<~ bloom

Fork me on GitHub

CALM: consistency as logical monotonicity

One of the key innovations underlying Bloom is the ability to *formally guarantee* consistency properties of distributed programs. This reasoning is based on the *CALM* principle, which was the subject of recent theoretical results. This theory applies to any programming paradigm, but Bloom's roots in logic make it easy for us to convert the theory into practical tools for Bloom programmers.

background

Informally, a block of code is logically *monotonic* if it satisfies a simple property: adding things to the input can only increase the output. By contrast, *non-monotonic* code may need to "retract" a previous output if more is added to its input.

In general terms, the CALM principle says that:

- logically monotonic distributed code is *eventually consistent* without any need for coordination protocols (distributed locks, two-phase commit, paxos, etc.)
- eventual consistency can be *guaranteed* in any program by protecting non-monotonic statements ("points of order") with coordination protocols.

It turns out that some of the important design maxims used by experienced distributed programmers are in fact techniques for minimizing the use of non-monotonic reasoning. The problem is that without language support, code built around these maxims is hard to test and hard to trust — especially when it is likely to be maintained by groups of developers over time.

keeping CALM with bloom

Bloom's roots in temporal logic make it amenable to simple checks for non-monotonicity. This enables us to build automated tools that assist Bloom programmers to guarantee consistency properties while using a minimum of coordination.

learn more

- An intuitive intro the CALM principle appears in this blog post.
- A paper in CIDR 2011 provides a more detailed introduction to the CALM principle, and shows how it relates to distributed design patterns used by experienced developers at Amazon and Microsoft. *Note: the Bloom examples in this paper were based on an early prototype, and syntax has changed since then. See the bud sandbox for working implementations.*
- The CALM principle was introduced as a conjecture in a keynote talk at PODS 2010 [slides] [video], along with a number of related conjectures regarding space, time, and complexity. This was written up in a paper in SIGMOD Record on The Declarative Imperative.
- An upcoming paper in PODS 2011 presents formal statements and proof of the CALM Theorem. A somewhat different formalism and proof appears in an upcoming technical report from Berkeley.

NEWS

bud 0.9.7 released
bud 0.9.6 released
bud 0.9.5 released

LINKS

GitHub
Mailing List
BOOM project

Fork me on GitHub

Set your Twitter account name in your settings to use the TwitterBar Section.

Bloom Programming Language

PAGES

- bloom language: research
- bloom team
- FAQ
- features
- get started with bud!
- home
- thanks
- bud: bloom under development
- calm: consistency as logical monotonicity

THE LATEST

bud 0.9.7 released

Bud 0.9.7 has been released and is available for download via RubyGems. […]

MORE