

Modern Software Engineering

We Create Digital Products

Why Agile Architecture and Design?

Every company is now a software company.

– *Forbes Magazine*

Software is eating the world, in all sectors.

In the future every company will become a software company.

– *Marc Andreessen, Wall Street Magazine*

The future is already here. It is just not evenly distributed.

– *William Gibson*

Form follows function.

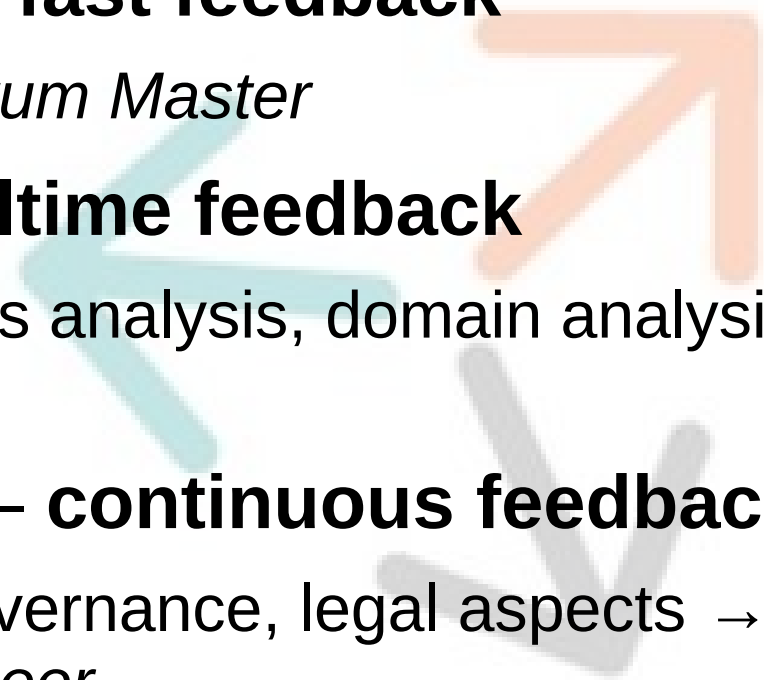
– *Louis Sullivan*

Principles

- Science and its practical application “**engineering**” are vital tools in making effective progress in technical disciplines.
- Our discipline is fundamentally one of **learning** and **discovery**, so we need to become **experts at learning** to succeed, and **science and engineering** are how we learn most effectively.
- Finally, the systems that we build are often **complex** and are increasingly so. Meaning, to cope with their development, we need to become **experts at managing that complexity**.

Farley, David. Modern Software Engineering (p. xxiii)

Agile Software Design

- Agile and DevOps approach – **fast feedback**
 - Scrum, Kanban, DevOps → *Scrum Master*
 - Functional requirements – **realtime feedback**
 - Requirements analysis, business analysis, domain analysis
→ *Product Owner*
 - Non-Functional requirements – **continuous feedback**
 - Fitness Functions, corporate governance, legal aspects →
Team: architect, designer, engineer
- 

What is MTS – COBAS 4800?

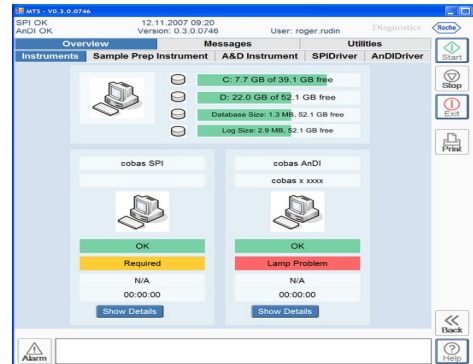
- **MTS** - Medium Throughput System
- Detection of two virus through DNA Analysis using PCR: HPV und CT/NG
- Additional detections (MRSA, KRAS, etc.) and customer designed analysis kit support








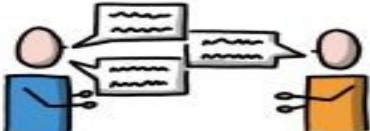




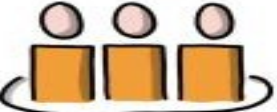

SPI
Sample
Preparation
Instrument



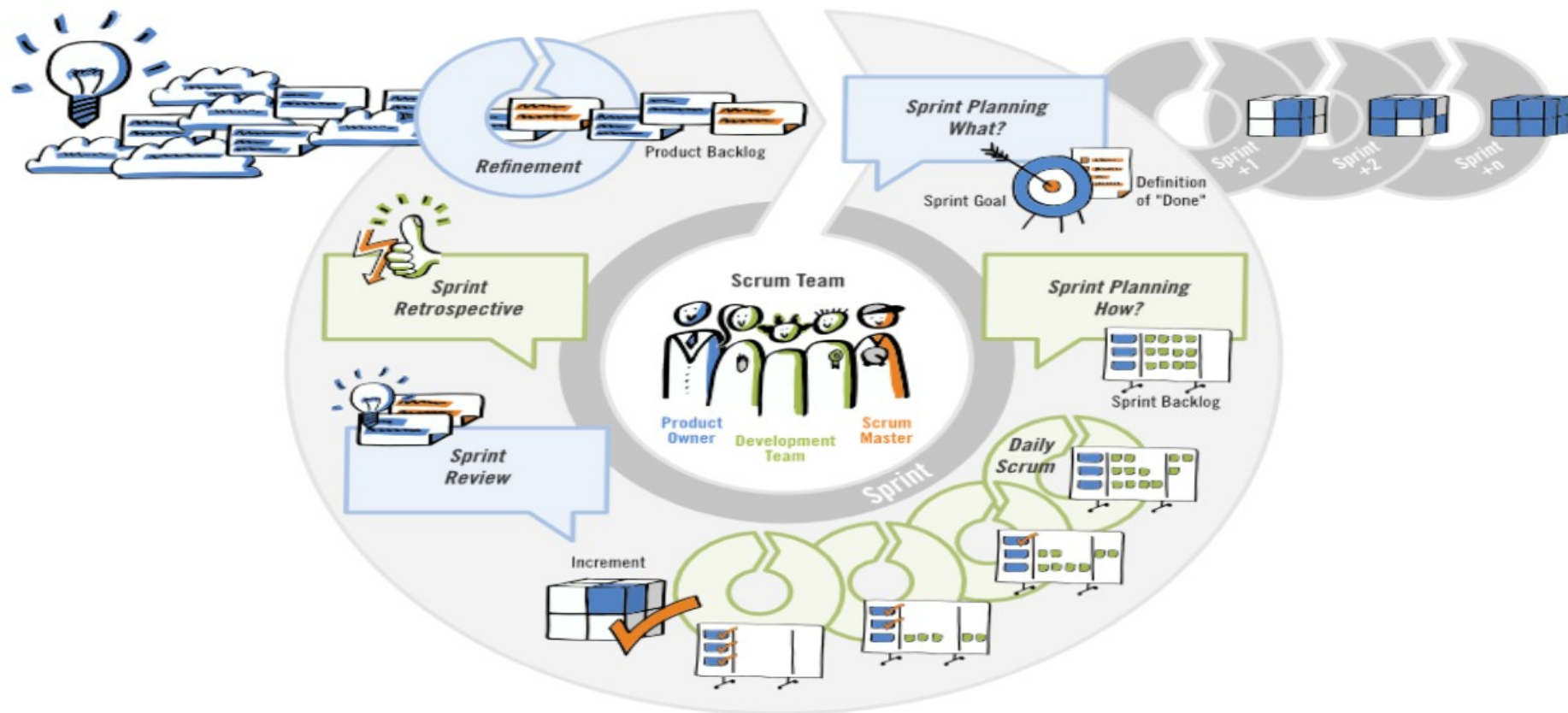
AnDI
Amplification and
Detection Instrument



Agile Manifesto Principles

<p>Satisfy the customer through early and continuous delivery of valuable software.</p> 	<h2>12 Agile Principles</h2> <p>@OlgaHeismann</p>		<p>Business people and developers must work together.</p> 
	 <p>Welcome changing requirements, even late in development.</p>	 <p>Deliver working software frequently.</p>	
<p>Build projects around motivated individuals. Give them the support they need. Trust them.</p> 	 <p>The most efficient and effective method of conveying information is face-to-face conversation.</p>	<p>Working software is the primary measure of progress.</p> 	 <p>The sponsors, developers, and users should be able to maintain a constant pace indefinitely.</p>
<p>Continuous attention to technical excellence and good design.</p> 	 <p>Simplicity—the art of maximizing the amount of work not done—is essential.</p>	<p>The best architectures, requirements, and designs emerge from self-organizing teams.</p> 	<p>The team reflects on how to become more effective and adjusts its behavior accordingly.</p> 

Scrum



Agile Architecture Rules

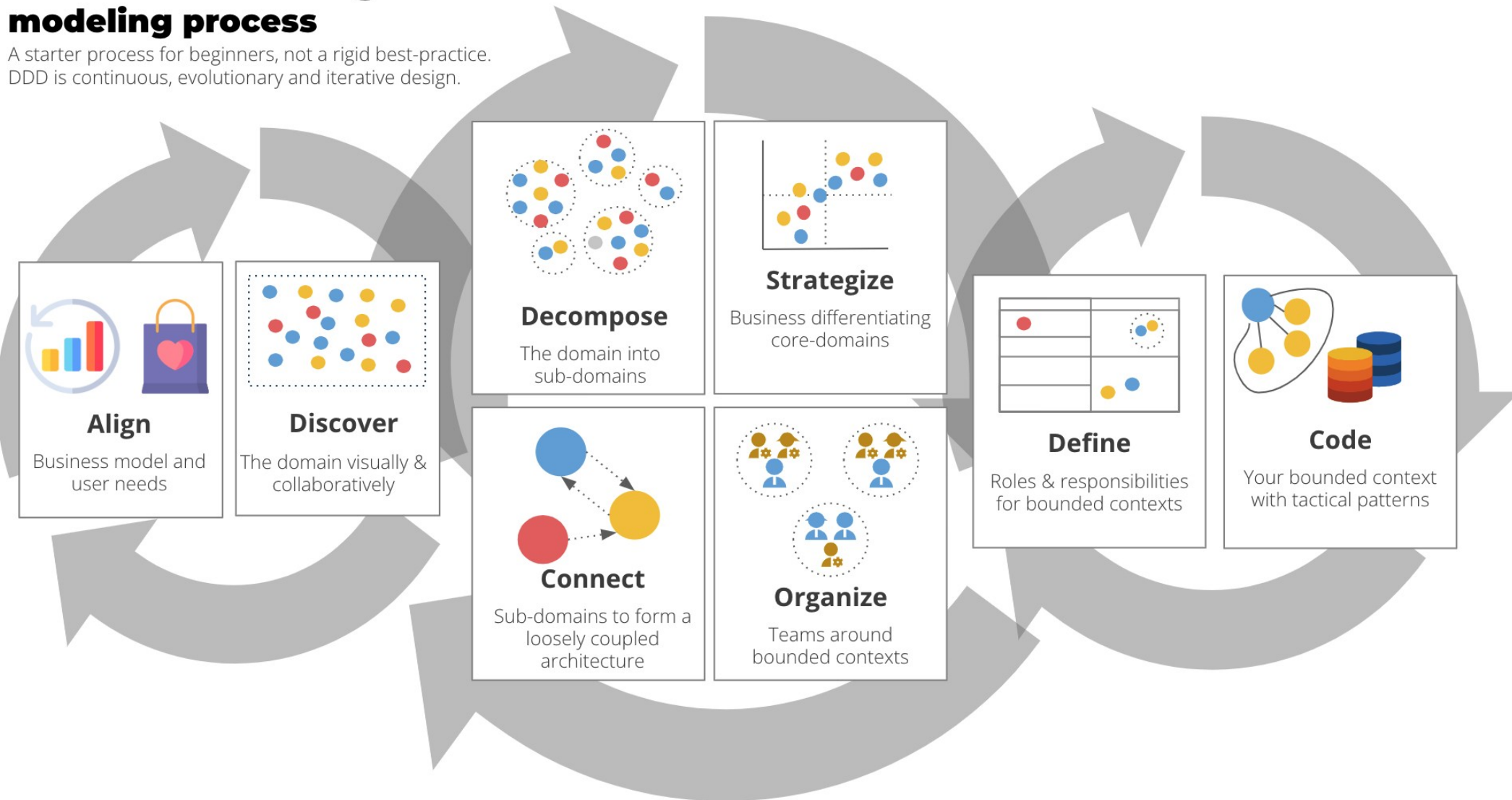
- Features should be validated through tests
- Tests should be automated
- Tests should be run before each release to avoid regression errors
- Releases are performed multiple times per Sprint

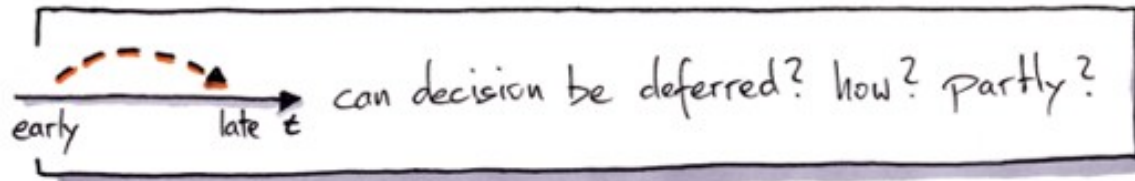
The source code is the architecture

- Difference between to *architect* (process) and an *architecture* (result)
- If you think good architecture is *expensive*, try bad architecture
- ***Waste*** in architecture and design
- Good architects should be ***good developers***

Domain-Driven Design starter modeling process

A starter process for beginners, not a rigid best-practice.
DDD is continuous, evolutionary and iterative design.





- persist data of your system to survive restart
- how to translate UI and data
- communication between parts of your system
- scaling (run on multiple threads, processes, machines)
- security (how to authenticate, authorize)
- journaling (Activities, data)
- reporting
- data migration / data import
- releasability
- backwards compatibility
- response times
- Archiving data

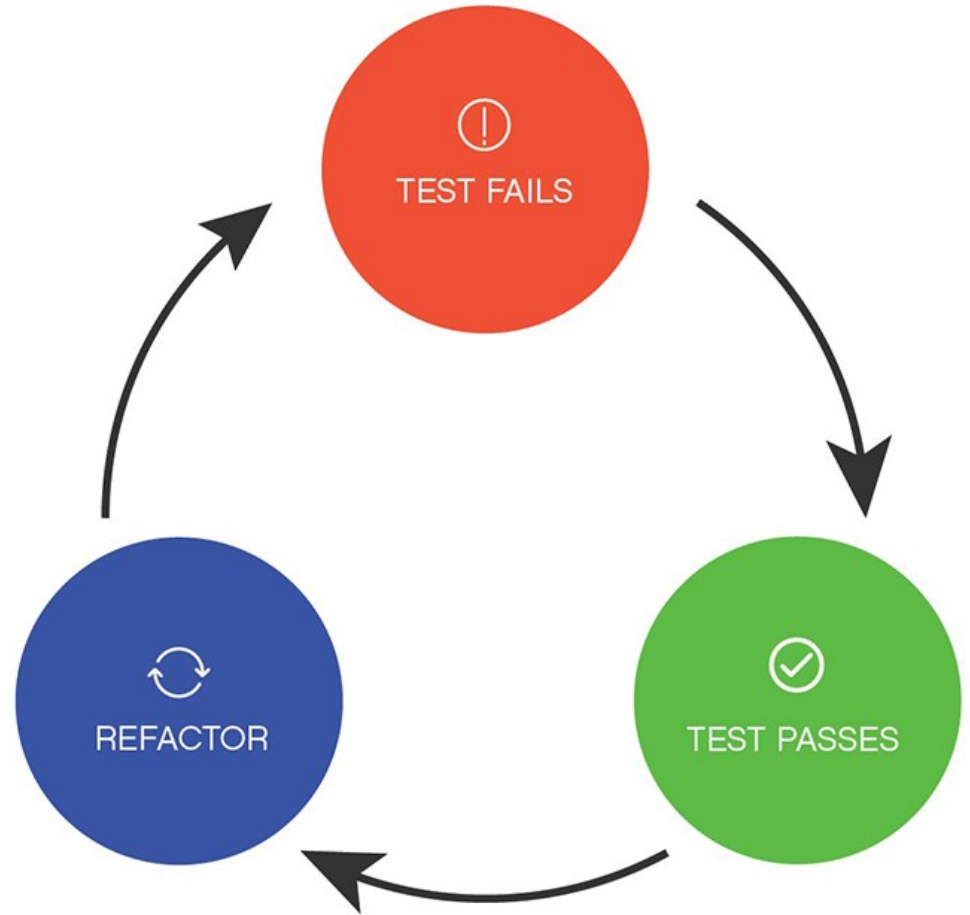
design to be independent
on decision

Test Driven Development

The duration of the cycle is a few minutes, never hours.

Upon completion of a cycle you should commit your changes.

TDD Cycle



3 Verification Report

3.1 Summary

Number of test cases	passed	25
	failed	0
Total number of test cases performed		<u>25</u>

3.2 List of Test Results

TC ID	TC Name	Author	Reviewer	Date / Time	Result
UTC291	RunDailyAndWeeklyMaintenance	Peter Rey / pr	n/a	4/24/2009 10:31:58 AM	PASSED
UTC292	AddInstrumentTrace2TraceLog	Peter Rey / pr	n/a	4/24/2009 10:31:59 AM	PASSED
UTC293	ConnectAutomatically				
UTC294	DisconnectInstrumentAndPhoenixPopup				
UTC295	ImplementInstrumentInter				
UTC296	InstrumentInformationCha				
UTC297	NotifyInstrumentView				
UTC298	InstrumentInitializationFail				
UTC298	InstrumentInitializationMaintenanceRequired				
UTC299	InstrumentInitializationOk				
UTC300	LogException				
UTC301	LogMethodEntryAndMetho				

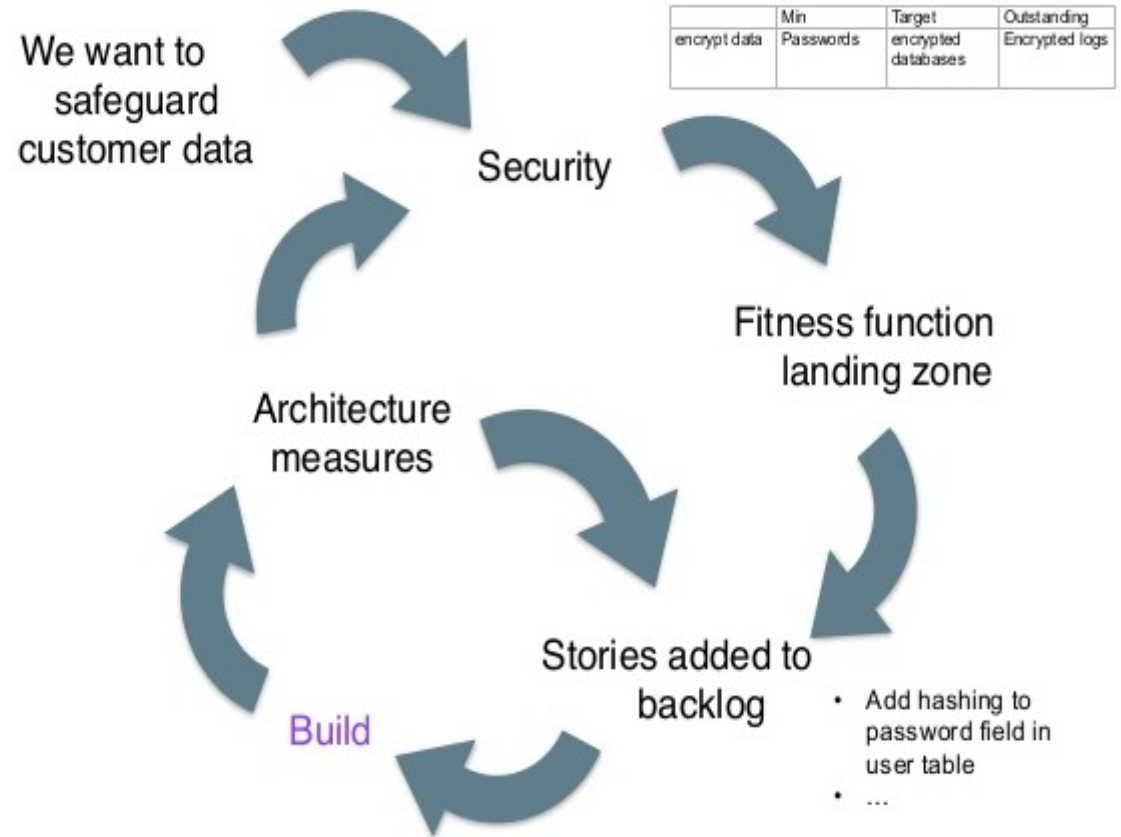
5.8 UTC298 - InstrumentInitializationMaintenanceRequired

ID	UTC298
Name	InstrumentInitializationMaintenanceRequired
Author	Peter Rey / pr
Reviewer	n/a
Description	If the ML_STAR instrument is switched on, the initialization of the ML_STAR instrument and the heater shaker was successful but there is outstanding maintenance, the instrument view shall be notified with the instrument status maintenance required
Test Methods	- Normal Case
Execution Date	4/24/2009 10:32:02 AM
Time	USP742
Host ID	OLC
User	pete
Environment	NUR
Pre-Condition	Non
Details	Desc Expe Outco PASS

Criticality: Low	UTC298	InstrumentInitializationMaintenanceRequired
Criticality: High	UTC310	UnexpectedErrorOnInstrument
Criticality: Low		

Fitness Functions (2/2)

Double loop architecture is a process that you can use to ensure that your architecture continues to satisfy the business needs of your product.



DevOps DORA Metrics

Deployment Frequency

>1/day

deployments per day

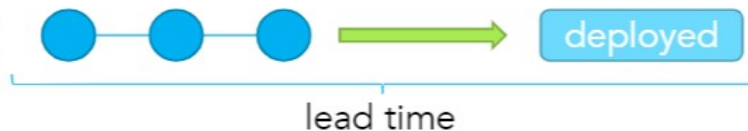


Throughput

Lead time for change

<1h

hours



Velocity

Time to Restore

<1h

hours

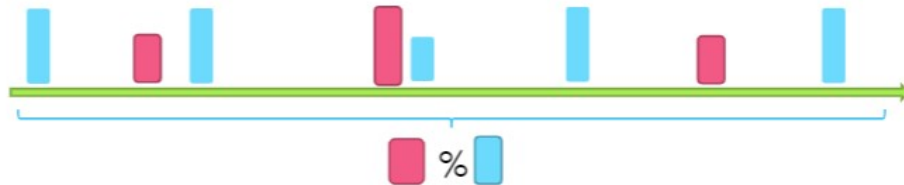


Velocity

Change Failure Rate

0-15%

Percentage of failed deployments



Risk

Static Web Sites - *Pages*

- [Hugo](#), Jenkyl
- [Docsy](#) plugin for Hugo
- Pages for [GitHub](#), [GitLab](#), [Bitbucket](#)
 - Updated through your CI pipeline
- JavaDoc, ADR, [PlantUML](#) as part of your static website
- Synchronized with your git repository
- Publish daily



Questions & Answers



marcel.baumann@tangly.net

<https://blog.tangly.net>