Verdi: Implementing and Formally Verifying Distributed Systems
verdi.uwplse.org

Verdi

Implementing correct distributed systems is difficult

Fault-tolerance techniques are complex and error-prone

Formalize fault models in Coq

Verify systems against models

Express fault-tolerance mechanisms as transformers between models

Extract to Ocaml and run!

Network Semantics

\[ H_{\text{inp}}(\Sigma[h], i) = (\sigma', o, P') \]

\[ \frac{(P, \Sigma, T) \rightsquigarrow (P \cup P', \Sigma[h \mapsto \sigma'], T++ \langle i, o \rangle)}{} \]

\[ H_{\text{net}}(dst, \Sigma[dst], src, m) = (\sigma', o, P') \]

\[ \frac{\{(src, dst, m)\} \cup P, \Sigma, T \rightsquigarrow (P \cup P', \Sigma[dst \mapsto \sigma'], T++ \langle o \rangle)}{} \]

\[ \frac{(\{p\} \cup P, \Sigma, T) \rightsquigarrow (P, \Sigma, T)}{} \]

Case studies

Lock server
Key-value store

Sequence numbering
Retransmission
Primary-backup replication
Consensus-based replication

The Raft Consensus Protocol

State machine replication protocol

First mechanically-checked proof of correctness, in Verdi

~50k LOC

Performance Evaluation

Implement and verify a key-value store

Compare with etcd, a similar unverified store

Overall, the Verdi store is ~10% slower

James R. Wilcox        Doug Woos        Steve Anton        Pavel Panchekha        Zachary Tatlock        Xi Wang        Michael D. Ernst        Thomas Anderson