The art of stopping

De kunst van het stoppen

Judith van Steen†

Dear all,

I dedicate this lecture in part to my recently deceased sister, Judith van Steen. Her passing has not only deeply affected me, but has also helped me to formulate the message of this lecture more clearly.

I will come back to this later.

En even in het Engels...

Let me start with expressing my gratitude to you all for joining this farewell lecture, and in particular to those who have come from far. I owe all of you a lot. Many, many thanks.

As announced, the lecture will be in Dutch, but the slides are in English, and with a bit of luck, you will see a reasonable translation into English using a live transcription. Take a shot of the QR code, and it should work.

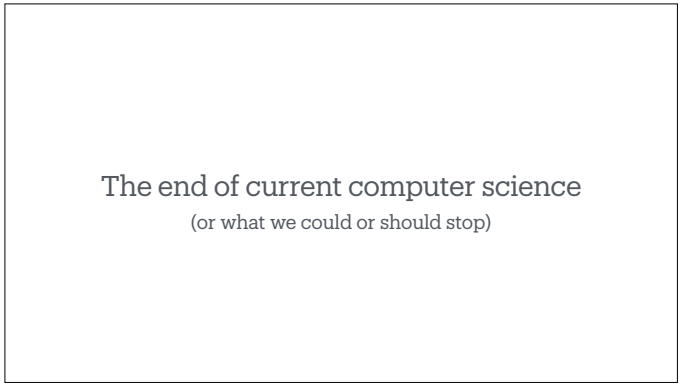
Meanwhile, I'll take a picture of you....

I want to talk about three endings:

- This lecture. It will last about 40 minutes, so that should be manageable. The protocol prescribes that I talk and you listen (or at least pretend to). Of course, I couldn't let such an opportunity pass me by.

- The end of a scientific career. Well, if I start talking about that, you know that the end of the lecture is also near.

- But the most important one is the end of current computer science. I will nuance that statement, and for those who know me well enough, that nuancing usually happens in stages, after I have launched a somewhat bold statement. And sometimes the nuances never come.



The end of current computer science. Or rather: what, in my view, should we consider more critically? Should we really continue with certain topics? And if not, then what should we pursue instead?

Well, if that isn't already quite a substantial nuance…

Let me start by taking a look at my own field of expertise.

Large-scale distributed systems

(my field of research)

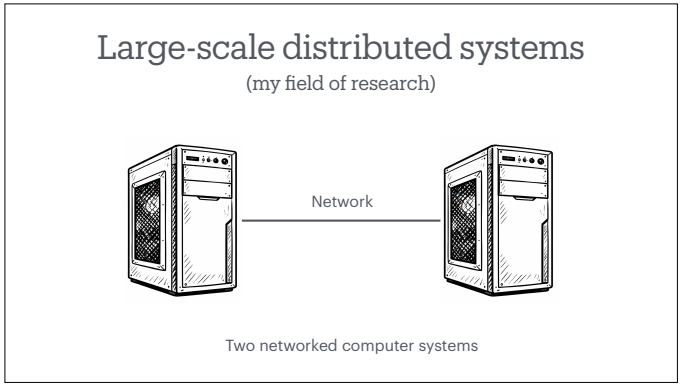
Computer system

For many years now, I have been doing research and teaching on so-called large-scale distributed computer systems.

Here you see a simple computer system, familiar to many of you: a desktop computer.

By the way, this is not a particularly modern version, but that doesn't matter for the moment.

One computer system is nice; two are even nicer.

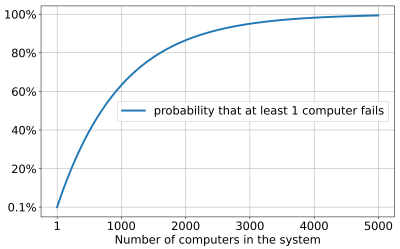


But once you have more than one, trouble begins. You don't keep multiple computers running for nothing:

- two can do more than one (compute faster)
- if one fails, the other can take over
- when things get busy, you can distribute the workload

But of course, more things can also break, and computers are remarkably good at the art of stopping. In fact, they are so good at it that sometimes you don't even realize that they've stopped working. And I haven't even mentioned failing network connections yet.

Failing computers



And to give an idea:

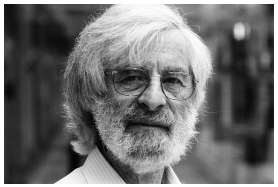
- Assume that the probability that a computer fails within a day is 0.1% (that’s not very much, right?).
- With 5,000 computers in your system, the probability that at least one fails each day is 100%.

In other words: there's always something broken!

It does make you wonder why BigTech builds such huge data centers…

Distributed (computer) system

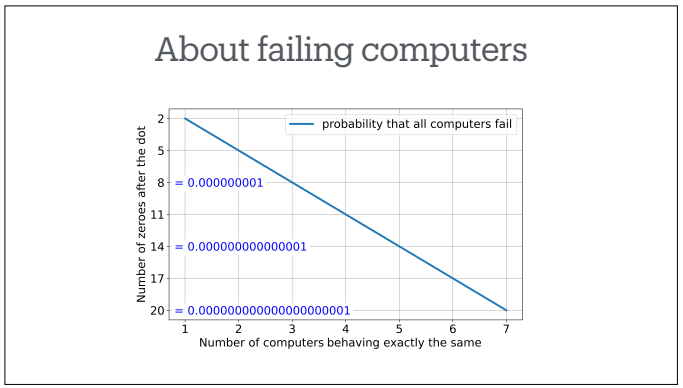
A definition



A distributed system is one in which the failure of a computer you didn't even know existed can render your own computer unusable.

Leslie Lamport

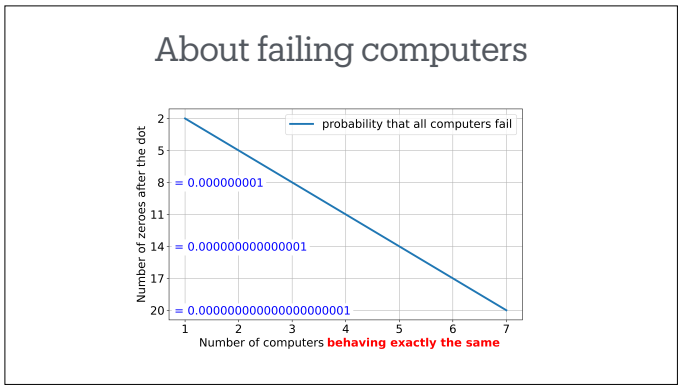
This led Leslie Lamport, a famous computer scientist long associated with Microsoft, to give the following insightul definition



But there is also another perspective.

Suppose a number of computers do exactly the same thing. That is convenient, because if one stops working, you can immediately continue with another, as if nothing has happened—until none of them work anymore. If the daily probability of failure is again 0.1%, it turns out that the probability that they all stop working becomes very small very quickly.

You can see that in this graph, where I have represented the probability by the number of zeros after the decimal point. Remember that 0.1% equals 0.001, which is why with one computer you see two zeros after the decimal point. What you also see is that with three or five computers, the probability that they all fail is already almost zero. In other words, the probability that at least one of them is still working is almost 100%.



The problem, of course, is ensuring that computers actually do exactly the same thing.

The same Leslie Lamport once wrote a beautiful report about this, which ended up in a drawer somewhere until colleagues told him he really should publish it. We are talking about 1989.

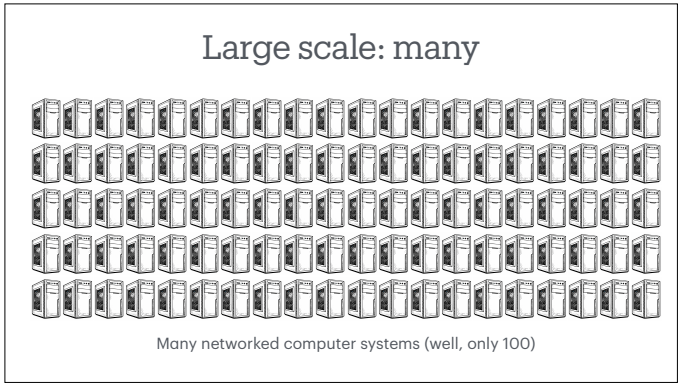
Once again it turned out to be one of his brilliant ideas, but so complicated that twenty years later scientists came up with alternatives—not so much to make it better, but to make it more understandable, so that you could at least be a bit more certain that your implementation actually did what it was supposed to do.

Being able to tolerate failing computers and other components is one of the most important topics of my field.

I'll leave you with the apparent contradiction that if you have many computers, something is always broken, but that with a small number of computers doing exactly the same thing you can achieve very high reliability. And if you're thinking, "Well then, just let all 5,000 computers do exactly the same thing," I have to disappoint you: the system would come to a grinding halt.

Large scale

In any case, there are three ways to look at large-scale systems, or scalability. The simplest one is that it means “a lot.”

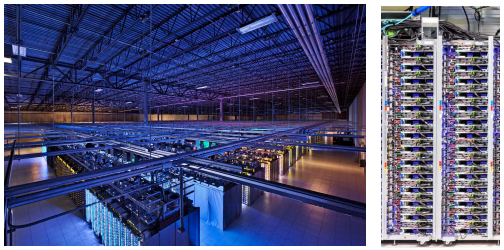


Here's an an image. Incidentally, this is not that many—just 100.

It is generally assumed that Google has somewhere between tens and several hundreds of thousands of computers in a single data center. That quickly adds up to millions of computers worldwide that are—indeed—all connected to each other.

Large scale: many

Inside one of Google's data centers



Luiz Barosso et al.: The data center as a Computer (2026)

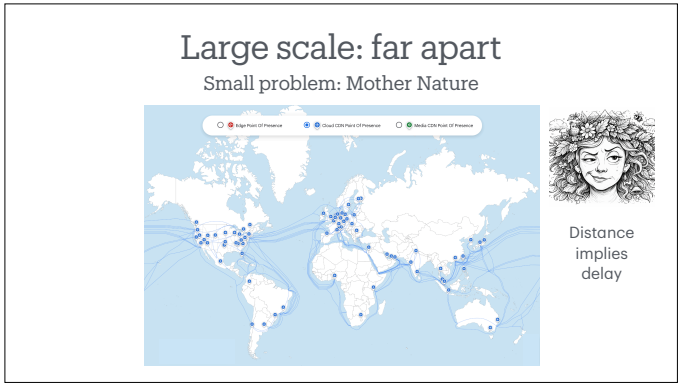
Here you see an impression of very many computers in one of the Google data centers: rows of racks, with each rack easily containing 40–50 computers.

That “very many” is what most people understand by scalability.

Yet this type of scalability is by far the easiest. By now we understand it well, and there are many techniques for achieving this kind of scalability.

And sometimes it simply turns out to be impossible to scale something up—but by now we also understand why that is the case.

For example, trying to make 5,000 computers always do exactly the same thing. I will actually show an important exception to this later.

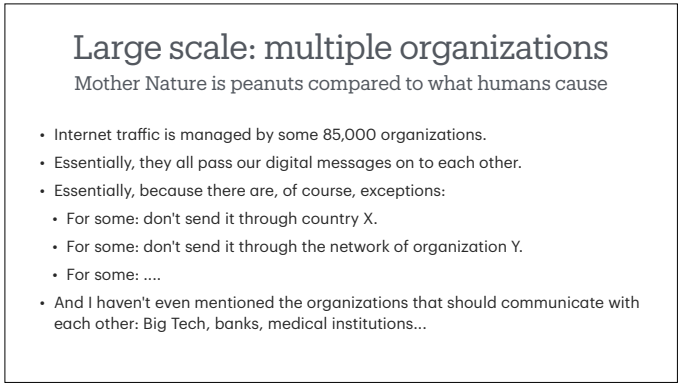


Another dimension of scale is the distance between computers. And that is where we do have a problem. While Mother Nature is usually a friendly being, she is not your best friend when it comes to communication networks.

Distance simply means obeying her laws, and those impose delays. It's not just about the speed of light, but also about delays caused by intermediate computers along the way.

What you see here again comes from Google and roughly shows the physical connections between different data centers. Communication between centers that are far apart can easily add up to tens of milliseconds. That is not a disaster in itself, but it becomes one when interaction patterns are involved, such as in gaming or automated financial transactions on a stock exchange.

Where possible, there is really only one solution: move the whole setup closer. And if you look at Netflix, you can bet that your movie or series is stored relatively close to your home.



But scalability really becomes an issue when your distributed system has to cross organizational boundaries.

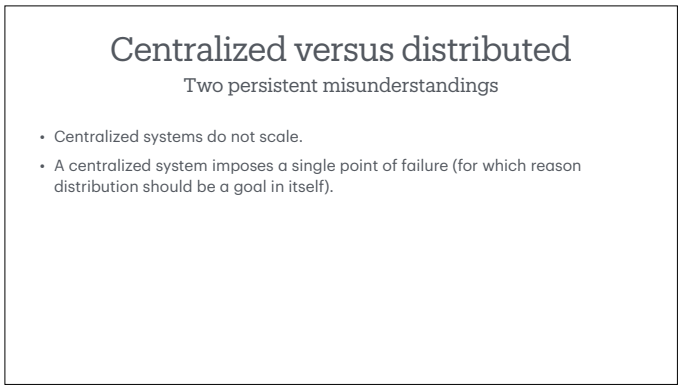
Then you have to deal with people. Terrible what they can all get up to. Compared to that, Mother Nature is nothing.

In fact, it is already close to a miracle that the Internet works as well as it does:

- It is managed by some 85,000 organizations.
- In principle, they all pass along our digital messages to each other.
- There are, of course, exceptions:
    - Some do not want their messages to pass through country X.
    - Others do not want them to pass through a network managed by organization Y.
    - Still others have different requirements…
- And that’s not even mentioning coordination between Big Tech, banks, medical institutions, and so on.

With some satisfaction, I can say that in my research I have always managed to keep people and organizations away—and where that was unavoidable, I managed to keep their role small.

Incidentally, keeping them away worked only in research. Outside of that it didn’t—but there it wasn’t the goal either.



Before I conclude this part, I would like to pause for a moment to discuss two very persistent misconceptions.

Misconceptions that have become so persistent that they have taken on a life of their own.

- Centralized systems do not scale.
- A centralized system is more vulnerable to failure than a distributed system, and therefore we must distribute.

Centralized versus distributed

Two persistent misunderstandings

- Centralized systems do not scale.
- A centralized system imposes a single point of failure (for which reason distribution should be a goal in itself).

Nothing so great as a centralized solution

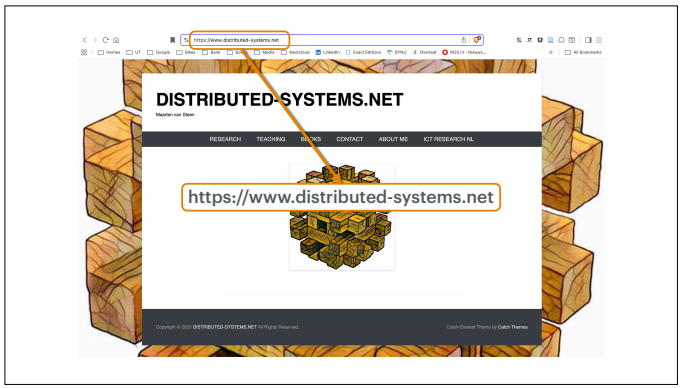
In contrast, I would argue that there is nothing quite as nice as a centralized solution: simple, manageable, fast, and scalable.

The mistake people make is that they confuse the means with the end.

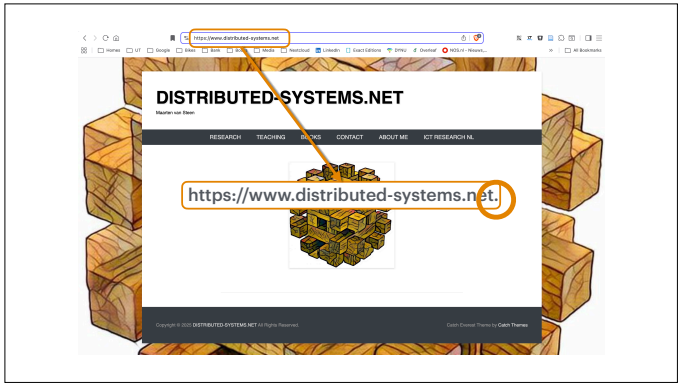
If the goal is to make something simple and manageable, then you should think in terms of centralized solutions.

If you also want that goal to work at large scale and be resilient to failures, then distribution is not such a bad idea.

But do not distribute more than is strictly necessary. Because distribution leads only to complexity. Distribution should never become a goal in itself.



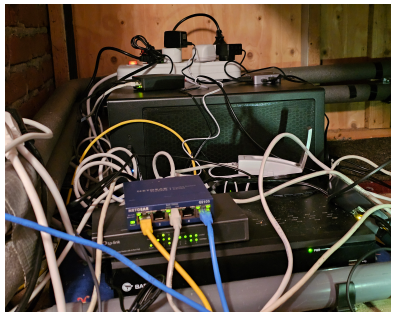
And to illustrate this difference, I would like to briefly take you into the world of URLs—like this one.



What is always missing here is the most important part: the dot!

The reason we have those URLs is to reach the server where that website is stored.

In this case, that is here, in the rural area outside Eibergen.



More precisely—this is it.

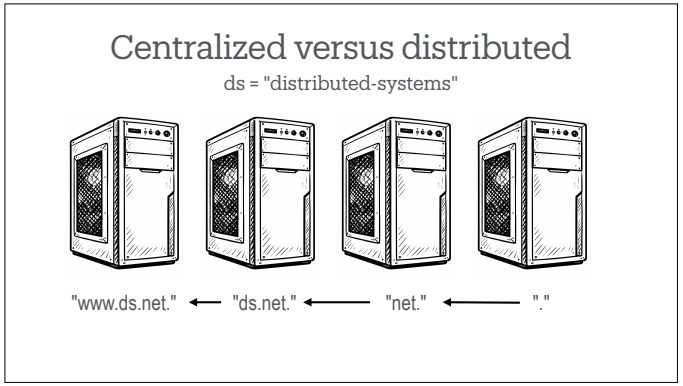
No, it doesn’t look quite like what you see at Google.

And yes, various provisions have been put in place here as well to prevent data loss and even to keep things running if the connection or something else fails.

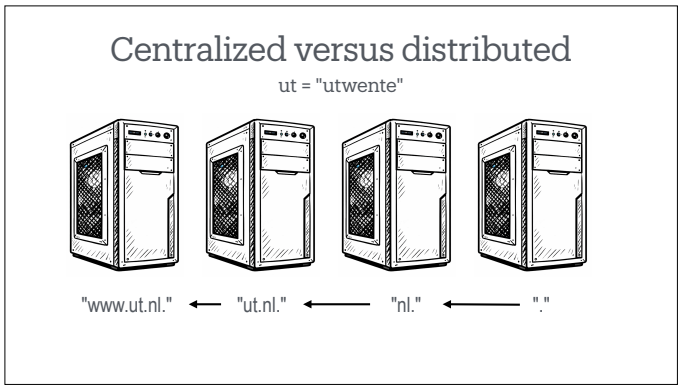
But not nearly as well as at Google.

By doing it this way, I discovered many things I didn’t know before. I will come back to that later.

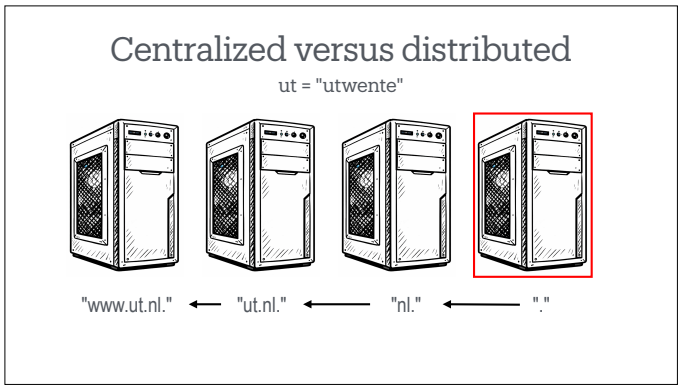
I should add that whenever I solved a problem, I tended to forget the solution again fairly quickly—so the next time I ran into the problem again, I had to think very hard about how I had tackled it previously.



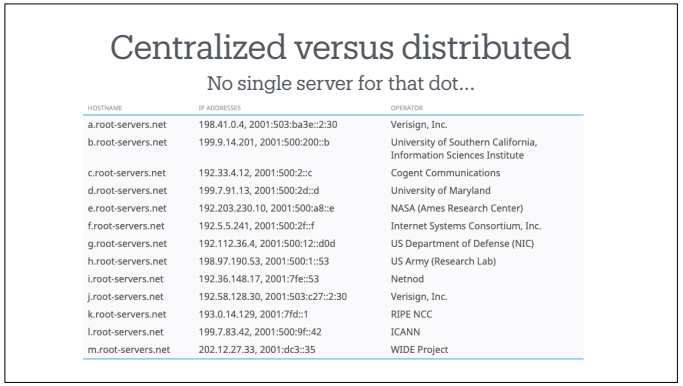
The idea is that you process the name in reverse order, starting with the final dot. That dot corresponds to a server that knows where to find the server responsible for “net”. That server knows where to find the server for “distributed-systems.net.”, which in turn knows where to find the server for “www.distributed-systems.net.”.



And the same applies to “www.utwente.nl.”, which actually runs on a completely different server than my site—and everyone, including me, is very happy about that.



Let’s take a moment to look at the server for the “dot.”



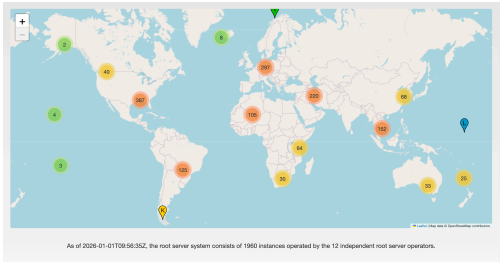
Oh, wait a moment. It's not just one—there are 13 of them. Incidentally, there is a somewhat awkward design decision here, because it seems that we can have at most 26 servers for the dot. I'll come back to that as well.

And something else that is striking: those 13 are managed by 12 different organizations. That is actually quite nice, because it spreads the responsibility a bit.

Sometimes it isn't such a bad idea to distribute things across multiple people or organizations after all...

Centralized versus distributed

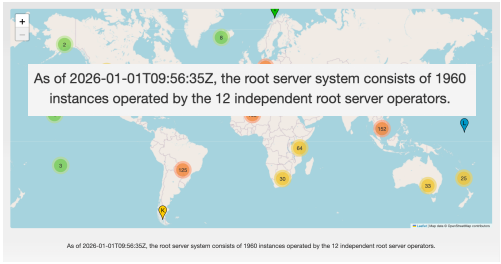
..... there about 2000, with instance = (cluster of) computer(s)



More precisely: those 13 are not single servers at all—they are roughly 2,000 clusters of servers.

Centralized versus distributed

..... there about 2000, with instance = (cluster of) computer(s)



All copies of each other, all doing exactly the same thing.

It scales extremely well here because those copies hardly ever change.

And on top of that they are spread across 12 different organizations.

Centralization as the goal, with just enough distribution to achieve scalability and fault tolerance—but distribution only as a means.

Long live the dot!

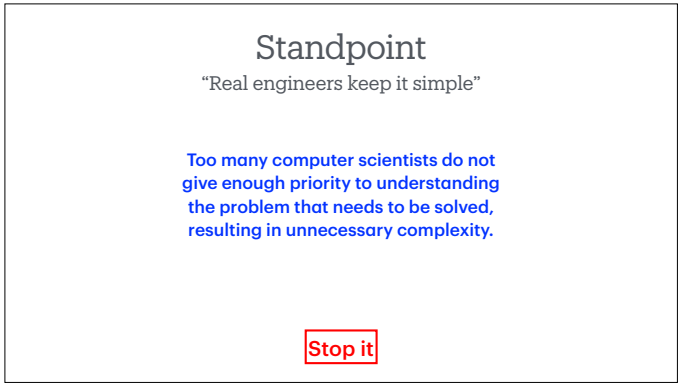
Centralized versus distributed

"In practice, centralized control is simpler to implement and generally yields more responsive control actions. At Google, we have tended towards centralized control models for much of our software infrastructure, unless it's impossible to do so ."

Luiz Barosso et al.: The data center as a Computer (2026)

And I'm not the only one who argues that striving for centralization is a good idea. They do the same at Google.

That brings me to the following proposition, which is important for giving substance to the art of stopping.



Years ago, and for many years, my slogan was: "real engineers keep it simple."

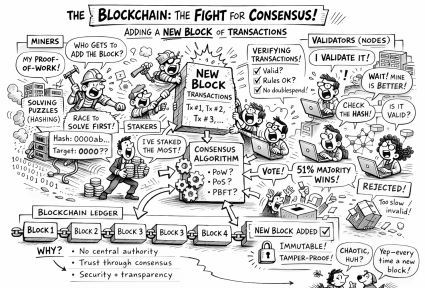
What I really care about is that too many computer scientists give too little priority to understanding the problem that actually needs to be solved, with the result of completely unnecessary complexity.

Stop it.

And let me give two examples from my own field. The first is blockchains.

Example: blockchains

- **Idea: provide a fully decentralized public ledger**:
  - No centralized authority that you would need to trust
  - Open for everyone to see which transactions have taken place



When you think of blockchains, just think of cryptocurrency.

The idea is that you want to have a public ledger of transactions, without having to trust a central authority such as a bank or a notary.

The rather complicated diagram reflects quite well, in my view, that there is a lot involved technologically—especially in the early solutions, which are still widely used.

Technically speaking, blockchains are quite a challenge.

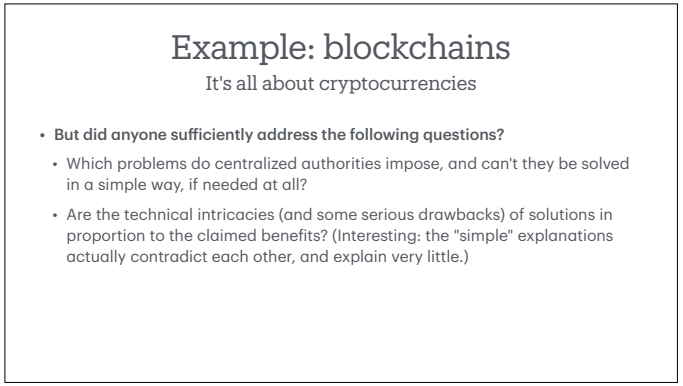
Example: blockchains

It's all about cryptocurrencies

- **But did anyone sufficiently address the following questions?**
  - Which problems do centralized authorities impose, and can't they be solved in a simple way, if needed at all?

But you do have to ask whether we have really looked carefully enough at a few issues:

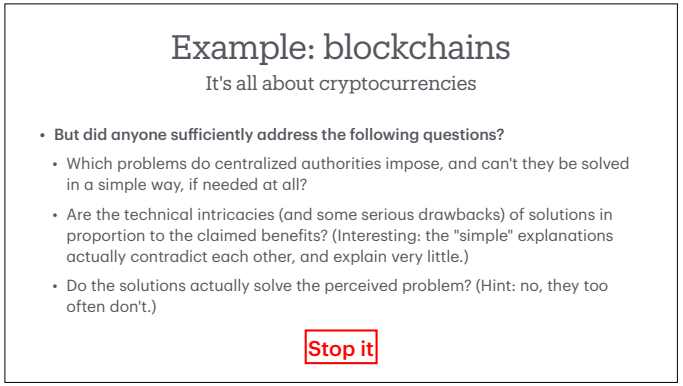
What problems do we actually have with a central authority, and could those not be solved in a simpler way? Who supervises the bank? Who supervises the notary? With URLs, we solved it.



It took me a long time to be able to explain blockchains. In other words, for a long time I simply didn't understand them. When I once started a discussion on LinkedIn with the question, "Shouldn't we start asking what problem blockchains are actually a solution for?", I got quite a backlash.

The best response came from someone who said, "Oh, another one who just doesn't get it!" That person was absolutely right—except that he didn't realize that he understood it even less than I did when he claimed that they had really achieved something wonderful at a municipality where people could anonymously participate in a project. However, you don't need to BCs for that. A solution in search of a problem.

There really is a problem when our solutions are technically so complicated that they can hardly be explained to the people who actually have to manage these systems. On top of that, the simple explanations you find on the internet contradict each other and, in fact, explain very little.



There does seem to be a use case for blockchains in situations where there is minimal mutual trust between a number of parties, combined with a simple refusal to appoint a central coordinator.

The question then becomes: how often does this actually occur? My suspicion is: far less often than we computer scientists think. Which raises the next question: couldn’t it then be done more simply?

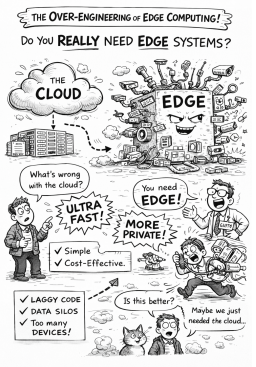
Stop it.

And then there is another one: so-called edge systems. BTW, this is a nice example where colleague Aaron Ding and I regularly disagree, yet at the same time organize discussions to getter a better grip of what the real problem is.

Example: edge systems

It's all about keeping stuff nearby

- **Idea: make sure that your data and computations are done close to you:**
  - Save on moving lots of data between where you are and a far-away data center.
  - Provide guarantees on minimal delays (remember Mother Nature?)
  - Assumption: keeping things close by are better for protecting privacy



The basic idea is to keep data and computation close to the end user.

That can save the cost of transporting data to and from a far-away data center.

It also gives you better guarantees when it comes to delays (I already mentioned Mother Nature earlier).

There is also a somewhat odd assumption: that everything close by is safer—especially when it comes to protecting privacy ("I always keep my backups with me").

Example: edge systems

It's all about keeping stuff nearby

- **But did anyone sufficiently address the following questions?**
  - Is there so much data to move around?
  - Is the connection to the data center really so bad?
  - How bad is that delay, and when is it critical?

But here as well, have we really looked carefully enough at a few simple questions?

Is there actually that much data to send to and from a data center?

Is the connection really that poor? I was recently on a train and, just for fun, measured the speed of my 5G connection. It was almost as fast as my fiber-to-the-home!

How serious is that delay, and when does it truly become critical?

Example: edge systems

It's all about keeping stuff nearby

- **But did anyone sufficiently address the following questions?**
  - Is there so much data to move around?
  - Is the connection to the data center really so bad?
  - How bad is that delay, and when is it critical?
  - Does keeping things close by really solve a privacy problem, or is there something else? (Hint: yes, and edge devices are not the solution.)

Stop it

Does keeping your own stuff with you really help protect your privacy? Of course not. It's about something else—namely that others cannot simply access your data. Edge systems do not solve that problem at all.

Note: unlike blockchains, there are indeed situations in which edge systems can help, or may even be necessary. Let's focus much more on those situations instead of putting forward questionable arguments just to justify working on new technology.

Stop it.

BighTech: some realistic assumptions

(when it comes to distributed systems)

- The best and largest distributed systems have been developed by Google, Amazon, Facebook, Microsoft, and others.
- A significant number of the best scientists in the field of distributed systems have moved to Big Tech.
- Many results from academia ultimately end up with Big Tech, if anywhere at all.
- Many of the most cited articles on distributed systems are (co-)authored by BigTech researchers.

What may also have become clear is that much of what I have discussed is already well covered by Big Tech. They have built the largest distributed systems and, in doing so, tackled a considerable number of the hardest problems.

Many scientists have by now found their way to Big Tech, such as the famous Leslie Lamport, but also prominent AI researchers.

I also see that a lot of work done by academics at universities eventually ends up being used by Big Tech—or nowhere at all.

And we have to acknowledge that many important scientific publications originate from Big Tech or from collaborations with them.

This raises an important question.

Question

Why should academic scientists, paid through public funding, do research that BigTech itself can do very well?

Note: I am not against Big Tech at all, and I am not against academic research that ends up benefiting Big Tech. On the contrary, I believe we should offer our students an education that allows them to choose whether or not they want to work for Big Tech.

But research for Big Tech funded with public money?

Stop it.

But what then?

But then what should we focus on instead? Here is a short fragment from a hearing in the United States Senate where Mark Zuckerberg is asked a question.

“We sell ads,” he says—with a slightly devilish smile…

To sell those ads effectively, Zuckerberg needs our data. Our personal data. He wants to know everything about us and does everything he can to learn as much as possible—so long as we remain glued to his screens. All of it to sell ads and become even richer.

And his apology for that “gluing us to the screen” in 2024 turned out to be worthless less than a year later, when he sent the fact-checkers home. And now he may have to answer in court for his engagement algorithms. But the terms for using his AI glasses also speaks for itself: you can’t even refuse the forwarding of images.

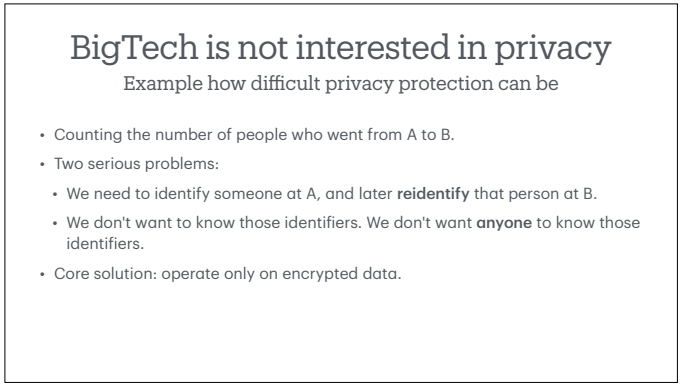
Facebook is an obvious example, but the same applies to Google and Amazon, and in a different way also to Microsoft. As a rule, they want to know everything about us.

BigTech is not interested in privacy

Big Tech is not interested in privacy.

And just to be clear: if you say you have nothing to hide, would you still be okay with your medical record ending up in the hands of Mark Zuckerberg? Or if you can address Google with “Hey Google,” are you certain about what else might be sent to their servers in the meantime? It’s not about hiding things; it’s about whether you can keep control over what you choose to share and what you don’t. That is something quite different. Experts have written extensively about this.

So what should we focus on instead? For example, more research on privacy—even though many technologies for privacy protection have already been devised. Combining them is more than just putting a few pieces together.



Let me illustrate this with a seemingly very simple problem: counting people who move from one location to another.

I have been working on this for some time. Not because I am interested in the fact that you can count, but in whether you can do it accurately while fully protecting privacy.

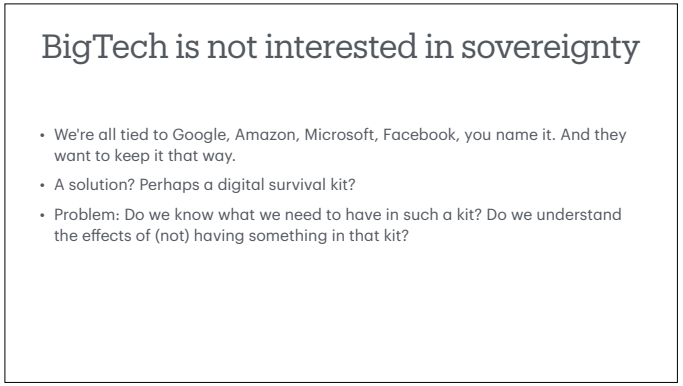
There are two essential problems:

- You need to be able to identify someone and recognize them later. That is not ideal when privacy is involved.
- We want to ensure that no one can discover those IDs—not even us.

The core of our solution is therefore: work only with and on encrypted data.

There were countless problems that we could only discover—and solve—through experimentation.

An interesting one concerns accuracy: if you are counting only a few people, you actually want the result to be quite imprecise. But with large numbers, you want the count to be precise.



Another important topic is how we can become more independent from Big Tech—because, regardless how you look at it, U.S. legislation ultimately has the final say there.

That is not what you want, yet it turns out to be remarkably difficult to move away from it. It requires much more attention, and certainly the attention of academic researchers.

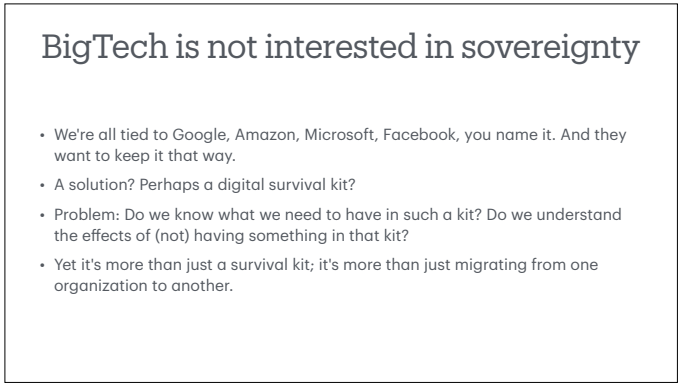
Here too the rule applies: only by doing things will we discover what we do not know, and discovering that is essential for finding solutions.

In the words of the ever-creative wordsmith and scientist Bart Jacobs (good to see you here, Bart): what should be in your digital survival kit?

Together with Radboud University and others (and I'm looking for Mark Dorenbusch from the CVD), we have devised an interesting experiment. We are simply going to disconnect volunteers from Microsoft.

When I suggested to Henk Swaters from our excellent IT department that we might also not announce in advance when that would happen, he did become a little nervous. Henk—are you ready for that now?

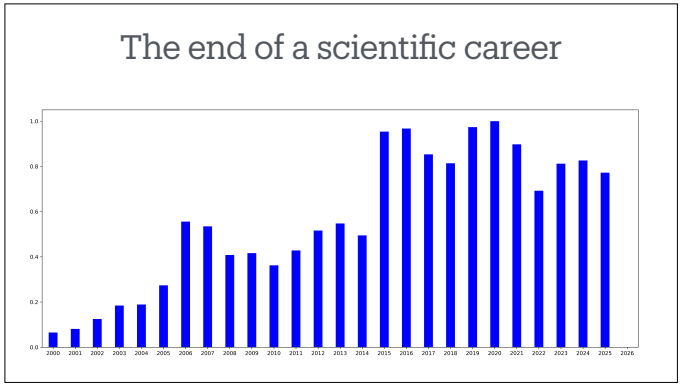
The message, of course, is this: you will never think of everything in advance. So just do it—and then see what you overlooked.



But while people mainly think about how to reduce dependence on American companies, the real issue should be how to switch more easily from one supplier to another. And to make things even more complicated: do you actually know which suppliers you depend on? Simply gaining insight into that is already a topic for scientists. I know that researchers at TNO are already trying to answer this question—and that it is a hell of a job. Digital sovereignty is a scientific problem.

So far I have mentioned only two examples from my own field that we could focus on, and several others that we should probably stop pursuing. I am convinced that if others also concentrate more on problems that Big Tech is not interested in, many more will emerge.

We could simply stop—and in doing so open the way for a renewed and inspiring research agenda for computer science.



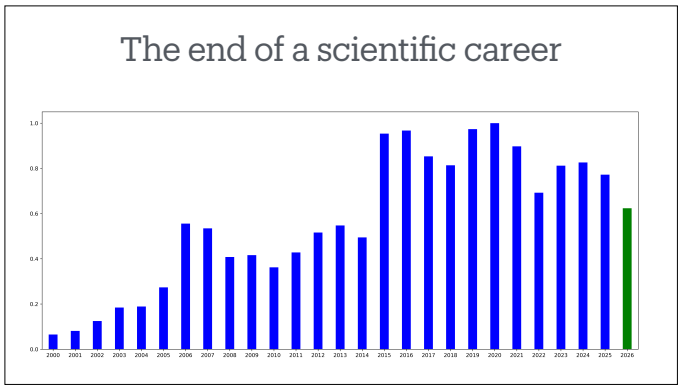
Does anyone have any idea what this is?

VU colleague Henri Bal once inspired me with the remark: “Before you delete away an email, first make a copy of it.” It might also have been about files, but that doesn’t really matter.

From around 2000 onward, I started automatically storing all my emails. What you see here is the relative number of emails per year that I explicitly—so manually—decided to keep. The absolute numbers don’t matter, and you wouldn’t want to know them anyway. But from this you can discover the course of my career.

In 2005 I became Director of Education at the Vrije Universiteit Amsterdam; in 2006 that started to make an impression on me. After I seemed to have been trained a bit, I became head of the Computer Science department at the VU in 2010, where about 200 people were working at the time. That went reasonably well for a while, but you can see that the number of saved emails increased quite a bit.

You can also see when I started working at the University of Twente (indeed, 2015). After that, things never improved. Oh yes—there was also 2022, when I had a sort of part-time sabbatical: two days a week working on a book, three days a week doing “normal” work. I would never, ever, ever recommend that combination to anyone. Sometimes you simply have to discover things the hard way.



But it’s still not entirely okay. In green you see the prediction of the number of emails, based on what I have saved so far this year. But no worries, it won't get worse.

Beyond the end

I have already been asked many times what I am going to do. My answer is always that keeping myself busy will not be a problem at all. My colleagues from the DSI team thought the same, and they put together a wonderful little book with personal cycling routes.

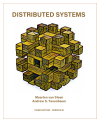
So I will certainly be doing some cycling. And quite a lot of it at times when most of you are at work.

Beyond the end

Working on the next edition without a deadline?


2001


2007



2017

2023


2029?

And to keep my brain cells active, I'm looking forward to finally taking the time to bring the series of the now worldwide used textbook on distributed systems to a fifth edition.

The nice thing is that I don't have to make any promises about when it will appear—or even if it will appear. The journey itself is already so rewarding.

Just being a student again.

Beyond the end, beyond zebra

There's no limit to how much you'll know,
depending how far beyond zebra you go.

But most importantly, I don't actually have to figure out what I'm going to do—trusting that the future will simply come to me.

Having grown up with Dr. Seuss, I couldn't resist pointing to the book On Beyond Zebra!, which asks the simple question: "Why should you stop at the letter 'Z'?"

It also offers a nice answer to the apparent limitation of those 13 servers for the missing dot that I mentioned earlier.

A big thank you

And for this slide I need to switch to English to do justice to some very special people. I have had the privilege of having to help a few, at that time, young colleagues with launching their careers after graduation. This list excludes the bonus PhD students who were mainly supervised by others and in which my role can only be marked as "modest."

Professors are called supervisors, but the fact is that over a period of some 4-5 years, and sometimes longer, the relationship between PhD student and supervisor is by far more symmetric than many of us realize. It is a lot of mutual learning, and I want to thank you, my dear PhD colleagues, for having given me the opportunity to grow and learn so much. Without you, I would not be standing here, and that is fact.

The VU team

Andy Tanenbaum

I would like to mention a few people explicitly. First of all, there is Andy Tanenbaum—not only the co-author of our book, but also someone from whom, even though I was not his PhD student, I could simply observe how to do science. And above all, he confirmed to me that there is no place for hierarchical relationships in science.

More generally, I would like to thank my colleagues at the Vrije Universiteit Amsterdam who were close by when things became a bit more intense. I would like to mention in particular Saskia Edixhoven, Hans Akkermans, Guszti Eiben, and Frank van Harmelen.

And then there is the DSI team! I also want to make sure to mention Ellen and Jorien as former members.

It is a very diverse team, with very different personalities, which can sometimes bring its share of awkward moments. At the same time, it has proven to be an incredibly strong team—precisely because whenever there was a gap, someone was always able to step in and fill it.

You are to a very large extent responsible for the tremendous pleasure I have had here at the University of Twente.

Thank you so much!

Special

Inald Lagendijk

Inald, you are one of the few colleagues who gradually moved into that twilight zone between colleague and friend—two worlds that I have always kept separate. But not with you.

Working together on all kinds of national, and often complicated, dossiers is one thing. Much more important to me is that, in choosing a rather solitary profession, I could rely on someone who understands perfectly well the world in which I operate.

Thank you for that, and I look forward to many more trips back and forth to your beautiful place in a little piece of no man’s land.

Special

Ben Verheijden

I would also like to mention Ben, who is now watching the livestream. Some people here know him as well. You have organizational experts who can also enter that twilight zone I just mentioned with Inald. And people like Ben usually say many sensible things. I've forgotten most of them, except this one:

"Maarten, you should start thinking more in terms of boundary conditions. And any solution that satisfies those is okay."

He spoke these now-legendary words when I was working at the Vrije Universiteit Amsterdam. They really landed—and very well too—and turned out to be the perfect match for the way I wanted to lead: simply **not** telling people what to do as much as possible.

But more than just for that, Ben—you have simply become very dear to me.

Those who you can't live without...

The art of stopping

...cannot do without...

openness to what you don't know

Finally, if after this lecture any of you are still wondering what the core message is, let it be this:

“The art of stopping cannot do without openness to what you don't know.”

The emphasis here is on “openness” and “not knowing.” And I truly do not mean this only in science—I mean it in everything. What you do not know cannot be planned. But openness cannot be planned either. It is an attitude.

If that openness truly succeeds, then what you do not know will simply come to you, like a friend. If it doesn't, the unknown can easily become an obstacle—or even a threat—that stands in the way of change and renewal.

I sincerely wish all of you to truly embrace the art of stopping.