goto; conference aarhus

# CHOOSING THE RIGHT TECHNOLOGY ISN'T ENOUGH


**Cons T Åhs**

*Klarna AB, Sweden*


INTERNATIONAL SOFTWARE DEVELOPMENT CONFERENCE

gotocon.com

klarna

# Choosing the right technology isn't enough

Growth pains

Dangers of technology focus

People matter more in the long run

Hiring - finding the right people

klarna

# Who am I?

- Written code for more than a couple of decades - large number of languages, paradigms and technologies (at lot of which are now obsolete)
- Teaching and research
- Developer with a focus on long term high quality development
- Focus on improving and maintaining code quality for long term survival
- Prefers statically typed functional languages

# Klarna - the business

- Handle payments for online shopping
- Invoice, the consumer pays *after* getting the goods
- Measure of success:
  - founded in 2005
  - active in seven countries and expanding
  - now some 750 employees and growing
  - revenue over 100M Euro

klarna

# THE REQUIREMENTS

- 24/7 business
  - people shop all the time
  - low tolerance for downtime, both from merchants and shoppers
  - highly available, scalable and fault tolerant
- We're handling large amounts of money that do not belong to us
  - zero tolerance for inconsistency
- In terms of the CAP theorem, consistency and availability are the most important

# The technology erlang/otp

klarna

- built for simple distribution and fault tolerance
- robust model for concurrent computation
- easy to upgrade system without downtime
- easy to upgrade hardware without downtime
- ..
- .. lots of really cool and impressive properties when building a system for high availability and consistency

- Use erlang and win big?

# People matter more in the long run

klarna

- The choice of technology is of course part of the success story, but very far from the whole story
- The initial choice of technology is important, but the crucial issue is putting the technology to good use
- This is done by people!
  - For initial success having the right people is crucial
  - For continued success during intense growth, finding more of the right people is even more important
  - Growing is difficult
  - Adding the right people is difficult

klarna

# What "technology"?

- Technology might be a language, framework, persistence layer, OS, hardware..
- In software development high quality and productivity are often sought after, but remain difficult to define and measure.
  - They cannot be measured by a single measurement
  - The actual value is apparent over time
  - They are about the long term

klarna

# FOCUSING ON TECHNOLOGY

- Risk of being locked in
- Relying too much on the technology
  - overusing it
  - being seduced by the dark side
  - difficult to migrate technology

# Don't lure people with the technology..

klarna

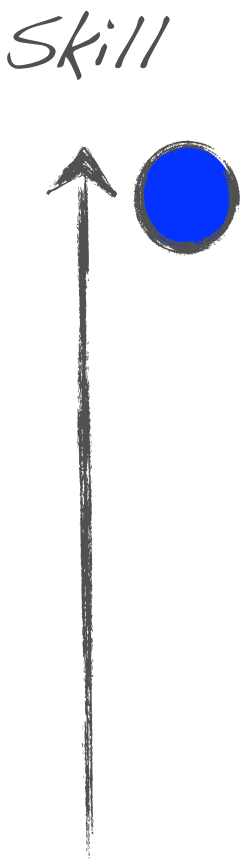
- The focus should be on the product(s) and domain(s), not the technology used
- Attracting developers with the technology
  - good short term solution
  - developers focused on the technology will apply the technology everywhere
  - a skewed sense of where the actual problems lie
  - produce code rather than solutions

klarna

# Technology vs people

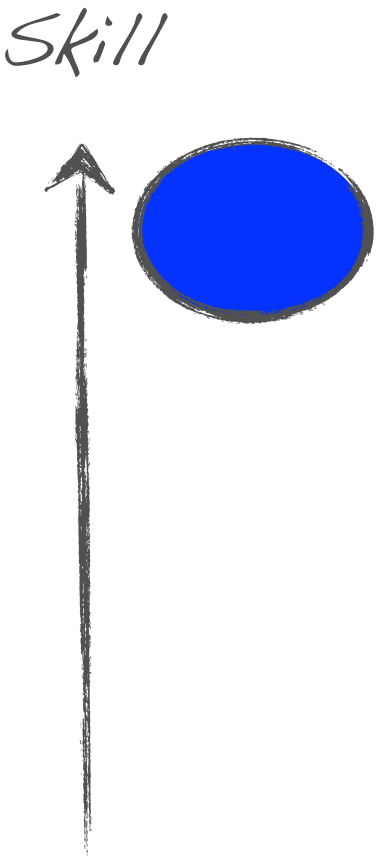
- The choice of technology does matter
- The right people will choose the right technology
- Focusing on solutions the dark side of the technology will be avoided
- Abstraction layers are used to avoid too strong coupling
- Technologies become obsolete, so the solutions are prepared for this

# Growth - start up




- Recipe for success:
  - Small team of highly skilled and focused developers
  - A great idea
- Close communication within developers and towards management (if there is any)
- Controlled by the short term
- Choice of technology is obvious
- Lots of code is produced (and thankfully thrown away)
- The code base is small

# Growth - Initial Expansion



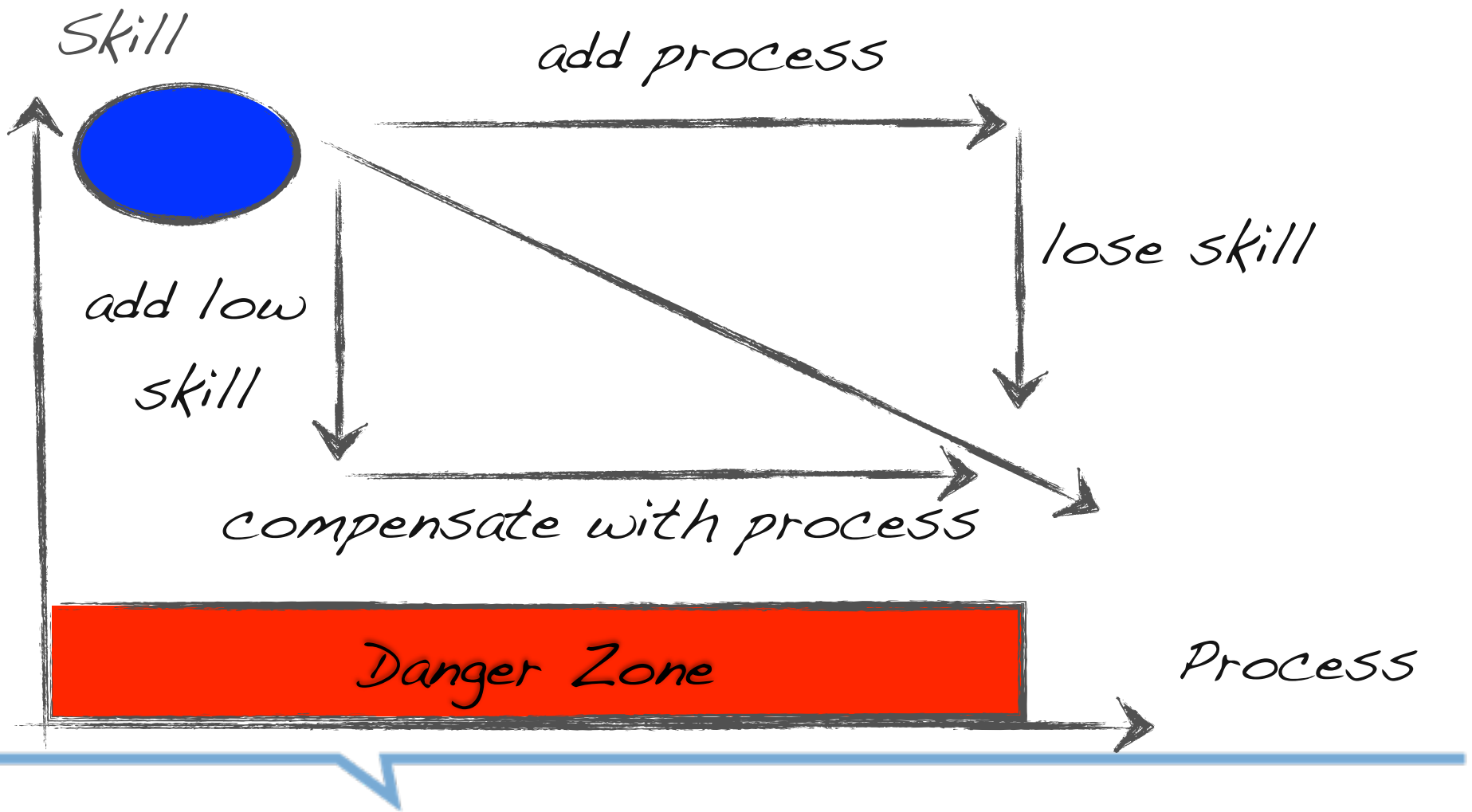

- Add more developers (and others) carefully
- By careful selection you can maintain productivity with the initial ways of working
- Distance of communication still rather short
- Time to market more important than perfect code
- The code base grows..
- .. and technical debt starts to show

klarna

# Growth - becoming large

- More developers needed, but the initial careful selection degrades (possibly by outsourcing)
- The initial team becomes several teams
- “Distance” grows in all directions
- Technical debt grows
- Production slows down
- Quality becomes unknown at best
- Management “loses” control and visibility and tries to solve the problems by
  - adding processes and methods
  - adding more developers

# Growth


Skill
add process
lose skill
add low skill
compensate with process
Danger Zone
Process

# Problems with successful growth

klarna

- Growing technical debt while trying to stay productive
- Honesty and expectations are difficult
- It's easier to just add body count than making the bodies you add really count
- Tech people are best at *understanding* tech people, but not too good at managing them
- The mindset and culture of a small startup does not always scale well

klarna

# Growth vs people

- Generalise at the right time
  - doing too early or too late might mean lost opportunities
- Communicate problems and consequences of short cuts
- Be part of the hiring process
- Take good care of new comers
  - Be honest about problems in the code
  - Too easy to follow existing patterns in the belief that they are good
- Skill is more important than processes and methods

klarna

# Hiring the right people

- Finding, and keeping, the right people are crucial issue for success in the long run
- People are more diverse than technologies
- Hiring the right people is important and difficult - let it take some time
  - Hiring the wrong people is quite expensive
  - Be prepared to do a lot of interviews
  - Be prepared to say no a lot
  - Use your developers as part of the hiring process
  - HR people do not often have a deep technical background

klarna

# Hiring the right people

- If you hire junior developers, be prepared to take the cost
  - education and mentoring to get them on track
  - If you're not a university, your main goal is probably not to give basic computer science education
- It's more difficult and expensive to change the mindset and educate developers at the low end of the skill scale

klarna

# Who are they?

- Focus on paradigms, not languages or technologies
- Problem solvers, not builders
- Focus on solutions, not writing code
- They choose the right technology for the task at hand, but don't assume that the choice is permanent
- Being quick is not key - it's better to avoid writing code.
- They are there for the long run, not just a short sprint
- They care about quality and maintainability
- The first solutions is not to rewrite everything from scratch

klarna

# Who are they?

- They know the deep meaning of abstraction
- Can handle the big picture
  - High level to low level
  - Knows and handles boundaries
  - Knows the value of using building blocks
  - Also has the ability to build the actual blocks if needed
- They know the differences between code, programs and systems
- They avoid working overtime because there is no need

# Experienced and well educated

klarna

- Having experience from several problem domains and work places results in a broader perspective - there is not only one solution or technology
- Having a solid education means knowing both possibilities and limits of algorithms, data structures, methods, technologies etc
- Having a solid education means knowing what you know and what you don't
- Eager to learn more and takes the time to learn more than "X for dummies" or "X in five minutes"

klarna

# Well read

- The have read, understood and apply the lessons from
  - The Mythical Man Month
  - No Silver Bullet
  - The Pragmatic Programmer
  - ..

# Well versed Communicators

klarna

- among peers
- mentors junior developers
- inform stake holders and management about the effects of choices being made
  - long term effect of cutting corners; being pragmatic means we have to do it, but not too often
- Not afraid of admitting that they're wrong or pointing out when someone else is
- Learn from mistakes (their own as well as others)

# Skill is not one dimensional

klarna

- Finding one great developer and cloning him/her will *not* solve your problems
- Great teams are made of great people with *different* skill sets, being specialists in different areas
- The technical leader needs to be a good communicator and role model for long term issues
- Narrow minded developers are good for the really hard problems - give them a problem and go away
- Add some generalists and hard workers/builders
- Now you're only left with the problem of managing the team..

klarna

# Summary

- People are more important than technology
- Growing up is difficult
- Finding the Right People is difficult
- The really good developers are few and far between
- Great teams are built out of people with different skills
- I still find code easier to understand than people