MALTE: Multi-Access Live Text Editor

Draft Report

Software Engineering Project, 1DL650

Team Name: #dream-team

I. Introduction

The overarching project goal is to allow users to collaboratively edit and compile code. The system should have supporting functionality that makes this task easy, which could include access to terminal, commenting functions, chat functions, code high-lighting, etc.

MALTE is... (More about MALTE itself here, features? target audience? more?)

II. Problem Formulation

MALTE is with no doubt a distributed system with many points of replication. Both the text in each file is replicated among all clients currently editing that file and the structure of the file system are as well. In this case, however, the promise of EC that replicas will converge sometime in the future is not enough.

III. Background

In order to understand what problems Conflict-Free Replicated Data Type (CRDT) solve, we need to introduce some theory about consistency models which we will use in the context of collaborative text editing.

We use some terminology as defined by Briot et al. (2016). Firstly, *replica*, which represents a "copy" of some shared state. Each replica can be modified with *operations*, which is data that represents what modification is made, and how it should be performed. The *operation* are then sent to other clients which apply them to their own replicas. This can be problematic in a distributed setting since some operations are not commutative, i.e., the order of operations influence the end state of the replica. For example, insertions into a string is not commutative since $a+b \neq b+a$. Different algorithms behave differently and can guarantee different properties of the final state. What properties can be guaranteed for an algorithm said to be the algorithms *consistency model*.

A. Consistency Model

All distributed systems who deal with replicated data have to deal with the problem of consistency. There are many different consistency model, of which two are especially important. Eventual Consistency (EC) is a common consistency model which allow different replicas to diverge as long as they sometime in the future converge [1].

Strong Eventual Consistency (SEC) is a specific consistency model within the EC consistency model family. This model promises that all replicas that have observed the same operations, in any given order, will converge to the same state [1] without the need of any further synchronisations.

The promises of SEC is suitable for a system as MALTE as it ensures that all operations are considered for the final state and thus is we would like to achieve. For the file tree, however, we are content to use the relaxed EC definition. The reasoning for this is that file system operations are less frequent and consistency can be solved with simpler means instead, e.g. checking if a file actually exists before trying to rename it. This is possible thanks to the use of a central server who has authority over the file system and that can resolve any potential race conditions.

B. Conflict-free Replicated Data Type

TODO: Summarize what CRDT is here

To improve the performance of a Conflict-Free Replicated Data Type (CRDT) called Replicable Growable Array (RGA), one can use an auxiliary structure as described in the paper *High Responsiveness for Group Editing CRDTs* [2]. The auxiliary data structure has no need to be synchronized between replicas, yet improves the upstream operation complexity considerably. Implementing this auxiliary structure is out of scope for this project, but it is nonetheless worthwhile to know that improvements to the underlying structure are possible.

IV. Related Work

There are already similar systems for collaborative, in-browser editing of code. These systems range from simple collaborative editors, such as Firepad, to full-fledged integrated development environments (IDEs) running on the cloud, such as Visual Studio Online [3], [4]. Visual Studio Online can be accessed through a browser and extended to allow for collaborative editing in a project. REPL.IT extends the concept of a project in Visual Studio Online by allowing for classrooms

where students can learn and collaborate while teachers can assist the students and grade their solutions [5].

Other related software that experiences synchronisation issues as ours are collaborative productivity tools such as Google Docs and Figma, all of which either uses Operational Transformations or Conflict-Free Replicated Datastructures to handle synchronisation issues [6]–[8].

V. Requirements

VI. Technical Documentation

// TODO: API spec, development docs, maybe extract typescript documentation as well

VII. Quality Control

// TODO, Tests, linting, CI/CD, peer review,

VIII. User Documentation

// TODO: URL and short description of functionality, also the command ‘docker run -p 4000:4000 malteuu:dev‘ to self-host the app

IX. Project Management

// TODO: Scrum-esque approach, standups, kanban board, git flow, pull requests with review, sprints and sprint reviews

X. Discussion

// TODO:

XI. Future Work

// TODO: software as service with meta server, more?

References

[1] J. Bauwens and E. Gonzalez Boix, “Memory efficient crdts in dynamic environments,” in *Proceedings of the 11th ACM SIGPLAN International Workshop on Virtual Machines and Intermediate Languages*. ACM, 2019, pp. 48–57.

[2] L. Briot, P. Urso, and M. Shapiro, “High responsiveness for group editing crdts,” in *Proceedings of the 19th International Conference on Supporting Group Work*. ACM, 2016, pp. 51–60.

[3] “An open source collaborative code and text editor.” [Online]. Available: https://firepad.io/

[4] “Developing with Visual Studio Online,” Apr 2016. [Online]. Available: https://code.visualstudio.com/docs/remote/vsonline

[5] Repl.it, “The World’s Leading Online Coding Platform.” [Online]. Available: https://repl.it/

[6] “Operational Transformation,” Oct 2019. [Online]. Available: https://en.wikipedia.org/wiki/Operational_transformation

[7] “Conflict-Free Replicated Data Type,” Nov 2019. [Online]. Available: https://en.wikipedia.org/wiki/Conflict-free_replicated_data_type

[8] “How Figma’s Multiplayer Technology Works.” [Online]. Available: https://www.figma.com/blog/how-figmas-multiplayer-technology-works/