Overview: Active Sensor networks

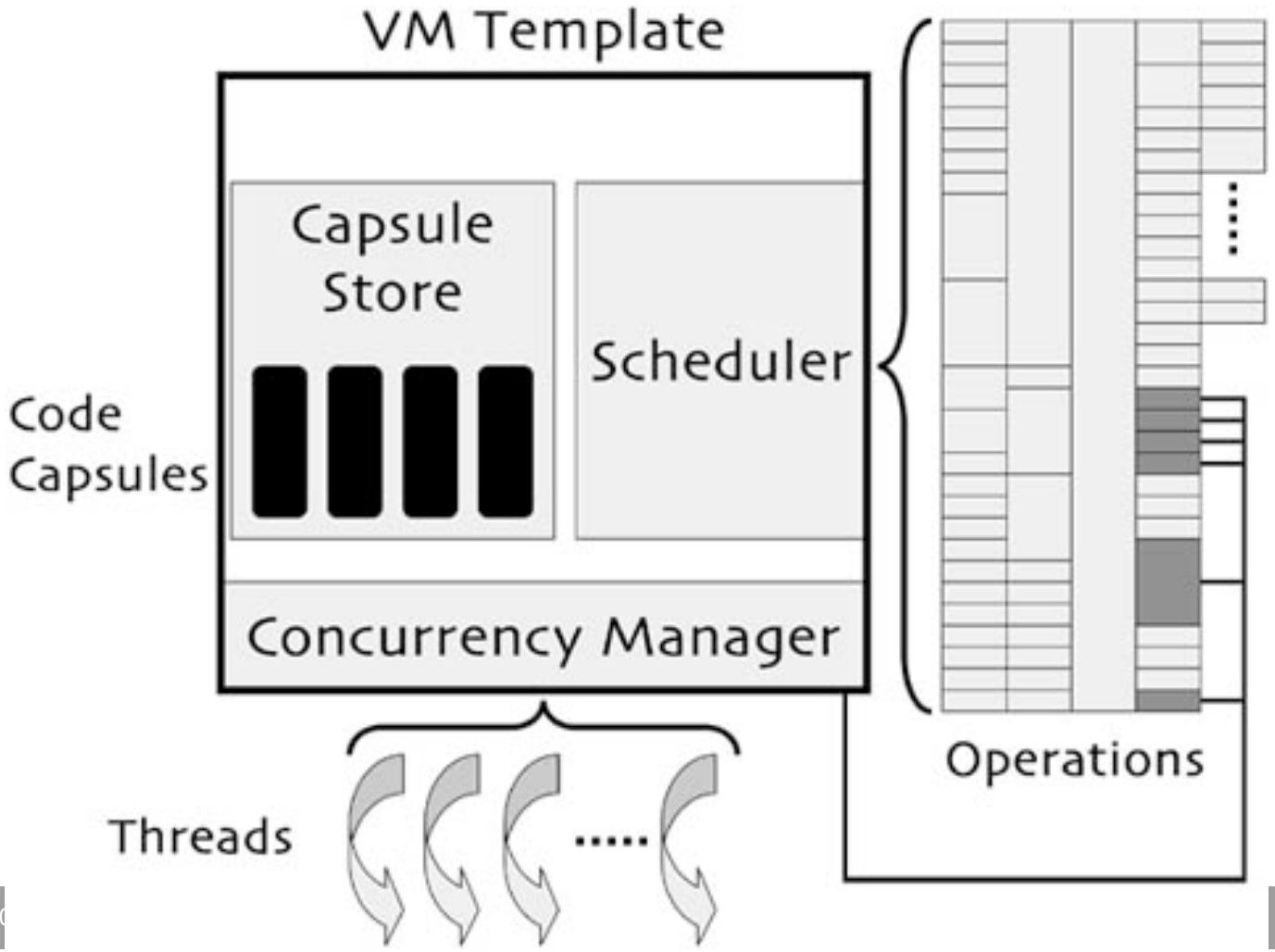
- Philip Levis, David Gay, and David Culler, **Active Sensor Networks**. In Proceedings of the Second USENIX/ACM Symposium on Networked Systems Design and Implementation (NSDI 2005)
  - Enabling dynamic programming on the sensor system
    - Recaliberate calibration after deployment
    - Post processing to reduce network traffic

Problem

- Sensor networks are deployed in areas where we don't know apriori what to expect
  - Dynamically reprogram sensors: change what they sense
- Question: How do we reprogram
  - send the new native code - could be large
  - send a script - new code depends on the script language
  - Code for a single VM (Mate) - not flexible
    - Limited functionality, not extensible for specific application requirements
  - JavaVM - too expensive for some applications
  - Send code as application specific virtual machine
    - Small size code, nearly as fast as native code
    - Extensible type support, concurrency control, code propagation

Design

- Each VM is made of a template
  - Operations: primitives (language specific), functions (language independent)



Data model

- Stack architecture
  - No program data beyond stack
- Scheduler: FIFO thread scheduler
- Concurrency manager: manage multiple handler threads to avoid race conditions
  - Threads only allowed to run when all resources are available
  - Reboots when new code arrives to reset all variable
- Capsule store: propagation
  - Selective execution, everyone has code, only some nodes execute it
- Code is trickled to other nodes using version vectors
  - Version packets, capsule status packets, capsure fragments

Evaluation

- Flexibility: Languages: Tinyscript, Motlle, Applications: RegionsVM, QueryVM
- Performance: With 6% of hand coded and 20% energy saving
  - Low overhead
    - Sensor CPU are mostly idle - not necessary to be as efficient as the native code