ComputerScience

Boston University

First Class Event-Driven Software Primitives

Dan Schatzberg, James Cadden, Orran Krieger, Jonathan Appavoo
Boston University, Computer Science Department

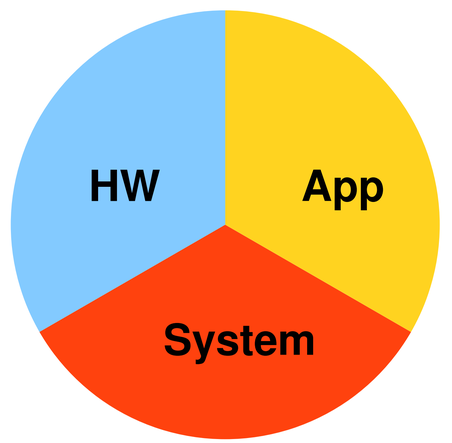
In cloud environments, programming with an event-driven paradigm is common. Webservers and other demand-driven applications map well to event-driven software primitives.

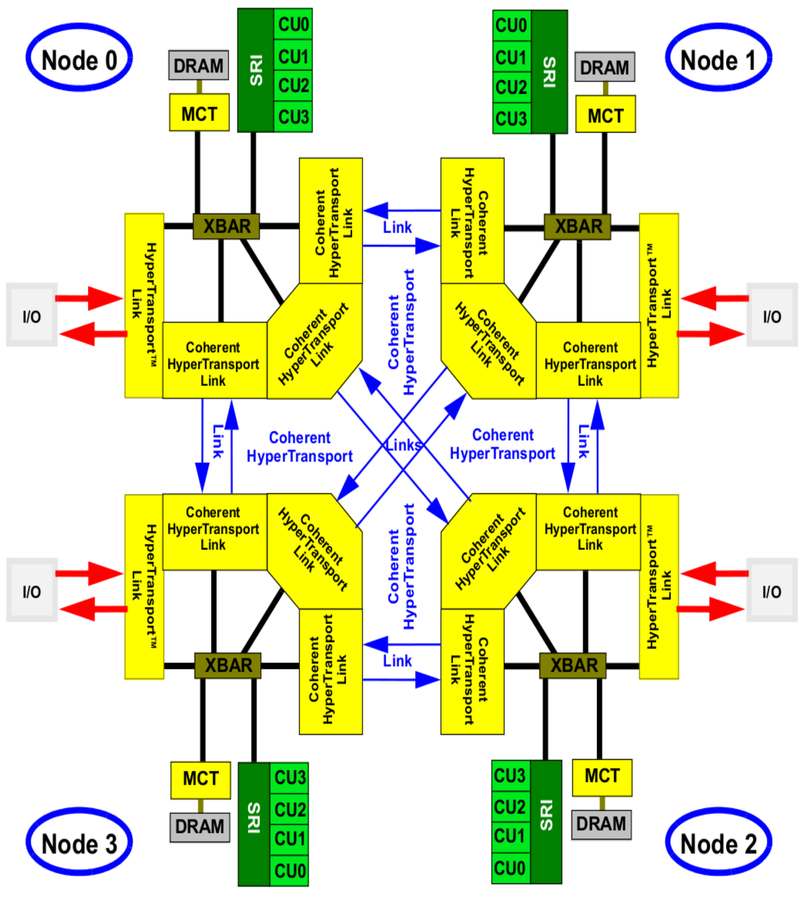
Frequently, a single tenant deploys a single application across many machines in a datacenter and uses the operating system primarily to efficiently use the hardware..

Typically a standard operating system is used to support the construction of event-driven software. These systems were developed in an environment where individual machines were multiplexed across many users and applications.

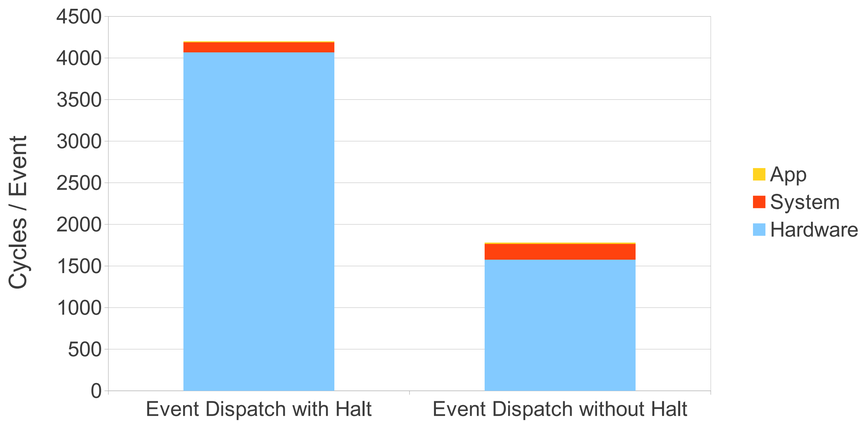
Is there sufficient justification to rethink the operating system for event-driven software in cloud environments?

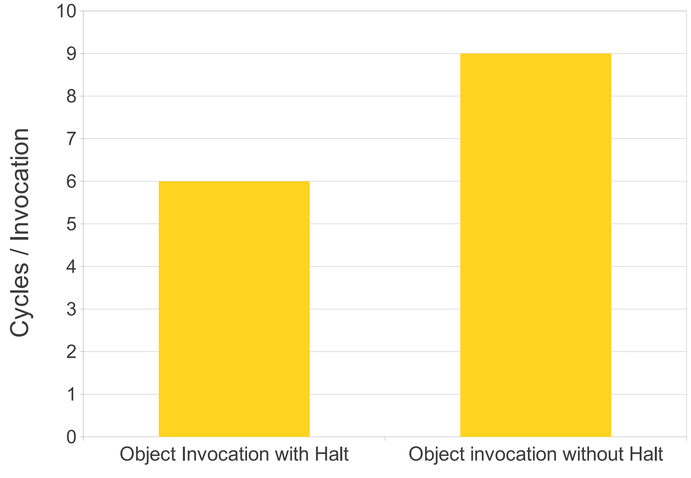
Event Cost Breakdown





Example: Halt versus Spin tradeoff






```
event_loop() {
  while(1) {
    if(flag) {
      flag = 0;
      send_ipi(self);
    }
    sti
    hlt or nop
    cli
  }
}
```

```
event_handler() {
  flag = 1;
}
```

Conclusions

If one wants to optimize the performance of an event driven application, it is necessary to customize for both the hardware characteristics as well as for the applications needs. Doing so in a modern operating system is difficult due to the strict kernel-user level boundary.

We believe that structuring a system for event driven software as a library OS enables the customization needed.