CAUSALLY ORDERED GROUP CHAT

DISTRIBUTED SYSTEMS PROJECT

18th June, 2024 **Milan, Italy**

Rishabh Tiwari, Simone Errigo, Andrea Migliorini

POLITECNICO MILANO 1863

TABLE OF CONTENTS

01 OVERVIEW

02 ASSUMPTIONS

03 KEY COMPONENTS

07 DEMONSTRATION

04 CONNECTION MANAGEMENT

05 NODE OPERATIONS

06 CAUSAL ORDER

08 CONCLUSION

CAUSALLY ORDERED GROUP CHAT

Rishabh Tiwari, Simone Errigo, Andrea Migliorini

OVERVIEW

Objective: Implement a distributed group chat application.

Key Features:

- Create and delete rooms.
- Deliver messages in causal order within each room.
- Fully distributed with no centralized server.
- High availability: users can read and write messages even when temporarily disconnected.

Rishabh Tiwari, Simone Errigo, Andrea Migliorini

ASSUMPTIONS

- **Reliability:** Clients and links are reliable**.**
- **Dynamic Participation:** Clients can join and leave the network at any time.
- **Static Rooms:** Set of participants in a room is fixed at creation.

Rishabh Tiwari, Simone Errigo, Andrea Migliorini

KEY COMPONENTS

- **Connection Management:** Handles network communication.
- **Node:** Represents a user in the system, managing rooms and message queues.
- **Room Registry:** Tracks active and deleted rooms.
- **Messages:** Various message types for communication (e.g., RoomMessage, logRequestMessage).

CONNECTION MANAGEMENT

- **Multicast and Broadcast:**
  - Multicast for room-specific messages.
  - Broadcast for general announcements (e.g., heartbeat).
- **Message Handling:** Send and receive messages, serialize and deserialize messages.
- **User Discovery:** Update known users from received messages.
- **Heartbeat Mechanism:** Regularly broadcast presence and room updates.

Rishabh Tiwari, Simone Errigo, Andrea Migliorini

NODE OPERATIONS

- **Creating a Room:**
  - Generates a unique multicast IP.
  - Broadcasts room creation.
- **Joining a Room:**
  - Joins the multicast group.
  - Synchronizes message logs using vector clocks.
- **Leaving a Room:**
  - Leaves the multicast group.
  - Updates room registry.
- **Sending Messages:**
  - Ensures causal order using vector clocks.

Rishabh Tiwari, Simone Errigo, Andrea Migliorini

ENSURING CAUSAL ORDER

- **Vector Clocks:**
  - Each message carries a vector clock.
  - Ensures messages are delivered in causal order.
- **Message Queue:**
  - Holds messages until they can be delivered in the correct order.
  - Updates local clock and logs.

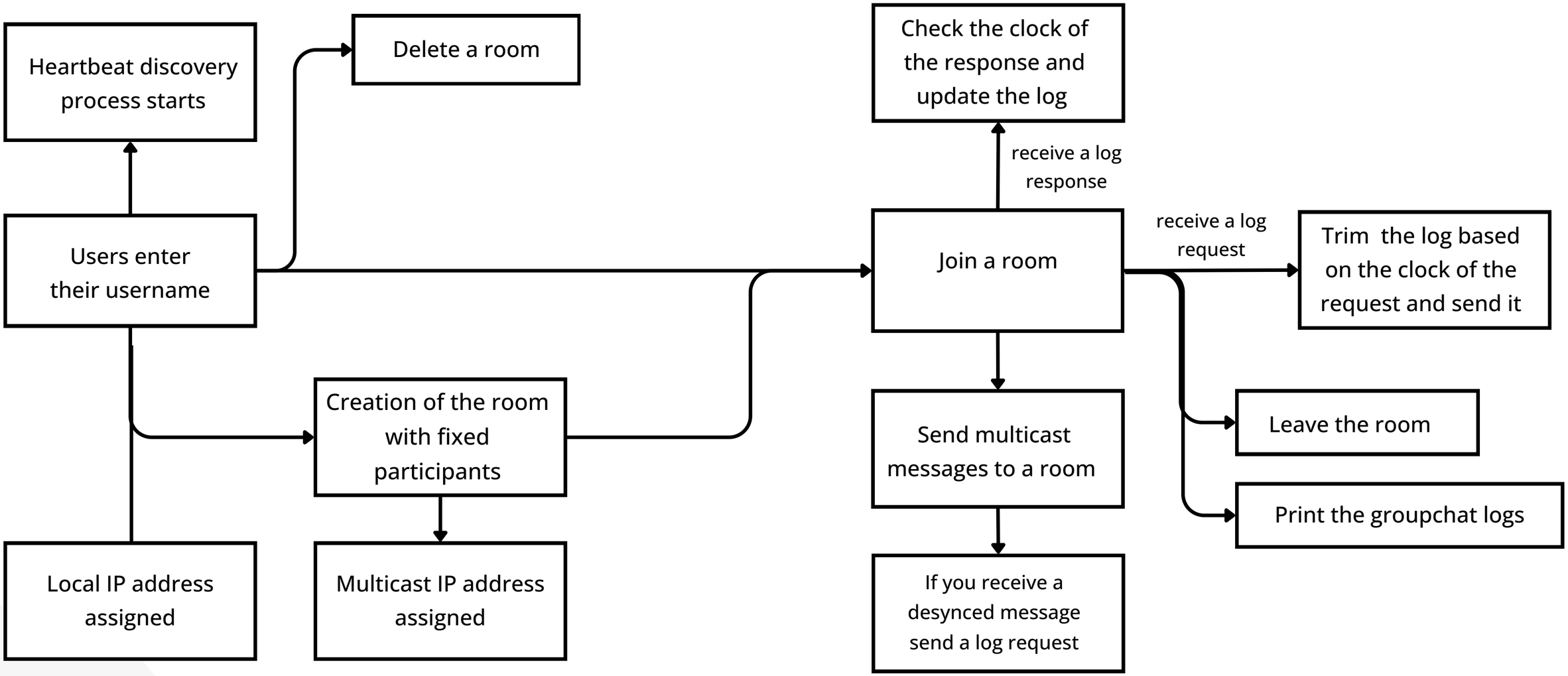
Rishabh Tiwari, Simone Errigo, Andrea Migliorini

HANDLING DISCONNECTIONS

- **High Availability:**
  - Users can continue to read/write messages while disconnected.
  - Messages are synchronized upon reconnection.
- **Log Requests:**
  - Request missing messages using logRequestMessage.
  - Respond to log requests with logResponseMessage.

Rishabh Tiwari, Simone Errigo, Andrea Migliorini

MESSAGE FLOW SYSTEM



Rishabh Tiwari, Simone Errigo, Andrea Migliorini

DEMONSTRATION

CAUSALLY ORDERED GROUP CHAT

Rishabh Tiwari, Simone Errigo, Andrea Migliorini

CHALLENGES AND SOLUTIONS

- **Causal Ordering:**
  - Implementing and managing vector clocks.
- **High Availability:**
  - Ensuring message delivery during and after disconnections.
- **Distributed Environment:**
  - Synchronizing state across multiple clients.

FUTURE IMPROVEMENTS

- **Dynamic Room Participants:**
  - Allow adding/removing participants after room creation.
- **Enhanced Fault Tolerance:**
  - Better handling of network partitions.
- **User Interface:**
  - Develop a more user-friendly interface for the chat application.

THANK YOU

ANY QUESTIONS?

18th June, 2024 **Milan, Italy**

Rishabh Tiwari, Simone Errigo, Andrea Migliorini