

Microservices and Erlang/OTP

Christoph Iserlohn



About me

Senior Consultant @ innoQ

MacPorts Team member

Agenda

- > Microservices
- > Erlang / OTP
- > How they fit together

Microservices

Attempt of definition

- > A system consisting of small, self-contained services. All running isolated from each other, communicating only over the network.

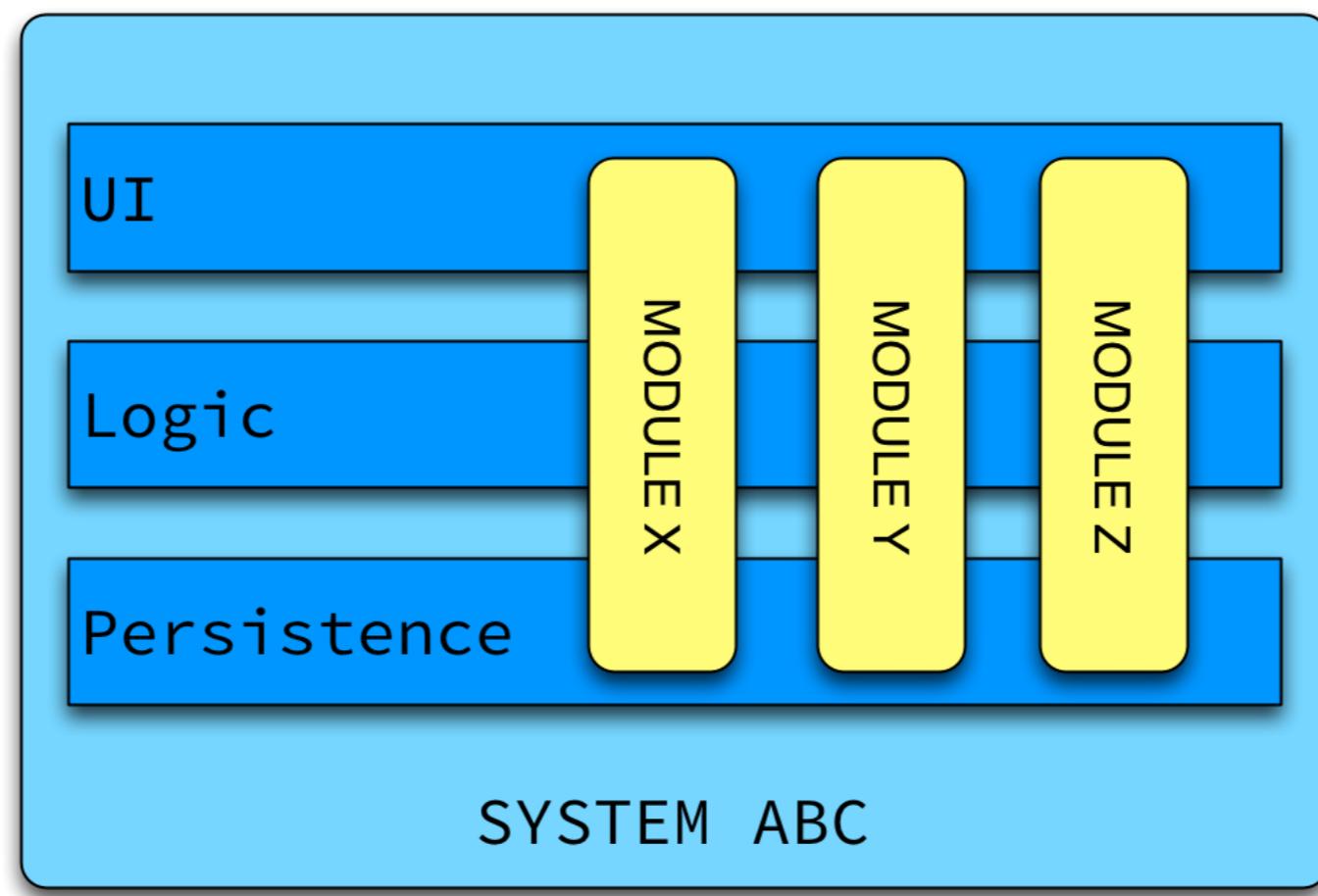
Monoliths

old and busted

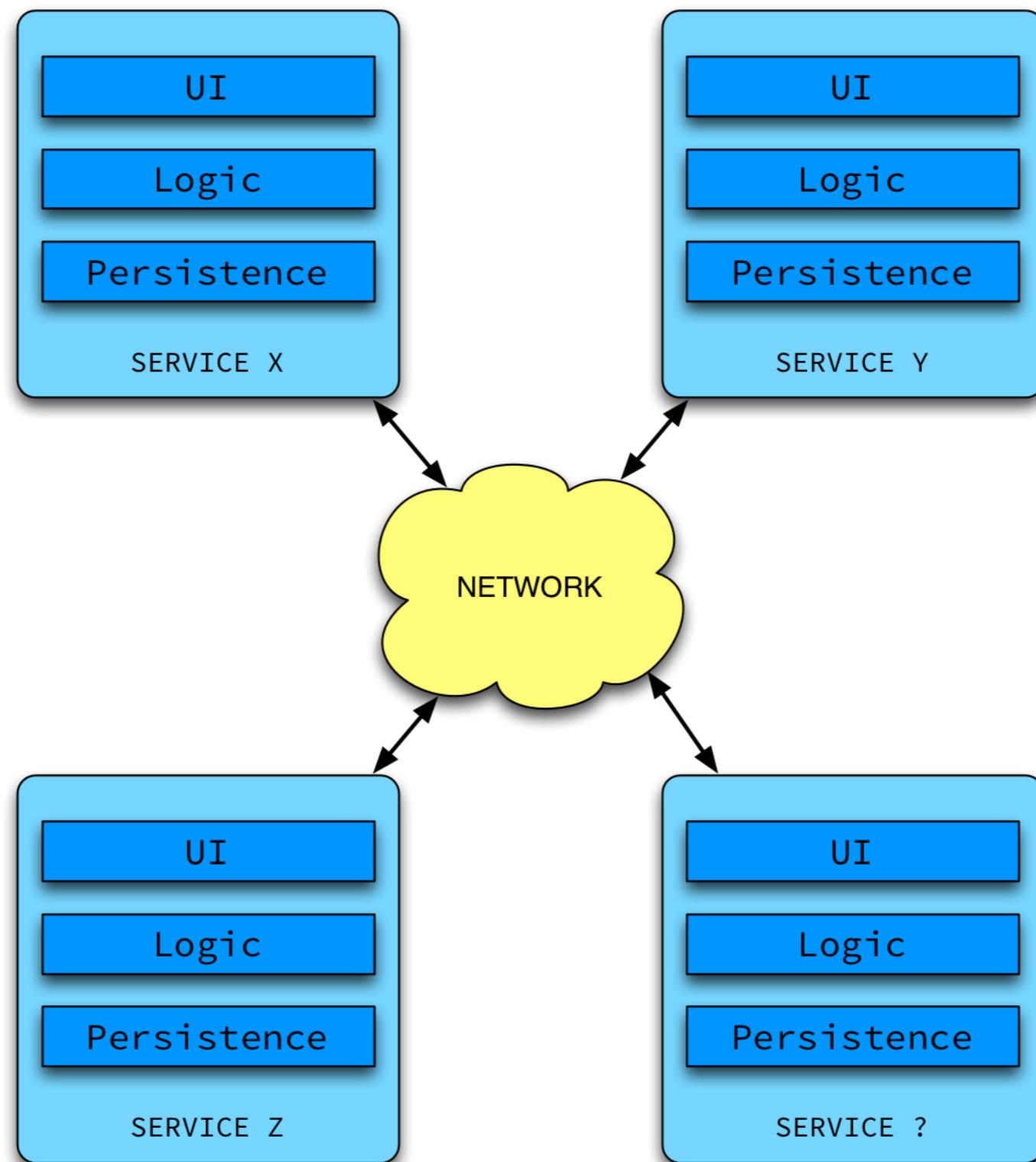


Microservices

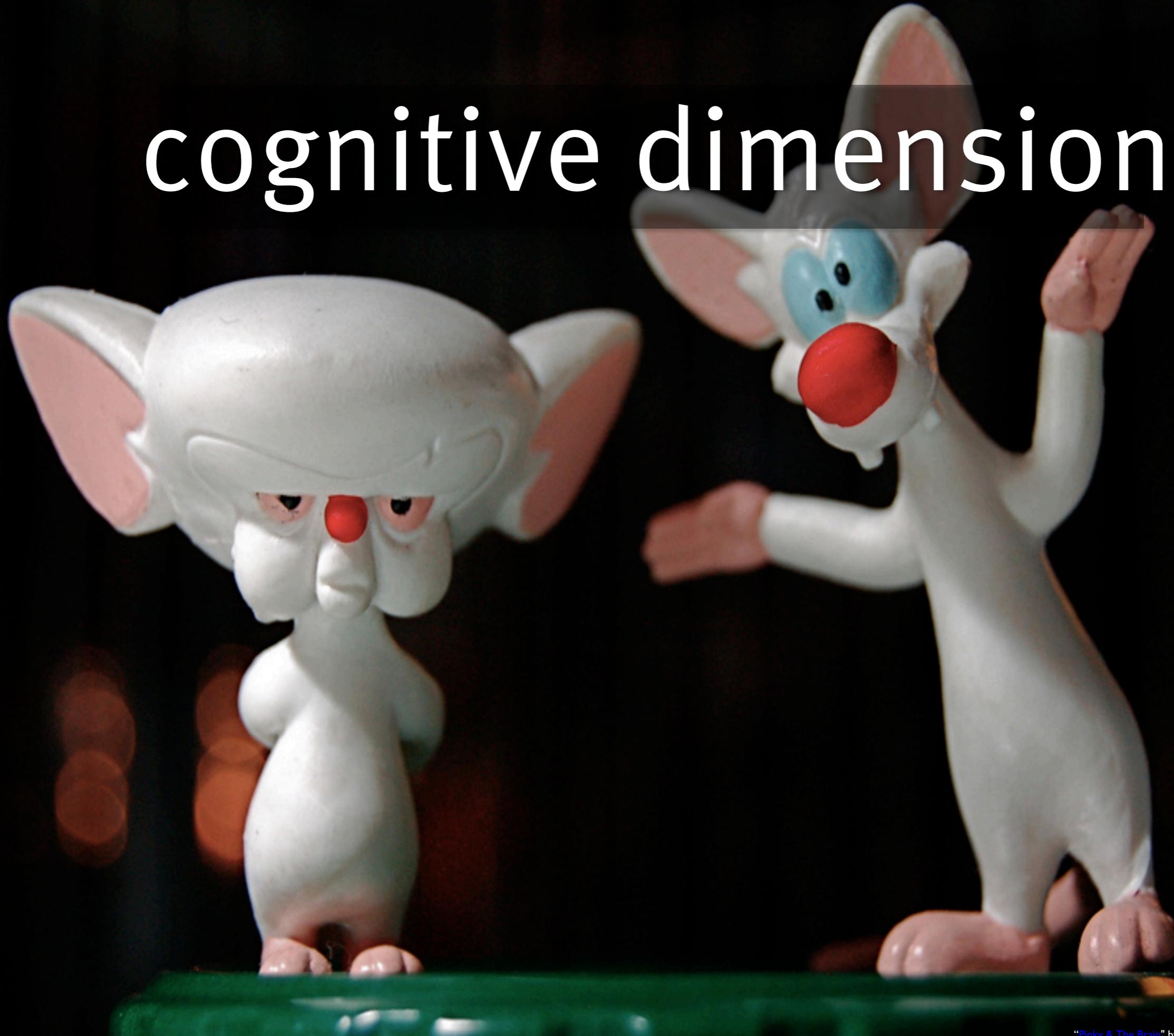
the new hotness



VS.



cognitive dimension





on the service level:
more comprehensible



on the system level:
unable to see the
big picture



organisational
dimension



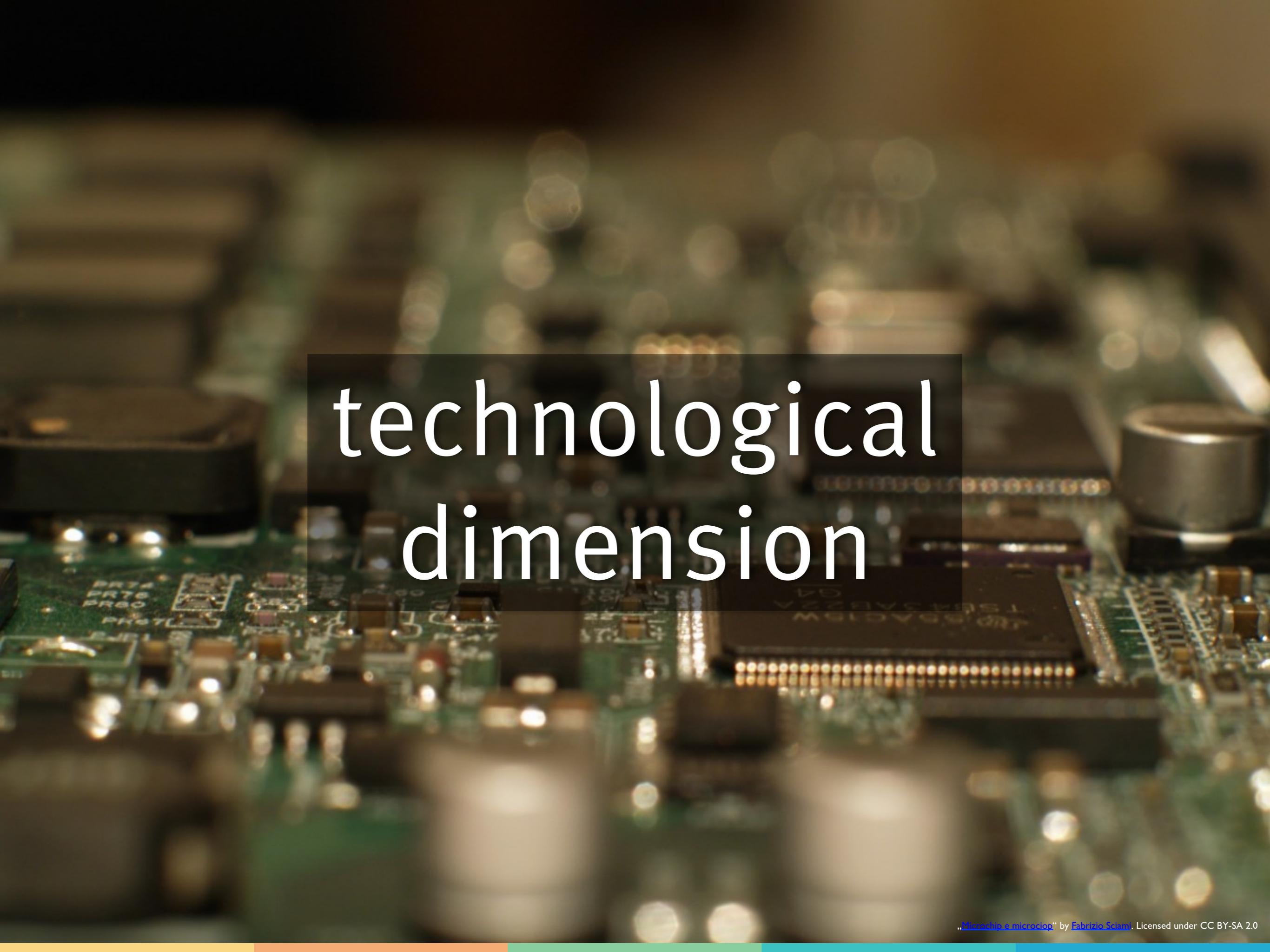
organized around
business capabilities

cross-functional teams



„you build it, you run it“





technological
dimension



fault tolerance resilience



asynchronous communication

coarse-grained interfaces

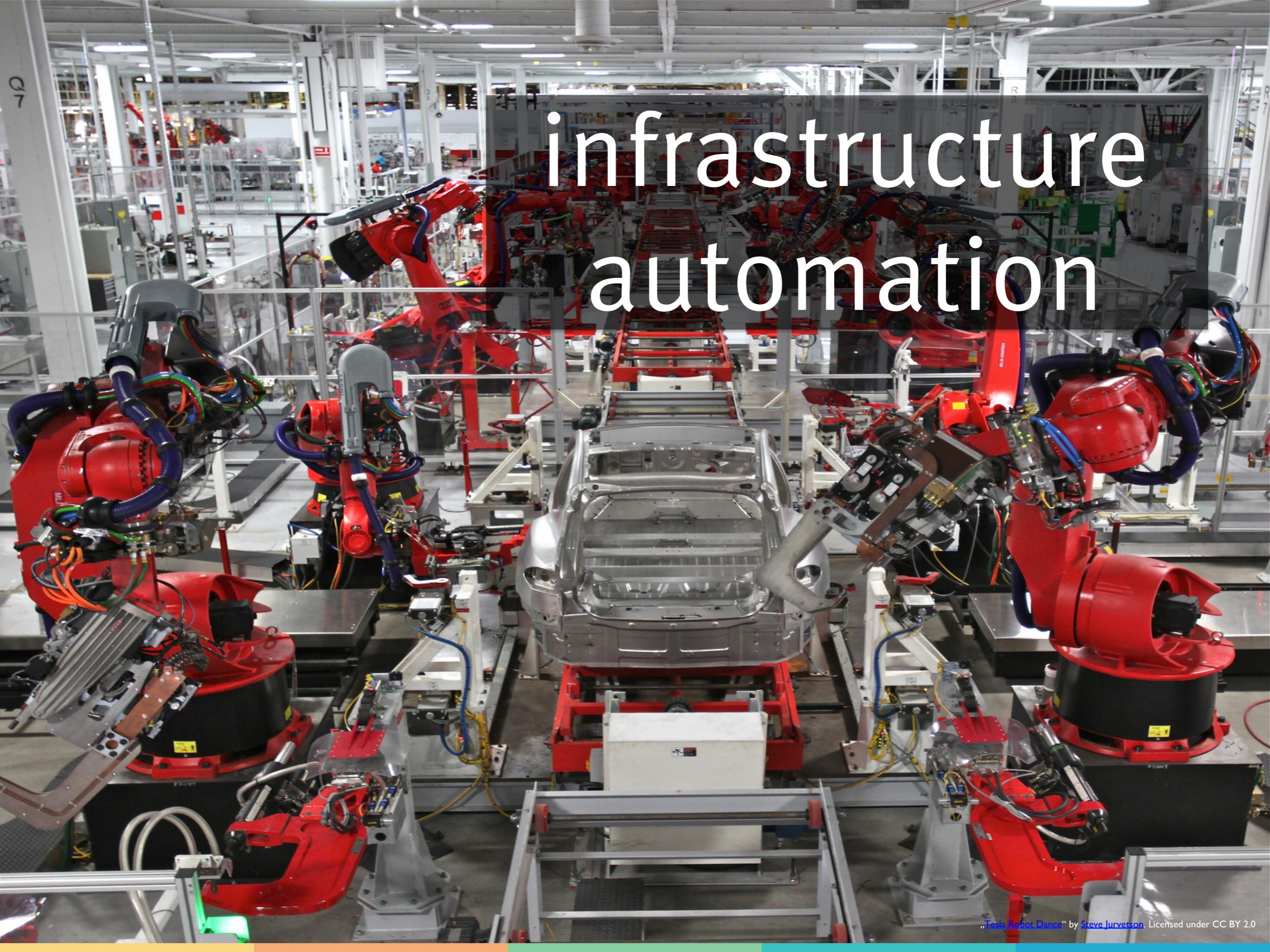


sophisticated monitoring



"Mission control center" by NASAOriginal uploader was Cjosefy at en.wikipediaLater version(s) were uploaded by TheDJ at en.wikipedia. - <http://spaceflight.nasa.gov/gallery/images/shuttle/sts-114/html/jsc2005e09159.html>Transferred from en.wikipedia. Licensed under Public Domain via Wikimedia Commons - http://commons.wikimedia.org/wiki/File:Mission_control_center.jpg#mediaviewer/File:Mission_control_center.jpg

infrastructure automation



Advantages

- > fast development cycle
- > it's easy to scale
- > flexibility of implementation
- > easy to get started for new developers
- > parts of the system can be replaced

Prerequisites

- > monitoring the whole system
- > central logging
- > tracing across service boundaries
- > automatic deployment
- > automatic provisioning

Challenges

- > service boundaries
- > contracts and governance
- > testing and refactoring
- > fallacies of distributed systems
- > support for a dozen technology stacks

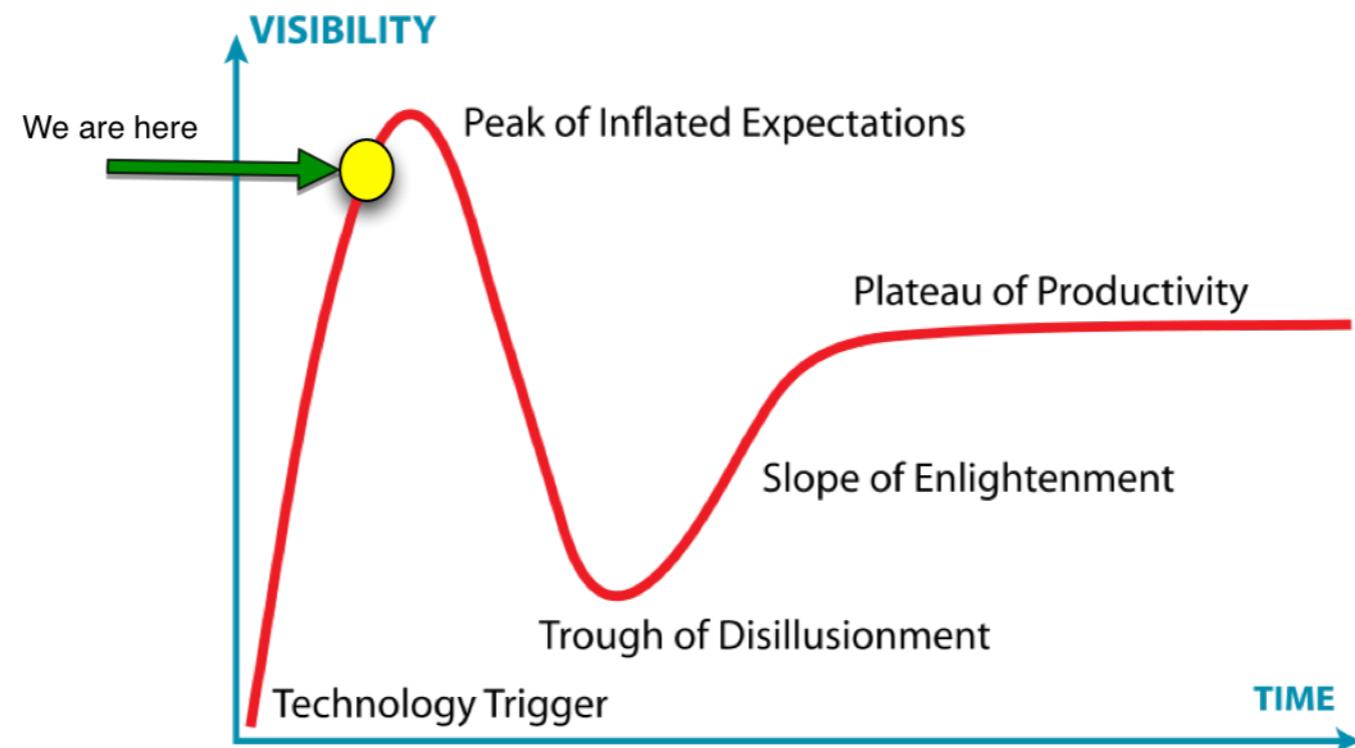
Open questions

- > how big?
- > isn't this just SOA?

Summary

- > it's a promising approach,
- > but don't start with it mindlessly

Where are we now?



Erlang / OTP

What is Erlang / OTP?

- > a general purpose programming language
- > runtime environment and VM
- > Open Telecom Platform: libraries, tools and design patterns for building highly concurrent, distributed, fault tolerant systems





PROBLEM

PROBLEM SOLVING

The image features a red rectangular road sign with a white border and rounded corners. The word "PROBLEM" is positioned at the top in large, bold, white capital letters. Below it, two white arrows point upwards towards the word "SOLVING". The word "SOLVING" is also in large, bold, white capital letters. The sign is mounted on a vertical gold-colored metal pole. The background consists of a clear blue sky with scattered white, fluffy clouds.



Your PC ran into a problem and needs to restart. We're just collecting some error info, and then we'll restart for you. (0% complete)

If you'd like to know more, you can search online later for this error: HAL_INITIALIZATION_FAILED

fault tolerant to software
and hardware errors

distributed systems



non-stop running -
continuous operation
over years

Principles

- > lightweight concurrency
- > asynchronous communication
- > isolation
- > error handling
- > simple high-level language
- > tools not solutions or products

Erlang – the language

- > high-level functional language
- > prolog inspired syntax
- > dynamically typed / safe
- > pattern matching everywhere
- > recursion
- > immutable data and variables

```
-module(factorial).  
-export([factorial/1]).  
  
factorial(N) when N >= 0 -> factorial(N,1).  
  
factorial(0,Acc) -> Acc;  
factorial(N,Acc) -> factorial(N-1,N*Acc).
```

Concurrency

- > millions of processes on one machine
- > processes are isolated
- > processes are used for everything:
 - > concurrency
 - > managing state
 - > parallelism
- > no global data

Message passing

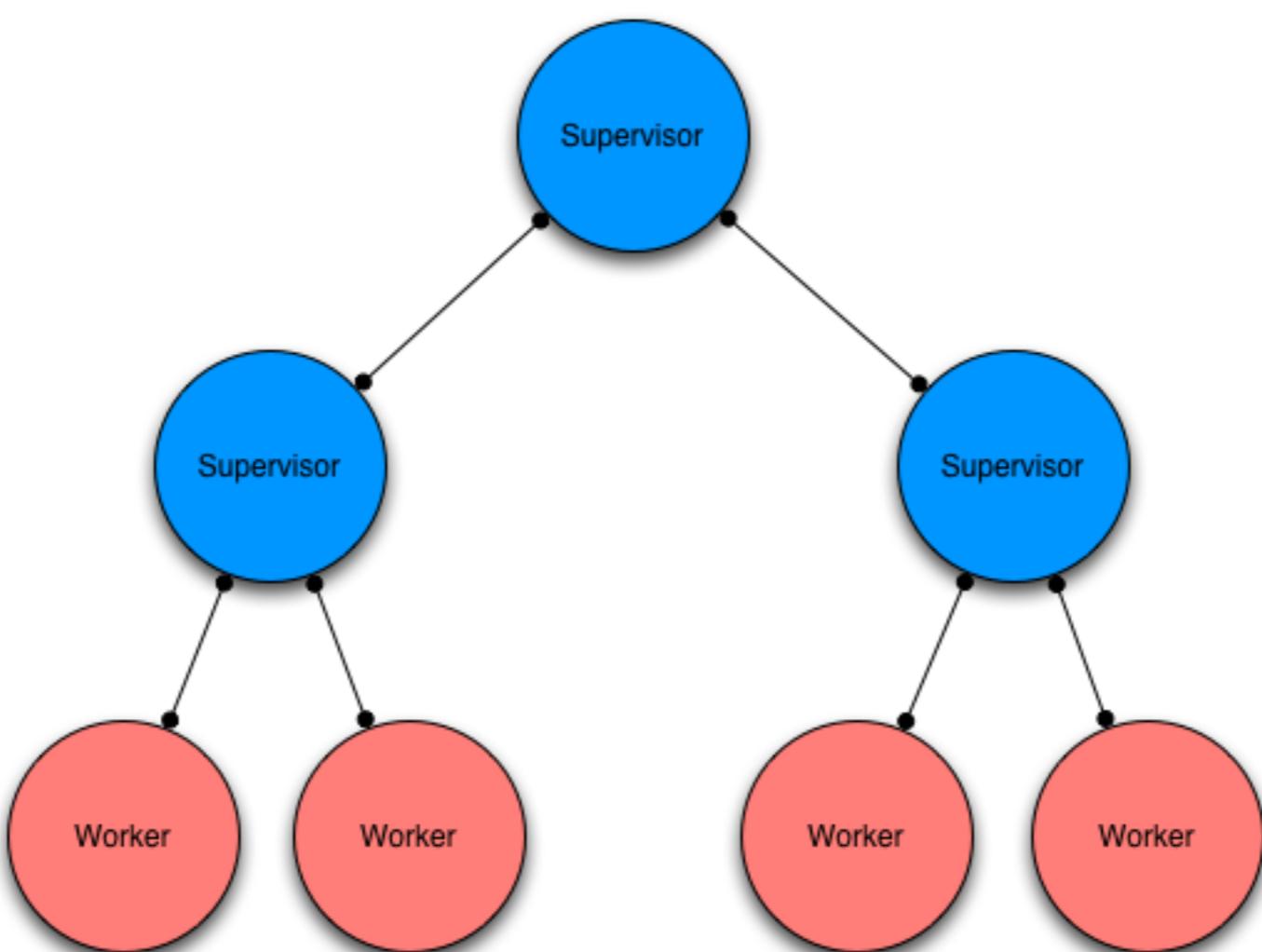
- > asynchronous
- > primitives:
 - > fire & forget send
 - > selective receive
- > more complex interactions can be built on top of these primitives

```
-module(pingpong).
-export([start/1]).  
  
start(N) when N > 0 ->  
    Pong = spawn(fun pong/0),  
    ping(N, Pong).  
  
ping(0,Pong) ->  
    Pong ! exit,  
    ok;  
ping(N, Pong) ->  
    Pong ! {self(), ping},  
    receive  
        pong ->  
            io:format("Pid ~p: got pong. ~p pings left~n", [self(), N-1])  
    end,  
    ping(N - 1, Pong).  
  
pong() ->  
    receive  
        {From, ping} ->  
            io:format("Pid ~p: got ping from ~p~n", [self(), From]),  
            From ! pong,  
            pong();  
        exit ->  
            ok  
    end.
```

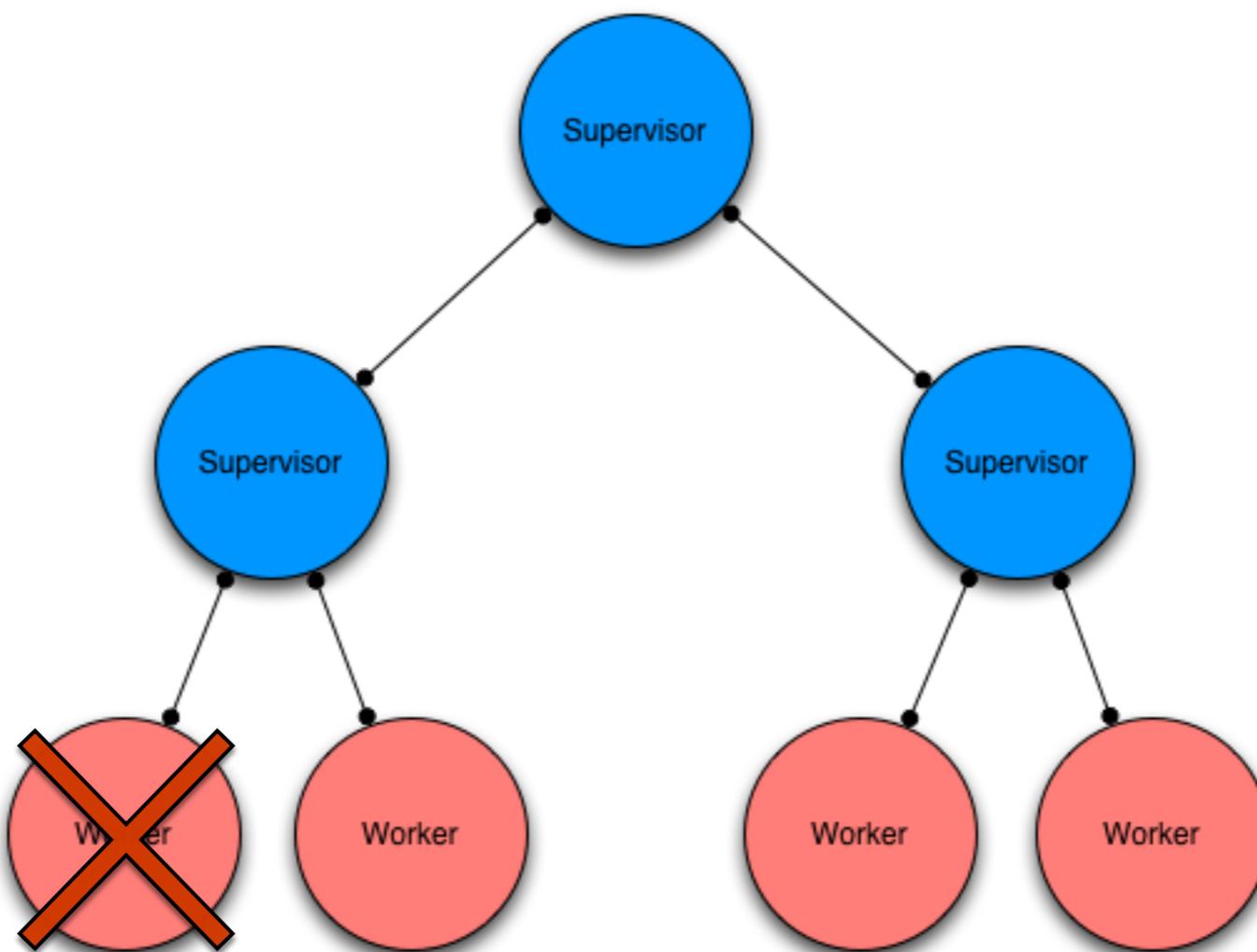
Error handling

- > avoid error checking code everywhere
- > let it crash
- > process based:
 - > link - bidirectional
 - > monitor - unidirectional
- > supervision trees

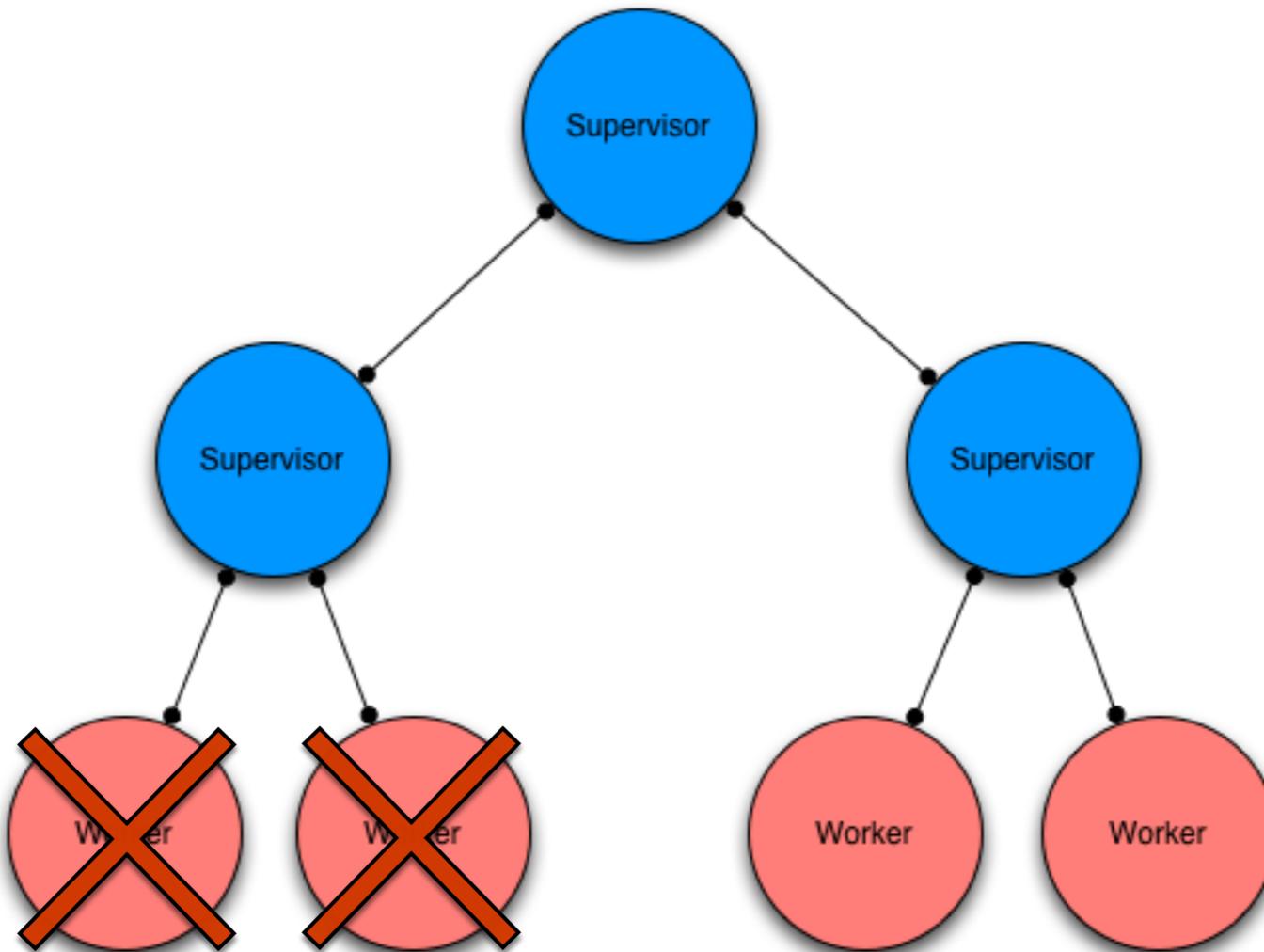
Supervision trees



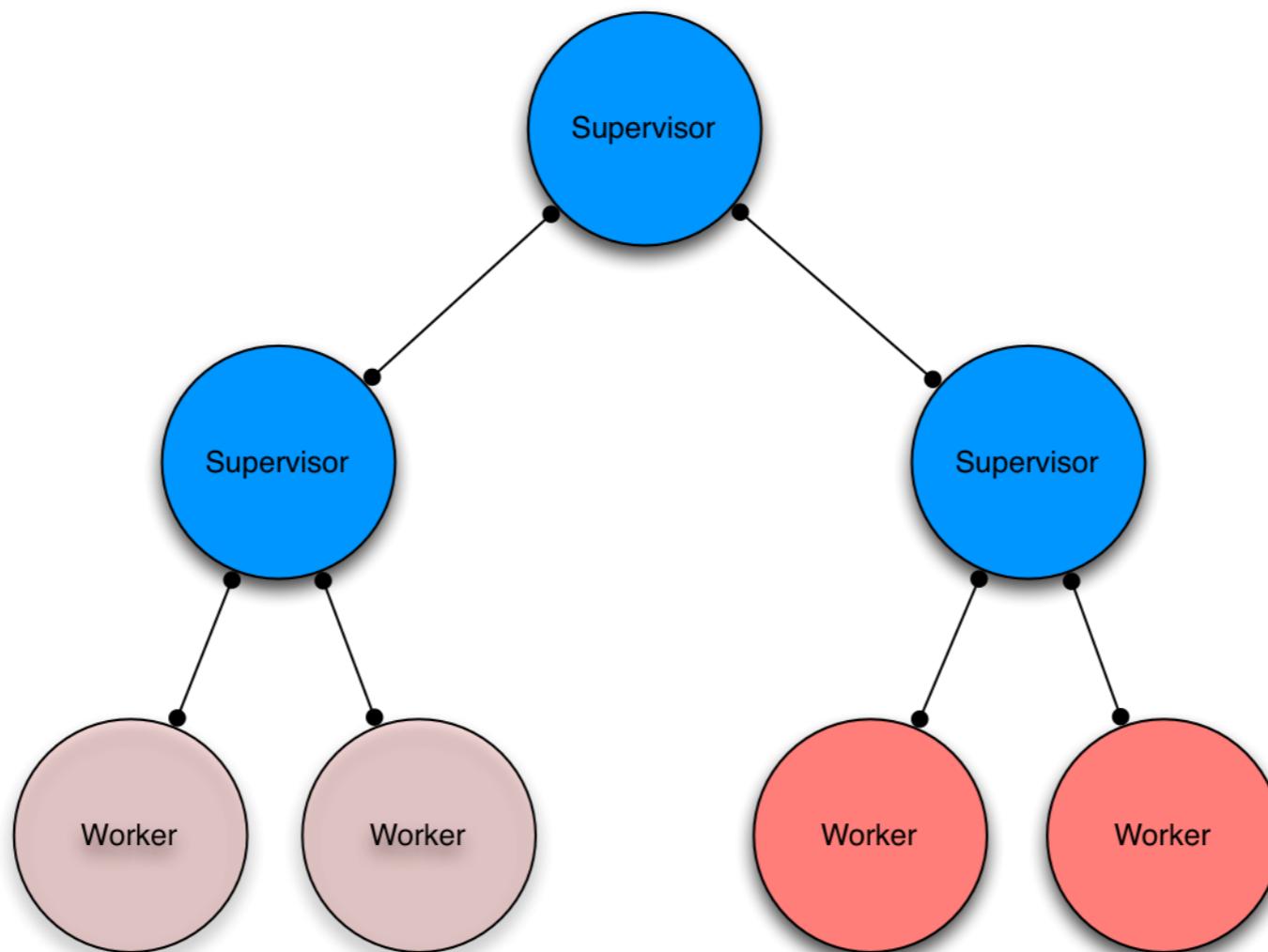
Supervision trees



Supervision trees



Supervision trees



Distribution

- > loosely coupled nodes
- > mostly transparent
- > TCP/IP based

OTP

- > helps creating:
 - > servers
 - > finite state machines
 - > event handler
 - > supervisors
 - > releases and upgrades

Hot code loading

- > module is unit of code handling
- > exists in two variants: old and current
- > controlled take over

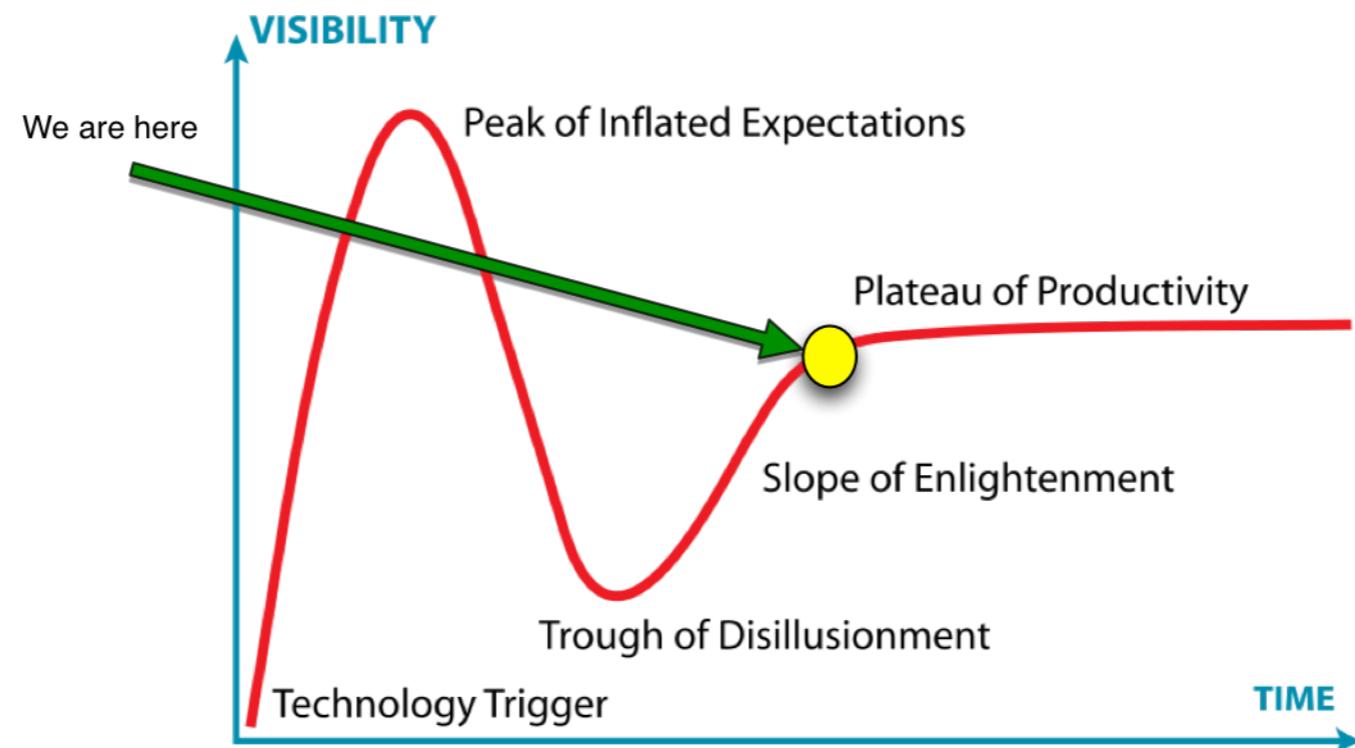
Instrumentation

- > can trace almost everything:
process events, send & receive
messages, function calls
- > process introspection:
memory, mailbox, links, cur. function...
- > interactive shell
- > SNMP based OAM

Summary

- > everything you need for building highly concurrent, distributed, robust systems
- > but not well suited for number crunching or maximum performance requirements

Where are we now?



Microservices & Erlang/OTP: how they fit together

How they fit together

- > Erlang / OTP has everything you need to build production-ready Microservices

How they fit together

- > fault tolerance / resilience
- > async communication is the default
- > amazing monitoring capabilities
- > tools for upgrading / downgrading running systems

A dark background featuring a person playing a drum set. Blue and white confetti is falling from above, creating a festive atmosphere. The text is overlaid on this image.

Erlang / OTP & Microservices

=

Insanely great!

Thank you!

- > Questions ?
- > Comments ?

Christoph Iserlohn

christoph.iserlohn@innoq.com