CT70A2000 Requirements Engineering

Shola Oyedeji

LUT University

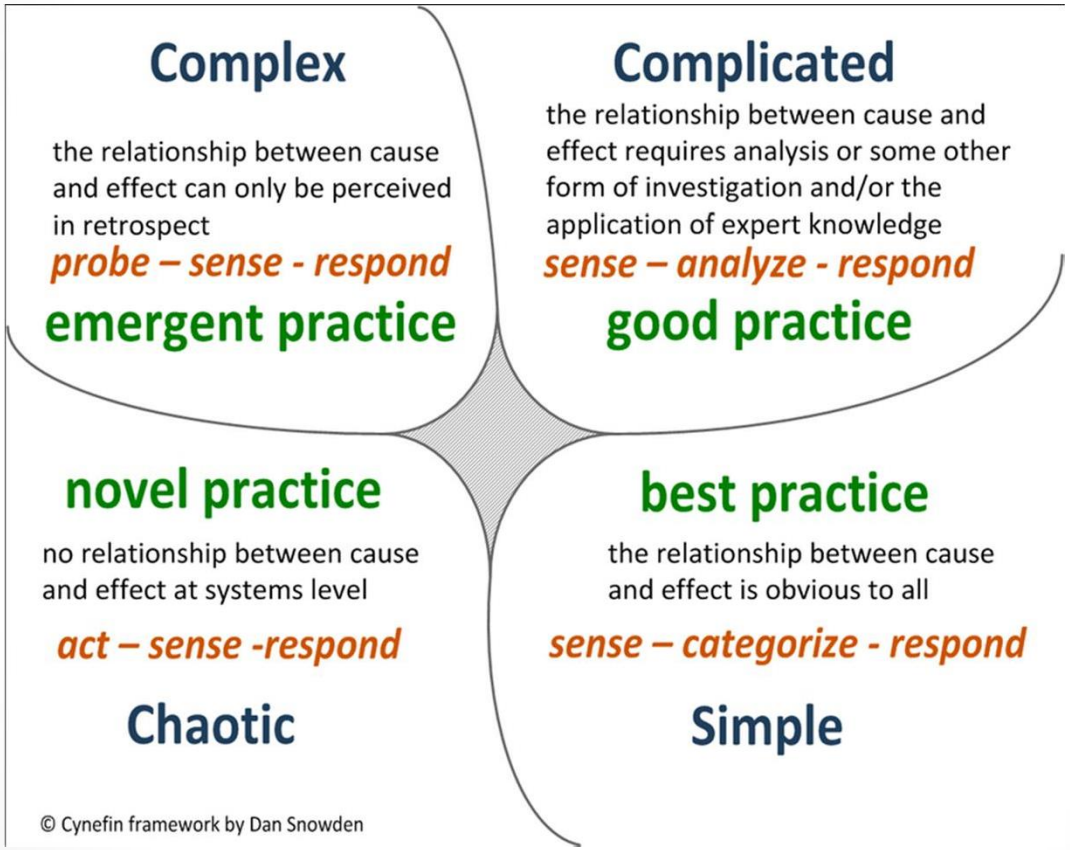
Lecture 9:
Towards present and future RE

LUT University

CYNEFIN FRAMEWORK

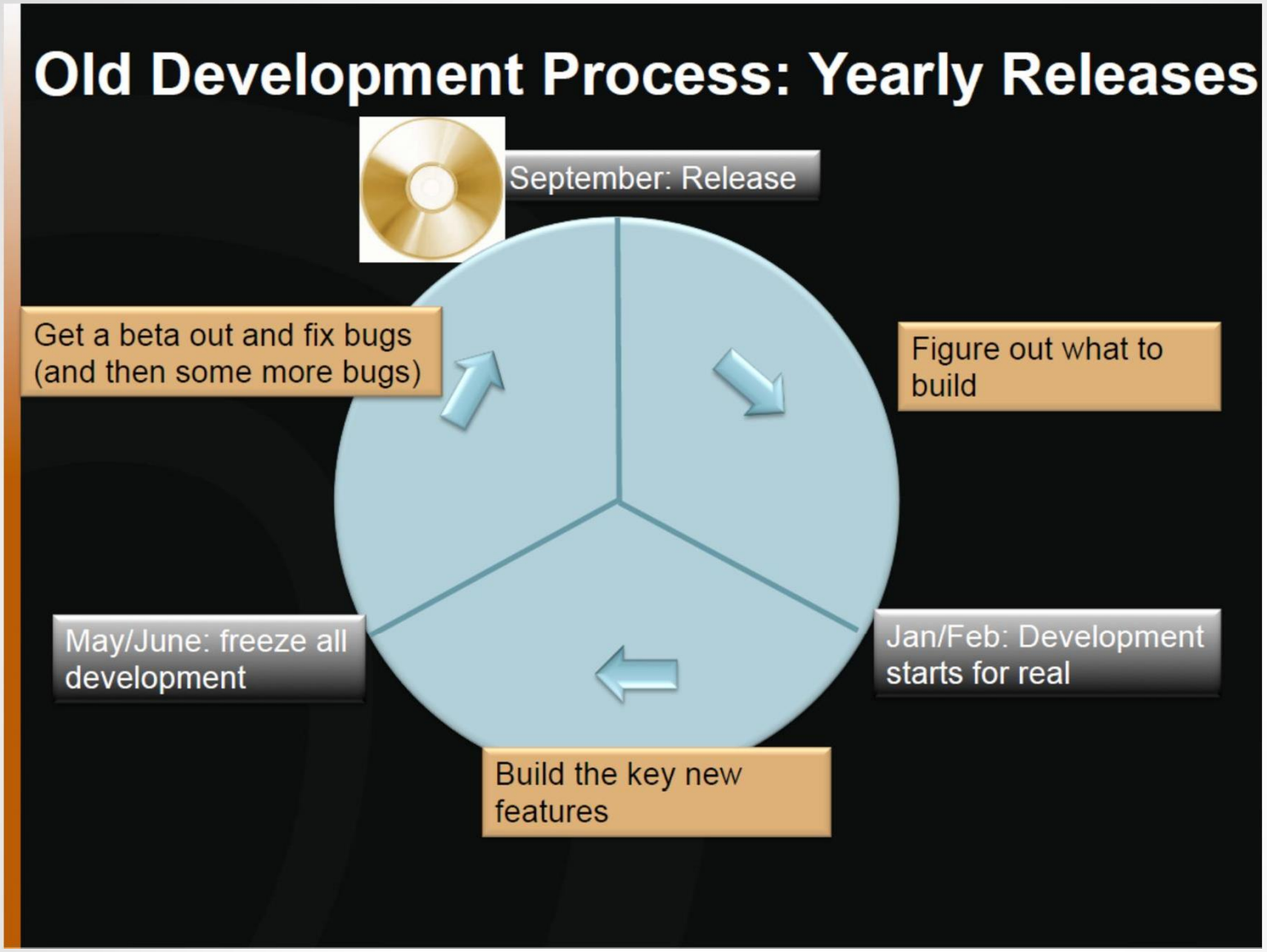
UNORDERED

ORDERED



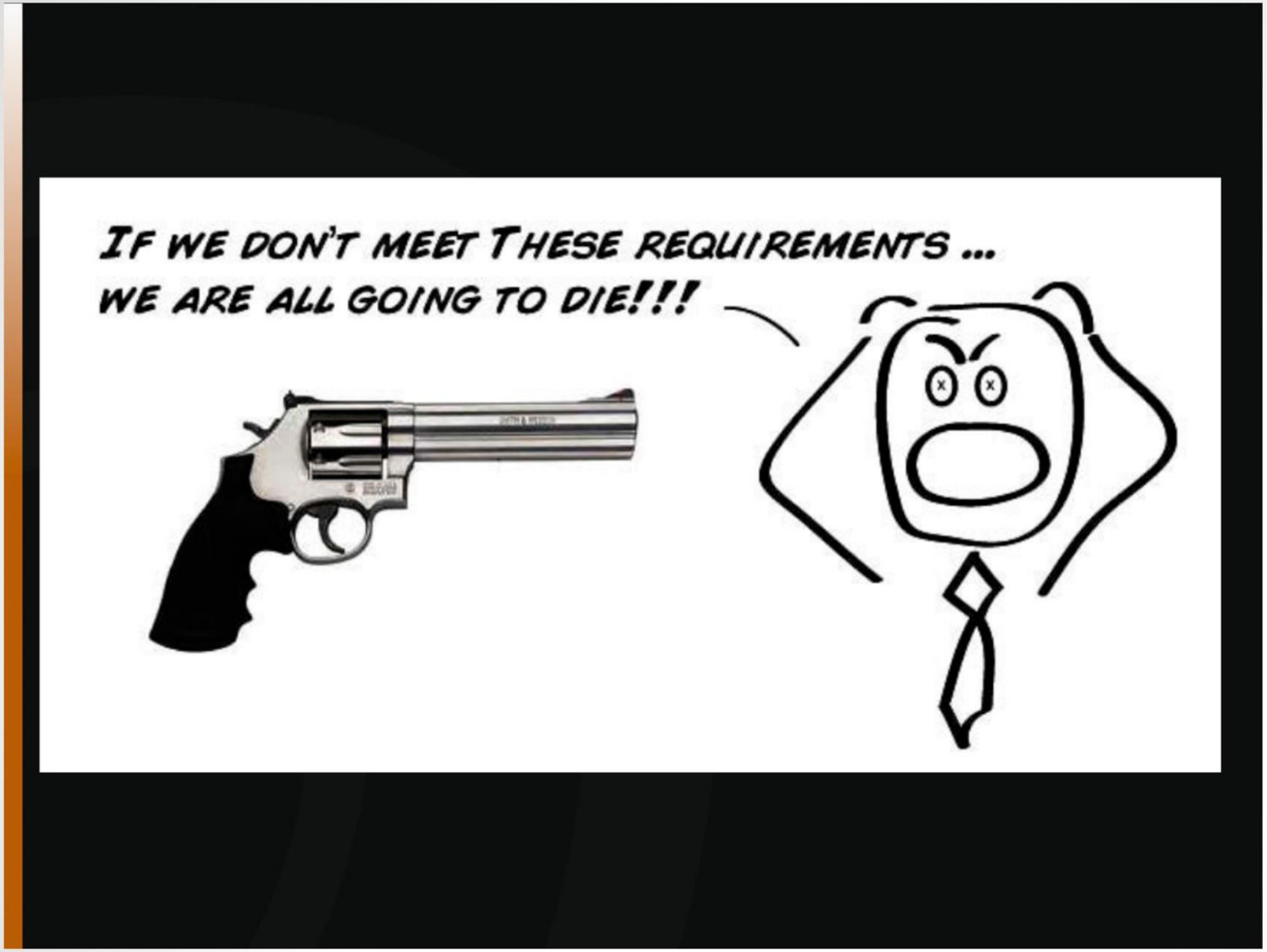
D. J. Snowden and M. E. Boone, "A Leader's Framework for Decision Making", *Harvard Business Review, vol. 85, 2007*
https://hbr.org/2007/11/a-leaders-framework-for-decision-making

Old Development Process: Yearly Releases

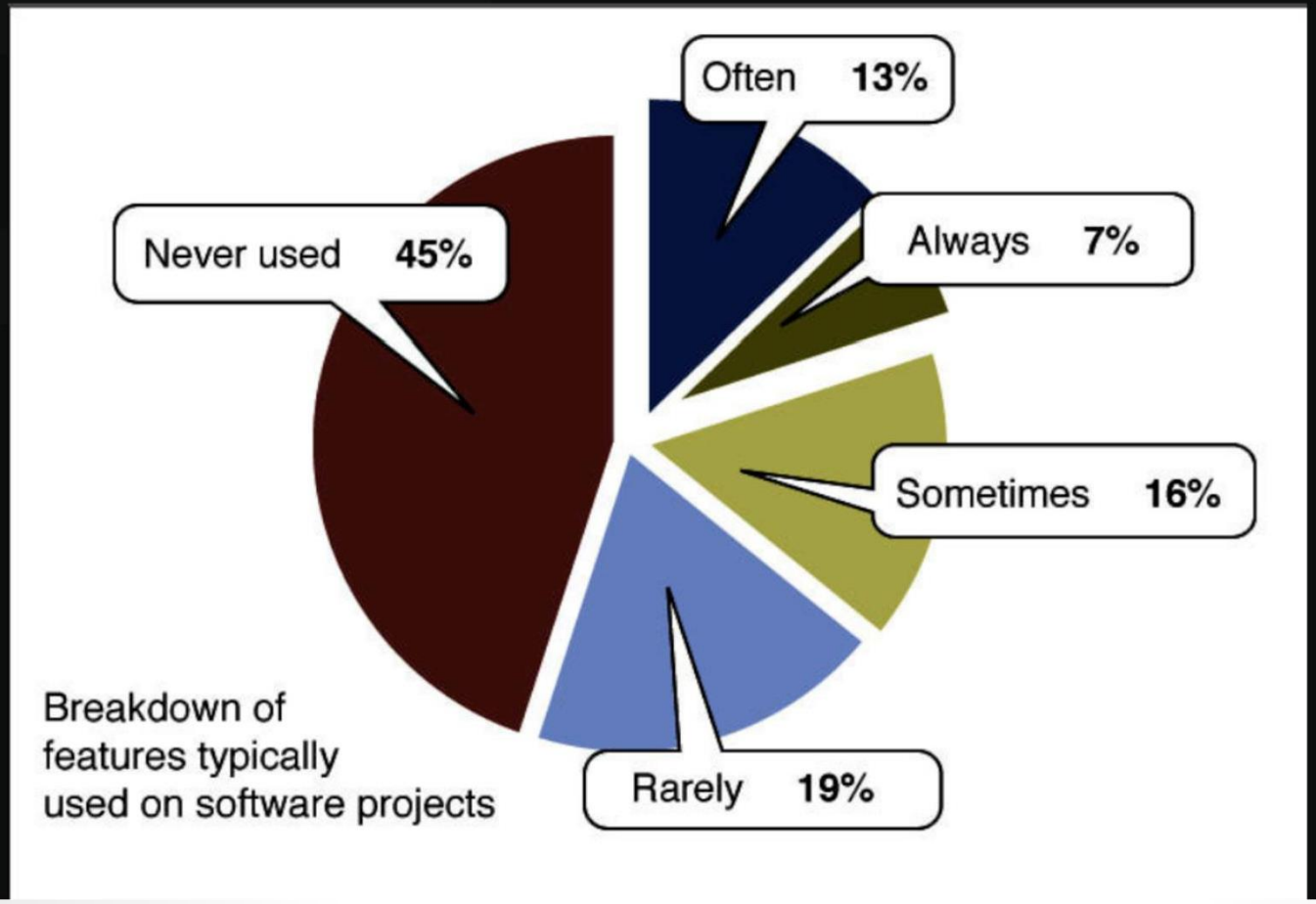


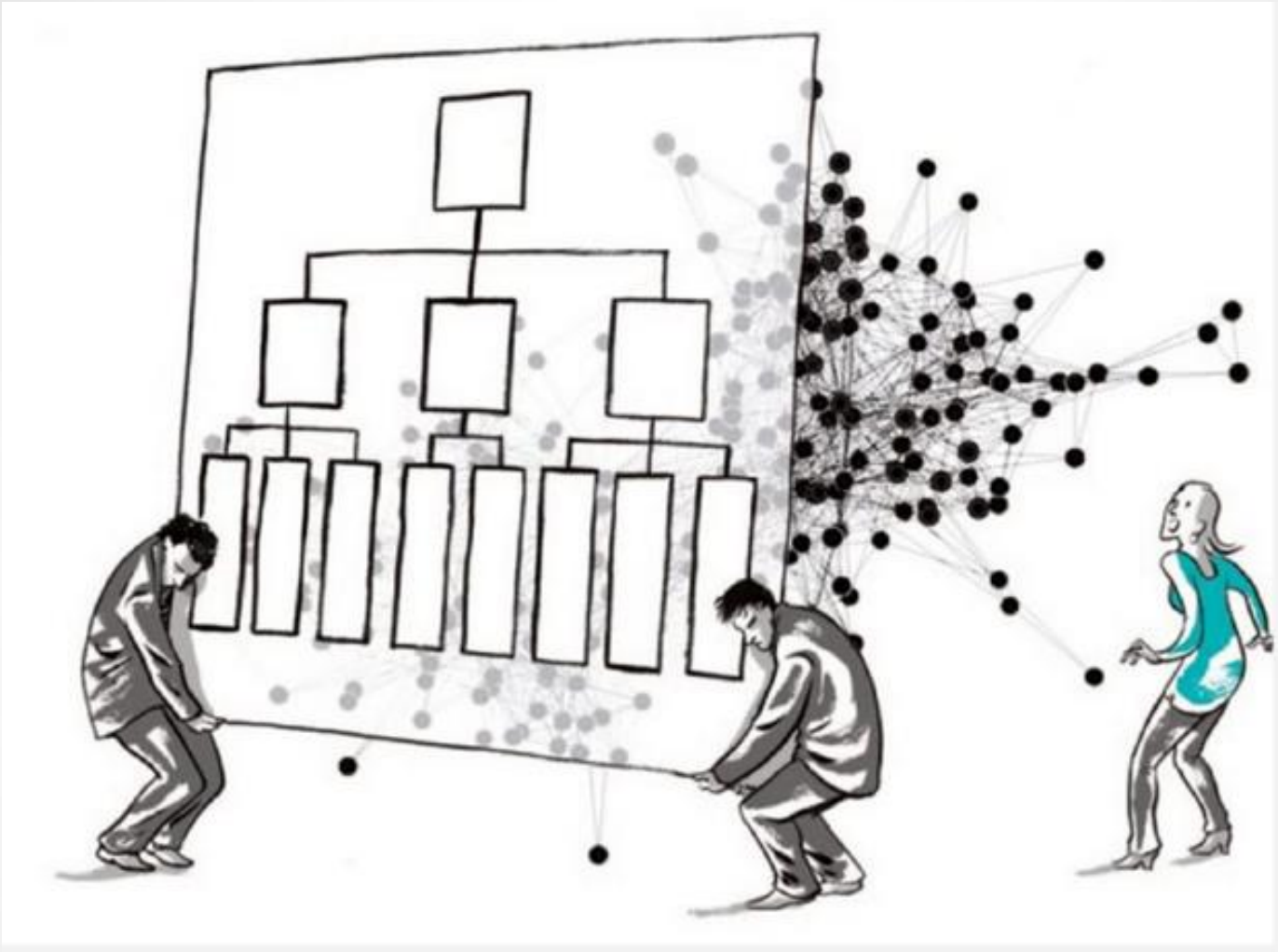
Old Development Process: Problems

- Lack of customer feedback
- Heavy, top-down development process
- Inefficient use of engineering resources
- Low engagement of team members


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

Featuritis

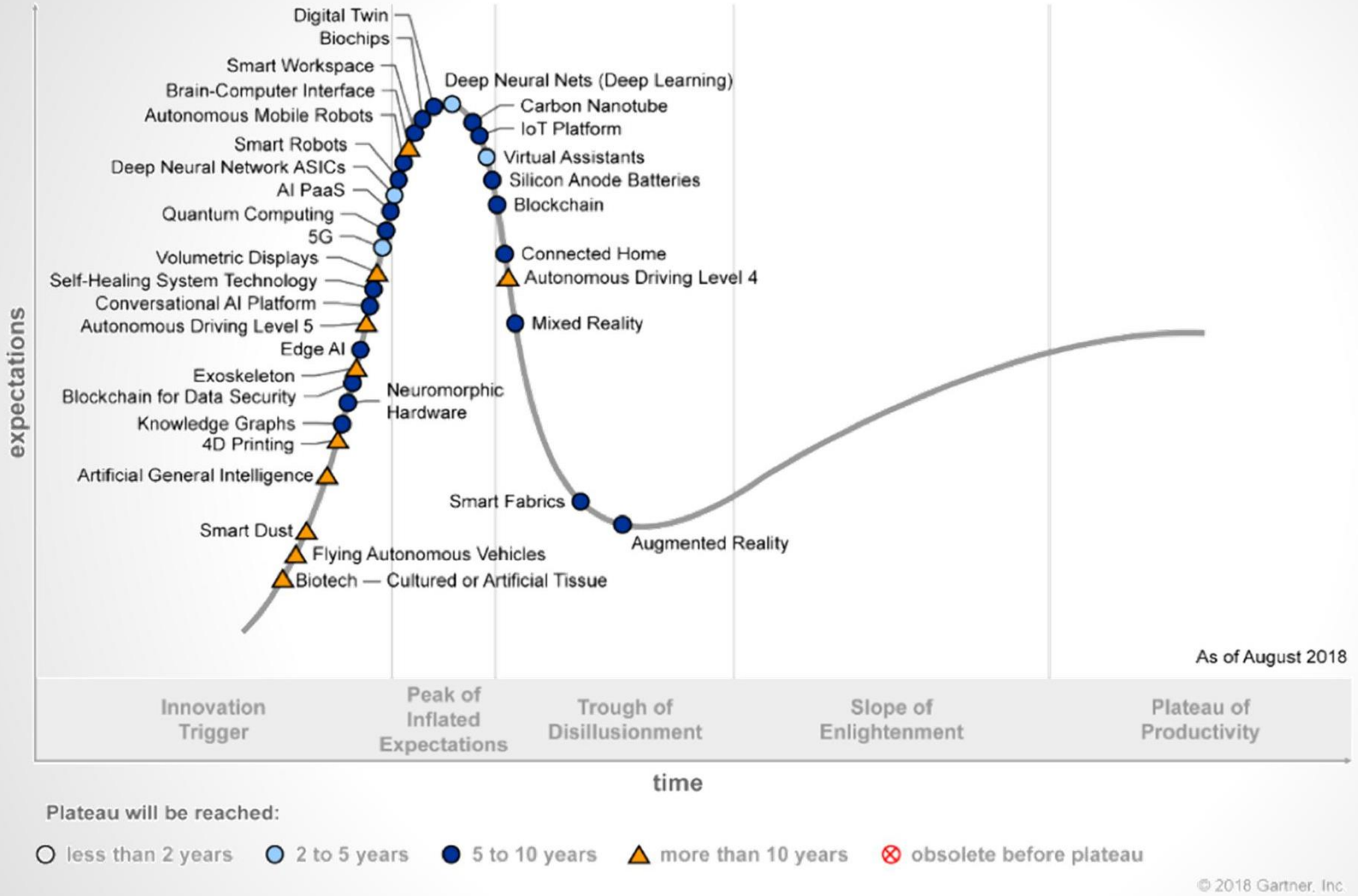

Often 13%
Always 7%
Never used 45%
Sometimes 16%
Rarely 19%
Breakdown of features typically used on software projects



Source: Frederic Laloux's Reinventing Organizations, 2016

WORLD IS CHANGING…

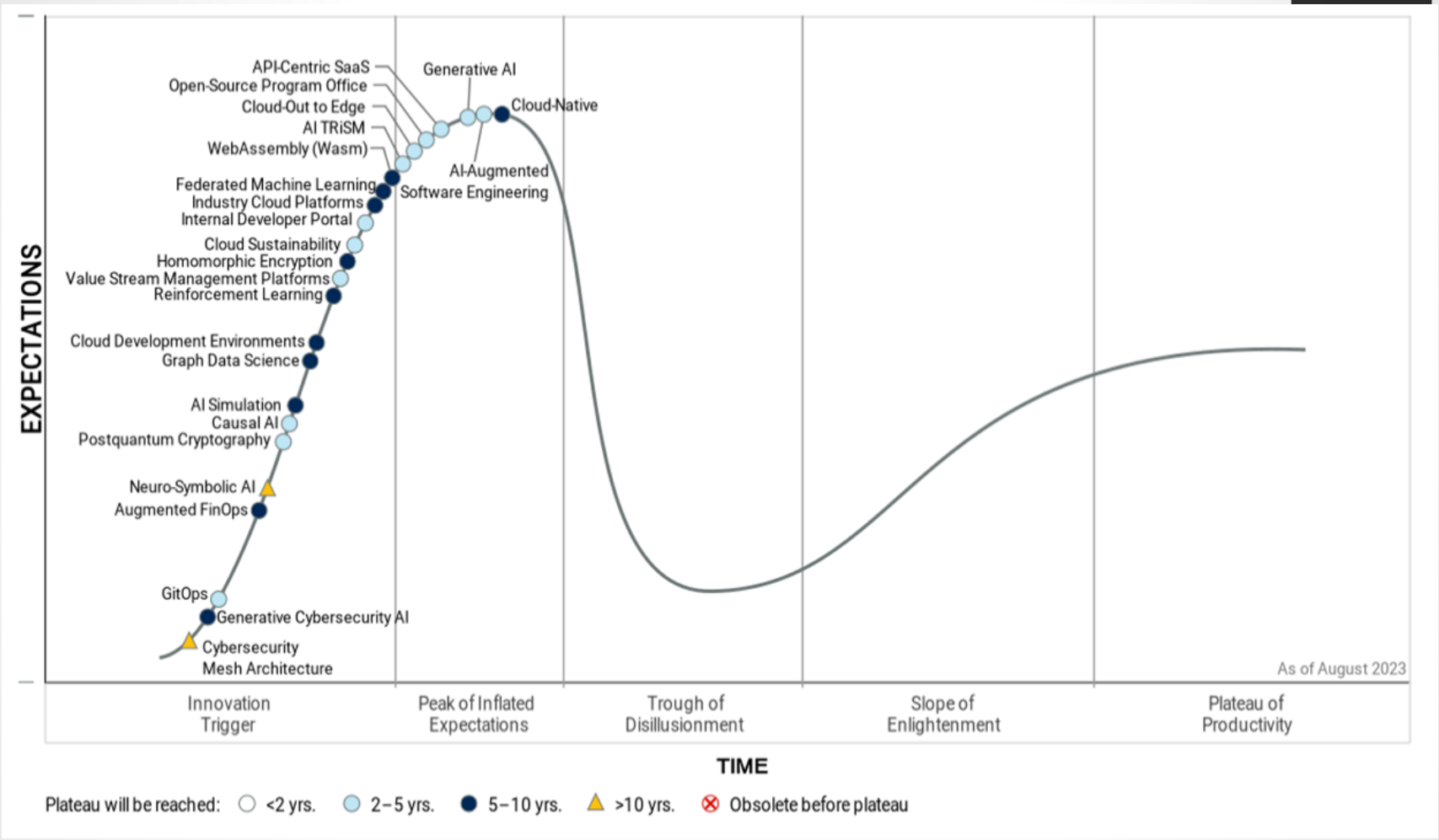
(GARTNER HYPE CYCLE 2018)



LUT University

WORLD IS CHANGING…

(GARTNER HYPE CYCLE 2023)

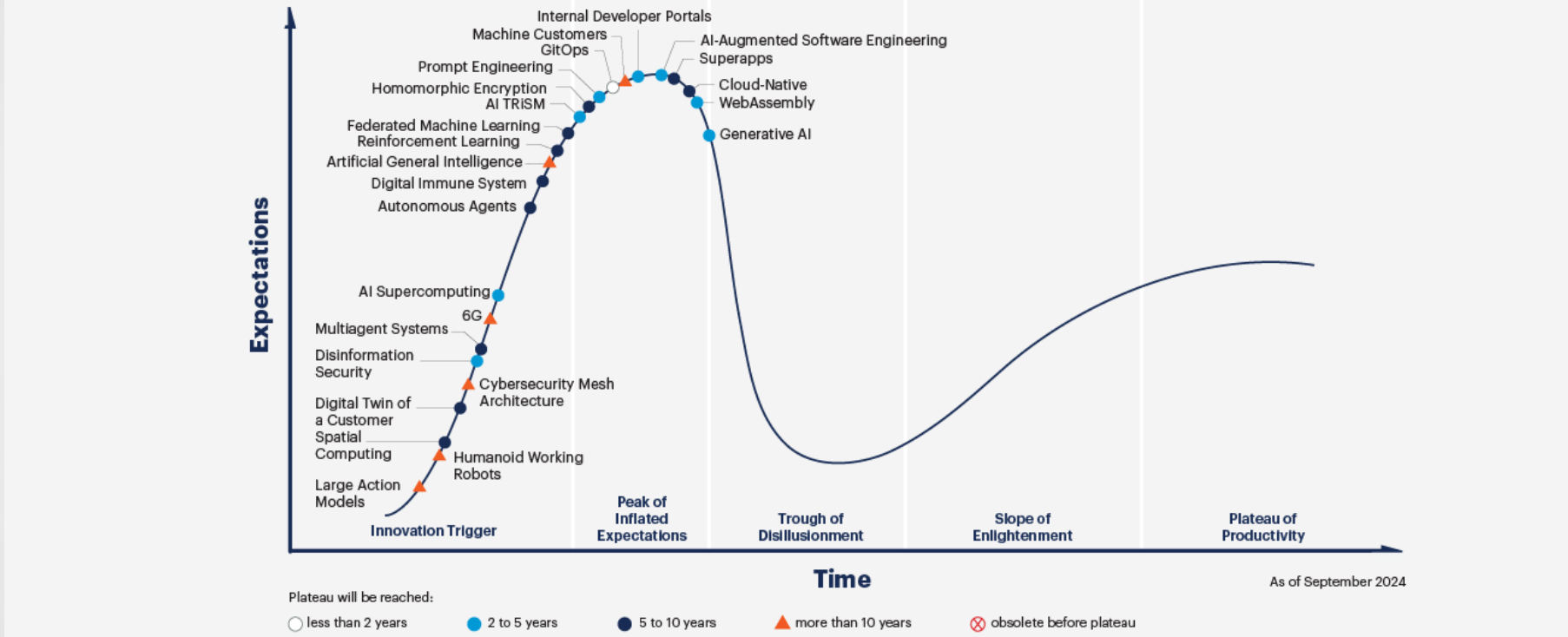


Gartner

LUT University

WORLD IS CHANGING…

(GARTNER HYPE CYCLE 2024)



Source: Gartner

Gartner

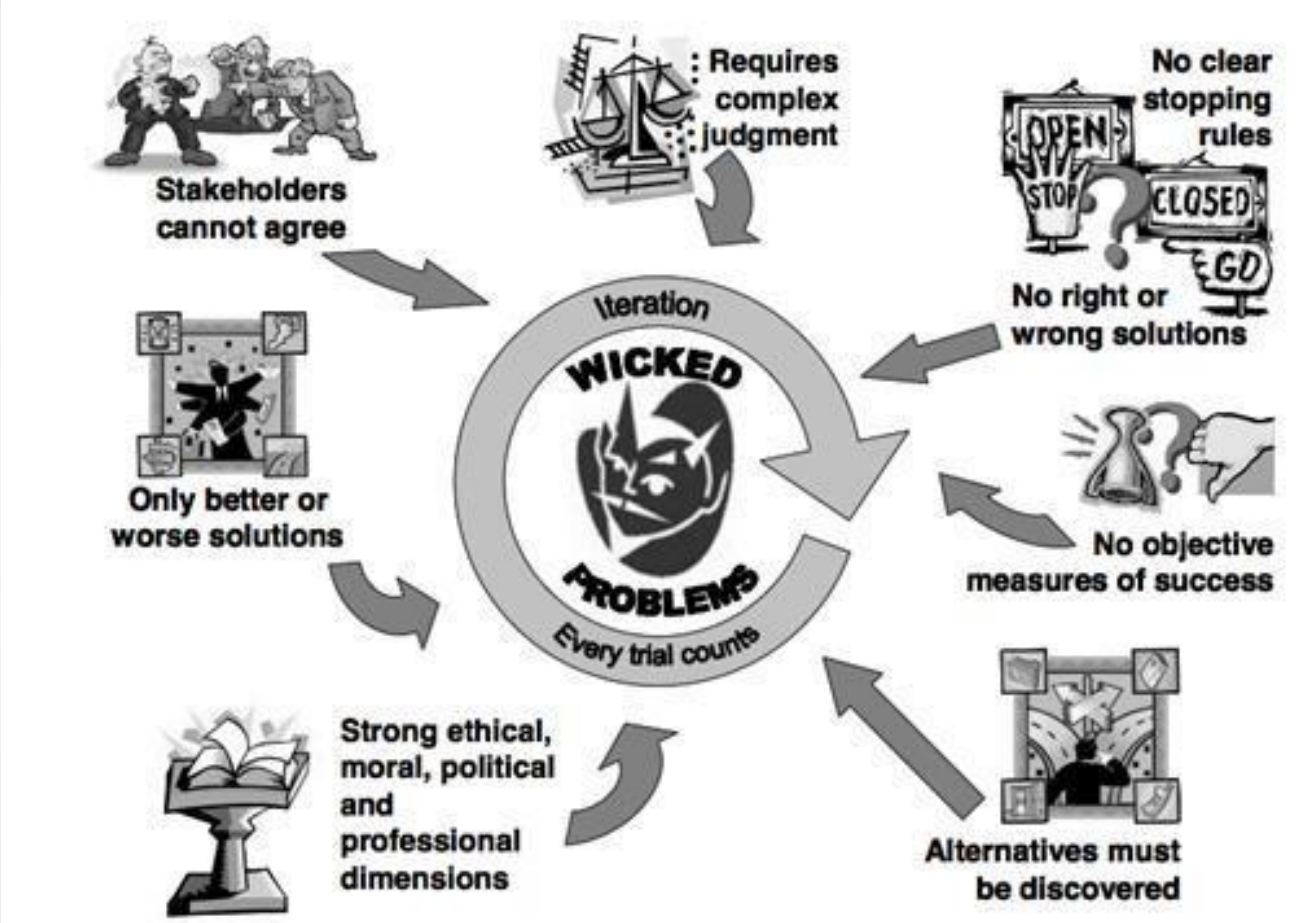
SOFTWARE IS EVERYWHERE

SOFTWARE IS EVERYWHERE

Trend: Products to Services

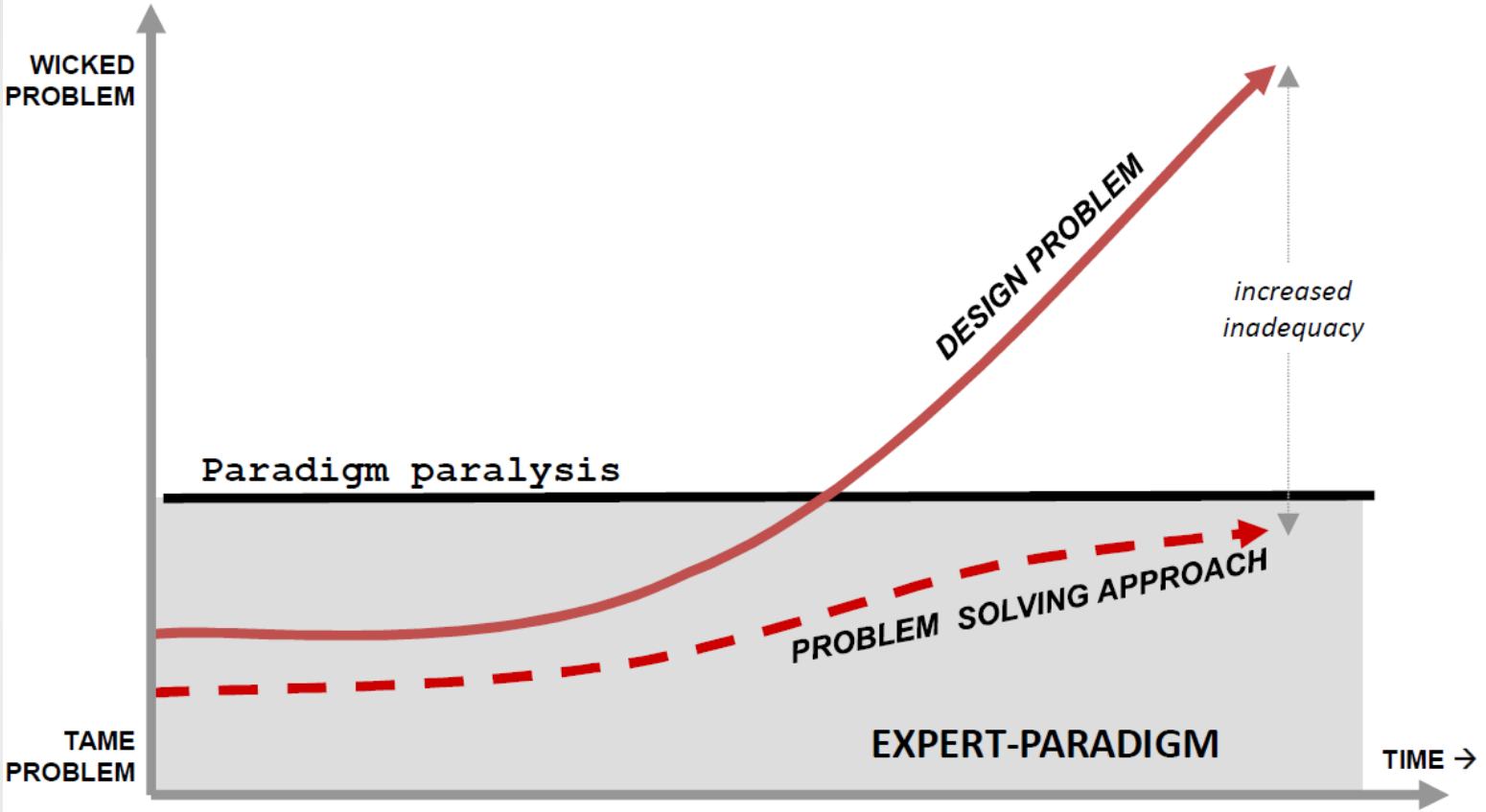
This requires continuous deployment throughout the lifetime of the product

WICKED PROBLEMS



Rittel, H. W. J. and M. M. Webber (1973). "Dilemmas in a General Theory of Planning." Policy Sciences **4(2): 155-169.**

WE NEED TO RECONSIDER OUR BELIEFS (BUT THIS IS HARD)



Jantunen, S. (2012). Making Sense of Software Product Requirements. Lappeenranta, Lappeenranta University of Technology.

RE NEED TO EXPAND & EVOLVE

CONTEXTUAL INTELLIGENCE

- Ability to understand the nature of design problem
- Ability to understand the limits of traditional development methods
- Development of new methods to cope better with wicked problems

AGILE REQUIREMENT PROCESS

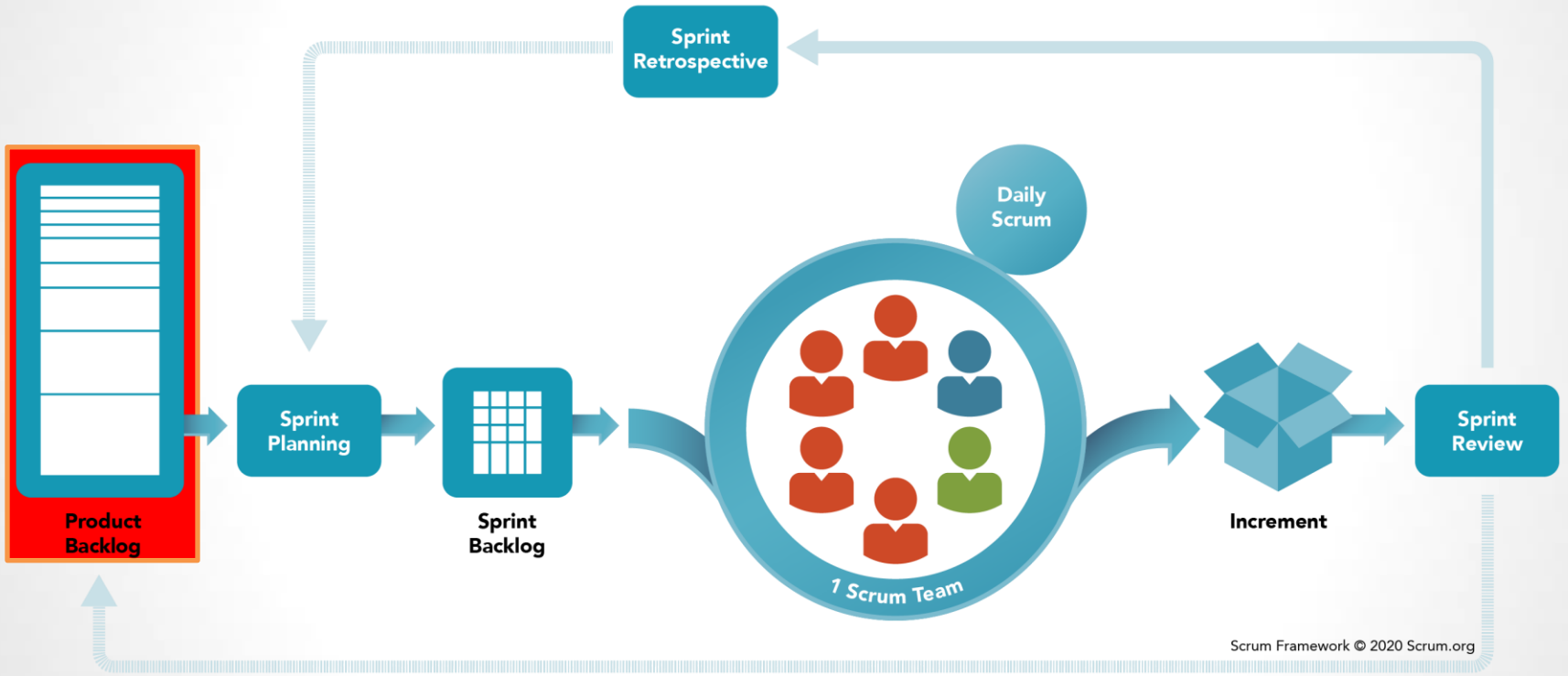
Agile Manifesto

Requirements Techniques for Agile Product Teams

1. **Interviews**
2. **Focus Groups**
3. **Prototyping**
4. **Workshops**
5. Questionnaires or Surveys
6. User Observation
7. Document Analysis
8. Interface Analysis
9. Brainstorming
10. Role-Play
11. Use Cases and Scenarios

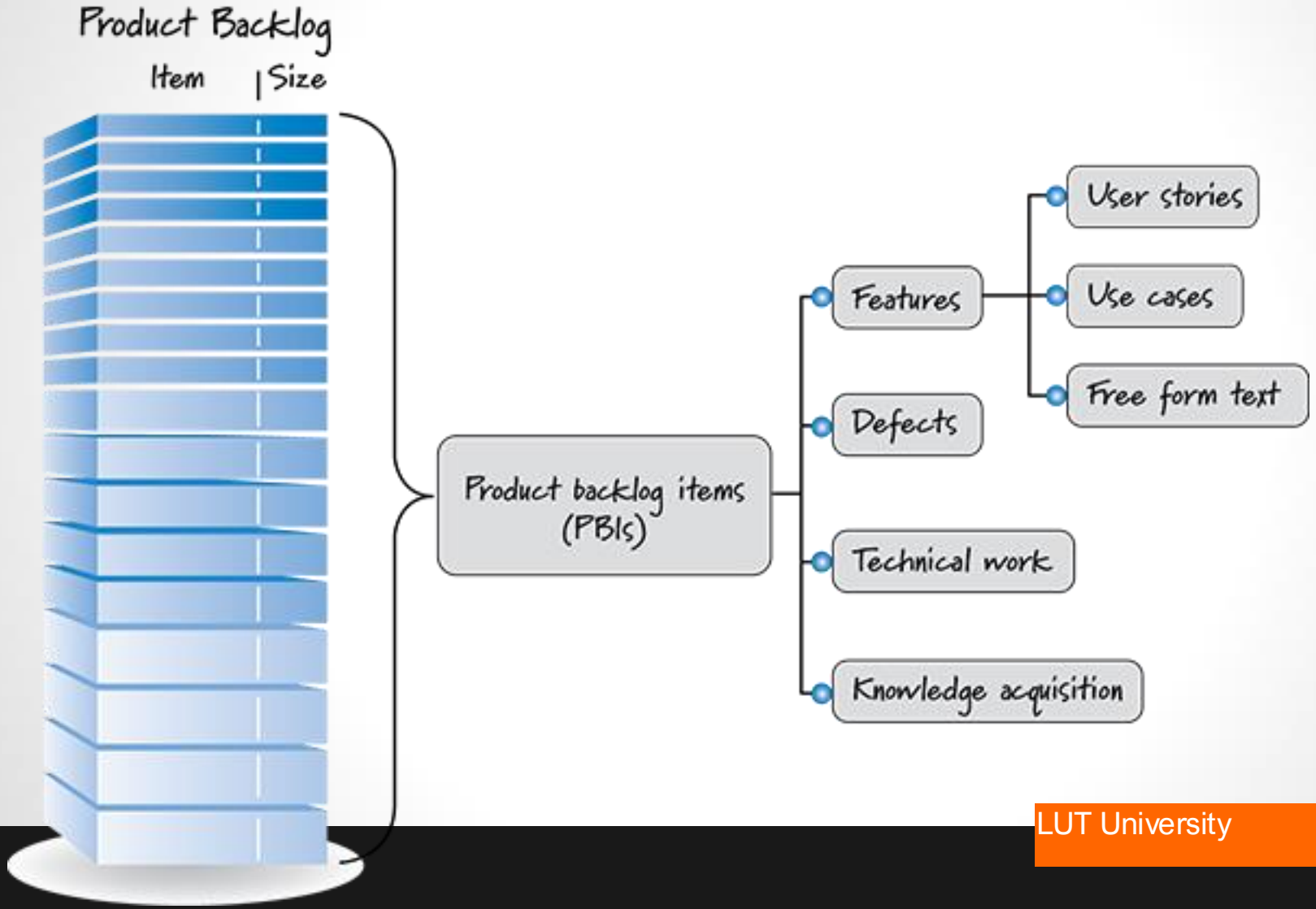
AGILE REQUIREMENT PROCESS

Product Backlog



AGILE REQUIREMENT PROCESS

Product Backlog



TRADITIONAL VS. AGILE RE

Metric	Traditional development process	Agile development process	Study that reported
SDLC	Linear	Iterative	[9] [10]
Development style	Traditional methods are predictive (plan–driven)	Agile methods are adaptive	[11]-[14]
Documentation	Enough documentation to be able to answer all questions that might be asked in the future.	Light (replaced by face to face communication)	[7] [15]
Customer involvement	in traditional approaches the customer is mainly engaged during the early phase of the project	Agile methods engage the customer throughout the whole development process.	[13] [16]-[18]
Change	Resistance to change	Welcoming to change, even changes are brought in late in the project.	[19] [20]
Size	traditional methods are able to manage effectively large project	Manage effectively requirements in small projects but not in large ones.	[13]
Planning scale	Long term	Short term	[13]
Management	process oriented, command and control	People oriented, leadership and conformity.	[11] [12] [15] [21] [22]
Team organization	Pre- structured teams	Self organizing teams	[23]
Ownership	Ownership belongs to only project manager	Shared ownership	[19]
Prioritization	Requirements are typically prioritized once.	Prioritize feature lists repeatedly during development	[24]
Customer feedback	At the termination of the project	At the completion of every sprint	[25]
Risk identification	No risk identification.	Early identification and mitigation in every sprint.	[25]
Time between specification & implementation	Long	Short	[13]
Delivery	Delivering artifacts phase wise and delivery of working software at the end of project.	Demonstration and delivering working software et the end of every sprint.	[25]
Measure of Success	Conformance to plan	Business value delivered	[3]

Abbas, J. (2016) Quintessence of Traditional and Agile Requirement Engineering. *Journal of Software Engineering and Applications*, **9**, 63-70. doi: 10.4236/jsea.2016.93005.

ISSUES RESOLVED BY AGILE RE

Issues of RE in Waterfall	Resolved by RE in Agile
Customer involvement: Customers are involved only during the beginning of requirement gathering and analysis [40].	Customers are involved throughout the complete process.
Prioritization of requirements: Complete requirements for the full project are prioritized upfront and the prioritization is kept up through the project lifecycle, and reprioritization is arduous [47].	Priorities are setup for all iterations that offer opportunities for getting desirable results and customer satisfaction [48] [49].
Documentation: Totally emphasizes at properly gathering organizing and documenting all requirements and excludes any live meetings/conferences [14].	User stories are concise and provide to-the-point explanation of user demands, obviate the need for maintaining long SRS documents.
Requirements validation: Validation happens late in the life cycle.	Prototyping helps in providing the customer with a blueprint of the product, and therefore helps in validating the requirements [50].
Communication: It is a major factor in the delay and failure of software projects [40].	It provides regular interaction with customer and among teams.
Over-scoping of requirements: It is the cause of rework, which in turn causes further investment.	Developers receive a list of features that are constantly prioritized so the chance of having to repeat allocation in projects is minimized.
Shall Argument: The worst thing of waterfall RE is "shall" argument *i.e.* system shall do it, etc. [46].	Agile introduces the real time system.

Abbas, J. (2016) Quintessence of Traditional and Agile Requirement Engineering. *Journal of Software Engineering and Applications*, **9**, 63-70. doi: 10.4236/jsea.2016.93005.

ISSUES OF AGILE RE

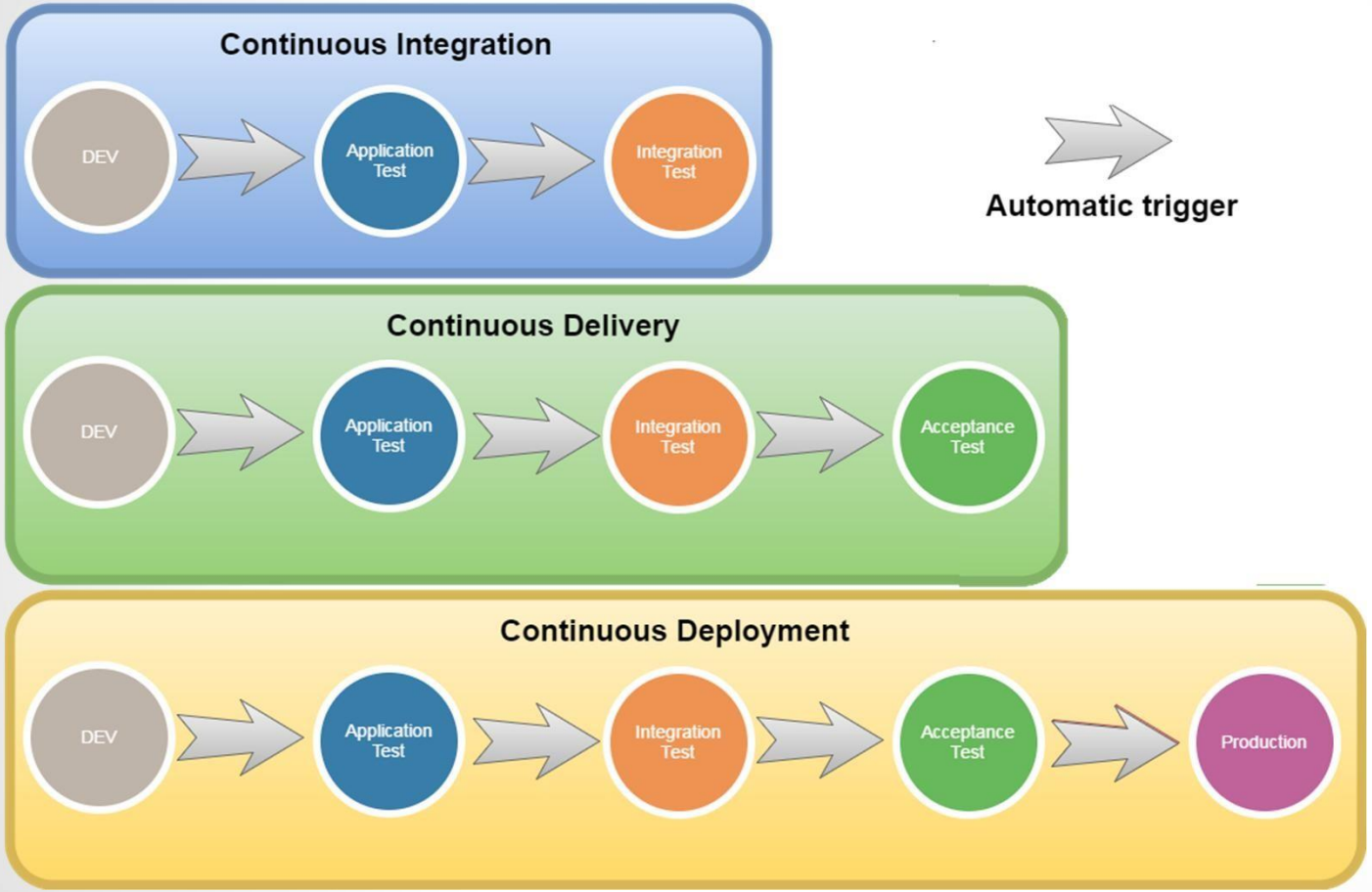
Table 8
Summary of challenges of agile RE.

Challenge	Description	Impact	Solutions
Minimal documentation (Cao & Ramesh, 2008)	User stories and product backlogs are the only documents in agile methods (Zhu, 2009)	Traceability issues (Zhu, 2009)	
Customer availability (Ramesh et al., 2010)	Availability of customer for requirements negotiation, clarification and feedback	Increase in rework	Surrogate customers (Ramesh et al., 2010)
Inappropriate architecture (Ramesh et al., 2010)	Inadequate infrastructure can cause problems during later project stages	Increase in cost	Code refactoring (Berry, 2002)
Budget and time estimation (Cao & Ramesh, 2008)	Initial estimates of time and cost are changed substantially by a change in requirements in subsequent stages	Project delays	Frequent communication
		Over-budgeting	Accurate modelling of user story
Neglecting non-functional requirements (NRFs)	User stories only satisfy system/product features	System security, usability, performance at stake	NRF modelling approach (Farid & Mitropoulos, 2012b). The NORMATIC tool (Farid & Mitropoulos, 2012a)
Customer inability and agreement (Daneva et al., 2013; Ramesh et al., 2010)	Incomplete domain knowledge and in consensus among customer groups	Increase in rework	Creation of delivery stories to accompany user stories (Daneva et al., 2013)
		Increase in cost	Frequent communication Iterative RE (Ramesh et al., 2010)
Contractual limitations (Cao & Ramesh, 2008)	Fixed-price contracts do not allow changes	Increase in cost	
Requirements change and its evaluation	To find the consequences of requirements change	Increase in work delay	RE-KOMBINE framework (Ernst et al., 2013)

Inayat, Irum, et al. "A systematic literature review on agile requirements engineering practices and challenges." *Computers in human behavior* 51 (2015): 915-929.

AGILE IS NOT THE END OF EVOLUTION

TREND: TOWARDS CONTINUOUS DEPLOYMENT

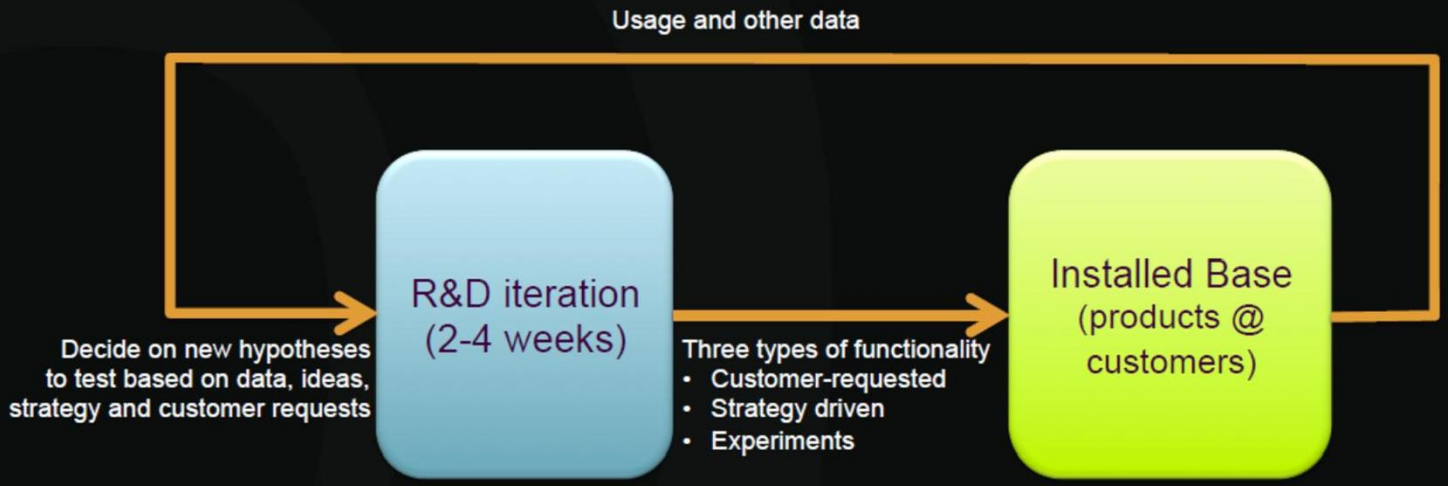


https://stackoverflow.com/questions/28608015/continuous-integration-vs-continuous-delivery-vs-continuous-deployment

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



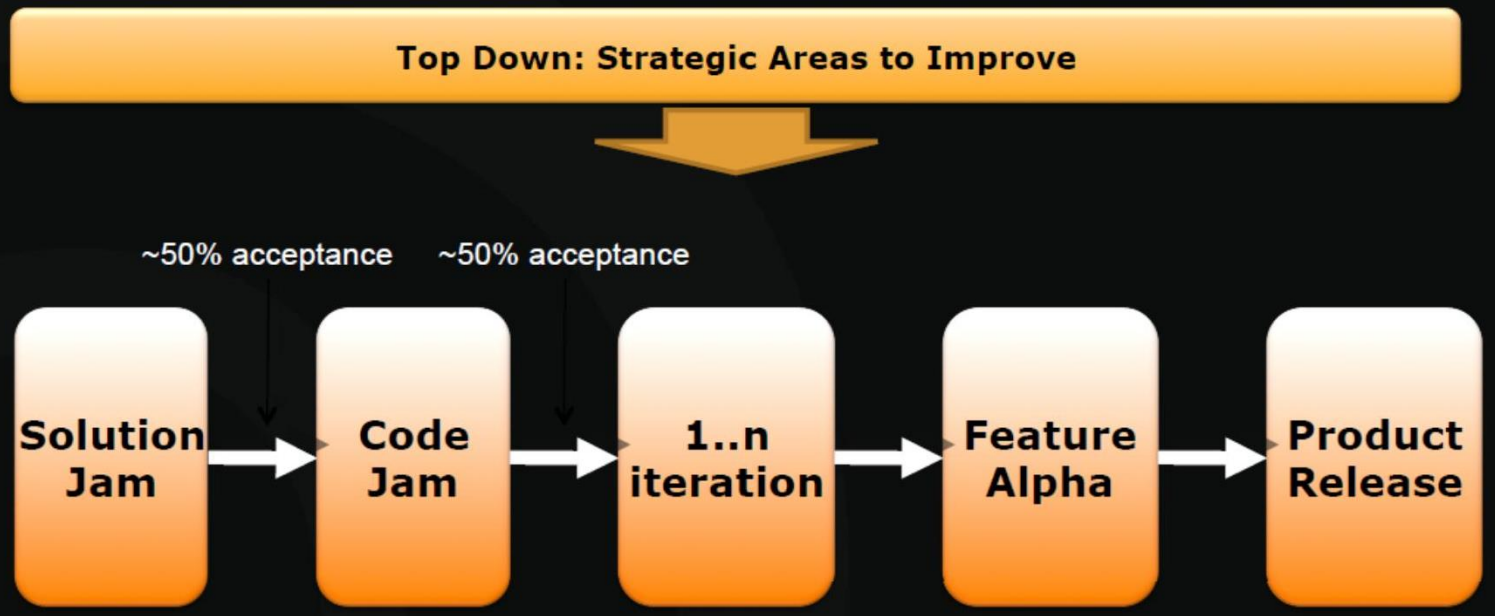
Decisions should be based on DATA, not opinions

Source: Jan Bosch (2013): "Do As I Say; Not As I Do? From Requirements Engineering to Experimenting with Customers"
Keynote presentation in REFSQ 2013 conference

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

New Process: Overview



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

Source: Jan Bosch (2013): "Do As I Say; Not As I Do? From Requirements Engineering to Experimenting with Customers"
Keynote presentation in REFSQ 2013 conference

Evolution: A/B Testing

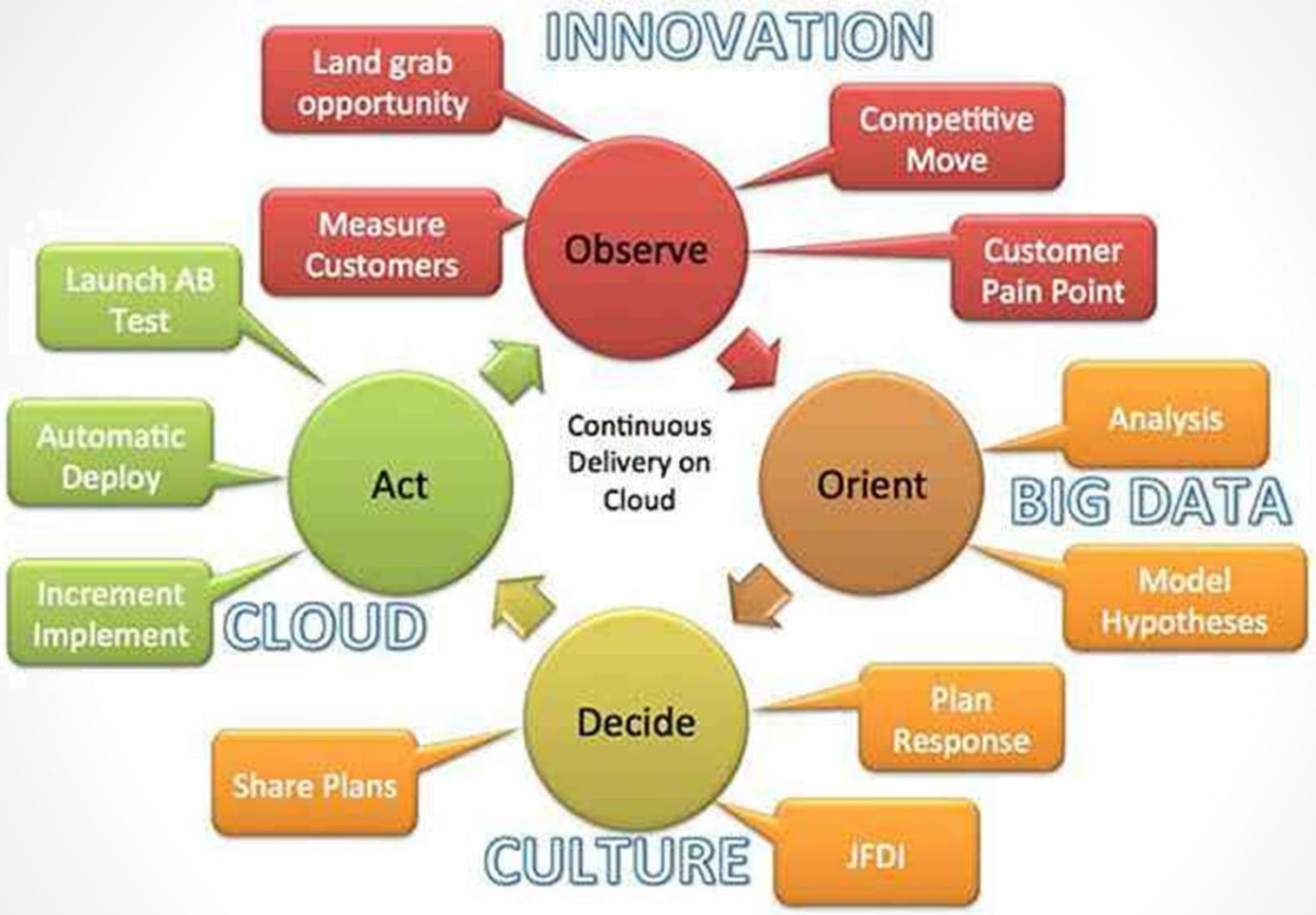
- What
  - A/B testing is a method of comparing a baseline control sample to a variety of single-variable test samples for improving some metric
- Alternatives
  - “Marketing” testing, e.g. colors, buttons and order of options
  - Alternative implementations of a feature
- To think about
  - Run multiple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

Source: Jan Bosch (2013): ”Do As I Say; Not As I Do? From Requirements Engineering to Experimenting with Customers”
Keynote presentation in REFSQ 2013 conference

A/B Testing Examples

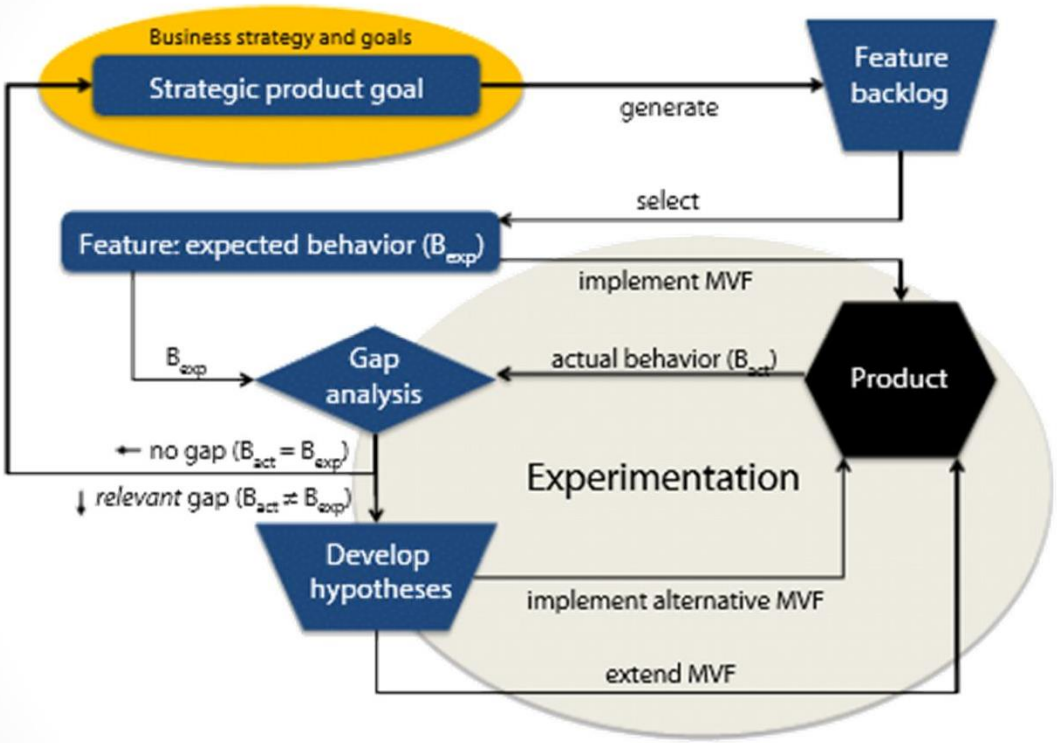
- 37signals tested the headline on its pricing page. It found that “30-Day Free Trial on All Accounts” generated 30% more sign-ups than the original “Start a Highrise Account.”
- Dustin found that “You should follow me on Twitter here” worked 173% better than his control text, “I’m on Twitter.”
- A surprising 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%.

Source: Jan Bosch (2013): ”Do As I Say; Not As I Do? From Requirements Engineering to Experimenting with Customers”
Keynote presentation in REFSQ 2013 conference



http://climalaboralycomunicacion.blogspot.com/2016/11/ciclo-ooda-nuestro-sexto-sentido.html

The HYPEX model



The HYPEX model: from opinions to data-driven software development
HH Olsson, J Bosch - Continuous software engineering, 2014 - Springer