specification”), to make use of the Specification without restriction, including without limitation the rights to implement the Specification in a software program, deploy the Specification in a network service, and copy, modify, merge, publish, translate, distribute, sublicense, or sell copies of the Specification, and to permit persons to whom the Specification is furnished to do so, subject to the condition that the foregoing copyright notice and this permission notice shall be included in all copies or substantial portions of the Specification. Unless separate permission is granted, modified works that are redistributed shall not contain misleading information regarding the authors, title, number, or publisher of the Specification, and shall not claim endorsement of the modified works by the authors, any organization or project to which the authors belong, or the XMPP Standards Foundation.

## Warranty

## NOTE WELL: This Specification is provided on an ”AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. ##

## Liability

In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall the XMPP Standards Foundation or any author of this Specification be liable for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising from, out of, or in connection with the Specification or the implementation, deployment, or other use of the Specification (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if the XMPP Standards Foundation or such author has been advised of the possibility of such damages.

## Conformance

This XMPP Extension Protocol has been contributed in full conformance with the XSF’s Intellectual Property Rights Policy (a copy of which can be found at <https://xmpp.org/about/xsf/ipr-policy> or obtained by writing to XMPP Standards Foundation, P.O. Box 787, Parker, CO 80134 USA).

# Contents


**1 Introduction** **1**

**2 Requirements** **1**

**3 Basic Architecture** **1**
- 3.1 Message Stanzas . . . 1
- 3.2 Presence Stanzas . . . 2
- 3.3 IQ Stanzas . . . 2
- 3.4 Example Session . . . 2

**4 Transport** **2**

**5 Internationalization** **3**

**6 Security Considerations** **3**

**7 IANA Considerations** **3**

**8 XMPP Registrar Considerations** **3**
- 8.1 Protocol Namespaces . . . 3
- 8.2 Service Discovery Identities . . . 3

**9 Acknowledgements** **4**

**10 XML Schemas** **4**

# 1 Introduction

Extensible Messaging and Presence Protocol (XMPP) is traditionally implemented in code, using XML stanzas transmitted over TCP/IP. However, such implementations lack *artistic flair* and are poorly suited for environments with no network connectivity but ample interpretive energy.
This XEP defines how to implement XMPP entirely through human movement. By encoding message, presence, and iq stanzas into gestures, jumps, and rhythmic flourishes, XMPP can finally achieve its destiny as both a communication protocol and a performative art form.

# 2 Requirements

An interpretive XMPP implementation MUST include:

- At least two dancers (a *client* and a *server*).
- A dance floor of sufficient size to represent the stanza queue.
- Background music approximating the rhythm of XML parsing (e.g., in 4/4 time).
- Optional: audience members to act as routers, federation gateways, or emotionally confused observers.

**This document expires when all participants are too exhausted to continue.**

# 3 Basic Architecture

Each XMPP entity is represented by a dancer. Stanzas are represented as dance motifs. Transmission occurs when a dancer performs a motif directed toward another dancer.

## 3.1 Message Stanzas

- <message type='chat'> - A gentle twirl followed by direct eye contact.
- <message type='groupchat'> - The dancer spins outward, gesturing inclusively toward all other dancers.
- <message type='error'> - Collapse to the floor dramatically.
- <message type='headline'> - A short, declarative arm thrust followed by stillness.

### 3.2 Presence Stanzas

- <presence/> - A calm, open posture, hands at sides.
- <show>away</show> - Slow backward steps.
- <show>dnd</show> - Crossed arms with stern gaze.
- <show>xa</show> - Exit the dance floor entirely.
- <status> - text may be mouthed silently or whispered to nearby participants.

### 3.3 IQ Stanzas

IQ ("Inquisitive Questioning") exchanges are structured duets:

- The *requesting* dancer performs a questioning pirouette.
- The *responding* dancer MUST reply with a confirming nod (type='result') or a sharp refusal gesture (type='error').

### 3.4 Example Session

1. Client dancer performs a graceful *connect* leap.
2. Server dancer bows, acknowledging the stream start.
3. Client sends <auth/> by extending both hands in supplication.
4. Server replies with <success/> by raising one arm triumphantly.
5. Both dancers proceed into a spontaneous <message/> exchange sequence until network fatigue sets in.

Note: The dance steps illustrated in the example session are non-normative and provided for demonstrative purposes only. Implementers MUST NOT assume that pirouettes, leaps, or interpretive hand-flutters described herein correspond to any formally defined XMPP stanzas, nor that they are executable by untrained engineers. The XSF recognizes that different implementations MAY employ alternative gestures, including but not limited to jazz hands, body rolls, or small interpretive shrugs. Such variations are considered interoperable provided all participants agree on tempo and dramatic intent.

## 4 Transport

All communication occurs via line-of-sight. Dancers MUST NOT rely on TCP/IP, though rhythmic clapping MAY be used as an unreliable transport layer for synchronization.

# 5 Internationalization

Gestures SHOULD be culturally sensitive. For example, <presence type='subscribe'/> should not involve invasive proximity in cultures where personal space is highly valued.

# 6 Security Considerations

End-to-end encryption MAY be achieved by performing in total darkness.
TLS handshakes can be mimed by elaborate finger-interlocking rituals.
SASL negotiation involves mutual nodding until both dancers feel emotionally authenticated.

# 7 IANA Considerations

This document requires no interaction with the Internet Assigned Numbers Authority (IANA) [1]. Attempting to assign a formal IANA registry entry to gestures, pirouettes, or dramatic pauses would likely result in specification drift, spontaneous applause, or outright audience revolt. IANA involvement is therefor unnecessary, and frankly, unsafe.

# 8 XMPP Registrar Considerations

## 8.1 Protocol Namespaces

Upon advancement of this document to a status of Draft, the XMPP Registrar [2] includes the following namespace in its registry of protocol namespaces (see <https://xmpp.org/registrar/namespaces.html>):

- urn:xmpp:dance:0

## 8.2 Service Discovery Identities

Upon advancement of this document to a status of Draft, the XMPP Registrar [3] shall add a category of 'dance' to its registry of service discovery identities, with one associated types:

[1] The Internet Assigned Numbers Authority (IANA) is the central coordinator for the assignment of unique parameter values for Internet protocols, such as port numbers and URI schemes. For further information, see <http://www.iana.org/>.

[2] The XMPP Registrar maintains a list of reserved protocol namespaces as well as registries of parameters used in the context of XMPP extension protocols approved by the XMPP Standards Foundation. For further information, see <https://xmpp.org/registrar/>.

[3] The XMPP Registrar maintains a list of reserved protocol namespaces as well as registries of parameters used in the context of XMPP extension protocols approved by the XMPP Standards Foundation. For further information, see <https://xmpp.org/registrar/>.

'interpretive', described as "Entity capable of sending/receiving XMPP stanzas via physical movement rather than XML serialization."

## 9 Acknowledgements

The author would like to thank the XMPP Standards Foundation for maintaining a sense of humor, and every dancer who has ever debugged XML with their body.

## 10 XML Schemas

```
<?xml version='1.0' encoding='UTF-8'?>
<!--
    XEP-XXXX: XMPP as Interpretive Dance
    XML Schema Definition (non-normative, rhythmically flexible)
-->

  <!-- Root element for an interpretive stanza -->
  <xs:element name='performance'>
    <xs:complexType>
      <xs:sequence>
        <xs:element ref='dancer' maxOccurs='unbounded'/>
      </xs:sequence>
      <xs:attribute name='tempo' type='xs:string' default='moderato'/>
      <xs:attribute name='time-signature' type='xs:string' default='
          4/4'/>
    </xs:complexType>
  </xs:element>

  <!-- Definition of a dancer -->
  <xs:element name='dancer'>
    <xs:complexType>
      <xs:sequence>
        <xs:element name='move' type='movementType' maxOccurs='
            unbounded'/>
      </xs:sequence>
      <xs:attribute name='jid' type='xs:string' use='required'/>
      <xs:attribute name='role' type='xs:string' default='client'/>
      <xs:attribute name='attitude' type='xs:string' default='
          expressive'/>
    </xs:complexType>
  </xs:element>

  <!-- Movement primitives -->
  <xs:simpleType name='movementType'>
```

```
    <xs:restriction base='xs:string'>
      <xs:enumeration value='pirouette'/>
      <xs:enumeration value='leap'/>
      <xs:enumeration value='gesture'/>
      <xs:enumeration value='collapse'/>
      <xs:enumeration value='nod'/>
      <xs:enumeration value='twirl'/>
      <xs:enumeration value='dramatic-pause'/>
    </xs:restriction>
  </xs:simpleType>

  <!-- Optional error stanza for failed synchronization -->
  <xs:element name='misstep'>
    <xs:complexType>
      <xs:attribute name='type' type='xs:string' default='minor'/>
      <xs:attribute name='severity' type='xs:integer' default='1'/>
      <xs:attribute name='description' type='xs:string'/>
    </xs:complexType>
  </xs:element>

</xs:schema>
```