The Clock That Ran Away: What Became of Time, Clocks, and the Ordering of Events

The Intellectual Genealogy of Lamport's 1978 Paper

Intellectual Genealogies Project

August 2026 — draft v0.1

Concept by Carlos Baquero · Execution by Claude agents

Seed paper: Leslie Lamport. "Time, Clocks, and the Ordering of Events in a Distributed System." *Communications of the ACM* 21(7), July 1978, pp. 558–565. doi:10.1145/359545.359563.

An intellectual genealogy follows ideas through history, not just papers through a citation network.

Abstract. In July 1978, Communications of the ACM published an eight-page paper by Leslie Lamport with four references, two of them relativity textbooks. The paper defined what "before" means in a system with no common physical time, gave an algorithm for logical clocks, showed how to totally order the events of a distributed system, and — in a passage most readers missed — described how a totally ordered log of commands can replicate any service. Forty-eight years later its citation curve has never declined: 9,110 citing works, a plateau near 300 per year from 2000 to 2019, and still around 200 per year today. Its concepts now run in language specifications, databases, and blockchains, mostly without citation. In this essay we reconstruct the fates of the paper's parts. The definitional preamble became a universal concept; the passage presented in passing became the highest-impact line; the result the paper called new was never taken up. History inverted the paper's own ranking of its contributions.

1 The Original Intervention

Let us set the scene. It is the ARPA-net era, and the Operating Systems section of Communications of the ACM, edited by R. Stockton Gaines, publishes an eight-page paper on what "before" and "after" can possibly mean when computers only communicate by exchanging messages (Lamport 1978). The author, Leslie Lamport, is at Massachusetts Computer Associates; the address line on the paper reads SRI.

A genealogy usually begins with a birth record, and this paper carries its own, in the acknowledgment: "The use of timestamps to order operations, and the concept of anomalous behavior are due to Paul Johnson and Robert Thomas." That is RFC 677 (Johnson and Thomas 1976), the 1976 work on maintaining duplicate databases — the same RFC that our previous genealogy (Report 0) traced as the source of the last-writer-wins register. One document from 1976 seeded two lineages that are usually told as separate stories; we will come back to this in the final section.

The generative insight, however, came from physics, and here we have the author's own testimony: "I happen to have a solid, visceral understanding of special relativity...This enabled me to grasp immediately the essence of what they were trying to do" (Lamport n.d.). The paper does not hide the debt. "This definition will appear quite natural to the reader familiar with the invariant space-time formulation of special relativity," it says of its central relation, and two of its four references are Schwartz's *Relativity in Illustrations* (Schwartz 1962) and Taylor and Wheeler's *Space-Time Physics* (Taylor and Wheeler 1966). Half of the bibliography of what would become one of the most cited papers in computer science consists of physics textbooks.

The analogy is not decoration; it does real work. In the invariant formulation of special relativity there is no universal present: observers can disagree on the order of distant events, and what all of them agree on is only the causal order, the order that signals could have established. Replace signals with messages and the distributed system is in the same predicament. A node has no way of knowing what "now" means at another node; the only order it can trust is the one that messages could have carried. That is the essence Lamport grasped immediately, and it is why the definition needs no clock to state.

Now the contents, in the order the paper presents them. First, the definition: *happened-before* is the smallest relation that respects the order of events within each process, orders the sending of each message before its receipt, and is transitive. Events unrelated by it are concurrent — not simultaneous, simply unordered, because no chain of local steps and messages connects them. There is no physical time anywhere in this definition; "before" is reconstructed entirely from what the system itself can observe. Second, the algorithm: logical clocks, counters satisfying the Clock Condition — if $a \rightarrow b$ then $C(a) < C(b)$ — and maintained by two simple implementation rules, IR1 and IR2. Third, the extension: breaking ties between equal clock values by an arbitrary ordering of processes (the paper's own qualifier is "somewhat arbitrary") turns the partial order into a total one. Fourth, an illustration of what a total order buys: a distributed mutual-exclusion algorithm in five rules. And then, almost as an aside, the passage that would matter most: "The synchronization is specified in terms of a State Machine ... Each process independently simulates the execution of the State Machine." Any service, replicated on every node, by feeding every replica the same totally ordered log of commands. Fault tolerance is explicitly deferred — "the failure of a single process will make it impossible for any other process to execute State Machine commands," and "the problem of failure is a difficult one, and it is beyond the scope of this paper." The conclusion promises: "A future paper will show how this approach can be extended to solve any synchronization problem."

The second half of the paper deals with what it calls anomalous behavior. A person issues a request at computer A, then makes a telephone call, and a second request B is issued elsewhere; the system, knowing nothing of the call, may order B earlier. Causality flowed through a channel the system cannot observe. The example draws the boundary between the causality a system can track and the causality that runs outside it, and everything in the second half follows from taking that boundary seriously. Fixing this requires the Strong Clock Condition, and the remedy is physical clocks kept in synchrony: conditions PC1 and PC2, a rule that clocks are never set back, and a theorem bounding how far the clocks can drift apart, $\varepsilon \approx d(2\kappa\tau + \xi)$, proved in an appendix. Of this theorem, and only of this,

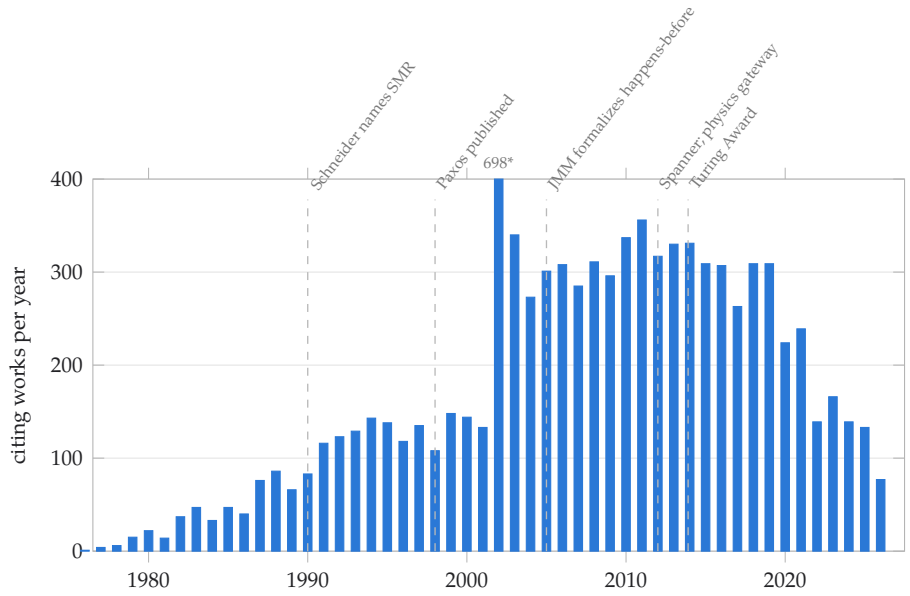

Figure 1: Citing works per year for the merged cluster (9,110 works, OpenAlex), 1978–2026. The 2002 bar (698*) is clipped: it is inflated by a cluster of venue-less, misdated records in the source database. Markers are intellectual events from the genealogy's timeline. 2026 is a partial year.

the paper claims novelty: "we believe this theorem to be a new result."

Read this way, the paper carries its own ranking of its contributions. The definitions are offered as natural; the mutual-exclusion algorithm is an illustration; the state machine is a passing remark; the theorem is the declared contribution. Keep this ranking in mind. History will not.

2 The First Descendants

Counting the descendants is less obvious than it looks. The 1978 paper — the seed, as we will call it from here on — was reprinted in a 2019 ACM Books anthology, and the anthology's citation record turns out to be 89% database artifact: 1,759 of its 1,982 recorded citers predate 2019, and a paper cannot cite a reprint that does not yet exist. Merging the cluster recovers 645 distinct works and yields our corpus: 9,110 unique citing works (Figure 1). Anyone doing quantitative history of science on top of raw citation databases should expect surprises of this kind.

The earliest wave, up to 1983, counts 146 works, and its composition is instructive: it is a scatter, not a school. Database concurrency control dominates — Bernstein and Goodman's 1981 survey (Bernstein and Goodman 1981), multiversion timestamp ordering. Thomas's majority consensus paper of 1979 is there (Thomas 1979), the sibling from RFC

677. So are security protocols (Merkle 1980), distributed elections (Garcia-Molina 1982), deadlock detection, version vectors (Parker et al. 1983), and distributed estimation (1982). Ricart and Agrawala (Ricart and Agrawala 1981) optimize the mutual-exclusion algorithm, treating the illustration as the contribution. Five years in, the paper was already being read for many different things — just not, as we will see, for the thing its author considered central.

What was it being read for, exactly? Of the citing works, 5,818 carry citation contexts, roughly 14,000 sentences in all. This is a cap-truncated sample, so the percentages that follow should be read as portraits rather than measurements. The intent labels split into 3,998 background, 1,629 methodology, and only 62 result citations: this is a paper cited for its concepts, almost never for a result. The distinction matters for what follows. A result can be superseded; a concept, once it enters a community's vocabulary, can only be renamed. Much of this genealogy is the story of those renamings. Among the concepts, happened-before (or happens-before) appears in 29.0% of the contexts, logical or Lamport clocks in 19.1%, and the state machine in 6.5%. The author has been keeping score himself: "This is my most often cited paper. Many computer scientists claim to have read it. But I have rarely encountered anyone who was aware that the paper said anything about state machines" (Lamport n.d.). And elsewhere: "People seem to think that it is about either the causality relation on events in a distributed system, or the distributed mutual exclusion problem."

The era-by-era numbers refine the picture (Figure 2). The state-machine reading stood at 2.4% of contexts in 1978–89, fell to 1.2% in the 1990s, then rose to 5.8% in the 2000s, 8.1% in the 2010s, and 7.7% in the 2020s — a revival whose cause we will identify in Section 4. Meanwhile the eponym grew without pause: "Lamport clock" appears in 418 contexts and "Lamport timestamp" in 187, terms the paper itself never uses, and the eponym share climbs monotonically from 3.5% to 10.0% across the eras. And the founding physics? Across those fourteen thousand sentences, relativity is mentioned 17 times and space-time 49. In reception, the framing that generated the paper is extinct.

3 The Branching

By a branch we mean a sustained line of research that took something from the seed and grew its own canon, venues, and vocabulary around it. A caution before counting: the branches below were assigned at survey depth, from metadata rather than full texts, so the figure of ten should be read as a map, not a census. With that reservation, the descent organizes into ten lines (Table 1).

	emerged	relation	eclipse of the seed*
SOURCES OF THE SEED			
timestamps, anomalous behavior — Johnson & Thomas, RFC 677	1976	documented	
special relativity — a discipline as ancestor	—	documented	
clock-synchronization literature — Ellingson & Kulpinski	1973	documented	
fault-tolerant multiprocess work — Lamport, ref. [3]	1978	documented	
BRANCHES OF THE LINEAGE			
databases: timestamp ordering — absorbed early	1979→	documented	absorbed
mutual exclusion — closed into text-books	1981→	documented	39%
causality theory: version vectors, vector clocks — the co-canon line	1983→	documented	24–25%
snapshots & global states — Flink checkpoints today	1985→	documented	67%
virtual time, optimistic simulation — detached field	1985→	documented	87%
clock-sync engineering — estranged; HLC 2014 recombines	1985→	documented	36–88%
causal broadcast & consistency — CATOCS critique 1993	1987→	documented	
SMR & consensus — Schneider names it; Paxos, BFT, blockchain	1990→	documented	75–79%
memory models & race detection — happens-before	2005→	indirect	final hops broken
temporal networks, physics — one-hop gateway	2012→	indirect	99.8%
DORMANT — CONTRIBUTIONS THAT NEVER GREW			
the clock-sync theorem — the claimed novelty	—	dormant	
the fairness footnote	—	dormant	
OUTSIDE THE LINEAGE			
event structures — Winskel 1986; independent Petri-net roots	1986	contact documented	
Bitcoin's longest-chain ordering	2008	no descent claimed	

Table 1: The genealogy at a glance — a view of the evidence ledger. Relations use the ledger's own vocabulary, not citation links; the ancestry includes a discipline, special relativity, alongside the documents. *Eclipse of the seed: the share of the branch canon's citers that does not cite the seed cluster, where a canon was measured (bar drawn at the range's upper bound; the clock-sync range spans HLC at 36% to NTP at 88%; details in the text).

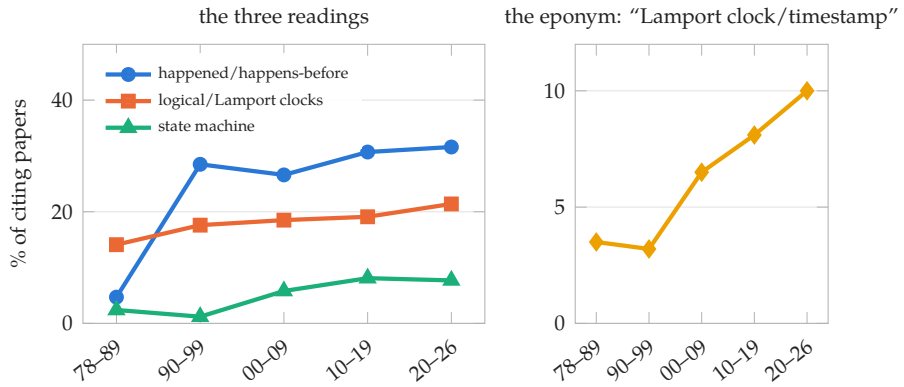

Figure 2: Readings of the seed in citation contexts, by era (share of context-bearing citing papers whose citing sentences mention each concept; bin sizes 85–2,435 papers; the S2 context data is a cap-truncated sample). Left: the causality reading dominates throughout; the state-machine reading nearly disappears in the 1990s and triples after 2000. Right: the eponym — vocabulary the paper never uses — grows without pause.

1. **Causality theory.** Vector clocks (Fidge 1988; Mattern 1989), preceded by Parker's version vectors (Parker et al. 1983); Charron-Bost; the school of Raynal, Garg, and Kshemkalyani. This is the faithful line: both vector-clock papers cite the seed, and even so, 25.0% and 23.7% of their own citers already cite them without it.

2. **Causal broadcast and causal consistency.** ISIS and Birman's protocols (Birman and Joseph 1987); causal memory (Ahamad et al. 1995); a revival in the COPS era. The line carries its own controversy, the 1993 CATOCS critique by Cheriton and Skeen (Cheriton and Skeen 1993) — which cites the seed and gathered 211 citations of its own.

3. **State machine replication and consensus.** Schneider's 1990 tutorial (Schneider 1990) names the origin: "The state machine approach was first described in Lamport [1978a] for environments in which failures could not occur." Then Paxos (Lamport 1998), PBFT (Castro and Liskov 1999), Raft, HotStuff and Tendermint, ZooKeeper (Hunt et al. 2010) and etcd. The passing remark became an industry.

4. **Mutual exclusion.** The illustration spawned a research line of its own (Ricart and Agrawala 1981), which matured into the textbooks and closed.

5. **Snapshots and global states.** Chandy–Lamport (Chandy and Lamport 1985). The Flink documentation today: "Flink uses a variant of the Chandy-Lamport algorithm known as asynchronous barrier snapshotting" (*Apache Flink documentation: Fault Tolerance via State*

Snapshots n.d.). A 1985 abstraction, running in production stream processors.

6. **Virtual time and parallel discrete-event simulation.** Jefferson (Jefferson 1985) and Time Warp; a field of its own, and one that left home early — 87.1% of its citers do not cite the seed, with a median year of 2003.

7. **Clock synchronization engineering.** Byzantine clock synchronization (Lamport and Melliar-Smith 1985), NTP (Mills 1991). Here too, 87.5% cite without the seed; the 1978 theorem lies dormant in the very field it anticipated.

8. **Memory models and race detection.** The relation migrated into programming languages; Section 4 follows it there.

9. **Temporal networks.** The idea returned to physics; Section 4 follows that journey too.

10. **Databases and transactions.** The earliest wave; the ideas were absorbed so quickly that the branch dissolved into the background of the field.

Even at survey depth, one pattern is visible and will recur at close range in what follows: the theoretical lines kept citing the origin, while the engineering lines left home early — and, in time, stopped writing home altogether.

4 Migration and Transformation

Concepts do not travel unchanged. Following them across community borders, we can identify seven recurring transformations.

Eponymization. Already visible in the reception numbers: the concept acquired its maker's name exactly as it detached from his paper. Like the volt and the hertz, "Lamport clock" and "Lamport timestamp" honor a scientist whom most of their users could not cite — and, as noted above, neither term was coined by the paper, which says "logical clocks" and nothing more. The name travels on its own; the reference stays behind.

Mutation of tense. In the paper the relation is *happened-before*: events that had already occurred, somewhere else, once. In programming languages it became *happens-before*: a rule that must hold in every execution of a program. The Java memory model foundation paper (Manson et al. 2005) cites the seed. The C++ memory-model paper (Boehm and Adve 2008) does not. The Java Language Specification, SE 21, §17.4.5 (*The Java Language Specification, SE 21, §17.4.5 (Happens-before Order)* n.d.),

defines "hb(x, y)" with no attribution anywhere in the chapter. In our contexts, happens-before now sits at parity with happened-before — 1,162 occurrences against 1,164 — with the crossover in the 2010s. The mutated tense marks the migration, from the description of a distributed computation to the specification of a programming language.

The state-machine revival. Reception caught up with the author's intent only after 2000, when Paxos and PBFT made fault-tolerant replication practical, and it was Schneider's tutorial that carried the attribution across the gap. The citers who read the 1978 paper as a state-machine paper are exactly the consensus lineage — Paxos, PBFT, Zyzzyva, HotStuff, Tendermint — and they cite the seed and the tutorial together, as a pair. The paper had to wait 22 years for the reader it was written for.

The physics round-trip. Reception shed the relativity framing almost immediately, as Section 2 showed. Decades later the idea re-entered physics through a single gateway, Holme and Saramäki's *Temporal networks* (Holme and Saramäki 2012), which credits: "the vector [φ...] is called i's vector clock. This framework was introduced by Lamport [90] and further developed by Mattern [104]" — attributing to Lamport the vector construction that belongs to Fidge and Mattern. Of the gateway's 2,654 citers, 99.8% do not cite the seed, and the physics citers in our corpus concentrate in the 2010s. An idea born from special relativity returned to physics carrying the wrong birth certificate.

Canon substitution. The intermediaries replace the origin. Of Schneider's citers, 74.8% do not cite the seed; of Paxos's, 78.9%; the median citing years are 2013–16. Hyperledger Fabric, ZooKeeper, and the blockchain surveys cite the canon without the seed. This is how a paper stops being read while its content spreads faster than ever: each generation cites the previous one, and the chain loses its first link. There is nothing improper in it — citing the nearest ancestor is exactly what good scholarly practice recommends. The forgetting is a property of the chain, not a fault of any author in it.

Recombination. Spanner (Corbett et al. 2012) orders transactions by synchronized physical clocks with bounded uncertainty; its "external consistency" is exactly the problem the 1978 paper posed with the Strong Clock Condition and the telephone call. Spanner cites 37 works, and the seed is not among them; if there is descent here, it runs through the clock-synchronization engineering literature. Hybrid Logical Clocks (Kulkarni et al. 2014), by contrast, cite the seed and set out to fuse the paper's two halves, logical and physical clocks — the two halves that reception had kept apart for decades, reunited by engineering need.

Dormancy. And the declared contribution? Among the 5,818 context-bearing citers, 190 use clock-synchronization vocabulary at all, and 19 mention the bound, the drift, or the theorem. Dormant is not quite dead —

the concerns of the second half kept resurfacing, as Spanner's engineering shows — but the theorem itself sleeps. The one result the paper believed new is the one part of it that was never taken up.

5 The Lost Citation Chain

Languages are full of words whose etymology no speaker carries — we all say "goodbye" without thinking of "God be with ye" — and the words work anyway. Something similar happens at the far ends of this genealogy, and it is worth walking out to the endpoints to see it.

The Java Language Specification is the cleanest case. A language specification consulted by millions of programmers defines "hb(x, y)" — the seed's relation, in mutated tense — and the chapter contains no attribution to anything. This is not carelessness by one author; it is the normal condition of specifications, which state rules rather than histories. The race-detection community shows the mechanism in miniature: FastTrack (Flanagan and Freund 2009), a canonical paper, cites the seed; the community's leaf tools — Goldilocks, RoadRunner, DPOR — cite Mattern without it. Memory here is two-tiered: the canon remembers, the leaves forget. The physics gateway of Section 4 is the extreme case — a single hop crossed a discipline and lost the ancestor, misattribution included. And Spanner runs a production answer to the paper's second-half problem, under the paper's own problem statement in different words, with a bibliography of 37 entries that does not reach it.

Genealogy also needs boundary discipline, because not every absence is a lost chain. Winskel's event structures (Winskel 1986) cite the seed: the true-concurrency tradition has documented contact with this lineage, not a broken link. And Bitcoin's longest-chain ordering shares none of this lineage's apparatus; that is a non-case, not an absorption. The genealogist must resist the temptation to find the ancestor everywhere. The claim here is narrower, and better documented: where the concepts demonstrably descend from the seed, the citation usually does not follow them all the way down.

6 Convergence, Fusion, and Reinterpretation

Lineages do not only branch; they collide, merge, and quarrel. The causal-systems line had its quarrel in public: the 1993 CATOCS critique, Cheriton and Skeen's argument against causally and totally ordered communication (Cheriton and Skeen 1993), sits inside the citation graph itself, the critique citing the very seed whose descendants it attacked. Whichever side one takes, the point for the genealogist is that the dispute is traceable: the

graph preserves the arguments as well as the doctrine. A genealogy that showed only reverence would not be credible; this one records its disputes.

The most consequential event is a fusion. The seed's total order, Schneider's fault-tolerance framing, and Paxos's algorithm merged into a single canon — state machine replication as one composite idea, with the joints no longer visible. No one who deploys a replicated log today needs to know which of the three parts came from where. The blockchain era consumes this canon as infrastructure, the way a building consumes electricity, and cites accordingly: the canon, not the origin. Hybrid Logical Clocks perform the complementary fusion, reuniting the paper's own two halves that reception had split apart.

By 2013 the fused legacy had a name and a prize. The Turing Award citation — we quote the wording as reproduced by Wikipedia from the ACM text (*Leslie Lamport — Wikipedia (quoting the ACM Turing Award citation)* n.d.) — credits "the invention of concepts such as causality and logical clocks, safety and liveness, replicated state machines, and sequential consistency." Causality and logical clocks; replicated state machines. The preamble and the aside, side by side at the head of the list. The theorem is not mentioned.

7 Decline, Survival, or Canonization

Citation curves normally rise, peak, and fade. This one rose and stayed: in 48 years there is no decline on record, with the plateau near 300 citations per year through 2000–2019 and still around 200 per year today. Formal canonization traced the same arc: the first PODC Influential Paper Award in 2000 (the award was later renamed the Dijkstra Prize), the SIGOPS Hall of Fame in 2007, the Turing Award in 2013, the anthology reprint in the ACM Books series in 2019. The theory community, the systems community, and the field at large each canonized the paper in their own house.

But canonization does not preserve a paper whole; it selects. Ranking the parts by their fates: happened-before is universal, at the price of a mutated tense; logical clocks survive under their maker's name rather than the paper's; vector clocks are the faithful heirs, citing and cited; the mutual-exclusion line closed honorably into the textbooks; the theorem is dormant; and the relativity framing is extinct in citation practice, alive only in retrospectives — including the author's own. Each part met a different kind of survival: as vocabulary, as eponym, as textbook material, as dormancy, or as memory. The paper was canonized as a whole; its parts were triaged.

8 The Intellectual Legacy

We can now state the inversion in full. The paper's parts met three fates, in reverse order of the paper's own emphasis. The definitional preamble — offered as something that "will appear quite natural" — became a universal concept, present in the working vocabulary of at least five communities. The generalization of the illustration, the state-machine passage presented in passing, became the consensus industry that underlies replicated databases and blockchains. The claimed novelty, the one result introduced with "we believe this theorem to be a new result," went dormant. Whatever a paper says about its own contributions, history reserves the right to re-rank them.

The transmission mechanism is the same in every branch, and we have named it two-tier memory. The canons remember: the Java memory model paper, FastTrack, Schneider, Holme and Saramäki, Hybrid Logical Clocks, and Winskel all cite the seed. The leaves forget: the Java Language Specification, C++11, the race-detection tools, the physics papers, the blockchain systems, and Spanner do not. Attribution survives in the layer that writes surveys and tutorials, and evaporates in the layer that ships software and defines standards. The eclipse also deepens with age: in Report 0's genealogy the intermediaries eclipse their seed at 22–54%; here the range runs from 24% to 99.8%.

There is a lesson here for anyone who measures science by citation counts. The counts see only the layer that remembers; the layer where the ideas do their daily work — specifications, production systems, another discipline's models — is invisible to them. By citation metrics alone, the deepest impact of this paper would be undetectable. Impact and attribution are correlated at the canon and decoupled at the leaves, and the decoupling grows with time.

One more observation, across genealogies rather than within one. RFC 677 fed both this lineage and Report 0's; Dynamo sits in both corpora; the causal-consistency line links them. The genealogies of a field are not separate trees but one interlocking structure, and tracing a second one begins to show the grain of the wood.

The closing fact is the simplest one. This paper has been cited some nine thousand times, and yet its deepest presences in the world — a relation in a language specification, a database's external consistency, a physics field's time-respecting paths — carry no citation at all. A concept has fully succeeded when it no longer needs its reference. The clock ran away from its maker. That, after all, is what good clocks do.

Provenance

This essay is built on 23 ledger claims, of which 22 are verified and 1 remains open — the ten-branch synthesis, held at survey depth. Confidence grades: 9 documented, 14 strong. The standing hedges, restated: branch membership was assigned from metadata, so branch counts and boundaries are indicative; the citation-context data (S2) is a cap-truncated sample, so context percentages are portraits, not measurements; and the Turing Award wording is taken from Wikipedia's quotation of the ACM citation. The full research notes, data pulls, and evidence ledger accompany this document as research provenance. Margin ledger references can be rendered by compiling with the `provenance` option.

Sources

Ahamad, Mustaque et al. (1995). "Causal memory: definitions, implementation, and programming". In: *Distributed Computing* 9.1, pp. 37–49. DOI: 10.1007/bf01784241.

Apache Flink documentation: Fault Tolerance via State Snapshots (n.d.). Accessed 2026-08-19. URL: https://nightlies.apache.org/flink/flink-docs-stable/docs/learn-flink/fault_tolerance/.

Bernstein, Philip A. and Nathan Goodman (1981). "Concurrency Control in Distributed Database Systems". In: *ACM Computing Surveys* 13.2, pp. 185–221. DOI: 10.1145/356842.356846.

Birman, Kenneth P. and Thomas A. Joseph (1987). "Reliable communication in the presence of failures". In: *ACM Transactions on Computer Systems* 5.1, pp. 47–76. DOI: 10.1145/7351.7478.

Boehm, Hans-J. and Sarita V. Adve (2008). "Foundations of the C++ concurrency memory model". In: *ACM Conf. on Programming Language Design and Implementation (PLDI)*, pp. 68–78. DOI: 10.1145/1375581.1375591.

Castro, Miguel and Barbara Liskov (1999). "Practical Byzantine fault tolerance". In: *USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, pp. 173–186.

Chandy, K. Mani and Leslie Lamport (1985). "Distributed snapshots: determining global states of distributed systems". In: *ACM Transactions on Computer Systems* 3.1, pp. 63–75. DOI: 10.1145/214451.214456.

Cheriton, David R. and Dale Skeen (1993). "Understanding the limitations of causally and totally ordered communication". In: *ACM Symposium on Operating Systems Principles (SOSP)*, pp. 44–57. DOI: 10.1145/168619.168623.

Corbett, James C., Jeffrey Dean, Michael Epstein, et al. (2012). "Spanner: Google's globally-distributed database". In: *USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, pp. 251–264.

Fidge, Colin J. (1988). "Timestamps in message-passing systems that preserve the partial ordering". In: *11th Australian Computer Science Conference*, pp. 56–66.

Flanagan, Cormac and Stephen N. Freund (2009). "FastTrack: efficient and precise dynamic race detection". In: *ACM Conf. on Programming Language Design and Implementation (PLDI)*, pp. 121–133. DOI: 10.1145/1542476.1542490.

Garcia-Molina, Héctor (1982). "Elections in a Distributed Computing System". In: *IEEE Transactions on Computers* C-31.1, pp. 48–59. DOI: 10.1109/tc.1982.1675885.

Holme, Petter and Jari Saramäki (2012). "Temporal networks". In: *Physics Reports* 519.3, pp. 97–125. DOI: 10.1016/j.physrep.2012.03.001.

Hunt, Patrick et al. (2010). "ZooKeeper: wait-free coordination for internet-scale systems". In: *USENIX Annual Technical Conference*.

Jefferson, David (1985). "Virtual time". In: *ACM Transactions on Programming Languages and Systems* 7.3, pp. 404–425. DOI: 10.1145/3916.3988.

Johnson, Paul R. and Robert H. Thomas (1976). *The maintenance of duplicate databases*. Internet RFC 677, Information Sciences Institute.

Kulkarni, Sandeep S. et al. (2014). "Logical Physical Clocks". In: *Int. Conf. on Principles of Distributed Systems (OPODIS)*. LNCS 8878, pp. 17–32. DOI: 10.1007/978-3-319-14472-6_2.

Lamport, Leslie (n.d.). *My Writings (annotated bibliography)*. Annotation for "Time, Clocks...". Accessed 2026-08-19. URL: https://lamport.azurewebsites.net/pubs/pubs.html.

— (1978). "Time, Clocks, and the Ordering of Events in a Distributed System". In: *Communications of the ACM* 21.7, pp. 558–565. DOI: 10.1145/359545.359563.

— (1998). "The part-time parliament". In: *ACM Transactions on Computer Systems* 16.2, pp. 133–169. DOI: 10.1145/279227.279229.

Lamport, Leslie and P. Michael Melliar-Smith (1985). "Synchronizing clocks in the presence of faults". In: *Journal of the ACM* 32.1, pp. 52–78. DOI: 10.1145/2455.2457.

Leslie Lamport — Wikipedia (quoting the ACM Turing Award citation) (n.d.). Accessed 2026-08-19; the ACM awards page blocks automated access. URL: https://en.wikipedia.org/wiki/Leslie_Lamport.

Manson, Jeremy, William Pugh, and Sarita V. Adve (2005). "The Java memory model". In: *ACM Symposium on Principles of Programming Languages (POPL)*, pp. 378–391. DOI: 10.1145/1040305.1040336.

Mattern, Friedemann (1989). "Virtual Time and Global States of Distributed Systems". In: *Int. Workshop on Parallel and Distributed Algorithms*, pp. 215–226.

Merkle, Ralph C. (1980). "Protocols for Public Key Cryptosystems". In: *IEEE Symposium on Security and Privacy*, pp. 122–134. DOI: 10.1109/sp.1980.10006.

Mills, David L. (1991). "Internet time synchronization: the network time protocol". In: *IEEE Transactions on Communications* 39.10, pp. 1482–1493. DOI: 10.1109/26.103043.

Parker, D. Stott et al. (1983). "Detection of Mutual Inconsistency in Distributed Systems". In: *IEEE Transactions on Software Engineering* SE-9.3, pp. 240–247. DOI: 10.1109/tse.1983.236733.

Ricart, Glenn and Ashok K. Agrawala (1981). "An optimal algorithm for mutual exclusion in computer networks". In: *Communications of the ACM* 24.1, pp. 9–17. DOI: 10.1145/358527.358537.

Schneider, Fred B. (1990). "Implementing fault-tolerant services using the state machine approach: a tutorial". In: *ACM Computing Surveys* 22.4, pp. 299–319. DOI: 10.1145/98163.98167.

Schwartz, Joseph T. (1962). *Relativity in Illustrations*. New York University Press.

Taylor, Edwin F. and John A. Wheeler (1966). *Space-Time Physics*. W. H. Freeman.

The Java Language Specification, SE 21, §17.4.5 (Happens-before Order) (n.d.). Accessed 2026-08-19. URL: https://docs.oracle.com/javase/specs/jls/se21/html/jls-17.html.

Thomas, Robert H. (1979). "A majority consensus approach to concurrency control for multiple copy databases". In: *ACM Transactions on Database Systems* 4.2, pp. 180–209. DOI: 10.1145/320071.320076.

Winskel, Glynn (1986). "Event structures". In: *Petri Nets: Applications and Relationships to Other Models of Concurrency*. LNCS 255, pp. 325–392. DOI: 10.1007/3-540-17906-2_31.