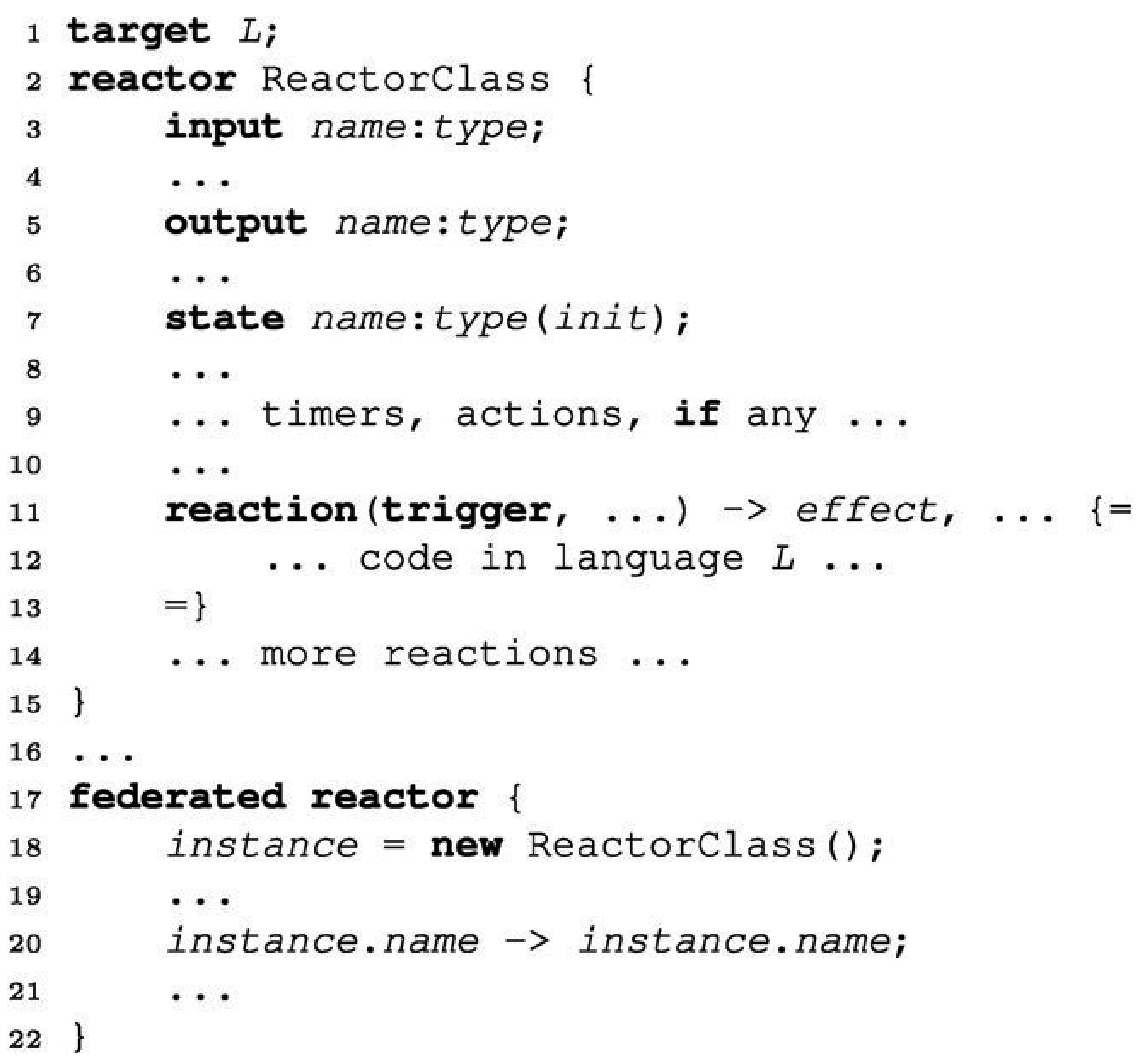
Exploring how to make better consistency and availability trade-offs in networks

April 6 2023



```
target L;
reactor ReactorClass {
    input name:type;
    ...
    output name:type;
    ...
    state name:type(init);
    ...
    ... timers, actions, if any ...
    ...
    reaction(trigger, ...) -> effect, ... {=
        ... code in language L ...
    =}
    ... more reactions ...
}
...
federated reactor {
    instance = new ReactorClass();
    ...
    instance.name -> instance.name;
    ...
}
```

Structure of a federated LF program for target language L. Credit: *Intelligent*

Imagine you want to withdraw some cash from an ATM. You expect it to show your account balance correctly and process your request quickly. However, network delays make it hard for the system to meet both of these simple expectations at the same time. If an ATM system tries to achieve high "consistency," meaning that it displays the latest account balance by checking a remote database, it could make you wait or even prevent you from accessing your accounts during busy times.

On the other hand, if an ATM system favors "availability," it could let you access your accounts fast, but risk showing inaccurate information. To avoid undesired results, designing ATM systems and other distributed systems requires making smart trade-offs.

Seeking to help system designers make those trade-offs, a group of researchers from the University of California, Berkeley and the Technical University of Dresden discovered a simple algebraic relationship between consistency, availability and network latency. This research was published in *Intelligent Computing*.

The researchers call this algebraic relationship the consistency-availability-apparent latency theorem; it quantifies consistency, availability and apparent latency as time intervals. The CAL theorem builds on Eric Brewer's well-known consistency-availability-network partitioning theorem.

Unlike the CAP theorem, which makes system designers choose upfront to sacrifice consistency, availability or both when a network lapse occurs, the CAL theorem allows system designers to adjust their choices depending on the situation—a method that enables "rigorous design with clearly stated assumptions."

Rigorous design is essential for distributed systems that control complex networks of connected devices such as factory robots, medical devices and security systems, which have varying latencies at different nodes and are prone to network failures. Using the CAL theorem and Lingua Franca coordination language, a powerful tool that lets programmers specify how different nodes should interact with each other, system designers can model complex networks and use the results to customize distributed systems for reliability and efficiency.

The researchers demonstrated the effectiveness of their approach using a simple ATM network for tracking balances and processing transactions. They used the CAL theorem to model a network of ATMs and derive bounds on the network's latency based on minimum consistency and availability requirements specified using the LF coordination language.

Keeping within these bounds, they were able to optimize the design of the network by making decisions about software placement and trade-offs between consistency and availability. In the real world, such optimization may be necessary for the achievement of business goals.

The researchers also showed how to detect and handle violations of network latency requirements after deploying such a system. With built-in fault handlers provided by the LF coordination language, system designers can choose to sacrifice consistency or availability and "handle such failures gracefully."

Additionally, the researchers implemented two coordination extensions based on the CAL theorem—one centralized and one decentralized—that support flexible trade-offs between consistency and availability as network latency changes. The centralized coordination mechanism prioritizes consistency, while the decentralized one prioritizes availability. A system employing these mechanisms can be customized according to the needs of the context.

More information: Edward A. Lee et al, Trading Off Consistency and Availability in Tiered Heterogeneous Distributed Systems, *Intelligent Computing* (2023). DOI: 10.34133/icomputing.0013

Provided by Intelligent Computing

Citation: Exploring how to make better consistency and availability trade-offs in networks (2023, April 6) retrieved 5 May 2024 from https://techxplore.com/news/2023-04-exploring-availability-trade-offs-networks.html