DMesos

-- Not only a re-implementation of Mesos

李枫
XianBei2011@gmail.com
Jun 21，2017

MesosCon ASIA
JUNE 20-22, 2017 | BEIJING, CHINA

Agenda

I. Background Information

- Container 2.0
- Cloud 2.0
- Growing Ecosystem of ARM

II. Why D

- Overview
- Compilers
- A potential candidate of system language
- A good fit for ARM
- Pros & Cons
- vibe.d
- Mir & DCompute

III. Arch & Design

- Design Goals
- Concurrency Model

- Distributed Consensus
- Storage
- Messaging/RPC
- Scheduling
- HPC
- Security
- Overall Architecture

IV. Current Implementation Status

- DLMDB
- DRaft
- DAkka

V. Mesos on ARM (since MesosCon Asia 2016)

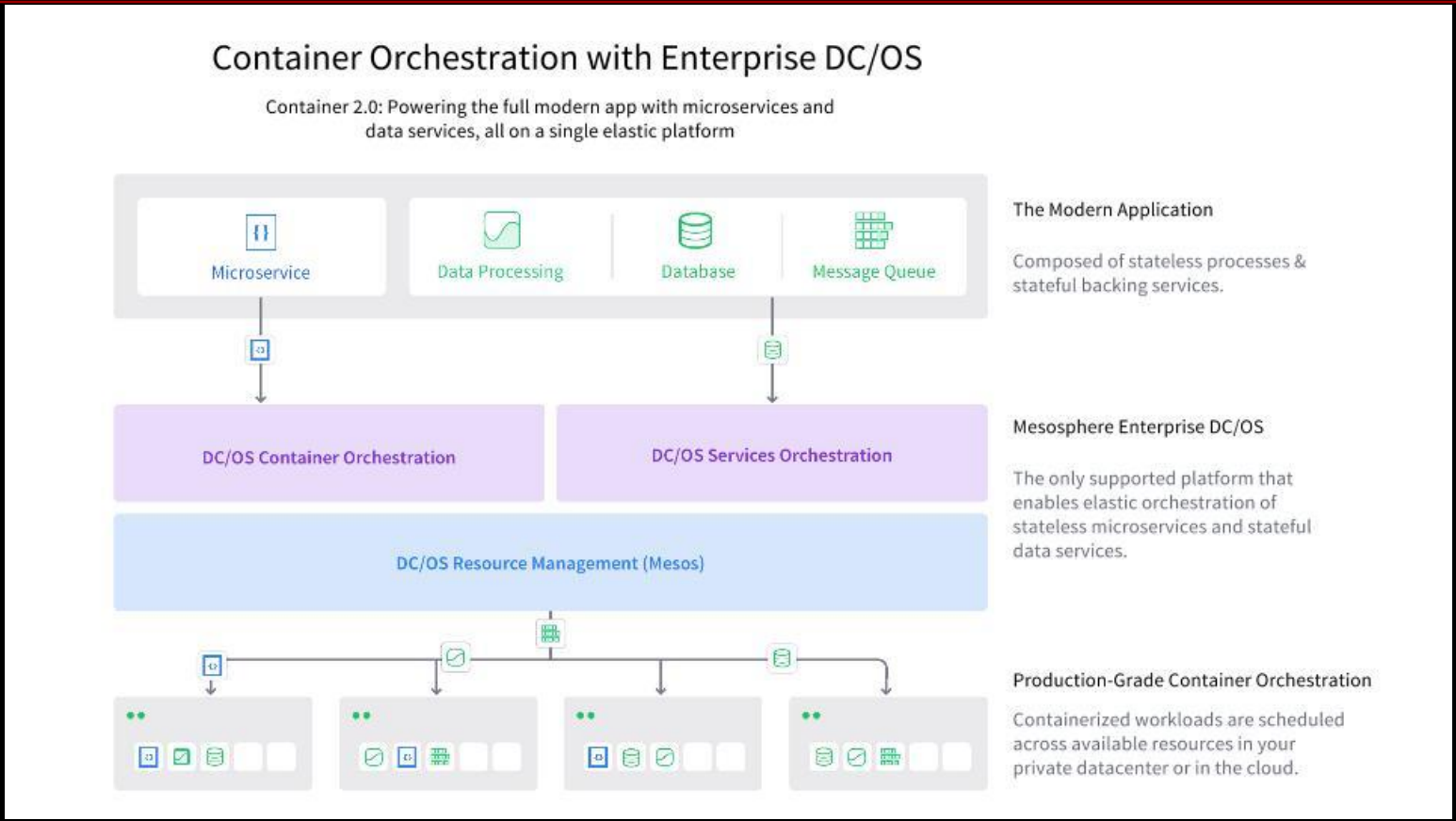
- LLVM 4.x
- Memory Optimization
- Mesos 1.3.0
- AARCH64 distros for RPi3

VI. Wrap-up

I. Background Information

1) Container 2.0

- https://mesosphere.com/blog/2016/08/01/container-2-0-dcos/



2) Cloud 2.0

- https://en.wikipedia.org/wiki/Cloud_computing
- http://www.computerworld.com/article/3074998/cloud-computing/google-says-welcome-to-the-cloud-20.html

--Diane Greene, SVP for Google's cloud businesses

A Focus on Data

"The 2.0 of the cloud is the data and understanding the data."

Artificial Intelligence

"Machine learning is changing the way companies use the cloud"

…

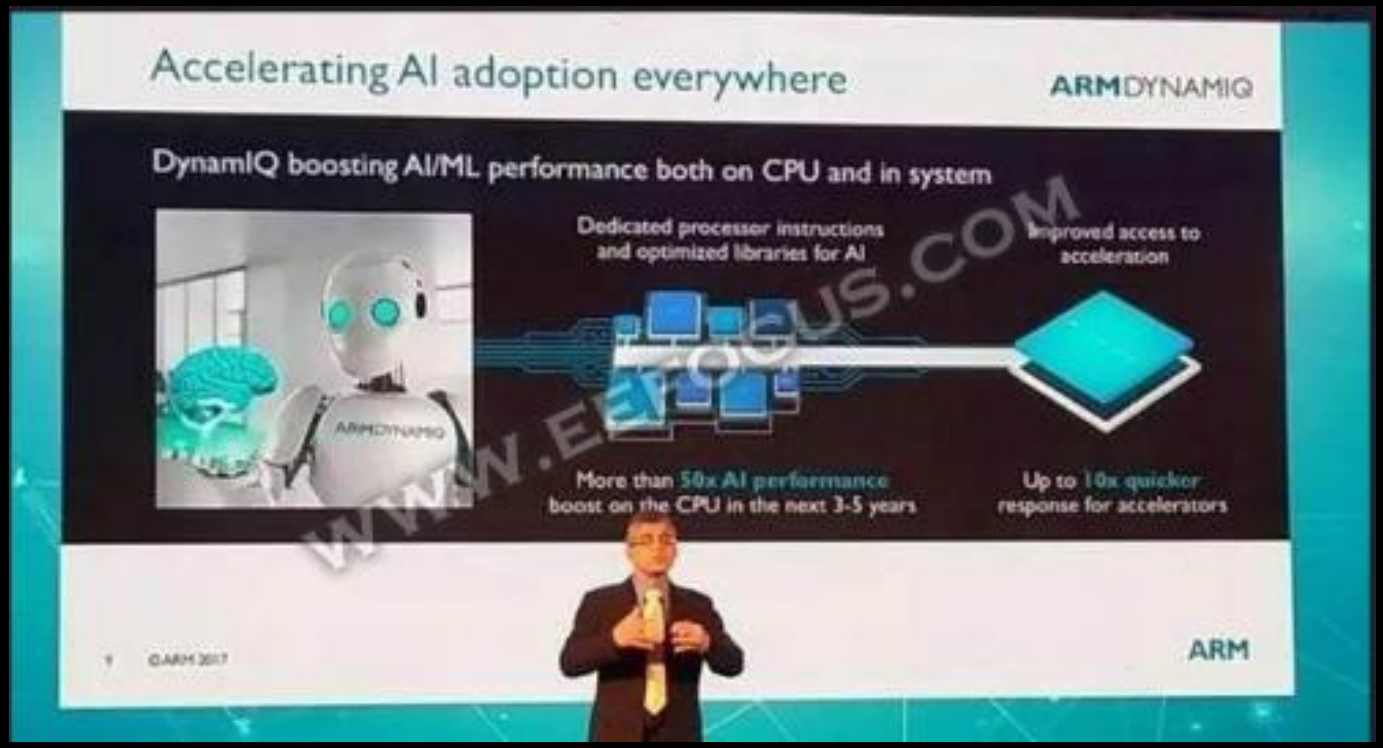
3) Growing Ecosystem of ARM

- https://en.wikipedia.org/wiki/ARM_architecture
- https://www.arm.com/

AI

DynamIQ

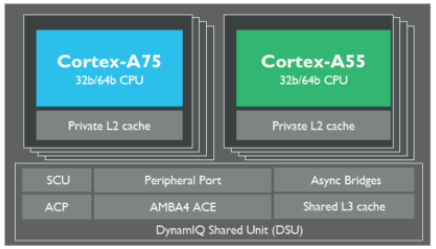
- http://pages.arm.com/dynamiq-technology.html

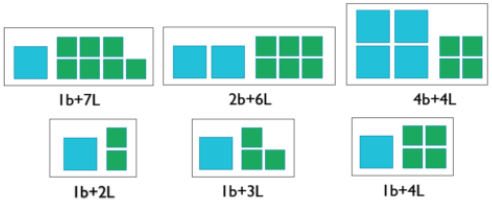


Cortex-A75

DynamIQ big.LITTLE

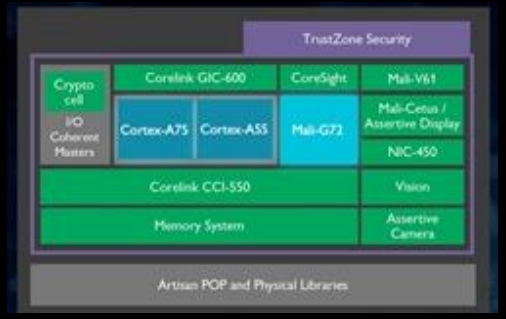
- New generation of big.LITTLE based systems built on DynamIQ technology
- DynamIQ big.LITTLE systems:
  - Expands into markets beyond mobile
  - Higher product differentiation
  - Improved energy efficiency
  - Increased user experience (UX)
- Software compatibility
  - DynamIQ big.LITTLE is supported by Energy Aware Scheduling (EAS)
  - Contains support for DynamIQ features





Example DynamIQ big.LITTLE configurations

ARM



Cloud

- **https://buildazure.com/2017/03/10/windows-server-running-on-arm-cpus-azure-is-next/**

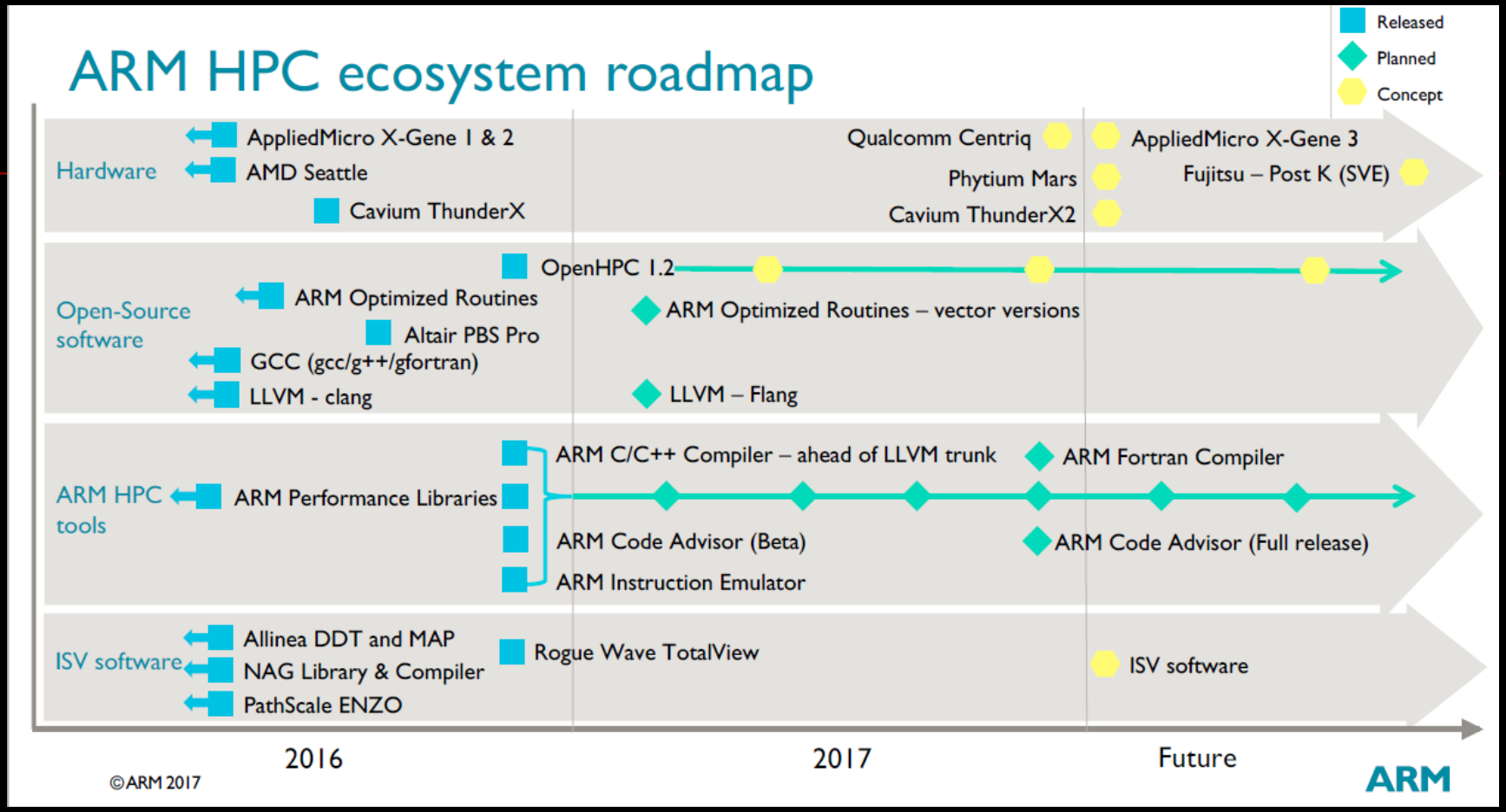
Qualcomm Centriq 2400 ARM Server CPU

packet.net

- **https://retout.co.uk/blog/2017/04/25/packet-net-arm64-servers**

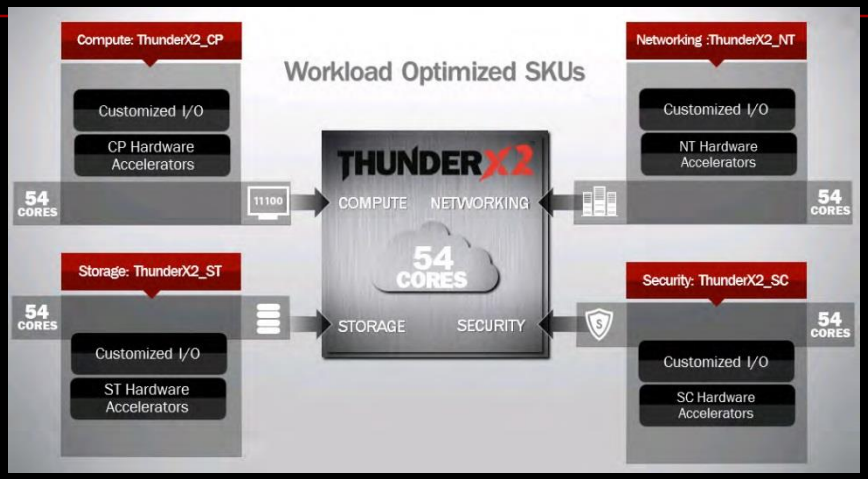
Packet.net offer an ARMv8 server with 96 cores for $0.50/hour.

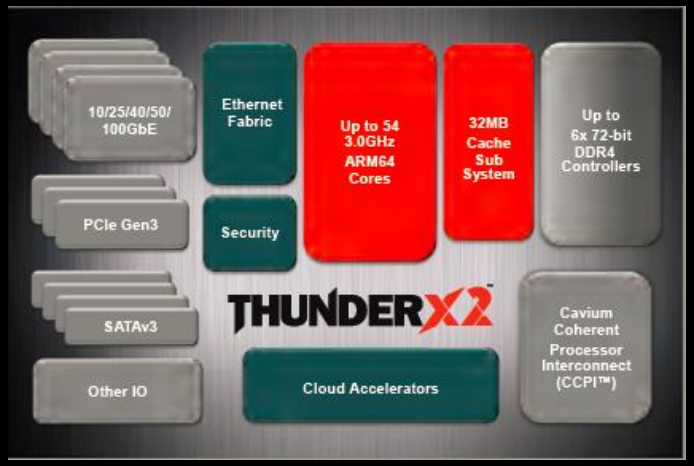
HPC


ARM HPC ecosystem roadmap
Released
Planned
Concept
Hardware
AppliedMicro X-Gene 1 & 2
AMD Seattle
Cavium ThunderX
Qualcomm Centriq
Phytium Mars
Cavium ThunderX2
AppliedMicro X-Gene 3
Fujitsu – Post K (SVE)
Open-Source software
OpenHPC 1.2
ARM Optimized Routines
ARM Optimized Routines – vector versions
Altair PBS Pro
GCC (gcc/g++/gfortran)
LLVM - clang
LLVM – Flang
ARM HPC tools
ARM Performance Libraries
ARM C/C++ Compiler – ahead of LLVM trunk
ARM Fortran Compiler
ARM Code Advisor (Beta)
ARM Code Advisor (Full release)
ARM Instruction Emulator
ISV software
Allinea DDT and MAP
NAG Library & Compiler
PathScale ENZO
Rogue Wave TotalView
ISV software
2016
2017
Future
©ARM 2017
ARM

The Machine

- https://www.theinquirer.net/inquirer/news/3010243/hpe-shows-off-arm-powered-the-machine-prototype-with-160tb-memory





Linaro

- http://www.linaro.org

96boards

- http://www.96boards.org/

II. Why

1) Overview

- https://en.wikipedia.org/wiki/D_(programming_language)

For other programming languages named D, see D (disambiguation) § Computing. For other uses, see D (disambiguation).

The **D programming language** is an object-oriented, imperative, multi-paradigm system programming language created by Walter Bright of Digital Mars and released in 2001. Bright was joined in the design and development effort in 2007 by Andrei Alexandrescu. Though it originated as a re-engineering of C++, D is a distinct language, having redesigned some core C++ features while also taking inspiration from other languages, notably Java, Python, Ruby, C#, and Eiffel.

D's design goals attempt to combine the performance and safety of compiled languages with the expressive power of modern dynamic languages. Idiomatic D code is commonly as fast as equivalent C++ code, while being shorter[citation needed] and memory-safe.[9]

Type inference, automatic memory management and syntactic sugar for common types allow faster development, while bounds checking, design by contract features and a concurrency-aware type system help reduce the occurrence of bugs.[10]

Paradigm	compiled, multi-paradigm: procedural, object-oriented, functional, generic, concurrent
Designed by	Walter Bright, Andrei Alexandrescu (since 2007)
Developer	Digital Mars, Andrei Alexandrescu (since 2007)
First appeared	8 December 2001; 15 years ago[1]
Stable release	2.074.0[2] / 10 April 2017; 3 days ago[3]
Typing discipline	strong, static, inferred
OS	Unix-like (FreeBSD, Linux etc.), Windows, macOS
License	DMD[4][5] and standard libraries: Boost; GDC: GPLv3+; LDC: GPLv2+, partially BSD[6]
Filename extensions	`.d`
Website	dlang.org
Major implementations	
DMD (reference implementation), GDC, LDC	
Influenced by	
C, C++, C#, Eiffel,[7] Java, Python	
Influenced	
MiniD, DScript, Vala, Qore, Swift,[8] Genie	

```
import std.stdio;

void main()
{
    writeln("Hello, world!");
}
```

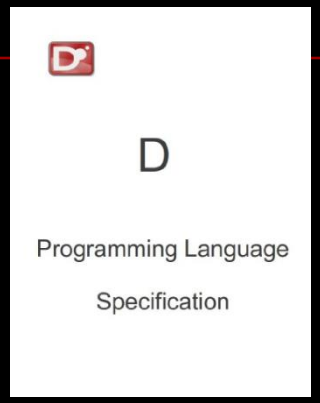
```
// Sort lines
import std.stdio, std.array, std.algorithm;

void main()
{
    stdin
        .byLineCopy
        .array
        .sort!((a, b) => a > b) // descending order
        .each!writeln;
}
```

- http://dlang.org (digitalmars.com/d)
- https://github.com/dlang

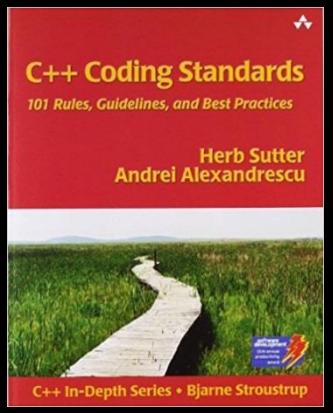
Designed by Experts

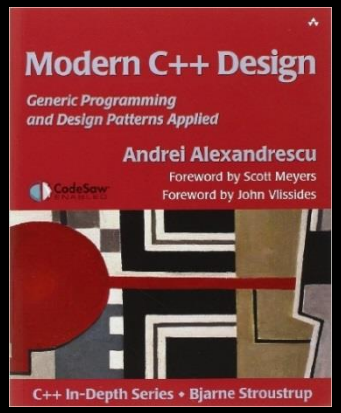
- **https://en.wikipedia.org/wiki/Walter_Bright**
  http://digitalmars.com/

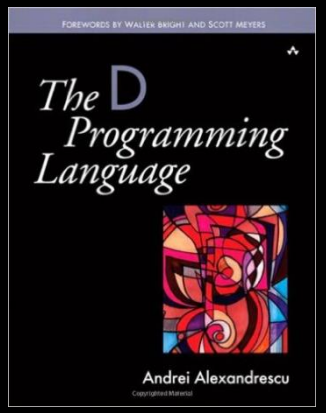
Digital Mars D compiler
Digital Mars C compiler
Digital Mars C++ compiler



- **https://en.wikipedia.org/wiki/Andrei_Alexandrescu**
  http://erdani.org/







Ranking

- **https://www.tiobe.com/tiobe-index/**

Position	Programming Language	Ratings
21	SAS	1.513%
22	D	1.413%
23	Dart	1.357%

- **http://redmonk.com/**

- https://medium.com/@hoffa/the-top-weekend-languages-according-to-githubs-code-6022ea2e33e8

The top weekend languages 2016:

Row	lang	ratio	weekday	weekend	sample_repo	sample_repo_2
1	rust	0.64	6268	3988	rust-lang/rust	matthiasbeyer/imag
2	glsl	0.63	4200	2663	d08ble/acpul-demo	Realm667/WolfenDoom
3	d	0.62	1129	696	nordlow/phobos-next	nordlow/justd
4	haskell	0.61	8351	5071	ghc/ghc	agda/agda
5	common lisp	0.6	1731	1032	ddmcdonald/sparser	roswell/roswell
6	kicad	0.59	1405	827	SchrodingersGat/kicad-library	esacinc/qrda
7	emacs lisp	0.57	13462	7694	tvraman/emacspeak	syl20bnr/spacemacs
8	lua	0.57	13940	7974	bthjonte/config	Mashape/kong
9	scheme	0.56	1545	861	mbakke/guix	justinethier/cyclone
10	julia	0.56	1755	989	JuliaLang/julia	JuliaLang/METADATA.jl
11	elm	0.55	1689	923	ravichugh/sketch-n-sketch	ianmackenzie/elm-opensolid-core
12	eagle	0.55	2521	1389	carpe-noctem-cassel/cnc-msl	DamonHD/OpenTRV
13	racket	0.55	1132	624	endobson/yaspl2	Javran/Thinking-dumps
14	dart	0.54	941	511	dart-lang/sdk	flutter/flutter
15	nsis	0.53	1159	613	KDE/emerge	greenshot/greenshot
16	clojure	0.53	6191	3269	uxbox/uxbox	kronkltd/jiksnu
17	kotlin	0.53	2836	1507	JetBrains/kotlin	dzharkov/kotlin
18	elixir	0.53	4967	2616	KronicDeth/intellij-elixir	elixir-lang/elixir
19	f#	0.52	1982	1025	FStarLang/FStar	fsprojects/Paket
20	ocaml	0.51	2043	1051	FStarLang/FStar	ocaml/opam-repository

Growing Ecosystem of 

- https://dlang.org/orgs-using-d.html
- http://wiki.dlang.org/Libraries_and_Frameworks
- https://wiki.dlang.org/IDEs

- http://code.dlang.org/
- https://wiki.dlang.org/Open_Source_Projects
- https://github.com/trending/d
- http://dconf.org

2) Compilers

- https://wiki.dlang.org/Compilers
- https://wiki.dlang.org/DMD
- https://wiki.dlang.org/GDC
- https://wiki.dlang.org/LDC

DMD

- Official reference compiler
- Latest D version
- Simple installation
- Very fast compilation speeds
- Architectures: i386, amd64

GDC

- GCC-based D compiler
- Strong optimization
- Great GDB support
- Architectures: i386, amd64, x32, armel, armhf, others

LDC

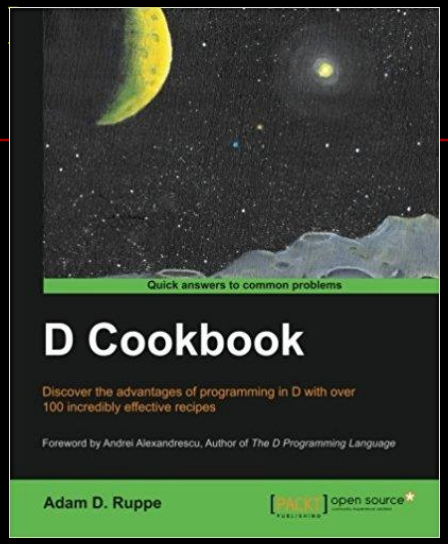
- LLVM-based D compiler
- Strong optimization
- Mobile support: iOS alpha, Android beta
- Architectures: i386, amd64, armel, armhf, others

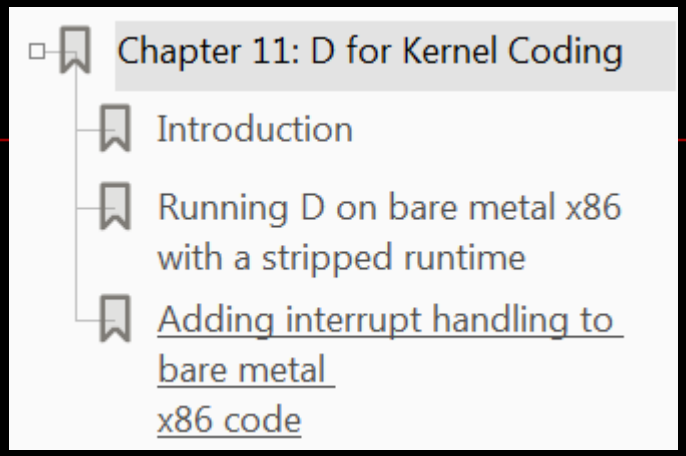
Trend

- D Language Front-End Proposed For GCC 8, ~800k Lines of Code
- How about Clang?

3) A potential candidate of system language

Bare Metal Programming

- 



- **https://gitlab.com/sarneaud/xanthe**
- **https://theartofmachinery.com/2017/02/28/bare_metal_d.html**
- **on bare metal ARM**

4) A good fit for ARM

- https://wiki.dlang.org/Compilers

GDC

- complete support armel, armhf
- partial or bare-metal only support aarch64

LDC

- complete support armel, armhf
- near-complete support aarch64

Ongoing development

- DCompute integration
- Latest LLVM support
- LLD integration
- JIT-compiled functions

for Android

- https://github.com/joakim-noah/android/releases

5) Pros & Cons

Pros

Features

- https://dlang.org/comparison.html
- https://wiki.dlang.org/Open_Source_Projects
- http://code.dlang.org/

Development Mode	community-driven
Productivity	a combination of C++/C/Java/Scala/Python… , auto/manually memory management
System Language	binary compatibility with C, pointer, inline assembler…
Interop	easily interface with legacy code in C/C++/Lua…
Programming Paradigms	including but not limited to imperative, object-oriented, metaprogramming, functional and concurrent (actor model)

Cons

- Still lack of popular frameworks/libraries
- Not as mature as commercial products, e.g. Memory Management

---

- Further optimization of runtime
- The still weak ecosystem when comparing with that of Java C++, Go…
- D still has a long way to go

6) vibe.d

- http://vibed.org/

Asynchronous I/O that doesn't get in your way, written in D

Fork me on GitHub

Productive

High-level declarative **REST** and **web application framework**

Full **HTTP(S)** stack with client, server and proxy implementations

Shipped with native database drivers for **MongoDB** and **Redis**

Complete **concurrency toolkit** and support for **low level I/O operations**

Fast

Asynchronous I/O for maximum speed and minimum memory usage

Compile-time "Diet" templates for unparalleled dynamic page speed

Compiled to **native machine code**

Multi-threading and integrated **load-balancing***

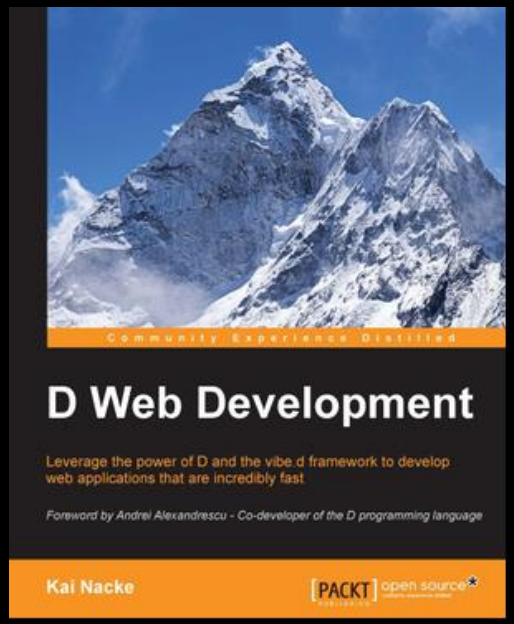
Simple

Fiber based blocking programming model for concise and intuitive development

Compact API with sensible default choices

Full support for **exception based** error handling

Simple access to third-party **extension libraries** using the DUB package system

- http://vibed.org/features
- 

7) Mir & DCompute

- https://github.com/libmir/
- https://github.com/libmir/dcompute

Separated Mir Projects

- **mir** -- Generic Numerical Library for Science and Machine Learning
- **dcv** -- Computer Vision Library for D
- **mir-algorithm** -- Multidimensional arrays (ndslice), Iterators, Algorithms
- **mir-glas** -- LLVM-accelerated Generic Linear Algebra Subprograms (GLAS)
- **mir-random** -- Professional Random Number Generators
- **dcompute** -- Native & Convenient Heterogeneous Computing for D
-

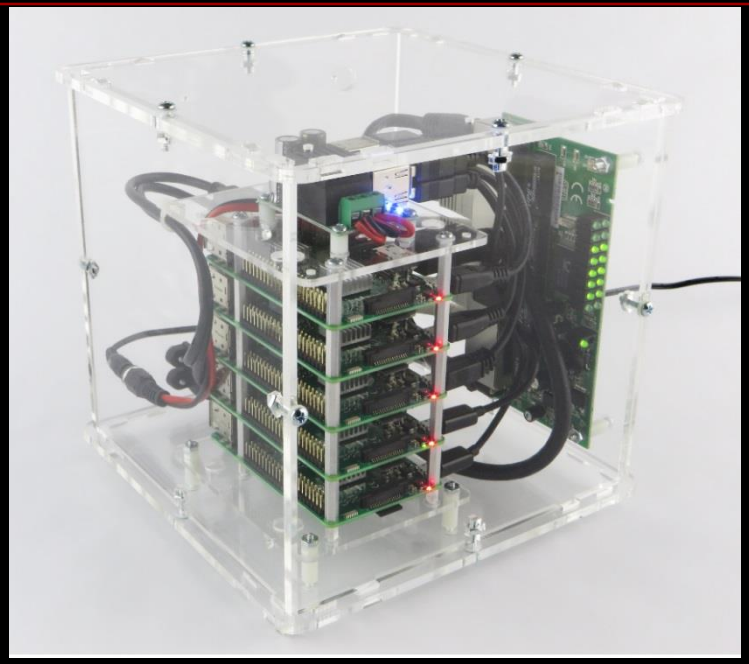
```
Future:
 - mir-runtime - lightweight always inlined `nothrow @nogc` Dlang Runtime
 - mir-neural - neural network library [WIP]
 - mir-fft - Fast and multidimensional FFT
 - mir-svm - support vector machines
```

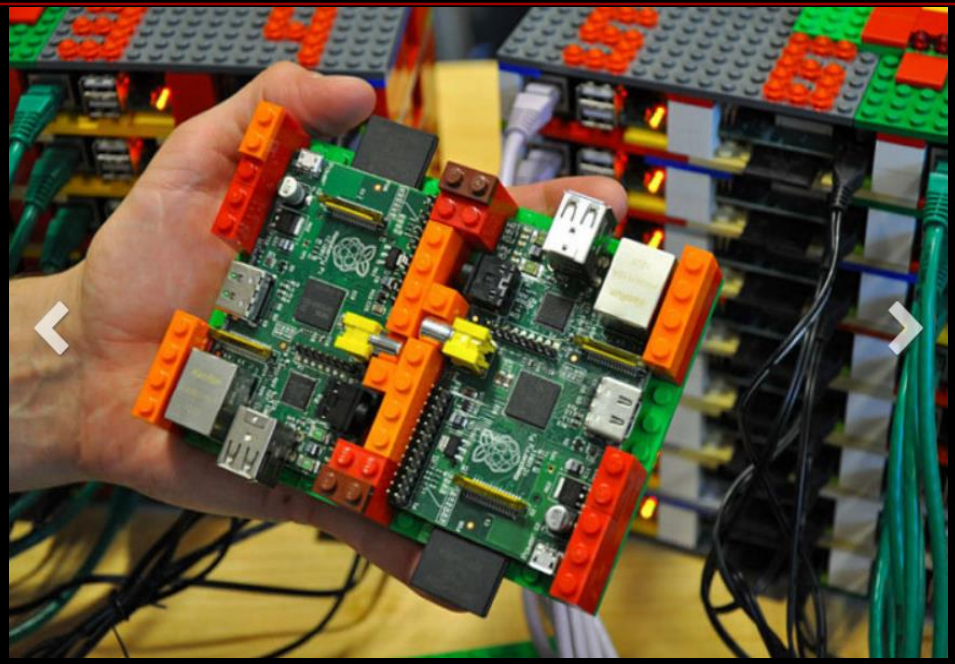
- …

III. Arch & Design

1) Design Goals

Infrastructure SW for & ARM based Micro Data Center

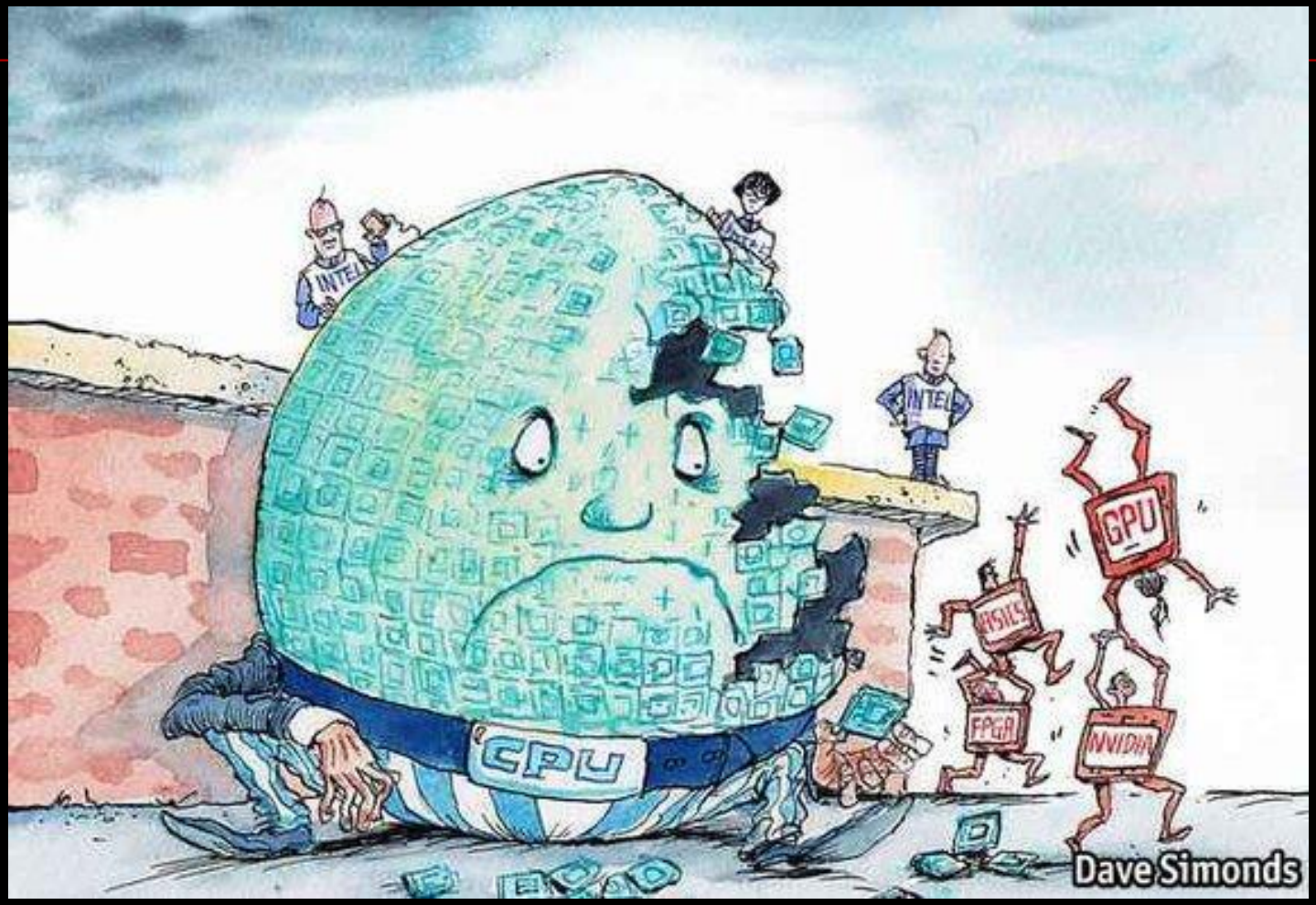




- Cheap ARM boards based cluster
- Just for Fun

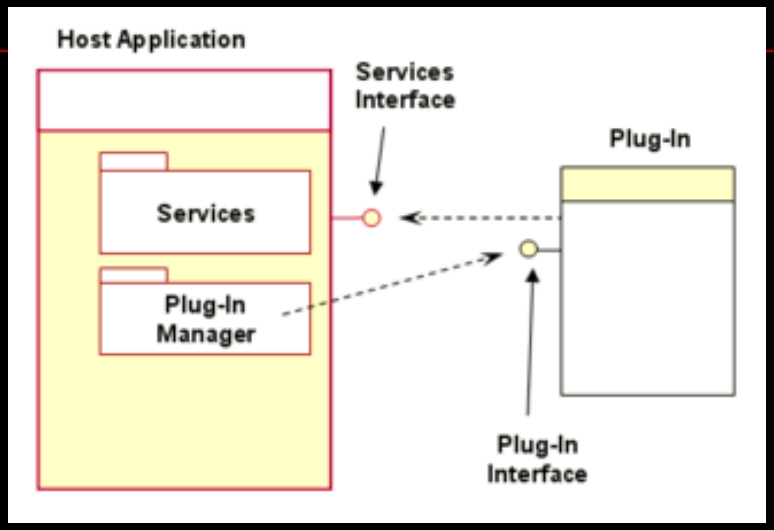
HPC-aware

- **High Performance Computing**
- **Heterogeneous Parallel Computing**



Modular Design

- https://en.wikipedia.org/wiki/Modular_design
- https://en.wikipedia.org/wiki/Plug-in_(computing)



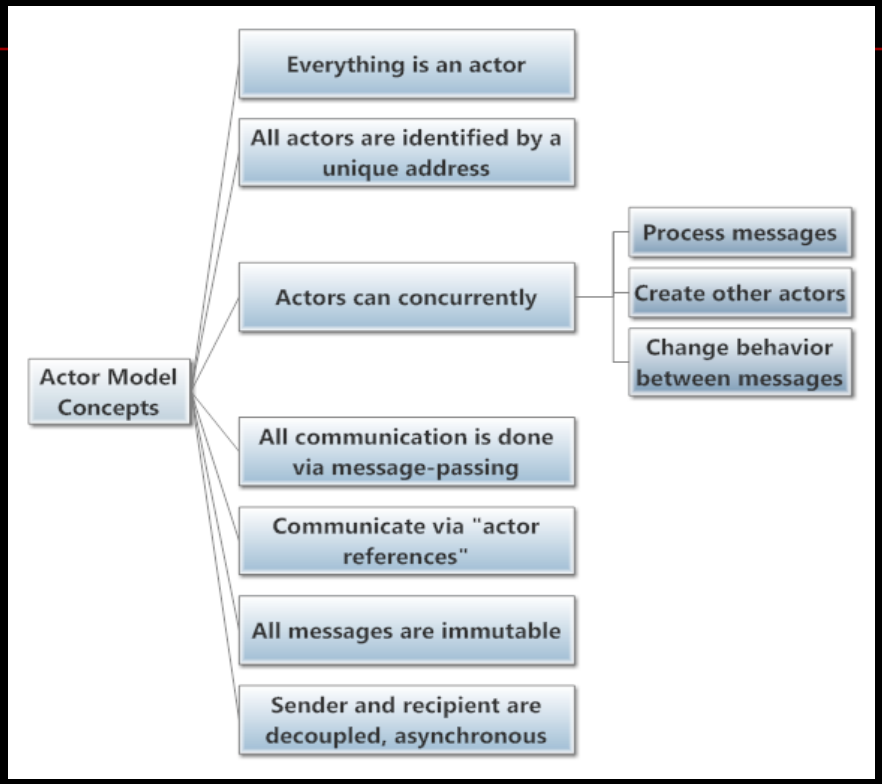
Next-Generation Dev System

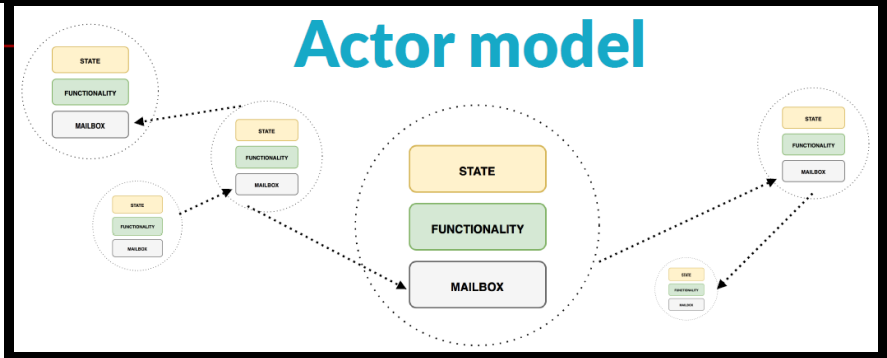
- Clang/LLVM based Toolchain
- Meson Build system
- In-device Dev
- …

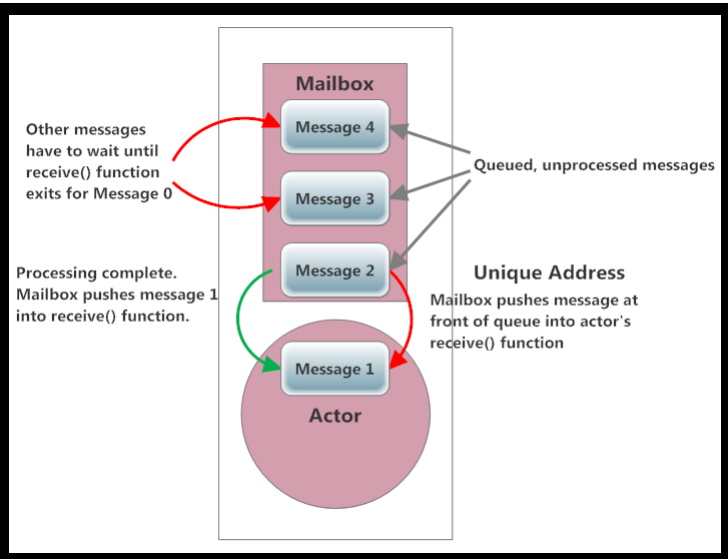
2) Concurrency Model

Actor Model

- https://en.wikipedia.org/wiki/Actor_model







Source: http://www.slideshare.net/

- Are Actors actually Nanoservices?

Coroutine/Fiber

- https://en.wikipedia.org/wiki/Coroutine
- https://en.wikipedia.org/wiki/Fiber_(computer_science)
- Coroutine landed in Clang/LLVM since May 2017

C++

- moved out from C++17☹
- Boost.Coroutine

D

- implements as standard library class **Fiber**
- http://dlang.org/phobos/core_thread.html#.Fiber

libprocess (Mesos)

Implementations	Native Execution	Garbage Collection	Pattern Matching	Copy-On-Write Messaging	Failure Propagation	Dynamic Behaviors	Compile-Time Type Checking	Run-Time Type Checking	Exchangeable Scheduler	Network Actors	GPU Actors	Backend
Erlang [13]	✗	✓	✓	✗	✓	✓	✗	✓	✓	✓	✗	BEAM
Elixir [70]	✗	✓	✓	✗	✓	✓	✗	✓	✗	✓	✗	BEAM
Akka/Scala [7]	✗	✓	✓	✗	✓	✓	✓	✓	✓	✓	✗	JVM
SALSA Lite [62]	✗	✓	✗	✗	✗	✗	✓	✓	✗	•†	✗	JVM
Actor Foundry [2]	✗	✓	✓	✗	✗	✓	✗	✓	✗	✓	✗	JVM
Pulsar [151]	✗	✓	✓	✗	✓	✓	✗	✓	✗	✓	✗	JVM
Pony [47]	✓	✓	✓	✗	✗	✗	✓	✓	✗	✓	✗	LLVM
Charm++ [109]	✓	•*	✗	✗	✗	✗	✓	✓	✗	✓	✗	C++
Theron [182]	✓	•*	✗	✗	✗	✓	✗	✓	✗	✓	✗	C++
libprocess [124]	✓	•*	✗	✗	✓	✗	✓	✗	✗	✓	✗	C++
CAF [44]	✓	•*	✓	✓	✓	✓	✓	✓	✓	✓	✓	C++

* Via reference counting, as opposed to tracing garbage collection.
† Only in SALSA, not SALSA Lite.

Table 2.2: Comparison of popular actor model implementations.

Source: https://www2.eecs.berkeley.edu/Pubs/TechRpts/2016/EECS-2016-55.pdf

CAF (C++ Actor Framework)

- http://actor-framework.org/

Akka

- **http://akka.io**
- Akka is a toolkit and runtime for building highly concurrent, distributed, and resilient message-driven applications for Java and Scala.
- **Build powerful reactive, concurrent & distributed applications more easily!**

DAkka (DMesos)

- **An imitator to Akka for D-based distributed and high concurrent computing framework**
- **But make fully use of D' s native support for Fiber**

3) Distributed Consensus

- https://en.wikipedia.org/wiki/Consensus_(computer_science)

Paxos

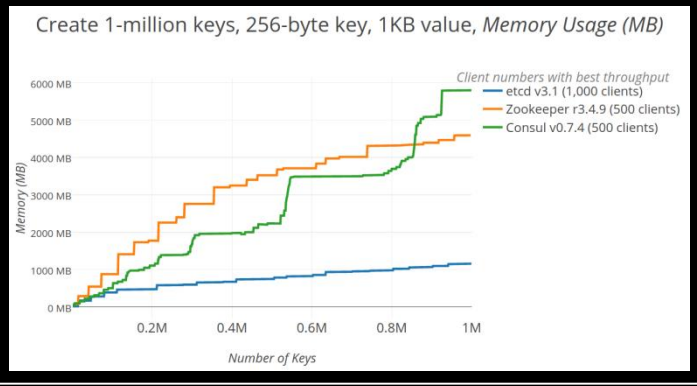
- https://en.wikipedia.org/wiki/Paxos_(computer_science)

ZooKeeper (Mesos)

- http://zookeeper.apache.org/

Raft (DMesos)

- https://raft.github.io/
- Designing for Understandability (Paxos Problems)
- https://coreos.com/blog/performance-of-etcd.html

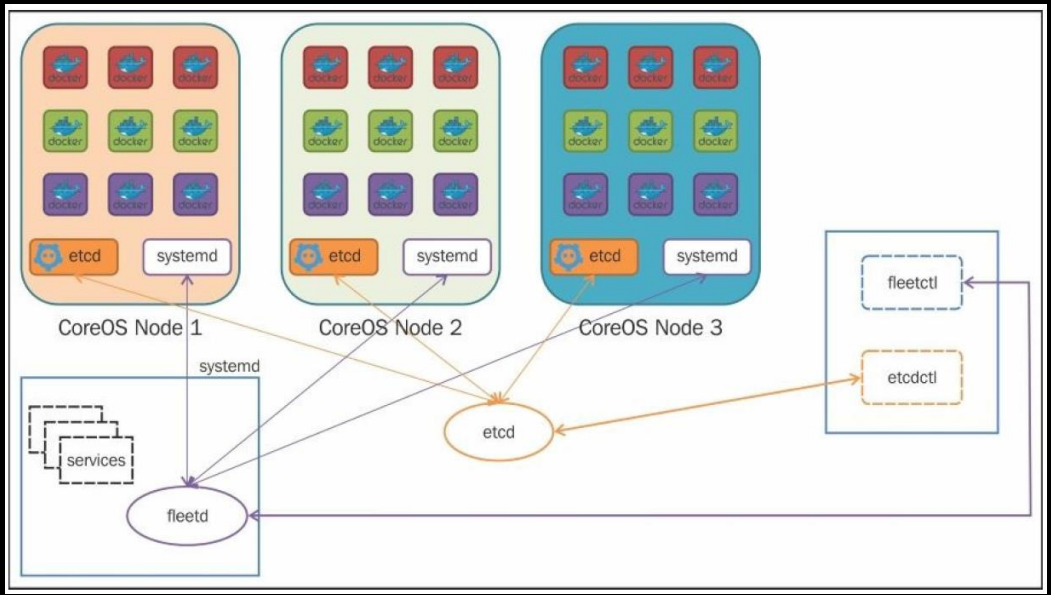


etcd

- https://coreos.com/etcd/
- https://coreos.com/blog/etcd3-a-new-etcd.html

A distributed key-value store:

- Base on Raft
- Distributed coordination primitives including distributed locks, elections, and software transactional memory
- Service & Security Mechanism
- Mainly Written in Go



DRaft

- An implementation of Raft consensus algorithm in D

DDTM

- An implementation of Distributed Transactional Memory in D

DDCF (DMesos)

- A D-based Distributed Computing Framework implements distributed coordination primitives that similar to **etcd**
- Based on **DAkka, DRaft, DDTM…**

4) Storage

- https://en.wikipedia.org/wiki/ACID
- https://en.wikipedia.org/wiki/Single-level_store
- https://en.wikipedia.org/wiki/Multiversion_concurrency_control
- https://en.wikipedia.org/wiki/Memory-mapped_file

LMDB

- https://en.wikipedia.org/wiki/Lightning_Memory-Mapped_Database
- https://symas.com/lightning-memory-mapped-database/

Features

- API inspired by Berkeley DB
- Fully-transactional, full ACID semantics with MVCC
- Key/Value store using B+tree
- Uses memory-mapped files, needs no tuning
- Uses Copy-on-Write and Single-Level Store
- Highly optimized, small code base(< 13k COL)

LevelDB (Mesos)

- http://leveldb.org/
- https://en.wikipedia.org/wiki/LevelDB

Features

- Based on concepts from Google's Bigtable database system
- Storage engine using LSM-tree
- Arbitrary byte arrays
- Compressed(Snappy) storage

DLMDB (DMesos)

- A rewriting of LMDB in D

Distributed Database

- https://en.wikipedia.org/wiki/Distributed_database

ActorDB

- http://www.actordb.com/

Features

- Distributed SQL database
- No single point of failure
- Horizontally scalable
- Consistent -- using Raft consensus algorithm
- On top of SQLite and LMDB
- MySQL and Thrift connectors
- Mainly written in **Elixir** (https://elixir-lang.org/)

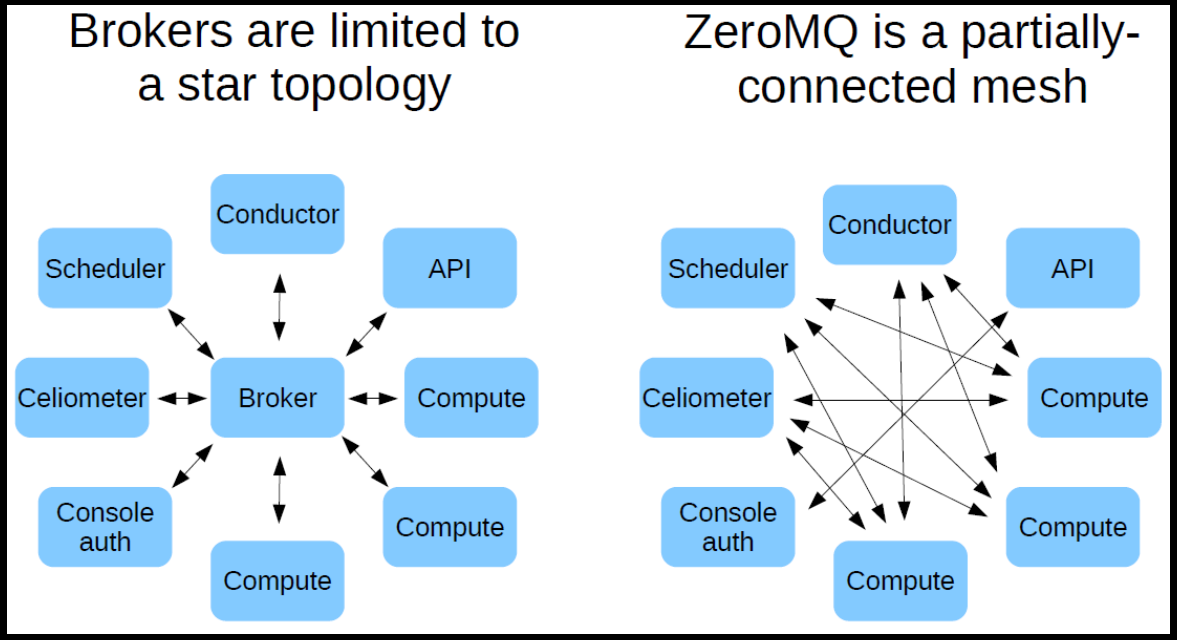
DActorDB (DMesos)

- A rewriting of ActorDB in D
- Distributed NewSQL database
- On top of DLMDB

5) Messaging/RPC

0MQ

- http://zeromq.org/
- https://en.wikipedia.org/wiki/ZeroMQ
- https://wiki.openstack.org/wiki/ZeroMQ



Source: https://www.openstack.org/assets/presentation-media/zmqslides2.pdf

- One goal for ØMQ is to get these "sockets on steroids" integrated into the Linux kernel itself

MessagePack

- http://msgpack.org/
- https://en.wikipedia.org/wiki/MessagePack

Mesos

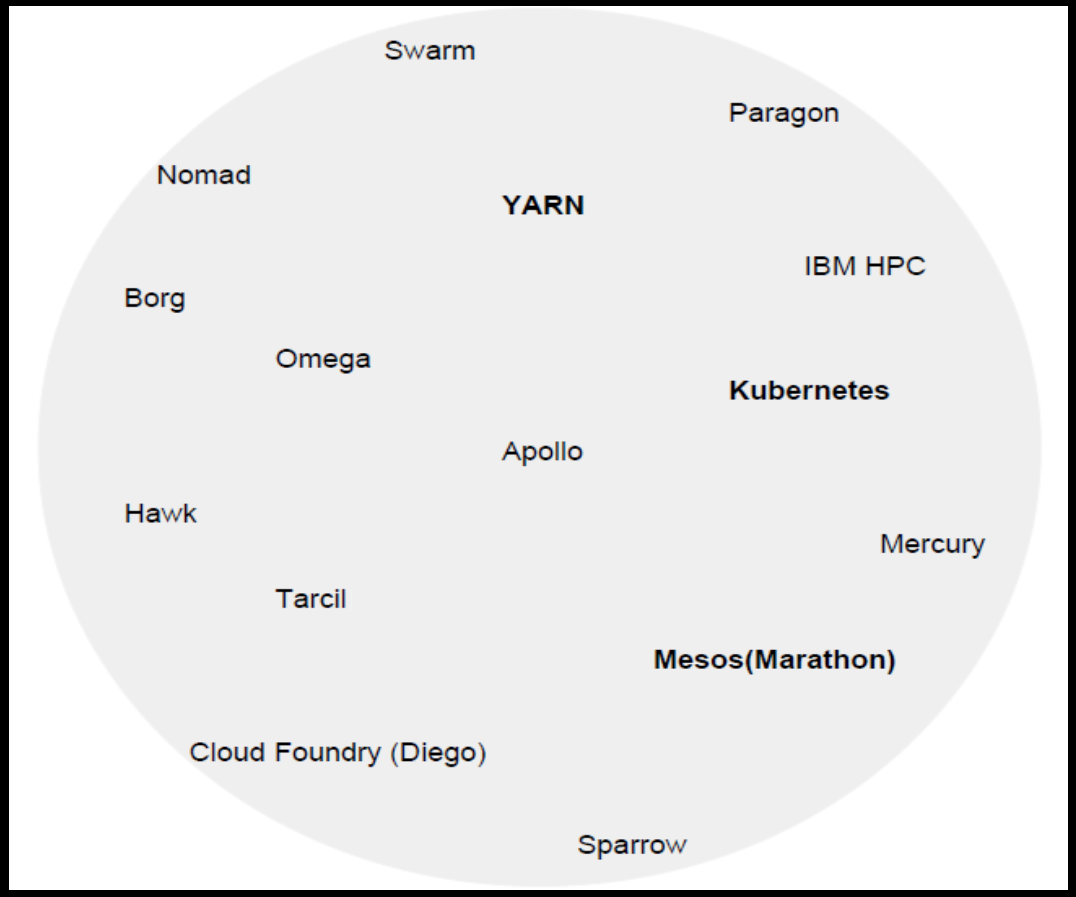
- http://msgpack.org/
- https://en.wikipedia.org/wiki/Protocol_Buffers

DRPC (DMesos)

- An imitator to 0RPC/gRPC based on 0MQ and MessagePack for distributed D apps

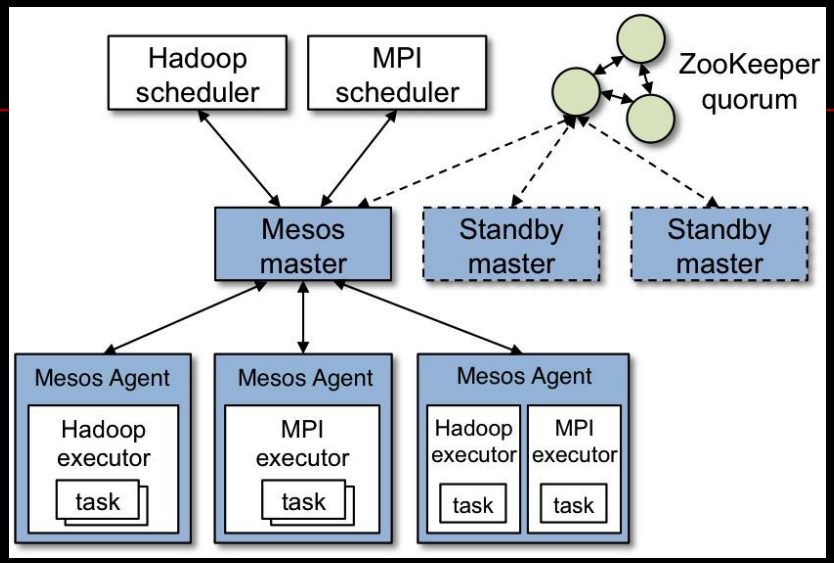
6) Scheduling

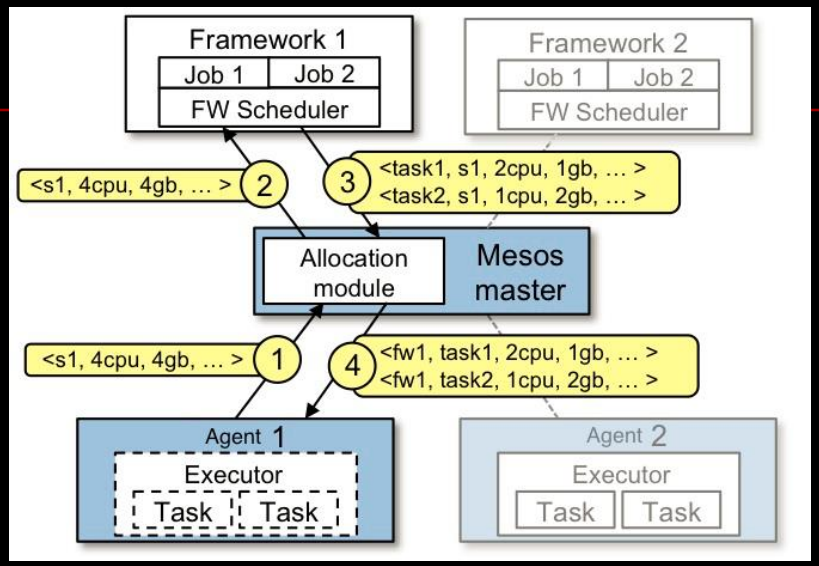
- https://en.wikipedia.org/wiki/Scheduling_(computing)
- https://en.wikipedia.org/wiki/Job_scheduler
- Cloud Resource Scheduling (CPU, Memory, Disk, Networking…)



Mesos

- **Two Level Scheduling**



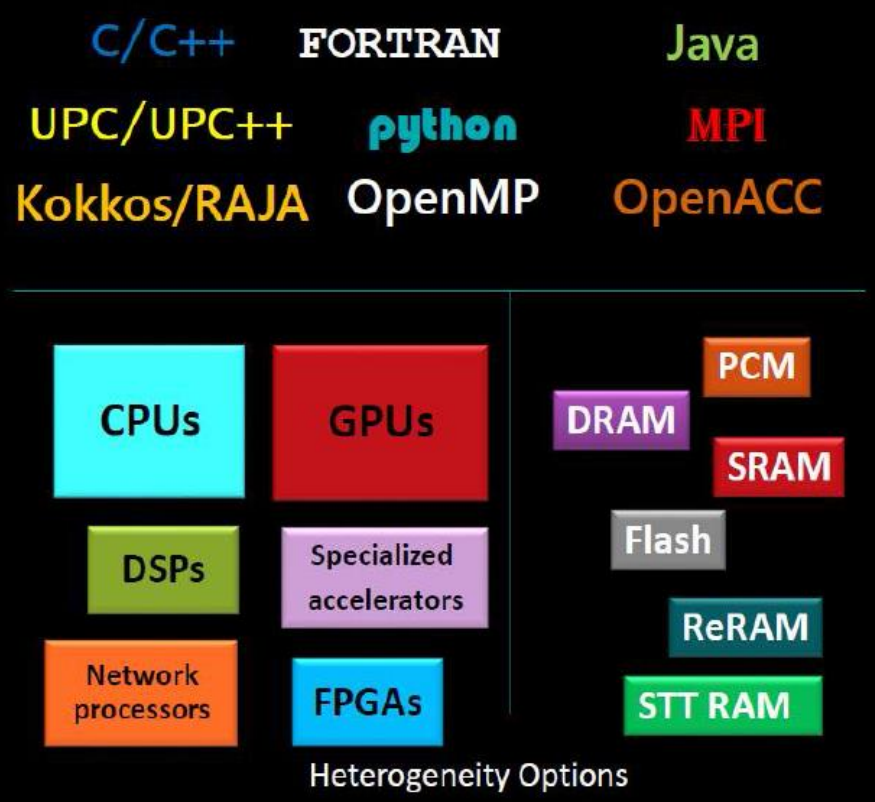


DMesos

- **Overall Design**
  - **User space/Kernel space Repartition & Unifying**
  - **Runtime Redesign**
  - **Without modifying the legacy Apps, or modifying them only slightly**

7) HPC

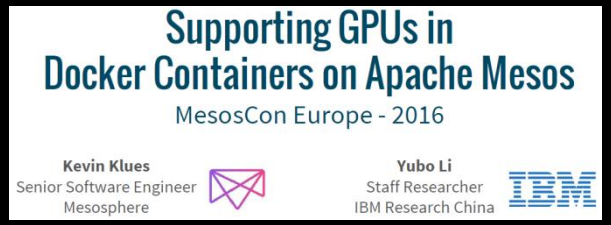
- High Performance Computing
- Heterogeneous Parallel Computing

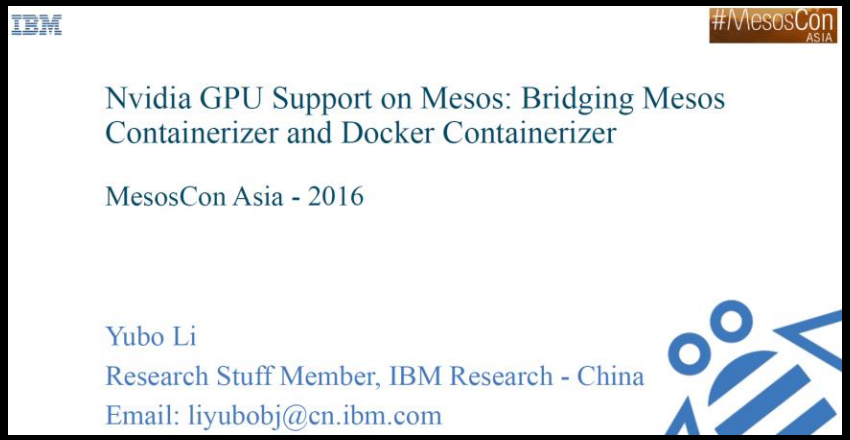


OpenCL

- https://en.wikipedia.org/wiki/OpenCL
- https://www.khronos.org/opencl/

GPU (Mesos)





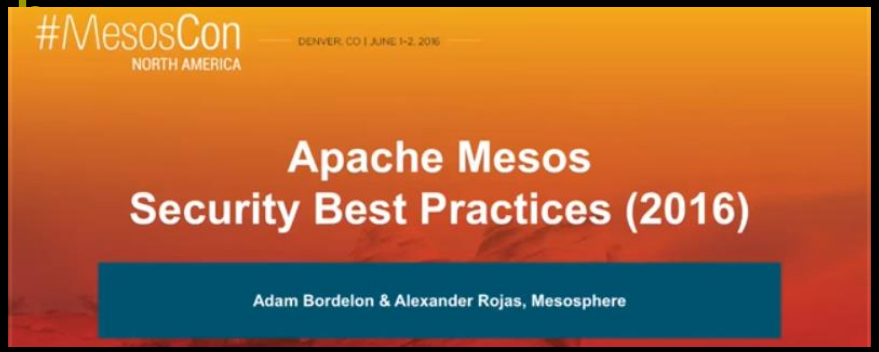
OpenCL Actors (DMesos)

- Inspired by CAF (https://github.com/actor-framework/actor-framework/wiki/OpenCL-Actors)
- Based on DAkka
- LDC/DCompute-assisted
- FPGA first

8) Security

- https://en.wikipedia.org/wiki/Cloud_computing_security

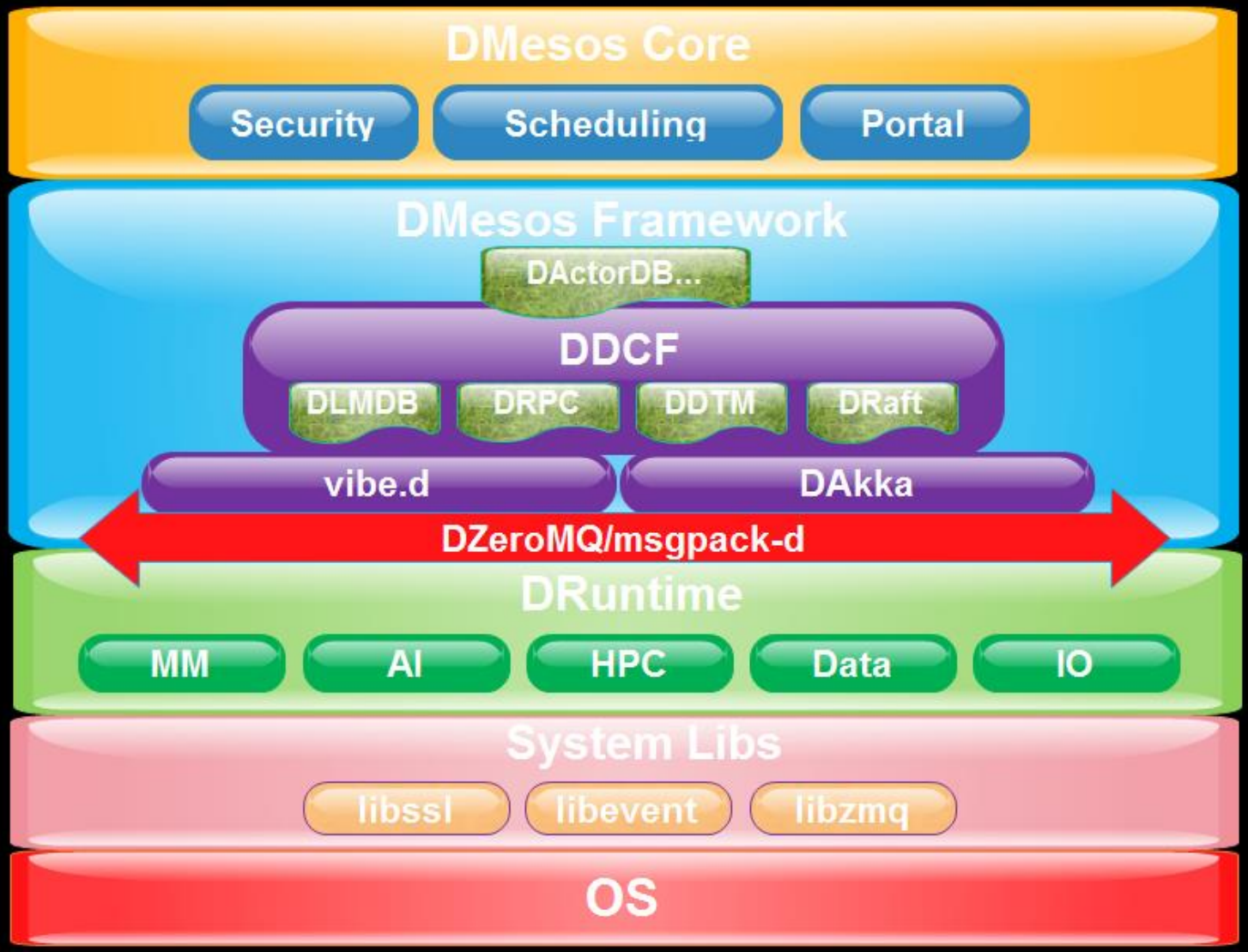
Mesos

- 

DMesos

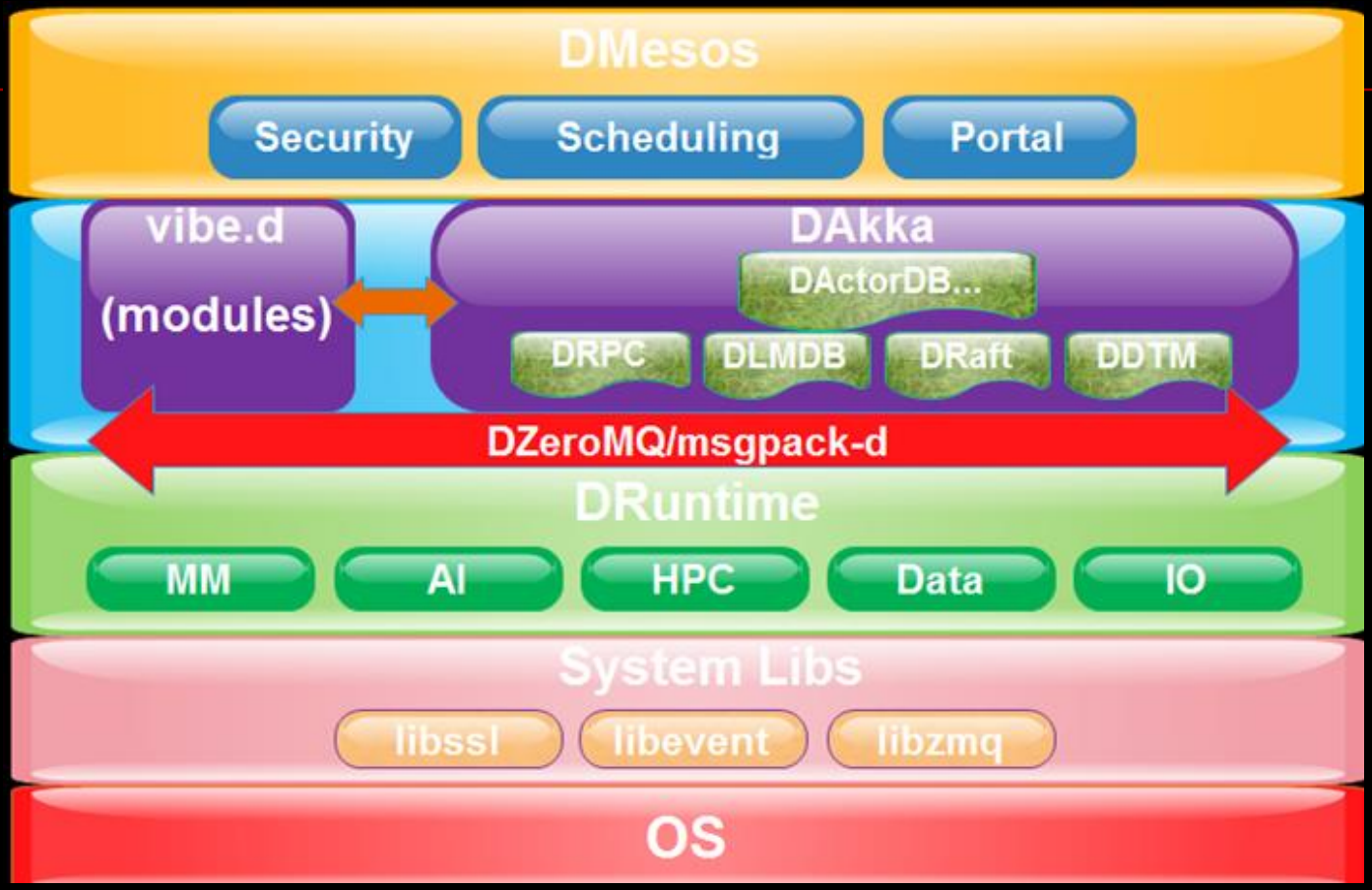
- Overall Design
  (together with Runtime and Scheduling)
- https://github.com/etcimon/botan
  Crypto Lib with HW-assisted support

9) Overall Architecture



Next

- gradually moving to the following arch when DAkka is mature enough



- More detailed arch will be available when the experiments for Kernel/Runtime/Scheduling redesign are ready

IV. Current Implementation Status

1) DLMDB

- Init release will be published on github in two months

2) DRaft

- https://github.com/apache/kudu //C++
- C++ to D :
  https://github.com/lhamot/CPP2D
  https://github.com/yebblies/magicport2
  https://github.com/jacob-carlborg/dstep
- incorporate the implementation of etcd and other projects

3) DAkka

- Main reference projects:
  https://petabridge.com/bootcamp/ //Akka.Net
  http://dotnet.github.io/orleans/
  https://github.com/orbit/orbit
  https://www.actor-framework.org/
  http://akka.io/
  https://www.pykka.org/
- A large code base

V. Mesos on ARM

- https://github.com/XianBeiTuoBaFeng2015/MesosOnARM/

1) LLVM 4.x

4.0.0

- improvements to ThinLTO (-flto=thin)
- experimental support for coroutines
- improvements to ARM/MIPS targets
- better GNU ld compatibility and significant performance improvements in LLD

Program	Output size	GNU ld	GNU gold [1]	LLD
ffmpeg dbg	91 MiB	1.59s	1.15s	0.78s
mysqld dbg	157 MiB	7.09s	2.49s	1.31s
clang dbg	1.45 GiB	86.76s	21.93s	8.38s
chromium dbg	1.52 GiB	142.30s [2]	40.86s	12.69s

- as well as improved optimizations, many bug fixes and more
- …

LLD

- Continuing to gain ground (enabling Multi-Threading, parallel ICF…)
- Some experimental to linking the Linux Kernel with LLD
- Ready for production on x86_64 ELF platforms
- But it is not production-ready for ARM☹

```
pi@raspberrypi:/usr/bin $ ll |grep ld
lrwxrwxrwx  1 root root          6 May 10 10:12 arm-linux-gnueabihf-ld -> ld.bfd*
-rwxr-xr-x  1 root root     511172 May 10 10:12 arm-linux-gnueabihf-ld.bfd*
-rwxr-xr-x  1 root root    4323180 May 10 10:12 arm-linux-gnueabihf-ld.gold*
-rwxr-xr-x  1 root root       6275 May 17 11:16 dpkg-buildflags*
-rwxr-xr-x  1 root root      26607 May 17 11:16 dpkg-buildpackage*
-rwxr-xr-x  1 root root       7503 May 17 11:16 dpkg-checkbuilddeps*
-rwxr-xr-x  1 root root      16371 May 17 11:16 dpkg-genbuildinfo*
-rwxr-xr-x  1 root root      30532 Feb 22 12:23 fold*
lrwxrwxrwx  1 root root          7 May 10 10:12 gold -> ld.gold*
-rwxr-xr-x  1 root root      22376 Apr  7 03:45 gtk-builder-tool*
lrwxrwxrwx  1 root root         72 Jun  4 15:40 ld -> /opt/MyWorkSpace/LLVM/clang-llvm-4.0.0-armv7a-linux-gnueabihf/bin/ld.lld*
lrwxrwxrwx  1 root root         26 May 10 10:12 ld.bfd -> arm-linux-gnueabihf-ld.bfd*
-rwxr-xr-x  1 root root       5308 May 28 17:29 ldd*
lrwxrwxrwx  1 root root         27 May 10 10:12 ld.gold -> arm-linux-gnueabihf-ld.gold*
-rwxr-xr-x  1 root root        125 Jun  1 15:09 perldoc*
-rwxr-xr-x  1 root root      14060 May 28 17:29 pldd*
lrwxrwxrwx  1 root root         26 Jan 25 14:47 pybuild -> ../share/dh-python/pybuild*
pi@raspberrypi:/usr/bin $
```

```
2017-06-03 14:53:10 /opt/MyWorkSpace/Mesos/mesos-master/build/3rdparty/protobuf-3.3.0/src/protobuf-3.3.0-build/src/.libs/lt-protoc: error
 while loading shared libraries: libprotobuf.so: cannot open shared object file: No such file or directory
2017-06-03 14:53:10 Makefile:8128: recipe for target 'unittest_proto_middleman' failed
2017-06-03 14:53:10 make[2]: *** [unittest_proto_middleman] Error 127
2017-06-03 14:53:10 make[2]: Leaving directory '/opt/MyWorkSpace/Mesos/mesos-master/build/3rdparty/protobuf-3.3.0/src/protobuf-3.3.0-buil
d/src'
2017-06-03 14:53:10 Makefile:1406: recipe for target 'all-recursive' failed
2017-06-03 14:53:10 make[1]: *** [all-recursive] Error 1
2017-06-03 14:53:10 make[1]: Leaving directory '/opt/MyWorkSpace/Mesos/mesos-master/build/3rdparty/protobuf-3.3.0/src/protobuf-3.3.0-buil
d'
2017-06-03 14:53:10 Makefile:1313: recipe for target 'all' failed
2017-06-03 14:53:10 make: *** [all] Error 2
2017-06-03 14:53:10 ninja: build stopped: subcommand failed.
Sat  3 Jun 14:53:10 UTC 2017
```

2) Memory Optimization

jemalloc

- http://jemalloc.net/
- https://github.com/jemalloc/jemalloc

- A state-of-the-art memory allocator
- Used in many famous projects like Android, Firefox, Redis, Oracle Servers…

```
#the way to enable jemalloc on RPi3
tar -xjvf jemalloc-5.0.0.tar.bz2;cd jemalloc-5.0.0
./configure --prefix=/usr/local/jemalloc
make -j4
sudo make install
```

```
pi@raspberrypi:/ $
pi@raspberrypi:/ $ cat /etc/ld.so.preload
/usr/local/jemalloc/lib/libjemalloc.so.2
/usr/lib/arm-linux-gnueabihf/libarmmem.so
pi@raspberrypi:/ $
```

3) Mesos 1.3.0

https://github.com/apache/mesos/archive/mesos-1.3.0.tar.gz (with cmake support)

http://www.apache.org/dist/mesos/1.3.0/mesos-1.3.0.tar.gz (std, no cmake support)

Patch

```
diff --git a/3rdparty/CMakeLists.txt b/3rdparty/CMakeLists.txt
index 96022ffaa..84d2b316e 100755
--- a/3rdparty/CMakeLists.txt
+++ b/3rdparty/CMakeLists.txt
@@ -389,6 +389,7 @@ if (NOT WIN32)
     INSTALL_COMMAND   cd ${ZOOKEEPER_C_ROOT} && make install
     URL               ${ZOOKEEPER_URL}
     URL_HASH          ${ZOOKEEPER_HASH}
+    BUILD_BYPRODUCTS  ${ZOOKEEPER_LIB}/lib/libzookeeper_mt.a
     )
 elseif (WIN32)
   ExternalProject_Add(
@@ -424,5 +425,6 @@ if (NOT WIN32)
     INSTALL_COMMAND   ${CMAKE_NOOP}
     URL               ${LEVELDB_URL}
     URL_HASH          ${LEVELDB_HASH}
+    BUILD_BYPRODUCTS  ${LEVELDB_ROOT}/out-static/libleveldb.a
     )
 endif (NOT WIN32)
diff --git a/cmake/CompilationConfigure.cmake b/cmake/CompilationConfigure.cmake
index a0032a35d..8755e24fc 100644
--- a/cmake/CompilationConfigure.cmake
+++ b/cmake/CompilationConfigure.cmake
@@ -143,7 +143,7 @@ endif (ENABLE_SSL AND (NOT ENABLE_LIBEVENT))
 # SYSTEM CHECKS.
 ################
 # Check that we are targeting a 64-bit architecture.
-if (NOT (CMAKE_SIZEOF_VOID_P EQUAL 8))
+if (NOT (CMAKE_SIZEOF_VOID_P EQUAL 4))
   message(
     FATAL_ERROR
     "Mesos requires that we compile to a 64-bit target. Following are some "
@@ -154,7 +154,7 @@ if (NOT (CMAKE_SIZEOF_VOID_P EQUAL 8))
     "    `cmake -G \"Visual Studio 15 2017 Win64\"`.\n"
     "  * OS X: add `x86_64` to the `CMAKE_OSX_ARCHITECTURES`:\n"
     "    `cmake -DCMAKE_OSX_ARCHITECTURES=x86_64`.\n")
-endif (NOT (CMAKE_SIZEOF_VOID_P EQUAL 8))
+endif (NOT (CMAKE_SIZEOF_VOID_P EQUAL 4))

 # Make sure C++ 11 features we need are supported.
 # This is split into two cases: Windows and "other platforms".
```

Build Mesos 1.3.0 by cmake/ninja on RPi3 (Raspbian)

Result: close to success!

Successfully build Mesos 1.3.0 on single RPi3 board with ninja + LLVM4.0.0 + ld.gold + jemalloc5.0.0!
(~5 hours, ninja –j4 ~3.25 hours)

Run test suite failed: ninja check
(same behavior on X64)

```
2017-06-11 04:25:26 [251/251] cd /opt/MyWorkSpace/Mesos/mesos-1.3.0/build && stout-tests && libprocess-tests && mesos-tests
2017-06-11 04:25:26 FAILED: CMakeFiles/check
2017-06-11 04:25:26 cd /opt/MyWorkSpace/Mesos/mesos-1.3.0/build && stout-tests && libprocess-tests && mesos-tests
2017-06-11 04:25:26 /bin/sh: 1: stout-tests: not found
2017-06-11 04:25:26 ninja: build stopped: subcommand failed.
```

(~12.5 hours)
This issue could be resolved soon…

4) AARCH64 distros for RPi3

HypriotOS-RPi64

- https://blog.hypriot.com/post/building-a-64bit-docker-os-for-rpi3/
- https://github.com/dieterreuter/workshop-raspberrypi-64bit-os

```
HypriotOS/arm64: pirate@black-pearl in ~
$ uname -a
Linux black-pearl 4.9.13-bee42-v8 #1 SMP PREEMPT Fri Mar 3 16:42:37 UTC 2017 aarch64 GNU/Linux
HypriotOS/arm64: pirate@black-pearl in ~
$
HypriotOS/arm64: pirate@black-pearl in ~
$ cat /proc/cpuinfo
processor       : 0
BogoMIPS        : 38.40
Features        : fp asimd evtstrm crc32
CPU implementer : 0x41
CPU architecture: 8
CPU variant     : 0x0
CPU part        : 0xd03
CPU revision    : 4
```

```
HypriotOS/arm64: pirate@black-pearl in ~
$ free -m
             total       used       free     shared    buffers     cached
Mem:           963        144        818          6          6         73
-/+ buffers/cache:         65        898
Swap:            0          0          0
```

- Debian 9 “Stretch” released on Jun 17

Alpine Linux

- https://alpinelinux.org/
- a security-oriented, lightweight Linux distribution based on musl libc
- do not officially support RPi3
- some Mesos dependency packages still need to be ported to Alpine

VI. Wrap-Up

- "Competitive Advantage with D"
  http://cppnow.org/2017-conference/announcements/2017/04/09/d-keynote.html
  http://ddili.org/AliCehreli_CppNow_2017_Competitive_Advantage_with_D.no_pause.pdf

- **Simple is always better**

  C++ → D

  autotools/make → cmake/ninja → meson/ninja

  GCC → LLVM

  glibc → musl + jemalloc

  Paxos → Raft

  X86 → ARM (Hardware Is The New Software?)

  …

- **My github:**

  https://github.com/XianBeiTuoBaFeng2015/

Q & A

THANK YOU!

Reference

Slides/materials from many and varied sources:

- http://en.wikipedia.org/wiki/
- http://www.slideshare.net/
- http://mesos.apache.org/
- https://github.com/apache/mesos/
- http://llvm.org
- http://zguide.zeromq.org/page:all/
- https://www.freebsd.org/
- https://cmake.org/
- http://mesonbuild.com/
- https://www.musl-libc.org/
- …