Omni

System Requirements Specification Team 6

Table of Contents

Introduction

Purpose of this Document

This document outlines the specification for the Omni server and the specification for a frontend and web server that support the Omni server. The frontend and web server are together known as “Omnichat” and are not a core part of the Omni system, but are provided as a reference for others who may want to build an Omni chat interface. Implementations of the Omni server must adhere to the specifications laid out in this document (section 1.3 specifically) to be considered a valid implementation. The intended audience for this document is the CMSC 447 section 2 teaching team.

References

SRSD Template

Umrawal, Abhishek K., et al “System Requirements Specification Document (SRSD) Instructions & Template”

Software Engineering 9

Sommerville, Ian, et al “Software Engineering 9th Edition”, March 2010, https://ifs.host.cs.st-andrews.ac.uk/Books/SE9/index.html

Purpose of the Product

The purpose of this product is to provide a decentralized chat platform where users may sign up on and chat across multiple different servers. This platform will appeal to users who are interested in decentralized technologies, data privacy, platform longevity, and online community spaces.

Fig. 1: Context Diagram

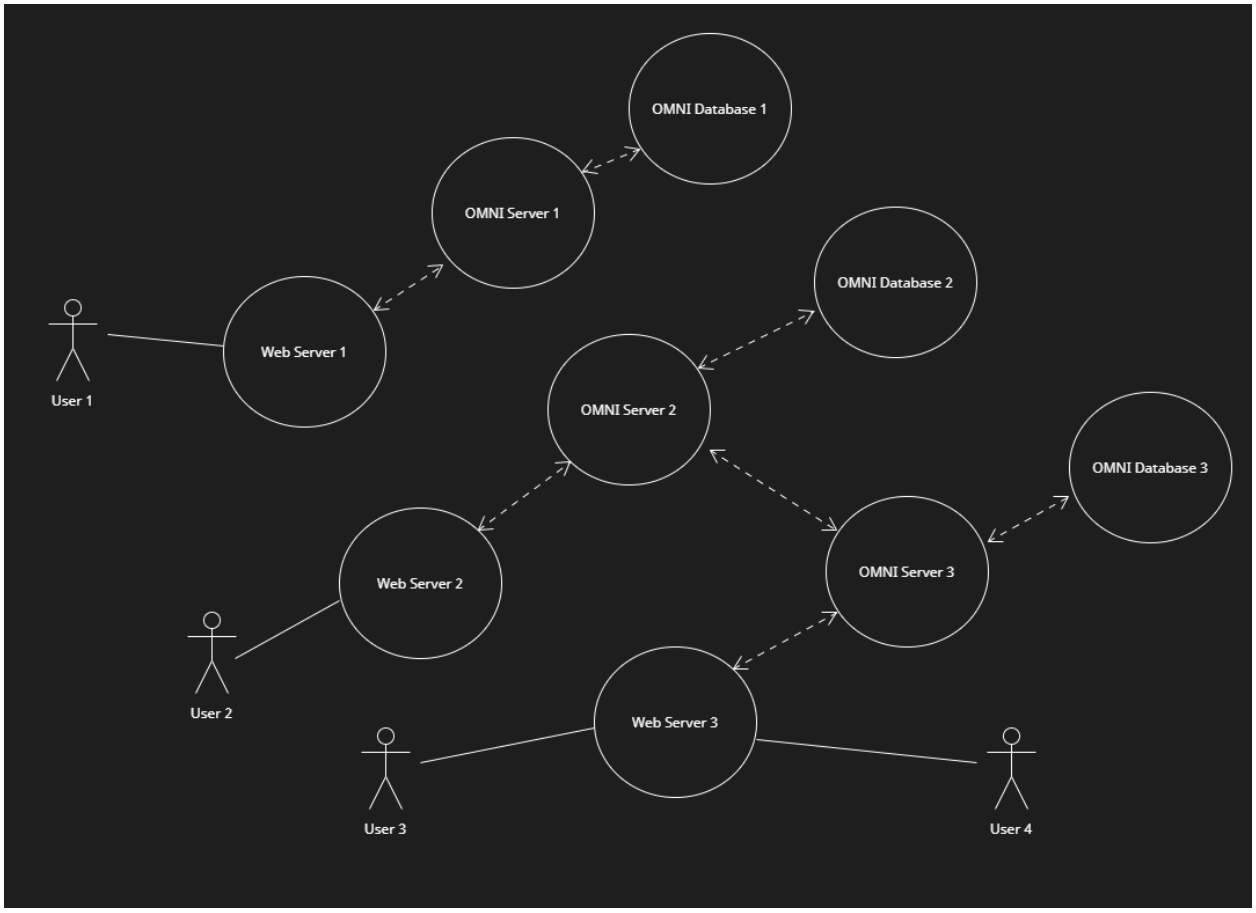

Figure 1 illustrates how the users will interact with a web server that hosts its own Omni server, which is connected to both its own database and possibly other Omni servers. It also portrays how a server could decide not to communicate with another Omni server.

Glossary

Term	Definition
Ban List	A list of users who are not allowed to send messages to or receive messages from the Omni server.
Channel	A virtual space in which messaging takes place.
Channel List	A UI element which contains the channels that the user may access.
Chat Interface	A webpage with a channel list, message pane, message box, and an online users list.
Credentials	A combination of a username and a passphrase. Valid credentials are ones

	which are present in the web server's user account database.
Inform	For one component of the system to pass data to another component, either through an API call or a callback.
Message Box	A UI element which is a text box into which a user may type a message to send.
Message Pane	A UI element which displays past and newly-sent or newly-received messages within a specified channel.
Omnichat	The reference implementation of a frontend and web server that will interface with the Omni server.
Pairing	The process of linking two Omni servers so that messages may be sent between them. Paired Omni servers are "peers." Peers have access to each other's public channel list.
Peer	An Omni server that is paired with another Omni server.
Session Token	A randomly-generated string of characters that is assigned to a user upon logging in. A valid session token is one which is present on the web server. Possession of a valid session token constitutes proof of ownership of a user account.
Time-out List	A list of users who are temporarily not allowed to send messages to the Omni server.

Functional Requirements (1)

Frontend (1.1)

Logged Out Users (1.1.1)

Condition	Action
User visits chat page	Do not load any messages, channels, or other online users, prompt user to log in
User clicks "Login" button	Redirect to login page
User clicks "Create Account" button	Redirect to create account page
User completes signup or login process (session token received from web server)	Redirect to chat page, grant user ability to use the chat interface

Logged In Users (1.1.2)

Condition	Action
User is logged in, user clicks “Logout” button	Inform web server of user logout, clear session, refresh page
User selects a channel	Clear messages pane, retrieve messages in selected channel from cache or from server, populate messages pane. Clear online users list, retrieve online users from cache or from server, populate online users pane
User types a message in the message box	Make the send button visible and send action available
User presses the send button or uses the send action	Inform web server of message send, add message to the message pane, clear the message box
Channel list update received from web server	Update the channel list
Online user list update received from web server	Update the online user list
Message received from web server, message is in the currently selected channel	Add the message to the message pane
Message received from web server, message is in a channel that is not currently selected	Add the message to the local cache of messages, to be displayed if the user chooses to open the channel

Web Server (1.2)

Logged Out Users (1.2.1)

Number	WS-1
Name	Account Creation
Priority	4
Summary	When an account creation request is received from the user, and all fields meet requirements, create the user account and log it in.
Preconditions	

Postconditions	User account stored in database, user session created
Primary Actor	User
Secondary Actors	Web Server
Trigger	User sends account creation request
Main Scenario	1: Create user account, save it in database 2: Create session for user 3: Add user to list of online users 4: Replay with session token
Branching Action	Requested username is already registered : WS-2 Requested passphrase doesn't meet requirements : WS-3

Number	WS-2
Name	Username taken
Priority	3
Summary	When an account creation request is received from the user, and the username is one that has been taken, the server shall reply with "username in use" error.
Preconditions	Wanted username is already registered in the database
Postconditions	Server sends error back that "username is in use"
Primary Actor	User

Number	WS-3
Name	Bad passphrase
Priority	3
Summary	When an account creation request is received from the user, and the passphrase does not meet requirements, the server shall reply with "invalid passphrase" error.
Preconditions	Given passphrase does not meet

	requirements
Postconditions	Server sends back “invalid passphrase” error
Primary Actor	User

Number	WS-4
Name	Logging in
Priority	4
Summary	When a login request is received from the user with valid credentials, the server shall create a session for the user.
Preconditions	User has an account
Postconditions	Session created for user’s account
Primary Actor	User
Secondary Actors	Web Server
Trigger	User sends login request
Main Scenario	1: Create session for user 2: Reply with session token
Branching Action	Invalid credentials : WS-5

Number	WS-5
Name	Invalid login credentials
Priority	4
Summary	When a login request is received from the user with invalid credentials, the server shall send back an authentication error.
Preconditions	Login request has invalid credentials
Postconditions	Reply with authentication error

Number	WS-6
Name	User already logged out

Priority	2
Summary	When a logout request is received from the user but the user is already logged out, the server shall send back a “user is not logged in” error.
Preconditions	User is logged out
Postconditions	Reply with “user is not logged in” error

Number	WS-7
Name	Request info while logged out
Priority	2
Summary	If the user requests certain data while the user is logged out, the server shall send back an authentication error.
Preconditions	User is logged out
Postconditions	Reply with authentication error
Trigger	Message received from user; User requests backlog of messages in channel; User requests list of online users in channel

Logged In Users (1.2.2)

Condition	Action
User requests list of online users in channel, user does not have access to channel	Reply with “invalid channel” error
User requests list of online users in channel, channel is not a valid channel	Reply with “invalid channel” error
User requests list of online users in channel, user has access to channel	Reply with list of online users in channel
Omni server requests a message be sent to a user	Send message to user
Omni server requests a message be sent to a group of users	Send message to all users specified by Omni server

Omni server replies to a channel creation or deletion request with a success message	Send channel list update to online users who have access to the channel
Omni server informs that a peer has created or deleted a channel	Send channel list update to online users who have access to the channel
Omni server informs that a user has gone online or offline	Send online user list update to online users
Omni server informs that a peer has gone online or offline	Send channel list update to online users who have access to the peer
Omni server informs of new pair request received	Send notification to server administrators through frontend
Omni server informs of successful pair	Send notification to server administrators through frontend

Number	WS-8
Name	Logging out
Priority	3
Summary	When a logout request is received from the user, the server shall delete the user's session.
Preconditions	User has an account, and is logged in
Postconditions	User session deleted
Primary Actor	User
Secondary Actors	Web Server
Trigger	User sends logout request
Main Scenario	1: Delete user from list of online users 2: Delete session token 3: Reply with successful logout message
Branching Action	User already logged out : WS-6

Number	WS-9
Name	Receiving messages from the user
Priority	5
Summary	When a message is received from the user, and the user has

	access to the channel, the web server shall forward the message and user ID to the Omni server.
Primary Actor	User
Secondary Actors	Web Server, Omni Server
Trigger	Message received from user
Branching Action	User is logged out : WS-7 User does not have access to channel : WS-10 Channel is not a valid channel : WS-11

Number	WS-10
Name	Invalid channel permissions
Priority	3
Summary	When a message is received from the user, and the user does not have access to the channel, the web server shall reply with “invalid channel” error.

Number	WS-11
Name	Invalid channel
Priority	3
Summary	When a message is received from the user, and the channel is not a valid channel, the web server shall reply with “invalid channel” error.

Number	WS-9
Name	Backlog request
Priority	5
Summary	When the user requests a backlog of messages in a channel from before timestamp *t*, and the user has access to the channel, the web server shall reply with a backlog of the last 50 messages which have time stamps equal to or previous to *t.*
Primary Actor	User

Secondary Actors	Web Server
Trigger	User requests backlog
Branching Action	User is logged out : WS-7 User does not have access to channel : WS-10 Channel is not a valid channel : WS-11

Omni Server (1.3)

Condition	Action
User message received from web server, destination is a private channel	Reply with a list of users to forward the message to, excluding any users in the ban list
User message received from web server, destination is a public channel	Reply with a list of users to forward the message to (excluding any users in the ban list), inform peers of a new message
User message received from web server, user is in the ban list or the time-out list	Reply with an empty array
Web server informs of a user going online/offline	Update list of online users, inform peers of a user list change
Web server requests a list of local channels	Reply with a list of channels hosted on the local Omni server
Web server requests a list of remote channels	Reply with a list of channels hosted on peer Omni servers
Web server requests a list of all channels	Reply with a list of channels hosted locally or on peer Omni servers
Web server requests a list of online users in a channel	Reply with a list of users online with access to the channel specified
Web server requests a list of online users in an invalid channel	Reply with “invalid channel” error
Web server requests a list of peer servers	Reply with a list of paired Omni servers
Web server requests a list of sent messages in an invalid channel	Reply with “invalid channel” error
Web server requests a list of *n* sent messages in a channel from before timestamp *t*	Reply with an array of messages in the specified channel, starting from the timestamp specified by the web server and

	going back *n* messages
Web server requests a ban of a user	Add user to the ban list
Web server requests a time-out of a user for *s* seconds	Add user to the time-out list, schedule the user to be removed from the time-out list at a time *s* seconds in the future
Web server requests creation of a new channel, channel name already exists in the list of local channels	Reply with “channel exists” error
Web server requests creation of a new channel, channel name does not exist in the list of local channels, channel is private	Create a new channel object internally, reply with a success message
Web server requests creation of a new channel, channel name does not exist in the list of local channels, channel is public	Create a new channel object internally, reply with a success message, send channel list update to peers
Web server requests deletion of an invalid channel	Reply with “invalid channel” error
Web server requests deletion of a private channel	Delete channel object internally, purge all messages from said channel from database, reply with a success message
Web server requests deletion of a public channel	Delete channel object internally, purge all messages from said channel from database, send channel list update to peers, reply with a success message
Web server requests purge of all messages in a private channel	Purge all messages from the specified channel from database
Web server requests purge of all messages in a public channel	Purge all messages from the specified channel from database, inform peers of a channel purge
Web server requests purge of all messages in an invalid channel	Reply with “invalid channel” error
Peer informs of a new message in a channel	Request the web server to send the message to local users with access to the channel
Peer informs of a channel list change	Update the cached channel list for the peer, inform web server of peer channel list change
Peer informs of a channel purge	Purge the local copy of all messages in the

	specified channel
Peer informs of an online user list change	Update the cached list of online users for the peer, inform web server of online user list change
Peer informs that it has gone online or will go offline	Update list of online peers, inform web server of peer status change
Peer sends unpair	Delete all data related to the peer, including all channels, messages, and online users, from the cache/database, and inform web server of peer going offline
Peer accepts pair request	Send list of channels and list of online users to peer, inform web server of successful pair, inform web server of online peer, inform web server of peer channel list and online users
Omni server is brought online	Inform peers that this server is now online
Omni server is being shut down	Inform peers that this server is now offline
Pair request received from Omni server	Inform web server that a pair request was received
Web server requests unpairing with peer	Inform peer of unpair, delete all data related to the peer, including all channels, messages, and online users, from the cache/database, and inform web server of peer going offline
Web server accepts peer request	Inform peer of pair request acceptance, send peer list of channels and online users, await list of channels and online users from peer

Non-Functional Requirements (2)

Number	1
Name	Editor for Omni development
Priority	1
Description	The development team shall use Visual Studio Code while working on Omni.

Number	2
Name	Software for creating UML diagrams
Priority	1
Description	UML diagrams for Omni shall be written using the UMLet extension for Visual Studio Code.

Number	3
Name	Programming Language for Omni server and web server backend
Priority	3
Description	The Omni Server and web server backend shall be written in Node.js.

Number	4
Name	Language for the Omni frontend
Priority	3
Description	The frontend shall be written in HTML, CSS, and JavaScript.

Number	5
Name	Source control for the Omni source code
Priority	4
Description	Changes to the Omni source code shall be tracked using Git and synced with the GitHub repositories.

Number	6
Name	Style of code for the Omni source code
Priority	2
Description	Code written by the development team shall follow the code style guide.

Number	7
Name	Development approach for the project
Priority	2
Description	The development approach for the Omni project shall be a variation of Agile.

Number	8
Name	Browsers that can use Omnichat
Priority	4
Description	Omnichat shall be compatible with Chrome and Firefox at minimum.

Number	9
Name	Maximum message transfer failure rate
Priority	4
Description	Messages sent by users of Omni shall not fail to be sent more often than one time in one thousand.

Number	10
Name	Maximum message delay
Priority	4
Description	The delay between a message being sent by one user and received by another shall be no greater than 5 seconds.

User Interface

See “User Interface Design Document for Omni”.

Deliverables

Copies of each of the following.

- Systems Requirement Specification (10/20/22)
- System Design Document (10/28/22)
- User Interface Design Document
- User Manual
- Administrator Manual
- Copies of all Biweekly Status Reports

A ZIP file containing the following.

- Systems Requirement Specification
- System Design Document
- User Interface Design Document
- User Manual
- Administrator Manual
- All Source Code (JavaScript source code constitutes an executable program)

Open Issues

Issues will be tracked on the GitHub issue tracking system. See the Omni issue tracker and the Omnichat issue tracker.

Appendix A - Customer and Contractor Agreement

The customer and Team including: Micah Havens, Scott Devere, C.J. Commodore, Adnaan Dasoo, and Josh Martin are agreeing to the implementation of Omni in accordance with the information listed in the information above. The team and the customer are agreeing that everything listed above is acceptable and sufficient for the task that the customer needs. If future changes need to be made to this document all members of the team will meet with the customer to explain what needs to be changed and why, and upon agreement the changes will follow.

Dated Signatures:

- Micah Havens, 10/20/22, X_____MH__
- Scott Devere, 10/20/22, X_____SD___
- C.J. Commodore, 10/20/22, X_____CC_____
- Adnaan Dasoo, 10/20/22, X_____AD___
- Josh Martin, 10/20/22, X_____JM__

Customer Area:

Customer Comments:
Date:_____ Signature:X_____

Appendix B - Team Review Sign-off

All members, including Micah Havens, Scott Devere, C.J. Commodore, Adnaan Dasoo, and Josh Martin, have reviewed the system requirements specification document for our software, named “Omni”. Each team member has reviewed this document for accuracy and completeness in all parts, including text, diagrams, bullets, charts, and tables.

Dated Signatures:

- Micah Havens, 10/20/22, X_____MH__
- Scott Devere, 10/20/22, X_____SD___
- C.J. Commodore, 10/20/22, X_____CC_____
- Adnaan Dasoo, 10/20/22, X_____AD____
- Josh Martin, 10/20/22, X_____JM____

Appendix C - Document Contributions

- Micah Havens
    - Worked on: Glossary and Functional Requirements
    - Percentage estimate: 45%
- Scott Devere
    - Worked on: Introduction and Deliverables
    - Percentage estimate: 15%
- C.J. Commodore
    - Worked on: Non-Functional Requirements
    - Percentage estimate: 15%
- Adnaan Dasoo
    - Worked on: Introduction and Open Issues
    - Percentage estimate: 10%
- Josh Martin
    - Worked on: Appendices and User Interface
    - Percentage estimate: 15%