MesosCon 2016 - Alex Rukletsov

Demystifying the hype around modern cluster managers

(to say nothing about containers)

MESOSPHERE

Alex Rukletsov

Distributed Systems Engineer

Alex Rukletsov is an Apache committer and Mesos PMC member at Mesosphere. Prior to that Alex was segmenting medical images and investigating behaviour of human vessels in several German research institutes. His areas of interests include distributed systems, object recognition, probabilistic and heuristic algorithms.

Clarke's Third Law:
Any sufficiently advanced technology is indistinguishable from magic.

(Arthur C. Clarke)

Goal of this talk

Reflect on the current **technology**

and

reasons why this particular technology is emerging

Contents

(1) The Two Minutes ~~Hate~~ Philosophy

(2) Why containers? as seen by a Mesos fella

(3) Why microservices?

(4) What about Apache Mesos?

(5) What's ahead? free prophecies from that Mesos fella

The Two Minutes ~~Hate~~ Philosophy

“Right” abstractions

In software engineering and computer science, abstraction is a technique for arranging complexity of computer systems. It works by establishing a level of complexity on which a person interacts with the system, suppressing the more complex details below the current level.

— Wikipedia

“Right” abstractions

The essence of abstractions is preserving information that is relevant in a given context, and forgetting information that is irrelevant in that context.

— John V. Guttag

Secret of success

"Right" abstractions + succinct DSL

↓

Level of indirection

↓

Efficiency through automation

Our daily job is...

… abstract away tedious concepts and hence give operators a new — better! — way to express their intent, making them more efficient and allow for more automation.

Examples

+375 17 284 97 28

Examples

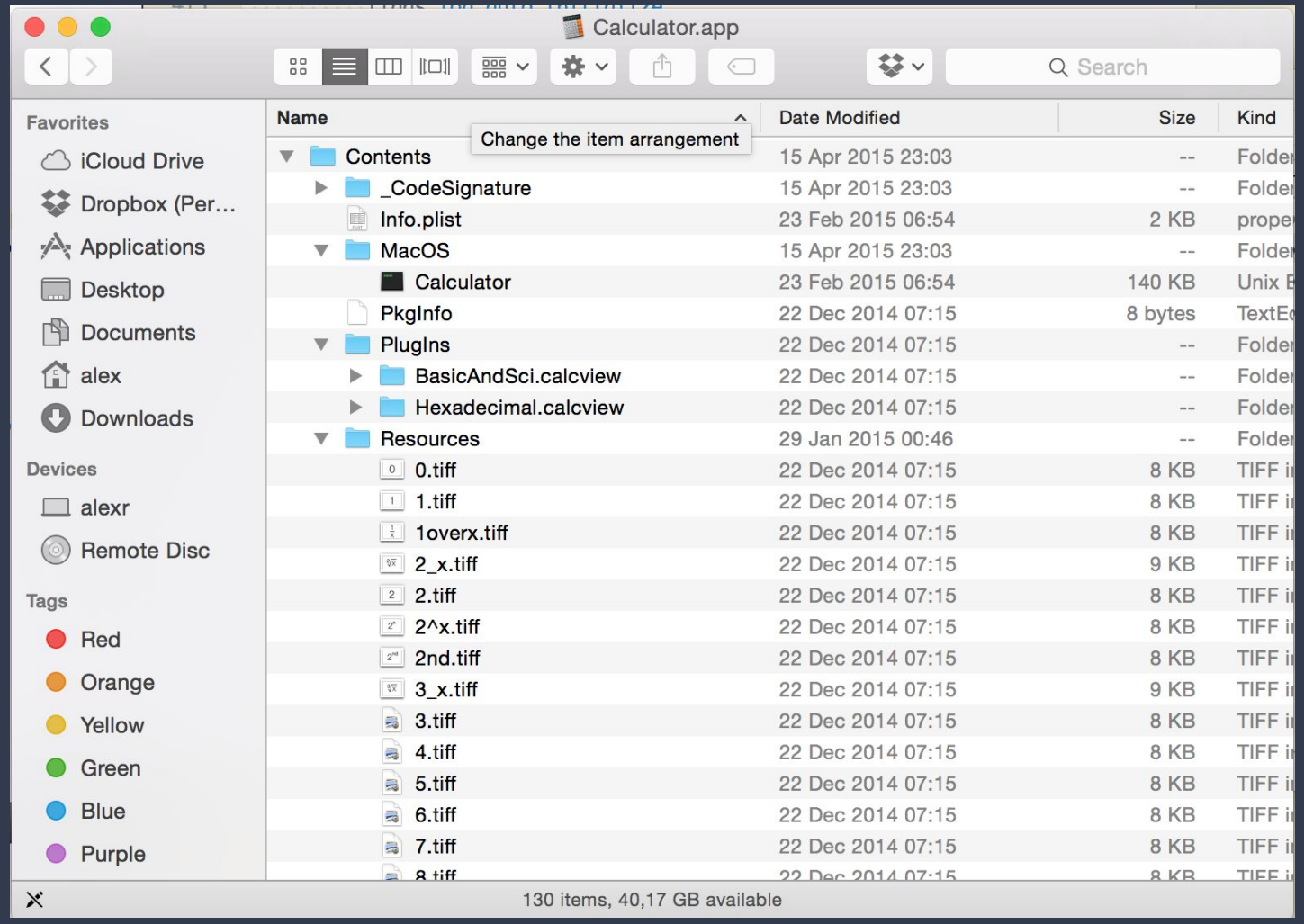
```
int socket = ::socket(AF_INET, SOCK_STREAM, 0);
::connect(socket, reinterpret_cast<sockaddr*>(&to), sizeof(to));
```

Why containers?

Isolation

```
I0627 20:04:34.257577 249618432 authenticator.cpp:519] Initializing server SASL
PC: @        0x1073b0197 std::__1::__hash_table<>::find<>()
*** SIGSEGV (@0xffffffffffffffff) received by PID 55929 (TID 0x10ec02000) stack trace: ***
    @     0x7fff881f6f1a _sigtramp
    @                0x2 (unknown)
    @        0x1073afe38 google::protobuf::FindPtrOrNull<>()
    @        0x1073b6370 google::protobuf::FileDescriptorTables::FindEnumValueByNumber()
    @        0x107369aaf google::protobuf::EnumDescriptor::FindValueByNumber()
    @        0x10746d39b google::protobuf::internal::NameOfEnum()
    @        0x106dbbdab mesos::internal::log::Metadata_Status_Name()
    @        0x106dbbd83 mesos::internal::log::Metadata::Status_Name()
    @        0x106cc8f45 mesos::internal::log::operator<<()
    @        0x106cc4516 mesos::internal::log::RecoverProcess::recover()
    @        0x106cf41a7 _ZZN7process8dispatchI7NothingN5mesos8internal3log14RecoverProcess
    @        0x106cf3f6f _ZNSt3__110__function6__funcIZN7process8dispatchI7NothingN5mesos8i
lProcessBaseEE_NS_9allocatorISQ_EEFvSP_EEclEOSP_
    @        0x10714f85b std::__1::function<>::operator()()
```

Packaging



Why microservices?

The Unix philosophy

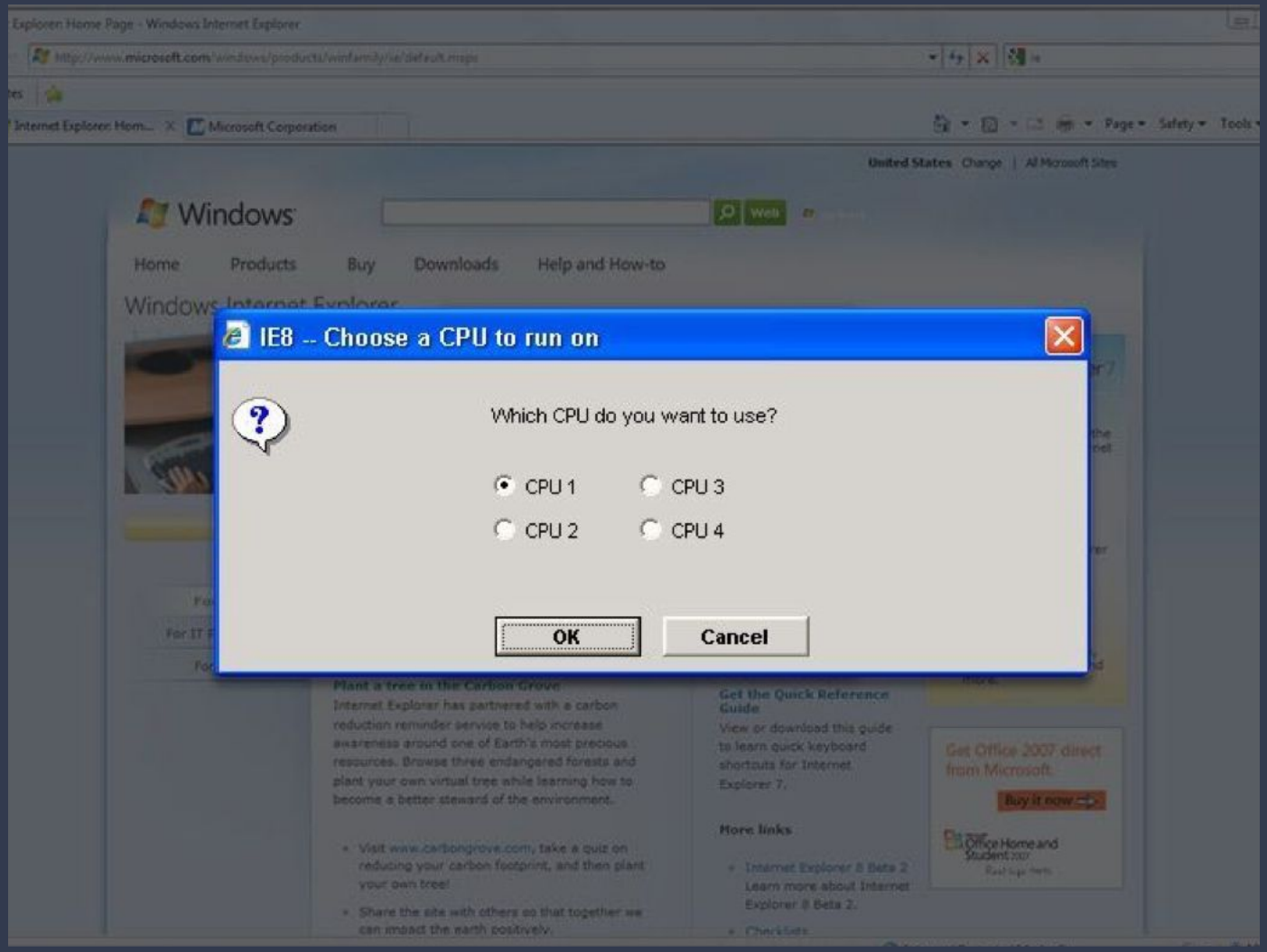
Microservices are hard…

… but are the only widely known way to efficiently scale applications.

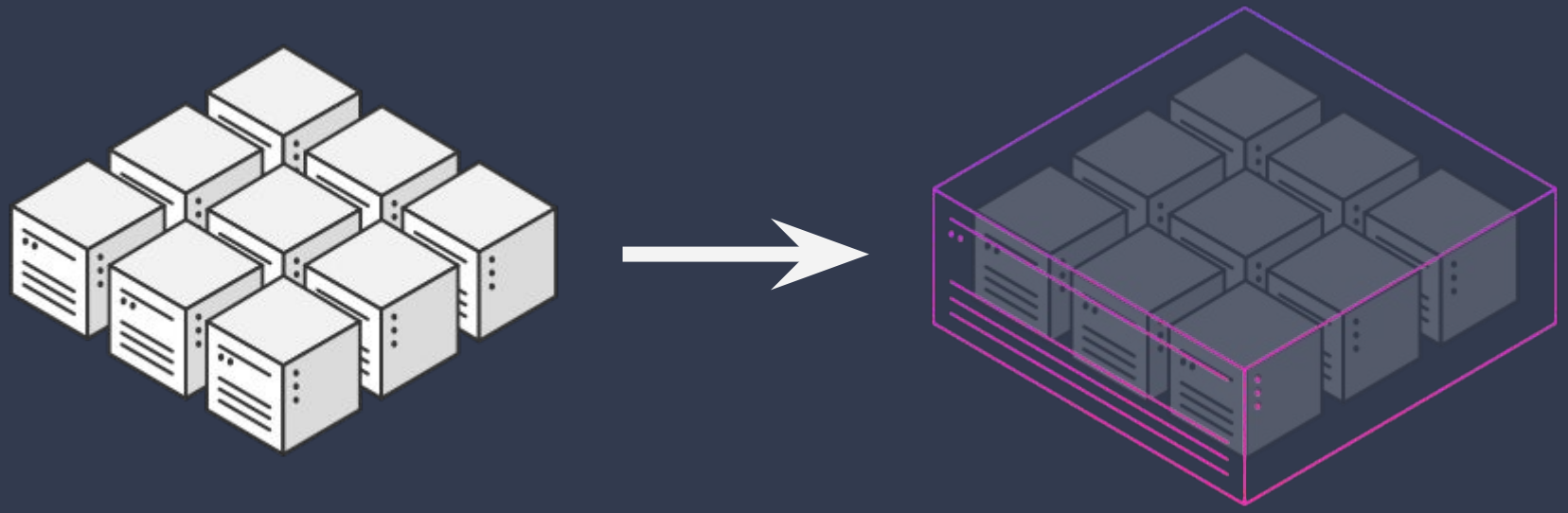
(A new abstraction, anyone?)

What about Apache Mesos?

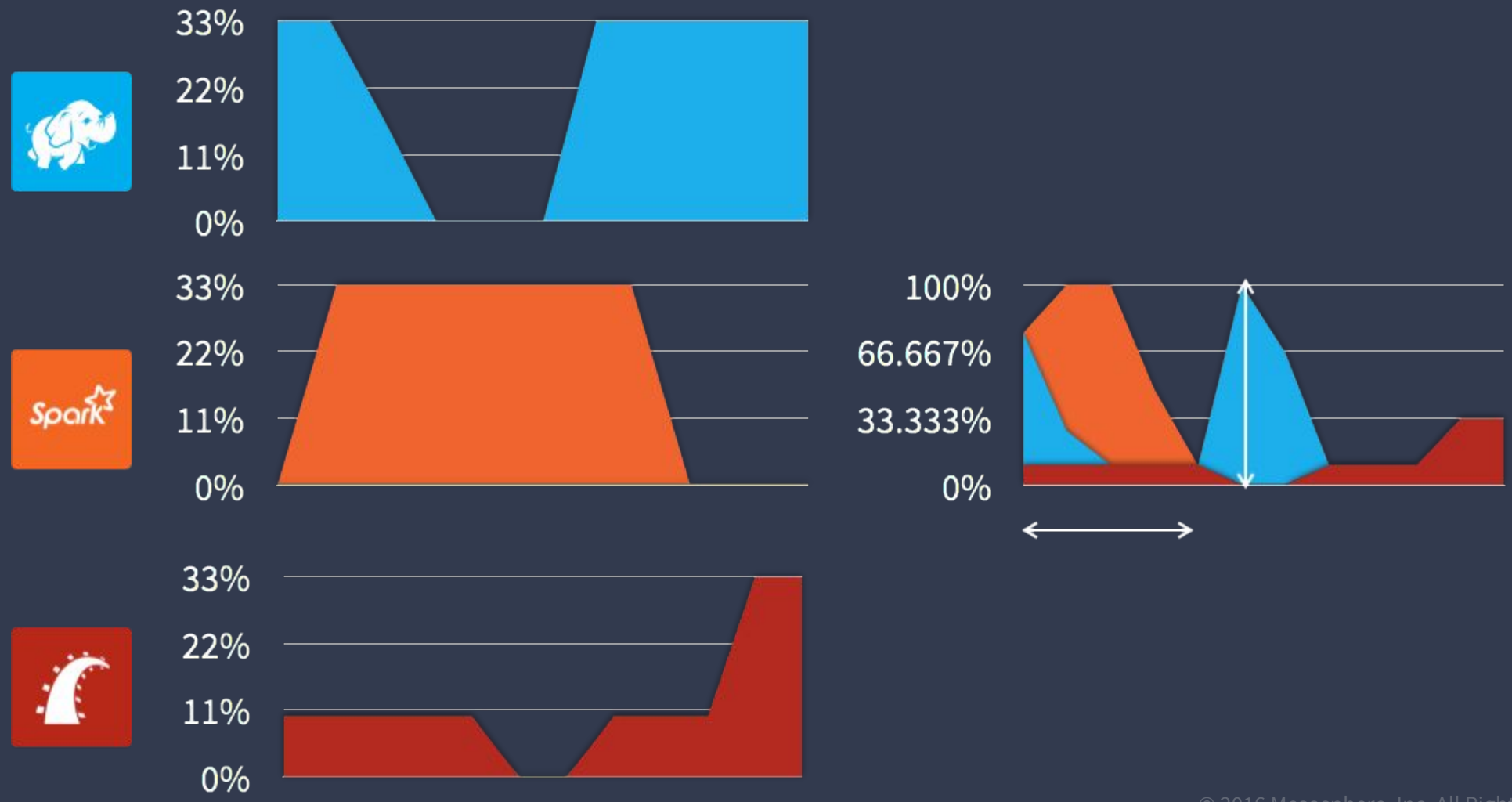
That's how we run datacenters today



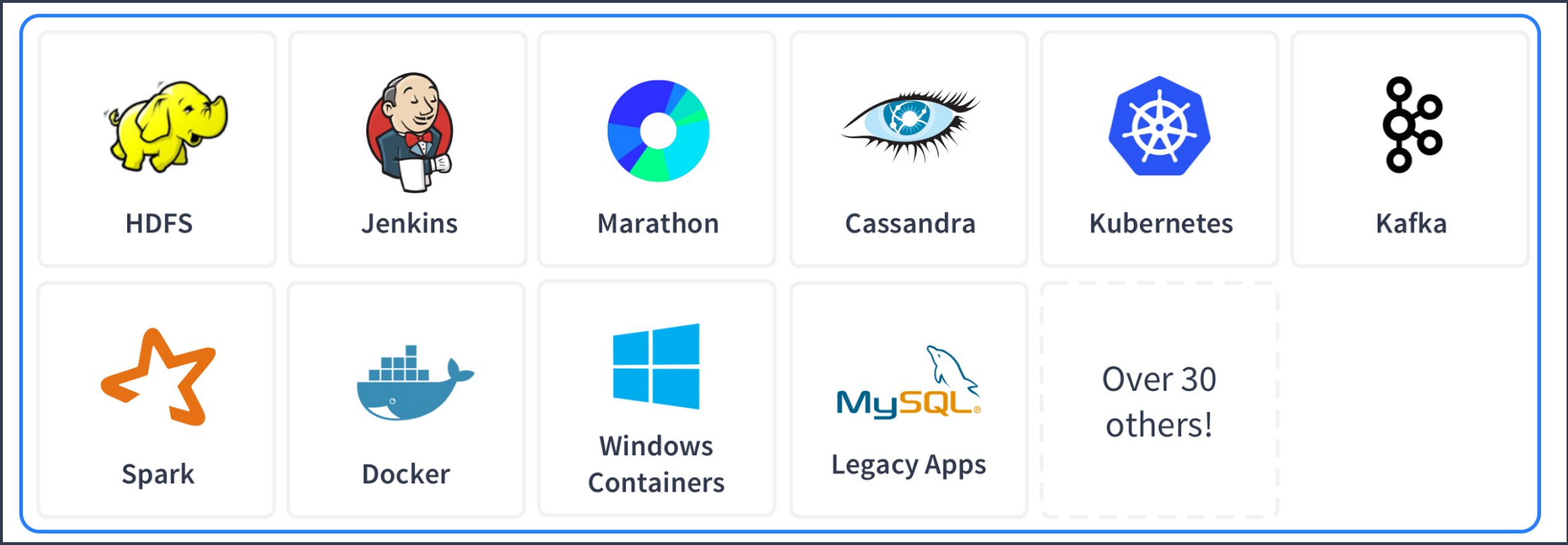
Idea 1: Treat a cluster as one big computer



Idea 2: Run everything in one cluster; manage automatically



Idea 3: A single DSL is not enough



Idea 3: A single DSL is not enough

- Provide primitives instead: `launchTask()`, `killTask()`, `createVolume()`, `statusUpdate()`.
- Enable people to tune their own systems (and create DSLs).
- Allow to port legacy apps.

Bringing all the pieces together

Bringing all the pieces together

Bringing all the pieces together

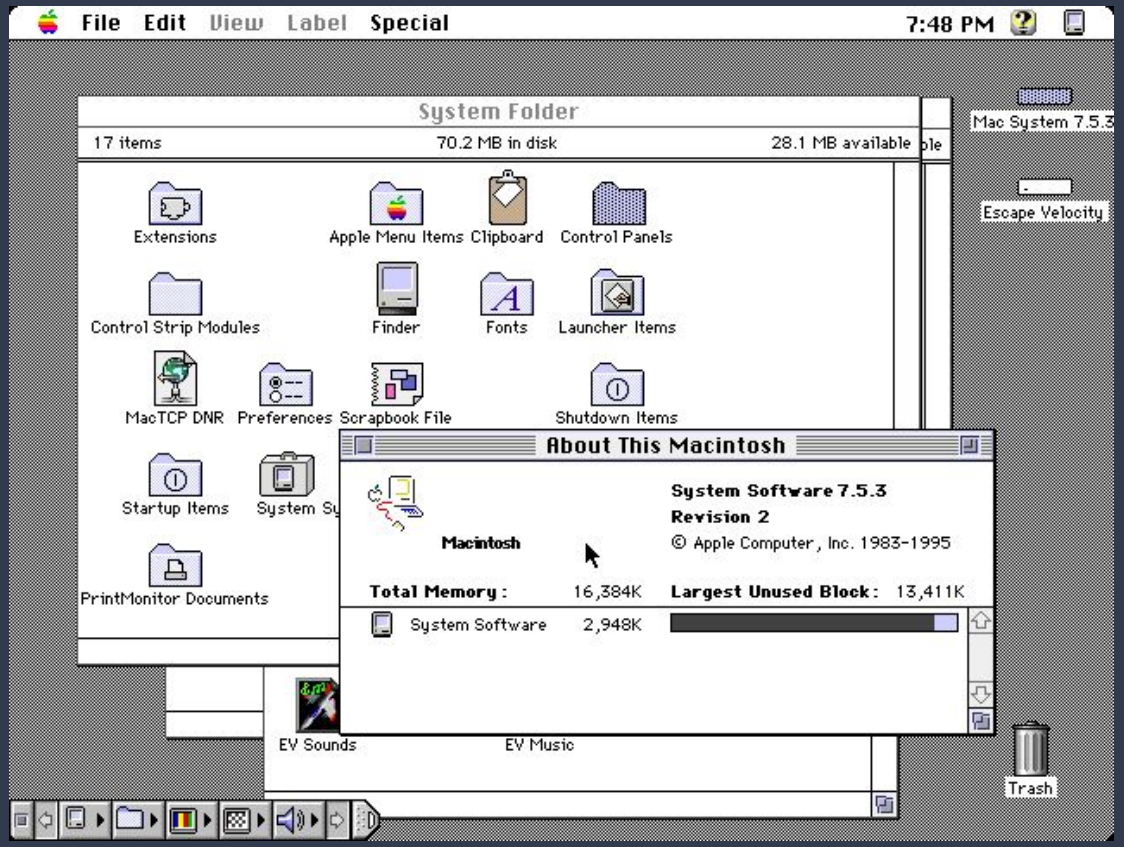
What's ahead?

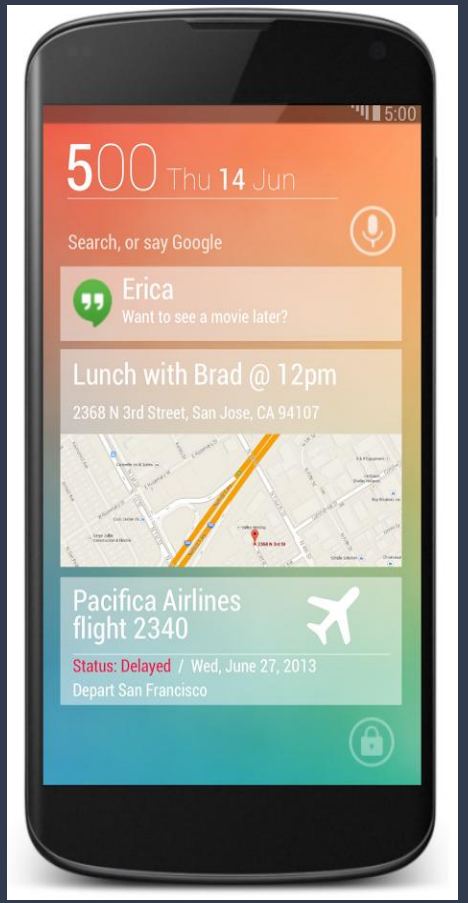
New features

- Priorities for tasks, applications, frameworks
- Quality of resources, i.e. revocable, scarce
- Improved multi-tenancy, e.g. CPI^2
- Tailored allocation algorithms

...

Competition

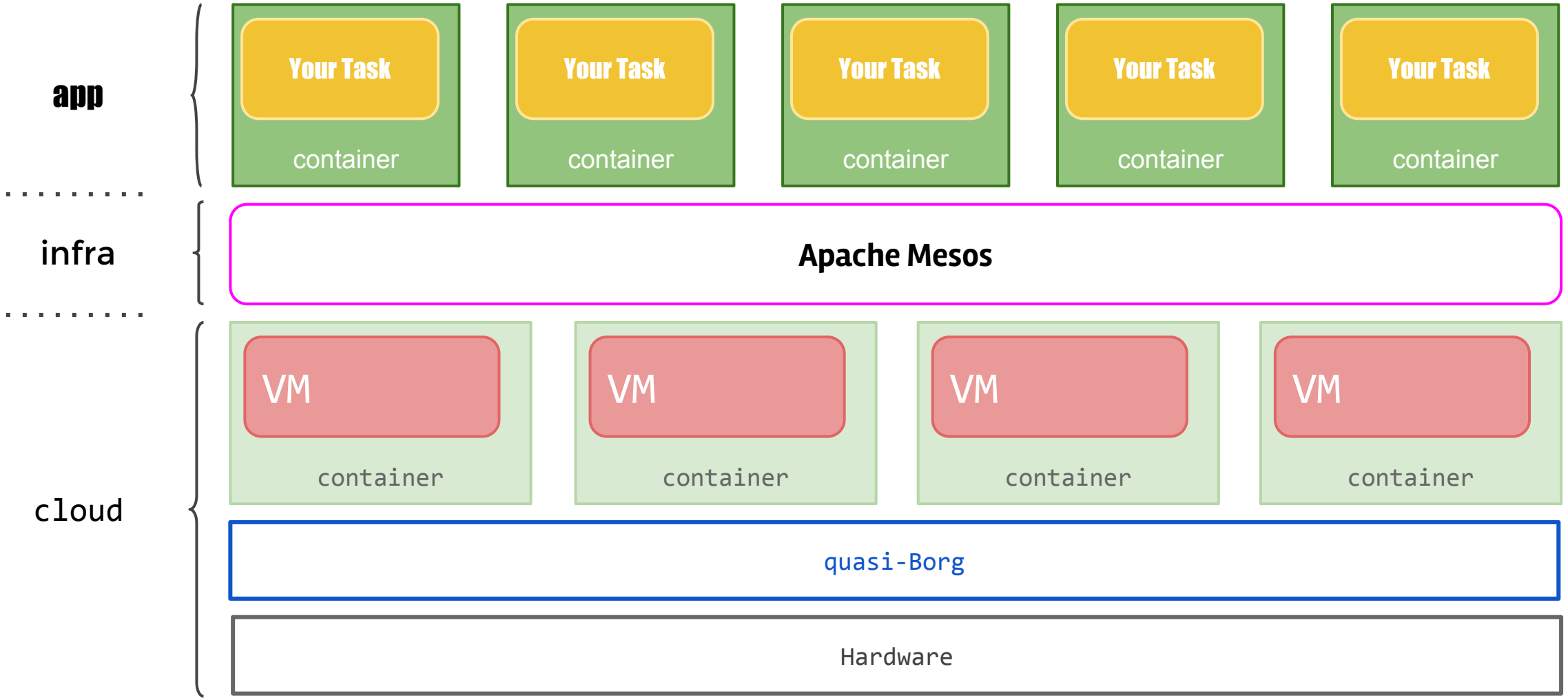




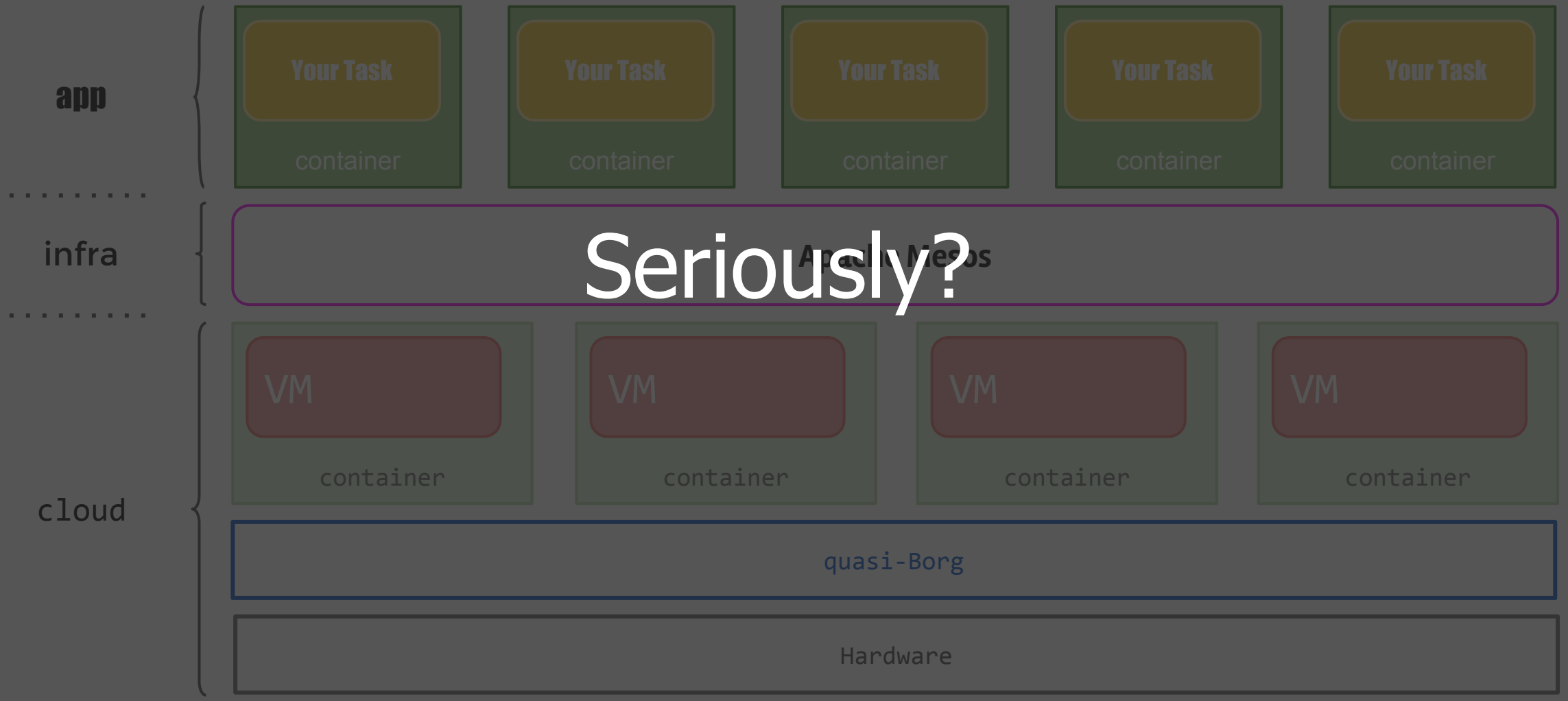
© apple.wikia.com; droid-life.com; youtube.com

Unified stable APIs

Simplified architecture



Simplified architecture



Seriously?

Higher utilization

VISIT OUR BOOTH

Learn more by visiting dcos.io and mesosphere.com

THANK YOU!