

# XMPP as the road to innovation

**Bartłomiej Górny**

[bartlomiej.gorny@erlang-solutions.com](mailto:bartlomiej.gorny@erlang-solutions.com)



# 1.

## INTRODUCTION



## INNOVATION

- ▶ New idea, device or method
- ▶ Not the same as “invention”
  - ▷ Practical implementation of an invention
  - ▷ Often manifested via the engineering process, when the problem being solved is of a technical nature

invention      =>      New service, new way to use IM

innovation      =>      implementation of new service  
                         novel implementation of existing features

## INNOVATION

- ▶ New idea, device or method
- ▶ Not the same as “invention”
  - ▷ Practical implementation of an invention
  - ▷ Often manifested via the engineering process, when the problem being solved is of a technical nature

invention      =>      New service, new way to use IM

innovation      =>      implementation of new service  
                                 novel implementation of existing features

**Extension of a protocol**

## eXtensibility done right

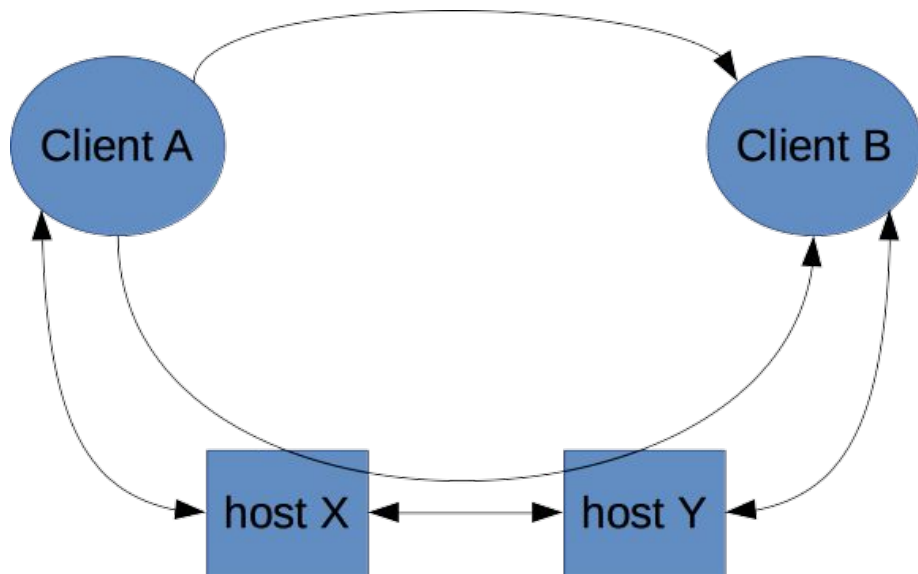
- ▶ Easy to extend
  - ▷ Like Drupal???
- ▶ Kept under control
  - ▷ XSF
  - ▷ Standardisation procedure
  - ▷ Limited number of non-overlapping extensions

# 2.

## XMPP - an overview

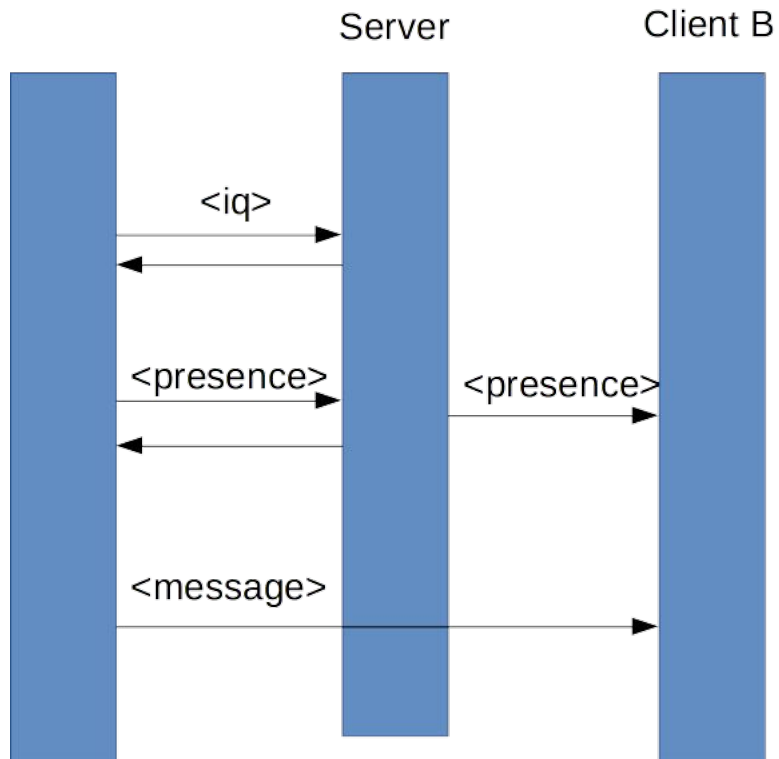


## CLIENT - SERVER - SERVER - CLIENT



## XML “STANZAS”

- ▶ Three types of stanzas
  - ▷ IQ
  - ▷ Presence
  - ▷ message





## STANZA STRUCTURE

- ▶ Top-level element

- ▷ Name
- ▷ Type
- ▷ Id
- ▷ From, to...

```
<message from='bernardo@shakespeare.lit/pda'  
        to='francisco@shakespeare.lit'  
        type='chat'>  
  <body>Who's there?</body>  
</message>
```

- ▶ IQ

- ▷ get, set, result, error

- ▶ Presence

- ▷ (empty), unavailable, probe...

- ▶ Message

- ▷ normal, chat, groupchat...

## INSIDE A STANZA

- ▶ xml elements to be processed by client or server
- ▶ Core protocol defines basic elements in “global” namespace

```
<presence>  
  <show>dnd</show>  
  <status>I'm on a call till 11:30</status>  
</presence>
```

## INSIDE A STANZA

- ▶ an element can introduce its own namespace
- ▶ A namespace identifies a feature being used

```
<presence>  
  <show>dnd</show>  
  <status>I'm on a call till 11:30</status>  
  <mood xmlns="http://jabber.org/protocol/mood">  
    <annoyed/>  
    <text>I'd rather be coding :(</text>  
  </mood>  
</presence>
```

**XEP-0107 "User Mood"**

## INSIDE A STANZA

- ▶ an element can introduce its own namespace
- ▶ A namespace identifies a feature being used

```
<presence>  
  <show>dnd</show>  
  <status>I'm on a call till 11:30</status>  
  <mood xmlns="http://jabber.org/protocol/mood">  
    <annoyed/>  
    <text>I'd rather be coding :(</text>  
  </mood>  
</presence>
```

**XEP-0107 "User Mood"**

- ▶ Every feature/extension/XEP has its own namespace

## NAMESPACE

- ▶ uniquely identifies a feature (XEP)
- ▶ used for determining support (disco)

```
<iq from='conference.wonderland.lit'  
  to='juliet@capulet.com/balcony'  
  type='result'>  
  <query xmlns="http://jabber.org/protocol/disco#info">  
    <feature var="http://jabber.org/protocol/muc"/>  
    <feature var="jabber:iq:register"/>  
    <feature var="vcard-temp"/>  
  </query>  
</iq>
```

XEP-0030 “Service Discovery”

## NAMESPACE

- ▶ Can be nested
- ▶ Greatly eases implementation
- ▶ Stanza payload may be arbitrarily set

```
<message type='chat'>  
  <body>I'll send you something in a while</body>  
</message>
```

```
<message type='chat'>  
  <composing xmlns="http://jabber.org/protocol/chatstates"/>  
</message>
```

**XEP-0085 “Chat State Notifications”**

## DOCUMENTATION

- ▶ RFCs
  - ▷ Basic features of the protocol
- ▶ XEPs
  - ▷ “XMPP Enhancement Proposal”
  - ▷ Standardised format and structure
  - ▷ Extensions, standards, best practices
  - ▷ Verbose
  - ▷ <http://xmpp.org/extensions>
  - ▷ XEP-0001

# 3.

## EXTENDING THE PROTOCOL





## WHERE TO START?

- ▶ Read existing XEPs
  - ▷ Cover a lot of areas
  - ▷ Guaranteed quality
- ▶ Extend an extension?
  - ▷ E.g. “rich chat state”

```
<message type='chat'>  
  <composing xmlns="http://jabber.org/protocol/chatstates">  
  
    </composing>  
</message>
```

## WHERE TO START?

- ▶ Read existing XEPs
  - ▷ Cover a lot of areas
  - ▷ Guaranteed quality
- ▶ Extend an extension?
  - ▷ E.g. “rich chat state”

```
<message type='chat'>  
  <composing xmlns="http://jabber.org/protocol/chatstates">  
    <furiously xmlns="urn:xmpp:typing_speed"/>  
  </composing>  
</message>
```

XEP-XXXX “Rich Chat State Notifications”

## CHOOSE A STANZA

- ▶ For client-server communication - IQ
- ▶ For client-client: message
- ▶ Why not presence?
  - ▷ Too much traffic
  - ▷ Sent to everybody
  - ▷ There are better ways

## EXAMPLE: GEOGRAPHICAL LOCATION

- ▶ we want to provide geographical location data
- ▶ First idea: let's use presence

```
<presence>  
  <location xmlns="urn:xmpp:geolocation">  
    <country>Italy</country>  
    <lat>52.1234</lat>  
    ...  
  </location>  
</presence>
```

## REASONS NOT TO DO THAT

- ▶ location info goes to everybody
- ▶ no way to archive
- ▶ no control over who gets it, can't get rid of it

## REASONS NOT TO DO THAT

- ▶ location info goes to everybody
- ▶ no way to archive
- ▶ no control over who gets it, can't get rid of it
- ▶ Personal Eventing Protocol (XEP-0163)
  - ▷ every user account acts as PubSub node
  - ▷ send location to subscribers only

## REASONS NOT TO DO THAT

- ▶ location info goes to everybody
- ▶ no way to archive
- ▶ no control over who gets it, can't get rid of it
- ▶ Personal Eventing Protocol (XEP-0163)
  - ▷ every user account acts as PubSub node
  - ▷ send location to subscribers only
- ▶ XEP-0080

## THEN

- ▶ free your mind
- ▶ design action sequence and error handling
- ▶ write XEP
  - ▷ for internal use
  - ▷ try to have it approved



# 4.

## REAL-LIFE EXAMPLES



## SIMPLE: SECRET MESSAGE

- ▶ invention: encrypted message, not archived, handled differently by client app
- ▶ innovation: protocol extension

```
<message type='secrechat'>  
  <body>Don't tell anybody, it is a secret!</body>  
</message>
```

## SIMPLE: SECRET MESSAGE

- ▶ invention: encrypted message, not archived, handled differently by client app
- ▶ innovation: protocol extension

```
<message type='secrechat'>  
  <body>Don't tell anybody, it is a secret!</body>  
</message>
```

**NO!!!**

## SIMPLE: SECRET MESSAGE

- ▶ invention: encrypted message, not archived, handled differently by client app
- ▶ innovation: protocol extension

```
<message type='secrechat'>  
  <body>Don't tell anybody, it is a secret!</body>  
</message>
```

**NO!!!**

```
<message type='chat'>  
  <body>Don't tell anybody, it is a secret!</body>  
  <secret xmlns="urn:xmpp:secretmessage"/>  
</message>
```

## ADVANCED: MUC LIGHT

- ▶ subset of XEP-0045
- ▶ simplified, targetting smartphones
- ▶ no presence exchange in rooms
- ▶ more scalable
- ▶ submitted to XSF
- ▶ implemented in MongooseIM
- ▶ used in production on many sites

## OTHER EXAMPLES

- ▶ Inbox
- ▶ Geodistribution (later today)