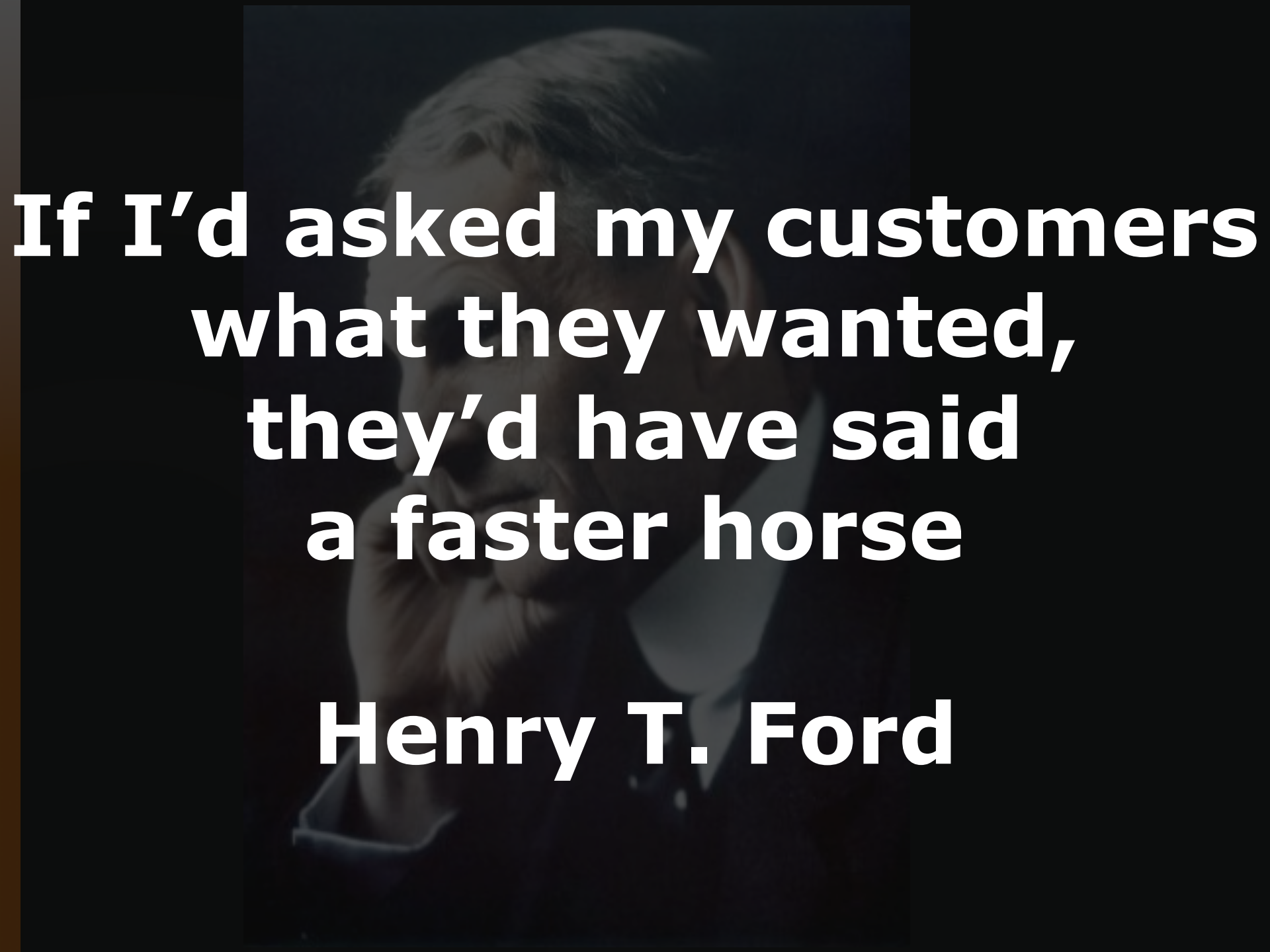


# Do As I Say; Not As I Do?

## From Requirements Engineering to Experimenting with Customers

Jan Bosch  
Professor of Software Engineering  
Chalmers University of Technology  
Gothenburg, Sweden.  
[www.janbosch.com](http://www.janbosch.com)

April 2013  
REFSQ 2013



**If I'd asked my customers  
what they wanted,  
they'd have said  
a faster horse**

**Henry T. Ford**

**Customers don't know what they want. It's very hard to envision the solution you want without actually seeing it.**

**Marty Cagan**

**The critical failing of user interviews is that you're asking people to either remember past use or speculate on future use of a system**

**Jakob Nielsen**



**[The assumption that a]  
reasonably well-defined set of  
requirements exists, if only we  
take the time to understand  
them, is wrong**

**Dean Leffingwell**

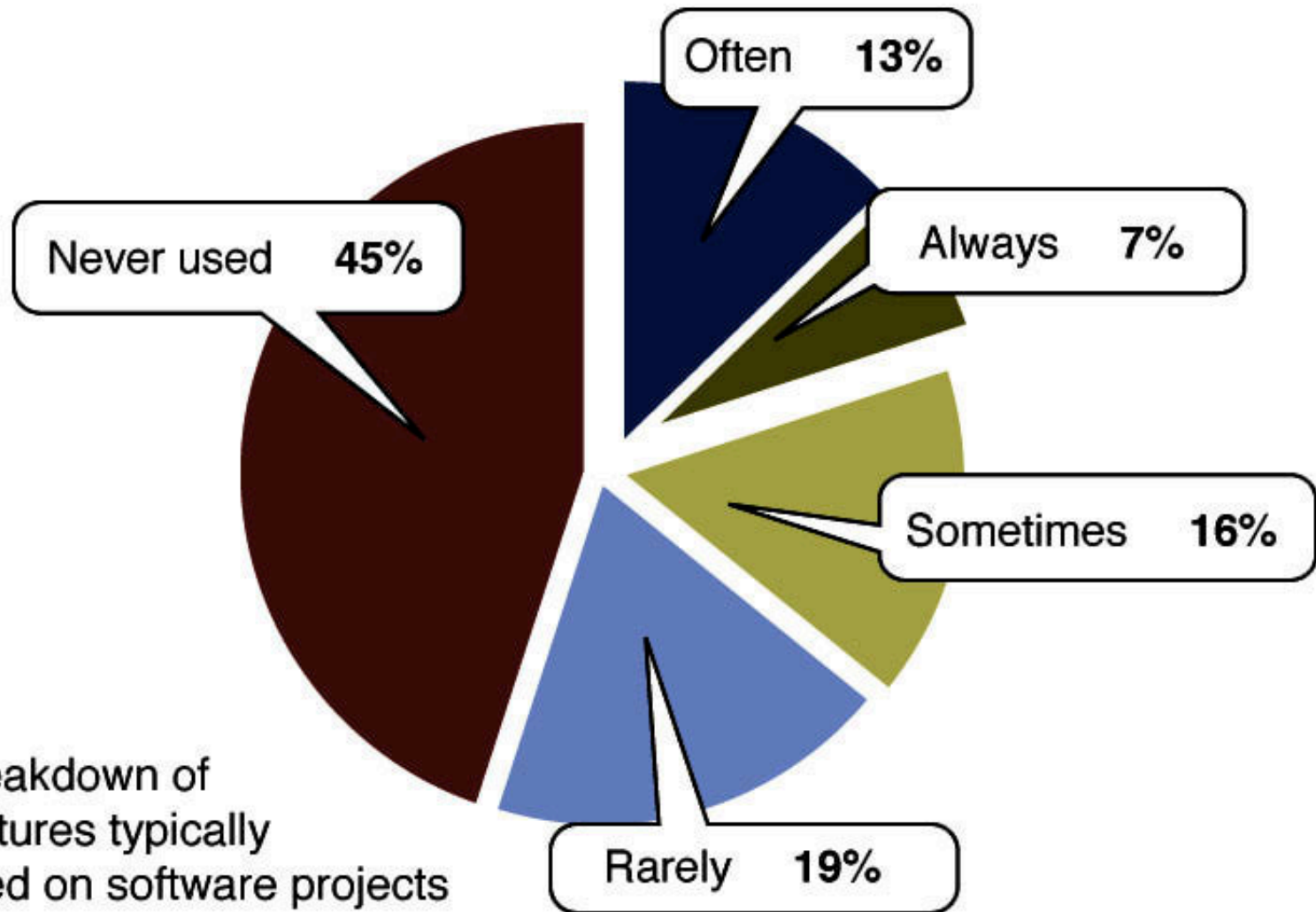
**Customers don't know what's  
possible. Most have no idea  
about the enabling  
technologies involved**

**Marty Cagan**

**You can't just ask customers  
what they want and then  
try to give that to them.  
By the time you get it built,  
they'll want something new.**

**Steve Jobs**

# Featuritis



***IF WE DON'T MEET THESE REQUIREMENTS ...  
WE ARE ALL GOING TO DIE!!!***



# Three Key Take-Aways

- Customer don't know what they want until you show it to them – this requires **fast experimentation**
- Increasing **SPEED** gives you the short cycle times that allow for fast experimentation
- Continuous deployment allows you to organize R&D as **innovation experiment system**

# Overview

- Vem är jag? Wie ben ik? Who am I?
- Trends in Software: Need for Speed
- Innovation Experiment Systems
- Legacy systems: case study
- Case study: Open Innovation Lab
- Does this apply to me?
- Conclusion

# From Research to Industry



**Academia  
(+ consulting)**



**Industrial  
research**







**Industrial  
development**

**Innovation**





# Software Center @ Chalmers

- Mission: Improve the software engineering capability of the Nordic Software-Intensive Industry with an order of magnitude
- Theme: Fast, continuous deployment of customer value
- Founding members
  - The Ericsson logo, featuring the word "ERICSSON" in blue capital letters next to three blue slanted parallel bars.
  - The Volvo logo, consisting of a silver circular emblem with a blue horizontal bar across the center containing the word "VOLVO" in white capital letters.
  - The Volvo logo, consisting of a silver circular emblem with a blue horizontal bar across the center containing the word "VOLVO" in white capital letters.
  - The Saab logo, featuring a blue circular emblem with a red crowned griffin head in the center and the word "SAAB" in white capital letters at the bottom.
- Dual success metrics
  - Academic excellence
  - Tangible industrial impact

# Overview

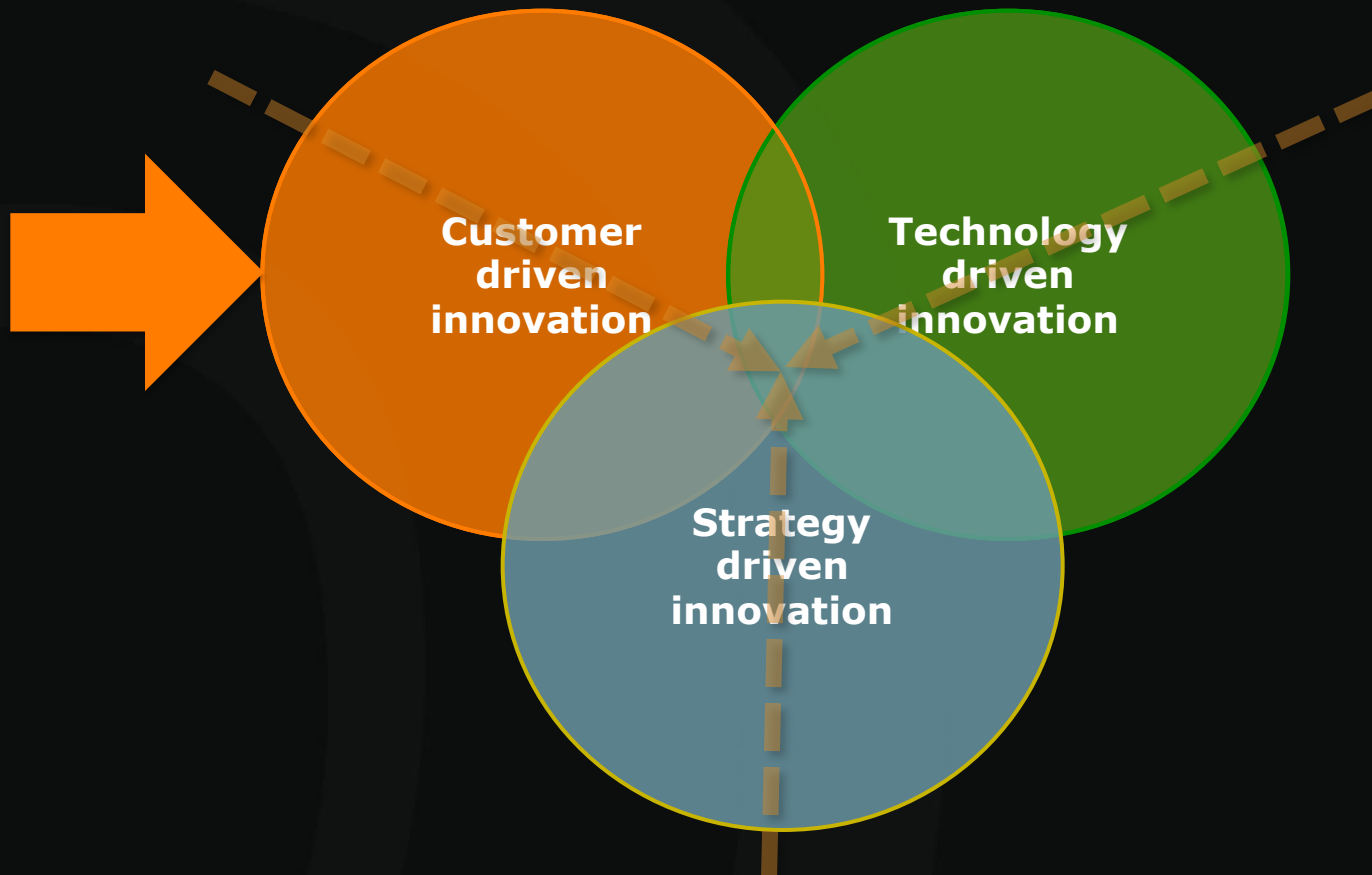
- Vem är jag? Wie ben ik? Who am I?
- Trends in Software: Need for Speed
- Innovation Experiment Systems
- Legacy systems: case study
- Case study: Open Innovation Lab
- Does this apply to me?
- Conclusion

# Trend: Products to Services

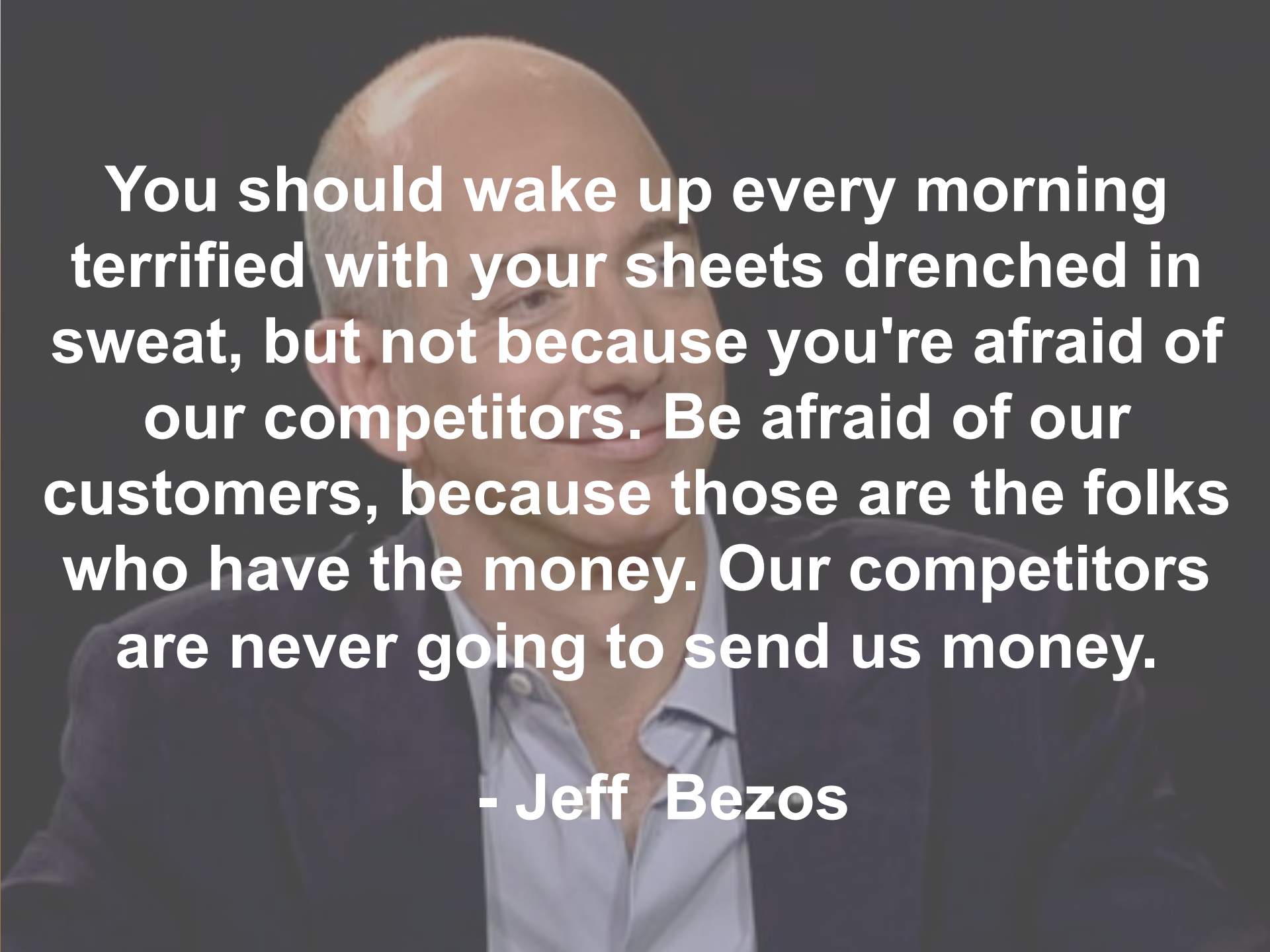


This requires continuous deployment throughout the lifetime of the product

# Innovation Approaches



This requires continuous experimentation with customers

A portrait of Jeff Bezos, the founder of Amazon, is shown in the background. He is a middle-aged man with a receding hairline, wearing a dark blue blazer over a light blue button-down shirt. He is looking slightly to his right with a faint smile. The background is a solid dark grey.

**You should wake up every morning terrified with your sheets drenched in sweat, but not because you're afraid of our competitors. Be afraid of our customers, because those are the folks who have the money. Our competitors are never going to send us money.**

**- Jeff Bezos**

# Trend: Need for Speed

## Value Creation Shifts

Emerging companies highlight importance of user contribution and social connectedness



Level of User Contribution

Founded	1984	1995	2004
1M users	~6 years	30 months	10 months
50M users	N/A	~80 months	~44 months



# Need for Speed in R&D – An Example

- Company X: R&D is **10%** of revenue, e.g. 100M\$ for a 1B\$ product
- New product development cycle: **12 months**
- Alternative 1: improve efficiency of development with 10%
  - **10 M\$** reduction in development cost
- Alternative 2: reduce development cycle with 10%
  - **100M\$** add to top line revenue (product starts to sell 1.2 months earlier)

**No efficiency improvement will outperform cycle time reduction**

# Need for Speed - Principles



## Team

- 2 pizza's
- self-selected, directed and managed
- quantitative output metrics



## Architecture

- simplicity – 3 API rule
- backward compatibility – no versions!
- focus on compositionality



## Release process

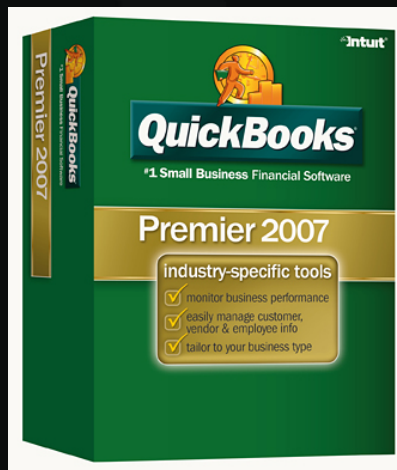
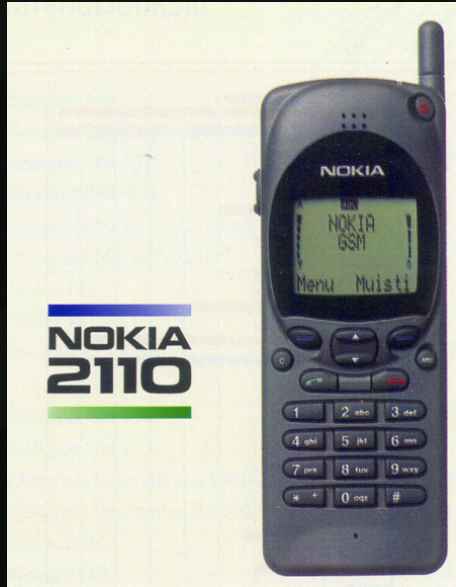
- continuous, independent deployment
- all the way to customers – installed base
- measure usage to feed back into development



# Overview

- Vem är jag? Wie ben ik? Who am I?
- Trends in Software: Need for Speed
- Innovation Experiment Systems
- Legacy systems: case study
- Case study: Open Innovation Lab
- Does this apply to me?
- Conclusion

# What Do These Product Have in Common?



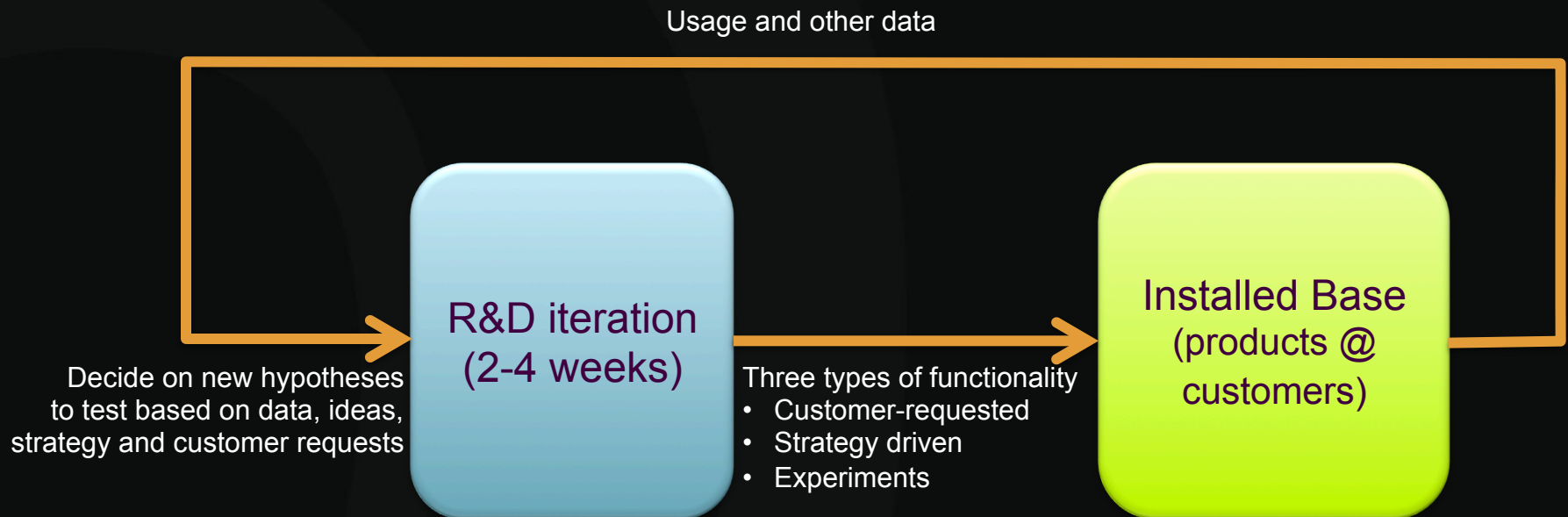
# Example: Apple

The Myth	The Reality
Inspired innovation	Create and winnow 10 pixel-perfect prototypes
Inspired design	Build a better backstory (intricate layers of business design behind the products)
Brilliantly inspired marketing	Engineer the perfect customer experience to create customer experience and buzz


# R&D as an Experiment System

**Learning: the company running the most experiments against the lowest cost per experiment wins**

Goal: increase the number of experiments (with customers) with an order of magnitude to ultimately accelerate organic growth



**Decisions should be based on DATA, not opinions**

A blurred background image of Ray Ozzie, a man with white hair and glasses, wearing a dark suit and a light-colored shirt. He is smiling and looking slightly to the right. The image is semi-transparent, allowing the text to be overlaid clearly.

We have an unprecedented opportunity to run A/B tests with online users and **innovate more quickly** based on actual user response. Microsoft needs to **shift the culture** from planning the exact features to planning a set of possible features, and **letting customers guide us.**

- Ray Ozzie

# Stages and Techniques

	<b>Pre-Development</b>	<b>Non-commercial deployment</b>	<b>Commercial deployment</b>
Optimization	Ethnographic studies	Independently deployed extensions	Random selection of versions (A/B testing)
New features	Solution jams	Feature alpha In-product surveys	Instrumentation/ collecting metrics
New Products	Advertising Mock-ups BASES testing	Product alpha Labs website In-product advertising	Surveys Performance metrics

# Pre-Development: Advertising

- What
  - Market a non-existing product (e.g. AdWords) to measure market interest
- Variations
  - Land on “product under development page
  - Ask users to leave an email address
  - Require payment before informing customer
- To think about
  - Measure the conversion funnel
  - Consider A/B testing on your ad & pages

# Pre-Development: Solution Jam

- Goal: Get as early feedback on an idea or concept as possible
- Length: 1 day
- How:
  - Invite staff to jam
  - Request “pain statements” beforehand
  - Select 10-15 customers based on “pain statements”
  - Staff self-organizes into small teams (3-6 typical)
  - Teams develop mock-up solutions to selected “pain statement”
  - Customers provide feedback on the mock-ups
  - Teams present at end of day
  - Customers + PM&PD leaders select most promising concepts
- Watch out for
  - Opinions instead of data



# Development: Feature Alpha

- What
  - Release last stable release of product with one new feature (under development) to selected customers
- Variations
  - Operate the product on copy of customer data
  - Allow user to turn feature on or off
- To think about
  - NEVER risk customer data
  - Measure everything – test alternative implementations

# Development: Labs Site

- What
  - Create destination for customers to try out new potential products, extensions and mock-ups
- Alternatives
  - Labs destination embedded in product or at company level
  - Combine with 3<sup>rd</sup> party developer efforts
- To think about
  - Brand impact of immature products
  - Energizing the innovation muscle of the company

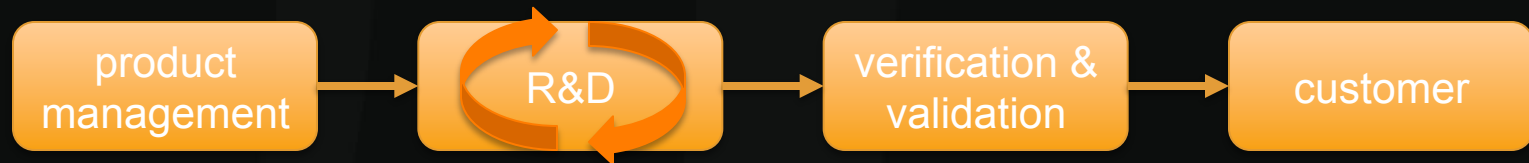
# Evolution: A/B Testing

- What
  - A/B testing is a method of comparing a baseline control sample to a variety of single-variable test samples for improving some metric
- Alternatives
  - “Marketing” testing, e.g. colors, buttons and order of options
  - Alternative implementations of a feature
- To think about
  - Run multiple experiments simultaneously
  - Verify statistical relevance (free online tools exist)

# A/B Testing Examples

- 37signals tested the headline on its pricing page. It found that “30-Day Free Trial on All Accounts” generated 30% more sign-ups than the original “Start a Highrise Account.”
- Dustin found that “You should follow me on Twitter here” worked 173% better than his control text, “I’m on Twitter.”
- A surprising conclusion from two separate A/B tests: putting human photos on a website increases conversion rates by as much as double.
- CareLogger increased its conversion rate by 34% simply by changing the color of the sign-up button from green to red.
- A software product company redesigned their product page to give it a modern look and added trust building elements (such as seals, guarantees, etc.). End result: they managed to increase total sales by 20%.

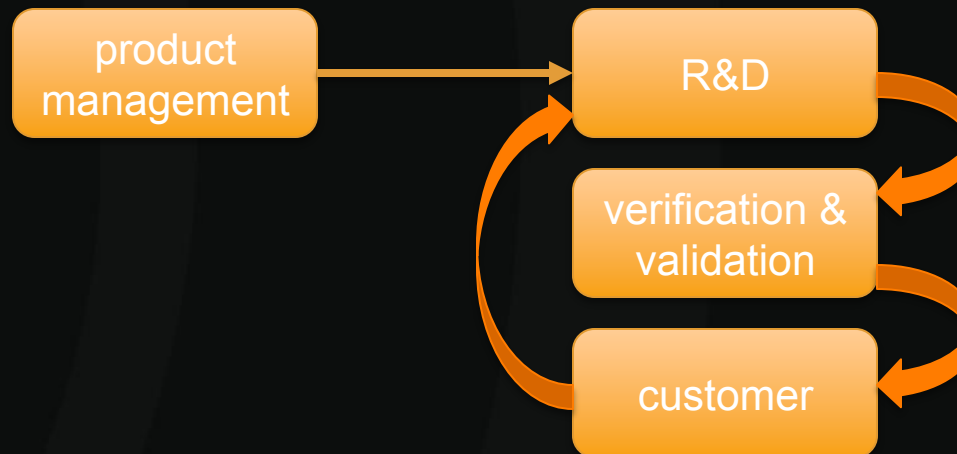
# Stairway to Heaven



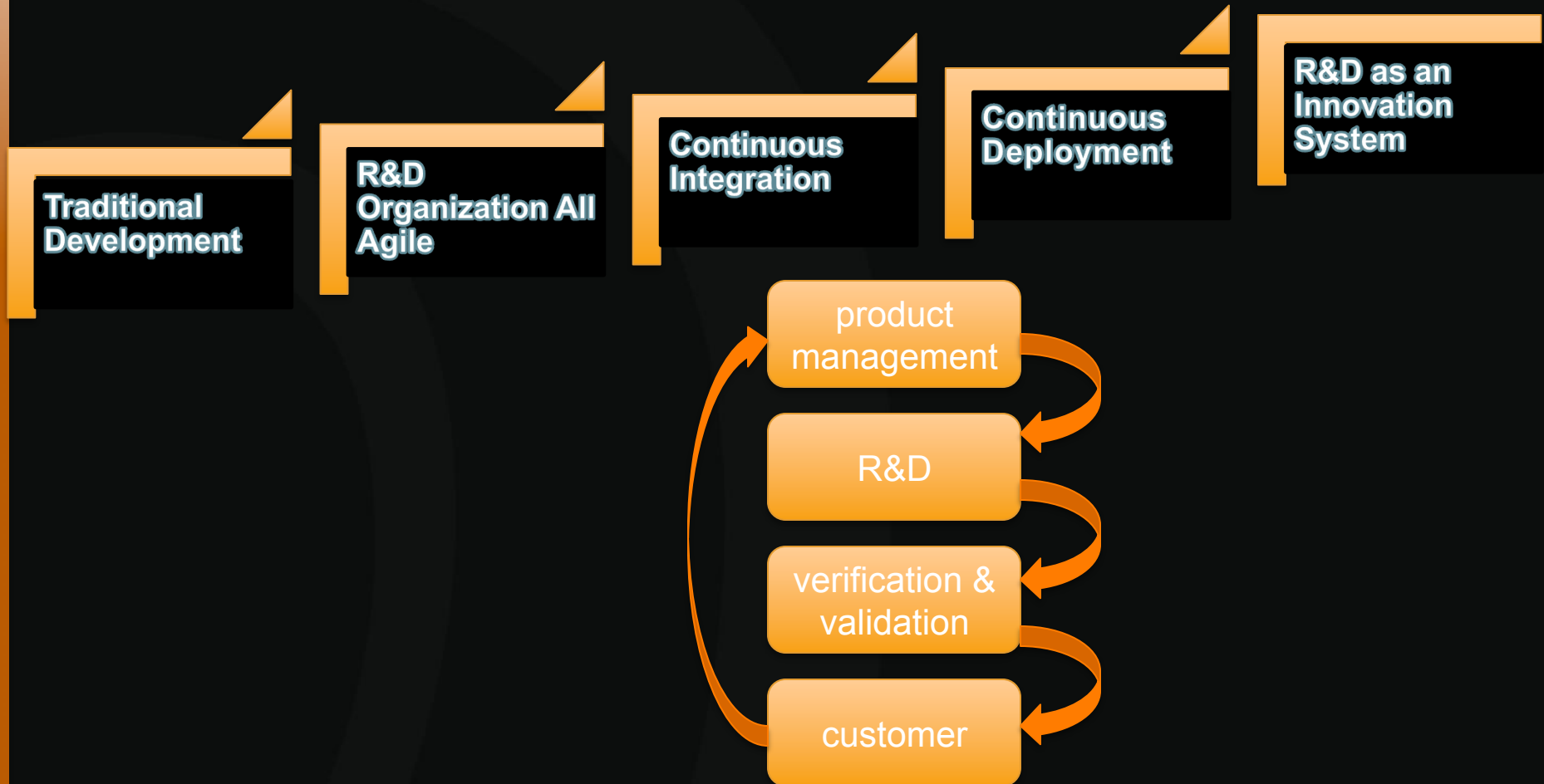
# Stairway to Heaven



# Stairway to Heaven

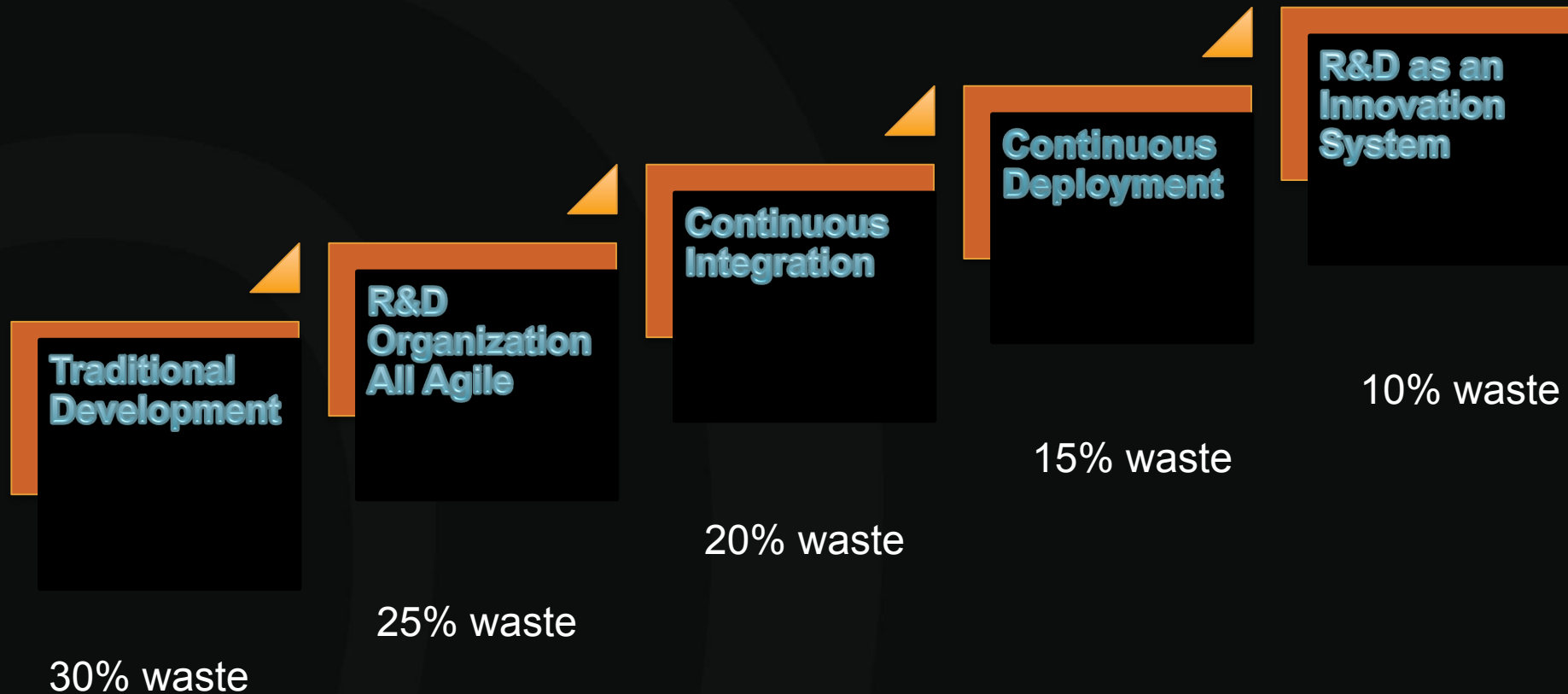


# Stairway to Heaven





# Stairway to Heaven



Rough estimation of waste and benefits

# Financial Impact Potential

## Ericsson

- R&D budget July 2011 – June 2012: 4,864 M\$
- Software R&D (80%): 3891 M\$
- Value of removing 5% waste:  
**195 M\$ (1280 MSEK)**

## AB Volvo

- Revenue 2011: 310 BSEK
- R&D budget 2011 (est. 5%):  
16 BSEK
- Software R&D (25%):  
4 BSEK
- Value of removing 5% waste: **200 MSEK**

# Overview

- Vem är jag? Wie ben ik? Who am I?
- Trends in Software: Need for Speed
- Innovation Experiment Systems
- Legacy systems: case study
- Case study: Open Innovation Lab
- Does this apply to me?
- Conclusion

# Intuit Quickbooks

[Website Services](#)[QuickBooks](#)[Payroll](#)[Payment Solutions](#)[Point of Sale](#)[Checks & Supplies](#)[US](#)[QuickBooks Overview](#)[Products & Services](#)[Industry Solutions](#)[Training & Learning](#)[Community](#)[Support](#)[Interested? Call Sales](#)[\(877\) 683-3280 M-F 5am-6pm PT](#)[Quick Survey](#)

## Save Time. Stay Organized. Get Paid Faster.

### New to QuickBooks?

Try QuickBooks Online for free!



**Less Paperwork**  
Instant invoicing and tracking



**Get Paid Faster**  
Automate customer payments



**Free Support & Upgrades**  
In all online versions

**Start Now. It's FREE**

[Learn More](#)

### Quickbooks

- Age: 20 years old
- Size: 5-10 MLOC
- Org: 100+ R&D staff

### Upgrade?

Level  
Customization tools



**2010 features help you get organized and keep accurate records for tax time!**

- Document Management
- Company Snapshot
- Invoice Customization
- Report Center

[See more new features](#)

QuickBooks pays for itself in the first 60 days - guaranteed on all [QuickBooks Products](#) or your money back.<sup>1</sup>

# Old Development Process: Yearly Releases



September: Release

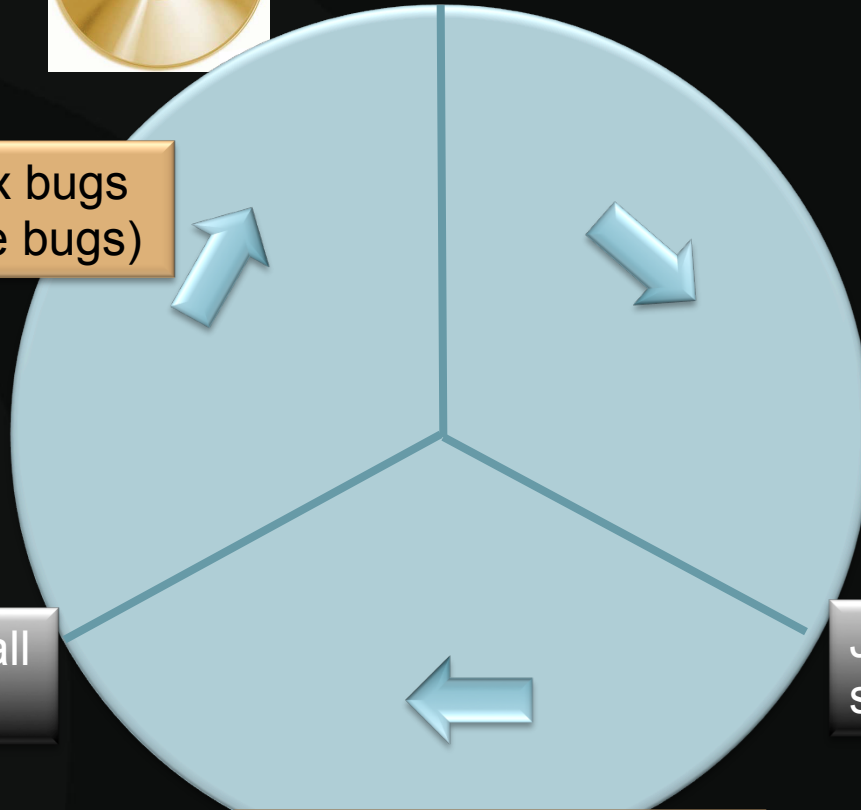
Get a beta out and fix bugs  
(and then some more bugs)

Figure out what to  
build

May/June: freeze all  
development

Jan/Feb: Development  
starts for real

Build the key new  
features



# Old Development Process: Problems

Lack of customer feedback



Heavy, top-down development process



Inefficient use of engineering resources



Low engagement of team members



# New Process: Overview

**Top Down: Strategic Areas to Improve**

~50% acceptance    ~50% acceptance

**Solution  
Jam**

**Code  
Jam**

**1..n  
iteration**

**Feature  
Alpha**

**Product  
Release**

1 day  
Team self selected  
Customer pain point  
hypothesis  
10-15 customers  
present  
Develop mock-ups of  
solution

2 days  
Build skeleton  
implementation  
Evaluate for technical  
difficulties  
No customers present

2 (3) week iteration  
Build "minimal viable  
product"  
Weekly (or more often)  
interaction with  
customers  
Test a specific hypothesis

# Key Characteristics

- Teams are self-selected, self-directed and self-managed
- Teams are small (typically 3 people, but 1-6 is the range)
- Data instead of opinions: Customers are deeply involved in the process
- Three stages:
  - Concept testing: immediate customer feedback
  - Prototype testing: sandboxed release of system that does not disrupt the customer's data
  - Feature alpha: old release with new feature included at customer site



# Overview

- Vem är jag? Wie ben ik? Who am I?
- Trends in Software: Need for Speed
- Innovation Experiment Systems
- Legacy systems: case study
- Case study: Open Innovation Lab
- Does this apply to me?
- Conclusion

# Case: Open Infotainment Labs



# Case: Open Infotainment Labs

- Feature development from a nominal lead-time of 1-3 years to 4-12 weeks?
- Working software was continuously validated in “real” environments
  - installed in both a driving simulator and real test cars
  - users evaluated the system
- 4th sprint: A/B experiment
  - Evaluating two layouts of the start screen
    - Implemented as two different launchers in Android
  - Mounted in a vehicle
  - 7 test drivers in total (3 used A, 4 used B)

# Overview

- Vem är jag? Wie ben ik? Who am I?
- Trends in Software: Need for Speed
- Innovation Experiment Systems
- Legacy systems: case study
- Case study: Open Innovation Lab
- Does this apply to me?
- Conclusion

# Experimentation beyond UX

- Any quality attribute can be experimented with!
- Process
  - Team defines the area of experimentation
  - Build feature thought to improve a user perceived quality
  - The response variable(s) are selected and implemented
  - The software is deployed to a statistically relevant number of devices
  - The data is uploaded and analyzed through the infrastructure
  - The development team draws conclusions about the new software

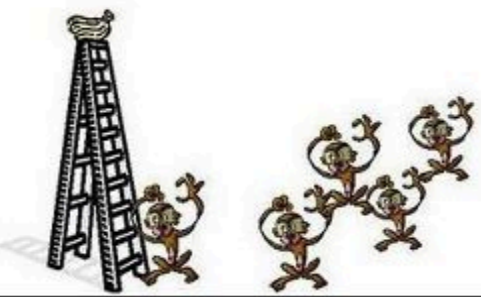
A group of scientists placed 5 monkeys in a cage and in the middle, a ladder with bananas on the top.



Every time a monkey went up the ladder, the scientists soaked the rest of the monkeys with cold water.



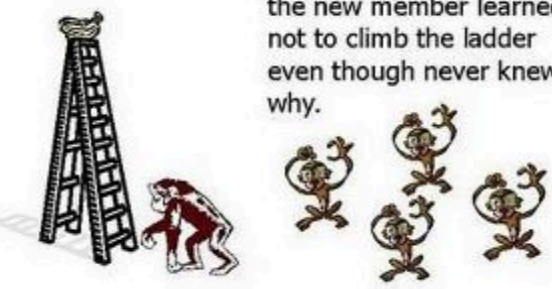
After a while, every time a monkey went up the ladder, the others beat up the one on the ladder.



After some time, no monkey dare to go up the ladder regardless of the temptation.

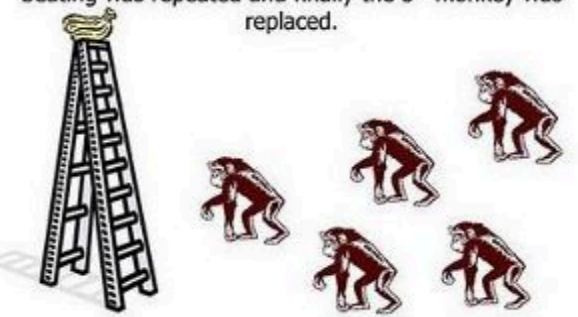


Scientists then decided to substitute one of the monkeys. The 1<sup>st</sup> thing this new monkey did was to go up the ladder. Immediately the other monkeys beat him up.



After several beatings, the new member learned not to climb the ladder even though never knew why.

A 2<sup>nd</sup> monkey was substituted and the same occurred. The 1<sup>st</sup> monkey participated on the beating for the 2<sup>nd</sup> monkey. A 3<sup>rd</sup> monkey was changed and the same was repeated (beating). The 4<sup>th</sup> was substituted and the beating was repeated and finally the 5<sup>th</sup> monkey was replaced.



What was left was a group of 5 monkeys that even though never received a cold shower, continued to beat up any monkey who attempted to climb the ladder.



If it was possible to ask the monkeys why they would beat up all those who attempted to go up the ladder....  
I bet you the answer would be....

**"I don't know – that's how things are done around here"**

Does it sounds familiar?



Don't miss the opportunity to share this with others as they might be asking themselves why we continue to do what we are doing if there is a different way out there.





# Shadow Beliefs

- Humans are better than machines in identifying known and new reliability issues – we are building business critical systems, after all!

My experience: data always trumps opinion; test and validation systems pre-deployment and extensive data-collection post-deployment inform decision making

- Software-intensive systems (large, complex, tough requirements) are different and approaches from other domains do not apply

My experience: system failure is devastating in several industries and avoided in Internet systems while adopting agile and continuous deployment

- We should avoid or delay adoption of new, more efficient engineering approaches

My experience: getting first to market with new functionality that closely aligns to customer needs is a significant competitive advantage that drives growth and results in market leadership

# Overview

- Vem är jag? Wie ben ik? Who am I?
- Trends in Software: Need for Speed
- Innovation Experiment Systems
- Legacy systems: case study
- Case study: Open Innovation Lab
- Does this apply to me?
- Conclusion



# Customers Don't Know



- Customer don't know what they want until you show it to them – this requires **fast experimentation**
- Test continuously with customers that you're on the right track
- Apply the right technique for the right stage and problem
- Focus on small, cross-functional teams, give them direction and get out of their way

# Speed

- Increasing **SPEED** gives you the short cycle times that allow for fast experimentation
- If you're not a front-line engineer, there is only ONE measure that justifies your existence: how have you helped teams move faster?
- Don't optimize efficiency, optimize speed



# Innovation Experiment Systems

- Continuous deployment allows you to organize R&D as **innovation experiment system**
- Legacy system != slow (necessarily)
- Traditional working methods no longer apply
- Decouple components, decouple teams and decouple organizations
- Lean and agile at scale

# Not My Job?!



Strong LEADERSHIP needed from YOU

**Jan@JanBosch.com**

**Chalmers/Göteborg University**