# Introduction

A while ago, I built an app called ”Tryout”, which uses the notorious ShareDB library to map operational transforms onto a collaborative coding environment. The app was fun to build, but it left me with a lasting desire to better understand how distributed data structures work, and how I might rebuild ”Tryout” using my very own structure (or at least one I can understand the internals of).

The ”big” thing in distributed data structures is *Conflict Free Replicated Data Types*, or CRDTs. They offer the strong eventual consistency guarantees that one might expect from any distributed data structure, but they include the added benefits of more efficient memory usage and scalability beyond two peers in practical settings.

## So what is this post?

I am collecting a bunch of notes and links about the topic in order to collect my thoughts. I will be reading a few papers and I figured my understanding of them may improve if I take the time to summarize certain parts. Furthermore, I will be curating a list of links I find useful and interesting.

### Seminal Paper: A comprehensive study of Convergent and Commutative Replicated Data Types

#### Background

- Eventual consistency allows for asynchronous replication with other users such that they all reach the same state *eventually*.
- CRDTs require no synchronization, allowing users to apply their updates immediately.
- CRDTs do not use consensus under the hood.
- Certain limitations requiring expensive operations, these can be delayed to a later period when the network is well connected.
- An *atom* is a basic data structure which is contained within an *object*.
- Objects can be replicated, and are independent of each other within the process in which they are located.
- An *operation* is applied on an object by a client, first applied to a source replica and is then propogated asynchronously to all other replicas.

### Links

#### Papers

##### Unique CRDT and Implementations

- Peritext

- Chronofold
  - Github

## Blog Posts

### Unique CRDT and Implementations

- Conclave
- Antimatter (self Pruning, guys from Braid)

## Forums and Discussions

- Be aware that CRDTs like automerge are solving a different (and harder) problem than Replicache.
- Main takeaways from toying with both Yjs and Automerge:

## Lists

- Awesome CRDT
- Choosing a realtime collaboration library

## CRDT Repos

- Automerge (Martin Kelppman)
- Diamond Types (Seph Gentle)
- YJS

## ShareDB

- ShareDB Postgres (outdated)
- Official Docs

## Random

- RepliCache (ie. has central authority?)
- How NAT Traversal works