FRANCESCO CESARINI

presents

OTP

Francesco Cesarini
Erlang Solutions

@FrancescoC
francesco@erlang-solutions.com
www.erlang-solutions.com

What Is Scalability?

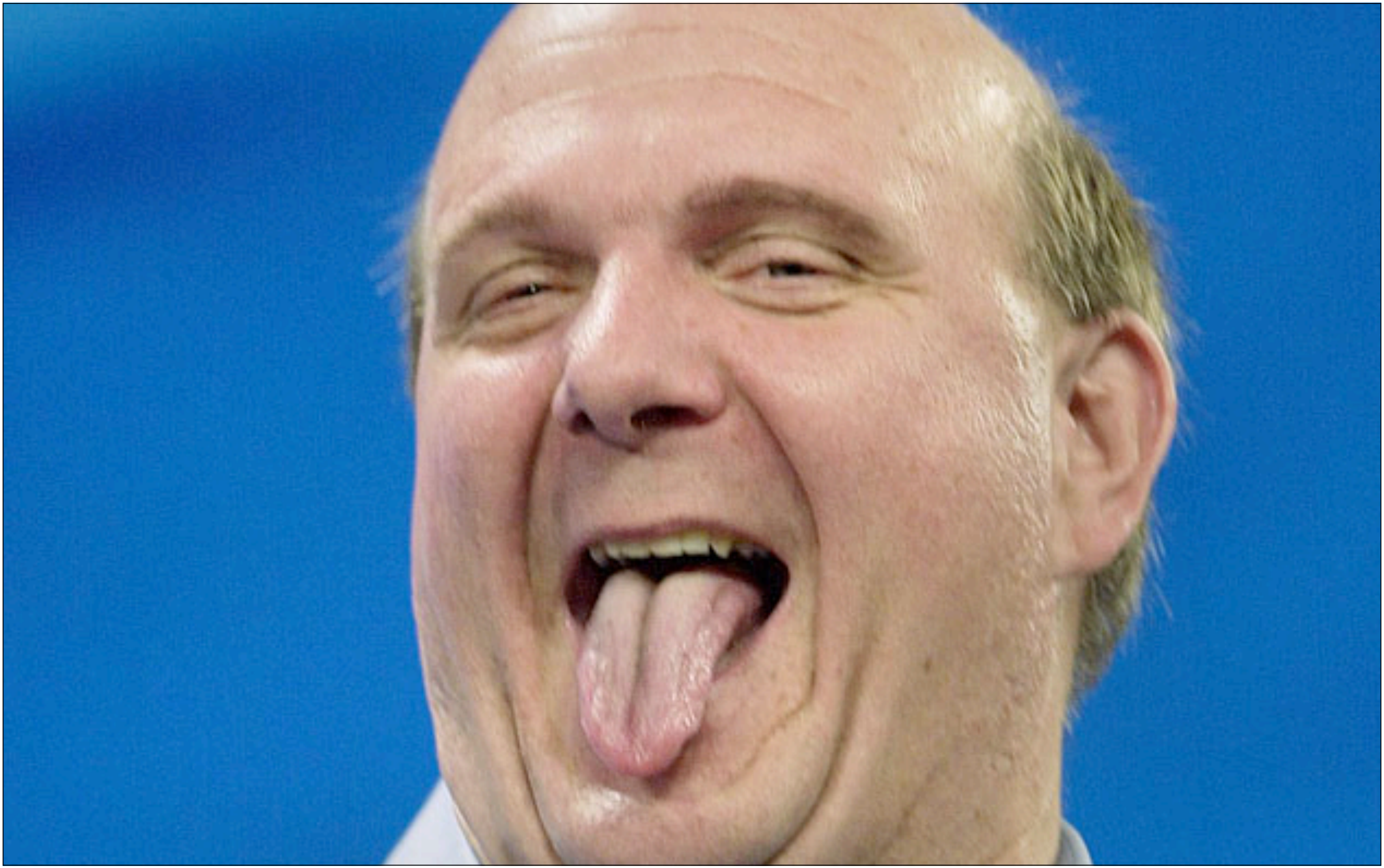
What Is (massive) Concurrency?


NOKIA

What Is High Availability?

What Is Fault Tolerance?

What Is Distribution Transparency?



Do you need a distributed system? Do you need a scalable system? Do you need a reliable system? Do you need a fault-tolerant system? Do you need a massively concurrent system? Do you need a distributed system? Do you need a scalable

YES, PLEASE!!!

system? Do you need a reliable system? Do you need a fault-tolerant system? Do distributed system? Do you need a scalable system? Do you need a reliable system? Do you need a fault-tolerant system? Do you need a massively

TO THE RESCUE

WHAT IS ERLANG

- Open source
- Concurrency-oriented
- Lightweight processes
- Asynchronous message passing
- Share-nothing model
- Process linking / monitoring
- Supervision trees and recovery strategies
- Transparent distribution model
- Soft-real time
- Let-it-fail philosophy
- Hot-code upgrades

WELL, IN FACT YOU NEED MORE.

ERLANG IS JUST
A PROGRAMMING LANGUAGE.

YOU NEED ARCHITECTURE PATTERNS.

YOU NEED MIDDLEWARE.

YOU NEED LIBRARIES.

YOU NEED TOOLS.

You need OTP.

What is Middleware?

Design Patterns

Fault Tolerance

Distribution

Upgrades

Packaging

MIDDLEWARE

What are Libraries?

Storage

O&M

Interfaces

Communication

LIBRARIES

What Tools?

OTP TOOLS

- Development
- Test Frameworks
- Release & Deployment
- Debugging & Monitoring

OPEN SOURCE

OTP IS

PART OF THE ERLANG DISTRIBUTION

OTP

Less Code

Less Bugs

More Solid Code

More Tested Code

More Free Time

Servers

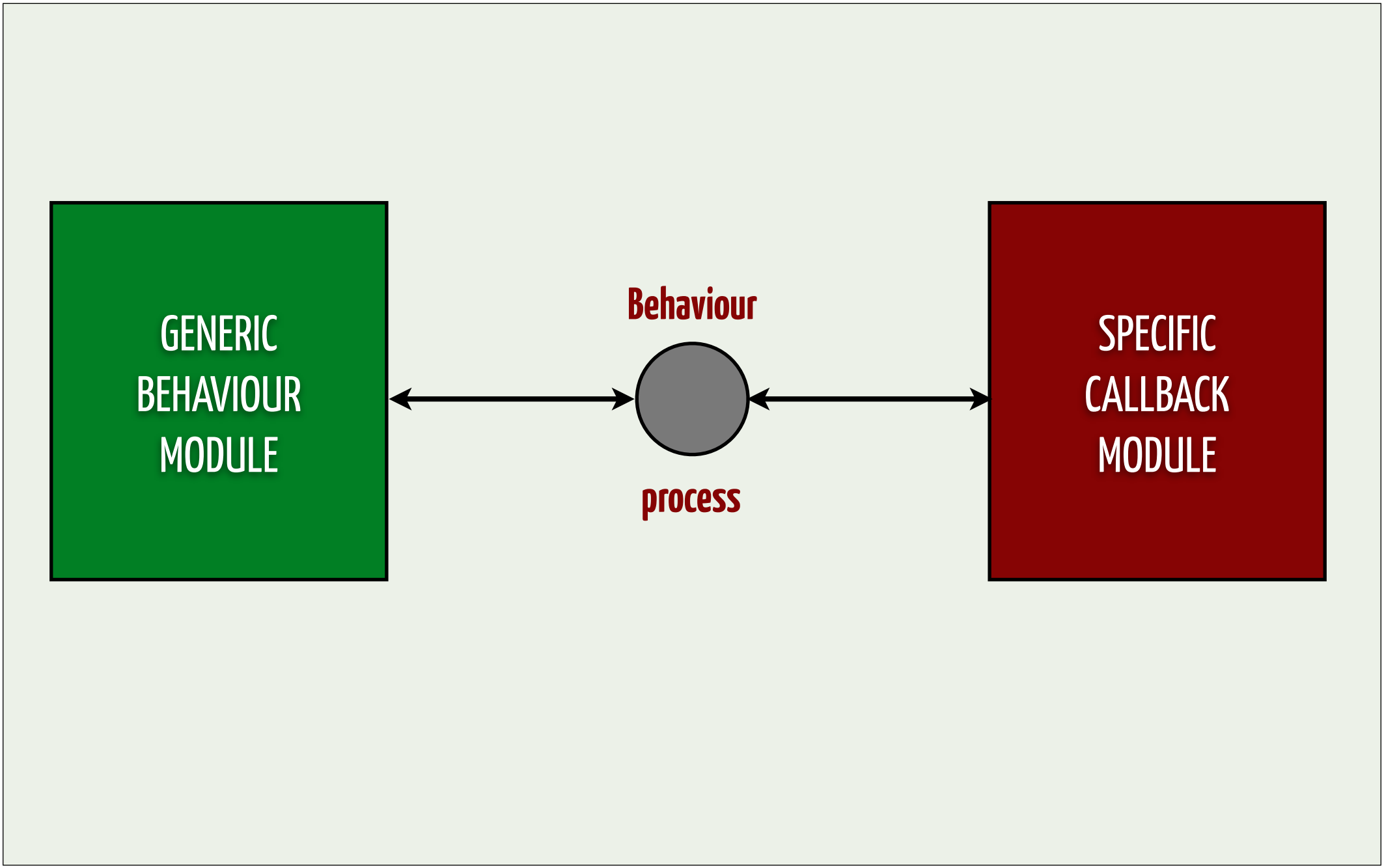
Finite State Machines

Event Handlers

Supervisors

Applications

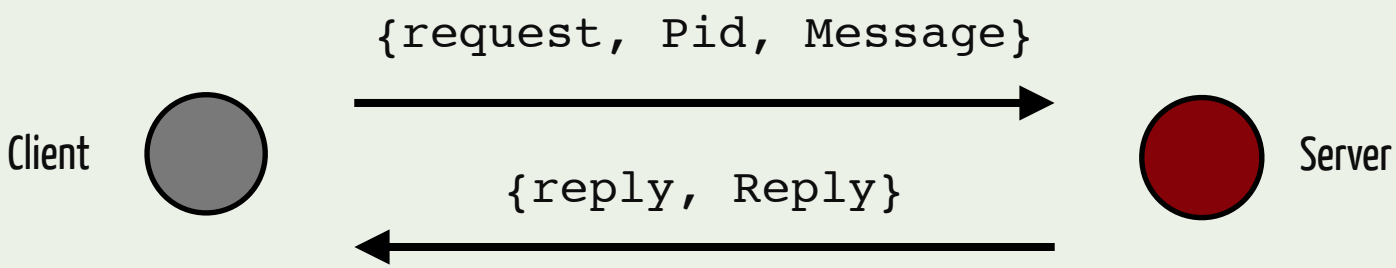
Behaviours


GENERIC
BEHAVIOUR
MODULE
Behaviour
process
SPECIFIC
CALLBACK
MODULE

OTP

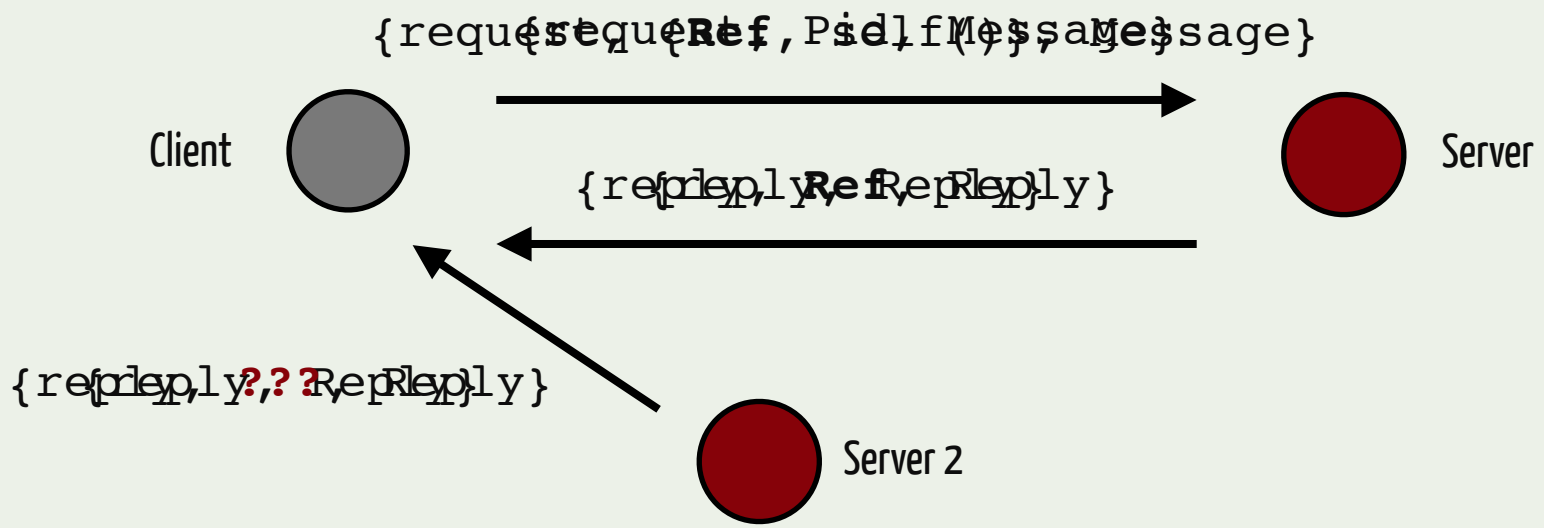
Less Code
Less Bugs
More Solid Code
More Tested Code
More Free Time

Servers
Finite State Machines
Event Handlers
Supervisors
Applications

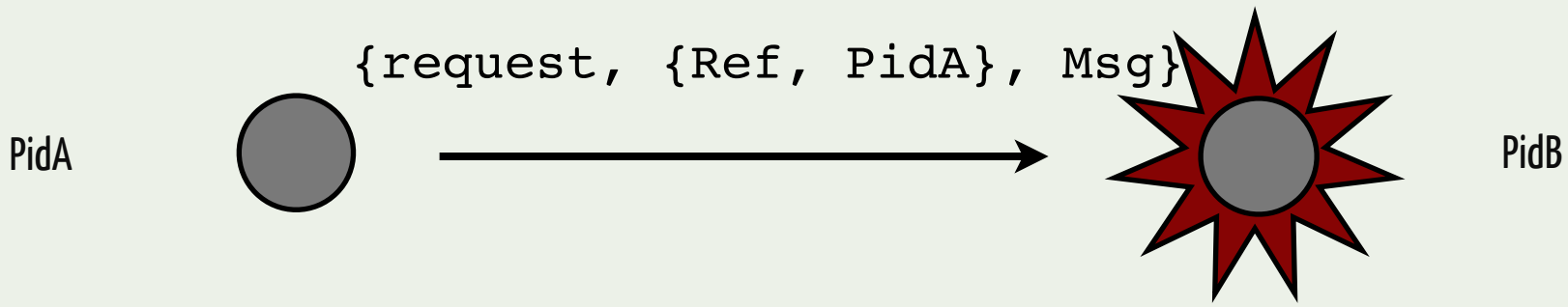



```
call(Name, Message) ->
    Name ! {request, self(), Message},
    receive
        {reply, Reply} -> Reply
    end.
```

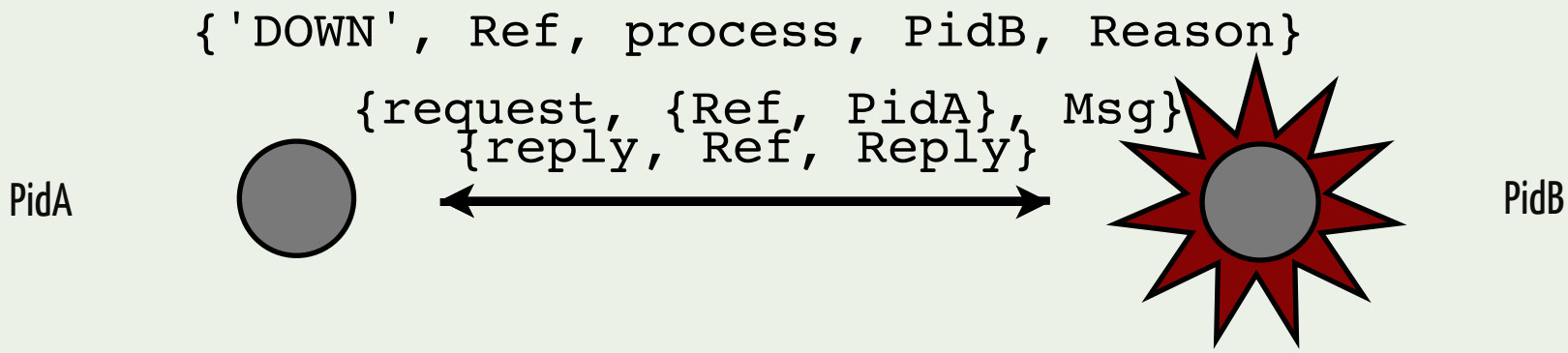
```
reply(Pid, Reply) ->
    Pid ! {reply, Reply}.
```




```
call(Name, Msg) ->
    Ref = make_ref(),
    Name ! {request, {Ref, self()}, Msg},
    receive {reply, Ref, Reply} -> Reply end.

reply({Ref, Pid}, Reply) ->
    Pid ! {reply, Ref, Reply}.
```




```
call(Name, Msg) ->
    Ref = erlang:monitor(process, Name),
    Name ! {request, {Ref, self()}, Msg},
    receive
      {reply, Ref, Reply} ->
        erlang:demonitor(Ref),
        Reply;
      {'DOWN', Ref, process, _Name, _Reason} ->
        {error, no_proc}
    end.
```




```
call(Name, Msg) ->
    Ref = erlang:monitor(process, Name),
    Name ! {request, {Ref, self()}, Msg},
    receive
      {reply, Ref, Reply} ->
        erlang:demonitor(Ref, [flush]),
        Reply;
      {'DOWN', Ref, process, _Name, _Reason} ->
        {error, no_proc}
    end.
```

BEHAVIOURS

TIMEOUTS

DEADLOCKS

TRACING

MONITORING

DISTRIBUTION

Let It Fail

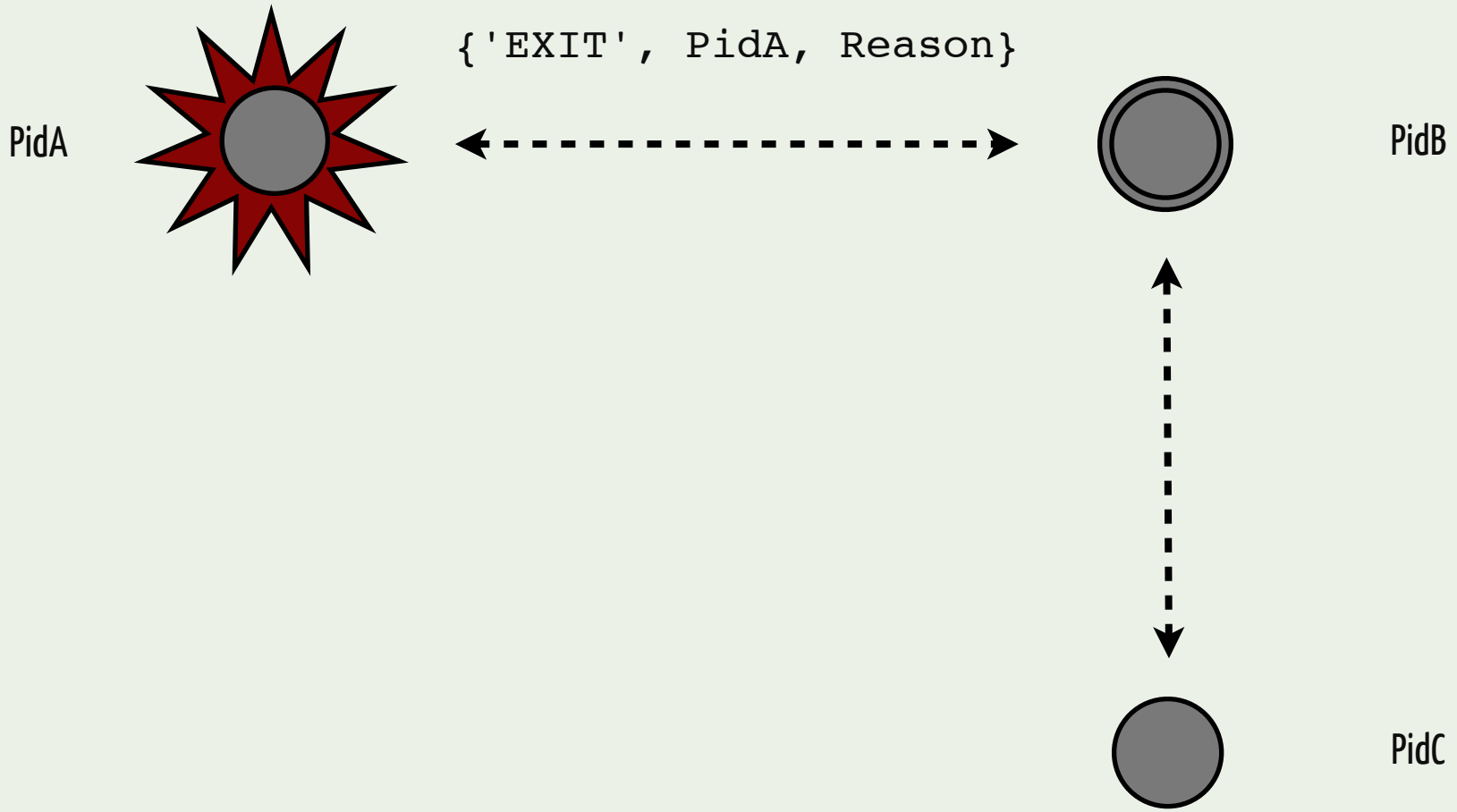
```
convert(Day) ->
  case Day of
      monday    -> 1;
      tuesday   -> 2;
      wednesday -> 3;
      thursday  -> 4;
      friday    -> 5;
      saturday  -> 6;
      sunday    -> 7;
      Other ->
          {error, unknown_day}
  end.
```

Let It Fail

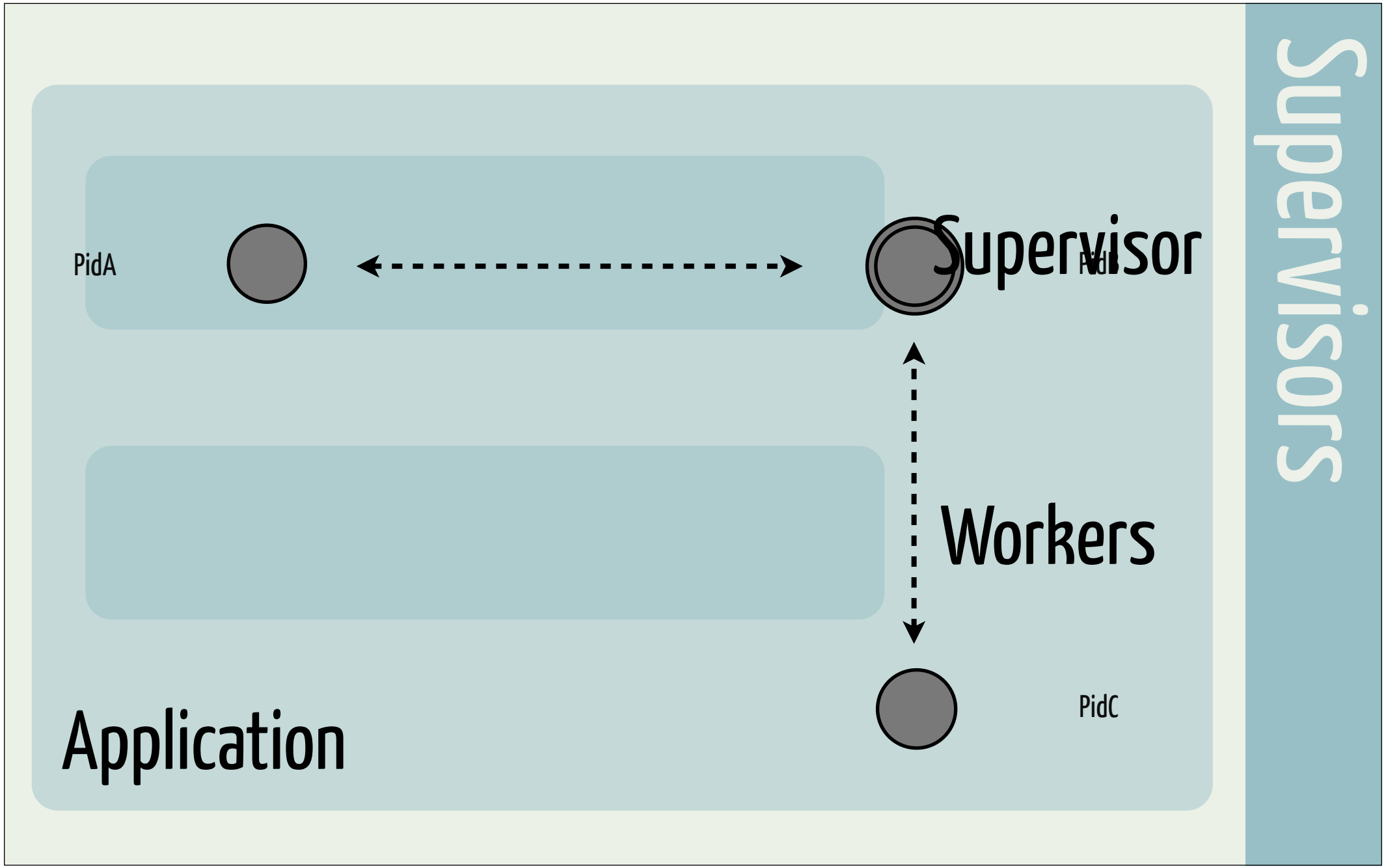
```
convert(Day) ->
  case Day of
      monday    -> 1;
      tuesday   -> 2;
      wednesday -> 3;
      thursday  -> 4;
      friday    -> 5;
      saturday  -> 6;
      sunday    -> 7


  end.
```

Isolate The Error!

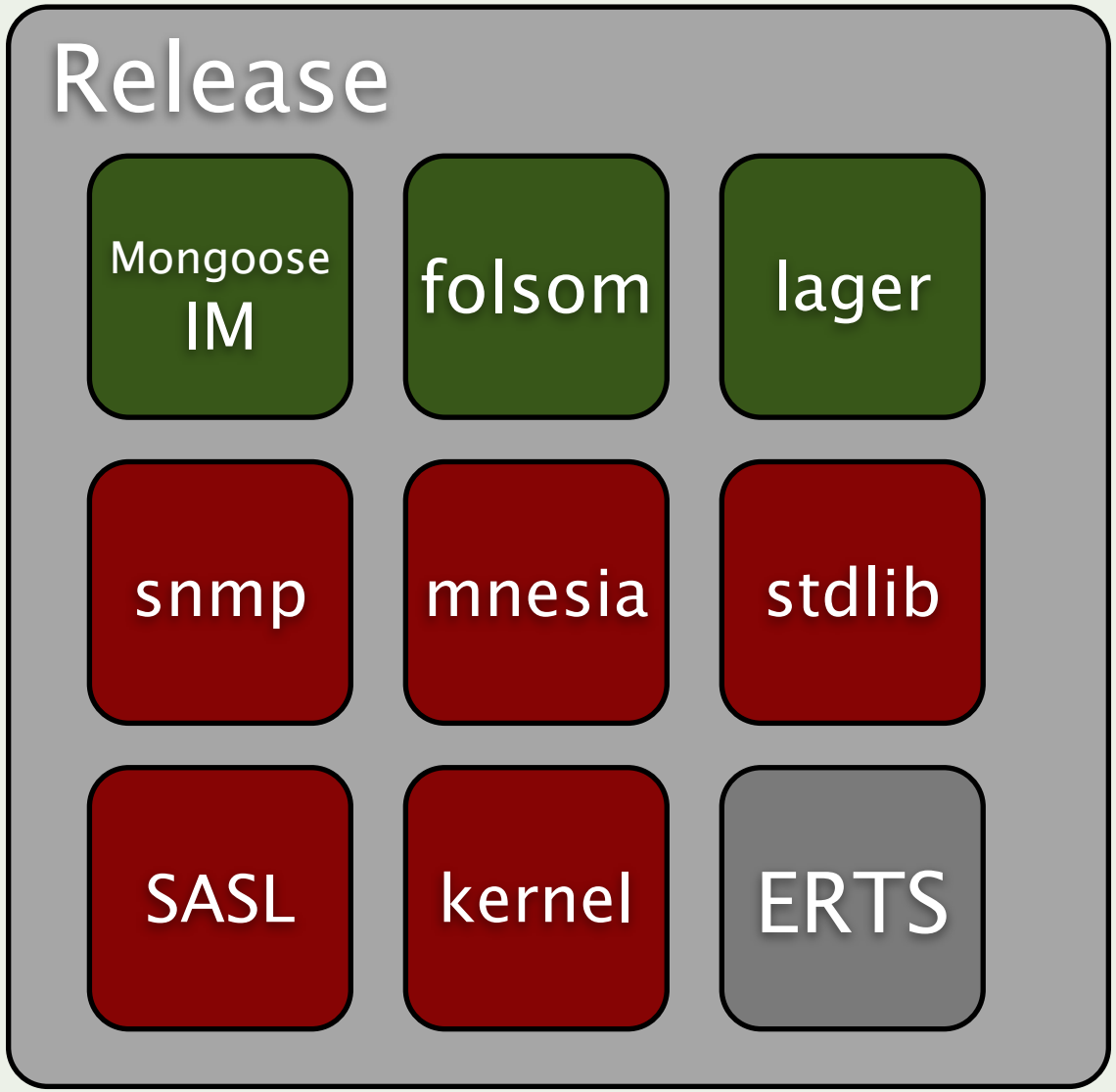
Trapping an Exit Signal



Trap Exit


Supervisors
PidA
Supervisor
Workers
PidC
Application

Releases


Release
Mongoose IM
folsom
lager
snmp
mnesia
stdlib
SASL
kernel
ERTS

Conclusions

Do you need a distributed system? Do you need a scalable system? Do you need a reliable system? Do you need a fault-tolerant system? Do you need a massively concurrent system? Do you need a distributed system? Do you need a scalable

USE ERLANG

system? Do you need a reliable system? Do you need a fault-tolerant system? Do you need a distributed system? Do you need a scalable system? Do you need a reliable system? Do you need a fault-tolerant system? Do you need a massively

Do you need a distributed system? Do you need a scalable system? Do you need a reliable system? Do you need a fault-tolerant system? Do you need a massively concurrent system? Do you need a distributed system? Do you need a scalable

USE ERLANG/OTP

system? Do you need a reliable system? Do you need a fault-tolerant system? Do you need a distributed system? Do you need a scalable system? Do you need a reliable system? Do you need a fault-tolerant system? Do you need a massively

Questions?

@francescoC