Why write a new Executor ?

- MultithreadedExecutor has different scheduling behavior compared to Executor
- Especially if a timer ‚overruns' it is directly executed again
  - Can lead to life lock

- We wanted a Multithreaded Executor, without the timer problems
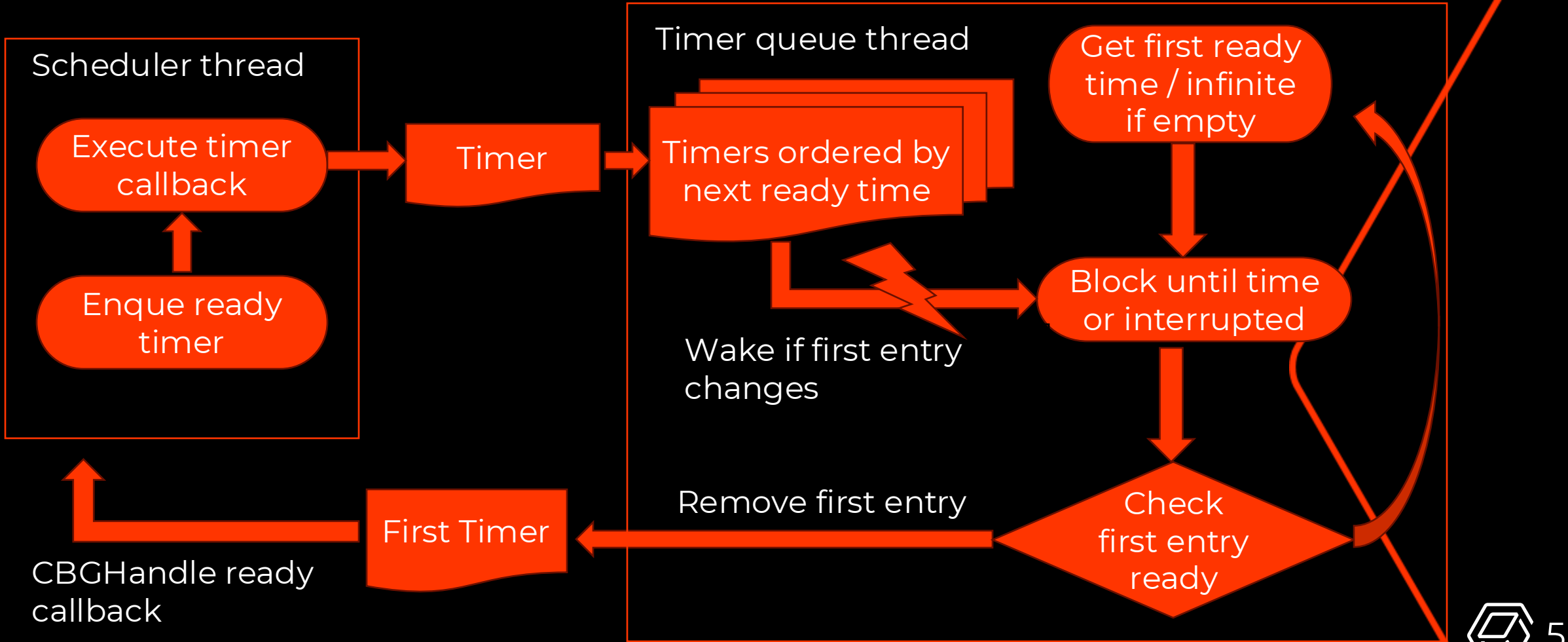- Mutated into a Scheduler Framework

2

CallbackGroupEventsExecutor

- Drop in replacement
  - Custom Waitables need to implement events functions
- No usage of waitsets
- Subsystems
  - TimerManager
  - RegisteredEntityCache
  - ReadyEntity
  - Scheduler

3

TimerManager

- Has a TimerQueue per clock type
  - ROS_TIME
  - SYSTEM_TIME
  - STEADY_TIME
- Provides callbacks when a registered timer becomes ready
- Needed to replace built in functionality of rcl_wait

4

TimerQueue

Scheduler thread
- Execute timer callback
- Enque ready timer

Timer

Timer queue thread
- Timers ordered by next ready time
- Get first ready time / infinite if empty
- Block until time or interrupted
- Wake if first entry changes
- Check first entry ready
- Remove first entry

First Timer

CBGHandle ready callback

5

RegisteredEntityCache

- Tracks entities added to a callback group
- Manages setting and removing of ‚ready' callbacks to RMW
- Requests per entity in the callback group a ‚ready' handler from the scheduler
  - Registers handlers of Subscriber, Client, Service, Waitable and GuardCondition directly
  - Registers timers with handler at the TimerManager

ReadyEntity

- Wrapper around a ready entity
- Created whenever a entity becomes ready
- During construction a monotonic unique id is assigned
- Ensures existence of entity during lifetime of ‚execute_function'
- Unified interface for
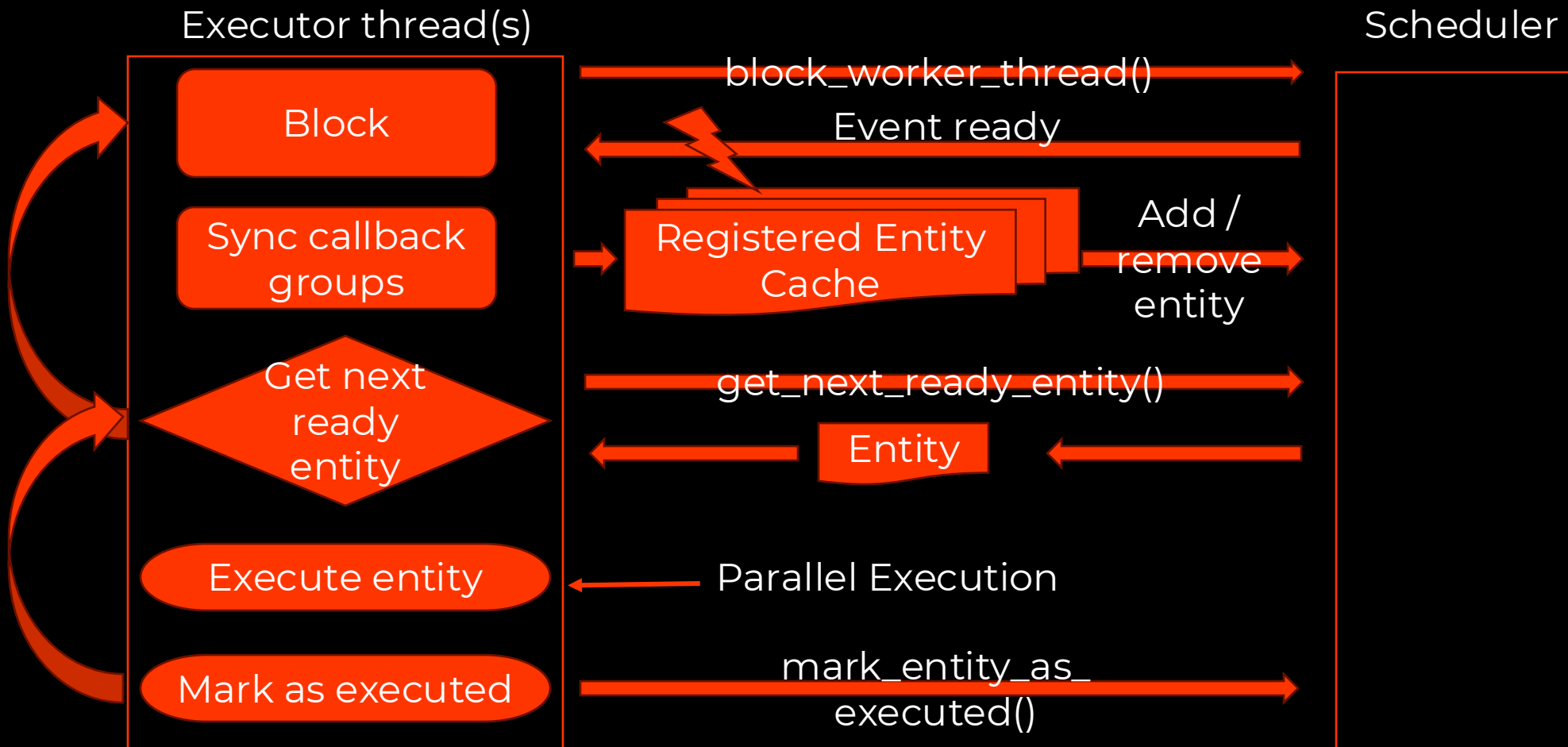 - expired()
  - get_execute_function()

Scheduler

- Interface class
- Provides:
  - Worker thread wakeup / blocking
  - Managed list of ready CallbackGroupHandle
- Scheduler specific:
  - get_handle_for_callback_group()
    - Will be called for every callback group registered at the executor.
  - get_next_ready_entity()
    - Will be called by the executor to query work.

8

CallbackGroupHandle

- Interface class
- Provides
  - Signaling to Scheduler that callback group has work
  - Bridge between RWM ready callbacks and scheduler
- Scheduler specific:
  - get_ready_callback_for_entity()
    - Returned function will be called by the RMW
  - Shall provide a list of ready entities of callback group
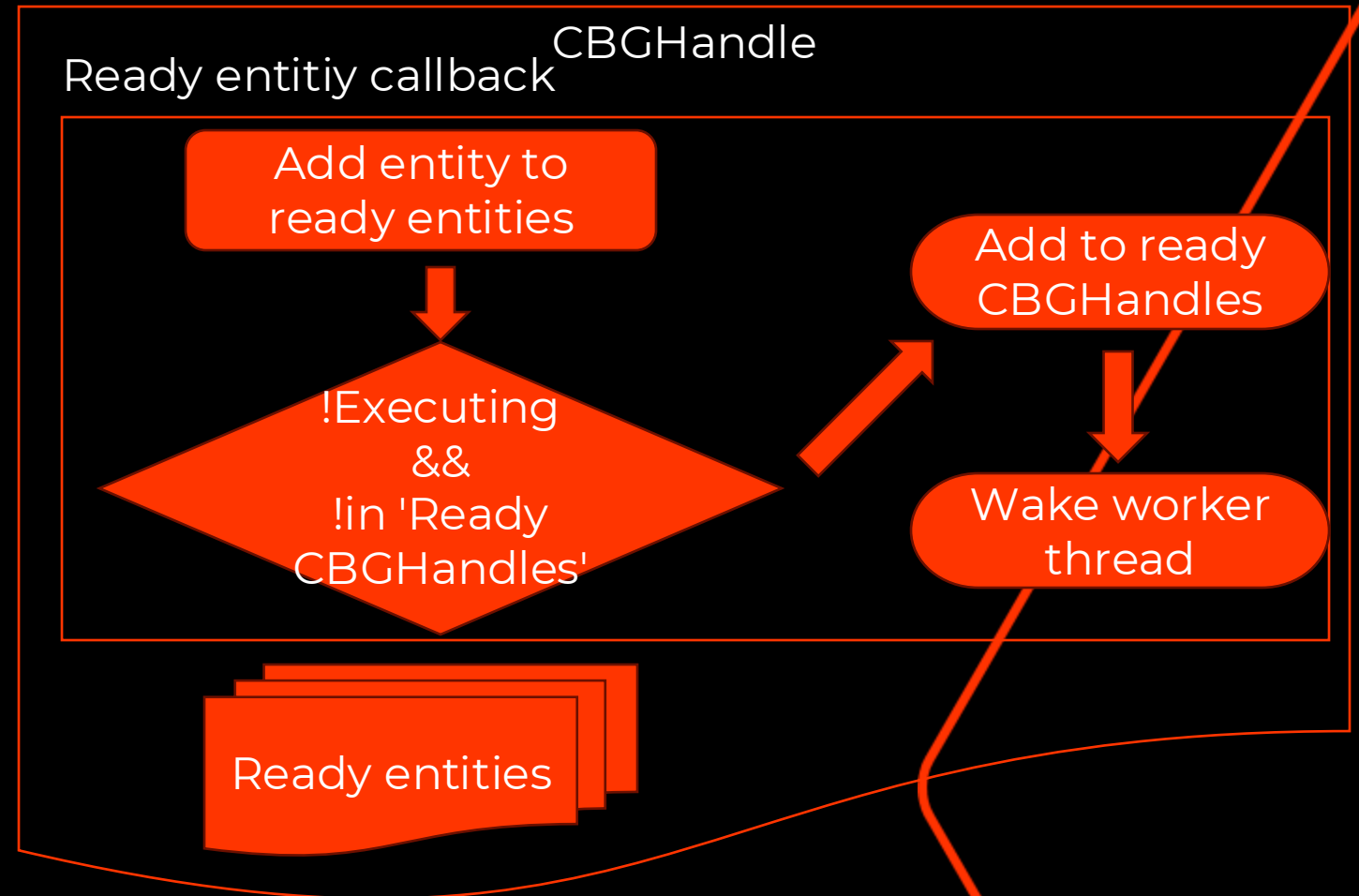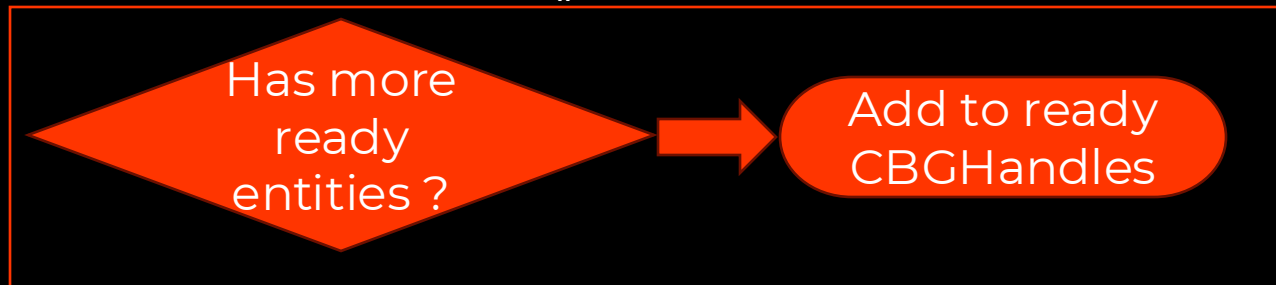
9

Usage of Scheduler by executor

Executor thread(s)

Scheduler

Block

block_worker_thread()

Event ready

Sync callback groups

Registered Entity Cache

Add / remove entity

Get next ready entity

get_next_ready_entity()

Entity

Execute entity

Parallel Execution

Mark as executed

mark_entity_as_executed()

FiFoScheduler

CBGHandle

Ready entitiy callback

get_next_ready_entity()

remove first entry of 'Ready CBGHandles'

remove and return first entry from 'Ready entities'

Add entity to ready entities

!Executing && !in 'Ready CBGHandles'

Add to ready CBGHandles

Wake worker thread

Ready entities

mark_as_executed()

Has more ready entities ?

Add to ready CBGHandles

Ready CBGHandles

11

spin_some (collect and execute)

Sync callback groups

Get last assigned monotonic id → Get next ready entity

Get next ready entity → Next ready id > last assigned id → Return

Get next ready entity → Next ready id <= last assigned id → Execute entity → Mark as executed → Get next ready entity

12
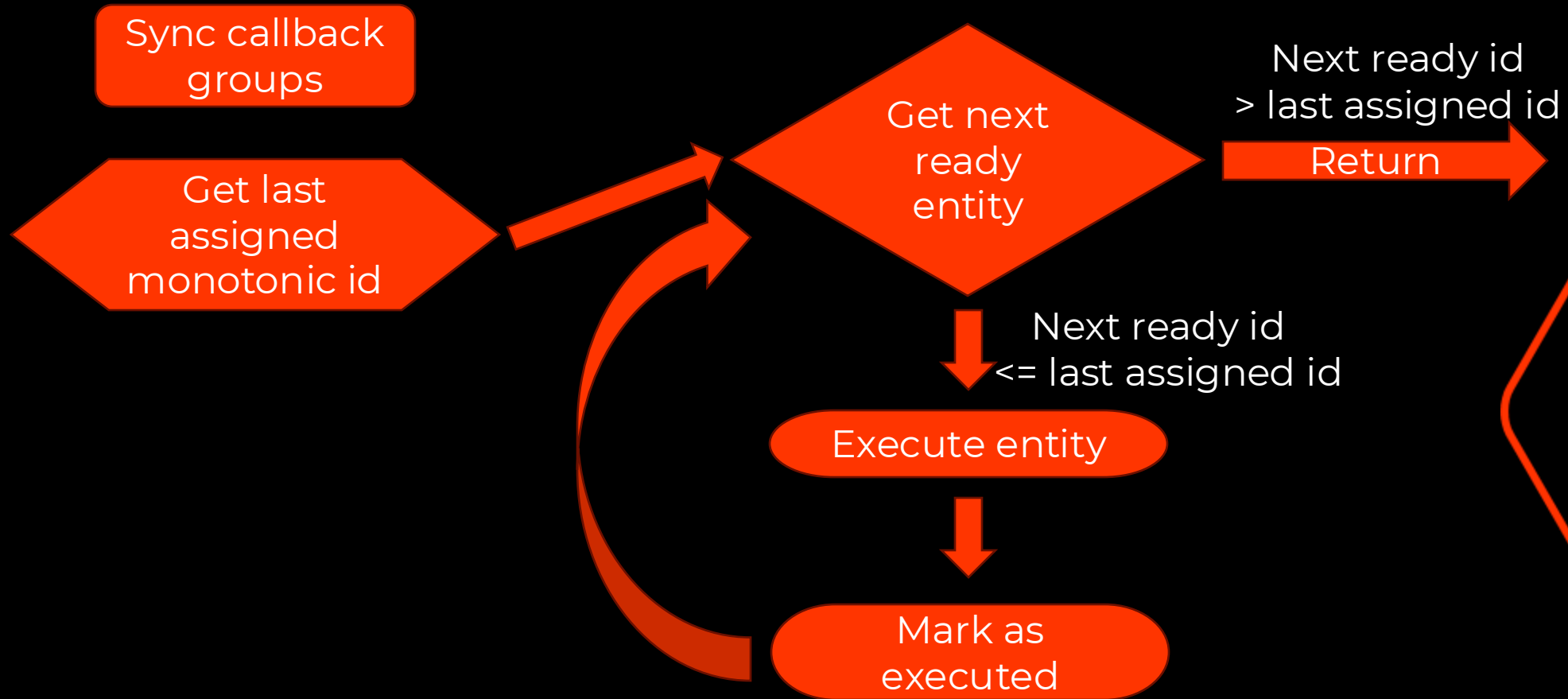
Micro benchmarks

Spin Some (1 Node, 10 Pubs, 10 Subs)

CBGEventsExecutor	
MultihreadExecutor	
SingleThreadExecutor	

0 — 5 — 10 — 15 — 20

■ Time in us
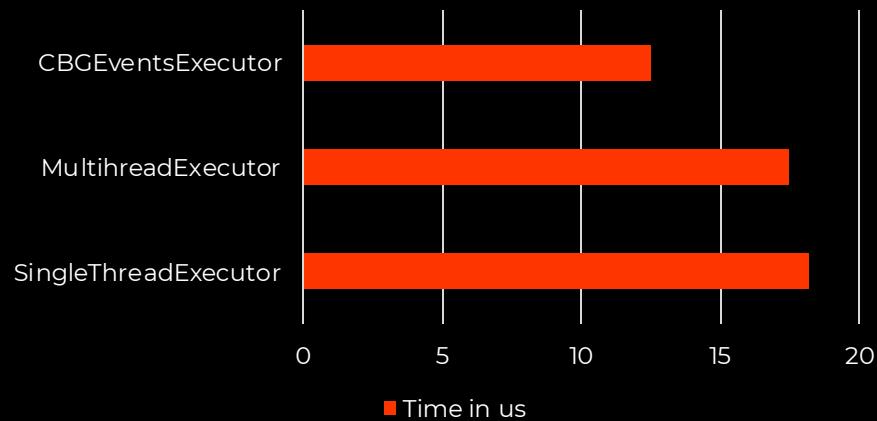
Cascaded triggering (1 Node, 10 Pubs, 10 Subs)

CBGEventsExecutor	
MultihreadExecutor	
SingleThreadExecutor	

0 — 5 — 10 — 15 — 20 — 25

■ Time in us
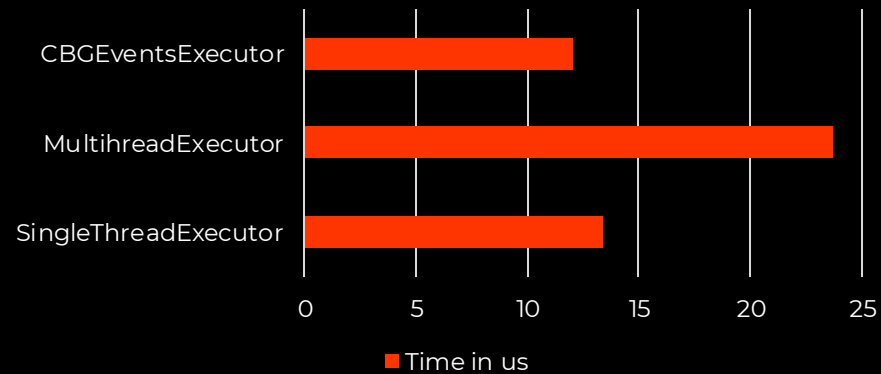
Spin Some (10 Nodes, 100 Pubs, 100 Subs)

CBGEventsExecutor	
MultihreadExecutor	
SingleThreadExecutor	

0 — 500 — 1000 — 1500 — 2000 — 2500 — 3000 — 3500

■ Time in us
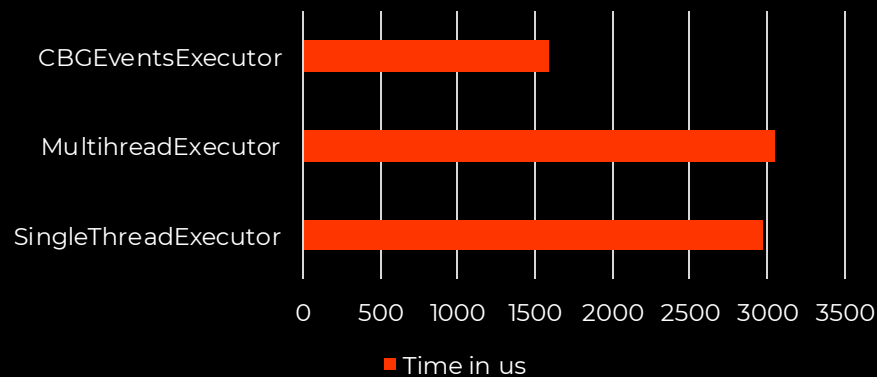
Cascaded triggering (10 Nodes, 100 Pubs, 100 Subs)

CBGEventsExecutor	59,37
MultihreadExecutor	13582,326
SingleThreadExecutor	4639,422

0 — 5000 — 10000 — 15000

■ Time in us

13
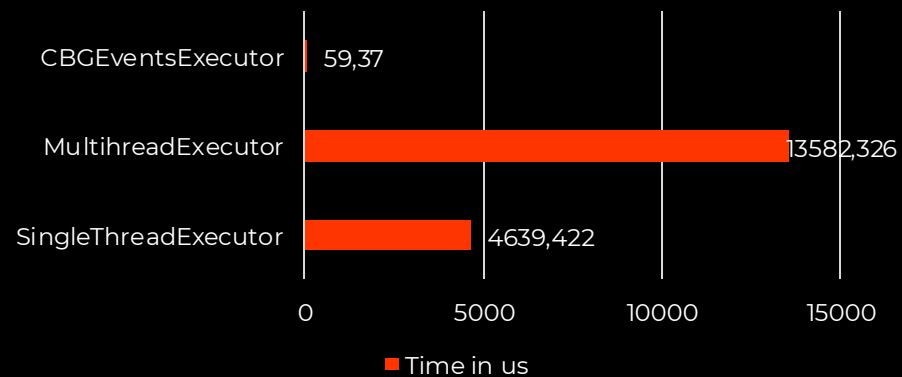
Real system simulation run

CallbackEventsExecutor   MultithreadedExecutor
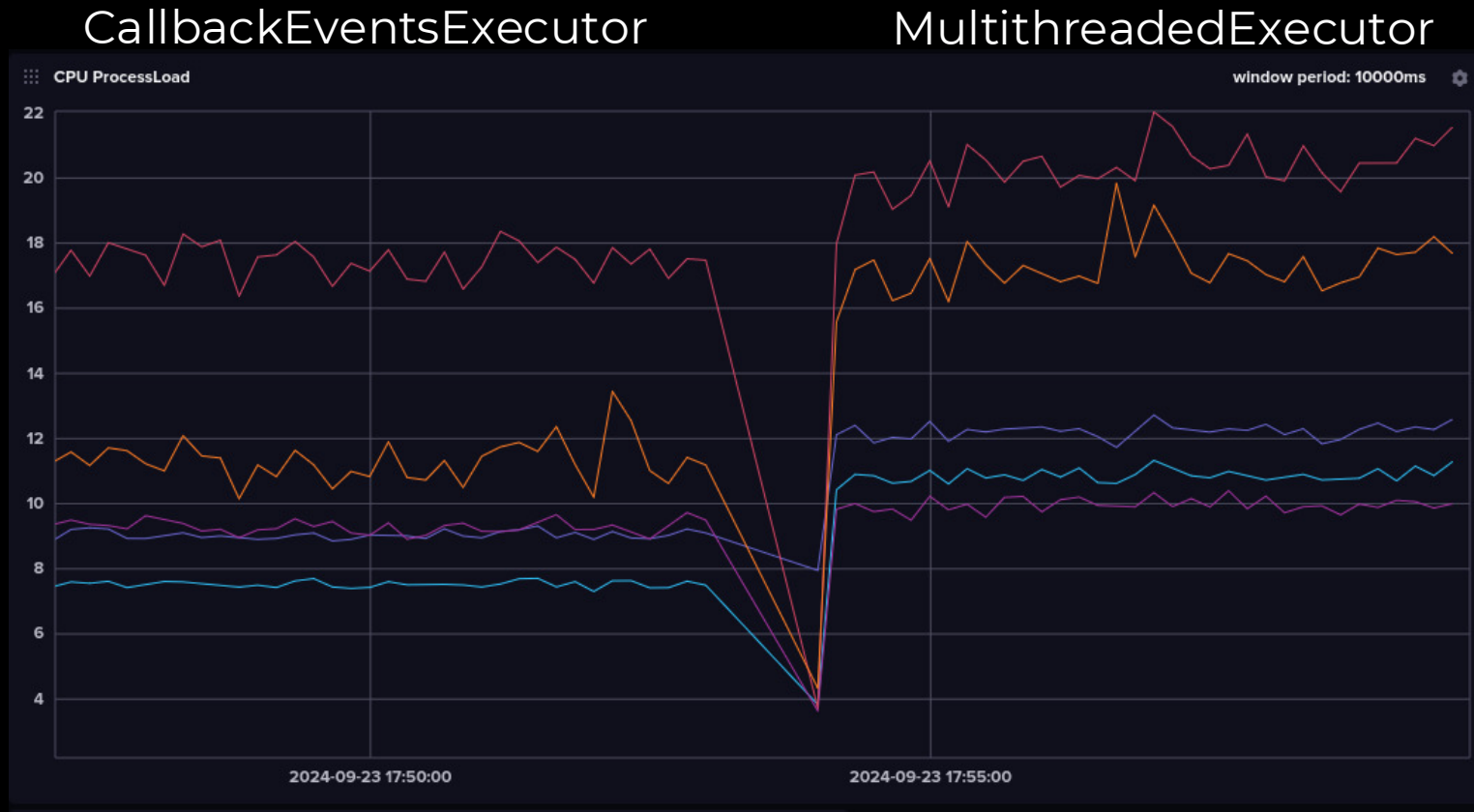
Each line is the cpu load of a Node

14

Snapshotter node

- Subscribed to 197 topics

CPU load average (5 min)

CPU load average (5 min)

15
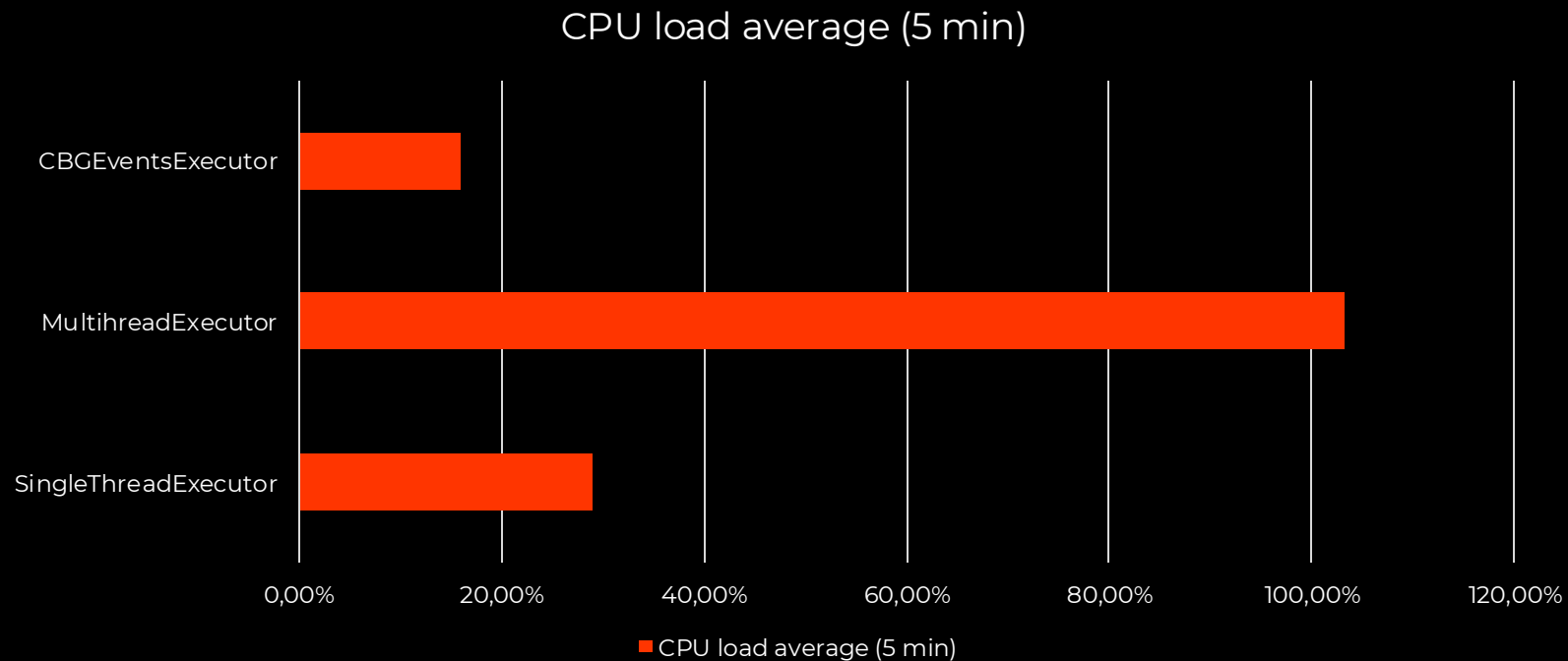
Big thanks to my company, for funding this work and my trip to ROSCon

Questions ?

Source:
github.com/cellumation/cm_executors