DISTRIBUTED SYSTEMS: ERLANG, RIAK CORE, RIAK PIPE

A few clues, a few rants, and a call to action

ABOUT ME

- Technical Evangelist for Basho
- @macintux
- jdaily@basho.com
- http://tinyurl.com/jrdlj13

300 sec	is how long it takes to launch a Linux instance to availability on Amazon EC2.
50 sec	is the time between power on and the lock screen for an Android phone.
0.4 sec	ago is when we received your request. Within this time we managed to create a new Xen instance, boot it, and run the application that rendered the page you are viewing. By the time you are done reading this, the instance will be gone.

ZERG.ERLANGONXEN.COM

THOU SHALT LOOK AT ERLANG

John's first commandment of infrastructure software

Tom Glover
@thomasglover

Follow

I'm convinced that if Tolkien had devised a programming language it would have been Erlang. I mean that as a compliment BTW :-)

Reply Retweet Favorite More

4
RETWEETS

10:29 AM - 4 Jul 13

Alain O'Dea
@AlainODea

Follow

@koleksiuk learning Erlang rewrote the way I think about production systems in too many ways to list. It was a profound benefit to me.

Reply Retweet Favorite More

4 RETWEETS

3 FAVORITES

12:21 PM - 6 Jul 13

The Erlang virtual machine provides fault-tolerance features and a highly scalable architecture while the Erlang programming language keeps the required source code small and easy to follow.

It's important to understand that few programming languages can provide real fault-tolerance with scalability. In fact, I'd say Erlang is roughly alone in this regard.

MICHAEL TRUOG

WHY DISTRIBUTED PROGRAMMING IS HARD

- Clocks

WHY DISTRIBUTED PROGRAMMING IS HARD

- Clocks
- Latency

WHY DISTRIBUTED PROGRAMMING IS HARD

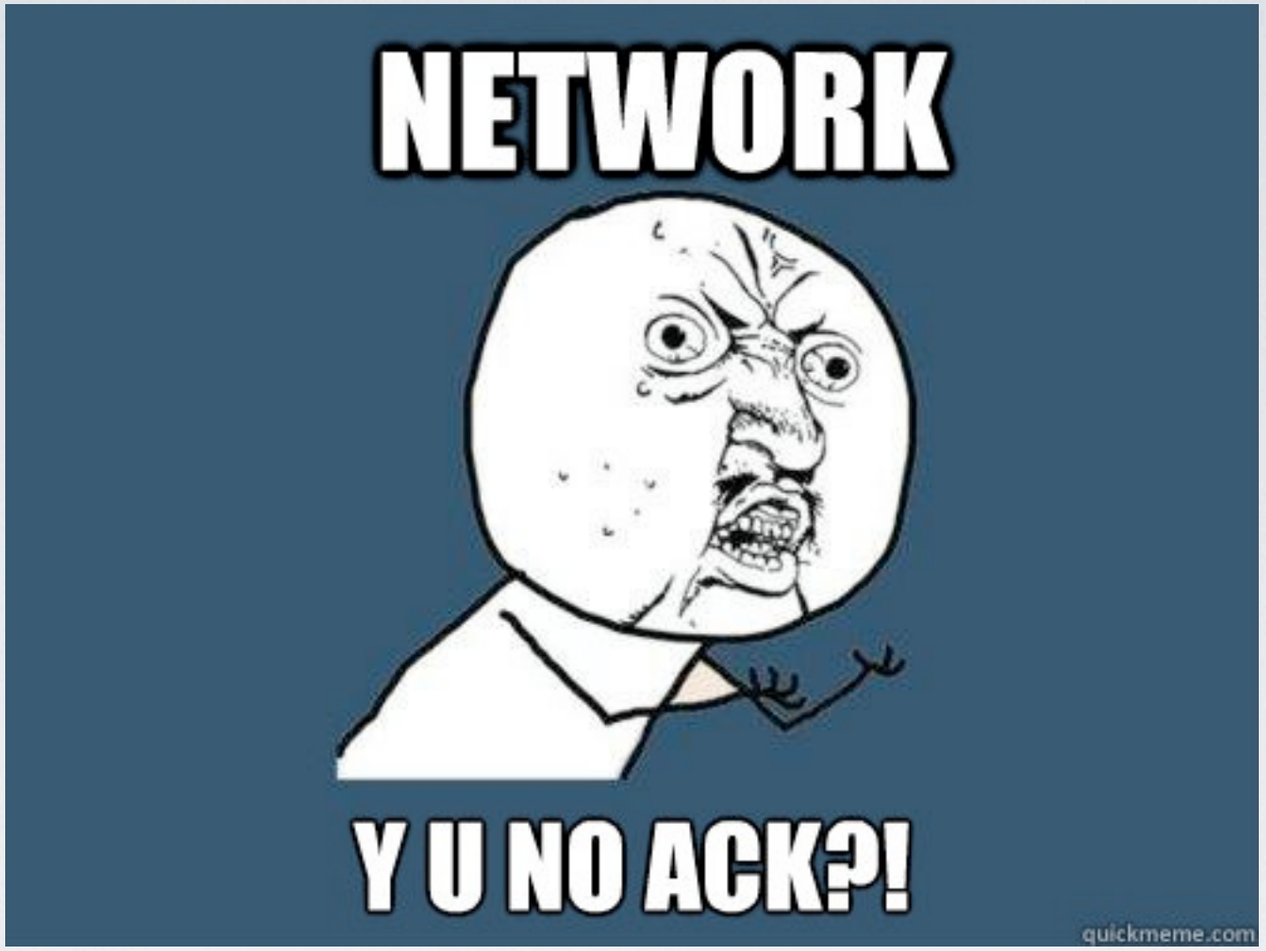
- Clocks
- Latency
- Lost messages

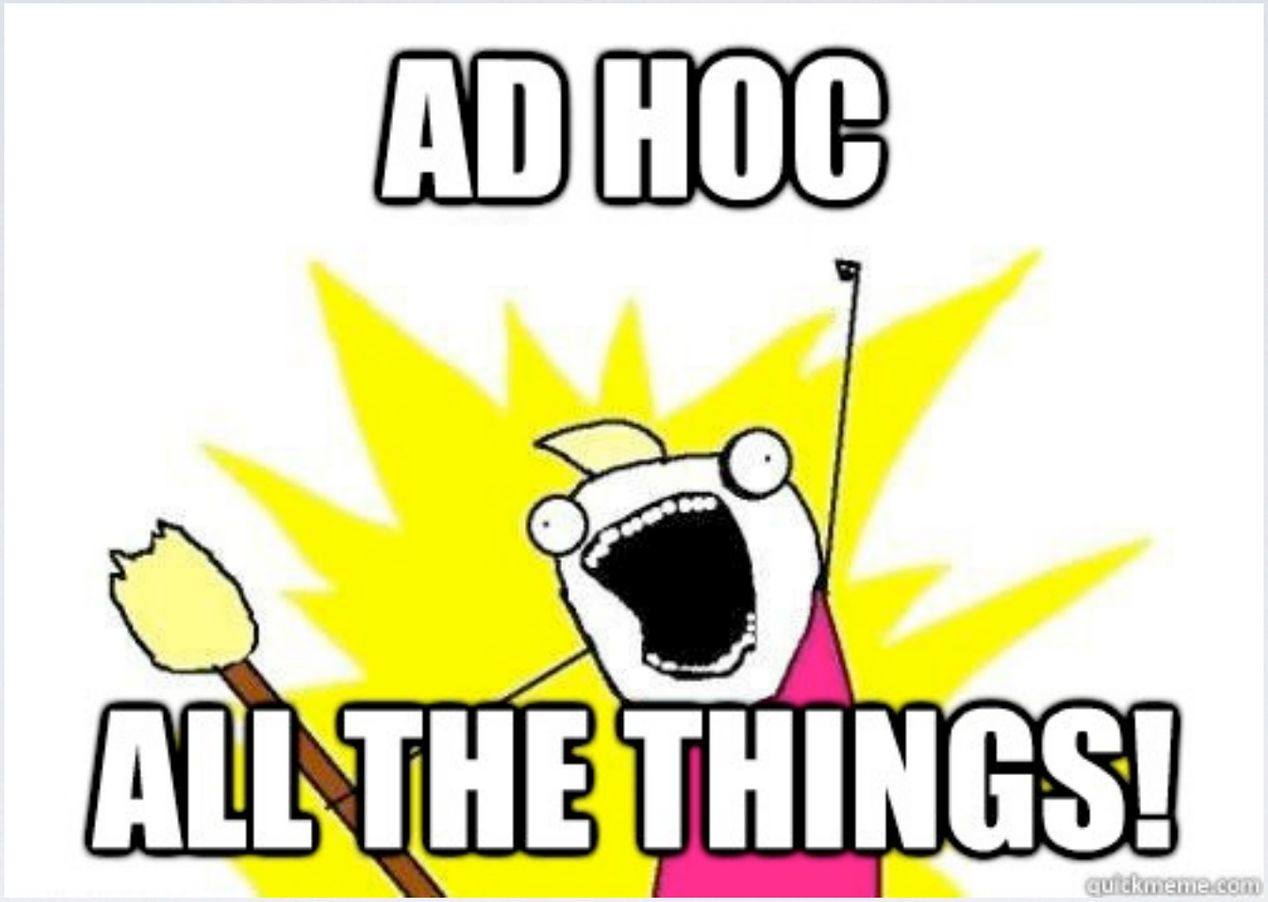
WHY DISTRIBUTED PROGRAMMING IS HARD

- Clocks
- Latency
- Lost messages
- Servers break

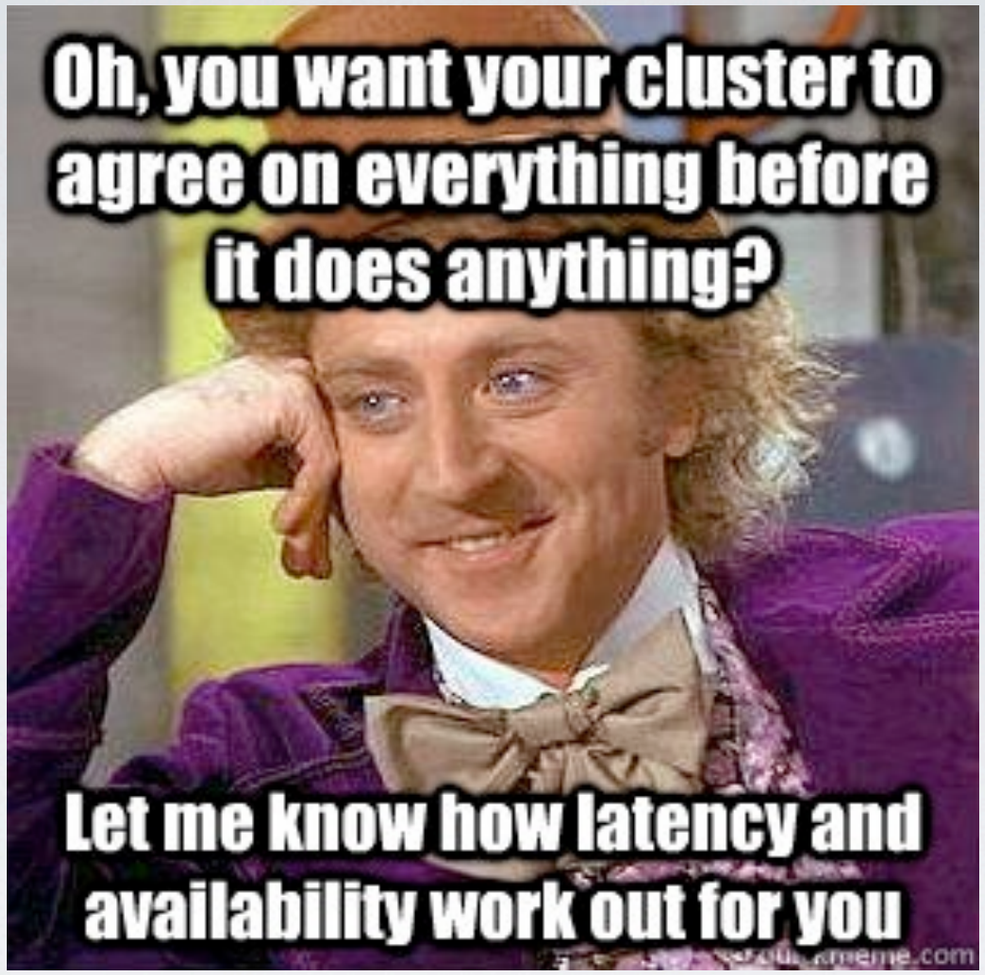

CHASED AN AMBULANCE
CLIENT ALREADY DEAD.
quickmeme.com

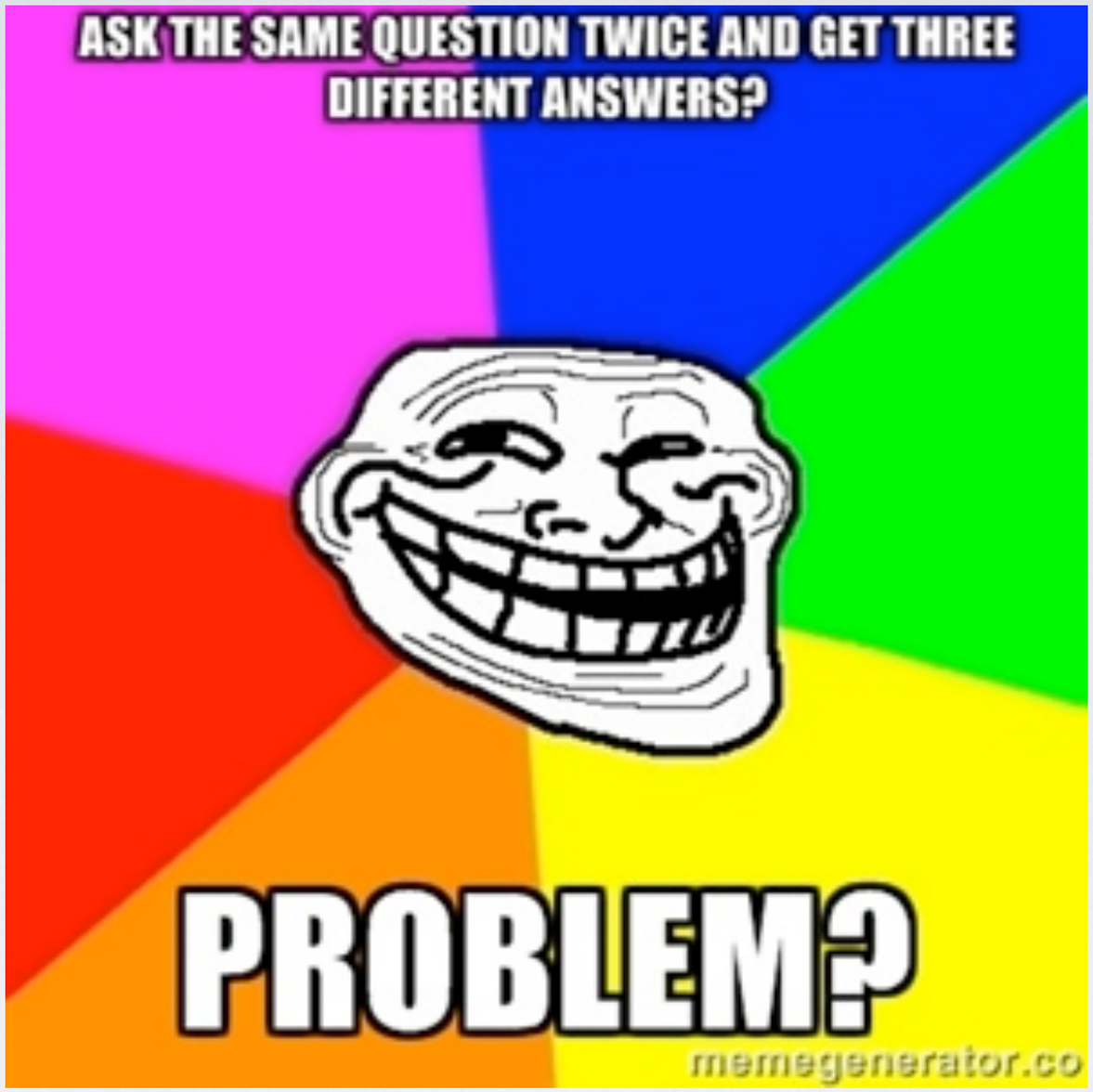
CAP LIGHTNING TALK


NETWORK
Y U NO ACK?!
quickmeme.com


AD HOC
ALL THE THINGS!
quickmeme.com


ONE DOES NOT SIMPLY
TURN OFF ALL THE SERVERS
quickmeme.com


Oh, you want your cluster to agree on everything before it does anything?
Let me know how latency and availability work out for you


ASK THE SAME QUESTION TWICE AND GET THREE DIFFERENT ANSWERS?
PROBLEM?
memegenerator.co

/CAP LIGHTNING TALK

/memes

DYNAMO & RIAK

- Key/value store

DYNAMO & RIAK

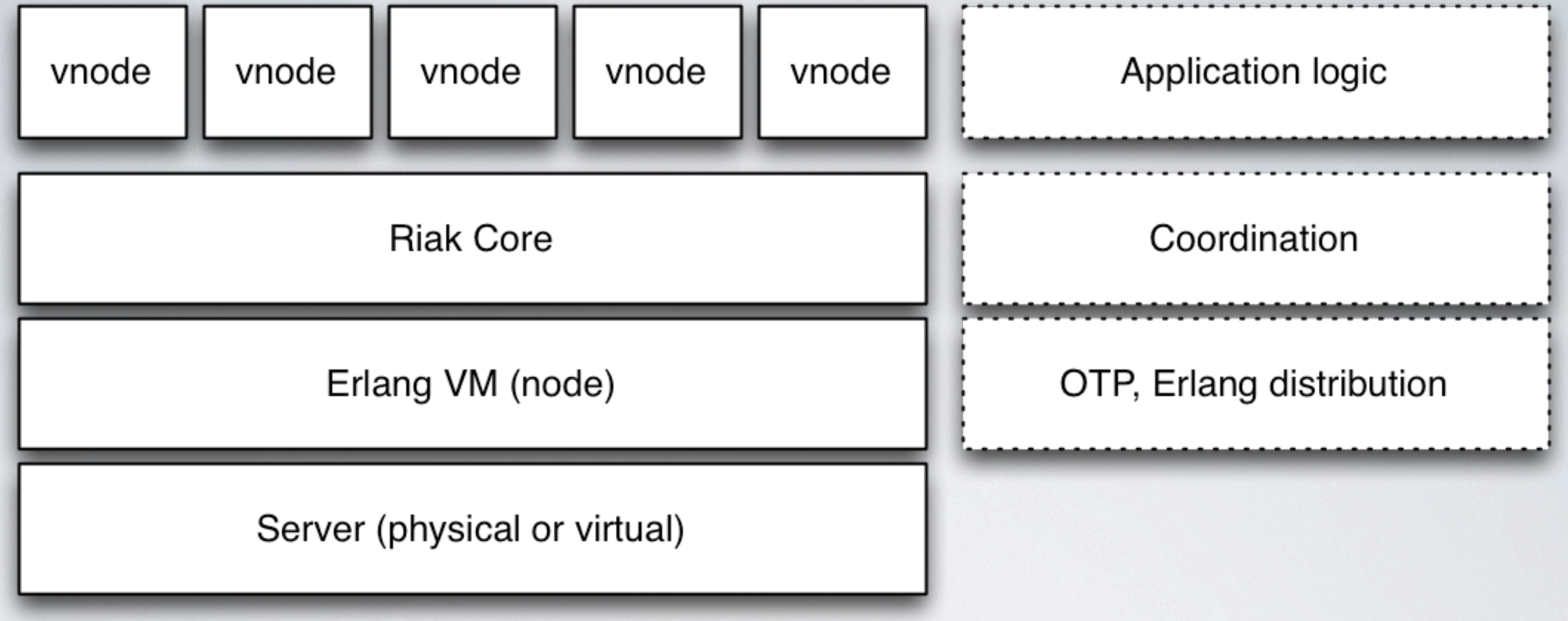
- Key/value store
- Consistent hashing

DYNAMO & RIAK

- Key/value store
- Consistent hashing
- Granular compute engines

DYNAMO & RIAK

- Key/value store
- Consistent hashing
- Granular compute engines
- Replication


vnode
vnode
vnode
vnode
vnode
Riak Core
Erlang VM (node)
Server (physical or virtual)
Application logic
Coordination
OTP, Erlang distribution

jingles

Web User Interface
- pluggable
- interacts via api
- speaks to howl to reflect live events
- speaks to wiggle to get data and handle its state.

howl

Live Communications Server
- reflects live updates in user interface when changes occur in the backend.
- speaks to snarl to check with live events users have permission to see
- speaks to jingles trigger event such as ,notifications etc

wiggle

API Server
- complete api
- speaks to snarl to authenticate users and check permissions
- speaks to sniffle to get information and data

snarl

Rights Management
- users
- groups
- fine grained RBAC style permissions

sniffle

Central Management Service "The Brain"
- where most important logic is performed
- where most important data is stored
- where actions are triggered e.g. vm create
- speaks to snarl for granting user rights and for filtering what a user is allowed to see
- speaks to howl so that users get live updates when changes happen in the backend via asynchronous model to ensure users who are already authenticated get live information events in realtime e.g vm boots, reboots
- speaks to chunter to automate tasks and trigger events

chunter

Hypervisor Global Zone Service
- interacts with SmartOS e.g via "vmadm" etc
- speaks to howl to stream data such as performance data and state changes etc

The spirit of the cloud is ‘every node can die’, writing the administration software for a cloud expecting the administration nodes to stay alive is hypocritical.

Yet just saying ‘hey the API will be down if you lose the controlling node’ kind of sucks so it comes down having a distributed system either with stateless nodes and a distributed database or with distributed nodes and a local database.

HEINZ GIES, PROJECT FIFO

```erlang
next_id() ->
    command(next_id).

%% @doc Pings a random vnode to make sure communication is functional
ping() ->
    command(ping).

command(Cmd) ->
    CmdBin = list_to_binary(atom_to_list(Cmd)),
    DocIdx = riak_core_util:chash_key({CmdBin, term_to_binary(now())}),
    PrefList = riak_core_apl:get_primary_apl(DocIdx, 1, riak_id),
    [{IndexNode, _Type}] = PrefList,
    riak_core_vnode_master:sync_spawn_command(IndexNode, Cmd, riak_id_vnode_master).
```

```
handle_command(next_id, Sender, #state{last_timestamp = TS, sequence = Seq, machine_id = Machine} = State) ->
    case get_next_seq(TS, Seq) of
        backwards_clock ->
            {reply, {fail, backwards_clock}, State};
        exhausted ->
            %% Retry after a millisecond
            erlang:sleep(1),
            handle_command(next_id, Sender, State);
        {ok, Time, NewSeq} ->
            {reply, construct_id(Time, Machine, NewSeq), State#state{last_timestamp = Time, sequence = NewSeq}}
    end;
handle_command(Message, _Sender, State) ->
    ?PRINT({unhandled_command, Message}),
    {noreply, State}.
```

RIAK PIPE

Like UNIX pipes, but not!

RIAK PIPE COMPONENTS

- Pipeline (client, fittings, sink)
- Fitting (specification of behavior, ***partitioner***)
- Coordinator (one/fitting/cluster)
- Worker process (one/fitting/vnode, implements a behavior)

RIAK PIPE PROCESSING

1. Client submits a pipeline specification
2. Coordinators are spawned from last to first
3. When input arrives at a vnode, if no worker for that fitting exists yet the appropriate coordinator will be asked for the fitting specification
4. The last fitting sends its output to the sink, which is typically the client

RIAK PIPE DEBUGGING

- Log and trace messages are optionally sent to the sink
- riak_pipe:status will return a list of fittings; each item is a list of worker details: node ID, state, queue length, idle time, etc
- Fittings can define a validator function to be invoked on inputs before processing
- Intermediate outputs can be forwarded directly to sink

THE END IS NIGH

- Riak 1.4
- RICON
- Meetups