



www.erlang-projects.org

Mickaël Rémond, Thierry Mallard
<contact@erlang-fr.org>

J-EAI – The first Enterprise Application Server built upon the XMPP standard

EAI tools are traditionally built around proprietary or restricted protocols:

- XMLBlaster is built upon its own XML protocol;
- JMS buses are restricted to a Java-based definition of the world.

However, Enterprise Application Integration is about interoperability and can hardly suffer such limitation.

J-EAI is the first EAI to take the best of both worlds, aiming at building a **standard-based EAI solution**.

XMPP stands for XML Messaging and Presence Protocol. This protocol is an **IETF** (Internet Engineering Task Force standard).

XMPP is an extensible protocol already supported by a community of users defining new features.

Supported distribution and routing algorithms

J-EAI supports several distribution mechanisms:

- **Point-to-point** communication, through the server
- Support **direct point-to-point message exchange** from client to client
- **Channel-based publish & subscribe** mechanism (conforms to JEP-0060)
- **Message pattern-based publish & subscribe** mechanism based on Xpath

Routing features are also outstanding, thanks to an **extensible server-to-server protocol**. Message can be routed between several J-EAI servers, run in different domains and organisations. The XMPP protocol is defined to be distributed across organisation. This is an important feature for cross enterprise or cross-administration data exchanges.

Unmatched EAI features

J-EAI supports:

- **Message queueing**: When a client is offline, messages are queued and can be retrieved later. No messages are lost.
- **Presence and status** of connected application is handled. This is an unprecedented EAI feature allowing application to take decision based on presence or status change of other applications.

Highly-scalable and fault-tolerant

J-EAI core has been built to support:

- **Distribution**: J-EAI can run on a cluster of computers.
- **Fault-tolerance and high-availability**: The application itself is built on the **Open Transaction Platform** (OTP), a framework developed by Ericsson to build fault-tolerant systems. OTP provides features for hot-code upgrade of the software to avoid software interruptions and provide unmatched uptime.
- **Load-Balancing**: A set of cluster nodes can serve the same J-EAI domain.
- **Scalability**: J-EAI is scalable. Scalability tests are made thanks to the **Tsunami** load-testing framework.

J-EAI supports several other enterprise-level features such as:

- LDAP authentication
- External script authentication

XML-based API

J-EAI is **extensible** through a standard XML-based API.

Components, modules and agents defined using this API can be reused as-is on other XMPP-based servers.

Architecture

J-EAI is build upon several existing components:

- **Jabber server:** The core of J-EAI is handled by the **Ejabberd** server. This is the component that gave the J letter in J-EAI name.
- **Connectors / Transformation:** Two standard approaches are proposed:
 - **XSLT** transformation.
 - **OpenAdaptor:** This is a widely-used adaptor and transformation mechanism. It allows connector development in Java language.
- **Publish & susbribe extensions:** Buster logical bus.

Open source and collaborative development

J-EAI is:

- Composed of several Open Source components (Ejabberd, OpenAdaptor, Buster) and working with those component teams to build a J-EAI community.
- Distributed under an Open Source licence.
- Open to external contributors to quickly expand the community.
- Built on open standards like XMPP, XML, XSLT, ...

Notes on Jabber

Currently, the most common use of the XMPP protocol is Instant Messaging.

Regarding logical buses, instant messaging is an highly demanding application that supports real-time collaboration of hundreds to thousands clients.

Jabber is used by several companies to support worldwide instant messaging deployment.

For example, France Telecom is using Jabber as its instant messaging platform for Voilà and Wanadoo services.

This is a proof of Jabber scalability and reliability.

Short-term roadmap

The main J-EAI components are already used in production environment.

Short-term roadmap covers integration development:

- **Buster integration:** transformation into an XMPP component.
- **Administration console** development to support J-EAI core management and message traceability.
- **OpenAdaptor source:** OpenAdaptor Jabber sink already exist but the source module is still missing.

Middle-term roadmap

Several innovative feature are planned to extend J-EAI:

- **Built-in webservice connector**
- **Jabberlang:** Erlang extension to accelerate XMPP based components and agent. Erlang built-in concurrency makes it perfect to get an high-level rapid components development environment.
- **Agent-based workflow:** It will be possible to defined Workflow through a set of external collaborating agents. Several environments are currently studied:
 - Narval
 - eXAT: A complete agent development platform supporting both behaviour and logic definition.

J-EAI supporters

J-EAI is supported by:

- Erlang-projects non-profit organisation

