Chatterbox Design

# Design

## Priority Features

* Everyone chat
* Chats
  + Create group
  + Invite
  + Leave
  + Boot?
* Personal chat (really a 2-person group chat)
* Retains history
* Archive button clears window and archives chat into history.
* Nickname yourself
* Nickname others
* See user info
  + user name
  + user nick name
  + your nick name
  + system name
  + system IP
  + away time
  + uptime
* See typing... messages as users are typing
* Semi-secure communication – Public/private keys encrypted messages.
* Formatted messages
* Send files

## Goals

* Spell checker

# Technical Design

* Main socket is multicast and is for health & status.
* Each conversation set’s up point to point communication for its users.

## Data Structures

### User Options

|  |
| --- |
| **ChatterboxOptions** |
| displayName:String  showPopups:boolean  isAway:boolean  activeChats:List<ChatInfo>  userOptions:List<UserOption>  inputs:MulticastsInputs |

|  |
| --- |
| **UserOption** |
| id:String  nickname:String |

### Main Structures

|  |
| --- |
| **Chatterbox** |
| connectionInputs:MulticastInputs  activeUsers:List<UserInfo>  defaultChat:Chat  activeChats:List<Chat> |

|  |
| --- |
| **Chat** |
| info:ChatInfo  hasUnread:Boolean  messages:List<ChatMessage> |

|  |
| --- |
| **ChatMessage** |
| from:UserInfo  sent:LocalTime  text:DecoratedText |

|  |
| --- |
| **ChatInfo** |
| id:String  name:String  users:List<String> |

|  |
| --- |
| **ChatUser** |
| id:String  displayName:String  available:boolean  lastSeen:LocalDateTime  typing:boolean |

### Handlers

|  |
| --- |
| **ChatterboxHandler** |
| connect(inputs:MulticastInputs):void  disconnect():void  createChat(name:String):Chat  addChat(chat:Chat):void  getUser(userid:String):ChatUser  setUserInfo(localUser:UserInfo):void  getDefaultChat():ChatHandler  getChatterbox():Chatterbox |

|  |
| --- |
| **ChatHandler** |
| sendMessage(text:DecoratedText):void  getChat():Chat |

### Chat History

|  |
| --- |
| **ChatHistory** |
| conversations:List<ConversationInfo> |

### Messages

A user message is sent every second to let everyone connected know who’s connected and when they were last seen.

#### User Message

|  |
| --- |
| **UserMessage** |
| id:String  displayName:String  available:boolean  lastSeen:LocalDateTime  activeChats:List<ChatInfo> |

#### Chat Message Message

|  |
| --- |
| **ChatMessageMessage** |
| chatId:String  txTime:LocalDateTime  senderId:String  text:String  attributes:TextStyleList |

#### Chat Invite Message

|  |
| --- |
| **ChatInviteMessage** |
| chatId:String  name:String  userId:String |

## Communications Net

The single communications channel is a multicast channel that relates user status, chat synchronization, and chat messages. When the software starts up, it sends a User Message at a rate of ~1 Hz.