Learn you some Erlang

UvA Summer School
Amsterdam

Arjan Scherpenisse
5 juli 2013

arjan@miraclethings.nl

MiracleThings ★

Agenda

- Intro
- Language essentials
- Concurrency / failure
- OTP
- Hands-on!

0:54 / 5:50

The Ghetto

gar1t · 11 videos

Subscribe 752

3,554

👍 61 👎 0

What is Erlang?

History

- Created in '86

- Open-sourced in '98

- "Programming Erlang" published in '07

- Taking off in popularity (3 more books on the way)

Erlang Philosophy

- The world is concurrent
- Things in the world dont share data
- Things communicate with messages
- Things fail

- Joe Armstrong

The Shell (super important)

- Single most important tool for Erlang developers!
- Use to experiment, learn, prove/disprove
- Important to know:
  - `erl` starts the shell!
  - `ctrl-c, ctrl-c` stops the shell abruptly!
  - `q().` shuts down cleanly
  - Expressions always end in a period!

Read

Eval

Print
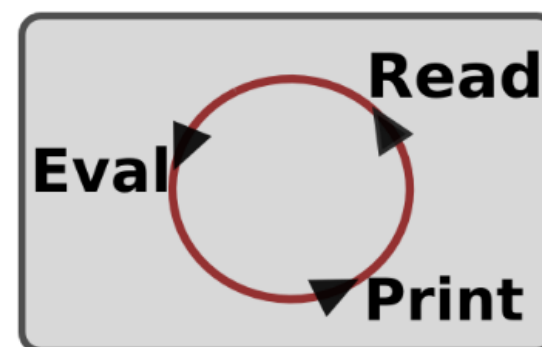
Language essentials

She's got the look

```erlang
-module(math_o_matics).
-export([square/1]).

square(X) ->
    X * X.
```

Data types

- Integer / float
- Atoms
- Tuples
- Lists
- Binaries

… so where are strings?

Datatypes (cont)

- Functions
- Pids
- Ports

MiracleThings ★

The "fun" in functional

- Anonymous functions

- Used for higher-order programming

```
Square = fun(X) -> X * X end.

Cube = fun(X) -> Square(X) * X end.
```

List Comprehensions

- Takes an expression and a set of qualifiers and returns another list (like Python's)

- Looks like: [X || Q1, Q2, ... Qn]

```
qsort([]) -> [];
qsort([Pivot|T]) ->
  qsort([X || X <- T, X < Pivot])
++ [Pivot] ++
  qsort([X || X <- T, X >= Pivot]).
```

Recursion (important)

- Function directly or indirectly calling itself

- It's how you iterate!

- Tail optimization enables long running loops (eliminates unbounded stack growth)

MiracleThings ★

Guards

- Simple tests against a pattern matching

- Can make code more concise

```
max(X, Y) when X > Y -> X;
max(X, Y) -> Y.
```

Side-Effects

- Erlang is not pure-functional language
- Side effects:
    - *Messages* - Pid ! Msg – send message to process Pid
    - *Signals* - i.e. your linked process is dead
    - *Exceptions*
    - *I/O*
    - *Process Dictionary* - similar to TLS (thread local storage)
    - *ETS / DETS* - fast in-memory / persistent lookup tables
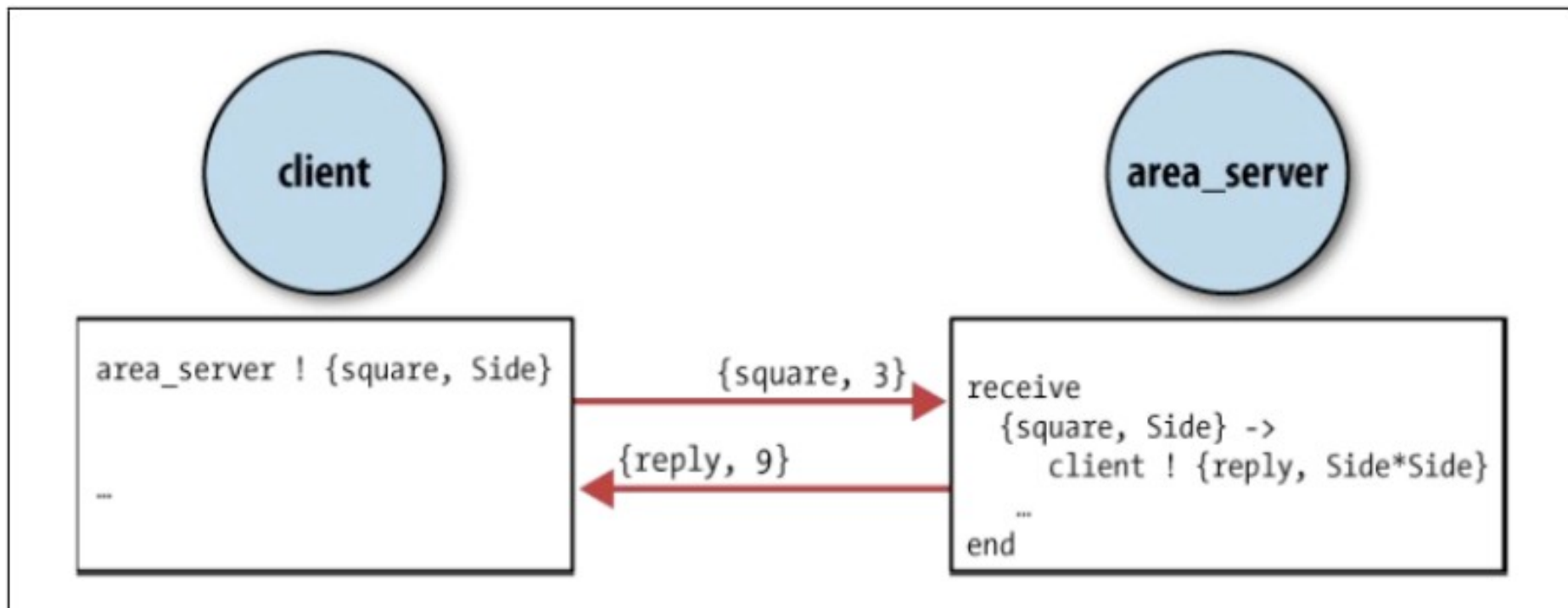   - *Mnesia* - distributed realtime database (STM)

Concurrency / failure

Concurrency

- **spawn, spawn_link**
- **!, receive**

- Concurrency properties
  - Lightweight process model in Erlang VM
    - (not OS processes or threads)
  - No data sharing (inherited from functional nature)
  - Built in distributed system support
    - Safe and conditional messaging
    - Process monitoring and failure notification

Concurrency and Messaging

- Model: create a process for every independent activity
  - Processes are lightweight
- Communicate between processes using messages
- (Crazy) Example to get in the Erlang mindset


```
area_server ! {square, Side}        {square, 3}      receive
                                                        {square, Side} ->
                                        {reply, 9}        client ! {reply, Side*Side}
...                                                     ...
                                                      end
```

The Actor Model

- Actors communicate through messages
- Actors make local decisions as to handle messages
  - Send more messages
  - Create more actors
  - Decide how to respond to future messages (change local state)
- Actor model prohibits shared state and is a nice formalism for parallel computing

- An example "worker pool" script

worker.erl
https://gist.github.com/arjan/4283899

LET IT CRASH

MiracleThings ★

Process Error Handling

- Features underlying built-in fault tolerance
- Ethos
  - "Let it crash and let someone else deal with it"
  - "Crash early"   (Crash often?)
- Process linking and monitoring
  - Notification if processes fail or are unavailable/disconnected
  - Symmetric (linked) and assymetric (monitor)
  - Exit signals (notification messages when processes terminate)

MiracleThings ★

link/1 and spawnlink/3

- link/1 connects to an existing process
- spawnlink/3 spawns a process linked to current proc.
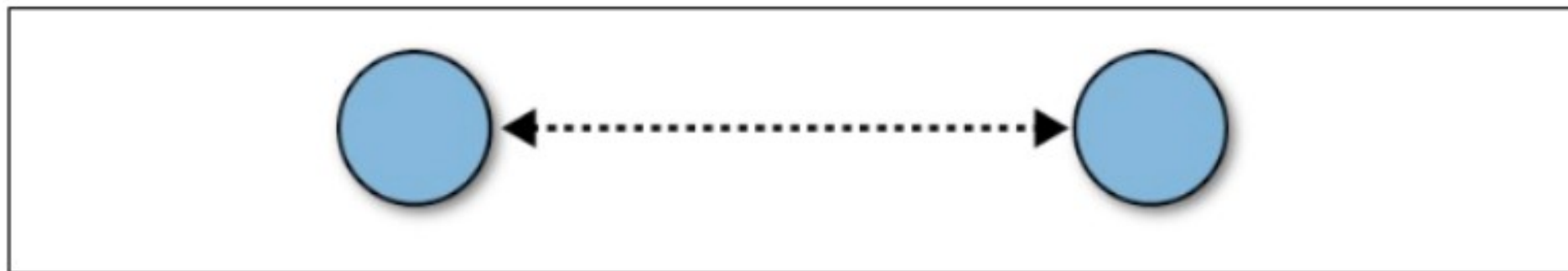 - This is atomic and almost always preferable to spawn then link

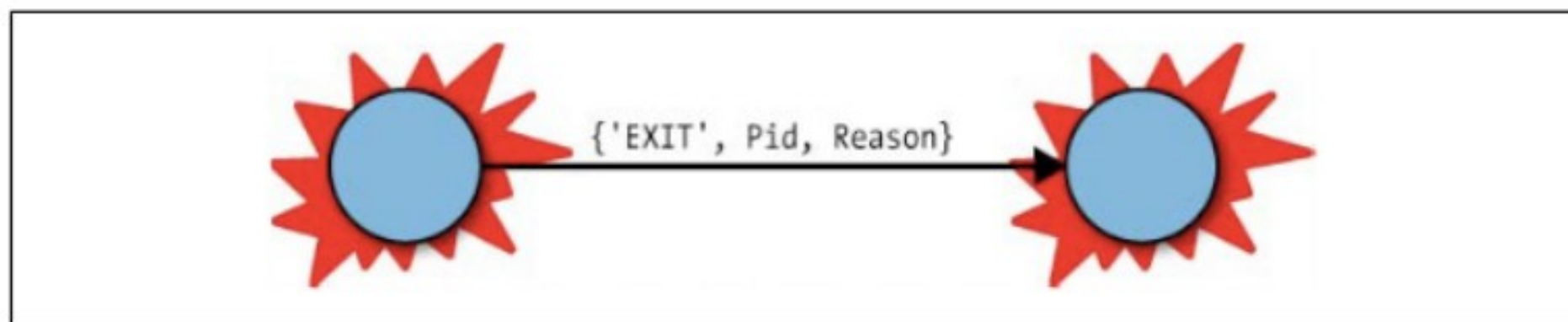

Figure 6-1. Linked processes

Figure 6-2. Exit signals

Error Propagation

- link/1 is a good technique to propagate errors
  - To fail all related process (the Erlang way) and reveal errors
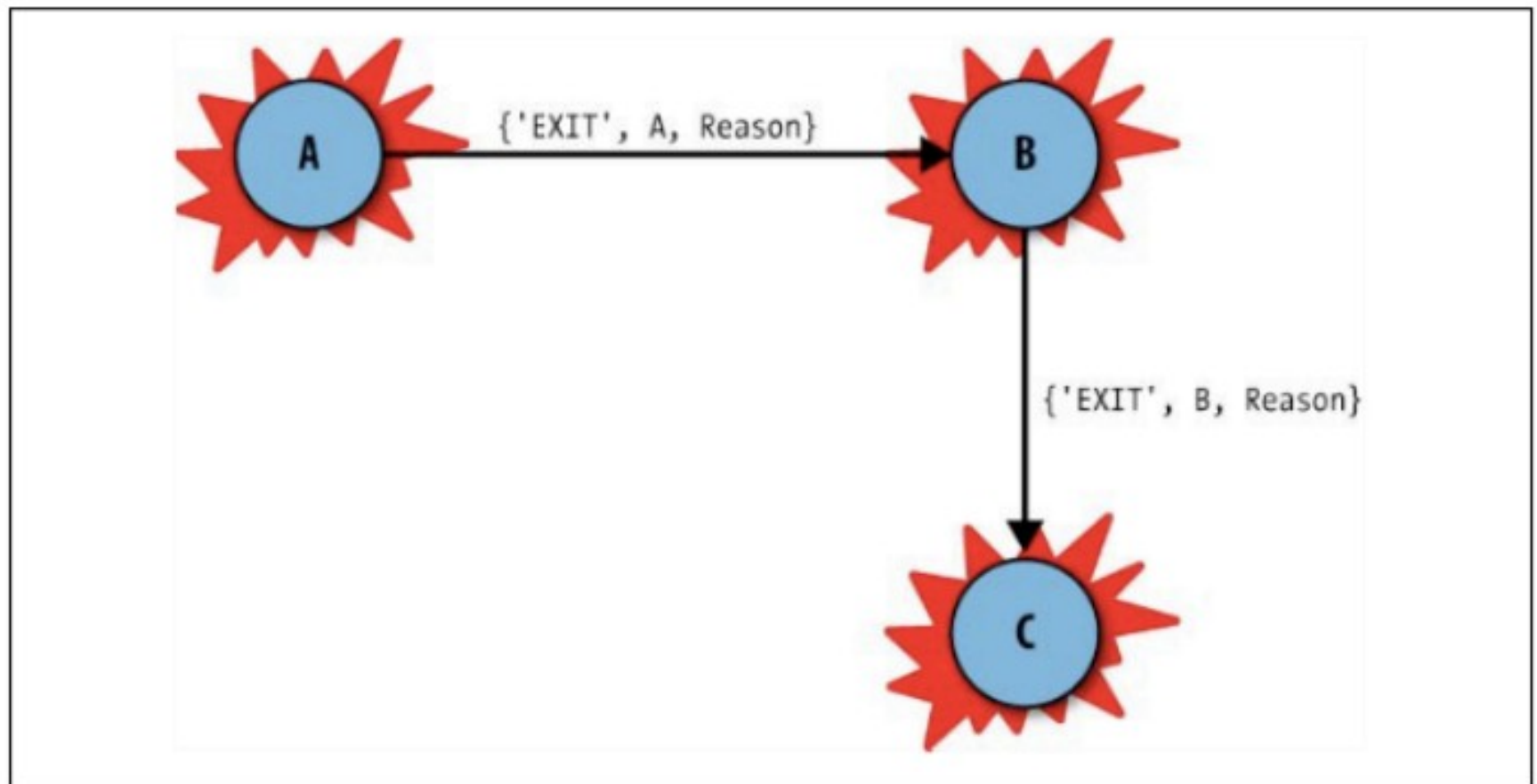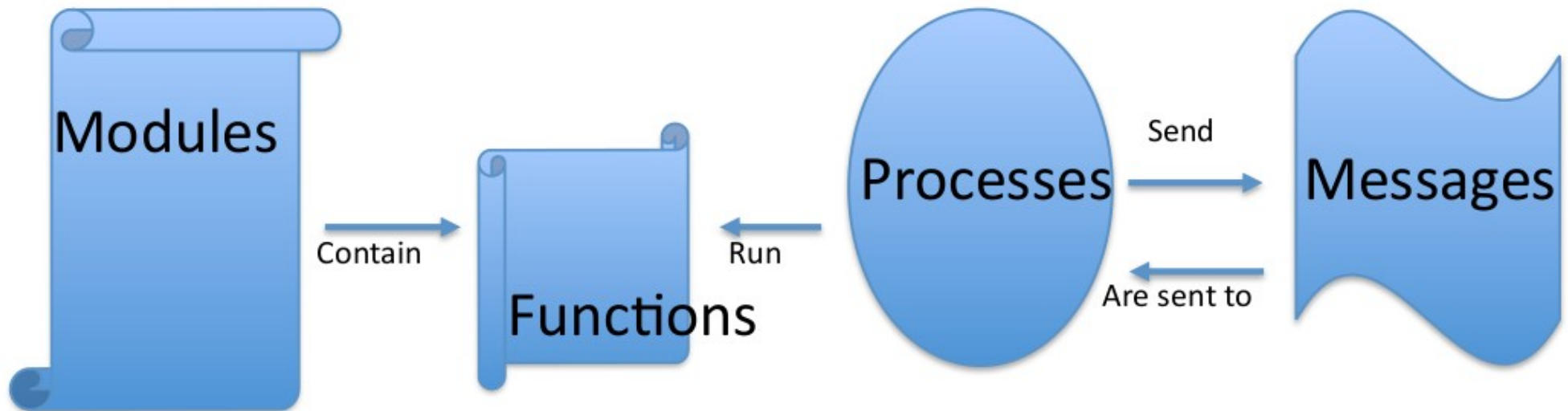 - To have a monitor restart

Figure 6-3. Propagation of exit signals

Fundamental pieces

OTP: Open Telecom Platform

OTP=Open Telecom Platform

- Library of implemented application frameworks
  - Users customize behavior by callback functions

- Client/server
  - OTP handles timeouts, safegaurds, message protocol, process registration

- Process monitoring
  - OTP handles fault detection, automatic restart, start and shutdown, managing process state

- Effective code reuse!

OTP

- application
  - Contains independent code, multiple applications per Erlang *node*

- supervisor
  - Supervises worker processes and supervisors

- gen_server
  - Basic work unit

Erlang is a good fit for:

- Irregular concurrency:
  - Task-level
  - Fine-grained parallelism
- Network servers
- Distributed systems
- Middleware:
  - Parallel databases
  - Message Queue servers
- Soft Realtime / Embedded applications
- Monitoring, control and testing tools

Not so good fit for:

- Concurrency for synchronized parallel execution
  - Data Parallelism
- Floating-point intensive code (HPC)
- Text Processing / Unicode
  - Unicode support now much better
- Traditional GUI (supports Tk and wxWidgets)
- Hard Realtime applications
- Extensive interop with other languages/VMs
  - improvement here with NIFs and Erjang - Erlang on JVM

What *I* mainly use it for

- Full-stack web development
  - http://zotonic.com/
- HA backend systems

Preventing stuff like this →
:-)

www.cwi.nl

Drupal

Site off-line

The site is currently not available due to technical problems. Please try again later.
Thank you for your understanding.

If you are the maintainer of this site, please check your database settings in the settings.php file
and ensure that your hosting provider's database server is running. For more help, see the
handbook, or contact your hosting provider.

Tools

- IDE:
  - Emacs, Distel, Plugins for Eclipse and Netbeans
- Testing:
  - e-unit, proper, meck, triq, QuickCheck (paid)
- Static code analyzer
  - Dialyzer
- Debug and trace
  - Built-in in VM
- Build:
  - emake, rebar
- Package Managers:
  - rebar, agner, sinan, CEAN

Books
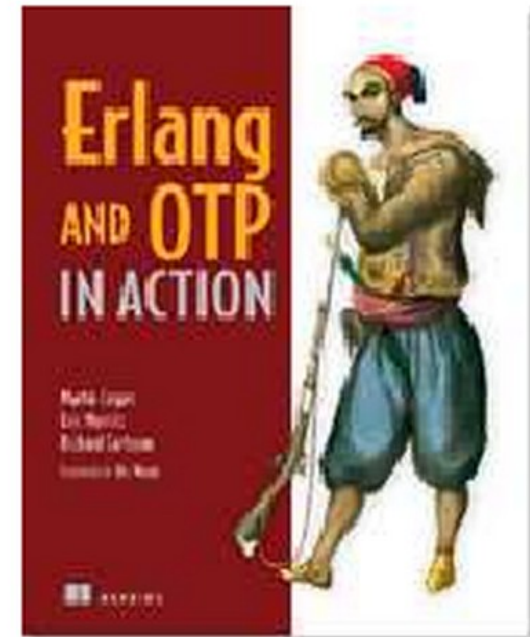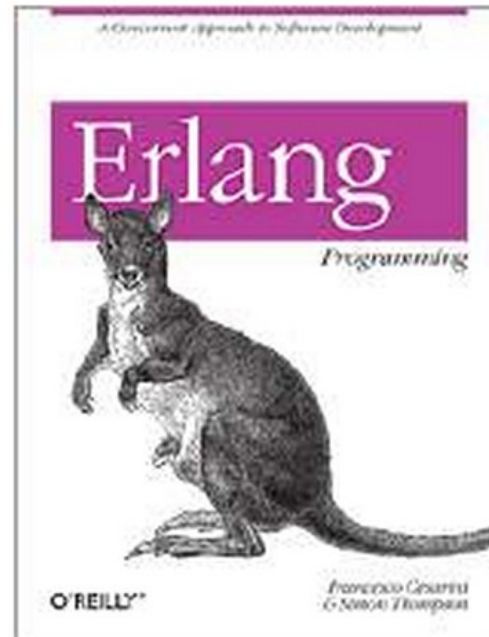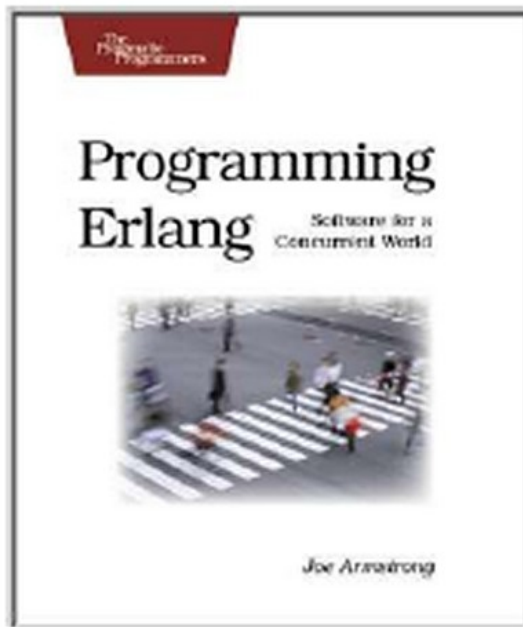
Sources of Inspiration

- Erlang - the ghetto

- Intro to erlang (slideshare)

- Erlang - message passing concurrency (slideshare)

- Erlang concepts (slideshare)

- Concurrency and Paralellism in Erlang (hopkins univ)

Getting practical

Assignment!

- Building a simple chat system using Erlang's process model

- Download the assignment PDF

- Best submission wins a t-shirt!

MiracleThings ★

Reclameblok!!

- Try Zotonic – the Erlang Web Framework and CMS

- Erlang Camp Amsterdam

  - 30 & 31 augustus

  - Slechts 75 euro!

MiracleThings ★

Thanks!

Arjan Scherpenisse
arjan@miraclethings.nl

http://twitter.com/acscherp
http://miraclethings.nl/

MiracleThings ★