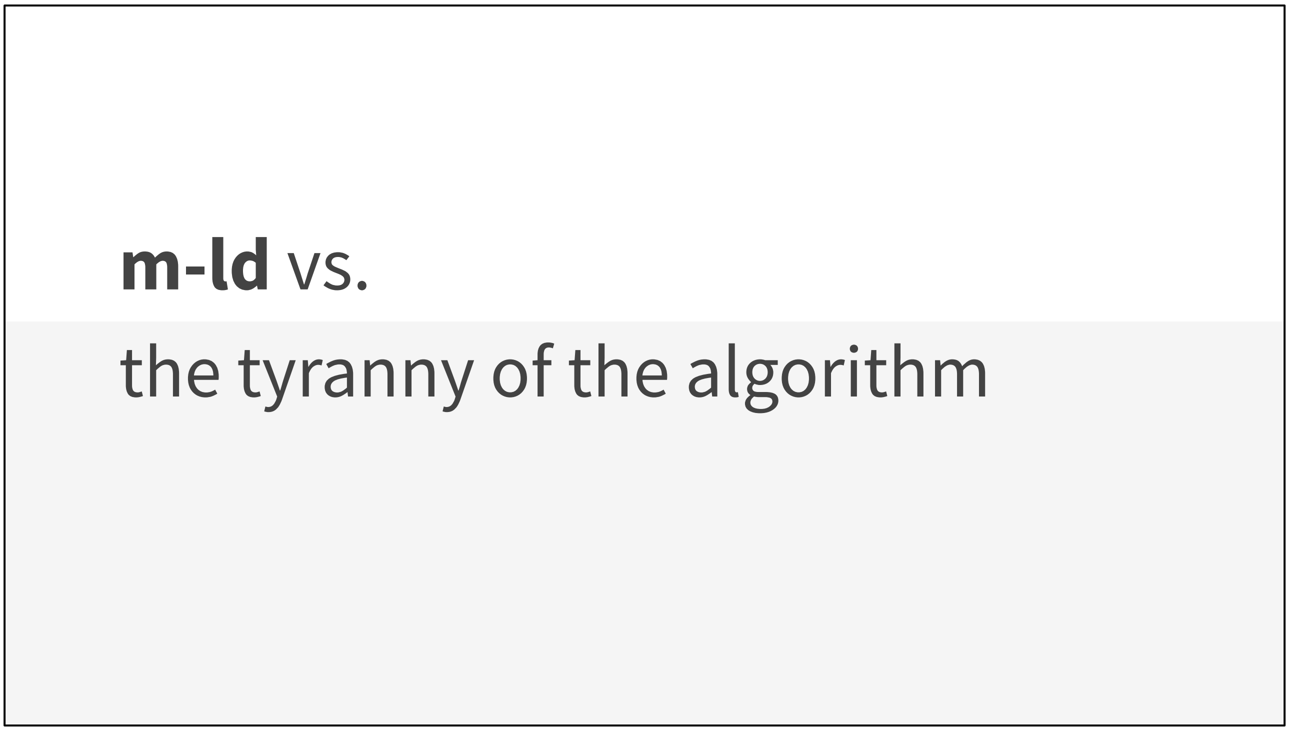
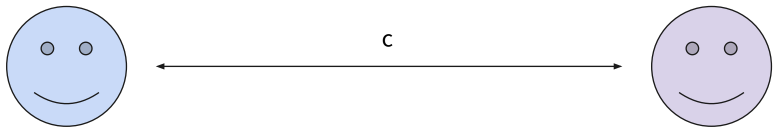


m-ld vs. the tyranny of the algorithm

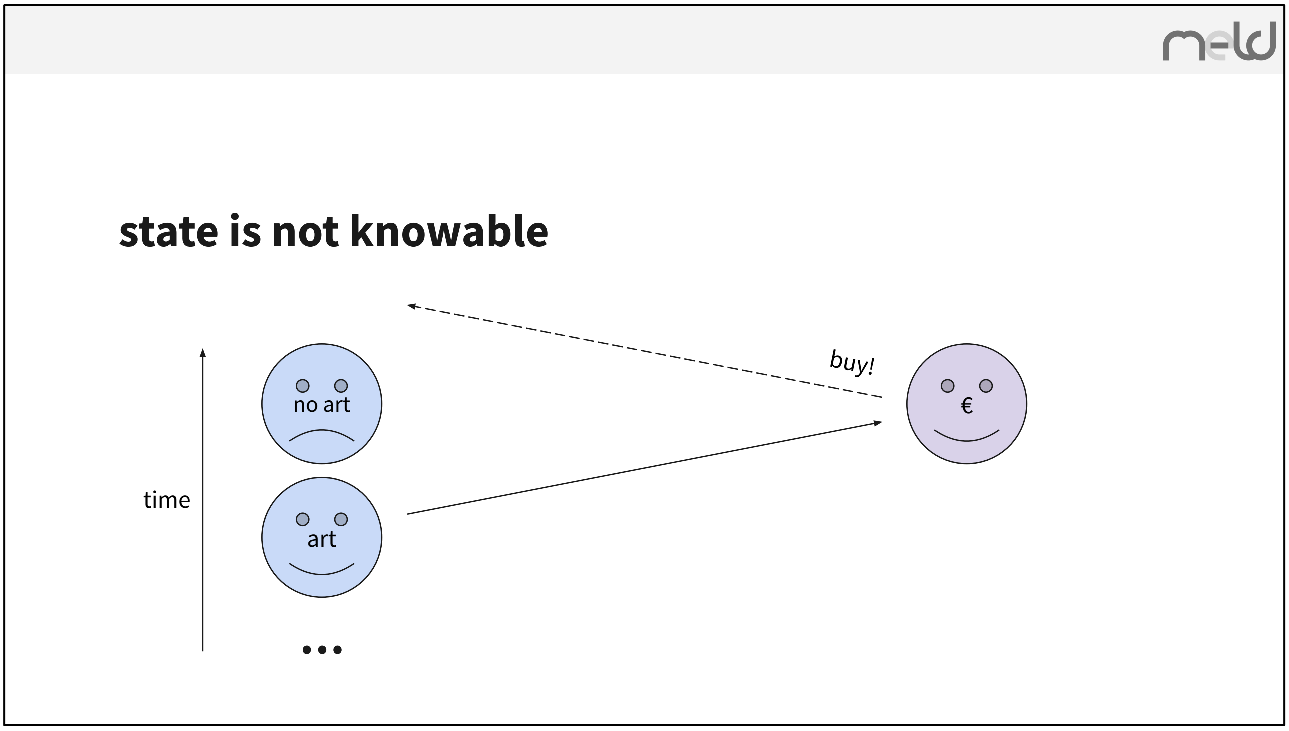
Welcome.

This talk is going to be about our software, called m-ld; about our project with NLnet and the NGI; about coordination in information systems and yes, about CRDTs. But it's also going to be about reality, and how the state of reality is a fun challenge to represent. I hope you enjoy it.

reality is not coordinated



Every *thing* in the universe exists in parallel. Information flows between things, limited by the speed of light. Two things separated by any distance can change concurrently without knowing about each other's change. If you want to know the state of something, you have to wait for the information to reach you. By then the state might have changed again. So reality fundamentally does not allow you to know the "current" state of anything.



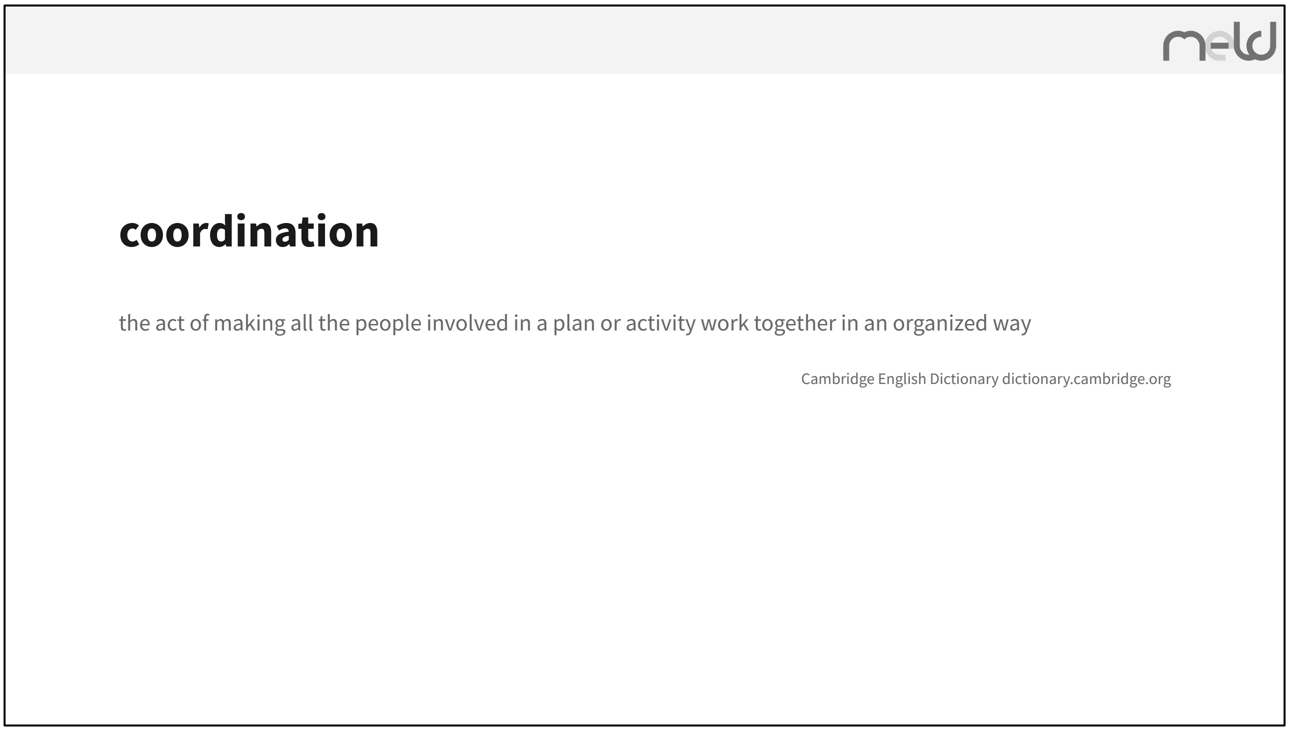
If we're talking about agents like humans and machines with dreams and goals, always existing in ignorance sounds like a problem. Knowing the present truth has utility for *future* behaviour.

Knowing state allows you to make decisions.

Let's say Blue is actually selling Purple some artwork. Purple wants to know that Blue still has the work; Blue wants to know that purple has the funds. Neither of these facts are knowable by the other participant.

There are a few ways forward, which we can think of intuitively. If a decision doesn't depend on anyone else you can just go ahead with it. If you're able to predict something else's state based on its *recent* state, that might be good enough. Blue and Purple could just assume that the last state they have seen is still accurate, and proceed with the sale.

But also, Blue might say something like, "I'll hold the work until you confirm you have the money." This is an example of *coordination*.

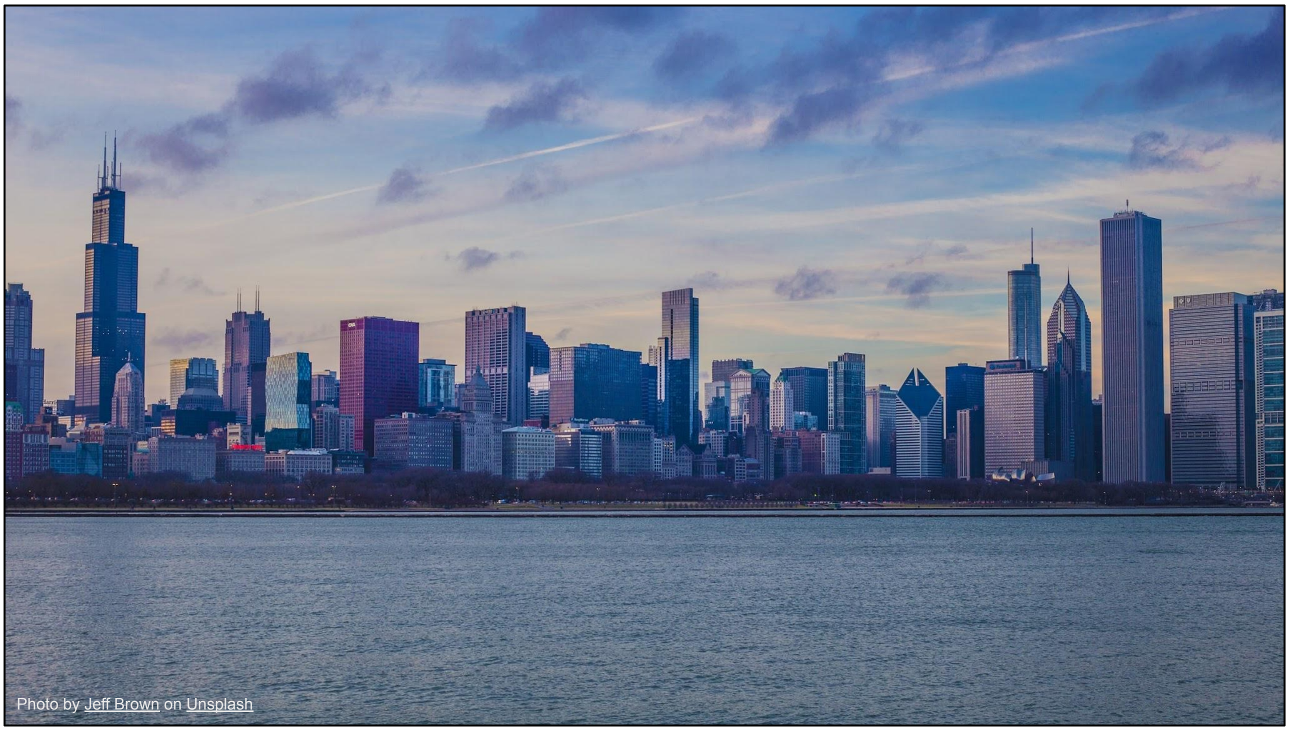


coordination

the act of making all the people involved in a plan or activity work together in an organized way

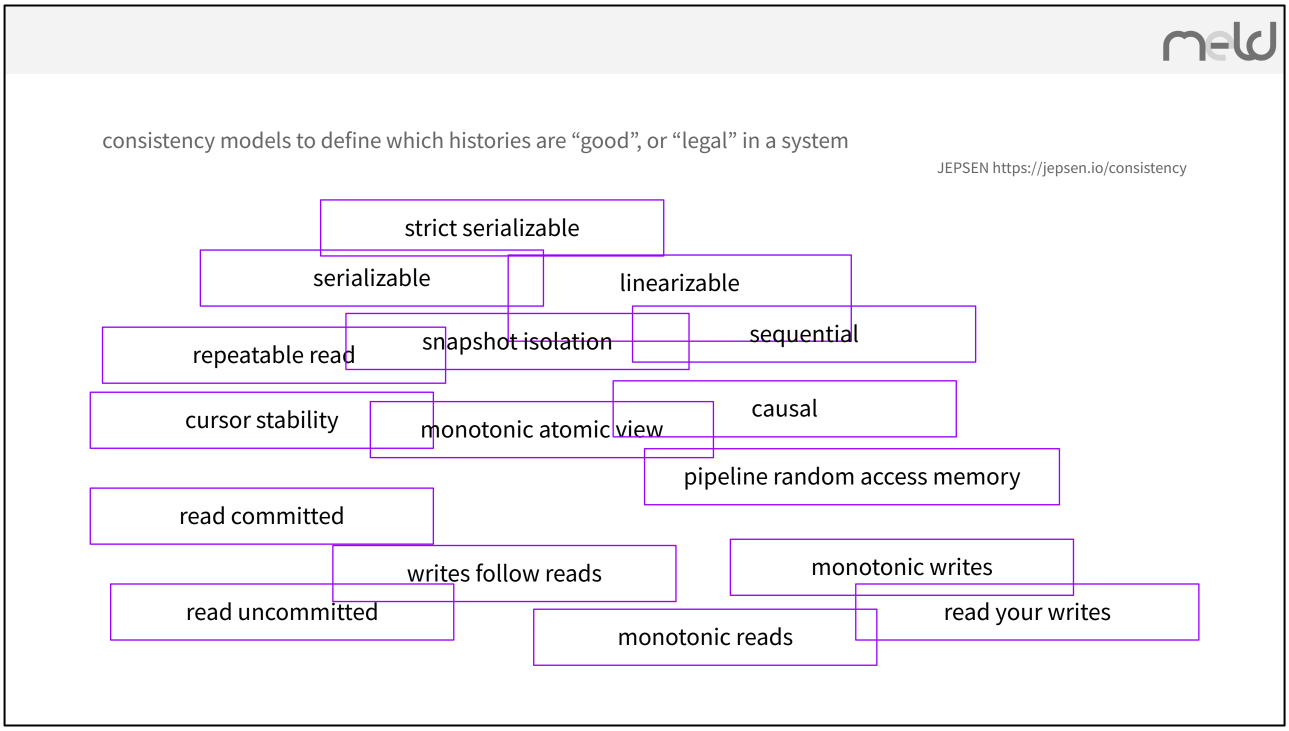
Cambridge English Dictionary dictionary.cambridge.org

Coordination allows agents in different locations and sometimes with different goals to be able to make decisions based on a common understanding of some important state.



With this tool, humans have created information systems, like political and financial systems, and, yes, databases, with elaborate rules. We use these systems to coordinate an enormous amount of value.

But coordination is expensive. It makes some people have to wait for other people, which slows things down, and compromises their sovereignty. Remember that Blue wasn't able to sell their artwork while waiting for Purple's confirmation – they had to *wait*. Coordination often requires data to be centralised somewhere. It's also brittle, because people can cheat – Blue could just break their promise and sell the art to someone else anyway.



consistency models to define which histories are “good”, or “legal” in a system

JEPSEN https://jepsen.io/consistency

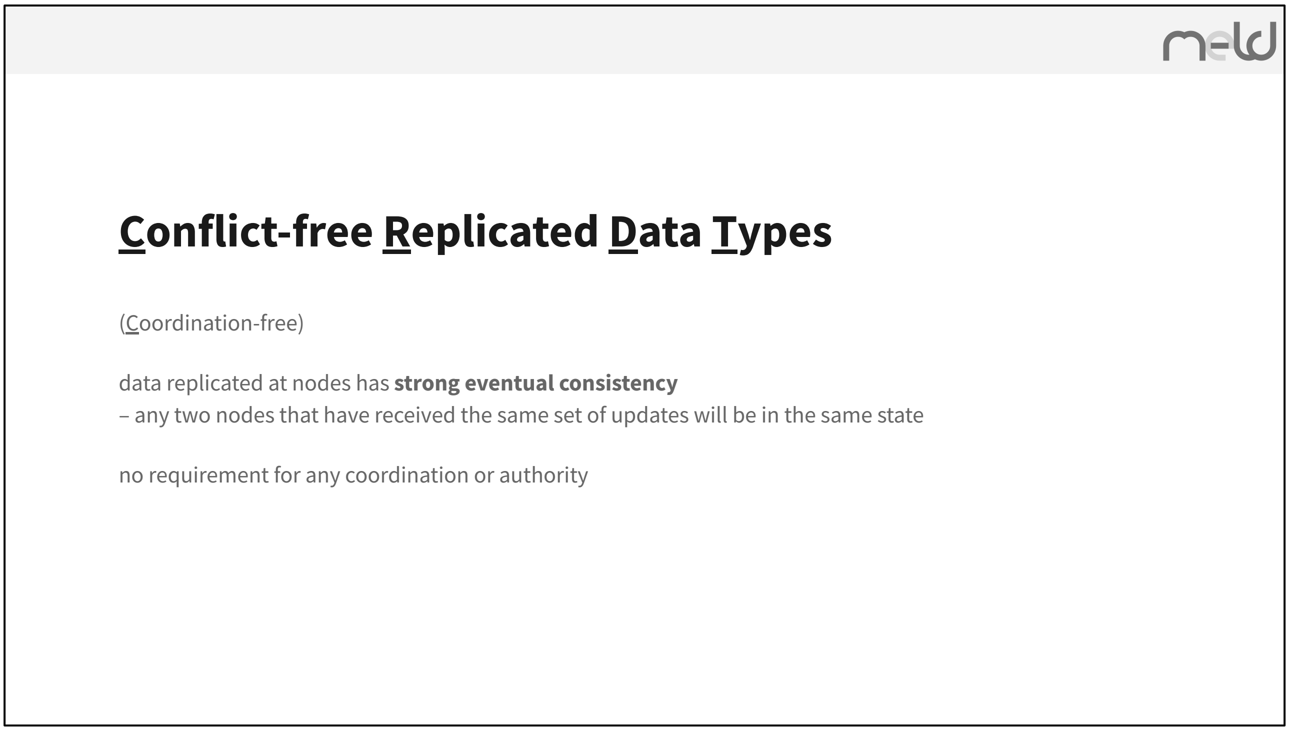
In software information systems, there are coordination algorithms deep in the software, dedicated to ensuring *consistency* – participants' view of the state should behave according to some specification. We already know that knowing the current state is impossible, so what we end up with is models in which we're presented with something hopefully *good enough* for us to proceed.

I want to notice three things about these models. First, they don't represent any kind of *reality*. Instead, they're the compromises that clever folks have managed to come up with so far, which are tractable for computers pushing bits around at below the speed of light.

Second, the coordination algorithms that support these models are not things that Blue or Purple understand. Blue's business process, to promise to hold back artwork, is something that has to be built *on top of* the information system's consistency model.

And third, the consistency model provided by the information system frequently forces background coordination *whether it's needed or not*.

Can we do better? Can we create software systems that more closely reflect reality, in which fundamentally independent agents contribute to a universal state, *without* having to coordinate?



Conflict-free Replicated Data Types

(Coordination-free)

data replicated at nodes has **strong eventual consistency**
– any two nodes that have received the same set of updates will be in the same state

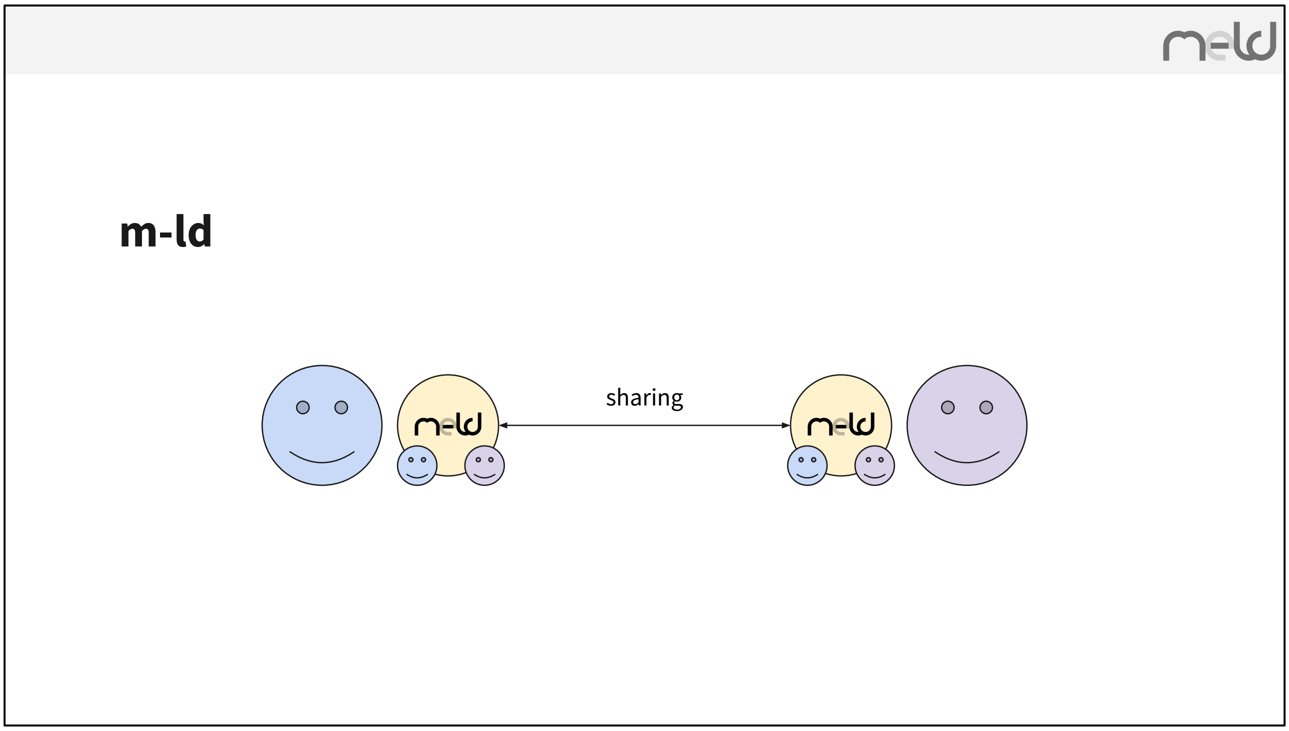
no requirement for any coordination or authority

Roll credits!

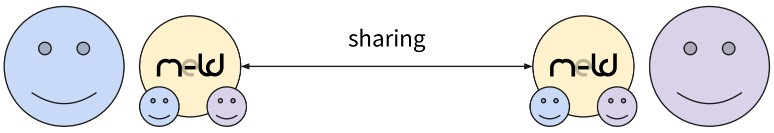
Computer scientists have been working on this for a long time (under many guises), and, yes, a recent breakthrough has been the characterisation of a class of data structures called Conflict-free Replicated Data Types. I'll say up-front that it should really be "Coordination-free" replicated data types. That's because the magic of CRDTs is that everyone's changes are fundamentally independent (like reality)...

… but if everyone stops changing, the state of the system is the same for everyone (like reality)...

… and none of that requires any coordination, or database, or any centralisation of data at all (like reality).



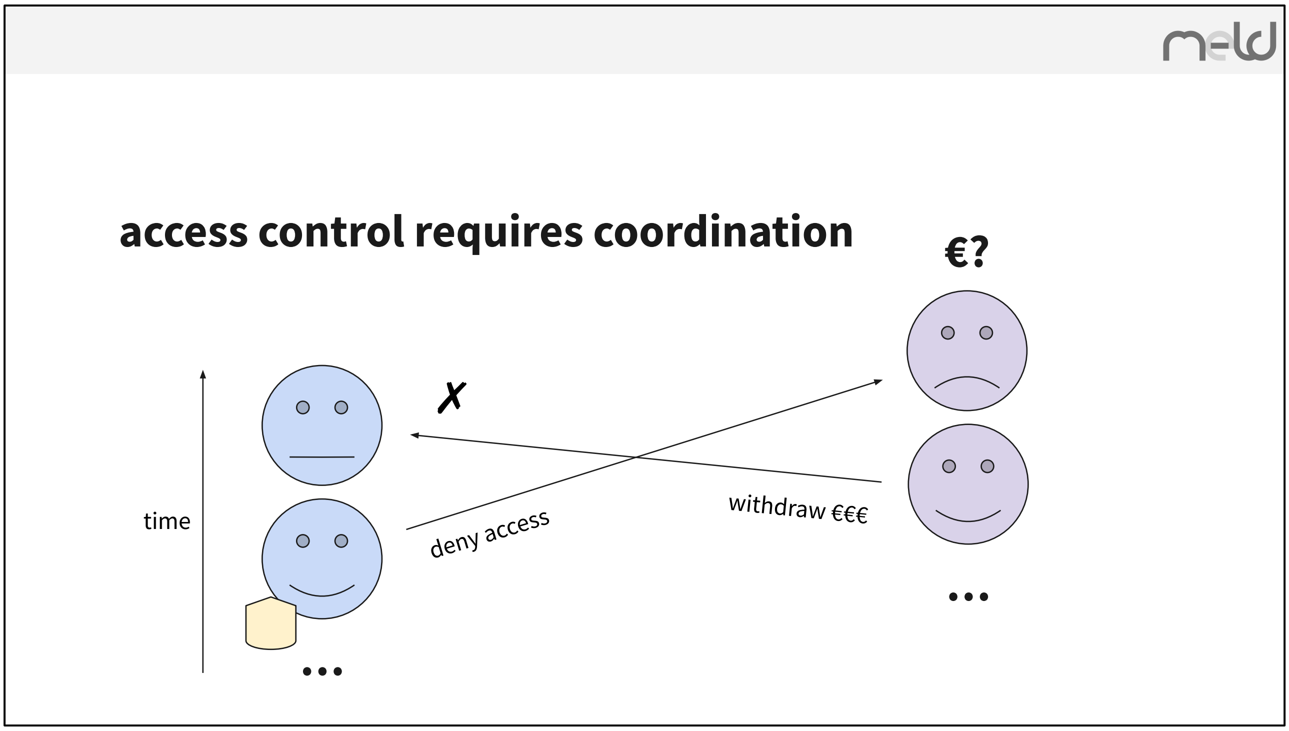
m-ld



Our software, m-ld, is built using a CRDT, for exactly this reason. The foundational premise of m-ld is that the participants in a shared information system *can* operate absolutely independently and still have a view on the overall state by means of the shared data structure. As you can see, the idea with m-ld is to co-locate a copy of the data close to each participant, fully peer-to-peer with no central data location. (That's unless you want one in your architecture, it's your choice – m-ld is not a platform but more like a library and local data store.)

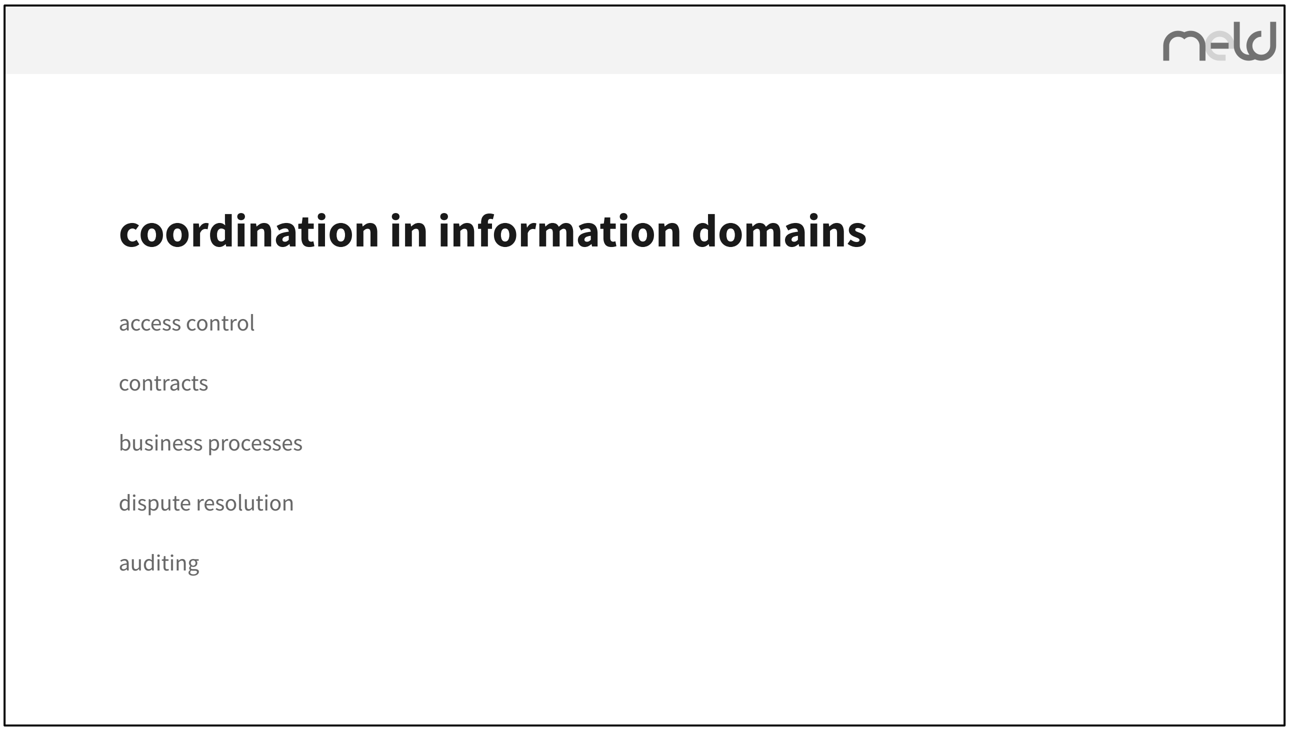
But as you can see, CRDTs and m-ld have the same problem as reality. The views we have of other participants are subject to latency, so we only know the latest events we have received, and we have no guarantee whatsoever that some important state hasn't changed since they arrived.

One area that seems particularly affected by this, is metadata describing the information being exchanged, such as security. In our project, funded by NGI Assure via NLnet, we're working on how to reliably apply security controls like fine-grained authorisation to this shared information. We chose this problem, not *just* because security is an important concern for any user of an information system, but also because access control often *requires* coordination between participants.


access control requires coordination

The reason for this is that access controls are expected to have an immediate effect. Events that happen *after* a change to access control should be rejected if they violate the new rules – *even if they come from someone who hasn't heard about it yet*. If everyone is making changes independently and expecting those changes to be committed immediately, this creates the possibility of a contradiction.

There are a number of ways we could resolve this scenario. We could demand that a change to access control can only be done by majority consensus, or that significant access controlled operations require a *token* from some authority, similar to Blue's promise not to sell the art while Purple makes their mind up.



coordination in information domains

access control

contracts

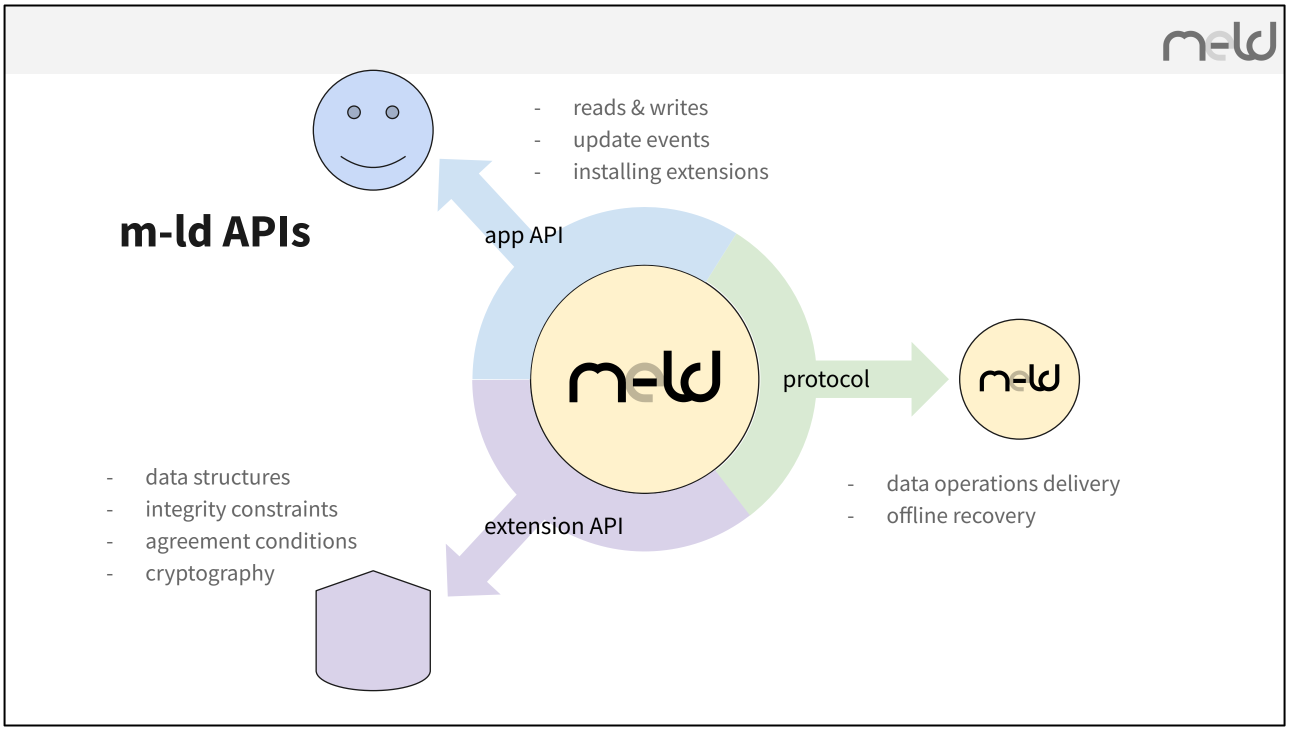
business processes

dispute resolution

auditing

I said that coordination has downsides, and it does, it's a trade-off. Waiting for consensus, say asking Blue for a token, takes time, and it very clearly puts limits on what people can do (in the case of access control, that's the whole point). But notice the difference here. We're introducing a *necessary* coordination for a very specific purpose, in the language of the information domain. It's not something that arrived in our world *because* we're using some technology like a centralised database, with some hard-wired consistency model.

We want adding coordination to an information system to be a conscious choice by application developers, and access control is by no means the only reason to do so. As a matter of fact, coordination becomes a necessity almost anytime you have multiple participants and you want to make a decision that affects anything outside of the information domain, like agreeing a sale, or completing a workflow.



Supporting this requires m-ld to be extensible with new coordination schemes. We want apps using m-ld to be able to choose which operations are coordination-free; which require promises; and which require consensus. In fact, we want basically everything beyond the core CRDT to be an extension; and so, m-ld has the concept of extensibility at a really fundamental level. One of the primary focuses of our research is to figure out what the extension API should look like. This picture gives you an idea of our current thinking.

There's so much more to talk about. I'd love to spend time discussing how the choice of coordination scheme itself needs coordinating; and in fact any information schema or data structure needs coordination if it changes. (Anyone who has been through a database upgrade knows this.) I'd also love to talk about other aspects of our approach, like how we're ensuring interoperability and data portability – those on the Linked Data webinar last year will have an idea already.

https://nlnet.nl

https://www.ngi.eu

https://m-ld.org

So I'd love to hear your thoughts in the discussion, and beyond. Many thanks to Joost and the NLnet team for this chance to hint to you what we're doing, and to NLnet and the NGI for supporting us.