DistTag - A Distributed Sytems Project

Iaguța Alen-Mihael

January 30, 2026

Abstract

DistTag, or given the full name, Distributed Tag is a representation of the childhood game Tag in a distributed form. In this report presentation, the full game theory, rules and, most importantly, the way a distributed system will be re-enacted.

Contents

1 Introduction

Distributed Systems represent a category of essential software applications, in which many independent components communicate and collaborate to achieve a shared objective

DistTag project is implementing a multiplayer game of 'Tag' type in a distributed manner, demonstrating the fundamental concepts of this domain. The game is split into regions, every region having it's own dedicated server, thus the global state of the game is coordinated by a central server called 'Master'. Every player interacts with the region server through a client, moves are processed concurrently, and events of type 'Tag' are synchronized in real time.

This project offers a practical representation of distributed systems principles, including data partitioning, communication between nodes, global state coordination and handling concurrency, in a intuitive and interactive way.

Once you're familiar with the editor, you can find various project settings in the Overleaf menu, accessed via the button in the very top left of the editor. To view tutorials, user guides, and further documentation, please visit our help library, or head to our plans page to choose your plan.

2 Motivation

DistTag project has at it's core aim the demonstration of the fundamental principles of distributed systems through an interactive and easy-to-understand way: a multiplayer game of type 'Tag'. The game implies many players which traverse across a map which is divided my regions, every region being coordinated by a dedicated server, whilst a central server (Master) coordinates the global state of the game, including the player 'IT' and transmitting the 'Tag' event towards other players.

This project highlights data partitioning, because every region server coordinates just a part of the map, thus reducing the load to a singular node and advertising the scalability of the system. Also, the project illustrates concurrency in the same manner [TVS17], thus many other players can send commands in the same time towards region servers, and the system needs to process these commands correctly and simultaneously.

Another essential aspect is the communication between the nodes, being created through gRPC [grp] and Protocol Buffers [pro], this allows fast and efficient sending of messages towards clients, region servers and master server. In addition, the project offers a clear demonstration of coordination of the global state, thus highlighting the manner of how the Master server maintains data consistency and how the state and role of player 'IT' is updated in real time for the other players.

Now, DistTag is not just an interactive game, it is also a practical example of a functional distributed system, offering a method to apply studied theories in this class. Simply use the section and subsection commands, as in this example document! With Overleaf, all the formatting and numbering is handled automatically according to the template you've chosen. If you're using the Visual Editor, you can also create new section and subsections via the buttons in the editor toolbar.

3 Architecture

The architecture of the game was conceived to demonstrate fundamental principles of distributed systems, like data partitioning, global state coordination and handling concurrency. The system is split between three types of components: server, regions and clients, every role having well-defined responsibilities to ensure correct function of the game.

- Master Server (GameMaster) - represents the central points of control of the game. It coordinates the global state of every player, maintaining player 'IT' state and managing events of type 'Tag' send throughout the region servers. In addition, Master Server receives information about new players and ensures communication between regions to maintain data consistency at a global level. This server is essential for the synchronization of the participants and preventing conflicts between server states.[CV07]

- Region Servers - manage a part of the map and player positions from that zone. Region servers detect events of type 'Local Tag' and they send this data to the master server to update the global state. This partitioning reduces central server loading and allows system scalability when the number of players grows. Region servers are also responsible for processing concurrent moves of the players throughout their region.

- Clients - represent the game's interface. They send commands to move towards the region servers and they get updates about positions of other players and the state of player 'IT'. Client does not process the game's logic, it is dependent on the distributed servers to maintain consistency of the game in real time

Data Partitioning is represented in the project by splitting the map into two regions (Region A: x=0-49, Region B: x=50-99), so that every server manages only local data, reducing global load.

Concurrency is shows by the region servers processing simultaneously the moves other of many players, ensuring a safe way of messaging between clients, server regions and master server.

gRPC and Protocol buffers allow fast and safe transmission of messages between clients, region servers and master server.

Master Server synchronizes events and updates the player 'IT', guaranteeing that all participants obtain the same information, in real time.

4 DistTag

DistTag game is based on many processes that communicate with each other to maintain global state of the game and to answer the fast actions of the game. In a distributed system, every process has roles, responsibilities, thus contributing to correct and efficient functioning of the game. They include:

- Start of the game is being done by the Running the Master Server:

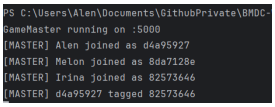

Fig 1: Start of the game.

- Starting of the region servers:

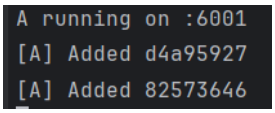

Fig 2: Start of the region A.

- Players joining:

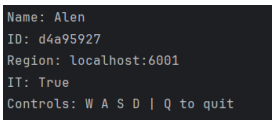

Fig 3: Player Alen has joined the game.

  The first player to join the game automatically becomes 'IT' player. While the other players try to avoid his position.

- Player movement:

Fig 4: Tag happened.

Player 'IT' has reached the same position as some other player, this means that the 'IT' status has been transferred to another player.

The goal of the DistTag game is for players to move around the map and avoid being tagged by the "IT" player. The "IT" player must tag others to make them IT, and the game continues in real time on a map divided into regions. The game is designed not only for fun and interaction but also to demonstrate key distributed systems concepts, such as data partitioning, global state coordination, and concurrent process management.

5 Conclusions

The DistTag project successfully demonstrates the application of distributed systems principles in a simple, interactive multiplayer game. By dividing the map into regions managed by separate servers and coordinating the global state through a Master server, the system illustrates data partitioning, concurrency management, inter-process communication, and global state consistency.

The project highlights the benefits of distributed architectures, including scalability and fault isolation, while providing a practical and engaging example for students to understand theoretical concepts. Moreover, it shows how modern technologies like gRPC and Protocol Buffers can be used to implement real-time communication between multiple processes efficiently.

Potential extensions of the project include adding more regions, implementing handoff when players cross region boundaries, leaderboards, and more complex gameplay mechanics, all of which can further showcase distributed system concepts in practice.

References

[CV07] D. Chandra and J. Vitter. Cloud computing for real-time multiplayer games. In *Proceedings of the 12th International Conference on Distributed Computing Systems Workshops*, pages 47–52, 2007.

[grp] grpc documentation. https://grpc.io/docs/.

[pro] Protocol buffers. https://developers.google.com/protocol-buffers.

[TVS17] Andrew S. Tanenbaum and Maarten Van Steen. *Distributed Systems: Principles and Paradigms*. Pearson, 3rd edition, 2017.