Agility at Scale

Rapid learning for humans in 1000 Squads, across the world, in a heavily regulated industry

Henk Kolk, Chief Engineer, ING

RDW TECHDAY, Groningen, March 14th, 2017

Rules don’t always apply

Context: only you can own yours

Agility*) is …

1. assess where you are
2. take a small step
3. evaluate
4. repeat

*) As intended in the Manifesto for Agile Software Development, 1999
According to “Pragmatic” Dave Thomas (“Agile is Dead” - Goto talk)

Agile Software Development is recycling some pretty solid (old) ideas

- **Scientific Method**
  Francis Bacon (1620)

- **Plan – Do – Study – Act loop of Total Quality Management**
  Shewart (1939) and W. Deming (1950)

- **The New New Product Development Game**
  Hirotaka Takeuchi and Ikujiro Nonaka (1986)

Outcomes over Impositions

Jeff Sussna, Designing Delivery

What is the

problem

that we’re trying to solve?

Software

is

eating the world

Marc Andreessen

Speed

is

market share

Adrian Cockroft

Platforms eat Pipelines

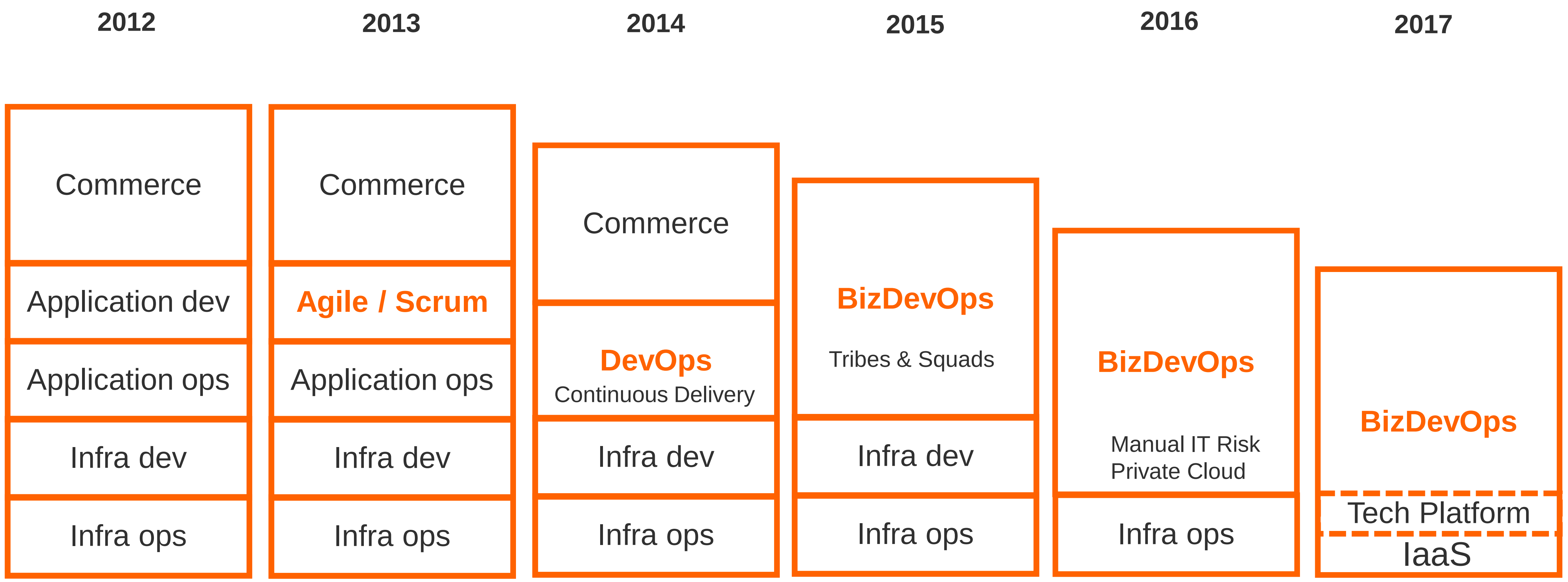
Ron Kersic, ING

The platform business model underlies the success of many of todays biggest, fastest growing and most disruptive companies. From Google, Amazon and Microsoft to Uber, Airbnb and eBay.

Platforms …

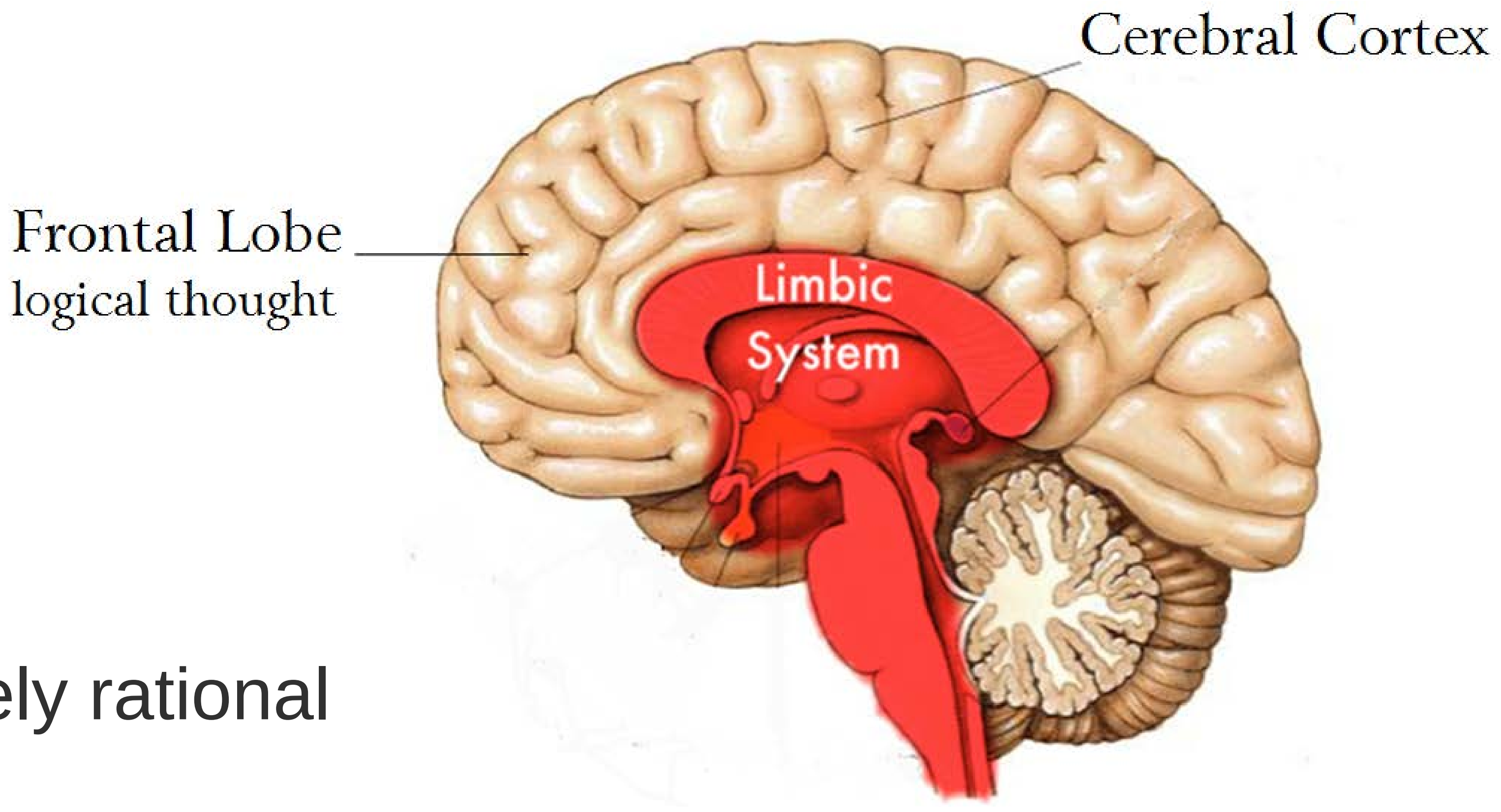
- Serve an **ecosystem** of **external producers and consumers**
- **Unlock new sources of value creation** and **supply**
- **Eliminate gatekeepers** to scale efficiently
- Run on a **not-even-mine inventory**
- Create **community feedback loops**
- **Are designed for global scalability**

In the past five years, ING has been reorganizing for speed and skill. Roles and responsibilities have shifted radically



Engineer: From single discipline to **full stack engineers**: designing, coding, test engineering, infra engineering, etc
Product Owner: From writing PIDs to product vision and backlog to **end to end bizdevops responsibility**
IT Manager: from delivery manager to perhaps the most differntiating role: **skill and competency coach**.

What makes agility hard to achieve?



We are human

- Perhaps not entirely rational
- that's OK

Direct the rider, motivate the elephant, shape the path

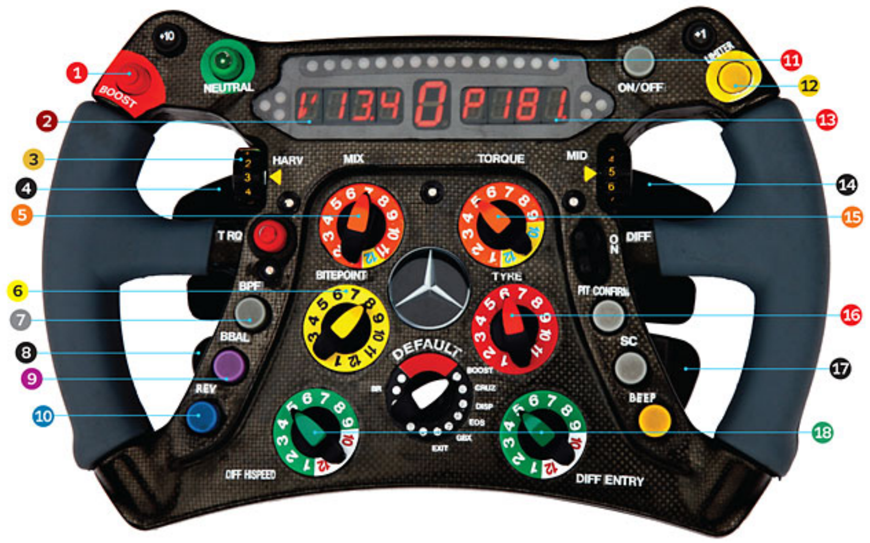
Just like Max Verstappen, we can only achieve high velocities if we are in control

ING

Technology has evolved to improve control at higher speeds

1990 Steering Wheel (McLaren)

2010 Steering Wheel (Mercedes)



We must start by leveraging todays technology, language, way of thinking and todays technology best practices for IT Control to assure ourselves and our auditors that we are in control

“BizDevSecRiskOps”

- Control Framework shifts left
- IT Risk is controlled 95% by design

Envisioned outcome

- CIRM / CORM set the conditions for teams to be in control
- Control is auditable, globally

Shift-Left

John Sharratt, Jon Lee, ING

Shift Left

1. Design Thinking (solve the right problem)

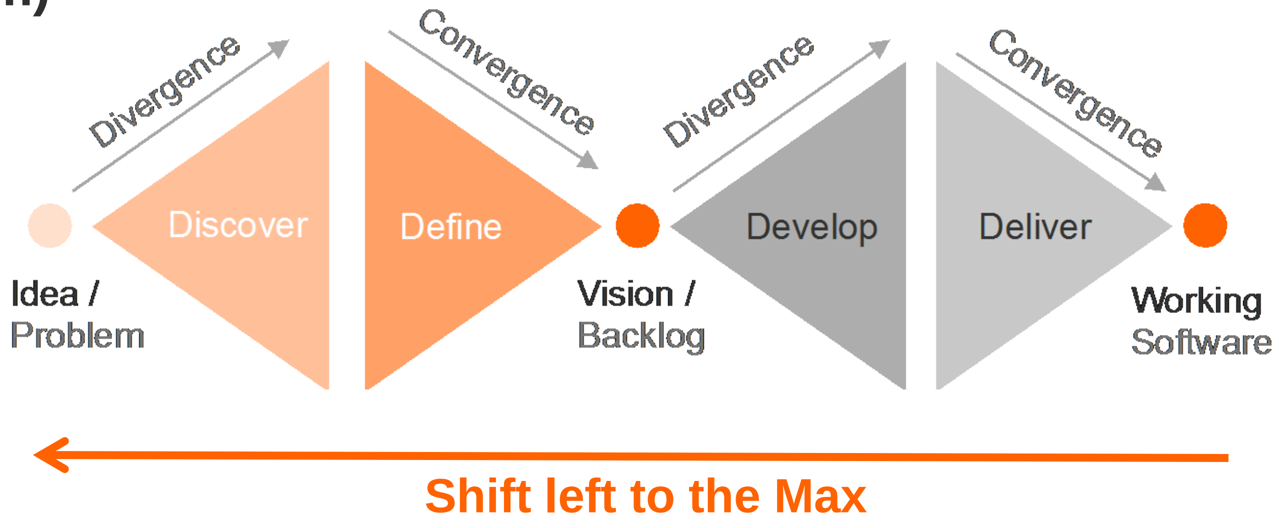
- Discover & define the right problem
- Define the right vision
- Design the right solution
- Test your vision as early as possible
- Test your design as early as possible

2. Hypothesis Driven Development (Francis Bacon)

- Every “requirement” is just a hypothesis
- Every hypothesis needs to be proven by data

3. Build quality in (W. Deming)

- Problem prevention over detection,
- Begin testing earlier than ever before



The more we shift the problem to the left, the cheaper it is to solve the problem and the faster we go

Humans vs Robots

Isaac Azimov

Immutable Servers

Martin Fowler

ING

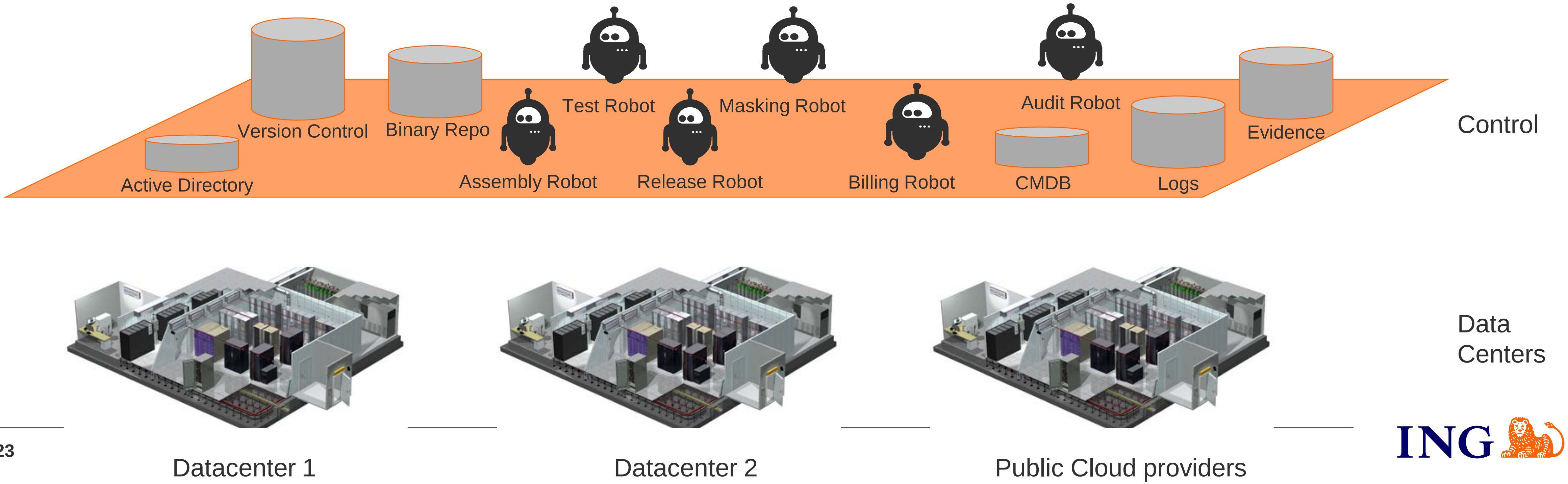
Cattle and Pets

John Sharratt, Jon Lee

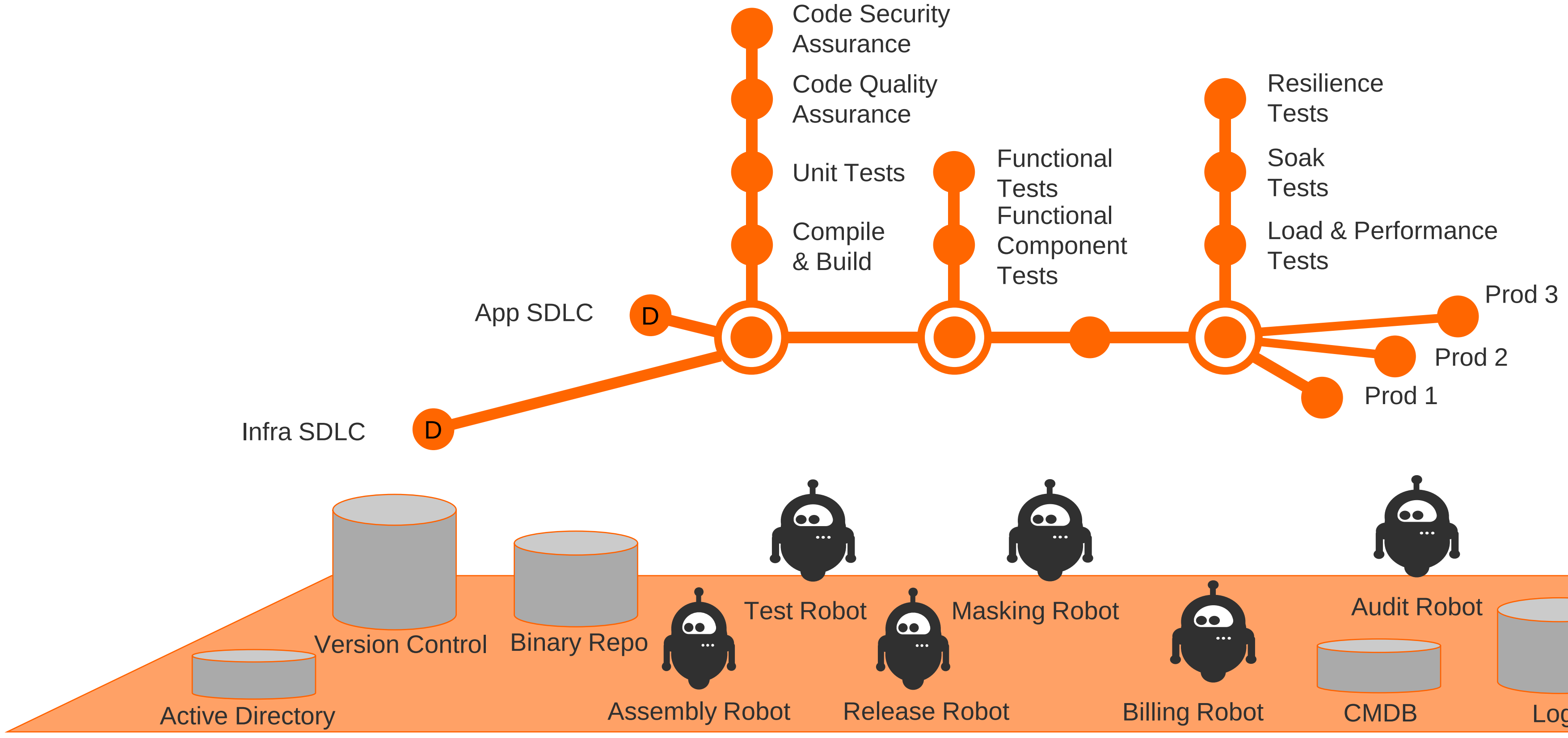
Infrastructure = code

John Sharratt, Jon Lee

Robots (=software) are taking center stage in release engineering (of software)



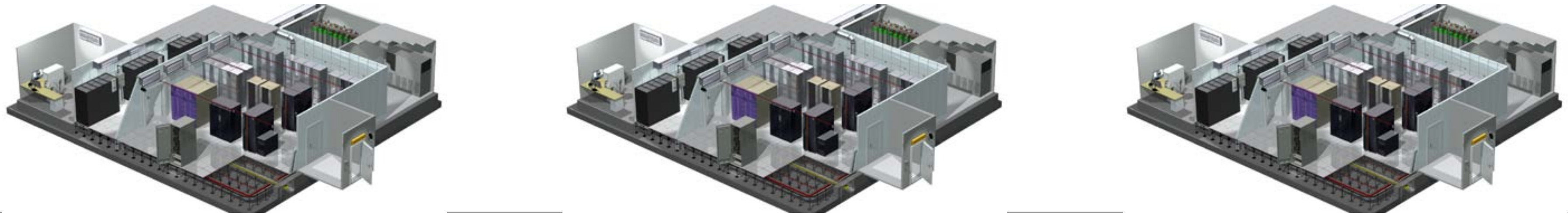
Humans leverage automated pipelines to deliver software



Continuous Delivery

Integrated for apps and infra

Control



Data Centers

Datacenter 1

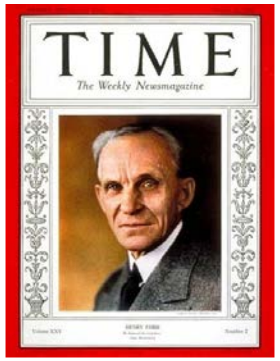
Datacenter 2

Public Cloud providers

Learning organization

Shitty assumptions about software engineers

- Business Architect
- Solution Architect
- Requirements Specifier
- Designer
- Coder
- Tester
- Deployer
- Chief Engineer ☺ →



ING radically changed its HR strategy to allow for the skill growth and creative liberation of multi skilled engineers

Expert

Proficient

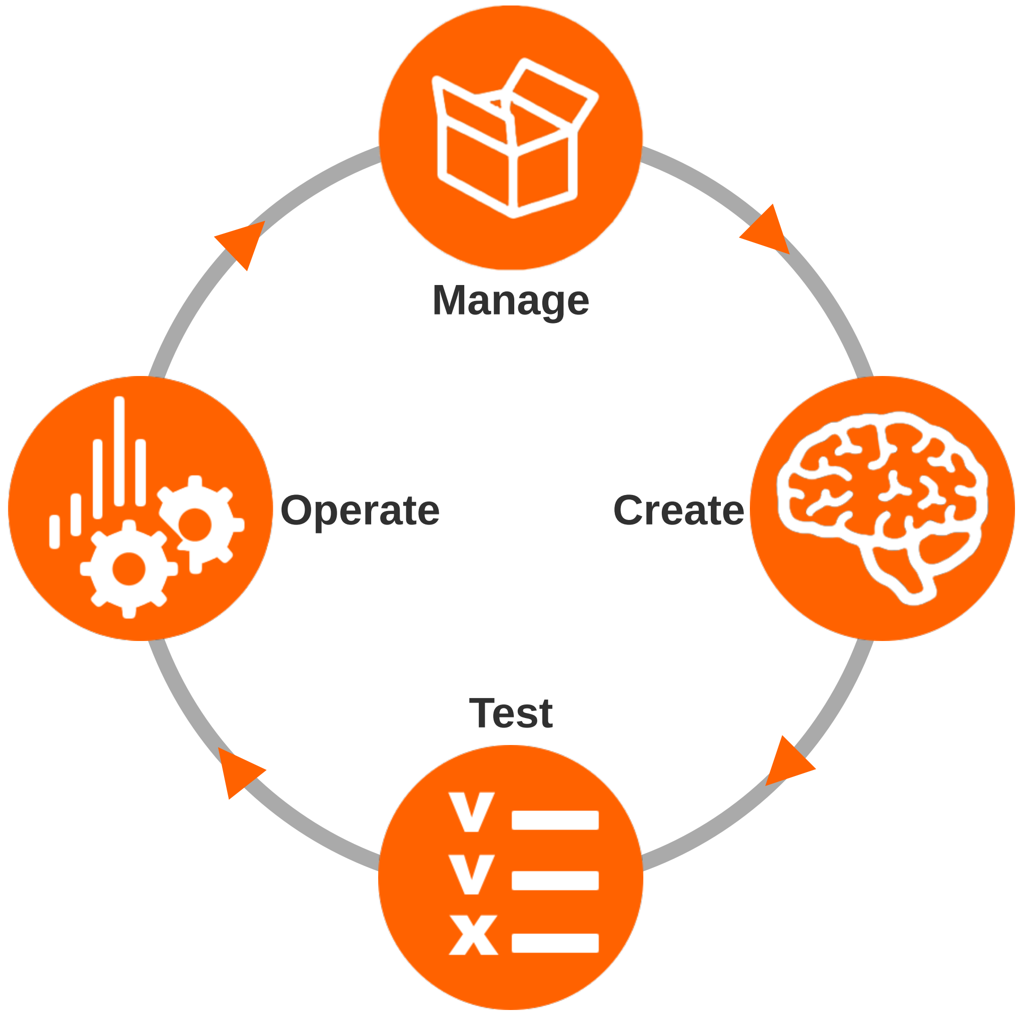
Competent

Advanced Beginner

Novice

Nicked from Andy Hunt
Please check out his GROWS method
Andy is onto something so right

INGs Way of Working is supported by an “Technology Platform”, that is designed as a giant Plan-Do-Study-Act (PDSA) cycle, adapted for agile software product management at scale.



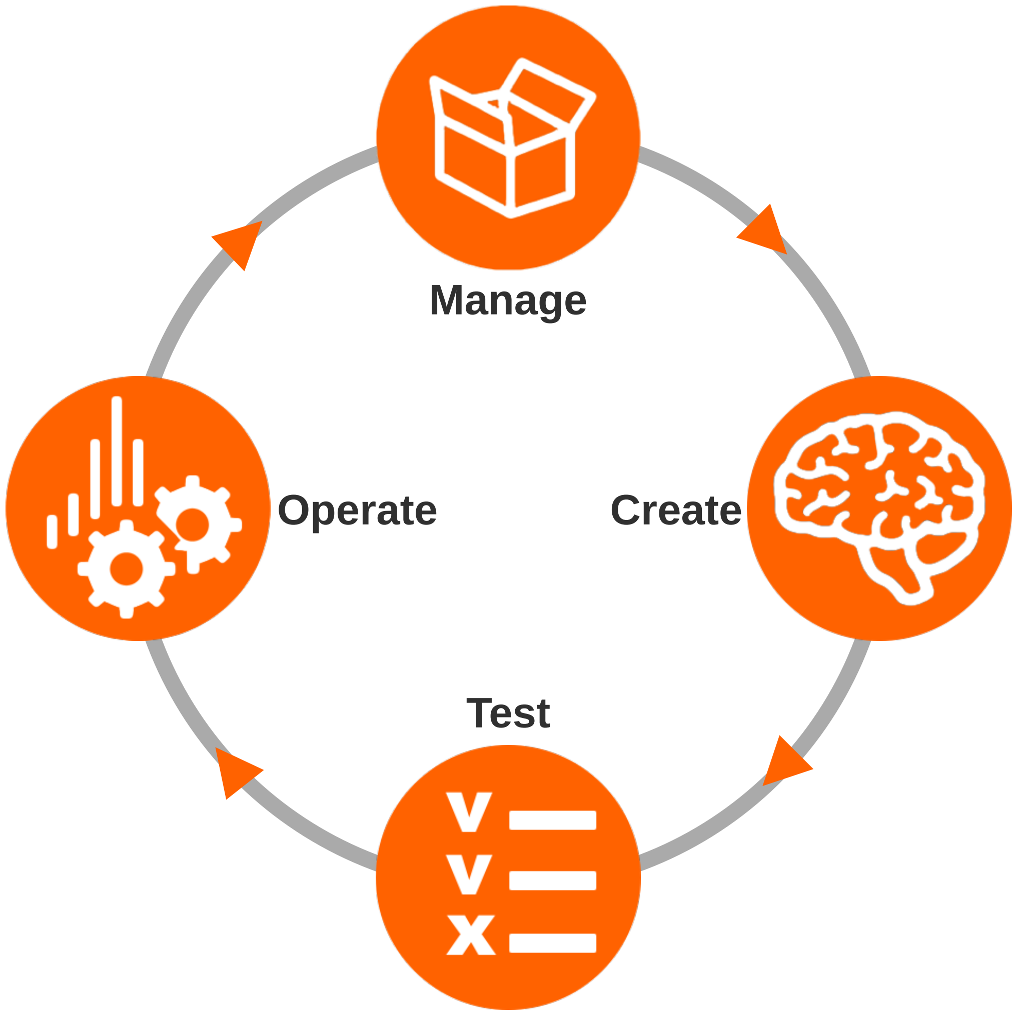
Manage (Plan) - Digital product management. e.g. portfolio management, key objectives and results (OKRs), backlog management, IT service management, virtual Obeya rooms (feedback on strategic results)

Create (Do) - design, share (knowledge management), engineering laptops, continuous integration (version control, build services, binary repositories), release engineering (assembly - and deployment robots)

Test (Study) – test engineering: control framework, feedback loops, test environments, integrated test environments, test data, data masking, unit testing, functional testing, integration testing, code quality testing, code security testing, pen testing, load testing, performance testing, resilience testing, test robot, audit robot, evidence repository

Operate (Act) - Monitoring, Alerting, Paging, Master Control Room, ChatOps, Run-time Immune Systems, Site Reliability Engineering

Wouldn’t it be great if somebody provided all European government agencies with a creative platform like this?



In the words of W. Edwards Deming

“Learning is not compulsory... neither is survival.”

Be(come) **AWESOME** everybody

Twitter: @henkkolk
E-mail: henk.kolk@ing.nl

We’re hiring

Some suggestions for further reading …

HBR The New New Product Development Game
Lean Enterprise: How High Performance Organizations Innovate at Scale
Jez Humble, Joanne Molesky & Barry O'Reilly
The Scrum Guide™
The Lean Startup
Eric Ries
Bohm On Dialogue
Designing Delivery: Rethinking IT in the Digital Service Economy
Jeff Sussna
Leading from the Emerging Future: From Ego-System to Eco-System Economies
Otto Scharmer and Katrin Kaufer
The Innovator's Dilemma: When New Technologies Cause Great Firms to Fail
Tribal Leadership: Leveraging Natural Groups to Build a Thriving Organization
Dave Logan, John King & Halee Fischer-Wright
Introduction to OKRs
Christina Wodtke
Creativity, Inc.
Ed Catmull
The End of Competitive Advantage
Rita Gunther McGrath
Platform Revolution
Geoffrey G. Parker, Marshall W. Van Alstyne, Sangeet Paul Choudary
Escape Velocity: Free Your Company's Future from the Pull of the Past
Geoffrey A. Moore
This is Service Design Thinking.
Switch: How to Change Things When Change Is Hard
Chip Heath & Dan Heath
Sprint: How to Solve Big Problems and Test New Ideas in Just Five Days
Jake Knapp
Agile Testing: A Practical Guide for Testers and Agile Teams
Lisa Crispin, Janet Gregory
Continuous Delivery
Jez Humble, David Farley
Site Reliability Engineering: How Google Runs Production Systems
Edited by Betsy Beyer, Chris Jones, Jennifer Petoff & Niall Richard Murphy
The DevOps Handbook: How to Create World-Class Agility, Reliability, & Security in Technology Organizations
Gene Kim, Jez Humble, Patrick Debois, & John Willis
Extreme Programming Installed
Ron Jeffries, Ann Anderson, Chet Hendrickson
Extreme Programming Explained: Embrace Change
Kent Beck with Cynthia Andres
Second Edition
Release Engineering: How Google Builds and Delivers Software
Dinah McNutt, Betsy Beyer, Tim Harvey
Nassim Nicholas Taleb Antifragile: Things That Gain from Disorder
Drift into Failure
Sidney Dekker
How Google Tests Software
James Whittaker · Jason Arbon · Jeff Carollo

ING