

Betting on FP (and winning?)

Erik Stenman

Introduction

I will talk about [KREDITOR](#), a company that bet it's future on Functional Programming

- Conventional wisdom...
 choose proven technology
- ... [the KREDITOR way](#) ...
 choose Erlang
- ... are they the same?

I will tell you what Kreditor does, how we do it, why we do it this way, and whether it worked out or not... at least, so far.

[KREDITOR](#)

Kreditor Europe AB

- The business model:
 - Bring trust to Internet shopping.
 - Bring old style billing into the new IT-economy.
- Brief background:
 - Founded in December 2004.
 - With < \$100,000 in venture capital.
 - Live system in March 2005.
- The company vision:
 - “Be the coolest company in Sweden.”

The Problem

- Internet shopping is a question of **trust**.
 - The **shop** has to **trust** the **customer** to get paid.
 - The **customer** has to **trust** the **shop** to send the right stuff.
- Many **customers** are **uncomfortable** using credit card over the Internet.
- Many banks are actually **worried about the security** of Internet **shops** handling credit card information.
- Also, doing a *partial return* when using credit card is a **hassle**, both for the **customer** and the **shop**.

The Solution

- Bring in *a trusted party*, i.e., **KREDITOR**.
- Send an invoice with the goods to the **customer**.
- The **customer** pays after receiving the goods and takes no risk. The **customer** does **not** have to trust anyone.
- The **shop** is guaranteed (by contract) to get money from **KREDITOR**. The **shop** only have to trust **KREDITOR** with whom they have a written contract.

KREDITOR

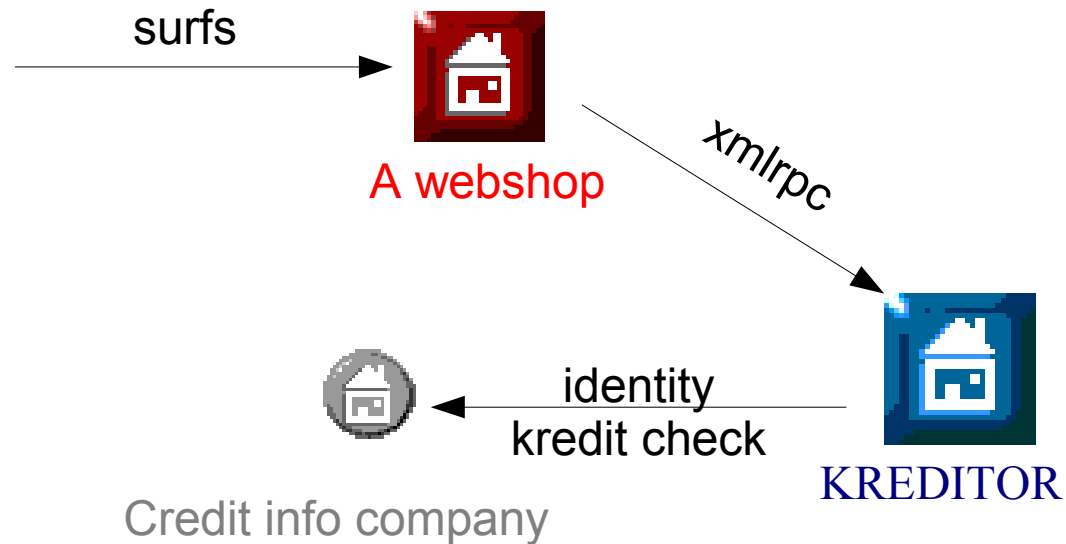
Added benefits

- The **customer** gets credit.
- The **customer** can pay using familiar methods.
- Returning goods is easy.
- Better fraud detection.
- Advanced credit assessments.
- Easy to add similar features like *pre-pay* and *subscriptions*.

The Implementation



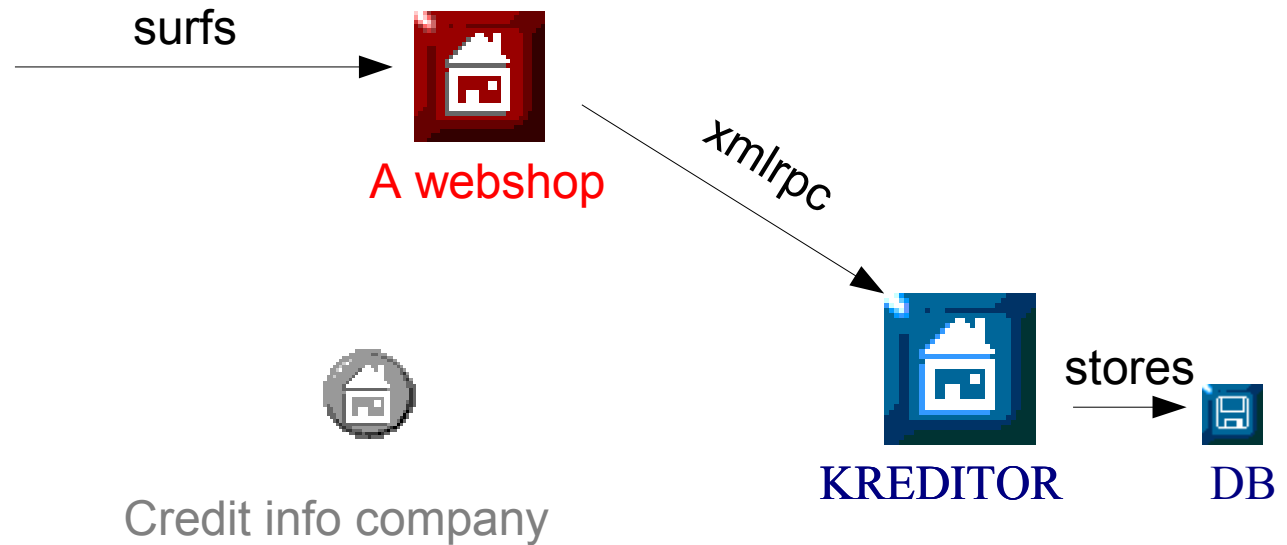
A customer



The Implementation



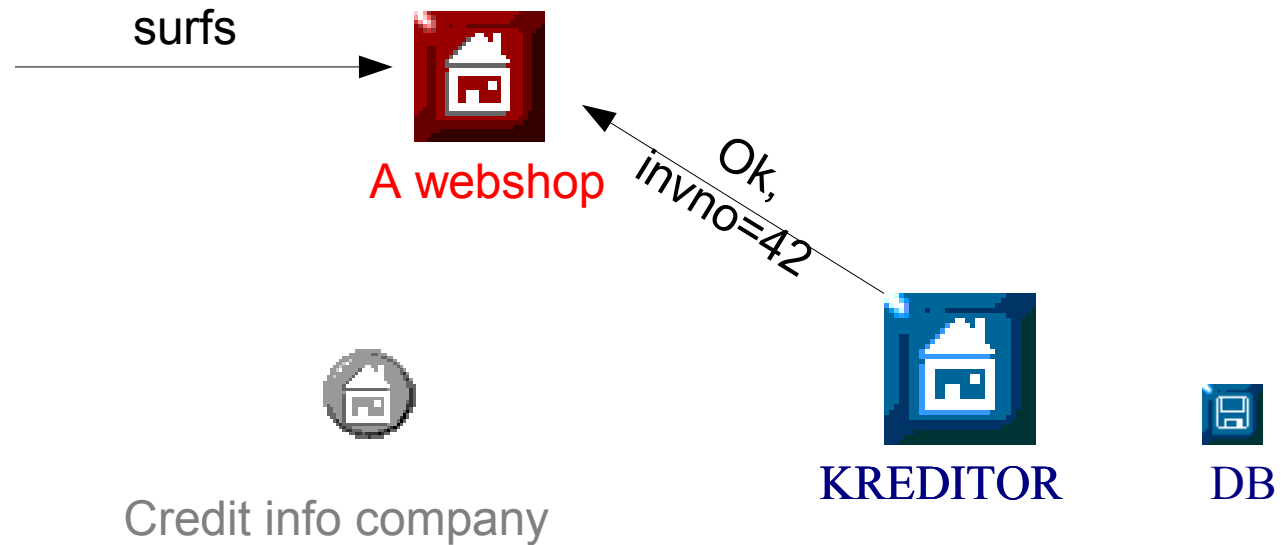
A customer



The Implementation



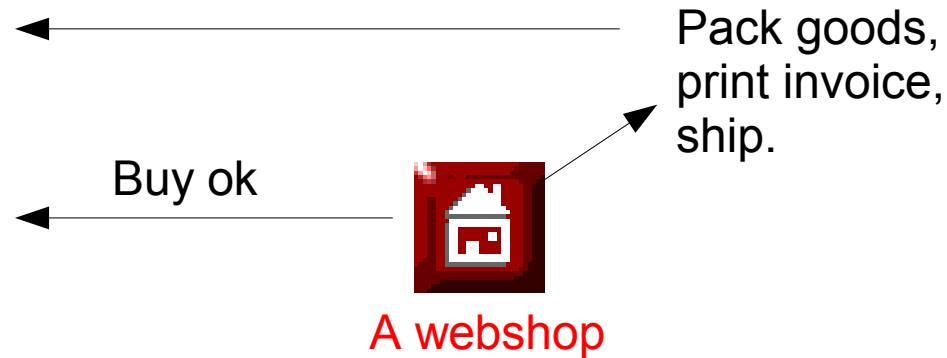
A customer



The Implementation



A customer



Credit info company



KREDITOR

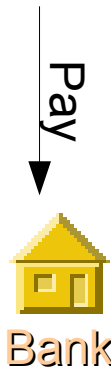


DB

The Implementation



A customer



Bank



A webshop



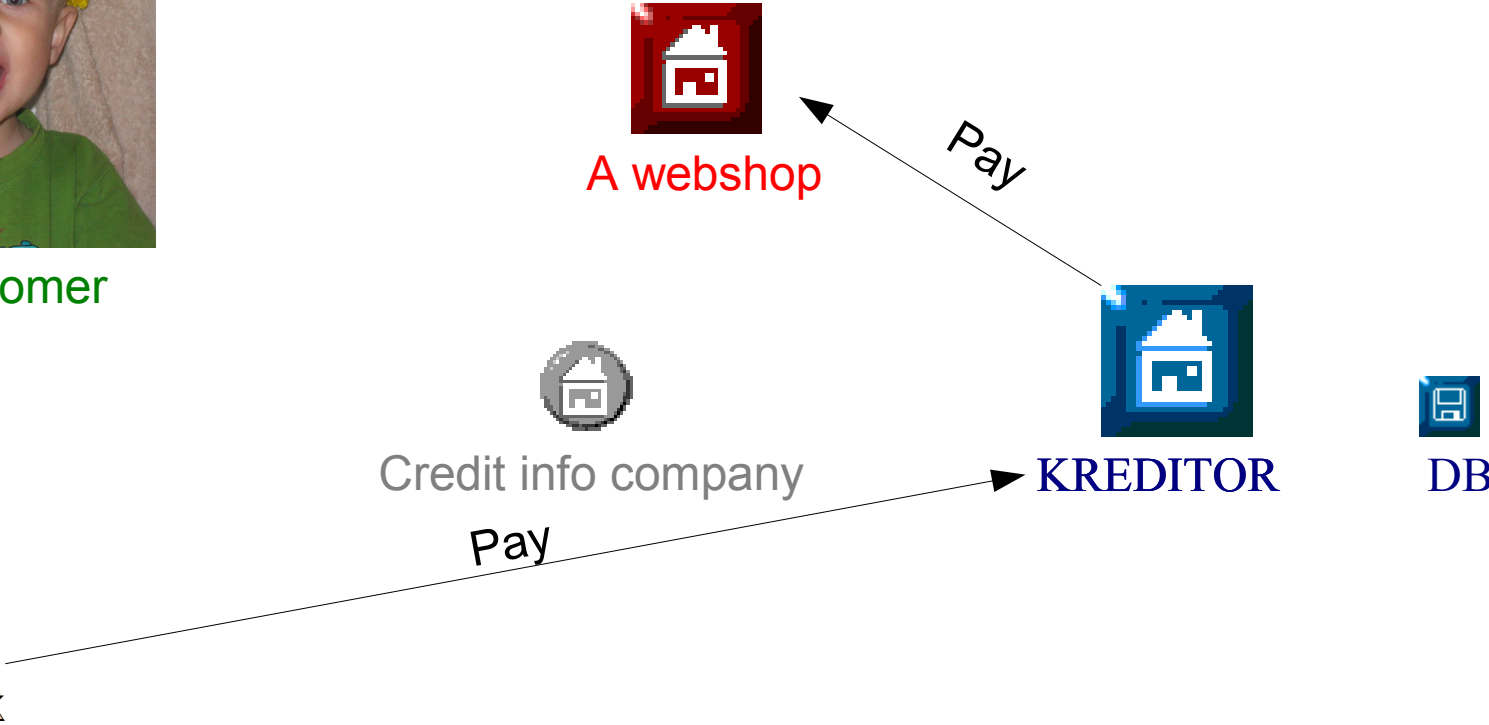
Credit info company



KREDITOR



DB



The Implementation



A customer



A webshop



Credit info company

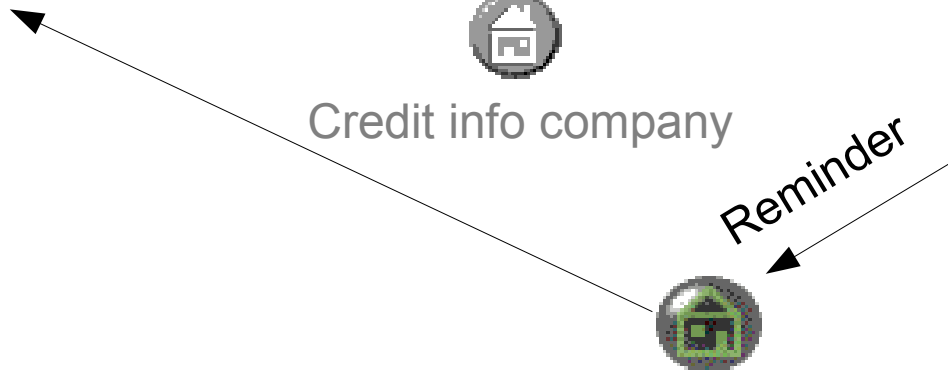


KREDITOR

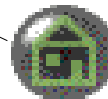


DB

Reminder



Bank



Print&mail company

The Implementation



A customer



A webshop



Credit info company



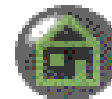
KREDITOR



DB



Bank



Print&mail company

Some details

- The system is built from scratch using LYME (Linux, Yaws, Mnesia, and Erlang).
- So far we only operate in Sweden and Norway.
- 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.
- The problem fits Erlang really well.

Why was Erlang Chosen?

- The three founders didn't really know what Erlang was.
- The developers didn't get the scope of the system. “It will be like the programming contest, we'll do it over the weekend” - Claes Wikström.
- Thus Kreditor and Teknikerl were born.
- Teknikerl built the first version of the system, in Erlang, in < 4 month. (While starting another company called Tail-f, but that's another story.)

Why was Erlang Chosen?

- Jane Walerud saw a business presentation by three enthusiastic entrepreneurs-to-be at a business “green house”.
- The Idea looked promising and she happened to know five former Bluetailers who were looking for something new to do.

Did it work?

- Erlang has been a great help in providing rapid development with maintained high availability.
- KREDITOR has introduced four new major services since last December, and added over 500 new customers.
- We have expanded into Norway.
- The system has never gone down **except for short maintenance stops.**

Did it work?

- Using Erlang has meant low development costs.
- Our main competitor busted this summer after burning more than 90 million SEK (~\$12M).
(I think they had some php/.net solution.)
- We know of some banks that have invested over 200MSEK to try to get systems that does something like our base service in a much more cumbersome way.

Did it work?

- The business model has been sound.
- Total investment < **\$100,000**
- Turnover:
 - 2004: ~0 SEK
 - 2005: 1.5 million SEK ~ \$200,000
 - *2006: 15 million SEK ~ \$2 million, (500million SEK in invoices)*
- Number of connected stores:
2004: 0, 2005: ~200, Today: > 700
- Number of employees > 17.
- **Second place on the list of Sweden's most promising entrepreneurs by the Swedish magazine "Internet World".**

Why not use Erlang?

- The main reasons that I have heard of are:
 1. Politics – Erlang is not C/Java, company policy.
 2. One provider – Concern that Ericsson will stop supporting Erlang.
 3. Lack of programmers – Erlang is still not mainstream how can we ensure we get qualified staff?
- When starting a new company, 1 is not a problem.
- I can't see 2 happening, and it's open source anyway.
- When setting up in Stockholm, 3 is not a problem.

Conclusion

KREDITOR took a bet on Erlang,
and so far seem to be winning.

Questions?