Composition in EbbRT

Dan Schatzberg, Boston University
Objective: Build more efficient software by constructing custom, application-specific operating systems.
Objective: Build more efficient software by constructing custom, application-specific operating systems.
How do we balance the desire to customize with the need to make development scale?
• Integration with general purpose systems for incremental development

• Event-driven, non-preemptive execution model maps software closely to hardware

• Reusable software components which developers can extend, replace, or discard to construct custom systems
• EbbRT is a toolkit for constructing library operating systems (single address space) for cloud applications

• Components in the small (memory allocators, timers) and in the large (distributed key-value stores, file systems)
Object Oriented Programming (C++, Java)

Procedural

Object

Data

Function

Function

Function

Send Message

Data

Method

Method

Method
Interoperation without shared memory (CORBA, protobufs, capnproto)

~5% of CPU time in Google datacenters is spent (de)serializing data [ISCA2015]
Elastic Building Blocks

Instance

VM0

VM1

VM2

Encapsulated, per-VM, per-core Representatives
EbbRef\langle T \rangle

Per-Core Indirection Table

C++ Object (type: T)
Intra-Ebb Communication is Encapsulated.
Free to use shared memory, TCP/IP, RDMA, etc.
Representatives are constructed on-demand

Invokes T::HandleFault() to construct a representative

EbbRef<T>

Per-Core Indirection Table

∅
• Ebbs as services vs data containers. E.g. should a network packet be an Ebb?

• C++ only - how crucial is this?

• Static dispatch (C++ templates) vs Dynamic Dispatch (virtual functions)
• How do we actually define interface semantics?
  • Types
  • Comments
  • Vague Implications
Start(std::chrono::microseconds timeout, bool repeat);
a_uniq.insert(t)  pair<iterator, bool>

Requires: If t is a non-const value expression, value_type shall be MoveInsertable into X; otherwise, value_type shall be CopyInsertable into X.

Effects: Inserts t if and only if there is no element in the container with key equivalent to the key of t. The bool component of the returned pair indicates whether the insertion takes place, and the iterator component points to the element with key equivalent to the key of t.

Average case \( O(1) \), worst case \( O(a_{uniq}\text{size}()) \).
Alloc → Free

SlabAllocator

PageAllocator

Fixed-size Allocator

Alloc → Free
malloc  free

Alloc  Free

GeneralPurpose Allocator

SlabAllocator  allocator  allocator  allocator

Fixed-size Allocator
Client-specified page fault handlers

PageAllocator

VMemAllocator

Alloc    Free

Virtual Memory

Identity Map

Physical Memory
movabs 0xffffffff00000010,%rax  # EbbRef<GPAllocator>
test   %rax,%rax
je     1870c0  # HandleFault
mov    0x8(%rax),%rdi  #Load SlabAllocator Ref
callq  19bd20 <ebbrt::SlabAllocator::Alloc()>
https://github.com/sesa/ebbrt
• Memory Allocators (Page, VMem, Slab, GeneralPurpose)

• Networking (Ethernet Driver, IPv4, UDP, DHCP, TCP)

• Filesystem (POSIX: read, write, open, rename, etc.)

• NodeAllocator (Boot (virtual) machine with a particular image, allocate logically isolated networks)

• Messenger (Send messages between Ebb representatives)
• Timer and EventManager (interrupt dispatcher)

• Distributed Key-Value Store (Put(key, value), Get(key))

• Application level data (e.g. Matrix, Image)