RUST CHINA CONF 2021 - 2022

第二届中国Rust开发者大会

2022.07.31 Online

MadSim

神奇的分布式系统确定性模拟器

Magical Deterministic Simulator for Distributed Systems

王润基

清华大学计算机系

Singularity Data

RUSTACEANS

❤️

+

王润基

Rust

系统编程

HELLO

SYSTEMS

分布式系统

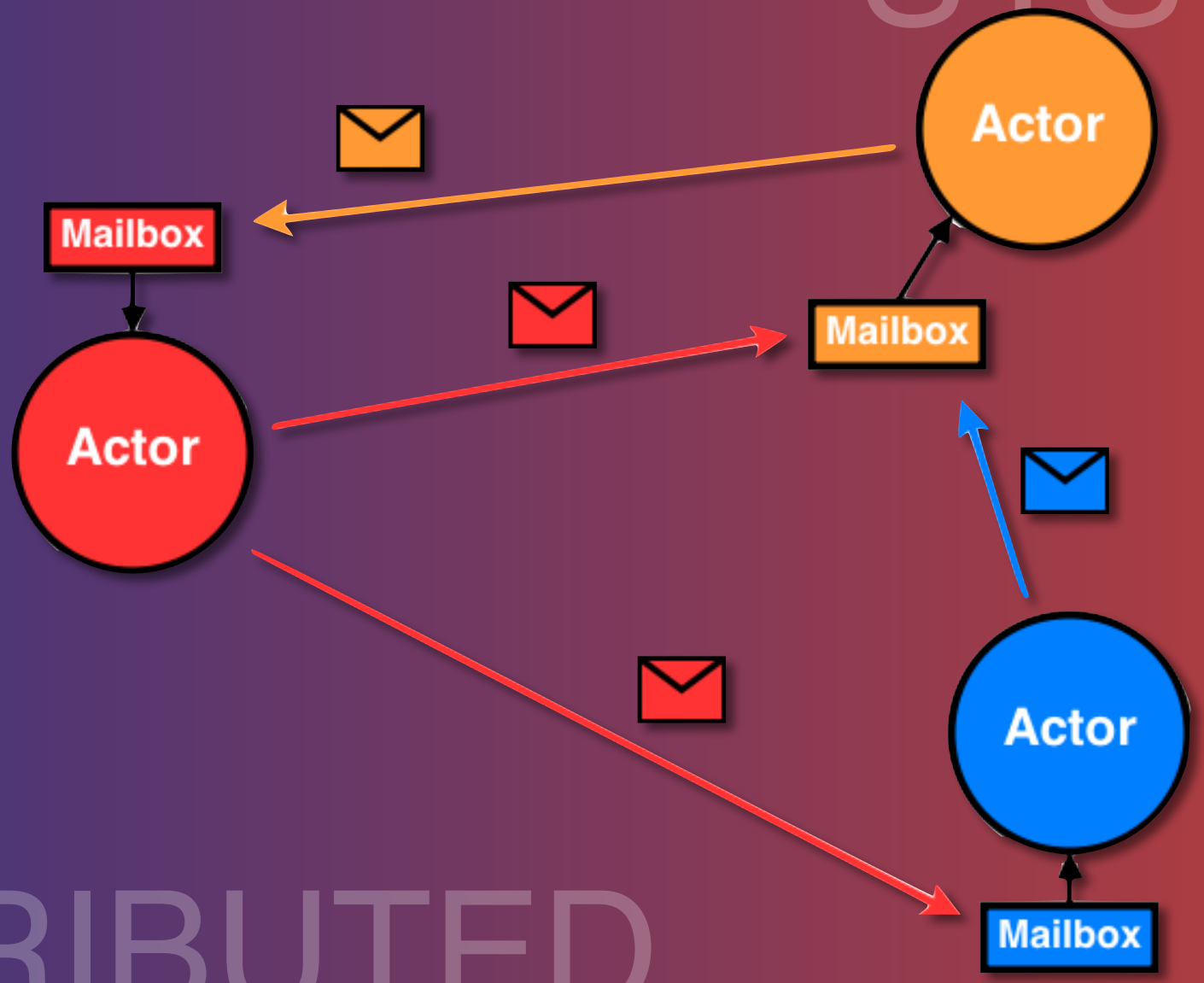
DISTRIBUTED

SYSTEMS

DISTRIBUTED

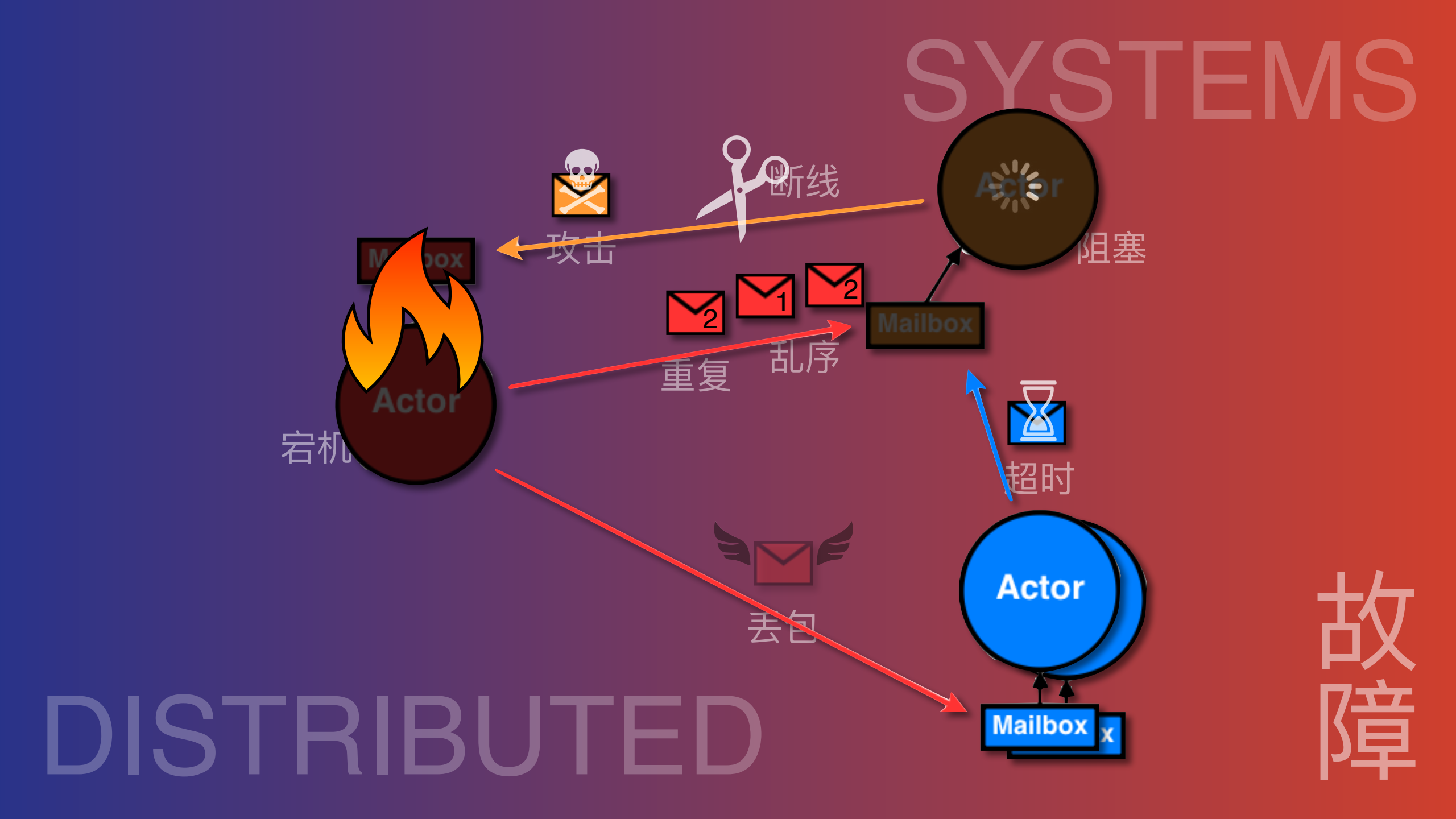
节点

SYSTEMS



DISTRIBUTED

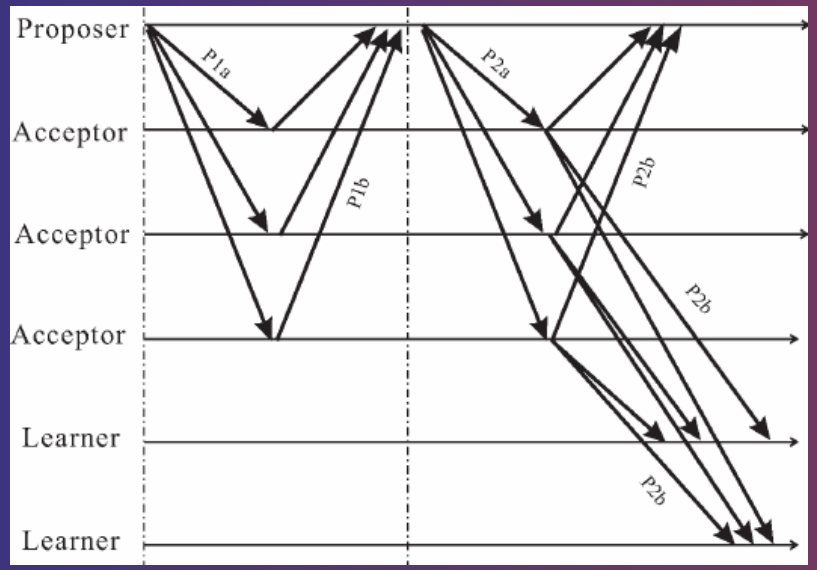
通信


SYSTEMS
DISTRIBUTED
故障
攻击
断线
阻塞
重复
乱序
宕机
超时
丢包
Actor
Mailbox
Actor
Mailbox
Actor
Mailbox
1
2
2

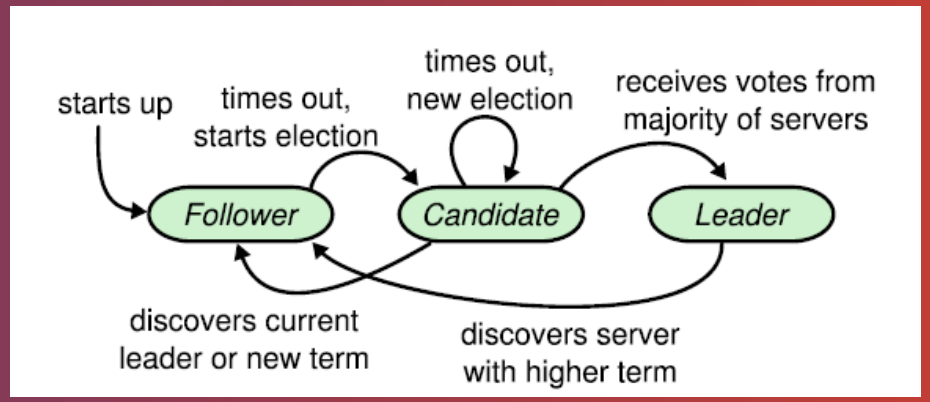
CONSENSUS

DISTRIBUTED

CONSENSUS



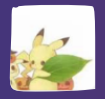
The Paxos algorithm,
when presented in plain English,
is very simple.



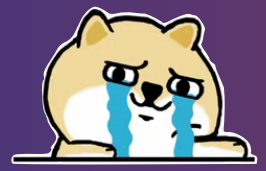
Raft is a consensus algorithm
that is designed to be easy to understand.

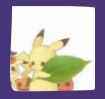
DISTRIBUTED

RAFT LABS



My Raft implementation fails 1 test out of 1000 runs. One follower node tried to commit a command that it had never received from anywhere, neither client nor RPC.





I'm sure there's no data race. Even there is, why would it produce a non-existing value...? Maybe I should buy ECC memory...

For Lab4b, has anyone encountered a bug where TestChallenge1Concurrent occurs roughly once every 150 times?

I have checked everything about persist() in lab2 to ensure the atomicity of the restart. The frequency of this test failure has decreased considerably, but it still happens occasionally

I kept the race detector running and did not detect any race issues yet.

MIT 6.824

Distributed Systems Safety Research

The wild success of Jepsen to destroy nearly every distributed system tested against it repeatedly shows us that we are building distributed systems in a fundamentally bug-prone way.

— sled simulation guide (jepsen-proof engineering)

https://sled.rs/simulation.html

How do we build Jepsen-proof systems?
A single Jepsen test can take 5 minutes. Let's run thousands per second using discrete event simulation!

Step 1: write your code in a way that can be deterministically tested on top of a simulator.

Step 2: build a simulator that will exercise realistic message passing behavior.

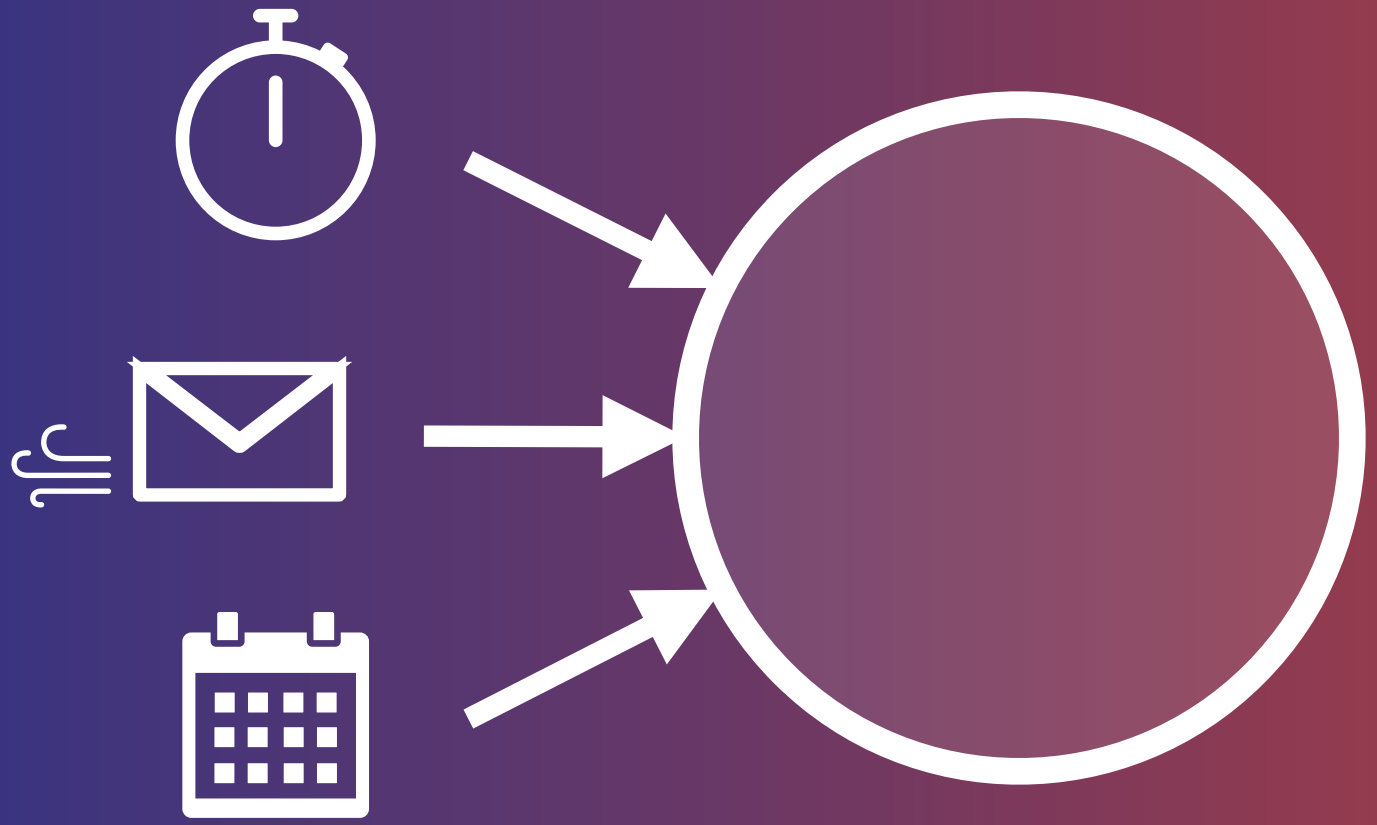
— sled simulation guide

https://sled.rs/simulation.html

STATE MACHINE

SETP1

STATE MACHINE



SETP1

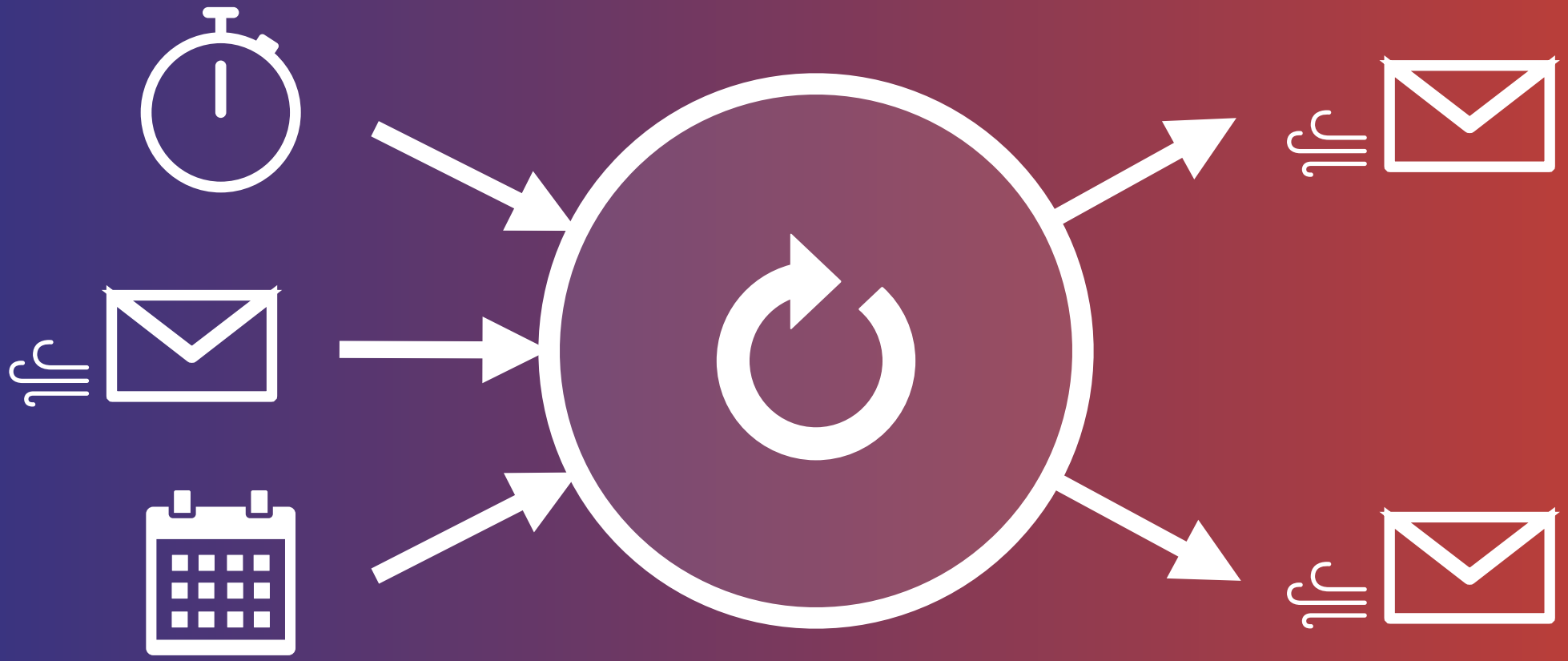
输入

STATE MACHINE

状态转移

SETP1

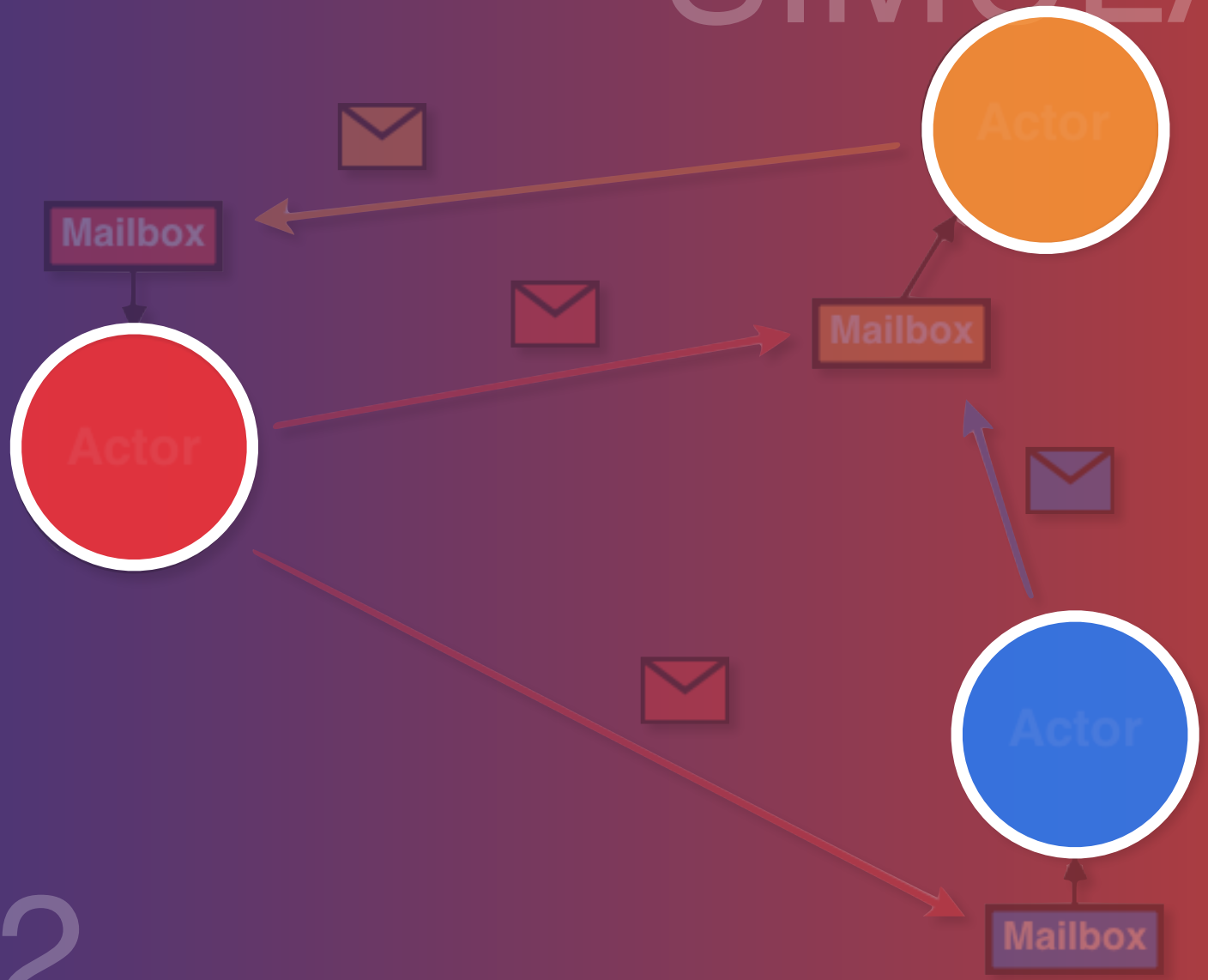
STATE MACHINE



SETP1

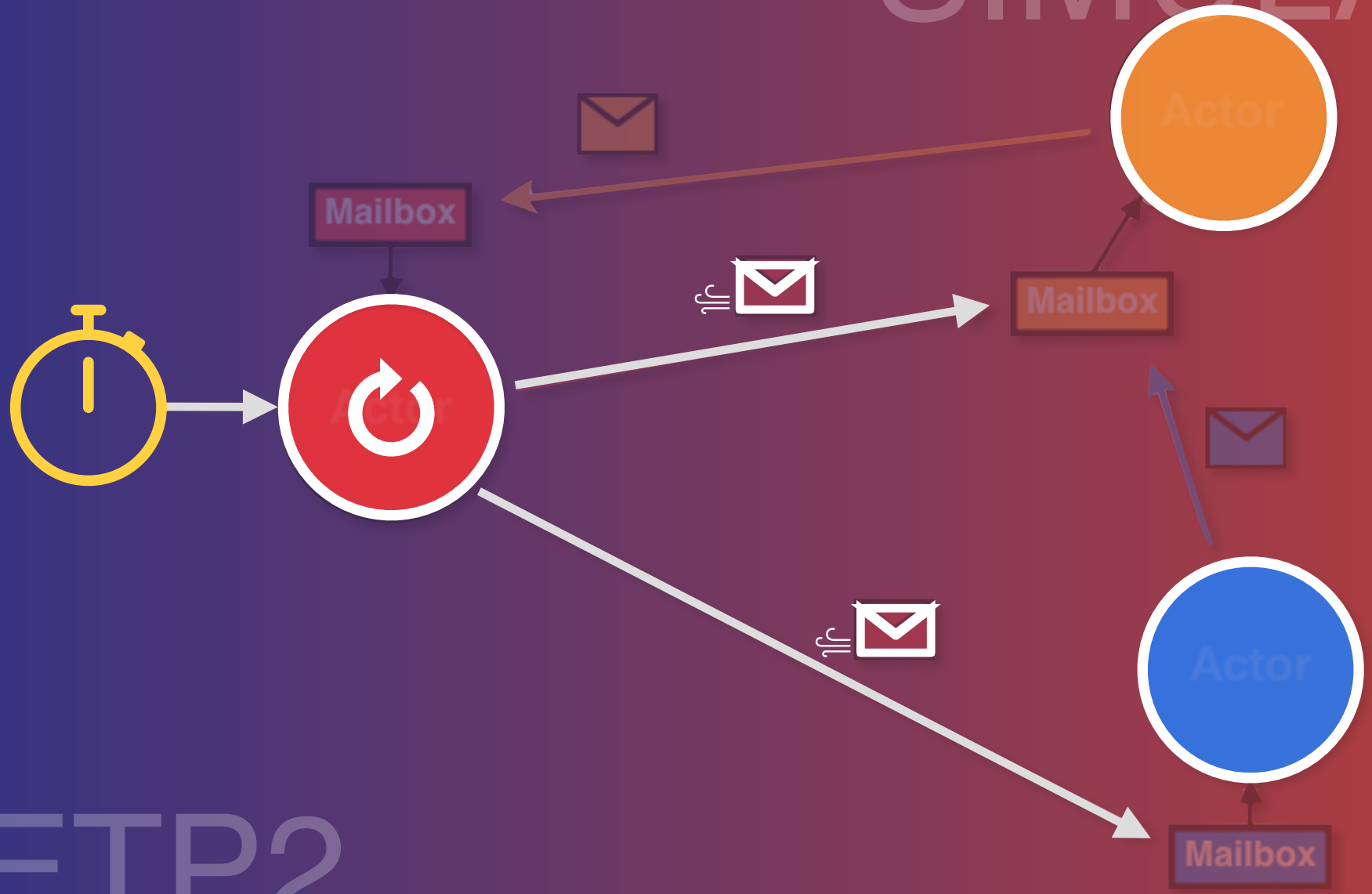
输出

SIMULATION



SETP2

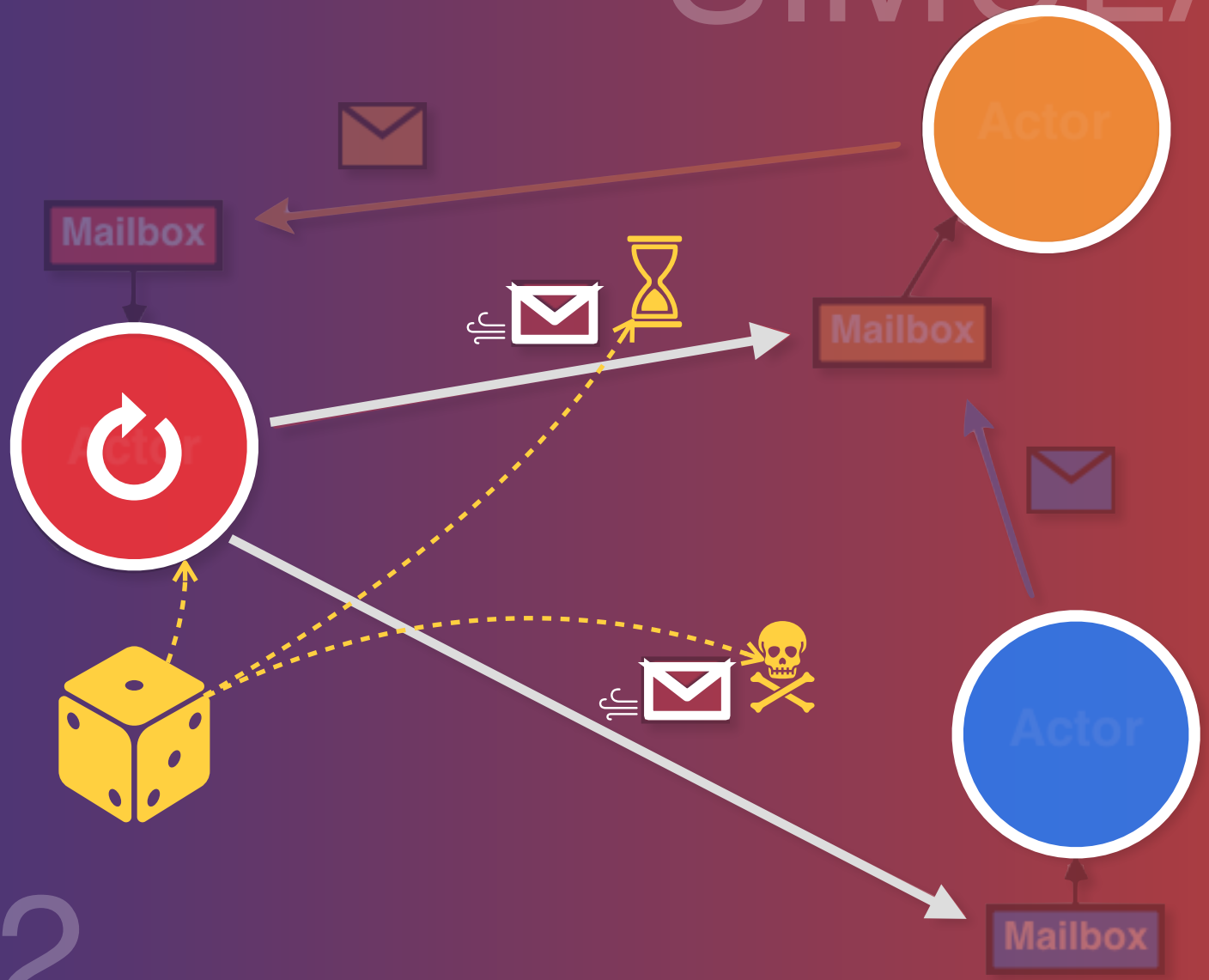
SIMULATION



定时器事件

SETP2

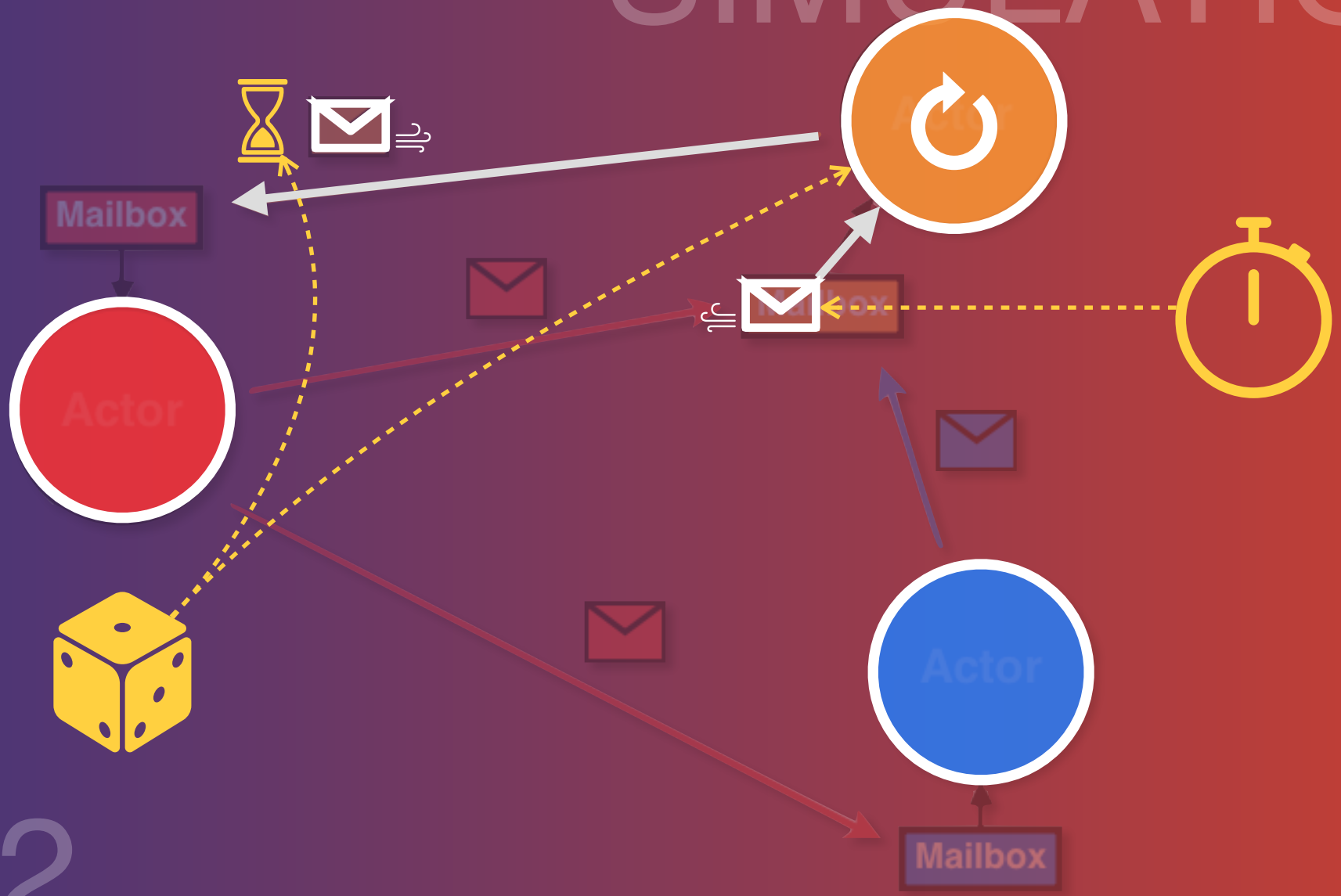
SIMULATION



随机事件

SETP2

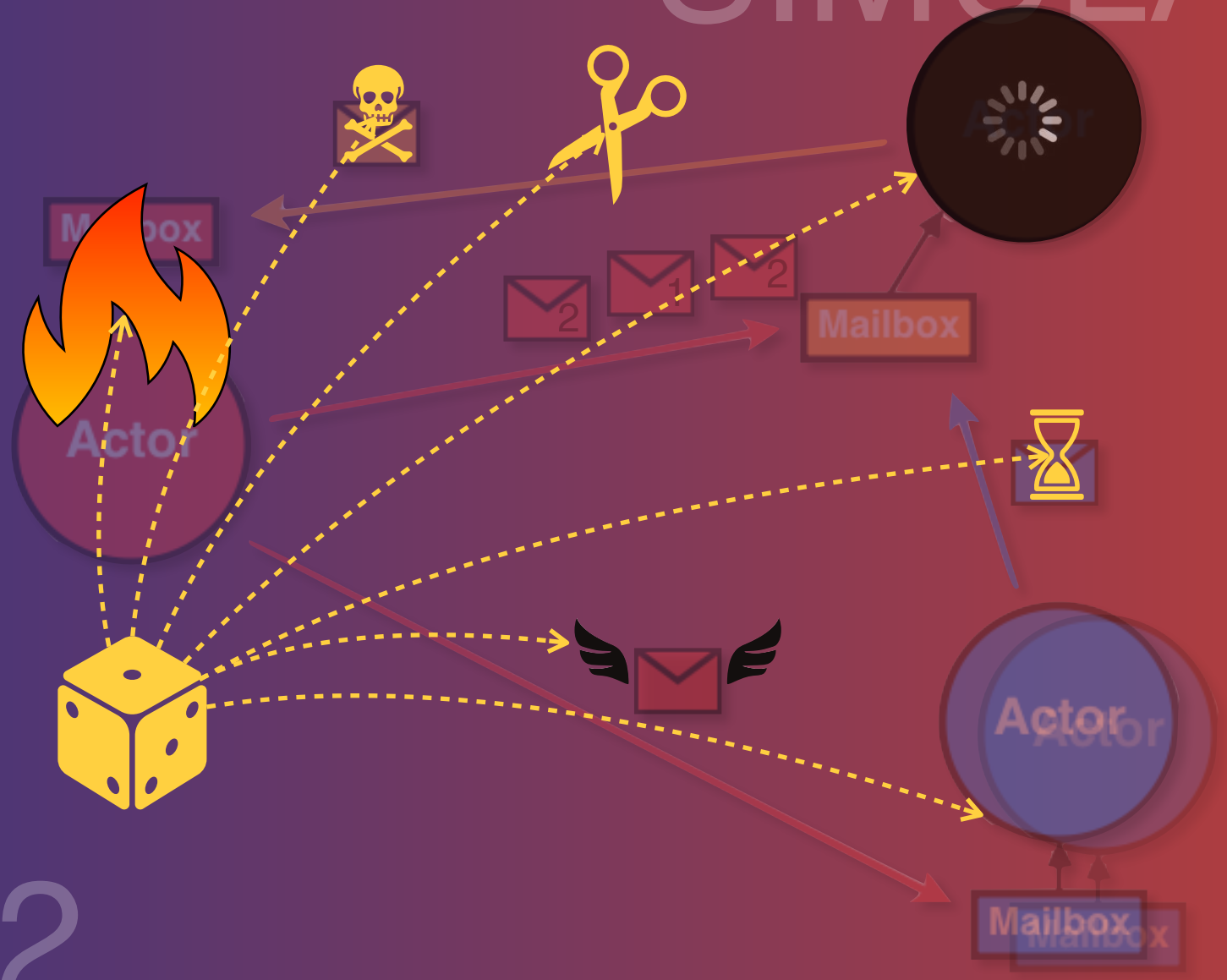
SIMULATION



传播事件

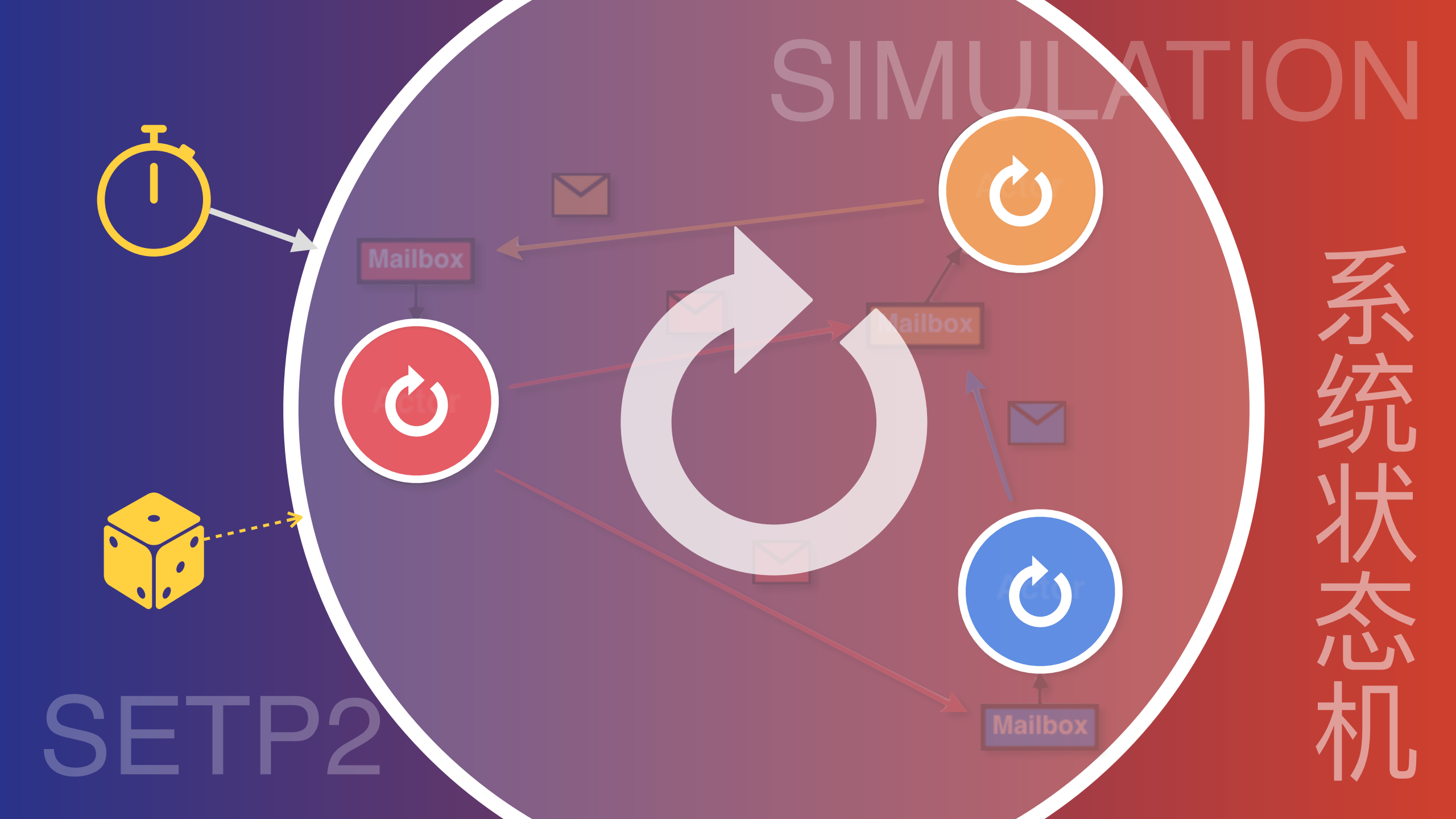
SETP2

SIMULATION

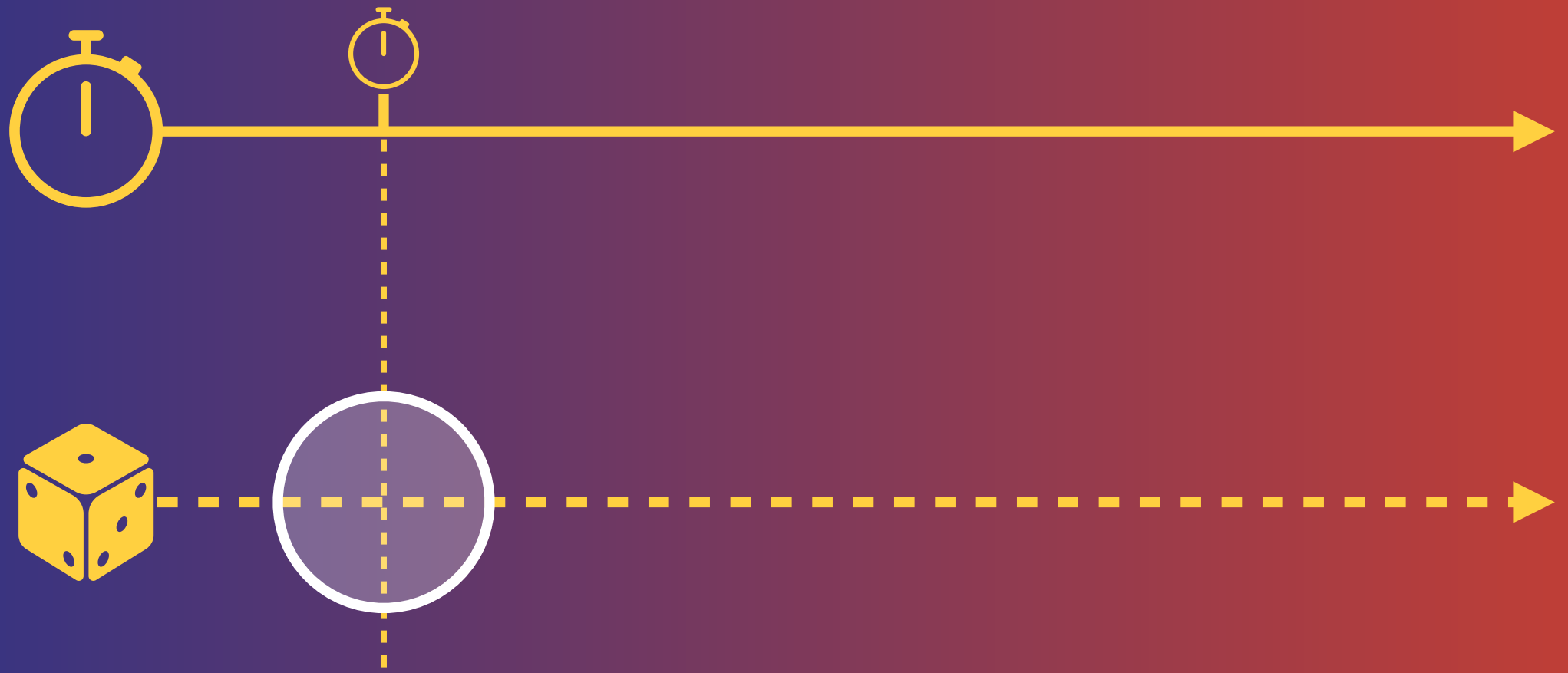


注入错误

SETP2

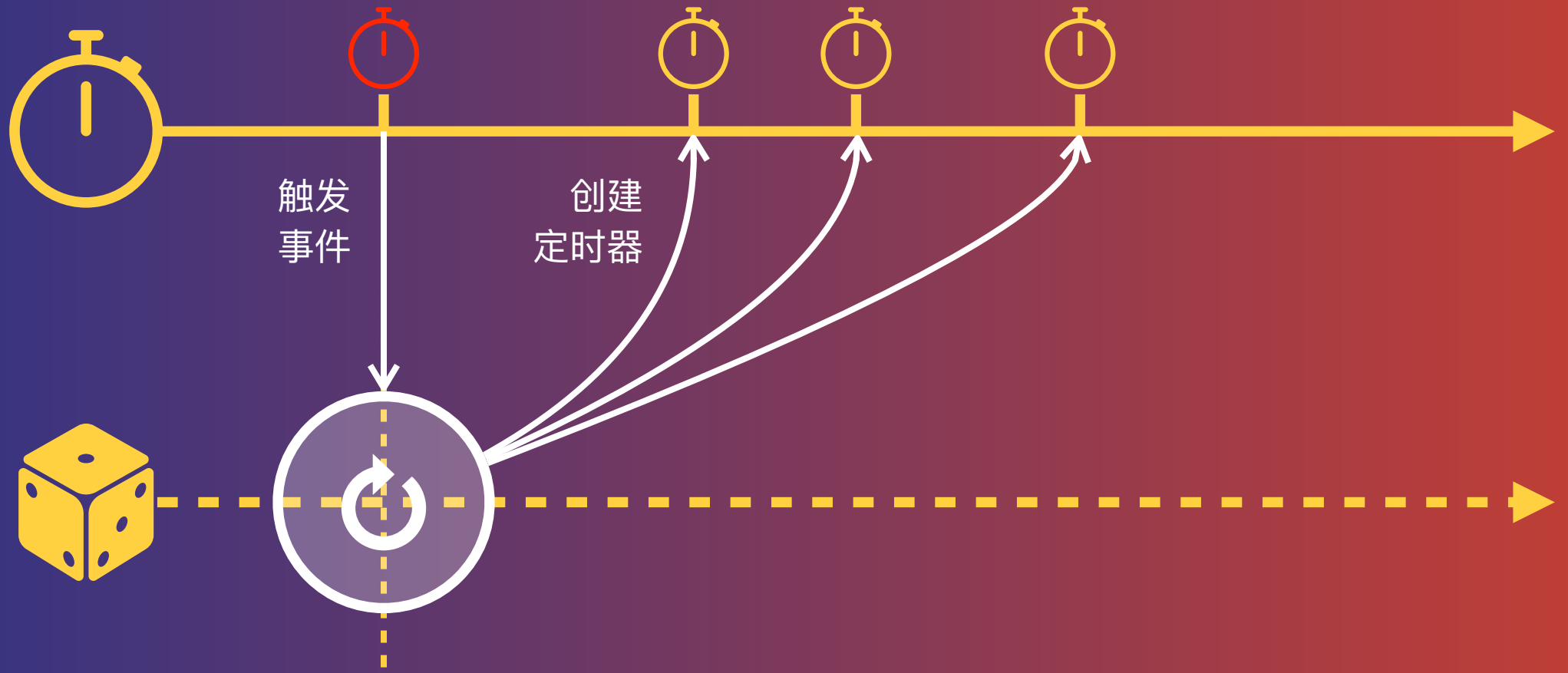

SIMULATION
系统状态机
SETP2
Mailbox
Mailbox
Mailbox

SIMULATION



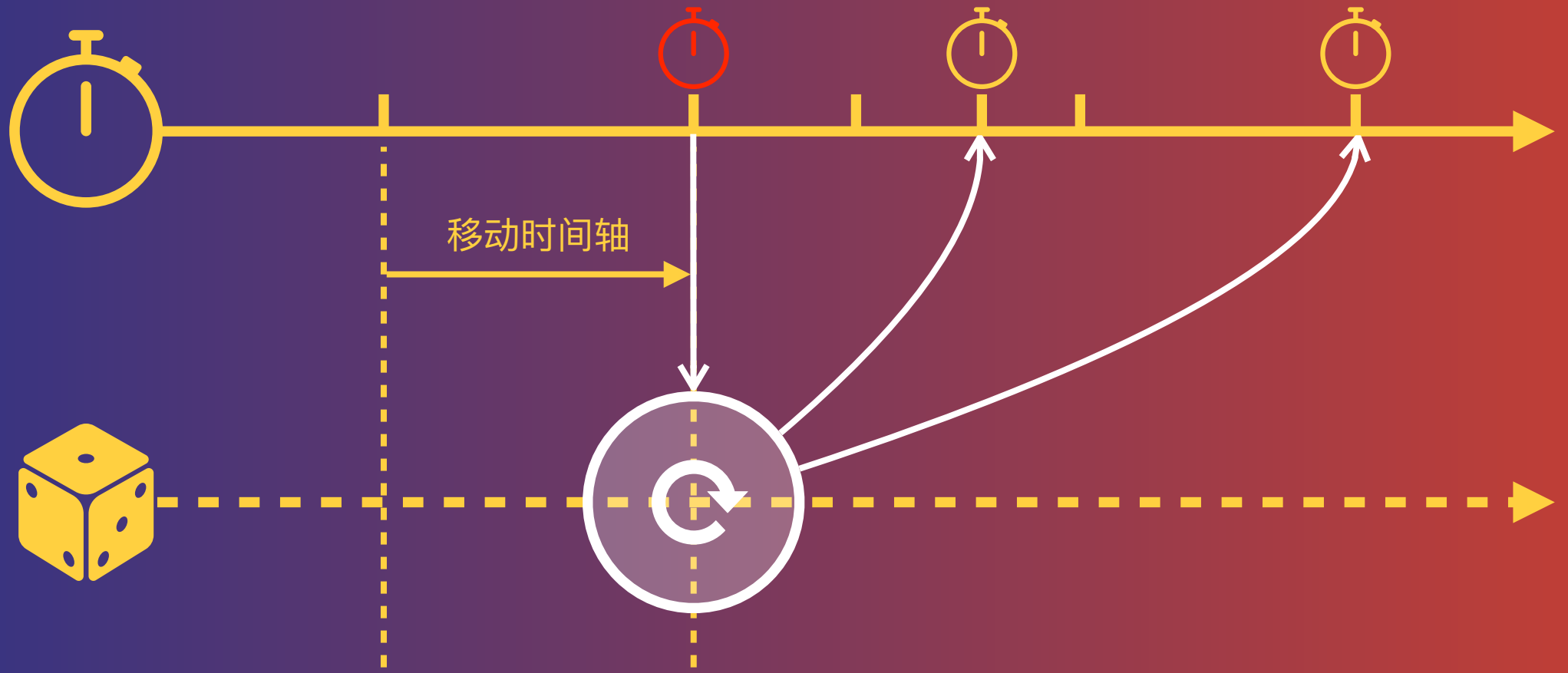
DISCRETE-EVENT

SIMULATION



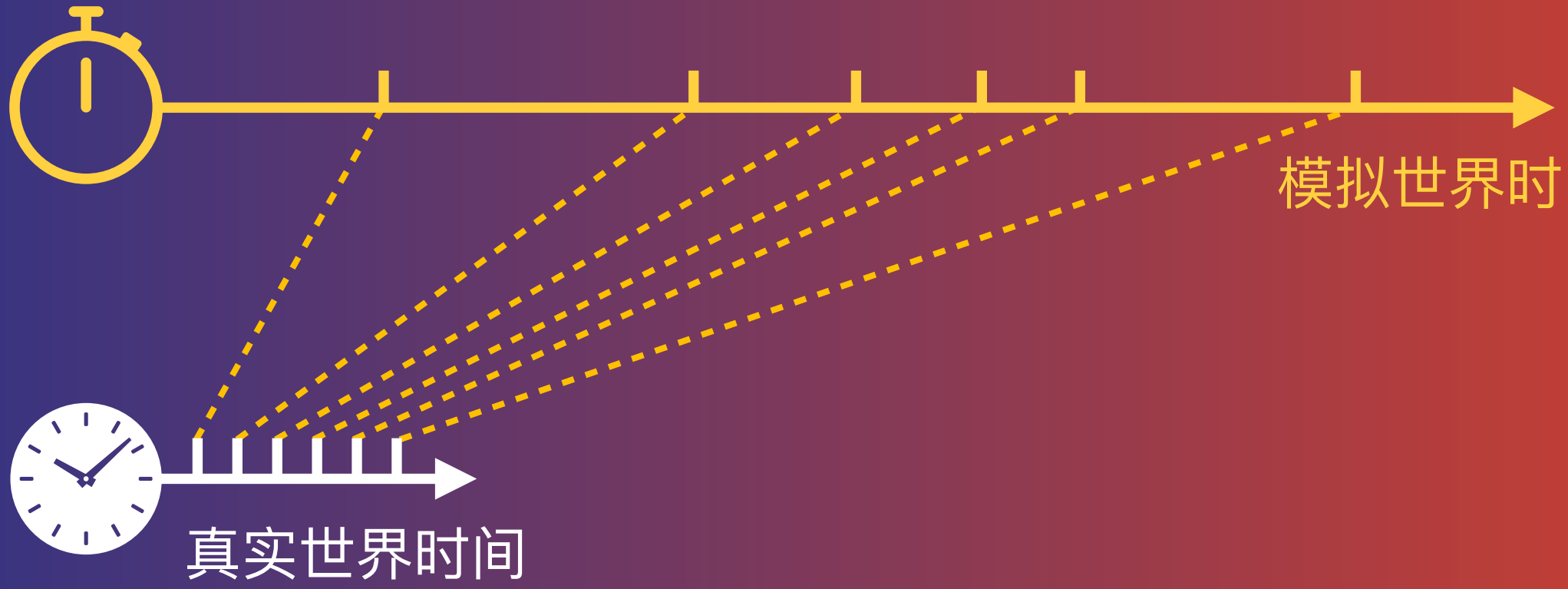
DISCRETE-EVENT

SIMULATION



DISCRETE-EVENT

SIMULATION



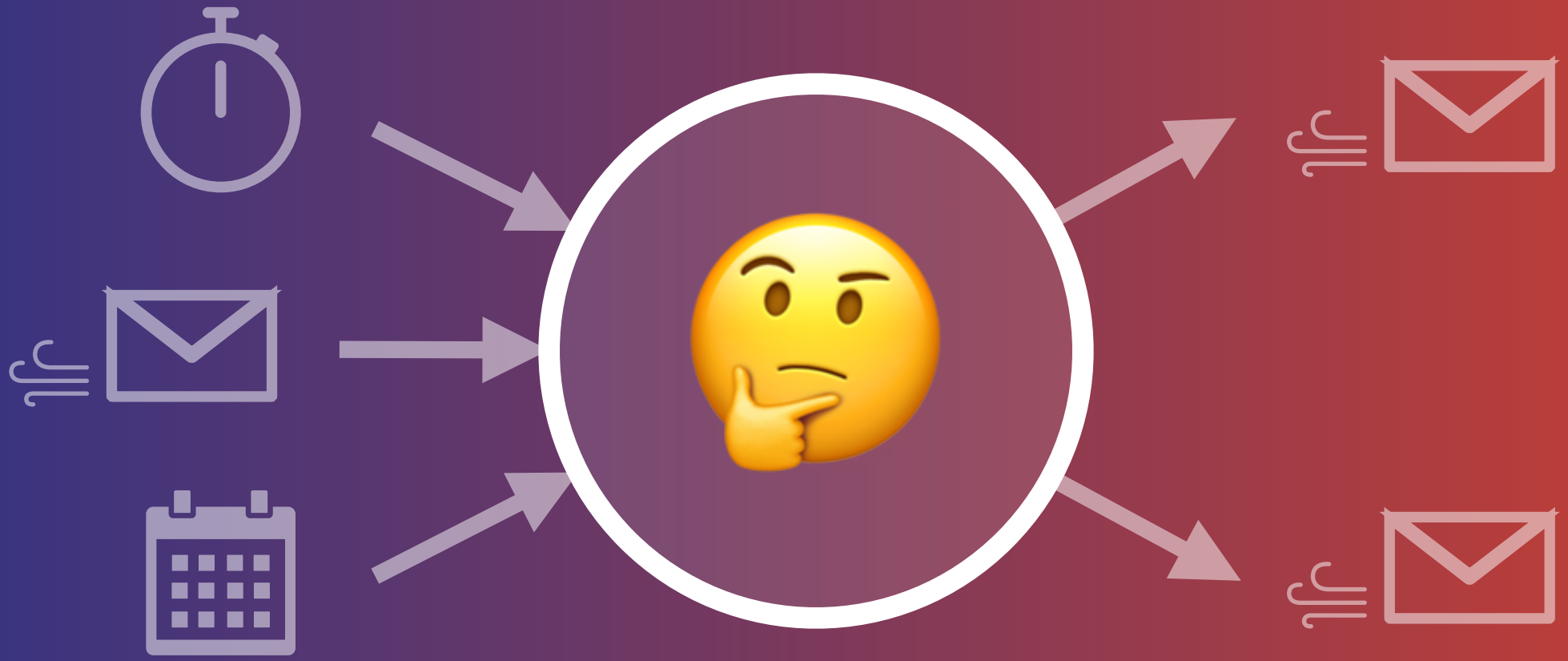
DISCRETE-EVENT

If you're coming into distributed systems, if you learn this technique, you will have a massive advantage over anyone who claims to be a distributed systems expert but just tests in production / their laptop / jepsen.

— sled simulation guide

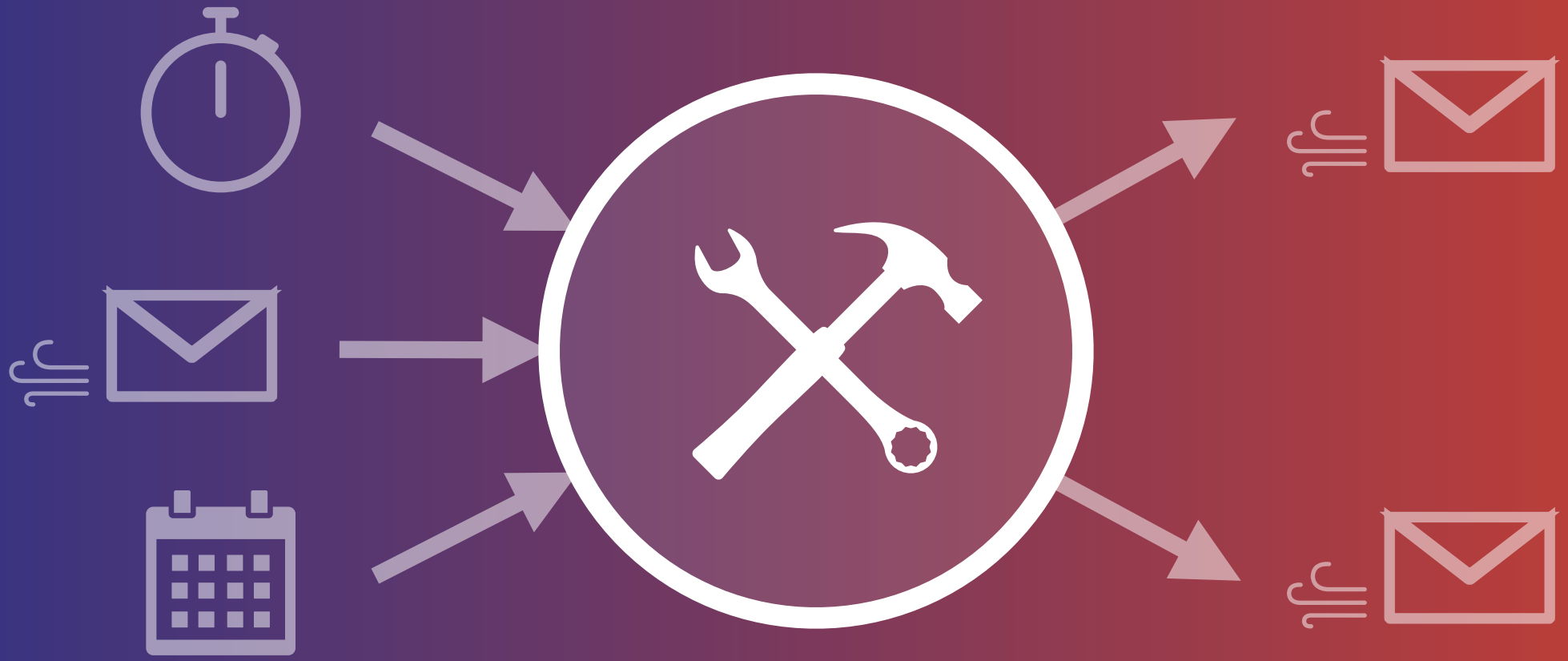
https://sled.rs/simulation.html

STATE MACHINE



IMPLEMENT

STATE MACHINE



IMPLEMENT

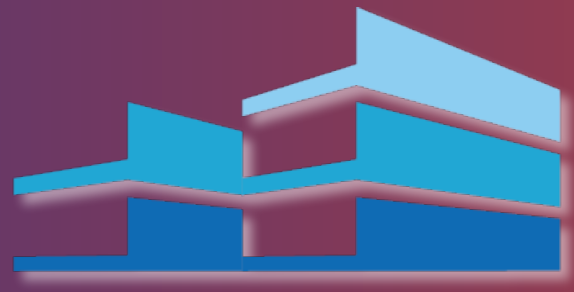
STATE MACHINE

```
impl StateMachine {
    // called when this state machine receives a message,
    // responds with outgoing messages
    fn receive(&mut self, msg, at) -> [(msg, destination)];

    // periodically called for handling periodic functionality,
    // like leader election, pending request timeout etc...
    fn tick(&mut self, at) -> [(msg, destination)];
}
```

Example: tikv/raft-rs

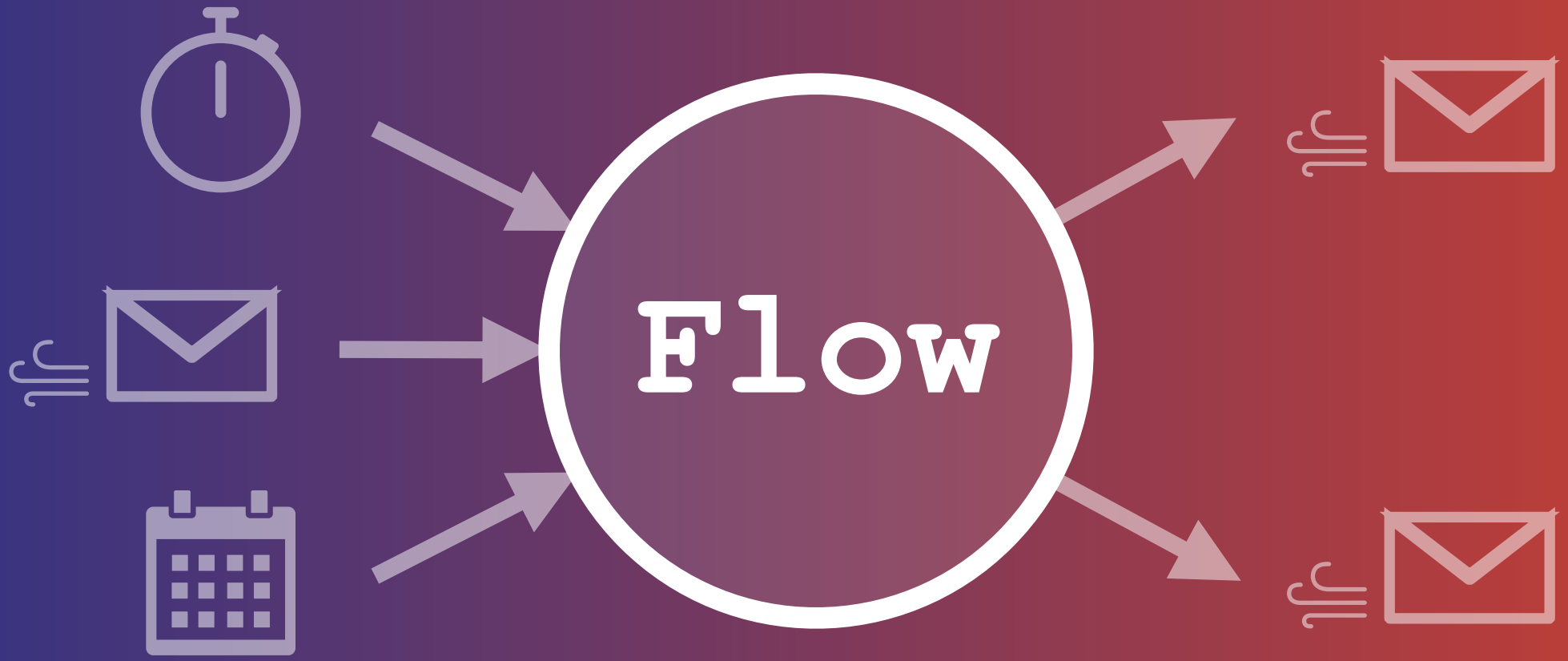
IMPLEMENT



FOUNDATIONDB

FoundationDB gives you the power of
ACID transactions in a distributed database.

STATE MACHINE



IMPLEMENT

FOUNDATIONDB

Compiler

Flow =  + 

C++11 async-await

2010s

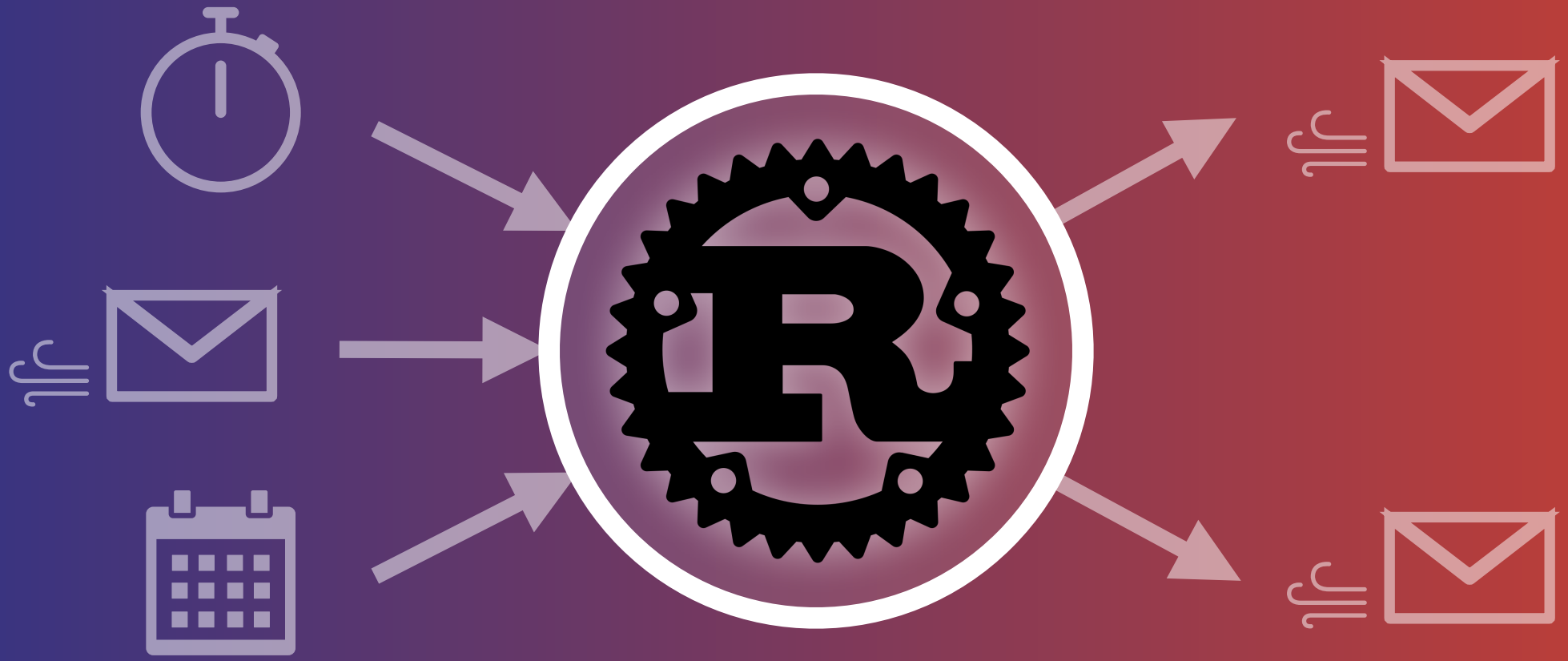
2020s

+

<<

C++11 async-await Rust 2018

STATE MACHINE



IMPLEMENT

master | 6 branches | 0 tags | Go to file | Add file | Code

gardnervickers Update README.md — 42fa7e6 on 25 Dec 2019 — 43 commits

.github/workflows	maybe fix ci?	2 years ago
docs	Work towards initial support for simulating a Tonic app (#4)	2 years ago
simulation-tonic	Work towards initial support for simulating a Tonic app (#4)	2 years ago
simulation	Fix client_server example (#10)	2 years ago
.gitignore	Work towards initial support for simulating a Tonic app (#4)	2 years ago
Cargo.toml	Work towards initial support for simulating a Tonic app (#4)	2 years ago
README.md	Update README.md	2 years ago

README.md

Note

The Simulation library is being refactored to integrate more directly with Tokio. Currently, Simulation is not compatible with Tokio 0.2.x. As a result, it's recommended that users wait for a future release of Simulation. The issue tracking Tokio integration progress can be found here https://github.com/tokio-rs/tokio/issues/1845.

simulation

The goal of Simulation is to provide a set of low level components which can be used to write applications amenable to FoundationDB style simulation testing.

Simulation is an abstraction over Tokio, allowing application developers to write applications which are generic over sources of nondeterminism. Additionally, Simulation provides deterministic analogues to time, scheduling, network

About

Framework for simulating distributed applications

async | simulation | tokio | deterministic

Readme

Releases

No releases published

Packages

No packages published

Contributors 3

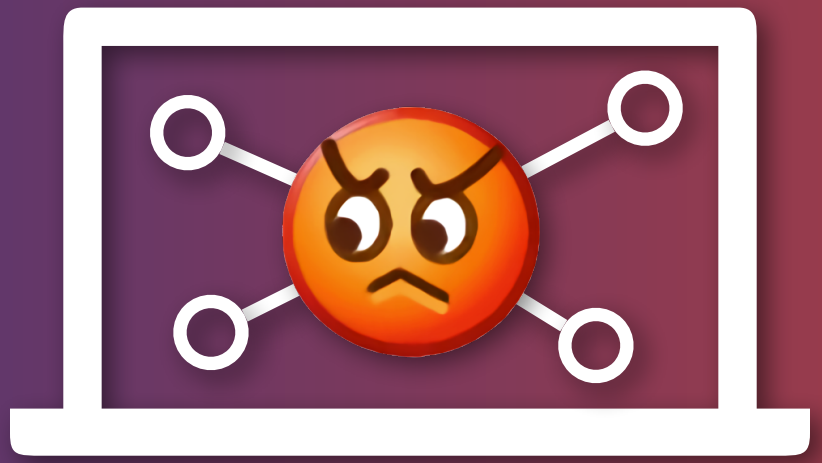
gardnervickers Gardner Vickers

LegNeato Christian Legnitto

davidbarsky David Barsky

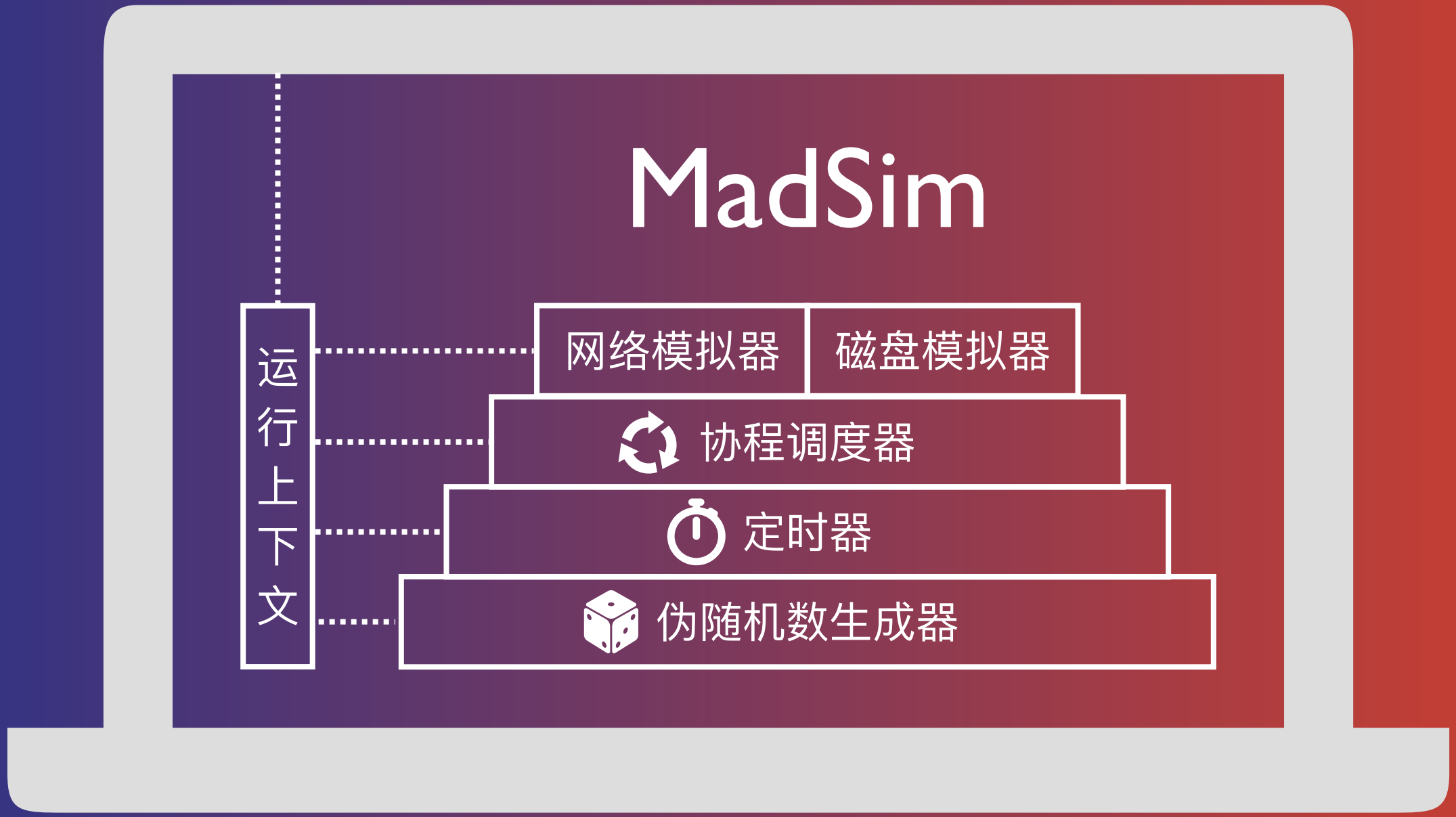
Languages

Rust 100.0%

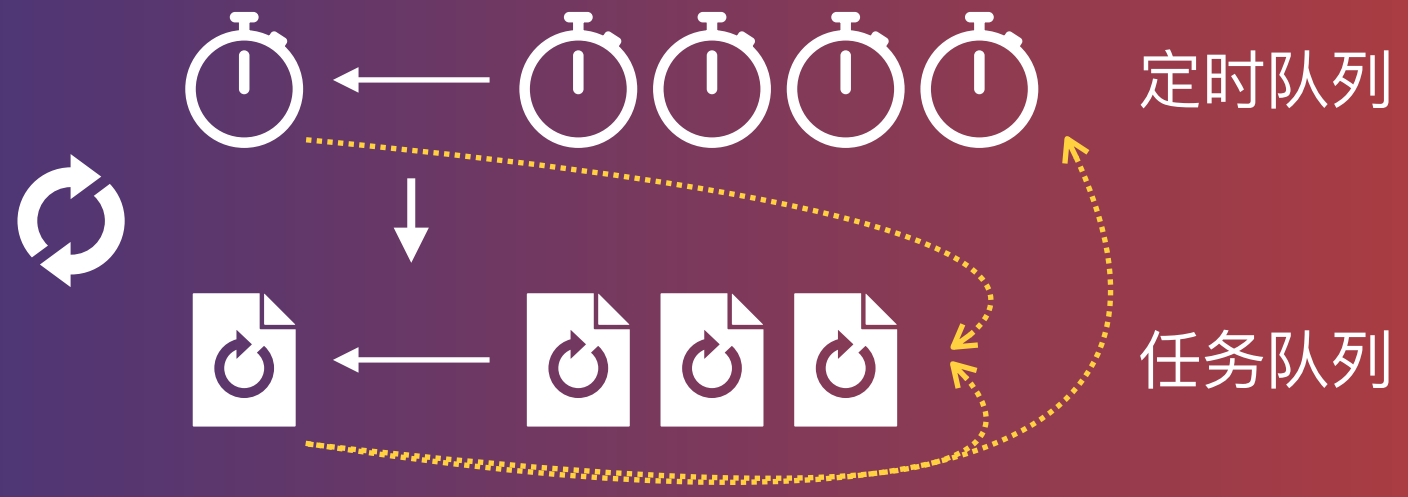


MadSim

Rust 分布式系统确定性模拟器

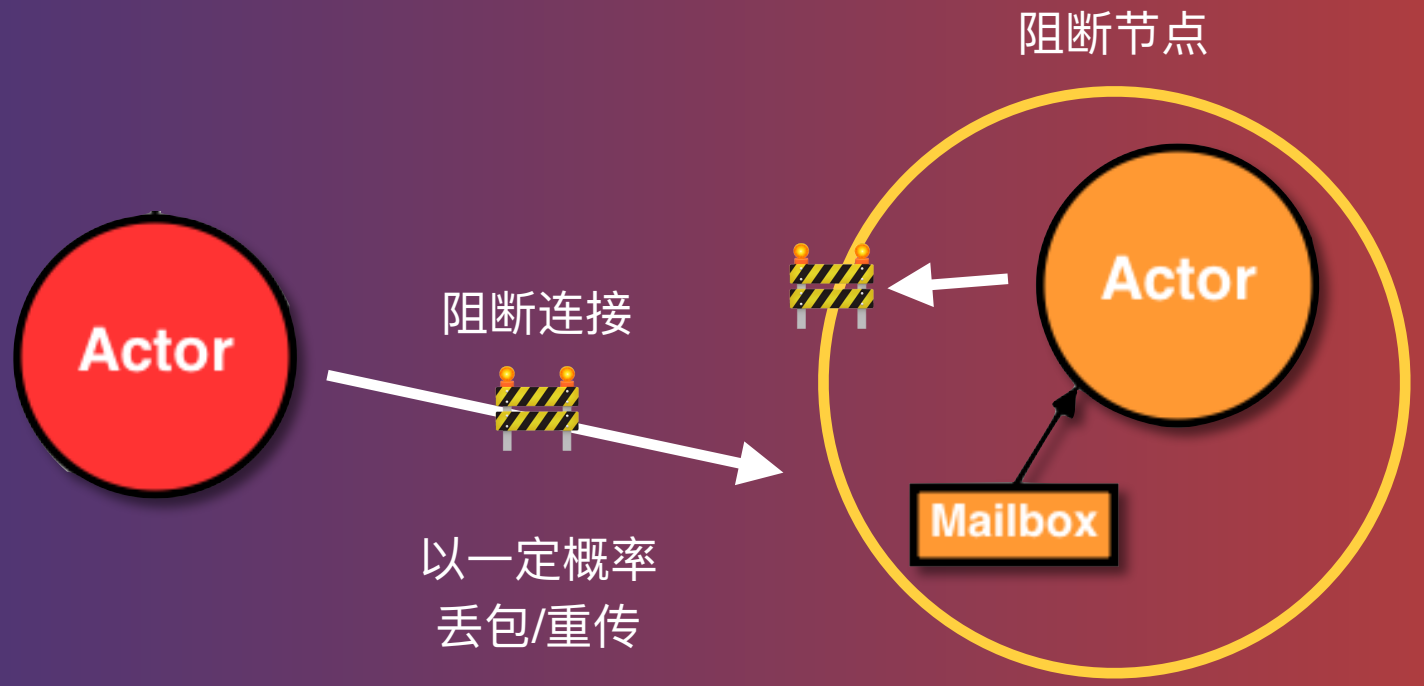

MadSim
运行上下文
网络模拟器
磁盘模拟器
协程调度器
定时器
伪随机数生成器

协程调度器



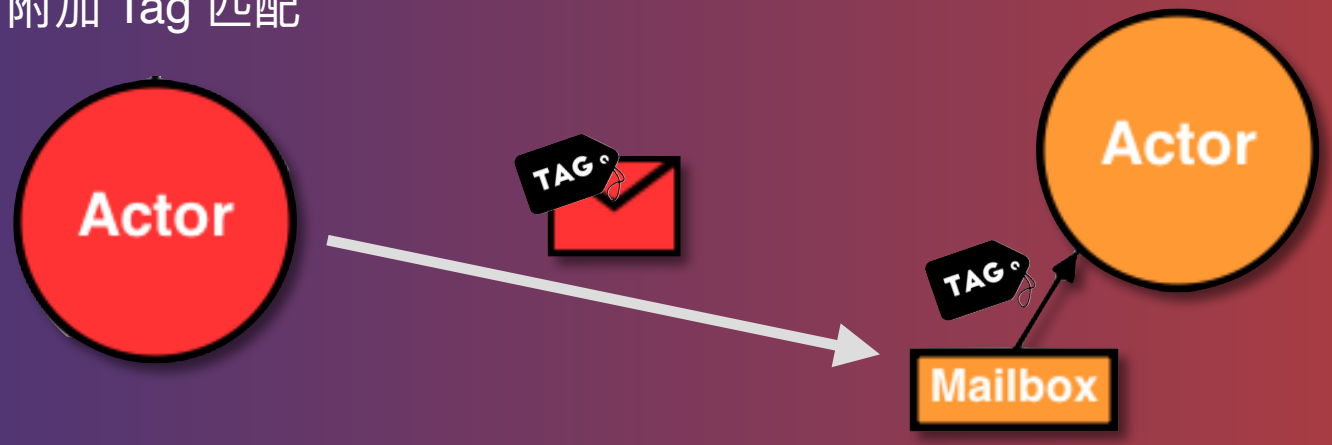
磁盘模拟器

网络模拟器



网络模拟器

无连接模型
以消息为单位
附加 Tag 匹配



RPC

```rust
// Get value of `key`.
fn get(key: Vec<u8>) -> Result<Vec<u8>>;
```

function
(tarpc)

```rust
#[derive(Debug, Serialize, Deserialize, Request)]
#[rtype("Result<Vec<u8>>")]
pub struct Get {
    pub key: Vec<u8>,
}
```

struct
(gRPC,actix)

01110101010011101010001010…

bytes

Crate madsim [-][src]

[−] A deterministic simulator for distributed systems.

Features

- `rpc`: Enables RPC through network.
- `logger`: Enables built-in logger.
- `macros`: Enables `#[madsim::main]` and `#[madsim::test]` macros.

Modules

fs	Asynchronous file system.
net	Asynchronous network endpoint and a controlled network simulator.
rand	Deterministic random number generator.
task	Asynchronous tasks executor.
time	Utilities for tracking time.

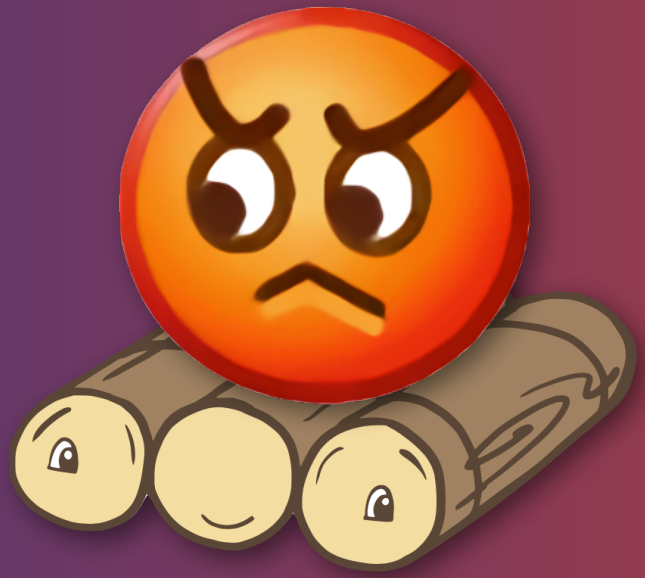
Structs

Handle	Supervisor handle to the runtime.
LocalHandle	Local host handle to the runtime.
Runtime	The madsim runtime.

Attribute Macros

main	Marks async function to be executed by the selected runtime. This macro helps set up a `Runtime` without requiring the user to use Runtime directly.
test	Marks async function to be executed by runtime, suitable to test environment.

Crate madsim
Version 0.1.1
See all madsim's items
Modules
Structs
Modules
fs
net
rand
task
time
Structs
Handle
LocalHandle
Runtime

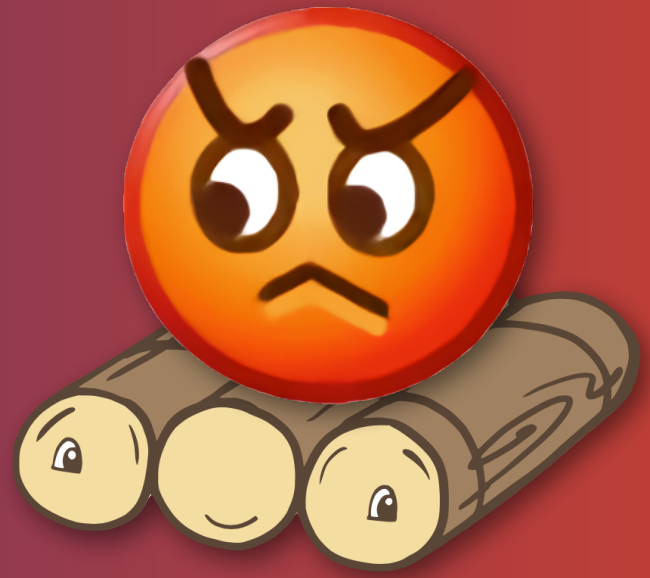


MadRaft

基于 MadSim 的 Raft 实验框架

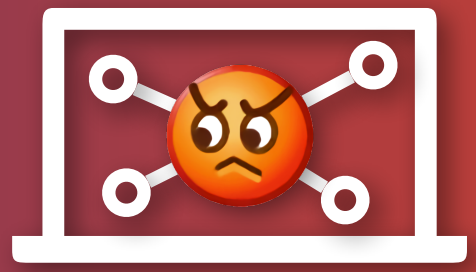
INHERITANCE

MIT 6.824 Labs →

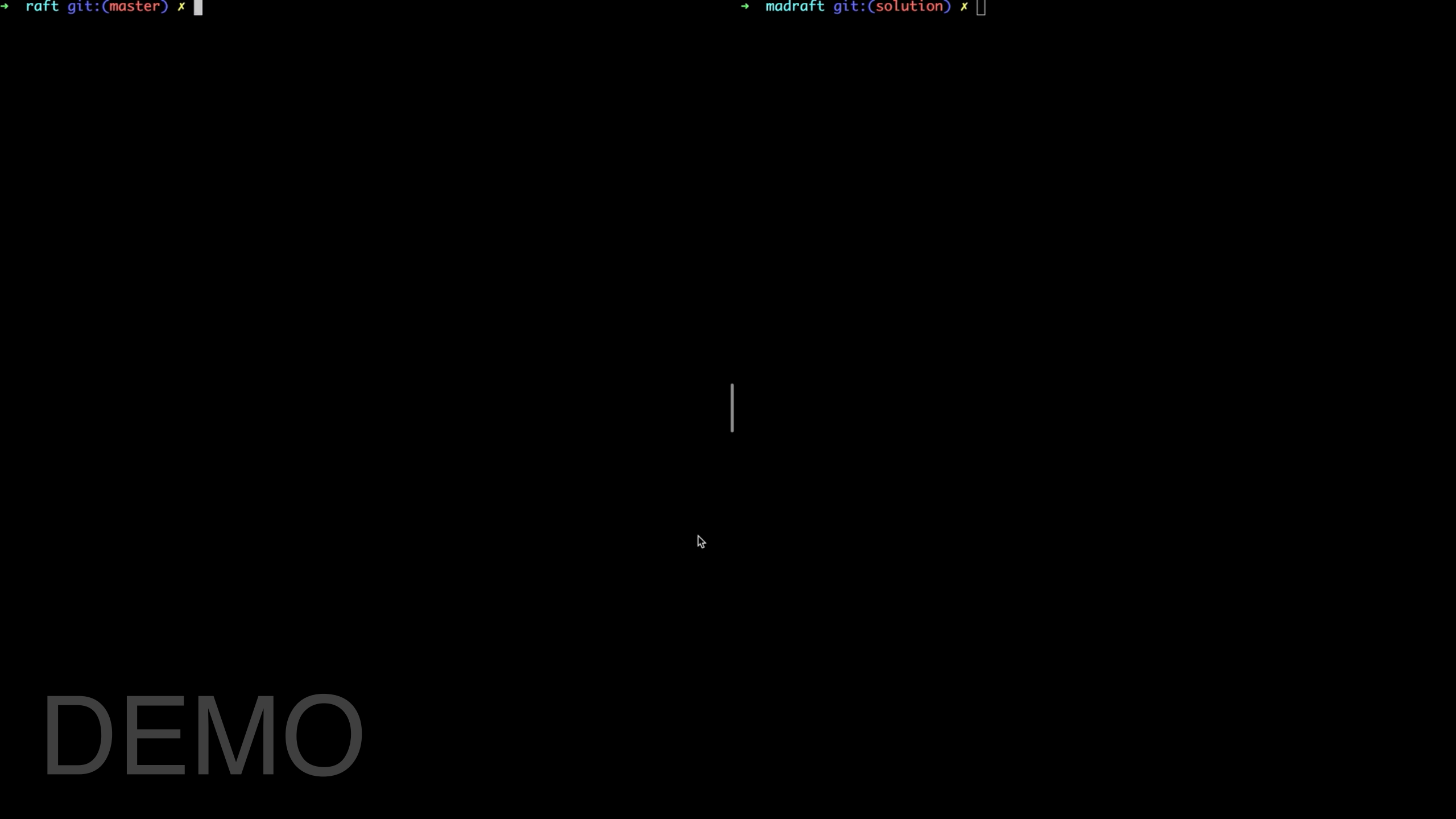


↓

Talent Plan →



MADRAFT


→ raft git:(master) ✗
→ madraft git:(solution) ✗
DEMO

```
rand::thread_rng()
```

```
madsim::rand::rng()
```

```
futures::select!
```

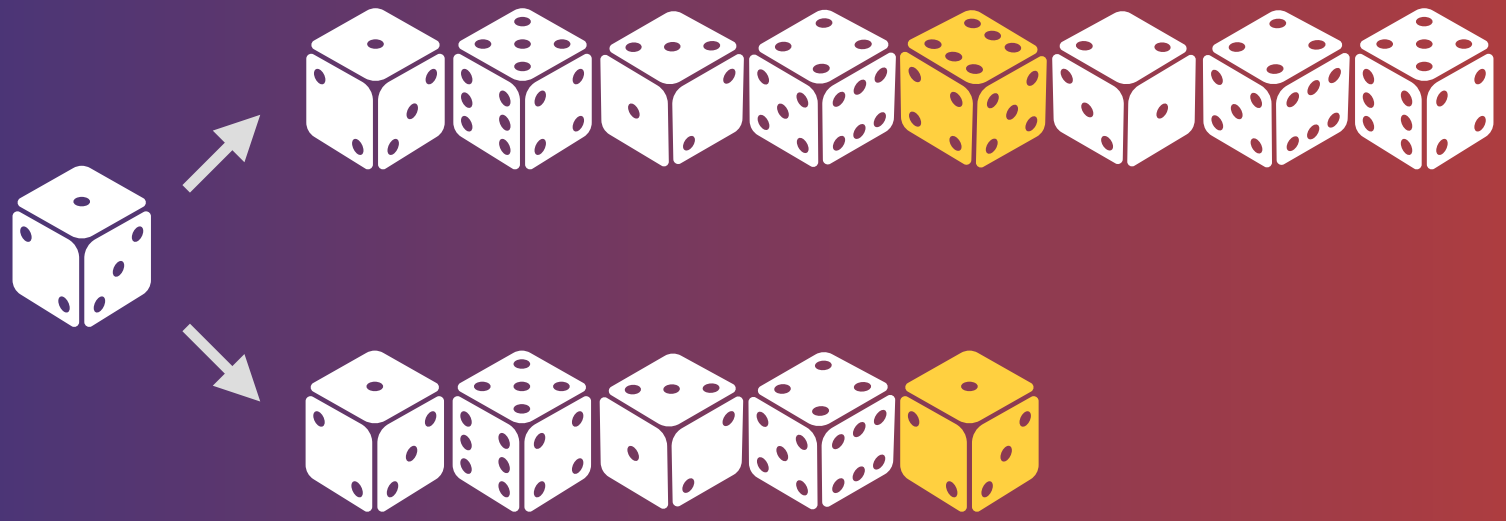
```
futures::select_biased!
tokio::select!
```

```
HashMap.iter()
```

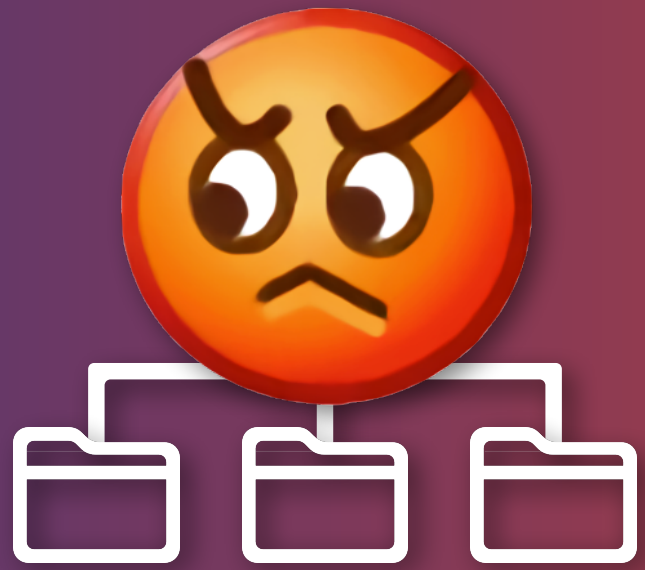
```
BTreeMap.iter()
```

NON-DETERMINISM

DETECTION

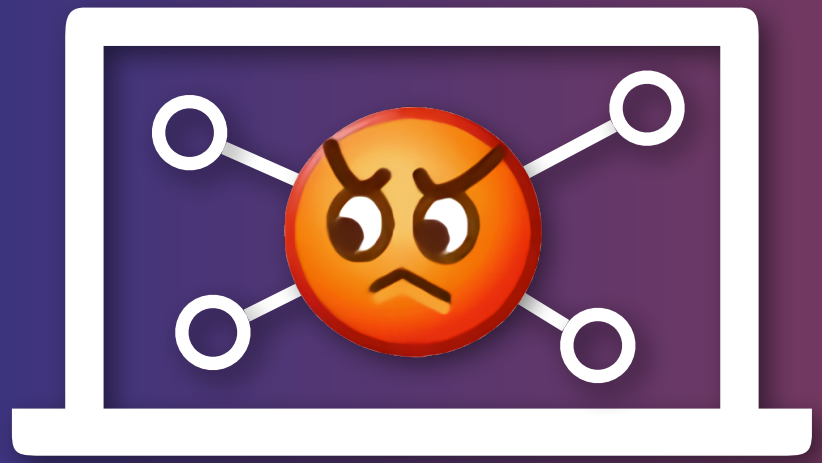


NON-DETERMINISM



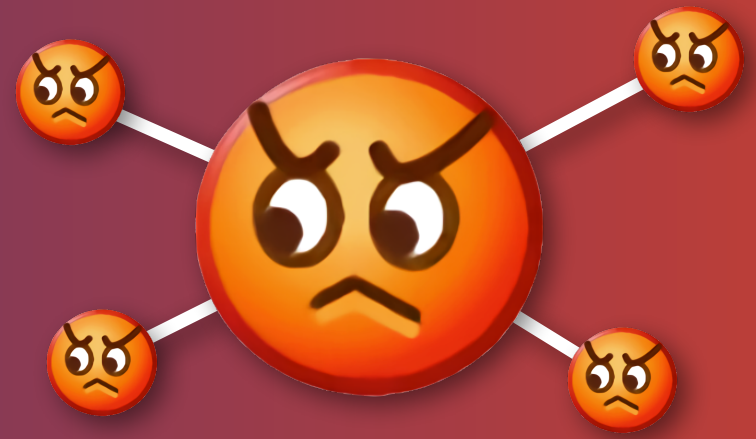
MadFS

打榜~~专用~~分布式文件系统



MadSim SIM

确定性模拟环境

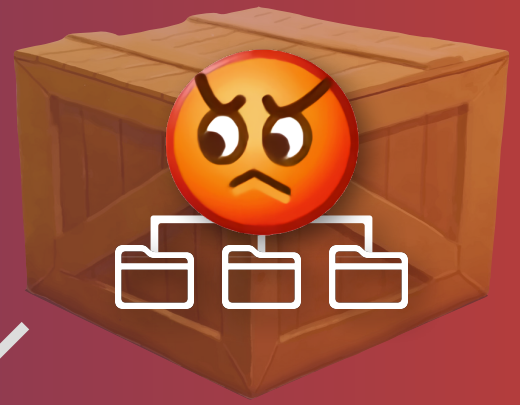


MadSim STD

对接真实环境

MadRaft

MadFS

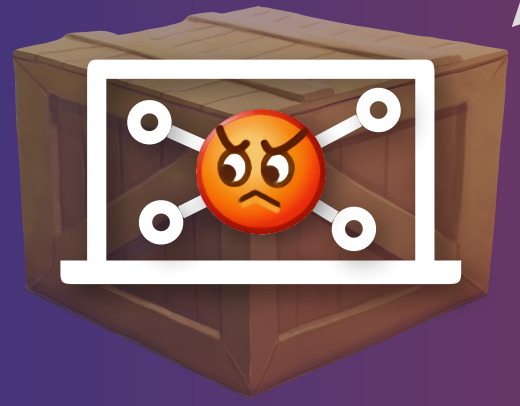


MadSim

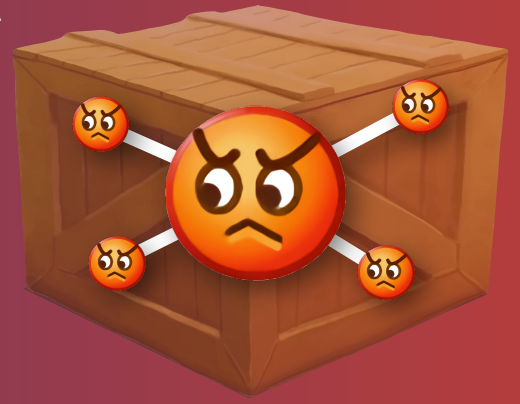
`#[cfg(madsim)]`

`#[cfg(not(madsim))]`

MadSim SIM

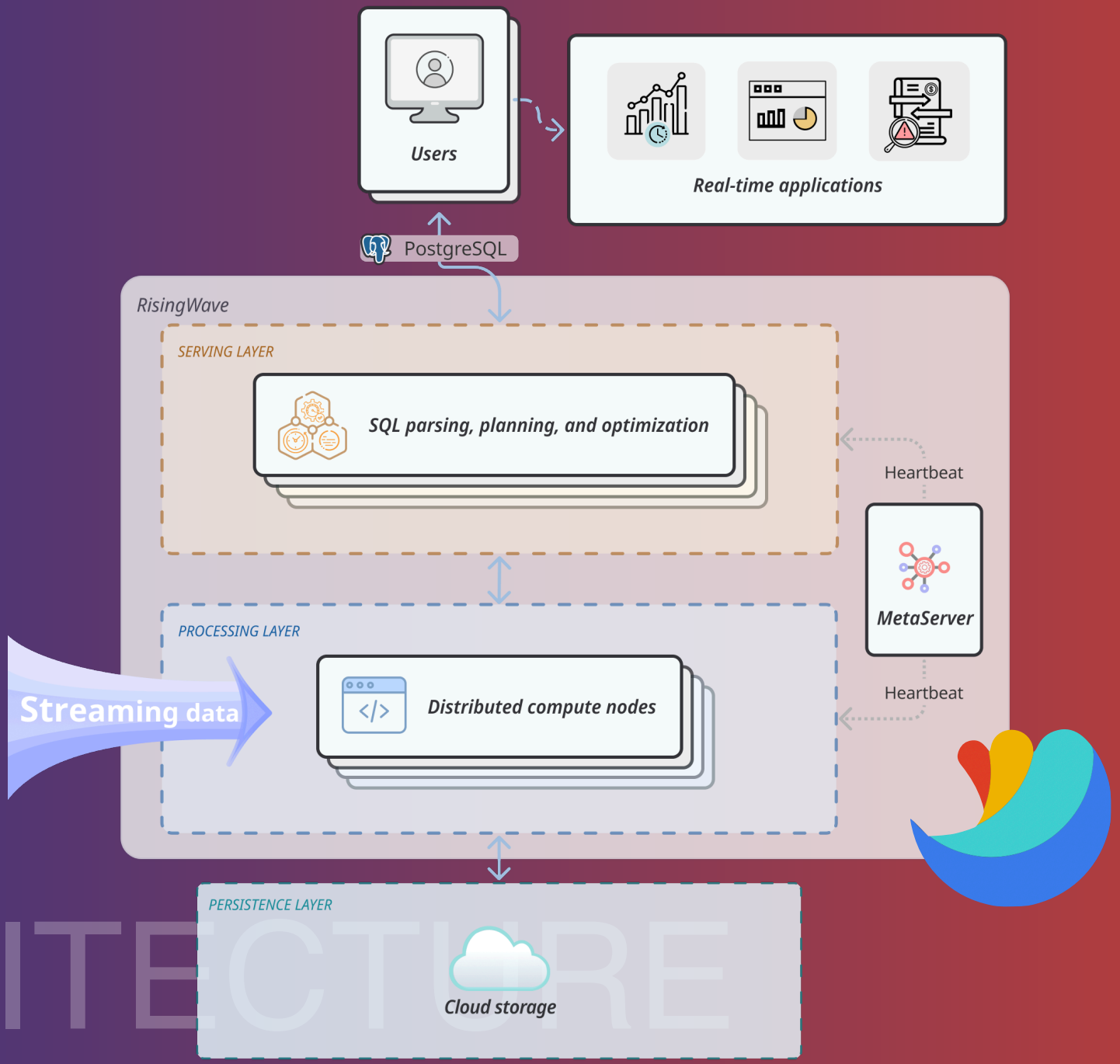


MadSim STD

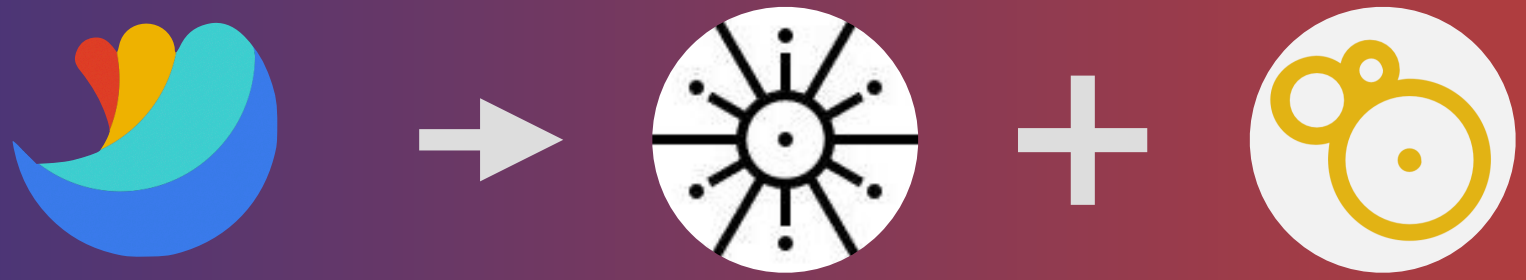


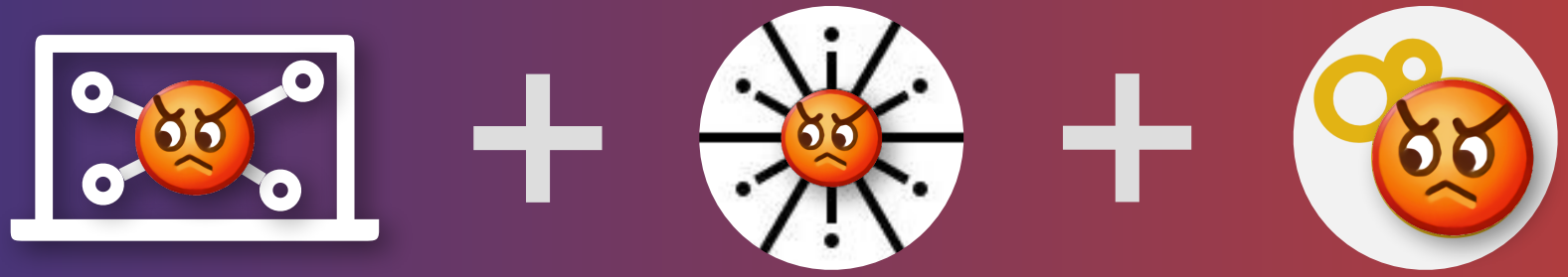
RisingWave

A Cloud-Native Streaming Database in Rust



ARCHITECTURE





DEPENDENCIES

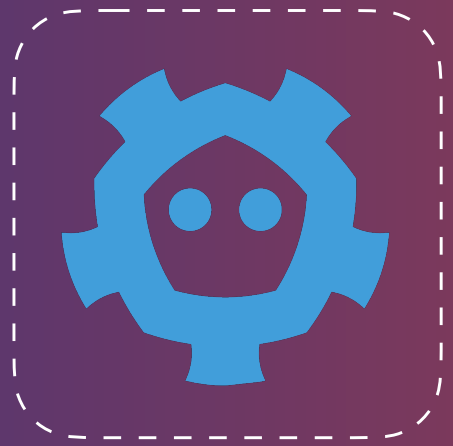
REPLACEMENT

```toml
[dependencies]
madsim = "=0.2.0-alpha.4"
tokio = { version = "=0.2.0-alpha.4", package = "madsim-tokio" }
tonic = { version = "=0.2.0-alpha.4", package = "madsim-tonic" }
[dev-dependencies]
tonic-build = { version = "=0.2.0-alpha.1", package = "madsim-tonic-build" }
```

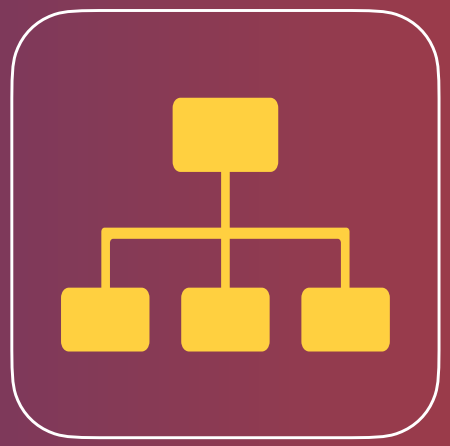
```
RUSTFLAGS="--cfg madsim" cargo test
```

DROP-IN

S3



etcd



FS

RPC

COMPONENTS

分布式系统确定性模拟

👍 Pros

稳定复现 Bug
大幅提升调试效率

模拟速度快
大幅提升发现问题概率

✅ 事务处理
✅ 一致性算法

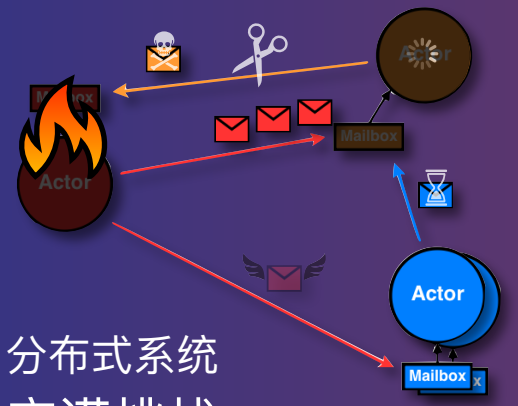
👎 Cons

框架依赖性强
难以整合第三方库

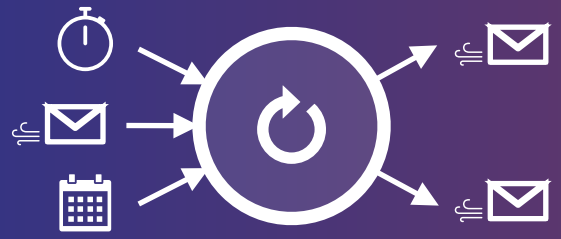
不确定性消除困难

模拟可能失真

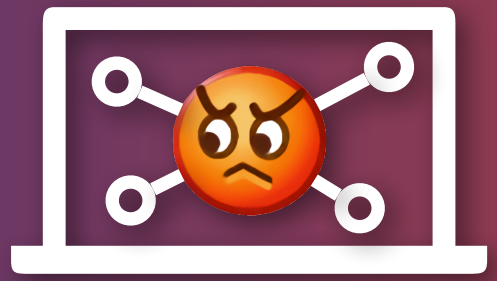
DISCUSSION



分布式系统
充满挑战



构建分布式系统的正确姿势
状态机 + 模拟器



MadSim

Rust 分布式系统
确定性模拟器

MadRaft

基于确定性模拟的 Raft 实验
大幅缩短测试时间，稳定复现 Bug

RisingWave

在实际系统中的应用

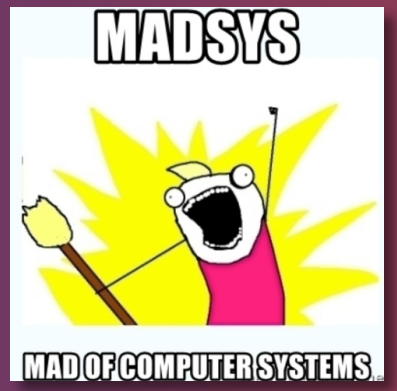
RECAP

欢迎体验

MadRaft

“May Lamport be with you!”

欢迎关注



madsys

“该学习学习计算机系统了”

欢迎加入

Singularity Data

“Innovating the next-gen cloud-native streaming database.”

THANKS!

Thanks

Rust China Conf 2021-2022 – Online, China