**Protocolo XMPP**

Arturo Bressán y Federico Vega

**Introducción:**

XMPP, conocido como el Protocolo Extensible de Mensajería y Presencia, es una plataforma de comunicación abierta que abarca una variedad de tecnologías para mensajería instantánea, presencia en línea, chat grupal, llamadas de voz y video, colaboración, intercambio de datos y enrutamiento XML.

Originado en la comunidad de código abierto Jabber, XMPP fue concebido como una alternativa abierta y descentralizada a los servicios de mensajería instantánea cerrados de su época. Su flexibilidad y ventajas sobre estos servicios incluyen:

* **Apertura**: Los protocolos XMPP son gratuitos, de código abierto, accesibles al público y fáciles de entender. Además, existen múltiples implementaciones disponibles en forma de clientes, servidores, componentes de servidor y bibliotecas de código.
* **Estandarización**: Los protocolos fundamentales de XMPP han sido formalizados por el Grupo de Trabajo de Ingeniería de Internet (IETF) como tecnología aprobada para mensajería instantánea y presencia. Las especificaciones de XMPP se han publicado como RFC 3920 y RFC 3921 desde 2004, y la XMPP Standards Foundation continúa desarrollando numerosas extensiones de protocolo XMPP.
* **Consolidación**: La tecnología Jabber/XMPP, desarrollada inicialmente por Jeremie Miller en 1998, ha alcanzado una notable estabilidad. Con cientos de desarrolladores involucrados y decenas de miles de servidores XMPP en funcionamiento en Internet, millones de personas utilizan esta tecnología para la comunicación en tiempo real a través de servicios públicos como Google Talk y otras implementaciones en organizaciones globales.
* **Descentralización**: La arquitectura de red de XMPP sigue un modelo similar al correo electrónico, permitiendo a cualquiera ejecutar su propio servidor XMPP. Esto otorga a individuos y organizaciones un mayor control sobre su experiencia de comunicación.
* **Seguridad**: Los servidores XMPP pueden ser configurados para operar de forma aislada de la red pública, lo que garantiza un nivel de seguridad robusto mediante mecanismos como SASL y TLS, como se especifica en los estándares centrales de XMPP. Además, la comunidad de desarrolladores está activamente comprometida en mejorar la seguridad mediante el cifrado de extremo a extremo.
* **Flexibilidad**: Gracias al formato XML, XMPP es altamente extensible, permitiendo la creación de funciones personalizadas además de los protocolos estándar. Las extensiones comunes se publican en la serie XEP para mantener la interoperabilidad, pero las organizaciones también pueden desarrollar sus propias extensiones privadas si lo desean.
* **Diversidad de Aplicaciones**: XMPP va más allá de la mensajería instantánea, abarcando aplicaciones como administración de redes, distribución de contenido, colaboración, intercambio de archivos, juegos, monitoreo remoto de sistemas, servicios web, middleware ligero y computación en la nube, entre otros.
* **Amplia Adopción**: Una amplia gama de empresas y proyectos de código abierto utilizan XMPP para crear y desplegar aplicaciones y servicios en tiempo real, garantizando que los usuarios no queden "atrapados" en una única tecnología.

Jingle ofrece a los clientes de Jabber una manera de establecer, controlar y finalizar sesiones multimedia. Estas sesiones pueden respaldar una variedad de aplicaciones, como chat de voz, videoconferencia y transferencia de archivos, utilizando diversos métodos de transporte de medios, como TCP, UDP, RTP e incluso XMPP dentro de la misma red. La señalización para iniciar una sesión Jingle se realiza a través de XMPP, mientras que los medios suelen transmitirse directamente de igual a igual o mediante una retransmisión de medios. Jingle proporciona un marco adaptable tanto para tipos de aplicaciones como para transportes de medios. En el caso específico del chat de voz y video, la negociación de Jingle a menudo implica el uso del Protocolo de Transporte en Tiempo Real (RTP) como método de transporte de medios, lo que garantiza la compatibilidad con tecnologías multimedia establecidas, como el Protocolo de Inicio de Sesión (SIP). Además, la señalización Jingle se ha diseñado para ser compatible tanto con SIP como con el Protocolo de Descripción de Sesión (SDP), facilitando la interoperabilidad entre redes XMPP y redes SIP mediante gateways de señalización.

MUC, una extensión de XMPP, facilita el intercambio de información entre múltiples partes de manera similar al Internet Relay Chat (IRC). Permite que varios usuarios de XMPP conversen en el contexto de una sala o canal. Además de las funciones estándar de una sala de chat, como temas e invitaciones, el protocolo establece un sólido sistema de control de la sala. Esto incluye la capacidad de expulsar y prohibir usuarios, designar moderadores y administradores, solicitar membresía o contraseñas para unirse a la sala, entre otras funciones. Debido a que las salas MUC se basan en XMPP, no solo permiten el intercambio de mensajes de texto plano, sino también una amplia variedad de datos XML.

**Breve Historia**

En 1999, Jeremie Miller anunció la creación de Jabber, una tecnología de mensajería instantánea y presencia de código abierto. Durante ese año, se desarrolla rápidamente un servidor de código abierto (jabberd), varios clientes y bibliotecas de código abierto, y protocolos de transmisión XML en tiempo real, junto con mensajería instantánea básica y extensiones de presencia. Estos protocolos básicos, con mejoras y extensiones, continúan utilizándose en la actualidad.

En agosto de ese año, Jeremie promete el apoyo de la comunidad Jabber al proceso de estándares del IETF, en línea con la misión fundacional del proyecto Jabber de fomentar la libertad de conversación y apoyar estándares abiertos e interoperabilidad en las comunicaciones en tiempo real.

En el año 2000, se lanza la versión 1.0 del servidor Jabberd y se estabilizan los protocolos base de Jabber. Además, se introdujo el protocolo de devolución de llamadas del servidor en octubre para evitar la suplantación de direcciones en la red de servidores Jabber en rápido crecimiento.

En 2001, se forma la Jabber Software Foundation (JSF) para coordinar los proyectos de código abierto y entidades comerciales relacionados con Jabber. El objetivo principal de la JSF es gestionar los protocolos utilizados dentro de la comunidad Jabber/XMPP, documentando los protocolos existentes y desarrollando extensiones de protocolo a través de un proceso de estándares abiertos.

En 2002, la JSF presenta un borrador de Internet que documenta los protocolos Jabber al Grupo de Trabajo de Ingeniería de Internet (IETF). En octubre, el IESG aprueba la formación del Grupo de Trabajo XMPP, y la JSF formalmente aporta los protocolos básicos de Jabber al Proceso de Estándares de Internet bajo el nombre de Protocolo Extensible de Presencia y Mensajería (XMPP). En noviembre, se lleva a cabo la primera reunión del Grupo de Trabajo XMPP en IETF 55.

En 2003, el Grupo de Trabajo XMPP finaliza el trabajo de formalización de los protocolos básicos de Jabber como protocolos de presencia y mensajería instantánea aprobados por el IETF. Además, se centran en mejorar la seguridad y la internacionalización de los protocolos base.

En 2004, el IETF publica RFC 3920 y RFC 3921, definiendo la funcionalidad principal de XMPP como estándares propuestos. Tras la publicación de estos RFC, el IETF anuncia la conclusión del Grupo de Trabajo de XMPP, aunque el desarrollo de más extensiones XMPP continúa en la XSF.

Y así sucesivamente, se desarrolla y evoluciona la tecnología XMPP a lo largo de los años, con la participación activa de la comunidad XMPP en eventos, cumbres y programas como Google Summer of Code.

**En resumen:**

XMPP es una tecnología que permite la transmisión de datos XML a través de una red. Surgió de la comunidad de código abierto Jabber en 1999, con el propósito original de ofrecer una alternativa abierta, segura y descentralizada a los servicios de mensajería instantánea dirigidos al consumidor, como ICQ, AIM y MSN. En 2004, estas tecnologías fundamentales fueron estandarizadas por la IETF bajo el nombre de Protocolo Extensible de Mensajería y Presencia (XMPP). Estas tecnologías clave incluyen:

* **Capa de Transmisión XML Base**: Permite la transferencia de datos en formato XML.
* **Cifrado de Canales con Transport Layer Security (TLS)**: Garantiza la seguridad de la comunicación mediante el cifrado de los datos transmitidos.
* **Autenticación Robusta con Capa de Seguridad y Autenticación Simple (SASL)**: Proporciona un sólido sistema de autenticación para garantizar la seguridad de las conexiones.
* **Uso de UTF-8 para Compatibilidad con Unicode**: Permite el manejo de caracteres de cualquier idioma, incluyendo direcciones completamente internacionalizadas.
* **Información de Presencia Incorporada**: Proporciona información en tiempo real sobre la disponibilidad de los usuarios en la red.
* **Suscripciones de Presencia para Autorización Bidireccional**: Permite que los usuarios autoricen mutuamente la visualización de su estado en línea.
* **Listas de Contactos Habilitadas para Presencia ("Listas")**: Facilita la gestión de contactos y la visualización de su disponibilidad en línea.

**Implementación:**

En esta implementación, creamos un sistema de chat utilizando el protocolo XMPP (Extensible Messaging and Presence Protocol). Para lograr esto, hemos utilizado ejabberd como servidor XMPP, Node.js para el backend y HTML/JavaScript para el frontend.

Para el servidor XMPP, elegimos ejabberd debido a su amplia adopción, características robustas y facilidad de configuración. Configuramos ejabberd para permitir conexiones HTTP y WebSocket, lo que nos permite comunicarnos con el frontend a través de solicitudes HTTP y WebSocket.

El backend se implementó utilizando Node.js, un entorno de tiempo de ejecución de JavaScript del lado del servidor. Utilizamos Express.js, un marco web para Node.js, para manejar las solicitudes HTTP del frontend. Implementamos dos rutas en el backend: `/send-message` para enviar mensajes al servidor XMPP y `/get-messages` para recibir mensajes del servidor XMPP.

Para interactuar con ejabberd desde el backend, utilizamos Axios, una biblioteca HTTP basada en promesas. Axios nos permite realizar solicitudes HTTP a ejabberd para enviar y recibir mensajes de chat.

El frontend se desarrolló utilizando HTML para la estructura y JavaScript para la interactividad. Creamos una interfaz de usuario simple con un campo de entrada para el nombre de usuario, un campo de entrada para escribir mensajes y un área para mostrar los mensajes de chat.

Utilizamos JavaScript para capturar los eventos del usuario, como enviar un mensaje o establecer un nombre de usuario. Cuando un usuario envía un mensaje, JavaScript hace una solicitud POST al backend para enviar el mensaje al servidor XMPP. También implementamos una función para recibir mensajes del servidor XMPP y mostrarlos en la interfaz de usuario en tiempo real.

Esta implementación nos proporciona un chat funcional utilizando el protocolo XMPP. La combinación de ejabberd como servidor XMPP, Node.js como backend y HTML/JavaScript como frontend nos permite crear una aplicación de chat en tiempo real que puede ser desplegada y utilizada en una variedad de escenarios.

En conclusión, esta es una implementación muy simple de XMPP que nos permite aprovechar las características robustas de ejabberd para la gestión de mensajes.

**Fuentes:**

<https://xmpp.org/about/technology-overview/>

<https://xmpp.org/about/history/>

<https://gemini.google.com/app/> (Para encontrar links de youtube)

<https://xmpp.org/software/?platform=windows> clientes de xmpp