Operating System Support for Embedded Devices

February 7, 2024

Jonas Wozar

Friedrich-Alexander-Universität Erlangen-Nürnberg

Embedded Devices and Operating Systems

Embedded Devices:

- small
- usecase-specific
- limited resources

Operating Systems:

- FreeRTOS
- Zephyr
- ...

Question

Are multi-purpose OSs the best solution for embedded devices?

Table of Contents

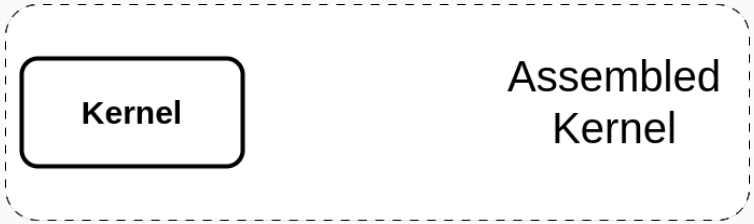
Related OS Concepts

Operating Systems Concepts

- Real Time Operating Systems (RTOS)
  - aimed at time-sensitive operations
- Unikernels
  - lightweight
  - designed to run a single application
- Exokernels
  - application-level ressource management
  - reduced OS abstraction
  - application-specific customization
- Library Operating Systems
  - customization with selected libraries
- ...

Tinkertoy

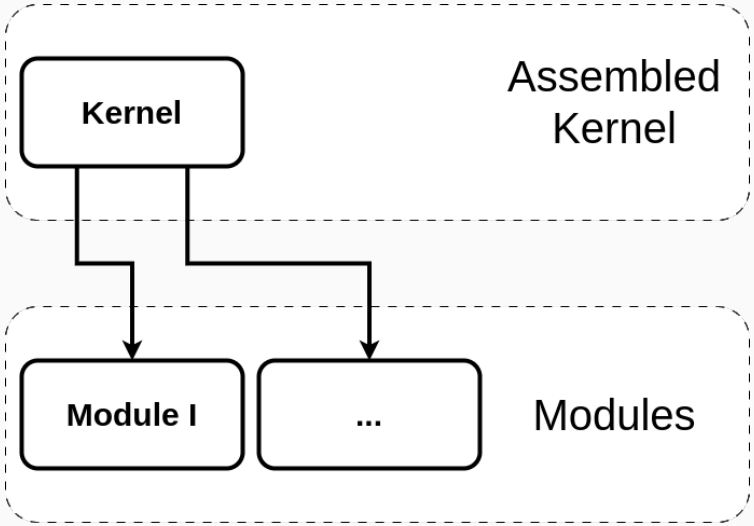
Overview



Tinkertoy

➔application-specific OS

Overview

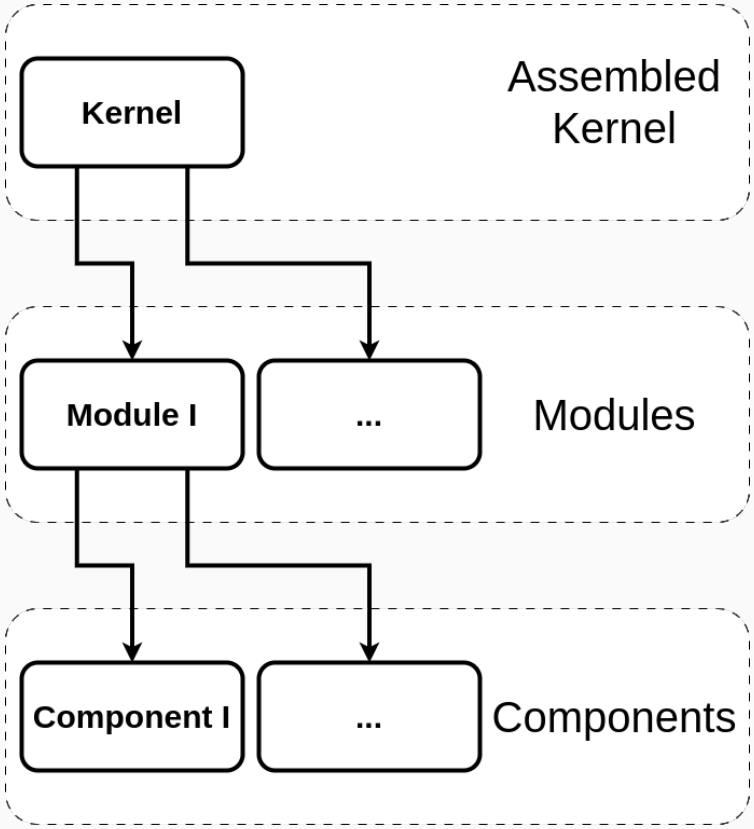


Tinkertoy

- set of modules

➔application-specific OS

Overview

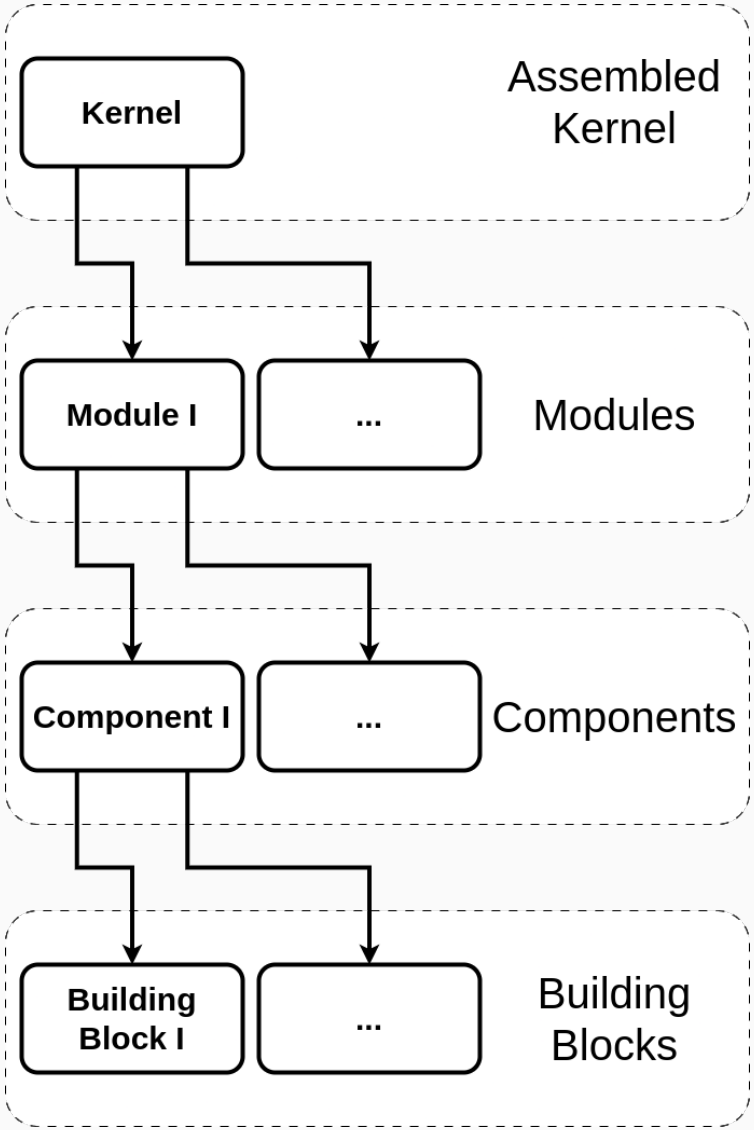


Tinkertoy

- set of modules
- components build modules

➔application-specific OS

Overview



Tinkertoy

- set of modules
- components build modules
- building blocks build components

➔application-specific OS

Modules

10 modules

- Constraints
- Scheduler
- Memory Allocator
- Context Switcher
- Execution State
- System Call
- Dispatcher
- Kernel Service Routines
- Execution Models
- Task Control Block

Modules

10 modules

- Constraints
- **Scheduler**
- Memory Allocator
- Context Switcher
- Execution State
- System Call
- Dispatcher
- Kernel Service Routines
- **Execution Models**
- **Task Control Block**

Templates

- generic definition of modules

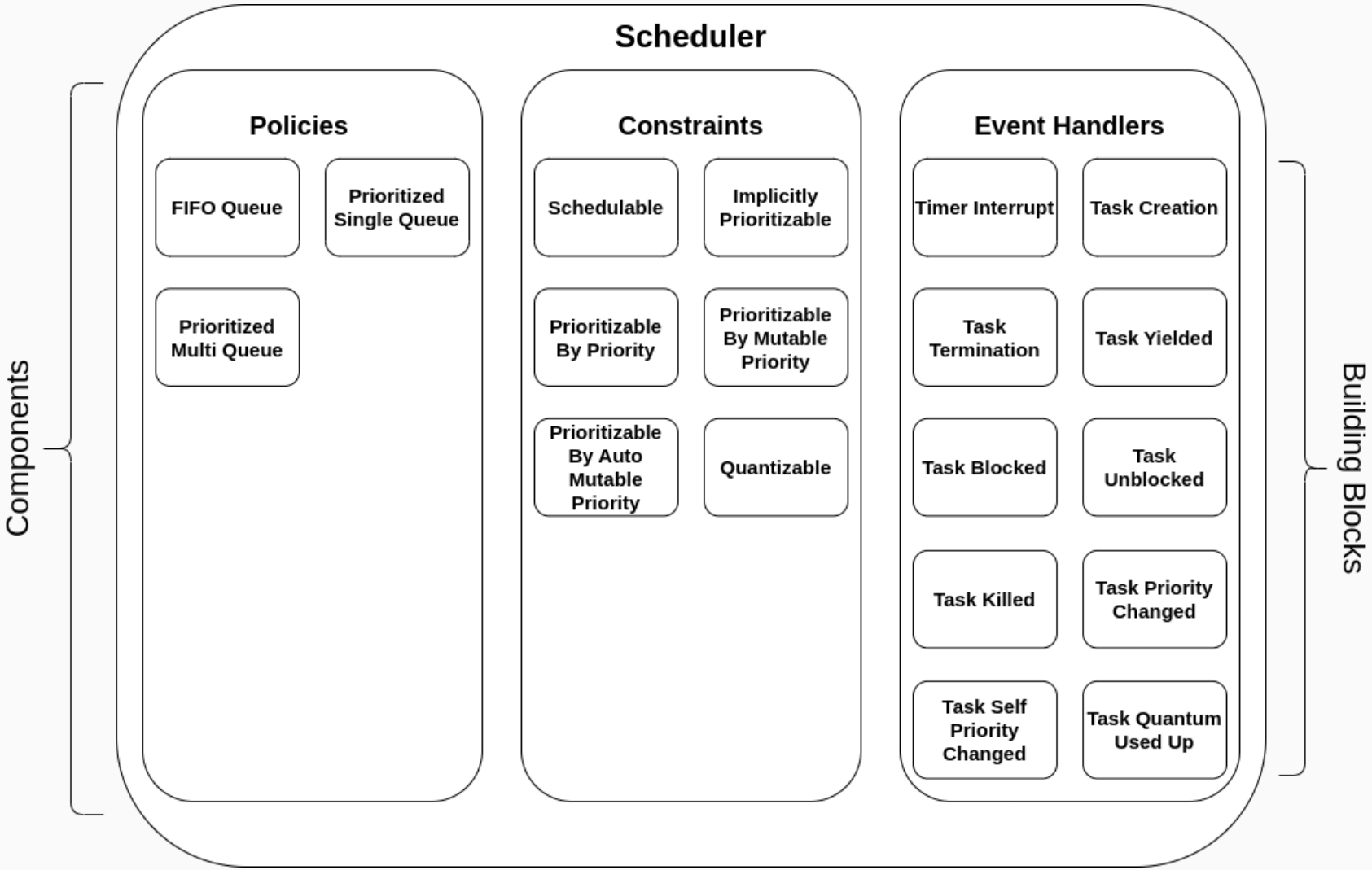
Concepts

- define constraints on types

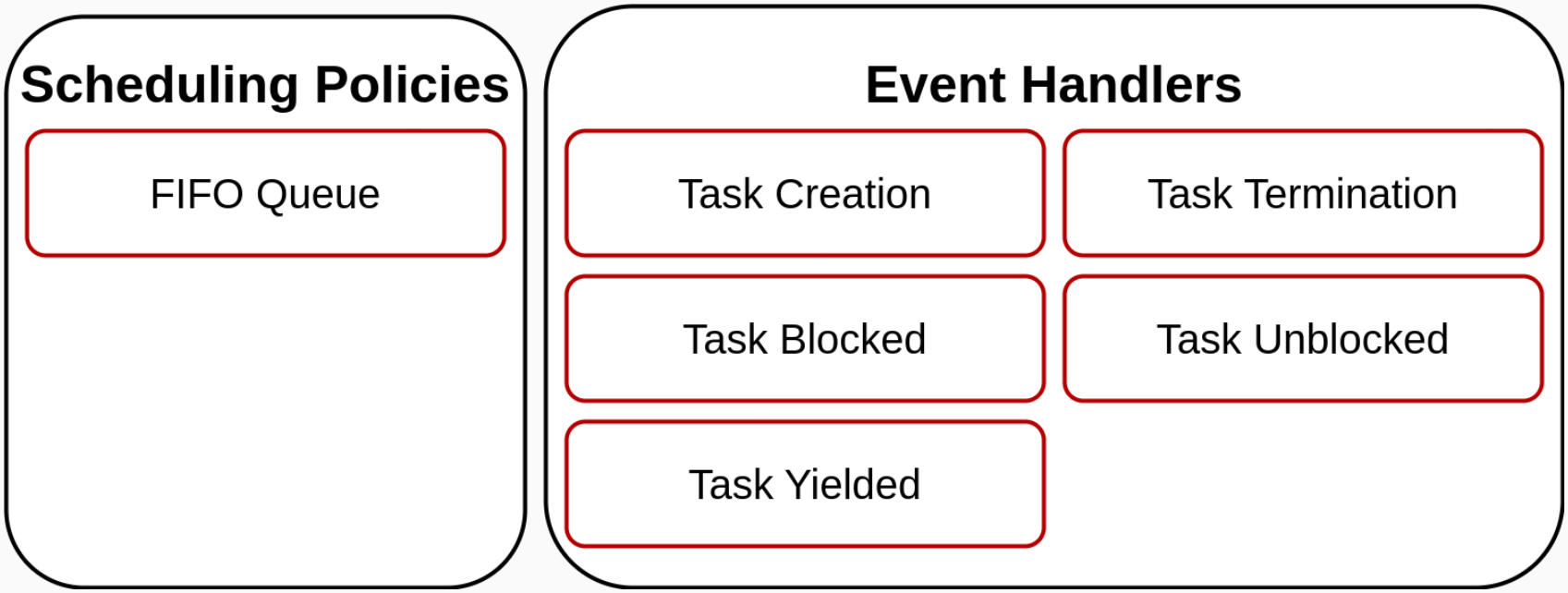
Functors

- encapsulate building blocks

Scheduler


Components
Scheduler
Policies
FIFO Queue
Prioritized Single Queue
Prioritized Multi Queue
Constraints
Schedulable
Implicitly Prioritizable
Prioritizable By Priority
Prioritizable By Mutable Priority
Prioritizable By Auto Mutable Priority
Quantizable
Event Handlers
Timer Interrupt
Task Creation
Task Termination
Task Yielded
Task Blocked
Task Unblocked
Task Killed
Task Priority Changed
Task Self Priority Changed
Task Quantum Used Up
Building Blocks

Example: FIFO Scheduler




```
using Policy = PolicyWithEnqueueExtensions<FIFO, Counter>;
class CustomFIFOScheduler : public SchedulerAssembler<Policy,
    TaskCreation::Cooperative::KeepRunningCurrent<Task>,
    TaskTermination::Common::RunNext<Task>,
    TaskBlocked::Common::RunNext<Task>,
    TaskUnblocked::Cooperative::KeepRunningCurrent<Task>,
    TaskYielded::Common::RunNext<Task>> {}
```

Execution Models

Event-Driven

- single/few thread(s)
- can be expressed as state-machine
- events define control flow
- example: automatic shades

Thread-Based

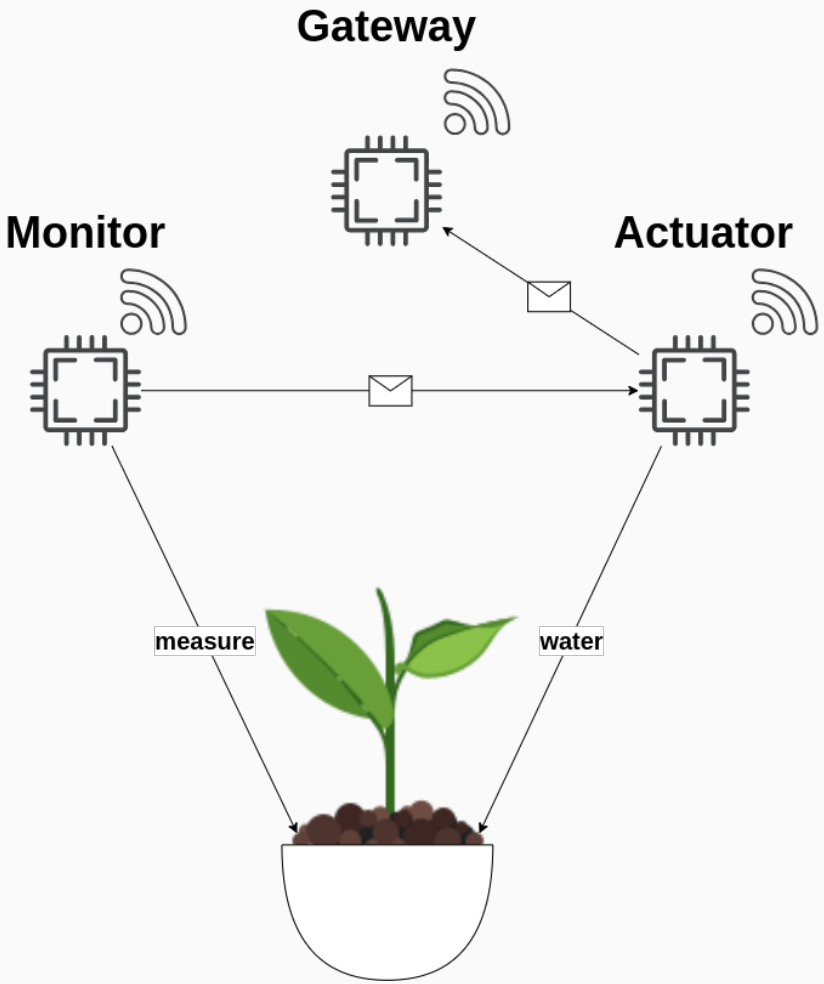
- number of threads
- typically short-lived
- concurrent task handling
- example: gateways

Task Control Block

- contain information about a task
- required for every task
- can be built from building blocks
- constraints (stack, system calls, ID, priority, state)
- initializer & finalizer

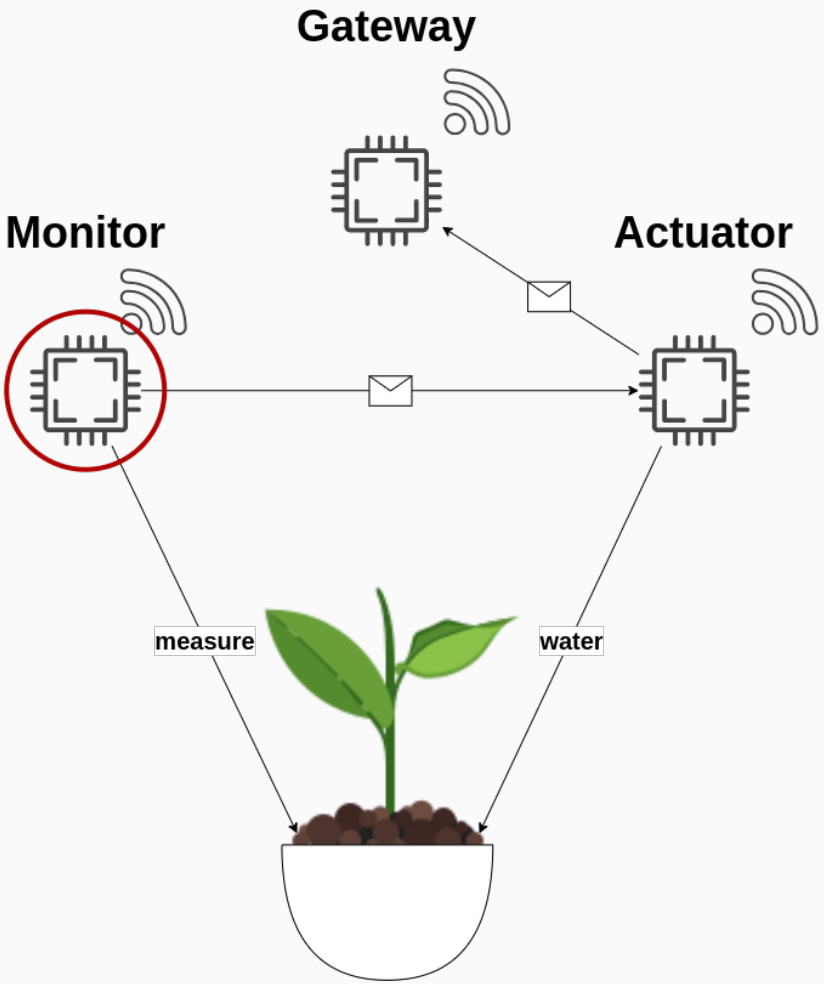
Evaluation

Setup



- Monitor
- Actuator
- Gateway

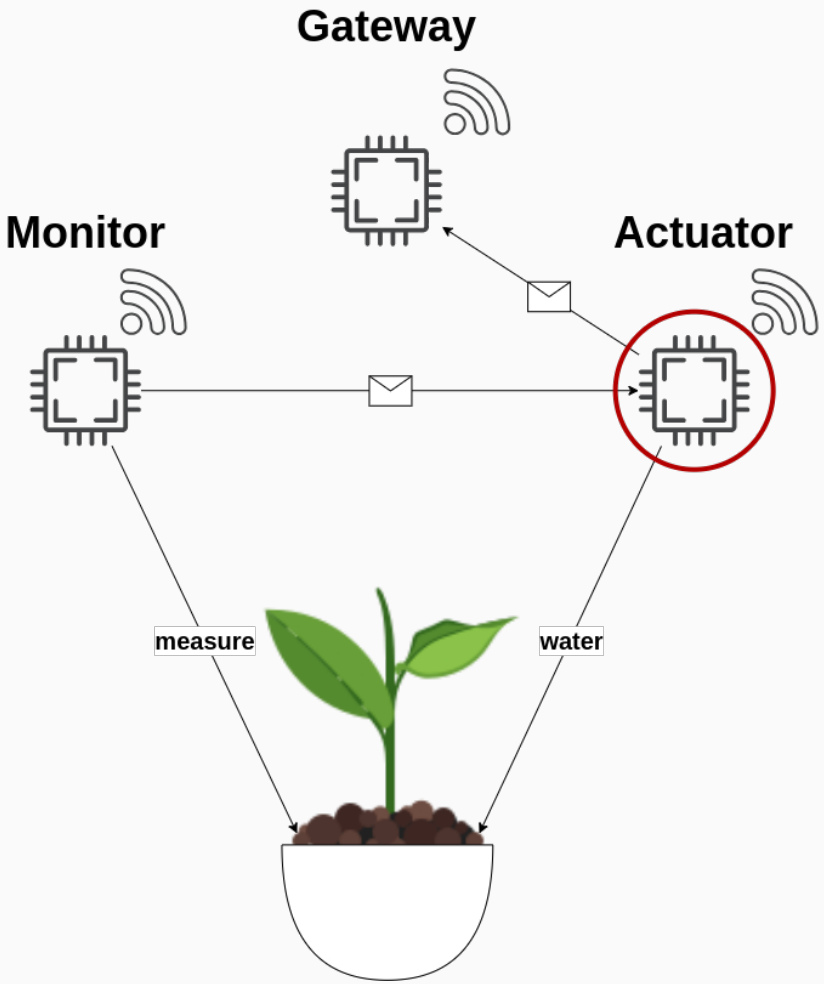
Monitor



Monitor

- event-driven
- measures soil moisture
- informs Actuator

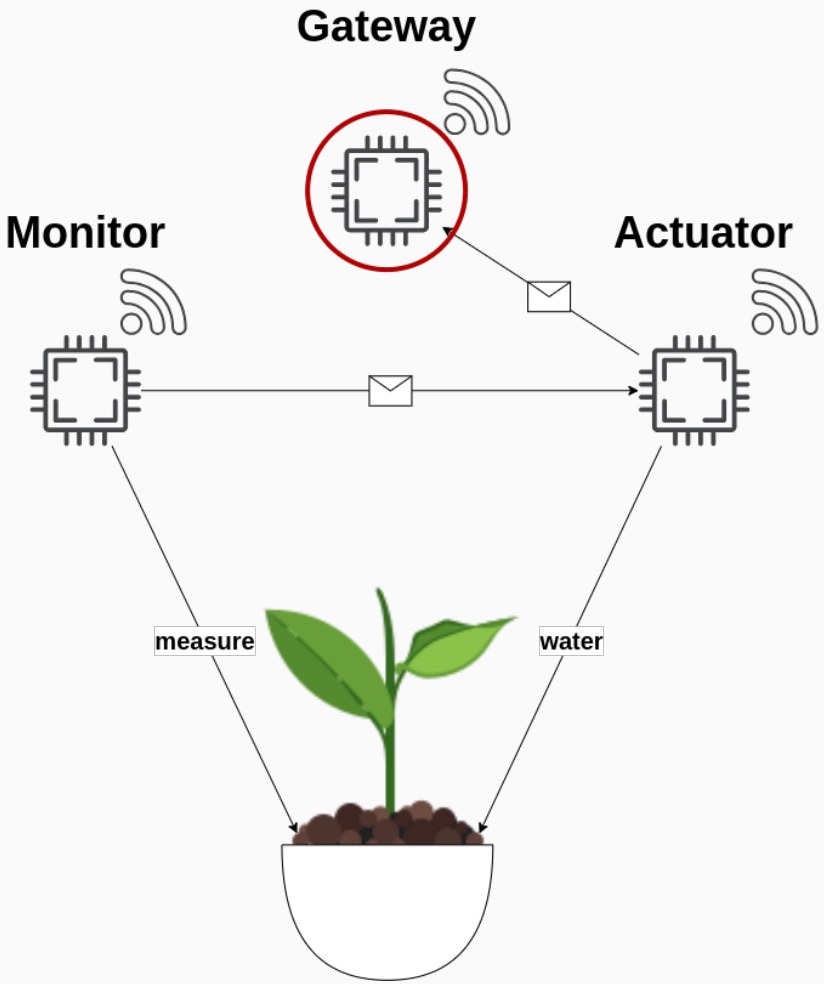
Actuator



Actuator

- event-driven
- starts/stops watering
- communicates with Gateway

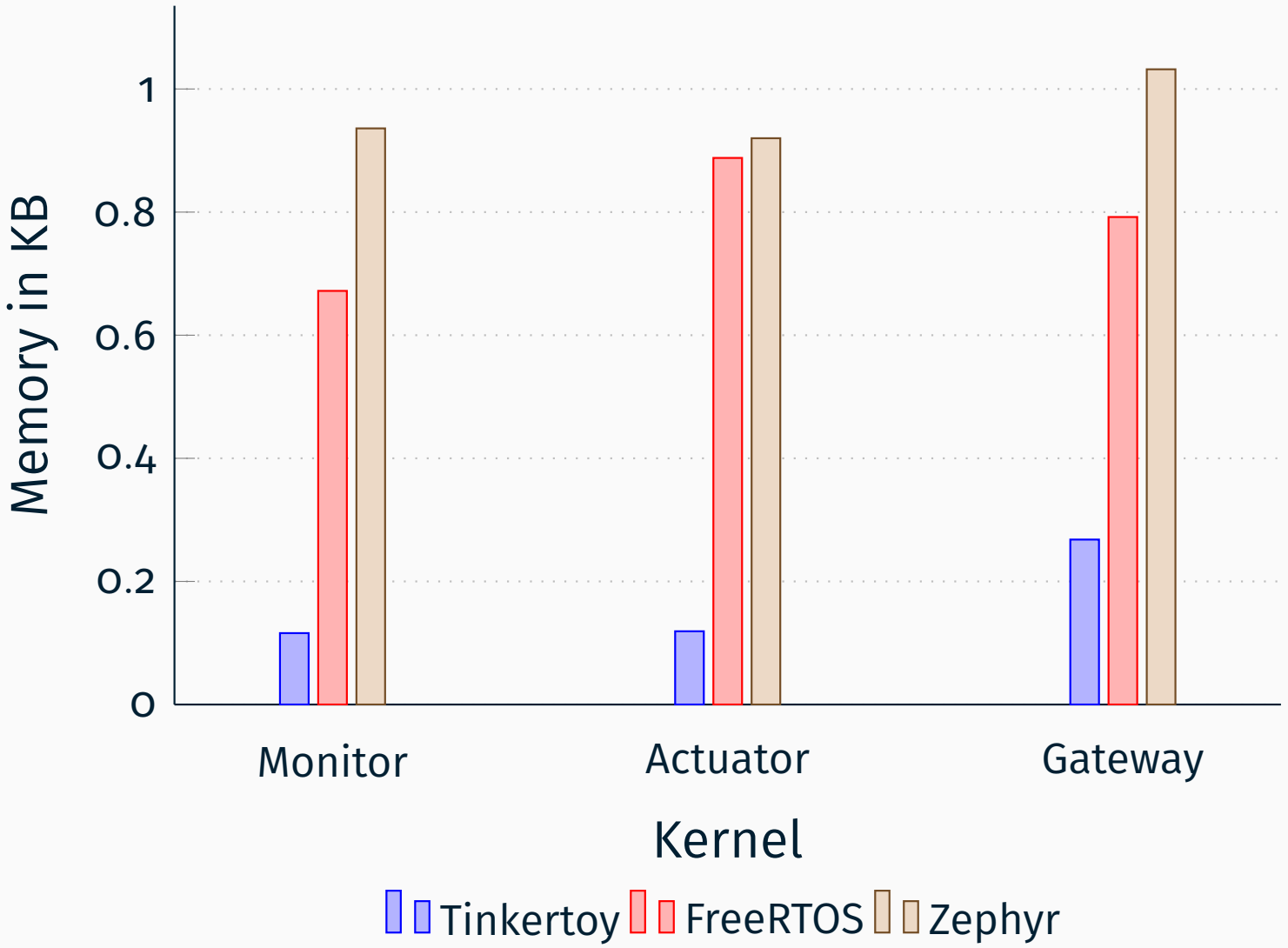
Gateway



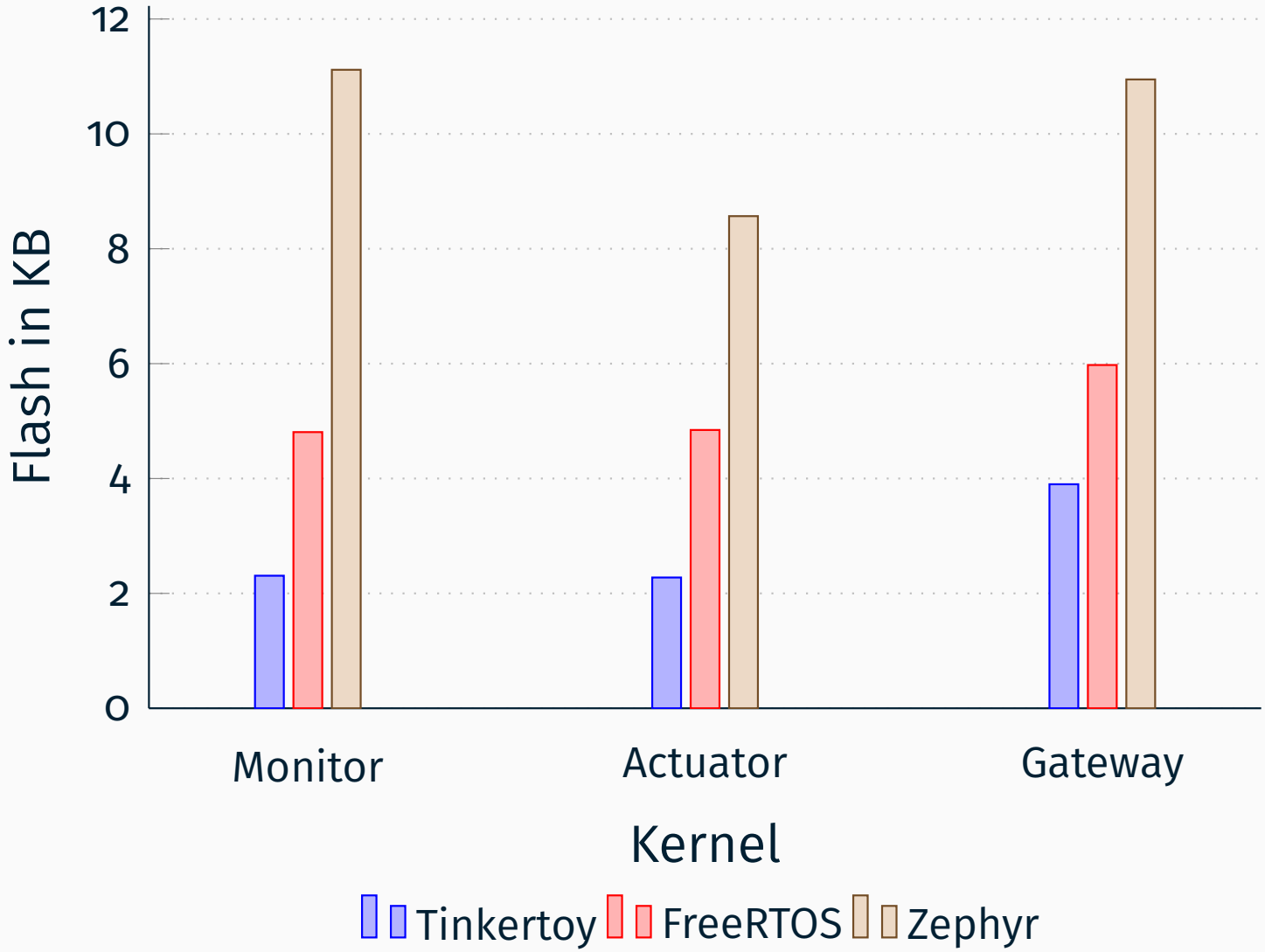
Gateway

- thread-based
- translates CoAP to HTTP

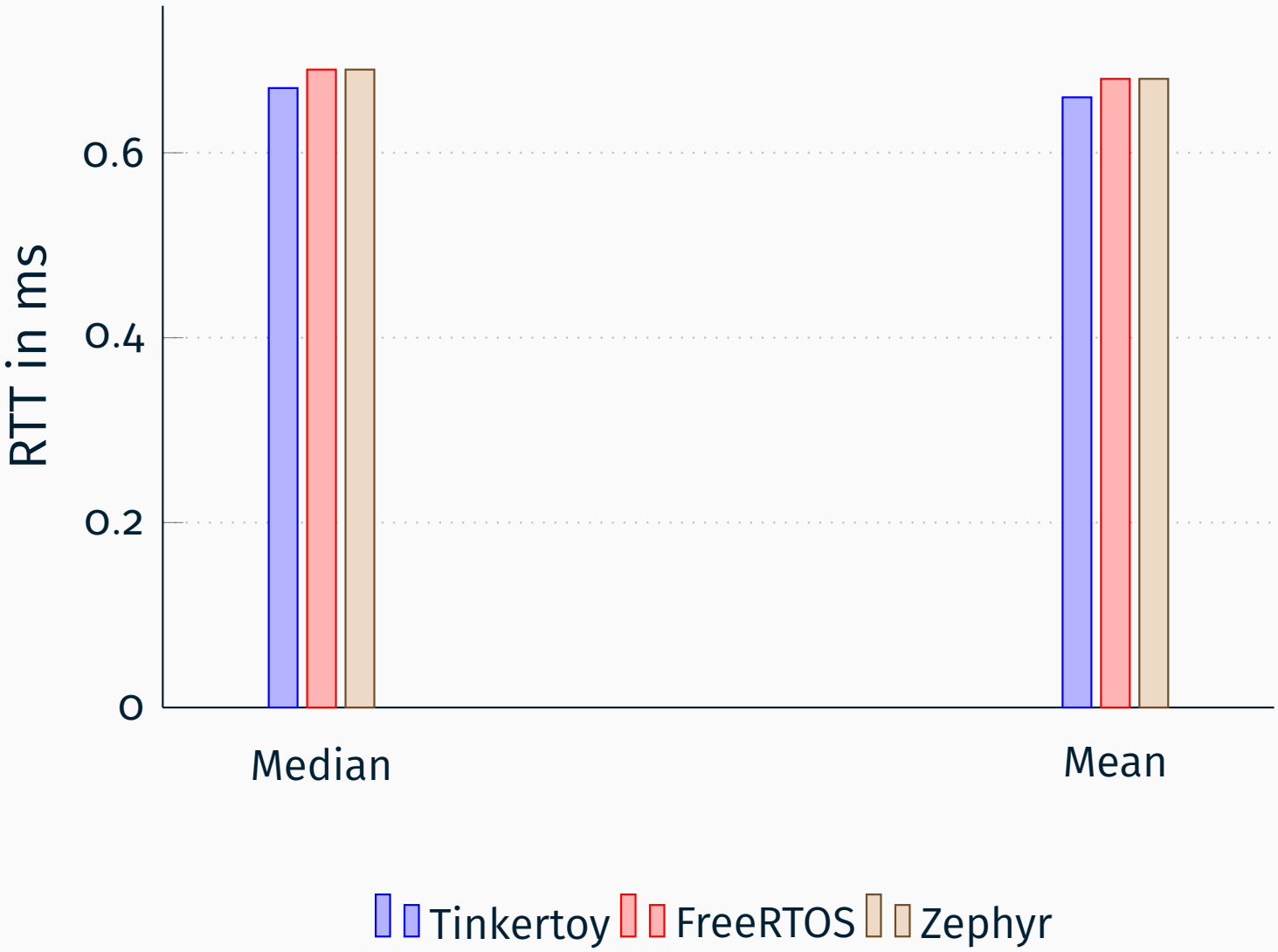
Memory Footprint



Flash Footprint



Performance



Conclusion

Conclusion

- few lines of code $\rightarrow$ usecase-specific OS
- significantly smaller memory footprint
- no performance impact
- no networking support
- no synchronization primitives

Questions?