

# Erjang — a JVM-based Erlang VM

Kresten Krab Thorup, Trifork

London 2010  
Tutorials: March 8-9  
Conference: March 10-12

www.qconlondon.com

INTERNATIONAL  
SOFTWARE DEVELOPMENT  
CONFERENCE

QCon London 2010

Speakers  
Schedule  
Tutorials  
Tracks  
Social Events

Exhibition  
Sponsors

Registration  
Volunteers

Venue  
Travel  
Hotels

User Groups @ QCon London

TRIFORK.  
InfoQ

About QCon  
Contact  
Archives  
Future events

**QCon is coming back to London**  
Tutorials: March 8-9, 2010  
Conference: March 10-12, 2010  
The fourth annual London enterprise software development conference designed for team leads, architects and project management is back! There is no other event in the UK with similar opportunities for learning, networking, and tracking innovation occurring in the Java, .NET, Ruby, Scala, Agile, and architecture

**Tracks for QCon London**

**Architectures You've Never Wondered About** - Features Facebook, Skype, Sky.com, Swift, League of Legends

**Scala Evolution** - The changing face of Agile and future directions

**How do you test that?** - Difficult problems and state of the art in testing: functional, performance, integration, and more.

**Aggregating on .NET** - Alpha gets go beyond .NET prescriptive .NET solutions, hear how.

**Cool Stuff with Java** - Practitioner examples of unusual cool problem-solutions in Java.

**Functional programming** - Everything you ever wanted to know about functional

Register before February 23 and save up to £187/€224  
Register now >>

**QCon Videos**

- 1 Rich Hickey - "Persistent Data Structures and Managed References"
- 2 Martin Fowler - "Three Years of Real-World Ruby"
- 3 Tony Hoare - "Null References: The Billion Dollar Mistake"
- 4 James Bond - "Frugate: Real-World Scala"
- 5 Aditya Agarwal - "Facebook: Science and the Social Graph"

## My Mission #1

- Trifork is a "successful" Java consultancy/solution provider; but I see challenges looming on the horizon...
- **Increase in concurrency, distribution, integration, unreliability**  
⇒ coordination overhead

## My Mission #2

- Understand “Actors”, as a means to make such systems *understandable* and *manageable*; which exhibit coordination, concurrency, distribution, etc. as dominant elements.  
⇒ learn Erlang, and understand the folklore of this community, it’s patterns, and generic solutions.

TRIFORK.

## My Mission #3

- Describe these learnings in a generic fashion, making them accessible to the “main stream” programmer.  
⇒ Write the bestseller “*Actor Design Patterns*” on how to make concurrent systems comprehensible, so that we can reason and talk about them.

TRIFORK.

## My Mission #4

- Generate interesting and relevant content for our conferences, *QCon*, *JAOO* and *Yow!*  
⇒ Meet practitioners with relevant problems and experience, and make them talk about it at our conferences.

TRIFORK.

## Explorer in Erlang-land

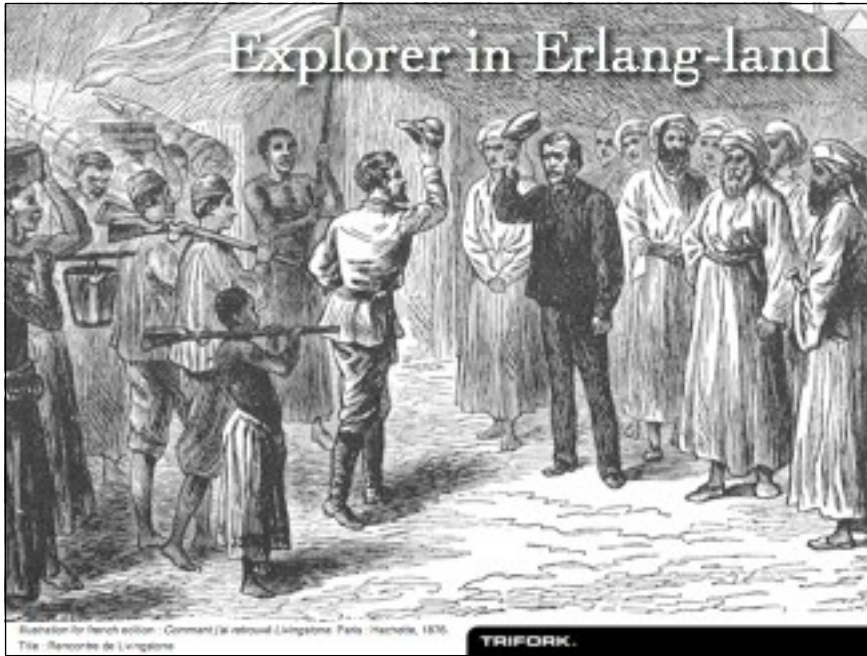


Illustration for French edition - Comment je réussis L'Algérie Paris - Hachette, 1876  
Titre - Récit de L'Algérie

TRIFORK.

## Java - the leap forward

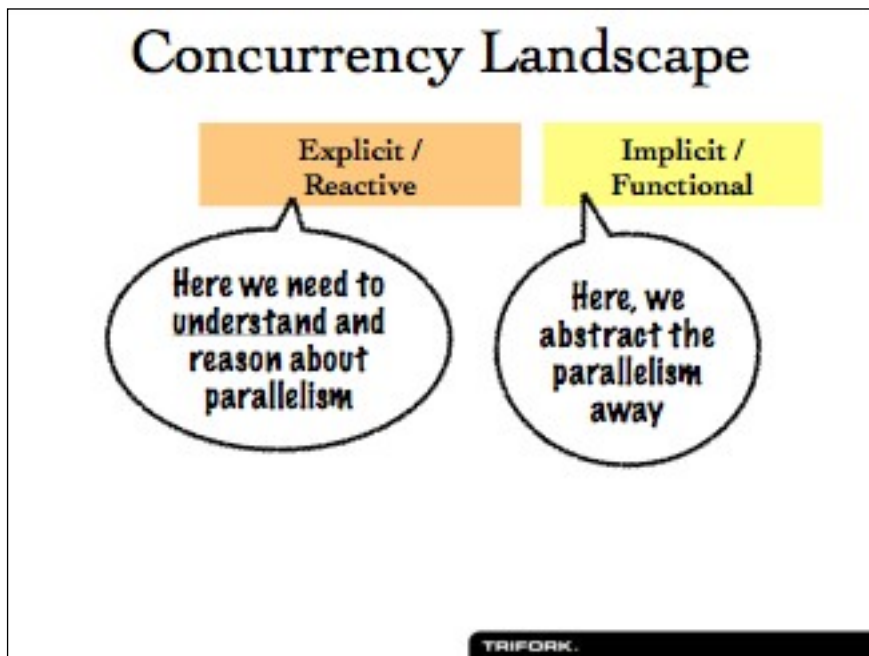
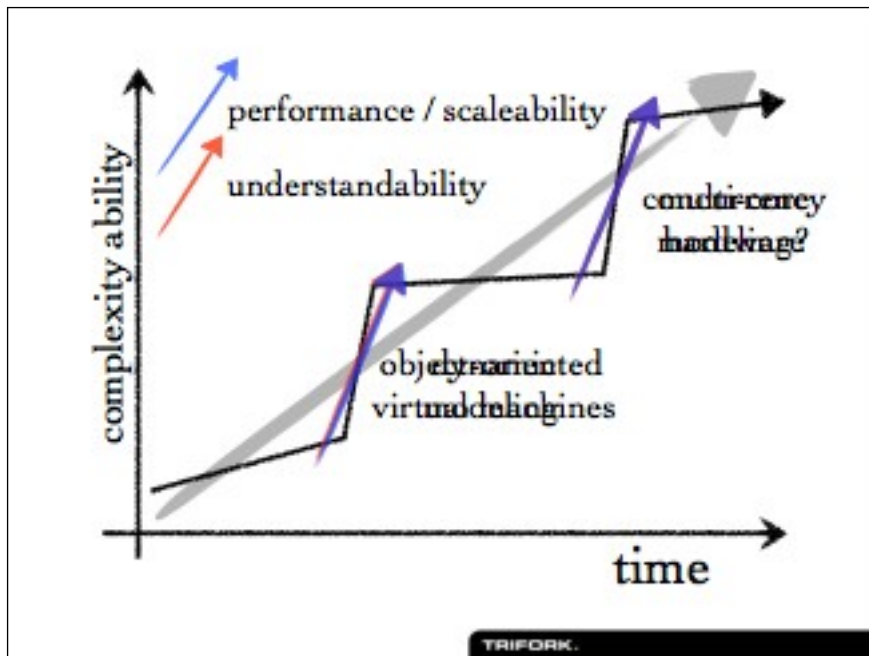
- **Simplifications in ...**
  - Error handling, Exceptions, Memory Management
  - Universal Platform
  - Large set of available Libraries

TRIFORK.

## The next leap...

- **Simplifications in the areas of ...**
  - Concurrency => Coordination
  - Distribution; Deployment
  - Fault tolerance

TRIFORK.



## Concurrency Landscape

	Explicit / Reactive	Implicit / Functional
Distributed	Telephone/Trading Systems Highly Available Storage  Erlang / CORBA Message Middleware	Search Engine Indexing  Model Simulations, Weather Forecasts
Local	GUI-applications Control Systems  Threads	Google/Hadoop Map-Reduce Data-Parallelism

TRIFORK.

## Trend in “New” Concurrency Languages

- Clojure
- F#
- Scala + Actors
- Erlang
- Comprehending concurrent systems is difficult
- Mutable state is bad
- Make things simpler

TRIFORK.

## And then ... Erjang

- Run Erlang on a JVM
  - Like JRuby, Jython, etc.
- Runs off the beam files of a “vanilla” OTP distro
- Better integration / or transition from Java to Erlang - a better jinterface
- My means to learn Erlang :-)

TRIFORK.

## Deployment Scenarios

- Use Erjang where BEAM is not “welcome” or possible
  - WebSphere, Tomcat
  - AS/400, Mobile [JVM, but no BEAM]
  - For Testing, easier distribution?
- You tell me!

TRIFORK.

# Erjang - Challenges

- Ultra Light-Weight Processes
- ~~Real-time Behavior~~
- Tail-recursion
- Arbitrary Precision Numbers
- Pattern Matching
- Erlang Drivers
- JVM is type safe, urgh!

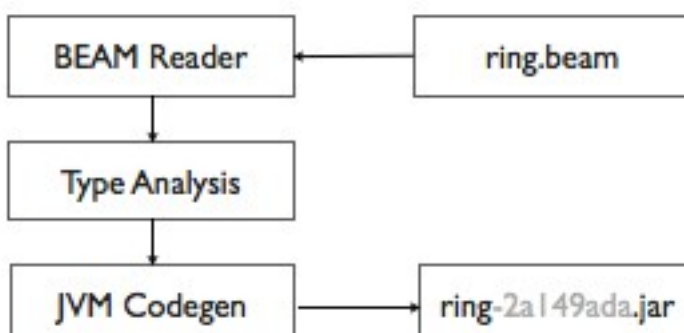
TRIFORK.

# BEAM files

- Erlang modules compile to byte code
  - .BEAM file
  - Module ring in ring.beam
  - Similar to "Java .class file"
- .beam files are read by an "error handler" when missing modules/functions are seen.

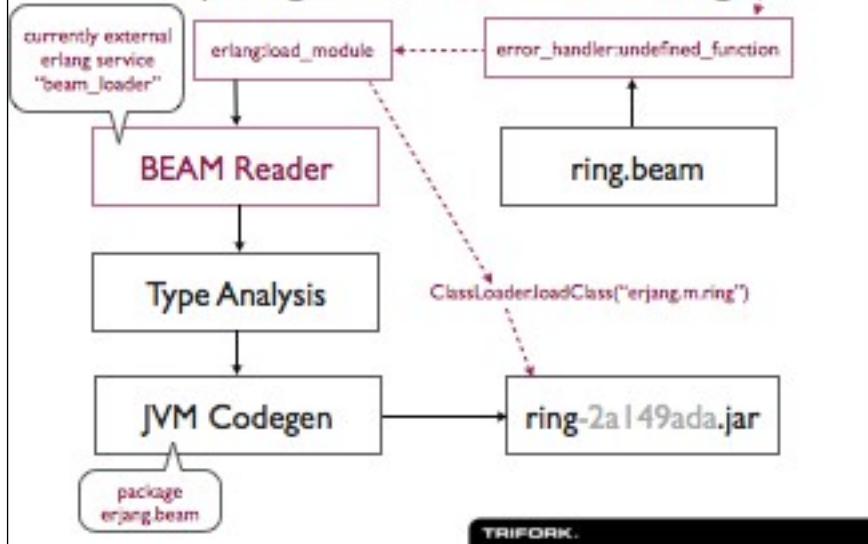
TRIFORK.

# Erjang Compiler



TRIFORK.

# Erjang BEAM Loading \*

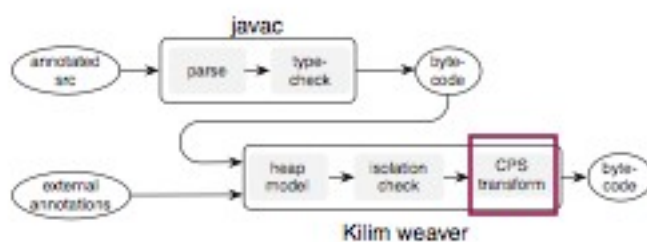


## Light-Weight Processes

- Threads don't cut it;
- Typical JVMs are limited to ~1000 threads
- Context switch for threads is very expensive
- Erjang uses *Kilim*, a separate project
- Run ~2-4 threads to utilize processors

TRIFORK.

## Kilim



# Kilim

- Rewrites code in methods declared 'throws kilim.Pausable' so that they can
- save call stack to "fiber data structure"
- and restore it later, continue execution...

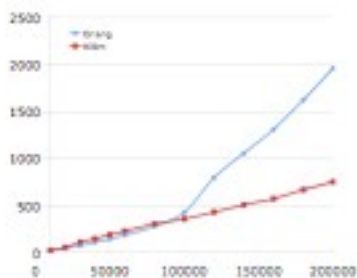
TRIFORK.

## Kilim Rewriting, in brief ...

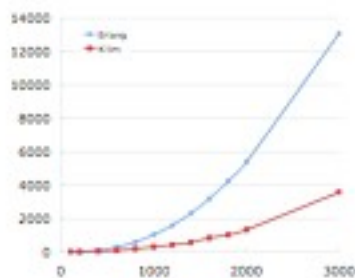
```
class Actor extends kilim.Task {  
  void execute() throws Pausable {  
  
    // also declared Pausable  
    mbox.get();  
  }  
}
```

```
class Actor extends kilim.Task {  
  void execute() { .. throw error ... }  
  void execute(Fiber f) throws Pausable {  
    switch (f.pc) {  
    case 0:  
      f.down();  
      mbox.get();  
      switch(f.up()) {  
      case SAVE_STATE:  
        case SAVE_NO_STATE:  
          .. f.stack.push(new State(...));  
      case NORMAL_NO_STATE:  
        case NORMAL_STATE:  
      }  
    }  
  }  
}
```

TRIFORK.



(a) Creation and Destruction



(b) Messaging

# Kilim

TRIFORK.

# Pattern Matching

```
ins_opts([H | T], T2) ->  
    ins_opts(T, ins_opt(H, T2));  
ins_opts([], T2) -> T2.
```

TRIFORK.

# The BEAM Code

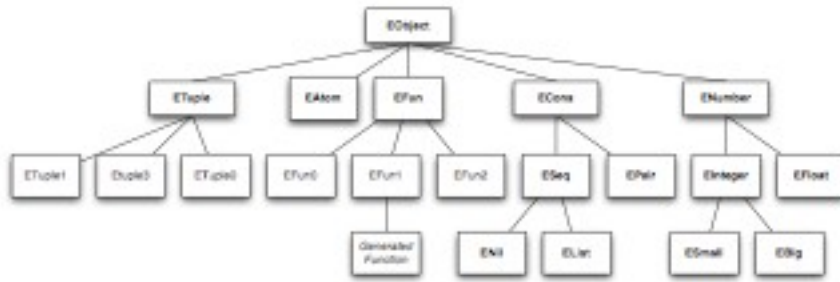
```
{function, ins_opts, {nargs,2}}.  
{label,264}.  
  {test,is_nonempty_list,{else,265},[{x,0}]}.  
  {get_list,{x,0},{x,0},{y,0}}.  
  {call,2,ins_opt}.  
  {move,{x,0},{x,1}}.  
  {move,{y,0},{x,0}}.  
  {call_last,2,ins_opts,1}.  
{label,265}.  
  {test,is_nil,{else,263},[{x,0}]}.  
  {move,{x,1},{x,0}}.  
  return.  
{label,263}.  
  {func_info,{atom,appmon_info},{atom,ins_opts},2}.
```

TRIFORK.

```
public static EObject ins_opts____2(EProc eproc, EObject arg1,  
EObject arg2)  
{  
  tail: while(true) {  
    ECons cons;  
    if((cons = arg1.test_nonempty_list()) != null) {  
      // extract list  
      EObject hd = cons.head();  
      EObject tl = cons.tail();  
      // call ins_opt/2  
      EObject tmp = ins_opt____2(eproc, hd, arg2);  
      // self-tail recursion  
      arg1 = tl;  
      arg2 = tmp;  
      continue tail;  
    } else if ((arg1.test_nil()) != null) {  
      return arg2;  
    }  
  }  
  throw ERT.func_info(atom_appmon_info, atom_ins_opts, 2);  
}
```

TRIFORK.

# Core Erjang Types



TRIFORK.

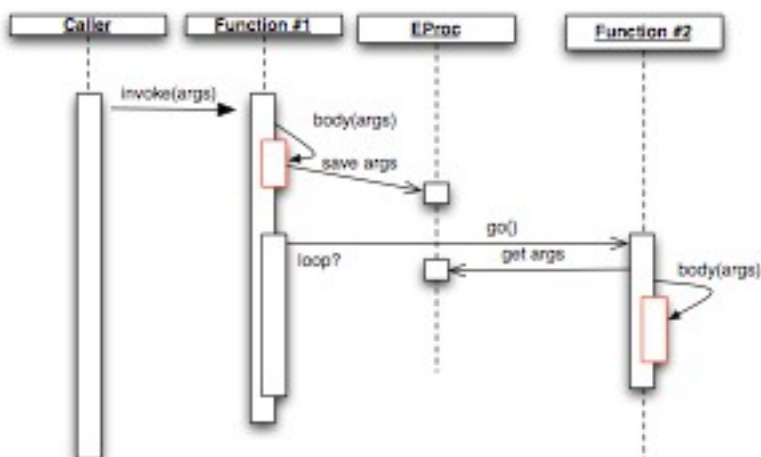
## ECons & ESeq



- ESeq is a well-formed list - similar to Clojure ISeq; .tail() is also an ESeq.
- EString is list of small integers [0..255]
- EPair is a non-well-formed list
- EBinList is [byte,byte,byte | Tail]

TRIFORK.

## Tail Recursion



TRIFORK.

# EFun / Functions

```
abstract class EFun extends EObject {  
  
    /** generic invoke, used only for apply */  
    public abstract EObject apply(EProc proc, ESeq args)  
        throws Pausable;  
  
    /** do one step in a call-recursive chain */  
    public abstract EObject go(EProc eproc)  
        throws Pausable;  
  
}
```

TRIFORK.

# EFun / Functions

```
/** generated base class for functions with 2 arguments... */  
abstract class EFun2 extends EFun {  
  
    public abstract EObject apply(EProc proc, ESeq args) throws Pausable {  
        EObject arg1 = args.head();  
        EObject arg2 = args.tail().head();  
        /* add error checking ... */  
        return invoke(proc, arg1, arg2);  
    }  
  
    public EObject invoke(EProc proc, EObject arg1, EObject arg2)  
        throws Pausable {  
        EObject res = this.body(proc, arg1, arg2);  
        while(res == TAIL_MARKER) { res = proc.tail.go(); }  
        return res;  
    }  
  
    ...  
  
}
```

TRIFORK.

# EFun / Functions

```
/** generated base class for functions with 2 arguments... */  
abstract class EFun2 extends EFun {  
  
    ...  
  
    public abstract EObject body(EProc proc, EObject arg1, EObject arg2)  
        throws Pausable;  
  
    public abstract EObject go(EProc proc)  
        throws Pausable {  
        EObject arg1 = proc.arg1;  
        EObject arg2 = proc.arg2;  
        proc.arg1 = null;  
        proc.arg2 = null;  
        proc.tail = null;  
        return this.body(proc, arg1, arg2);  
    }  
  
}
```

TRIFORK.

# Tail Recursion

```
fibo_tr( 0, Result, _Next)
-> Result; %% last recursion output
fibo_tr( Iter, Result, Next) when Iter>0
-> fibo_tr( Iter -1, Next, Result + Next).

fibo(N) -> fibo_tr( N, 0, 1) .
```

TRIFORK.

# Tail Recursion

```
class Fibo extends EFun3 {
  EObject body(EProc proc, EObject iter, EObject result, EObject next)
  {
    if (iter == Fibo.cst_zero) { return result; }
    if (ErlBif.op_gt_guard(iter, Fibo.cst_zero) != null) {
      proc.arg1 = iter.minus(Fibo.cts_one);
      proc.arg2 = next;
      proc.arg3 = result.plus(next);
      proc.tail = this;
      return TAIL_MARKER;
    }
    throw ERT.badmatch(iter,result,next);
  }
  EObject go(EProc proc) {
    return body(proc, proc.arg1, proc.arg2, proc.arg3);
  }
}
```

TRIFORK.

# Demo!

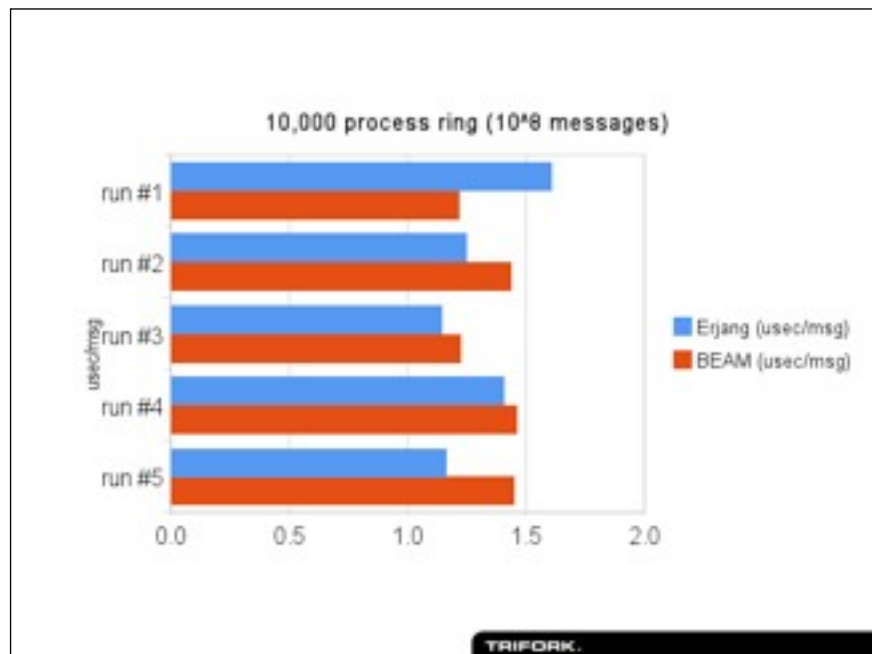
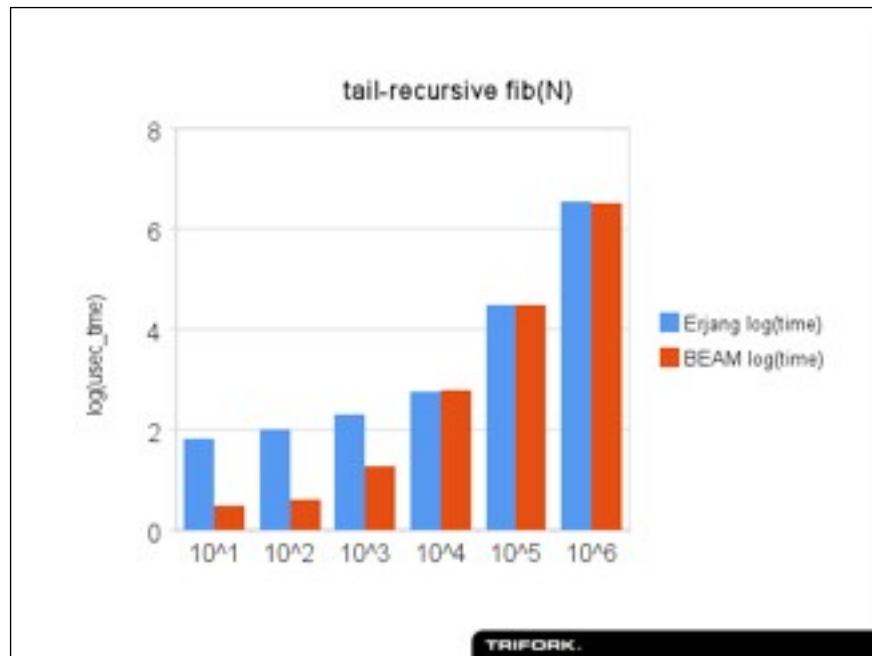
Tail-recursive Fibonacci,

10, 100, 1000, 10000, 100000

Ring example

10.000 processes x 10.000 cycles

TRIFORK.



## Interfacing to Java

- Erlang's primitive operations "BIFs" are implemented in java - obviously
- @BIF annotation makes a static-public method available from Erlang.
- Erlang port concept for "drivers"

# Example BIF

```
@BIF public static EObject
spawn_link(EProc proc, EObject mod, EObject fun, EObject args)
throws Pausable {

    EAtom m = mod.testAtom();
    EAtom f = fun.testAtom();
    ESeq a = args.testSeq();
    if (m==null||f==null||a==null)
        throw ERT.badarg(mod, fun, args);

    EProc p2 = new EProc(proc.group_leader(), m, f, a);

    p2.link_to(proc);

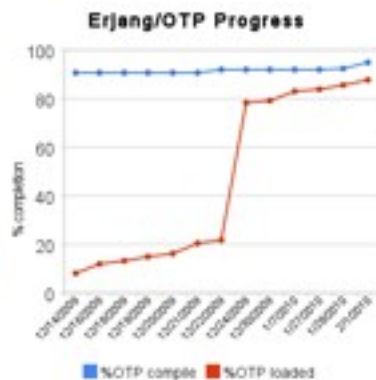
    ERT.run(p2);

    return p2.self_handle();
}
```

TRIFORK.

## Status today?

- I'm working on OTP boot; i.e., getting to the prompt of "erl" command line.
- 86% there, measured by modules loaded/compiled.



TRIFORK.

You are most welcome  
to help!

My blog: [javalimit.com](http://javalimit.com)

And: [erjang.org](http://erjang.org) [GitHub]

TRIFORK.



London 2010

Tutorials: March 8-9  
Conference: March 10-12

# QCon

[www.qconlondon.com](http://www.qconlondon.com)

INTERNATIONAL  
SOFTWARE DEVELOPMENT  
CONFERENCE

#### QCon London 2010

[Speakers](#)  
[Schedule](#)  
[Tutorials](#)  
[Tracks](#)  
[Social Events](#)

[Exhibition](#)  
[Sponsors](#)

[Registration](#)  
[Volunteers](#)

[Venue](#)  
[Travel](#)  
[Hotels](#)

[User Groups @ QCon London](#)

[Sponsor](#)  
**TMFORN**  
**InfoQ**

[About QCon](#)  
[Contact](#)  
[Archives](#)  
[Future events](#)



#### QCon is coming back to London

Tutorials: March 8-9, 2010

Conference: March 10-12, 2010

The fourth annual London enterprise software development conference designed for team leads, architects and project management is back! There is no other event in the UK with similar opportunities for learning, networking, and tracking innovation returning in the Java, .NET, Ruby, GEM, Agile, and architecture

#### Tracks for QCon London

**Architectures You've Always Wondered About** - Features Facebook, Skype, Sky.com, Ben, League of Legends

**Agile Evolution** - The changing face of Agile and future directions

**How do you test that?** - Difficult problems and state of the art in testing: functional, performance, integration, and more

**AlphaBlinks on .NET** - Alpha goes beyond .NET prescriptive .NET solutions, hear how

**Deal Stuff with Java** - Practitioner examples of reusable code problem-solutions in Java

**Functional programming** - Everything you ever wanted to know about functional

Register before February 23

and save up to £187/€224

[Register now >>](#)

#### QCon Videos

- 1 [Rich Hickey - "Persistent Data Structures and Managed References"](#)
- 2 [Martin Fowler - "Three Years of Real-World Ruby"](#)
- 3 [Tony Hoare - "Null References: The Billion Dollar Mistake"](#)
- 4 [Simon St Laurent - "Frugate Real-World Scala"](#)
- 5 [Aditya Agarwal - "Facebook: Science and the Social Graph"](#)