File formats for Conflict-free Replicated Data

Pierre-Évariste Dagand Gilles Muller Marc Shapiro
Whisper team Regal team

The Internet was created to provide data replication across geographically-distributed computing centers. Today's cloud storage infrastructure looks a lot alike the Network of the early days. Persistency and low latency is achieved through replication in datacenters distributed over the world. However, support for large-scale editing is somewhat lacking.

Storage providers, such as Dropbox for example, offer the possibility to share files over the Internet, replicating data near the users. Storage granularity is at the file level: access to a single file must be exclusive, or conflicts might arise between diverging replicas. Version control systems (VCS), such as Git, address this issue by integrating a semi-automatic merging mechanism: by keeping the file history, a VCS can try to reconcile diverging replicas by taking advantage of their common ancestor. However, by treating files as unstructured streams of characters, a merge may fail or result in an ill-formed file. Indeed, merging does not take into account the semantic information carried by the file.

In this internship, we propose to design a filesystem for large-scale collaborative and distributed editing. By implementing the filesystem as a single conflict-free replicated datatype (CRDT) [Shapiro et al., 2011, Preguica et al., 2009, Martin et al., 2012], we wish to support the replication of files across datacenters while automating replicas' convergence. We shall therefore relate the file format with semantical properties of the data being persisted. To do so, we will design a domain-specific language (DSL) [Mernik et al., 2005] for specifying conflict-free file formats, guaranteeing that – by construction – formats thus described always converge.

Let us for example consider a vector graphics file format. Fundamentally, such a file describes the content of a canvas, which can be thought of as an infinite 2-dimensional grid together with a finite collection of layers. This canvas has some interesting mathematical structure [Yorgey, 2012]. For instance, the interaction of users operating over two distinct points A and B is *commutative*: drawing a blue dot at A after drawing a red dot at B is equivalent to first drawing a red dot at B followed by drawing a blue dot at A. Commutativity eases the merging of replicas: it tells us that whichever order we apply the modifications do not matter.

However, this description is simplistic: a disc drawn at A might overshadow with a square drawn at B. Depending on the layer on which A stands, the square may appear below or above the disk: drawings are non-commutative. To merge two vector graphic files automatically, we must therefore specify a *merging policy*. For instance, one could specify that discs should always be layered above squares.

This example illustrates two key design principles. First, to converge automatically, a file format must reflect the semantics of its application domain (here: the structure of the canvas). Second, to account for conflicting edits, one has to specify a merging policy through which non-commutative operations normalize to a single, non-ambiguous object.

To build a conflict-free filesystem involves a mixture of techniques, ranging from the more theoretical aspects of computer sciences to the more practical ones:

- *Semantics:* we shall adapt the mathematical structure of CRDTs to structured file formats, yielding a theory of convergent file formats
- *Language design:* we shall design a programming language to describe convergent file formats, proving that any format thus described is eventually consistent
- *System design:* we shall implement a collaborative cloud-oriented filesystem, relying on semantic information to provide fine-grained replication and merging

This internship can easily be tailored to the student's interest. For instance, one could consider a single file format and focus on system design. Alternatively, one could focus on developing the theory, or focus on language design by re-implementing existing CRDTs.

The team: This internship will be held at LIP6 (UPMC, Jussieu campus) in the Regal and Whisper teams (Inria). It will be jointly supervised by Pierre-Évariste Dagand (for the language aspects) and Marc Shapiro (for the systems aspects). This project is part of an ongoing collaboration with Vianney Rancurel, head of research at Scality, a start-up providing peta-scale storage services.

References

S. Martin, M. Ahmed-Nacer, and P. Urso. Controlled conflict resolution for replicated document. In *Collaborative Computing: Networking, Applications and Worksharing*, pages 471–480. IEEE, Oct. 2012.

M. Mernik, J. Heering, and A. M. Sloane. When and how to develop domain-specific languages. *ACM Comput. Surv.*, 37(4):316–344, Dec. 2005. doi:10.1145/1118890.1118892.

N. Preguica, J. M. Marques, M. Shapiro, and M. Letia. A commutative replicated data type for cooperative editing. In *IEEE International Conference on Distributed Computing Systems*, ICDCS '09, pages 395–403, 2009.

M. Shapiro, N. Preguiça, C. Baquero, and M. Zawirski. Conflict-free replicated data types. In *Proceedings of the 13th International Conference on Stabilization, Safety, and Security of Distributed Systems*, SSS'11, pages 386–400, 2011.

B. A. Yorgey. Monoids: Theme and variations (functional pearl). In *Proceedings of the 2012 Haskell Symposium*, Haskell '12, pages 105–116, 2012. doi:10.1145/2364506.2364520.