Erlang and ESL

Juan Puig, Systems Engineer
erlounge@madrid, March 2011
About me

• Computer Engineering  2004
  UNIVERSIDADE DA CORUÑA

• Master’s thesis  2009
  CHALMERS

• Erlang Solutions Ltd.  2010
Erlang for newbies
"The world is concurrent
Things in the world don't share data
Things communicate with messages
Things fail"

- Joe Armstrong
Massively concurrent
Fault-tolerant
Distributed
Scalable
Benchmarking Java vs Erlang

http://weblog.plexobject.com/?p=1570
Benchmarking Apache vs Yaws

http://www.sics.se/~joe/apachevsyaws.html
Other features

- Performance
  - Linear performance up to 32 cores.
  - Efforts to raise it up to 64 cores.
- Soft-real time systems
  - Miliseconds order responses
- Hot code swapping
  - Software maintenance
- Upgrading
Timeline

1986    Joe Armstrong
1995    Open Telecom Platform
1998    Released as open source
Now      R14B02
Requests per month to www.erlang.org
System Design Principles
Design patterns within OTP

• As other software design patterns but,
• Fault-tolerant, scalable oriented
• Built-in and well tested functionalities.
• OTP behaviours and user defined behaviours.
• Less code to develop.
• High % of enterprise systems are based on client-server model.
How does ESL contribute?

- Learning
- Tools
- Research
- Continuous integration
Learning
Erlang R13B04 (erts-5.7.5)
> (hello, erlang, org).
(hello,erlang.org)
> T = (hello, erlang, org).
(hello,erlang.org)
> element(2, T).
erlang
>]

Very good!

Building a list
Erlang offers many ways to store collections of elements, not just tuples. For example, it's possible to define lists. Lists in Erlang are delimited by square brackets and their elements are comma-separated. Just like tuples, the elements of a list can belong to different types.

Examples of lists are:
- [january, february, march]
- [123, pigeon, [a, b, c]]

Try to define a valid list now.

© 1999-2011 Erlang Solutions Ltd.
Main Page

Your Erlang Community Site

Welcome to trapexit.org, the Erlang community site where you can read news and weblogs related to Erlang/OTP, discuss projects, issues and ideas with other Erlang developers, and read and publish articles and HowTos related to Erlang/OTP.

Trapexit News

tryerlang.org is Live!

tryerlang.org is an interactive hands-on tutorial that allows you to try the power of Erlang directly in the browser. Try the power of the Erlang shell without installing anything to your machine! Visit: http://www.tryerlang.org

What is Erlang and OTP?

Erlang is a programming language used to build massively scalable soft real-time systems with requirements on high availability. Some of its uses are in telecoms, banking, e-commerce, computer telephony and instant messaging. Erlang's runtime system has built-in support for concurrency, distribution and fault tolerance. Originally developed at Ericsson, it was released as open source in 1998.

Latest Forum Posts

Erlang
RE: strange behavior in pg2 module
(Wed Mar 23, 2011 9:02 pm)
RE: Iterate through a nested list of strings?
(Mon Mar 21, 2011 2:40 pm)

Open Telecom Platform (OTP)
RE: Native drivers for Oracle
(Sun Mar 13, 2011 2:17 pm)
RE: otp_supervisor immediately goes to "shutdown"
(Thu Mar 10, 2011 10:30 pm)

Advanced Erlang/OTP
RE: ODBC Mac OSX 10.5 MySQL
(Wed Mar 16, 2011 6:57 am)
RE: How to drop privileges in Erlang?
(Wed Mar 16, 2011 6:56 am)
Tools

https://github.com/esl
meck

• mocking library for erlang
• automatically renaming of modules
• useful to isolate unit testing

github.com/esl/meck
Eshell V5.7.5  (abort with ^G)
1> meck:new(dog).
   ok
2> meck:expect(dog, bark, fun() -> "Woof!" end).
   ok
3> dog:bark().
   "Woof!"
4> meck:validate(dog).
   true

my_test() ->
    meck:new(library_module),
    meck:expect(library_module, fib, fun(8) -> 21 end),
    ?assertEqual(21, code_under_test:run(fib, 8)),
    ?assert(meck:validate(library_module)).
erlang-web

• Web framework for HTTP-based applications.
• Dynamic and web services.
• Built upon OTP principles.
• Can be run on top of both inets and yaws.
• MVC, full-stack framework.
• Scalable and fault-tolerant website.
entop

- Top-like Erlang monitoring tool.
- Monitors remote erlang nodes.
Research
Erlang Embedded

- Erlang applications running on embedded systems
- Collection of thesis projects supervised by ESL.
- OS (Unix based) + Erlang release.
- R14B01 (kernel + stdlib) optimised and compiled for ARM arch.

Linux based distro

Beagle Board

Gumstix

© 1999-2011 Erlang Solutions Ltd.
Android based distro

Samsung Galaxy Tab (Thanks to Robin T)

HTC Desire

HTC Supersonic (Evo) (Thanks to Garret S)

Motorola Droid (Thanks to Chris J)

Samsung Galaxy I9000 (Thanks to neurofen)

HTC Desire HD (Thanks to Mikkel M)
Classical approach
Agile means ... ?
Keeping it short means,

- Early and continuous delivery of valuable software.
- Changes on requirements are always welcome.
- Delivering *working* software frequently
- Business + Development
- Retrospective + Adjustments
Specific Improvements From Implementing Agile

- Accelerated Time-to-Market: 86%
- Increased Productivity: 87%
- Reduced Software Defects: 86%
- Reduced Cost: 63%
Flavours

- SCRUM
- Kanban
- XP
Erlang + Agile + Business?
Erlang + Agile + Business?

Erlang

Highly dynamic

Easy to write/change
Erlang + Agile + Business?

Erlang

- Highly dynamic
- Easy to write/change

Agile

- Highly formalised
- Highly structured
Erlang + Agile + Business?

Erlang

Highly dynamic
Easy to write/change

Agile

Highly formalised
Highly structured

DSDM
DSDM

• Yet another Agile dev. method framework.
• On-time and within budget deliveries.
• Business perspective.
• Adjusting for changes along the way.
• Self-organised with own decisions.
• Driven by users’ feedback.
• Timeboxing, MoSCoW, prototyping...
Continuous integration within ESL
Continuous integration

- Iterative and often quality control
- Aims to
  - Improve software quality
  - Reduce time to market
- Automated self-testing builds
- Easy to release deliverables
- Automated deployment
Finding the need

• Erlang/OTP successfully moved to Github (2009)
  - Easy to fork
  - Open to contributions
  - No CI for the open source community.
  - Hard to estimate the development progress.
  - Tough patch submission procedures
  - Lack of tools (doc. contribution?)
Hosted continuous integration tool
Hosted continuous integration tool

TaaS

open source

© 1999-2011 Erlang Solutions Ltd.
Hosted continuous integration tool
MOEBIUS

- Continuous integration server.
- Written in Erlang.
- Builds and tests automation.
- Immediate feedback for any patch / update.
- Multi-platform output.
MOEBIUS

• Continuous integration server.
• Written in Erlang.
• Builds and tests automation.
• Immediate feedback for any patch / update.
• Multi-platform output.

QUICK FLOW
At the moment
At the moment

Erlang/OTP
Github
At the moment
At the moment

User

fork

Erlang/OTP

Github
At the moment

User

fork

Erlang/OTP

Github

dev
At the moment

User

fork

New patch

dev

Erlang/OTP

Github
At the moment

User

fork

New patch

dev

Erlang/OTP

Github
At the moment
At the moment

User → dev → New patch

User → fork → Erlang/OTP

OTP Team

CI Validation

Erlang/OTP

Github
At the moment

1. User
2. New patch
3. OTP Team
4. CI Validation
5. Erlang/OTP Github
6. fork
dev
MOEBIUS

User

Erlang/OTP
Github

© 1999-2011 Erlang Solutions Ltd.
MOEBIUS

User

fork

Erlang/OTP

Github

New patch

dev
MOEBIUS

New patch

dev

User

fork

Erlang/OTP Github

© 1999-2011 Erlang Solutions Ltd.
MOEBIUS

New patch

Mac OS X

Debian

Ubuntu

Windows

User

fork

Erlang/OTP

Github

Erlang/OTP

Github
MOEBIUS - Goals

- Web UI - current status
- HTTP API
- System integration
  - Bug trackers
  - Mailing list
- Minimise manual effort
- Offer better packages
MOEBIUS - Package support

• OS packages for supported platforms
  - Easy for 1st time users

• Topic packages (e.g. web development, embedded)
Questions?

juan@erlang-solutions.com