Paper Trail

Computer Systems, Distributed Algorithms and Databases

Distributed systems theory for the distributed systems engineer

Gwen Shapira, SA superstar and now full-time engineer at Cloudera, asked a question on Twitter that got me thinking.

My response of old might have been "well, here's the FLP paper, and here's the Paxos paper, and here's the Byzantine generals paper…", and I'd have prescribed a laundry list of primary source material which would have taken at least six months to get through if you rushed. But I've come to thinking that recommending a ton of theoretical papers is often precisely the wrong way to go about learning distributed systems theory (unless you are in a PhD program). Papers are usually deep, usually complex, and require both serious study, and usually *significant experience* to glean their important contributions and to place them in context. What good is requiring that level of expertise of engineers?

And yet, unfortunately, there's a paucity of good 'bridge' material that summarises, distills and contextualises the important results and ideas in distributed systems theory; particularly material that does so without condescending. Considering that gap lead me to another interesting question:

What distributed systems theory should a distributed systems engineer know?

A little theory is, in this case, not such a dangerous thing. So I tried to come up with a list of what I consider the basic concepts that are applicable to my every-day job as a distributed systems engineer; what I consider 'table stakes' for distributed systems engineers competent enough to design a new system. Let me know what you think I missed!

First steps

These four readings do a pretty good job of explaining what about building distributed systems is challenging. Collectively they outline a set of abstract but technical difficulties that the distributed systems engineer has to overcome, and set the stage for the more detailed investigation in later sections

Distributed Systems for Fun and Profit is a short book which tries to cover some of the basic issues in distributed systems including the role of time and different strategies for replication.

Notes on distributed systems for young bloods – not theory, but a good practical counterbalance to keep the rest of your reading grounded.

A Note on Distributed Systems – a classic paper on why you can't just pretend all remote interactions are like local objects.

The fallacies of distributed computing – 8 fallacies of distributed computing that set the stage for the kinds of things system designers forget.

Failure and Time

Many difficulties that the distributed systems engineer faces can be blamed on two underlying causes:

1. processes may fail
2. there is no good way to tell that they have done so

There is a very deep relationship between what, if anything, processes share about their knowledge of *time*, what failure scenarios are possible to detect, and what algorithms and primitives may be correctly implemented. Most of the time, we assume that two different nodes have absolutely no shared knowledge of what time it is, or how quickly time passes.

You should know:

* The (partial) hierarchy of failure modes: crash stop -> omission -> Byzantine. You should understand that what is possible at the top of the hierarchy must be possible at lower levels, and what is impossible at lower levels must be impossible at higher levels.

* How you decide whether an event happened before another event in the absence of any shared clock. This means Lamport clocks and their generalisation to Vector clocks, but also see the Dynamo paper.

* How big an impact the possibility of even a single failure can actually have on our ability to implement correct distributed systems (see my notes on the FLP result below).

* Different models of time: synchronous, partially synchronous and asynchronous (links coming, when I find a good

reference).

The basic tension of fault tolerance

A system that tolerates some faults without degrading must be able to act as though those faults had not occurred. This means usually that parts of the system must do work redundantly, but doing more work than is absolutely necessary typically carries a cost both in performance and resource consumption. This is the basic tension of adding fault tolerance to a system.

You should know:

* The quorum technique for ensuring single-copy serialisability. See Skeen's original paper, but perhaps better is Wikipedia's entry.

* About 2-phase-commit, 3-phase-commit and Paxos, and why they have different fault-tolerance properties.

* How eventual consistency, and other techniques, seek to avoid this tension at the cost of weaker guarantees about system behaviour. The Dynamo paper is a great place to start, but also Pat Helland's classic Life Beyond Transactions is a must-read.

Basic primitives

There are few agreed-upon basic building blocks in distributed systems, but more are beginning to emerge. You should know what the following problems are, and where to find a solution for them:

* Leader election (e.g. the Bully algorithm)

* Consistent snapshotting (e.g. this classic paper from Chandy and Lamport)

* Consensus (see the blog posts on 2PC and Paxos above)

* Distributed state machine replication (Wikipedia is ok, Lampson's paper is canonical but dry).

Fundamental Results

Some facts just need to be internalised. There are more than this, naturally, but here's a flavour:

- You can't implement consistent storage and respond to all requests if you might drop messages between processes. This is the CAP theorem.
- Consensus is impossible to implement in such a way that it both a) is always correct and b) always terminates if even one machine might fail in an asynchronous system with crash-* stop failures (the FLP result). The first slides – before the proof gets going – of my Papers We Love SF talk do a reasonable job of explaining the result, I hope. *Suggestion: there's no real need to understand the proof*.
- Consensus is impossible to solve in fewer than 2 rounds of messages in general

Real systems

The most important exercise to repeat is to read descriptions of new, real systems, and to critique their design decisions. Do this over and over again. Some suggestions:

Google:

GFS, Spanner, F1, Chubby, BigTable, MillWheel, Omega, Dapper. Paxos Made Live, The Tail At Scale.

Not Google:

Dryad, Cassandra, Ceph, RAMCloud, HyperDex, PNUTS

Postscript

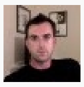
If you tame all the concepts and techniques on this list, I'd like to talk to you about engineering positions working with the menagerie of distributed systems we curate at Cloudera.

Published: August 9, 2014
Filed Under: Distributed systems

42 Responses to "Distributed systems theory for the distributed systems engineer"

1. *James Creasy* says:
   August 9, 2014 at 9:08 pm

Surprised you don’t mention the work of Dr. Aahlad, who designed a 100% data safe WAN capable consensus algorithm for distributed systems. An implementation of this algorithm is available to provide unmatched high availability to Cloudera producst and technologies.

2.  *Henry* says:
   [August 9, 2014 at 9:15 pm]()

   James – I don’t mention it because a) I’m unfamiliar with it (do you have a reference?) and b) unless the work contributed something theoretically new and non-incremental, it doesn’t belong on this list, except maybe in the ‘real systems’ section, but as I say there’s no obvious reference with which to judge it.

3. *Aurojit Panda* says:
   [August 9, 2014 at 11:27 pm]()

   I think Knowledge and Common knowledge by Halperm and Moses (http://www.cs.utexas.edu/users/lorenzo/corsi/cs380d/papers/p549-halpern.pdf) might be a good addition to the fundamental results section,

4. *[Distributed systems theory for the distributed systems engineer : Rifec Security]()* says:
   [August 9, 2014 at 11:36 pm]()

   […] Distributed systems theory for the distributed systems engineer […]

5. *Marc-Philippe Huget* says:
   [August 9, 2014 at 11:38 pm]()

   Interesting reading. Maybe you can mention as well Record & Replay technique for distributed, consistent save of machine state, studied for years by academics

6. *Tom Samplonius* says:
   [August 10, 2014 at 12:03 am]()

   @James Creasy

   From what I’ve read, Dr. Aahlad just created an implementation of Paxos for the company WANdisco.

   WANdisco has moved into the Hadoop market with a NameNode alternative.

   Source: http://www.information-age.com/technology/information-management/123457005/hadoop-will-be-the-operating-system-for-the-data-centre–says-wandisco

7. *tyler neely* says:
   [August 10, 2014 at 12:15 am]()

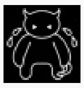
   It was interesting for me to learn about CRDT’s and LVars after studying dynamo-style databases and thinking about the problem of determinism in an eventually consistent system.

8. *Alex Newman* says:
   [August 10, 2014 at 12:25 am]()

   https://www.google.com/patents/US8364633?dq=Aahlad&ei=6B3nU_alMKaWigKt3YGgDg&cl=en

9. *[Share: Distributed systems theory for the distributed systems engineer : Paper Trail | Toy Elephants]()* says:
   [August 10, 2014 at 12:49 am]()

   […] Distributed systems theory for the distributed systems engineer : Paper Trail. […]

10.  *Sergio Bossa* says:
    [August 10, 2014 at 4:12 am]()

    I think a “modern” distributed systems engineer should also have some understanding of the latest research in transactional systems, such as Calvin and RAMP.

Also, in terms of time and consistent snapshots, I find very interesting the latest research on Hybrid Logical Clocks.

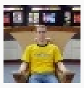
Otherwise, great writeup full of fundamental resources 🙂

11. *[Distributed systems theory for the distributed systems engineer : Paper Trail | rndmblg](#)* says:
    [August 10, 2014 at 5:01 am](#)

    […] http://the-paper-trail.org/blog/distributed-systems-theory-for-the-distributed-systems-engineer/ […]

12. *Victor Lowther* says:
    [August 10, 2014 at 7:36 am](#)

    It would probably be a good idea to throw a link to raft (http://raftconsensus.github.io/) as an easier to reason about alternative to paxos.

13.  *David Phillips* says:
    [August 10, 2014 at 2:42 pm](#)

    James Creasy seems to be referring to this: http://www.wandisco.com/hadoop/non-stop-hadoop-cloudera

14. *[Links & reads for 2014 Week 32 | Martin's Weekly Curations](#)* says:
    [August 11, 2014 at 3:13 am](#)

    […] Distributed systems theory for the distributed systems engineer […]

15. *[面向分布式系统工程师的分布式系统理论 – 极客巴士](#)* says:
    [August 11, 2014 at 9:22 pm](#)

    […] 原文：Distributed systems theory for the distributed systems engineer […]

16. *[[翻译]面向工程师的分布式系统理论 | Gopher beyond Eliphants](#)* says:
    [August 11, 2014 at 9:32 pm](#)

    […] 原文在此。 ————翻译分隔线———— […]

17. *Marco Voelz* says:
    [August 11, 2014 at 11:59 pm](#)

    Dear Henry,

    that is an amazing list you compiled here, thanks for the great effort! If I could add a single paper, it would probably be Chandra and Toueg’s “weakest failure detector” one, as it gives a nice idea on how to solve the dilemma that FLP puts you in.

    Warm regards
    Marco

18. *Vasia Kalavri* says:
    [August 13, 2014 at 6:47 am](#)

    Nice list! If I would have to add one thing, that would be the Chord paper. There can exist no distributed systems reading list without it 🙂
    Regarding the 8 fallacies, take a look at “The Seven Deadly Sins of Distributed Systems” paper: https://www.usenix.org/legacy/event/worlds04/tech/full_papers/muir/muir.pdf. Same idea, more detail.
    And, finally, 3 books for reference: (a) “Distributed Systems: Concepts and Design”, by G. Coulouris, (b) “Distributed Systems: Principles and Paradigms” by Tanenbaum and (c) “Introduction to Reliable and Secure Distributed Programming” by C. Cachin.

19. *[Distributed Systems Primer | A Polyglot's Tale](#)* says:
    [August 14, 2014 at 1:29 pm](#)

    […] Robinson, the smart Cloudera SDE behind The Paper Trail blog, just posted a great primer on distributed systems. I highly recommend anyone interested in this field go and take a look – if you’ve been […]

20. *[Sysadmin Sunday 190 - Server Density Blog](#)* says:
    [August 17, 2014 at 8:01 am](#)

[…] Distributed systems theory for the distributed systems engineer […]

21. *分散式系統的基礎理論 | Gea-Suan Lin's BLOG* says:
    August 19, 2014 at 7:27 am

    […] 這篇「Distributed systems theory for the distributed systems engineer」列出了分散式系統的許多理論，以及後來開發的經典應用。 […]

22. *tom* says:
    August 20, 2014 at 6:55 am

    i love you! it is so beatiful

23. *Best of Big Data Reads: August 2014 - Evan Volgas* says:
    August 22, 2014 at 7:28 am

    […] start things off, there's a great article on distributed systems theory and practice over at The Paper Trail. The article provides some references to a number of great articles about […]

24. *Brad King* says:
    August 27, 2014 at 1:49 am

    I think Marc Shapiro's papers on CRDTs merit a spot in a distributed system theory readers list. His video/paper at Microsoft is a good start "Strong Eventual Consistency and Conflict-free Replicated Data Types"

25. *Distributed Systems | Tales from a Trading Desk* says:
    August 27, 2014 at 7:54 pm

    […] systems theory for the distributed systems […]

26. *Hour 1 | My Distributed Systems Notes* says:
    August 29, 2014 at 9:41 pm

    […] http://the-paper-trail.org/blog/distributed-systems-theory-for-the-distributed-systems-engineer/ […]

27. *Raghavender Boorgapally* says:
    August 30, 2014 at 11:17 am

    Nice blog to start mastering distributed systems.

28. *York Tsai* says:
    November 4, 2014 at 7:00 pm

    A great collection of knowledge about distributed system, listing many famous products.

29. *joseph* says:
    November 12, 2014 at 10:20 pm

    Good read on learning distributed systems

30. *Karthik* says:
    November 13, 2014 at 11:29 am

    Henry,
    A very nice post. I notice that Google Dremel is missing form the list. Any particular reason why you excluded Dremel ?

31. *Learning distributed systems | null pointers* says:
    December 29, 2014 at 10:56 pm

    […] Distributed systems theory for the distributed systems engineer […]

32. *Links & reads from 2015 Week 13 | Martin's Weekly Curations* says:
    March 31, 2015 at 1:07 pm

[…] Oldie but goldie: Distributed systems theory for the distributed systems engineer […]

33. *Balkrishnan* says:
    May 11, 2015 at 9:41 pm

    Great page Henry! This was the first link that showed up when I searched for “how to learn distributed computing theory” 🙂

34. *Distributed Systems: Getting Started Resources | Björn Holdt* says:
    June 4, 2015 at 7:22 pm

    […] Academic Papers to read […]

35. *Data Engineering Reading Materials: Spark, Machine Learning, and Distributed Systems Resources | Denny Lee* says:
    July 11, 2015 at 12:01 pm

    […] about distributed systems theory and called out Henry Robinson’s blog post of great papers Distributed systems theory for the distributed systems engineer.   Some fun ones that I would include in the list […]

36. *广州信用卡还款* says:
    July 29, 2015 at 12:06 am

    争做文明上中人敬爱的老师，亲爱的同学们：　　大家好！　　蓝天和白云的心一样，希望沈阳信用 沈阳信用卡垫还 卡垫还白鸽自由地翱翔。　　老师和父母的心一样，喜欢我们健康地成长，　　今天我演讲 沈阳信用卡套现 的主题是：做文明上中人。　　华夏大地，礼仪之邦，几千年源源流沈阳信用卡套现长是我们祖祖辈辈，世世代代传承下来的文明礼仪。文明礼仪是种美，令人惊艳，令人追求，这是一种吸引人的美，它光彩夺目。文明礼仪给生命带来动力，给生命带来激情，这是一种可以吸引人灵魂的美。　杭州信用卡代办　校园因什么而文明呢？校园因你我的文明而文明。文明 杭州信用卡代办 是路上相遇时一个灿烂的微笑，是同学有困难时一个帮助的手，是见到师长时一声问好，是不小心撞到他人时一声对不起，文

37. *Distributed systems links* says:
    September 5, 2015 at 2:22 pm

    […] that there is a big gap between academic and industry knowledge right now. This is discussed in a post on Henry Robinson’s excellent Paper Trail blog where he provides a guide to digging deeper […]

38. *Interesting Links | Ikenna Consulting* says:
    November 3, 2015 at 11:47 am

    […] Distributed Systems Theory for the Distributed Systems Engineer […]

39. *A reading list on Microservices Architecture | DSEC* says:
    December 27, 2015 at 3:31 am

    […] Distributed systems theory for the distributed systems engineer A blog post on The Paper Trail with many links relevant for distributed systems engineering. This will keep you busy for some time. […]

40. *The Seven Deadly Sins of Microservices (Redux) | Voxxed* says:
    January 26, 2016 at 4:04 am

    […] The vast majority of microservice-based applications are going to be running as distributed systems, and therefore we must respect this, both as developers and operators. As developers we need to code defensively, and utilise fault tolerant patterns, such as time-outs, retries, circuit-breakers and bulkheads. Michael Nygard’s excellent ‘Release It!’ book should be mandatory reading, as should a visit to the Netflix OSS tooling Github repository! I’ve also put several links in the presentation to distributed systems articles I have found useful, such as “Distributed systems theory for the distributed systems engineer” […]

41. *learner* says:
    March 18, 2016 at 5:41 am

    Thank you for this!
    Link to Skeen’s original paper seems to be dead.

42. *BibSonomy :: url :: Distributed systems theory for the distributed systems engineer : Paper Trail* says:
    April 3, 2016 at 11:35 am

    […] http://the-paper-trail.org/blog/distributed-systems-theory-for-the-distributed-systems-engineer […]

Leave a Comment

Name: *Required* Email: *Required, not published* Homepage:

Comment:

Post Comment

« Previous Post

Elsewhere

Search