Distributed Tic Tac Toe: Peer-to-Peer Team-Based Gaming System

Panagiotis Antoniou, Emily Band, Paulius Dilkas, Dafin Kozarev, Joshua Styles

***Abstract*—This article describes a Distributed Multiplayer Tic Tac Toe software game written in Java using the RMI API. While previous implementations attempting to serve a similar purpose exist, none of them boast good design or efficiency and neither of them deliver the same functionality. The developed piece of software allows clients to play locally or globally a game of the classic tic tac toe - but with a twist. Players are separated into teams and decide on what the next move should be by voting. The implementation is derived from a design that follows the distributed applications guidelines and is prepared to handle corner cases such as client disconnects for example.**

I. Introduction

Coding is the main occupation of many people nowadays, and is also a past-time for many too. Most people get their first steps in writing code following the local single-threaded model. Often, it can be difficult for beginner programmers, or even those with more experience, to abstract the key knowledge they have obtained using this model and then apply it in a different way.

Distributed software requires a new look on the system the number of devices has gone up and with it so have the number of processes, concerns and complexities. Some concepts have now changed and need to be dealt with differently; One such example if this would be mutual exclusion. We believe programs that serve as examples and focus on simple local tasks turned into distributed ones can help many people trying to make this step with a hands-on experience of what they are preparing for. Seeing theory in practice is a proven technique and there is evidence for this in every university course that features practical work. This is especially true where games are concerned, proving to be an effective means of communicating programming methods and techniques in an engaging way [1]. For this reason, the team has decided to take a simple, well known game and develop it into a distributed system; Tic Tac Toe played as a peer-to-peer, team-based gaming system where the players are organised into teams and required to vote on moves for their team.

By refraining from introducing additional complexity as part of the goal of the project, and allowing the users to see the distributed nature of the software being demonstrated via a simple game, the focus is shifted towards the implications of the distributive aspects of the system and the way they are handled. The finished product can serve to entertain the end users, as well as increase their curiosity and insight about computers and distributed systems. Using Tic Tac Toe as a foundation to demonstrate a distributed system benefits from both the simplicity and familiarity of the game itself, meaning no complex rulesets or in-depth mechanisms need be explained to the users, allowing the distributed nature to be a novel way of experiencing a widely known game.

II. System Design

Tic Tac Toe is a simple game played on a grid of 3x3, consisting of two players. One player uses an X marker and the other player uses an O marker to select a grid space to occupy. The objective of the game is to take turns until one player manages to occupy three grid spaces in a straight line, be it horizontal, vertical or diagonal.

The proposed design takes this simple concept and uses it to demonstrate a distributed system, whereby the two players are instead two teams of up to 5 players, which take turns to vote on where their team should place their markers. Each team has a designated ”leader” which communicates data with their team’s peers, and the opposing team’s ”leader” to synchronise current gameplay state.

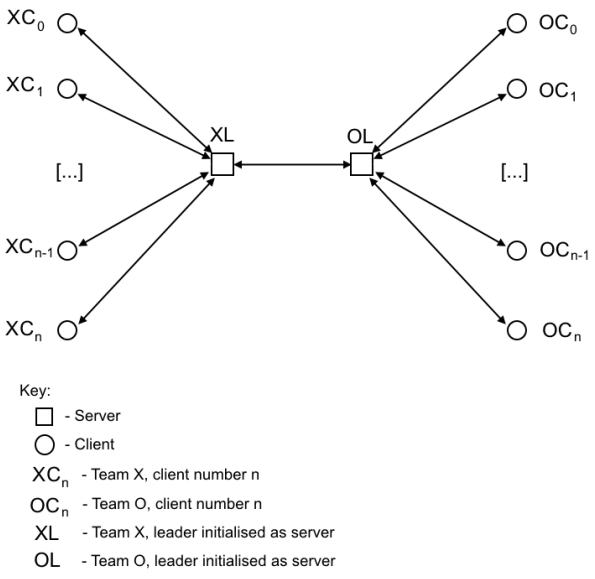

Fig. 1. Topology for Distributed Tic Tac Toe

The leaders communicate the vote of their team to the opposing leader so that they may pass that result on to the peers of that team, as pictured in figure 1.

III. Implementation

IV. Evaluation

V. Related Work

While other multiplayer distributed games already exist, they have not been created with this projects task in mind and therefore do not serve the same purpose. They can be split into two categories very unprofessional and too professional. For example, hobby implementations exist that deal with distributed problems in a bad fashion, promoting bad and incorrect programming practises. Similarly, a single-player-per-team distributed tic-tac-toe game in Java can be found online that does not comply with any design guidelines and puts all functionality in a single class and as such, cannot teach students good practice and has the potential of confusing the learner further about distributive programming [2].

Separately, very in-depth materials that provide state-of-the-art practices can be found [3] but there is a high possibility that they might be incomprehensible to a beginner reader looking to tackle basic distributed concepts to begin with. Furthermore, none of the similar implementations residing on the Internet take advantage of the Java RMI API which must be considered a good starting point for distributed systems learners for it to be taught as part of our degree and is a requirement for this project. Instead, they use different sets of other technologies which can further confuse their reader and introduce a steeper learning curve.

VI. Conclusions and Future Work

References

[1] J. R. Kiniry and D. M. Zimmerman, "Verified gaming," in *Proceedings of the 1st International Workshop on Games and Software Engineering*, ser. GAS '11. New York, NY, USA: ACM, 2011, pp. 17–20. [Online]. Available: http://doi.acm.org/10.1145/1984674.1984681

[2] L. Kotthoff, "LLAMA: leveraging learning to automatically manage algorithms," arXiv, Tech. Rep. arXiv:1306.1031, Jun. 2013. [Online]. Available: http://arxiv.org/abs/1306.1031

[3] R. Diestel, *Graph Theory, 5th Edition*, ser. Graduate texts in mathematics. Springer-Verlag, 2016, vol. 173.