Quality Cruising
Making Java Work for Erlang

Erik (Happi) Stenman

Introduction

Introduction

I will talk about automated testing, and how to make Java do that work for you.

Introduction

I will talk about automated testing, and how to make Java do that work for you.

Introduction

I will talk about automated testing, and how to make Java do that work for you.

But first a short recap for those of you who weren't here last year or don't remember my talk about Kreditor.

Kreditor Europe AB

- Founded in December 2004.

KREDITOR

Kreditor Europe AB

- Founded in December 2004.
- Bring trust to Internet shopping, by providing old style billing through hi-tech solutions.

KREDITOR

Kreditor Europe AB

- Founded in December 2004.
- Bring trust to Internet shopping, by providing old style billing through hi-tech solutions.
- More than 1700 Internet shops connected.

KREDITOR

Kreditor Europe AB

- Founded in December 2004.
- Bring trust to Internet shopping, by providing old style billing through hi-tech solutions.
- More than 1700 Internet shops connected.
- The company vision:

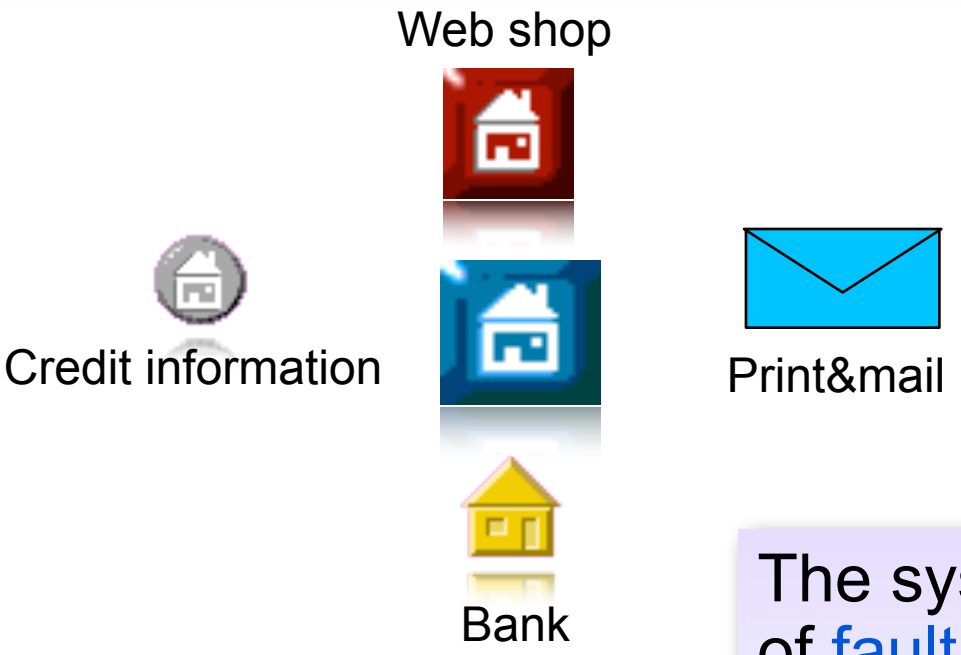
KREDITOR

Kreditor Europe AB

- Founded in December 2004.
- Bring trust to Internet shopping, by providing old style billing through hi-tech solutions.
- More than 1700 Internet shops connected.
- The company vision:
  - "Be the coolest company in Sweden."

KREDITOR

The Implementation



The system is a combination of fault tolerant servers, lots of protocol- and glue-code, persistent storage, and a web-UI.

KREDITOR

Some implementation details

- The system is built in-house from scratch using LYME (Linux, Yaws, Mnesia, and Erlang).
- We have a distributed system with multiple servers to provide a fault tolerant, high availability solution.
- We aim for 5 nines availability, in a setting where we introduce new features in the system every week (almost every day).
- The problem fits Erlang really well.

Process

KREDITOR

Process

- We have have an informal agile process.

Process

- We have have an informal agile process.
- We have very short time to market,
  for simple changes the time from idea to finished integration can be less than one hour.

Process

- We have have an informal agile process.
- We have very short time to market,
  for simple changes the time from idea to finished integration can be less than one hour.
- It is crucial for us to have an automated comprehensive test suite.

Process

- We have have an informal agile process.
- We have very short time to market,
  for simple changes the time from idea to finished integration can be less than one hour.
- It is crucial for us to have an automated comprehensive test suite.
  - With a framework that works.

KREDITOR

Process

- We have have an informal agile process.
- We have very short time to market,
  for simple changes the time from idea to finished integration can be less than one hour.
- It is crucial for us to have an automated comprehensive test suite.
  - With a framework that works.
  - Which is **used**.

KREDITOR

Process

- We have have an informal agile process.
- We have very short time to market,
  for simple changes the time from idea to finished integration can be less than one hour.
- It is crucial for us to have an automated comprehensive test suite.
  - With a framework that works.
  - Which is **used**.
  - **Always**.

Process

- We have have an informal agile process.
- We have very short time to market,
  for simple changes the time from idea to finished integration can be less than one hour.
- It is crucial for us to have an automated comprehensive test suite.
  - With a framework that works.
  - Which is **used**.
  - **Always**.
- Enters Yatsy and CruiseControl.

Yatsy

- Yet Another Test Server – Yaws compatible
  (Yatsy is Swedish for Yatzee – testing is a bit like a dice game.)

Yatsy

- Yet Another Test Server – Yaws compatible
  (Yatsy is Swedish for Yatzee – testing is a bit like a dice game.)
- Why (yet) another test server?

Yatsy

- Yet Another Test Server – Yaws compatible
  (Yatsy is Swedish for Yatzee – testing is a bit like a dice game.)
- Why (yet) another test server?
  - The "*released*" OTP-test server is from 2004.

Yatsy

- Yet Another Test Server – Yaws compatible
  (Yatsy is Swedish for Yatzee – testing is a bit like a dice game.)
- Why (yet) another test server?
  - The "*released*" OTP-test server is from 2004.
  - It isn't really open source.

Yatsy

- Yet Another Test Server – Yaws compatible
  (Yatsy is Swedish for Yatzee – testing is a bit like a dice game.)
- Why (yet) another test server?
  - The "*released*" OTP-test server is from 2004.
  - It isn't really open source.
  - We just couldn't get it to work.

Yatsy

- Yet Another Test Server – Yaws compatible
  (Yatsy is Swedish for Yatzee – testing is a bit like a dice game.)
- Why (yet) another test server?
  - The "*released*" OTP-test server is from 2004.
  - It isn't really open source.
  - We just couldn't get it to work.
- First version hacked together over a weekend by Tobbe.

Yatsy

- Yet Another Test Server – Yaws compatible
  (Yatsy is Swedish for Yatzee – testing is a bit like a dice game.)
- Why (yet) another test server?
  - The "*released*" OTP-test server is from 2004.
  - It isn't really open source.
  - We just couldn't get it to work.
- First version hacked together over a weekend by Tobbe.
  - Released as open source:
    `http://code.google.com/p/yatsy/`

Yatsy - example

```
-module(example_SUITE).
-export([all/1, init_per_suite/1, fin_per_suite/1, init_per_testcase/2,
         fin_per_testcase/2, simple/1]).
-include("yatsy.hrl").

all(doc) -> ["Test cases for example."];
all(suite) -> [simple].

init_per_suite(Config) when list(Config) -> Config.
fin_per_suite(_Config) -> ok.
init_per_testcase(_TestCase, Config) when atom(_TestCase),list(Config) -> Config.
fin_per_testcase(_TestCase, _Config) -> ok.
```

```
simple(doc) ->
     ["Check that we can get an new example."];
 simple(Conf) when is_list(Conf) ->
     [] = example:new(),
     ok.
```


KREDITOR

CruiseControl (CC)

- CruiseControl is a framework for *continuous integration*.

CruiseControl (CC)

- CruiseControl is a framework for *continuous integration*.
- It is open source.

CruiseControl (CC)

- CruiseControl is a framework for *continuous integration*.
- It is open source.
- It is written in Java.

CruiseControl (CC)

- CruiseControl is a framework for *continuous integration*.
- It is open source.
- It is written in Java.
- It automatically

CruiseControl (CC)

- CruiseControl is a framework for *continuous integration*.
- It is open source.
- It is written in Java.
- It automatically
  - checks out the latest version from a repository,

CruiseControl (CC)

- CruiseControl is a framework for *continuous integration*.
- It is open source.
- It is written in Java.
- It automatically
  - checks out the latest version from a repository,
  - does a build,

CruiseControl (CC)

- CruiseControl is a framework for *continuous integration*.
- It is open source.
- It is written in Java.
- It automatically
  - checks out the latest version from a repository,
  - does a build,
  - runs all tests,

CruiseControl (CC)

- CruiseControl is a framework for *continuous integration*.
- It is open source.
- It is written in Java.
- It automatically
  - checks out the latest version from a repository,
  - does a build,
  - runs all tests,
  - and it measures any metric you have automated.

CruiseControl (CC)

- CruiseControl is a framework for *continuous integration*.
- It is open source.
- It is written in Java.
- It automatically
  - checks out the latest version from a repository,
  - does a build,
  - runs all tests,
  - and it measures any metric you have automated.
  - As soon as **anyone** checks **anything** in.

CC – Components

CC consists of:

CC – Components

CC consists of:

- A set of plugins

CC – Components

CC consists of:

- A set of plugins
  - Version control pollers (e.g. `svn st -u`)

CC – Components

CC consists of:

- A set of plugins
  - Version control pollers (e.g. `svn st -u`)
  - Compile and test systems (e.g. ant or make)

CC – Components

CC consists of:

- A set of plugins
  - Version control pollers (e.g. `svn st -u`)
  - Compile and test systems (e.g. ant or make)
  - Publishers (e.g. web site, email, rss, irc)

CC – Components

CC consists of:

- A set of plugins
  - Version control pollers (e.g. `svn st -u`)
  - Compile and test systems (e.g. ant or make)
  - Publishers (e.g. web site, email, rss, irc)
- A build queue

CC – Components

CC consists of:

- A set of plugins
  - Version control pollers (e.g. `svn st -u`)
  - Compile and test systems (e.g. ant or make)
  - Publishers (e.g. web site, email, rss, irc)
- A build queue
- An admin GUI

CC – Components

CC consists of:

- A set of plugins
  - Version control pollers (e.g. `svn st -u`)
  - Compile and test systems (e.g. ant or make)
  - Publishers (e.g. web site, email, rss, irc)
- A build queue
- An admin GUI
- Lots of glue code

The CC loop

- Poll for event
  - –A version control update
  - –A time has come
- Checkout, build, and run tests
- Gather results
- Publish results

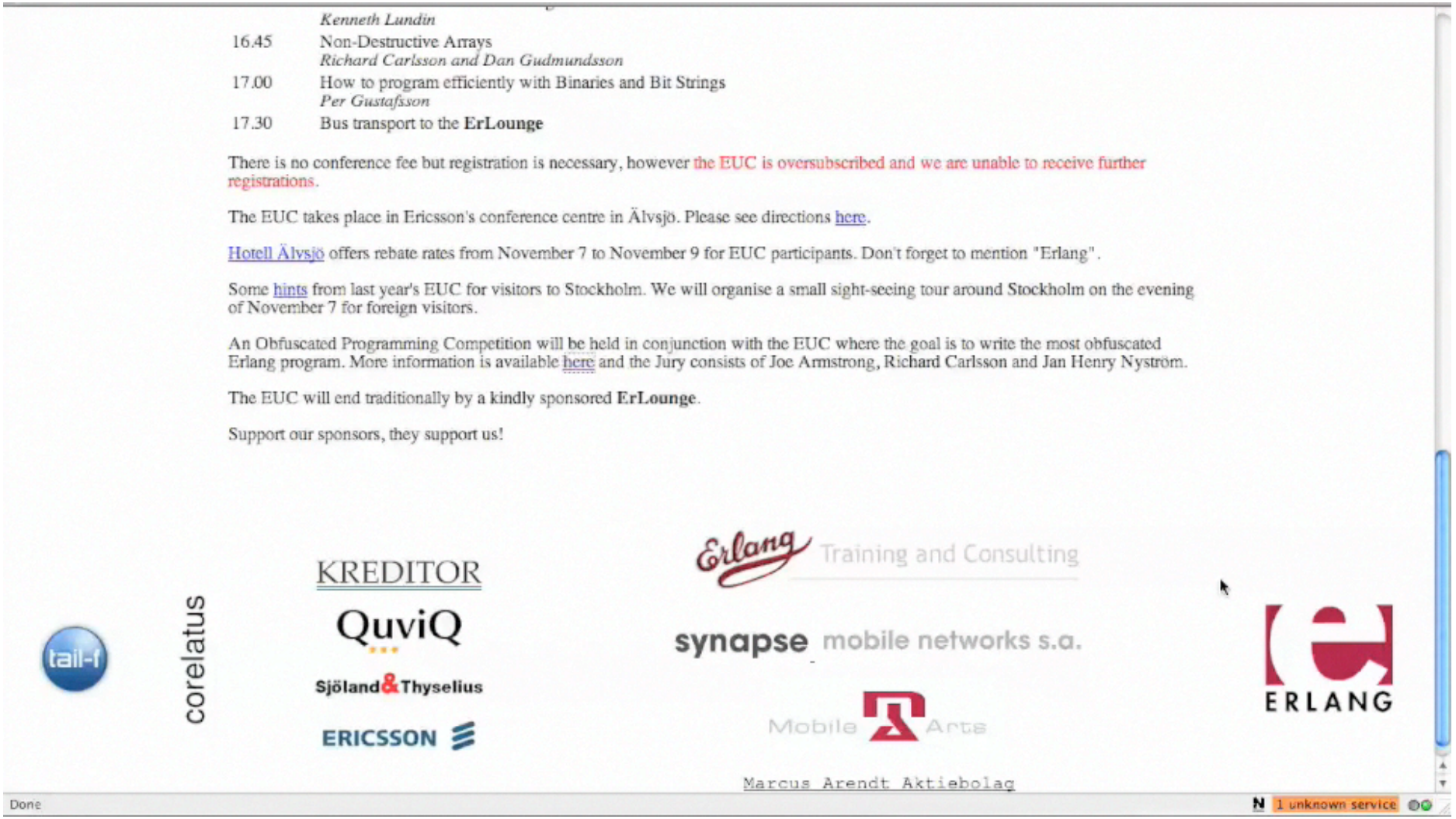
Our CC Setup

- A set of projects
  - commit-[branch], runs on every commit
  - nightly-[branch], runs each night
- CC starts a script
  - svn update
  - make
  - run yatsy
- Reporting
  - Yatsy reports test results trough an xml file
    - Reverse engineered from JUnit
  - CC sends email to responsible developer in case of error

Integrating Cover with Yatsy

- There is an application called Cover in Erlang/OTP which can give you a line by line coverage analysis of a module.
- We have added a module `yatsy_cover` to Yatsy which will let you run cover on a test suite.
- If you run yatsy with the "cover flag" all beam files are "cover compiled" before the test suite is run.
- Yatsy produces a source code file annotated with execution count.

CC-Demo



KREDITOR

Conclusion

- CC gives you *immediate* feedback on the test status of all your branches all the time.

KREDITOR

Conclusion

- CC gives you *immediate* feedback on the test status of all your branches all the time.
- **Our contribution**:

Conclusion

- CC gives you *immediate* feedback on the test status of all your branches all the time.
- **Our contribution**:
  - Open sourced test suite – Yatsy.

KREDITOR

Conclusion

- CC gives you *immediate* feedback on the test status of all your branches all the time.
- **Our contribution**:
  - Open sourced test suite – Yatsy.
  - Nice simple integration between CC and Erlang, including cover and dialyser, and EUnit.

Conclusion

- CC gives you *immediate* feedback on the test status of all your branches all the time.
- **Our contribution**:
  - Open sourced test suite – Yatsy.
  - Nice simple integration between CC and Erlang, including cover and dialyser, and EUnit.
  - We let Java do some work for the Erlang developer.

Conclusion

- CC gives you *immediate* feedback on the test status of all your branches all the time.
- **Our contribution**:
  - Open sourced test suite – Yatsy.
  - Nice simple integration between CC and Erlang, including cover and dialyser, and EUnit.
  - We let Java do some work for the Erlang developer.
- CC is just as valuable as a VCS.

Conclusion

- CC gives you *immediate* feedback on the test status of all your branches all the time.
- **Our contribution**:
  - Open sourced test suite – Yatsy.
  - Nice simple integration between CC and Erlang, including cover and dialyser, and EUnit.
  - We let Java do some work for the Erlang developer.
- CC is just as valuable as a VCS.
- If you are not using CC for your projects, start using it **NOW**.