ICT Improve
Part of ICT Group

This track is sponsored by:

PHILIPS

Peopl

Rob Albers, Ronald Holthuizen & Martijn van Tienen

BDD, (A)TDD and DevOps practices as a recipe for continuous compliance

16.15 – 17.00 – P083

PHILIPS



BDD, (A)TDD and DevOps practices

as a recipe for continuous compliance

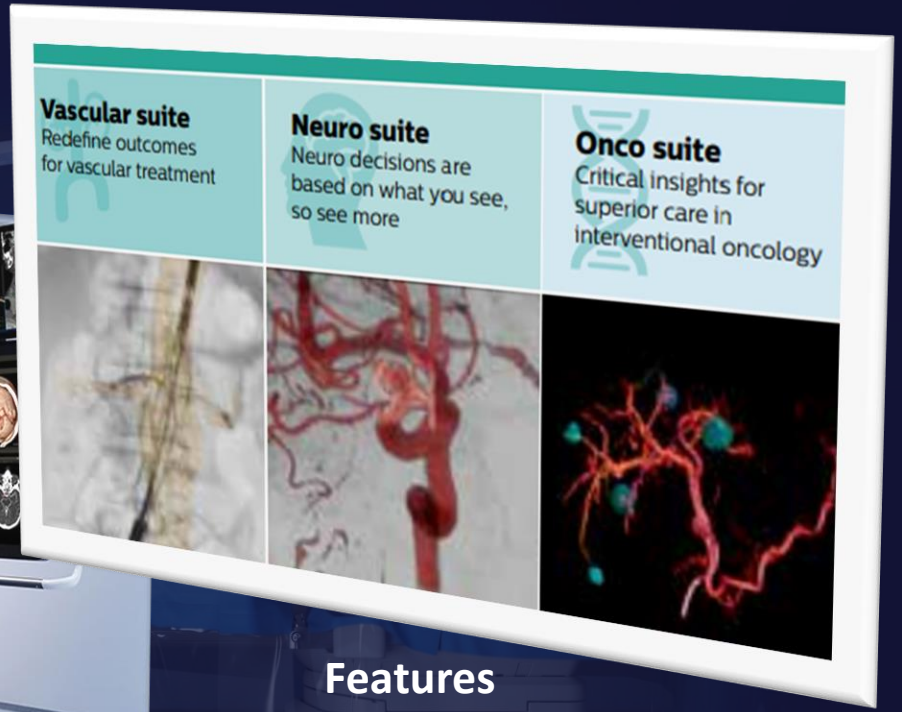
Martijn van Tienen, Ronald Holthuizen & Rob Albers
Image Guided Therapy Systems – R&D – iApps
April 2025

innovation you


PHILIPS

Leadership ask

Platforms



Features

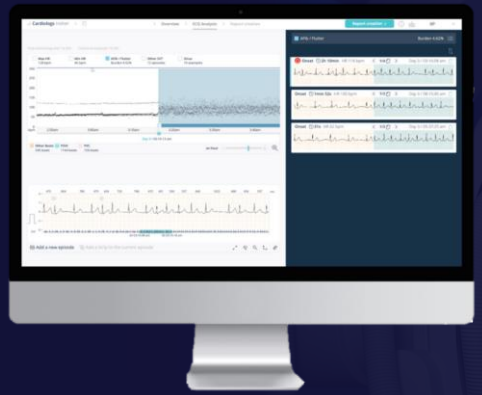
Updates

Problem statement: Release cadence > 1 year

- 15+ Software as Medical Devices
- 2M+ Software Lines of Code
- Agile Release train : 8 feature factory teams
- Releasing can take more than a year

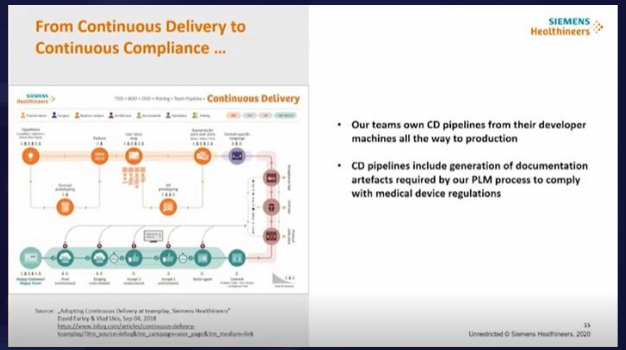
A look at our competition

Philips CardioLogs



>60 updates in 5 years (2018-2024)

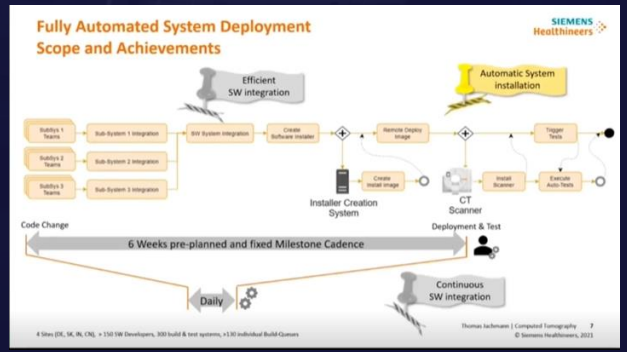
Siemens Teamplay



Cydar Maps

>100 updates in 5 years (2016-2021)

Siemens CT



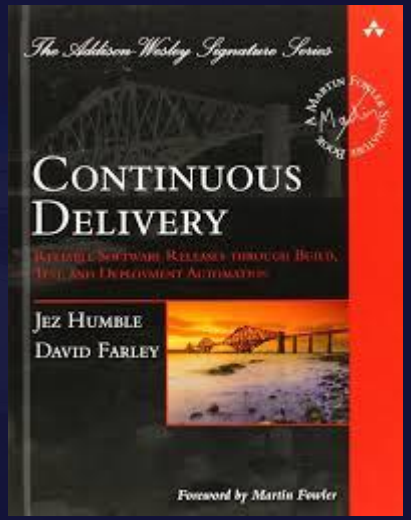
Leadership ask

Releasing software can take more than one year

How to reduce the time <u>after</u> end of development to 1 week <u>AND</u> with improved quality?

"If it hurts, do it more often"

Jez Humble

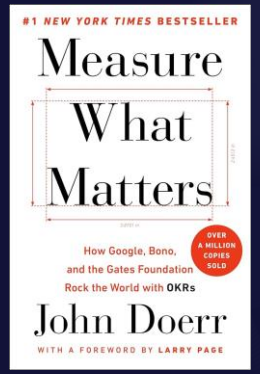


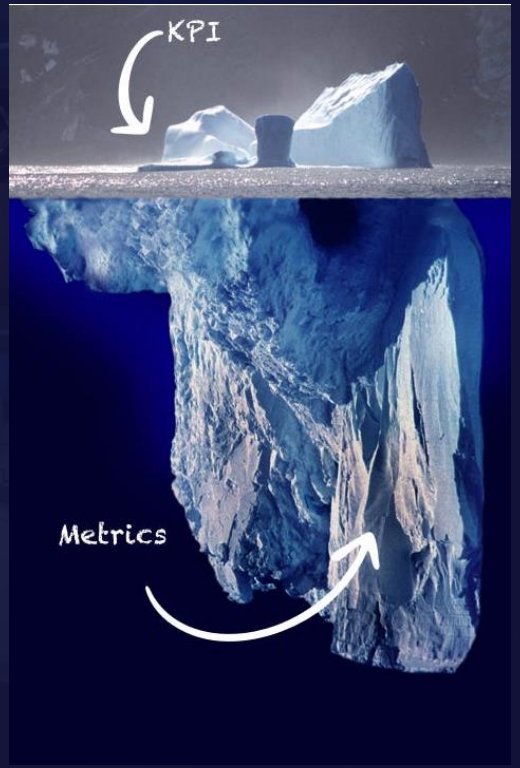
We are used to track many unhelpful KPIs

- Feature estimation predictability
- Team capacity
- Team burndown
- Team velocity
- # of bugs found
- % code coverage

Measure the most critical and impactful: business outcomes

- *“Start celebrating results: you cannot cheat shipping!”* Microsoft
- Value stream management | DORA metrics | Customer feedback





- DORA metrics combined with customer feedback inform teams where to focus improvement efforts and how to position their product and services against competitors

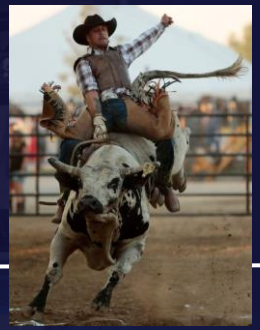
Value stream management

(System) Requirements Engineering → (Software) Development Engineering → Testing & Bug Fixing Features → Testing & Bug Fixing Regressions → End of Development → Verification & Validation → Quality & Regulatory → External Release

Bug Fixing Requirements

Bug Fixing Features

Bug Fixing Regressions



Production Lead Time (for Change)

Release Time

Release time can take up more than one year !

How to reduce this to 1 week <u>AND</u> with improved quality?

Value stream management

(System) Requirements Engineering

(Software) Development Engineering

Testing & Bug Fixing Features

Testing & Bug Fixing Regressions

End of Development

Verification & Validation

Quality & Regulatory

External Release

Bug Fixing Requirements

Bug Fixing Features

Bug Fixing Regressions

Bug Fixing Requirements

Bug Fixing Features

Bug Fixing Regressions

Shift Left

NEW

(System) Requirements Engineering

(Software) Development Engineering

Verification Features

Verification Regressions

Validation Internal Release

Quality & Regulatory

External Release

Value stream management

Bug Fixing Requirements

Bug Fixing Features

Bug Fixing Regressions

Shift Left

NEW

(System) Requirements Engineering

(Software) Development Engineering

Verification Features

Verification Regressions

Validation Internal Release

Quality & Regulatory

External Release

BDD: Requirements as Acceptance tests

Automation & Living Documentation

Where we now

- **Behavior Driven Development**
- **3-Amigo sessions (creating aligned view)**
- **Single Source of Truth (feature file) in GIT**
- **Continues integration / Continues deployment (CI/CD) through GitHub**

Learnings so far

- **Writing feature files takes more time! (pain taken upfront)**
- **More issues are found earlier**

Learnings so far

- **Scale-up is hard!**
- **Created guidelines document related to BDD:**
  - **Process and way of working**
  - **Domain Specific Language explanation and usage**

```
@Status.Feature.Productized
@DocGen.Section.FR.General
@PR.SmartNavigator.FR.DataIdentification
Feature: PR.SmartNavigator.FR.DataIdentification
The clinical product shall support the display of related patient informati

<Rationale>It must be clear for the clin  user to which patient the clin
<Order>0040</Order>
```

Status tag of Feature, see @Status

Naming of feature matching PRS-HighLevel

Naming of Feature matching PRS-HighLevel

Feature text matching PRS-HighLevel requirement text

Rationale text matching PRS-HighLevel Rational, see <Rationale> </Rationale>

Position of Requirement inside of PRS paragraph, see <Order> </Order>

Learnings so far

- **Scale-up is hard!**
- **We want back from 3 smaller teams into 1 big team (with central 3-Amigo)**

Learnings so far

- **Feature file typically focused on (the happy flow) scenarios to explain the rule**
- **Formal evidence should contain (more) corner cases**
- **2-Step approach to unblock development as soon as possible**

Learnings so far

- **Automation using Image Comparison**
  - **Pro:**
    - **Easier to prove correctness in formal evidence**
    - **Roughly doing what manual tester would do**
  - **Con:**
    - **Testing at highest level**
    - **Hard to be used on non-deterministic parts of the system (e.g. Radiation)**

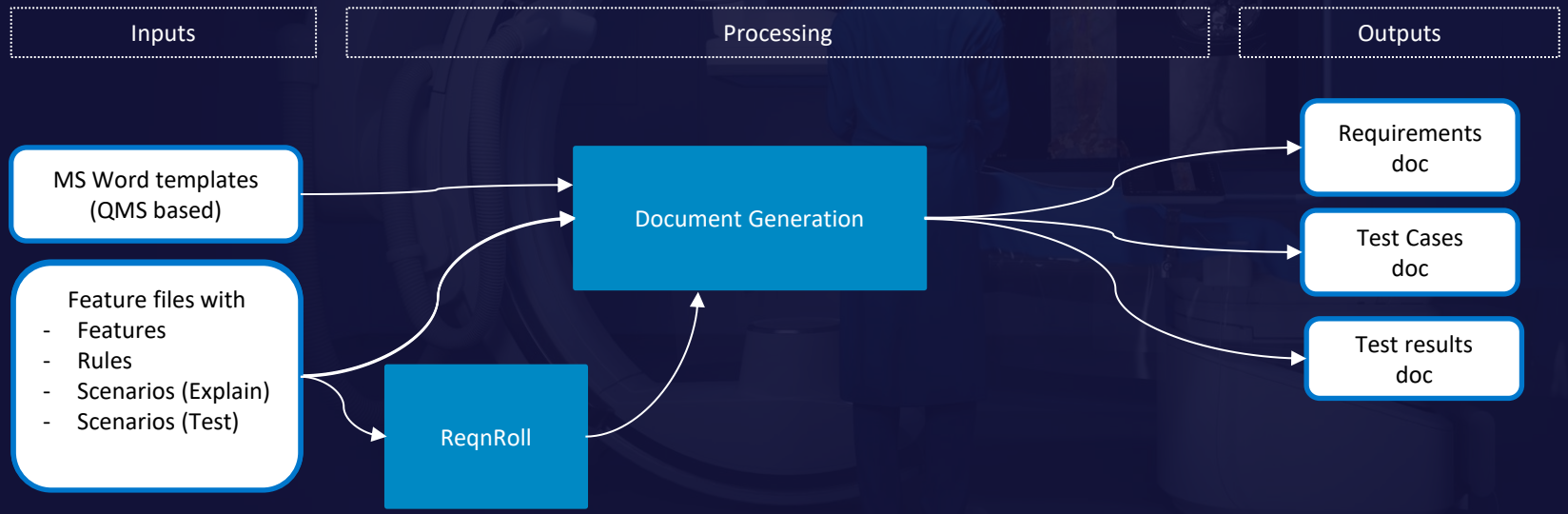
  - **Ongoing: Coupling formal evidence with class level output**

Learnings so far

- **Manual testers still needed!**
- **Focus shifting from regression like testing to exploratory / workflow testing**

Learnings so far

- **We needed custom Document generation tooling**



Learnings so far

- **Introduce promotional model for publishing requirements to formal documentation**

Documentation == Product

Last update slide: May 2024

Feature file: Tagging @Status.Rule.Draft, @Status.Rule.Approved, @Status.Rule.Productized

Tags to control official status of Rules

- @Status.Rule.Draft
  - This Rule (and its scenario's) were **not** yet reviewed by stakeholders
  - This Rule (and its scenario's) will **not** end up in any generated document
  - Do not create implementation + StepDefinition as it might still change
- @Status.Rule.Approved
  - This Rule (and its scenario's) were reviewed and approved by our stakeholders (e.g. System Design, Clinical Marketing,...)
  - This Rule (and its scenario's) will **not** end up in our generated documents
  - Implementation + StepDefinition update can start
- @Status.Rule.Productized
  - This Rule (and its scenario's) are fully completed (Stakeholders agree, implementation is done, BDD testcase is passing)
  - This Rule (and its scenario's) will end up in our generated documents
  - Matrix team can check this Rule on an actual system

```
@Status.Rule.Productized
Rule: PR.SmartNavigator.FR.DataIdentification.Patient
SmartNavigator shall always display the patient demograph

- Patient name (Last name, First name, Middle name),
- Patient id,
- Patient date of birth.
```

Value stream management

Bug Fixing Requirements

Bug Fixing Features

Bug Fixing Regressions

Shift Left

NEW

(System) Requirements Engineering

(Software) Development Engineering

Verification Features

Verification Regressions

Validation Internal Release

Quality & Regulatory

External Release

Requirements as Acceptance tests

Automation & Living Documentation

Value stream management

NEW

Bug Fixing Requirements

Bug Fixing Features

Bug Fixing Regressions

(System) Requirements Engineering

(Software) Development Engineering

Verification Features

Verification Regressions

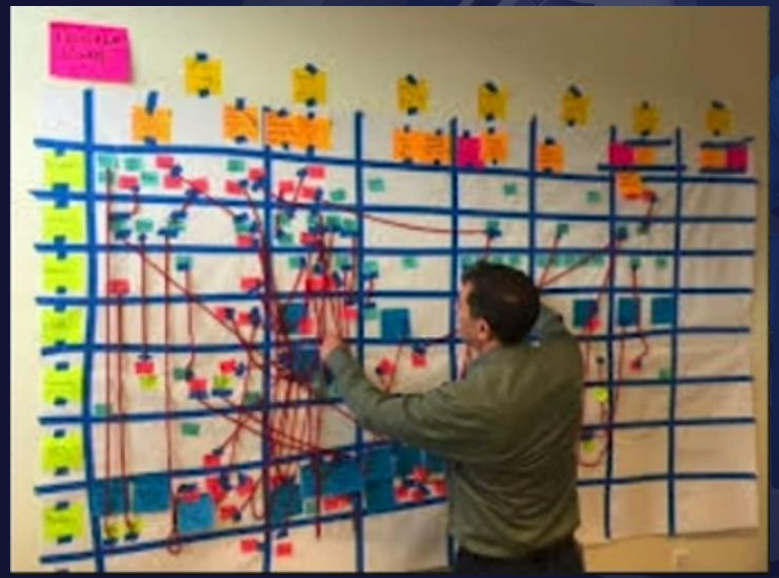
Validation Internal Release

Quality & Regulatory

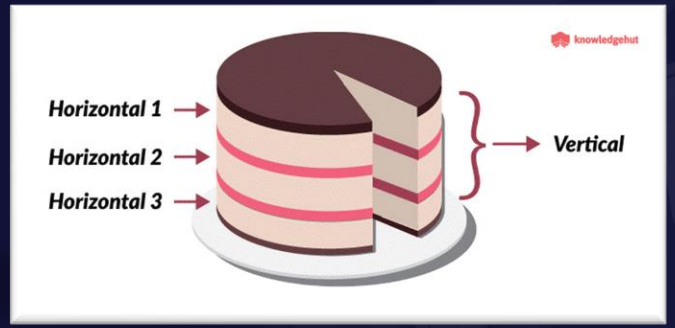
External Release

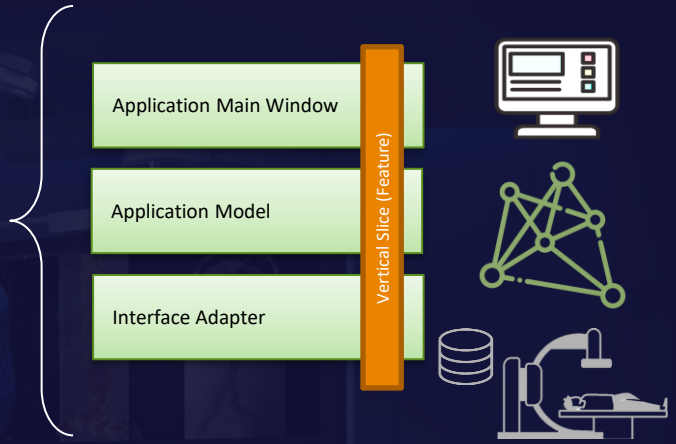
Scaling Engineering for fast flow

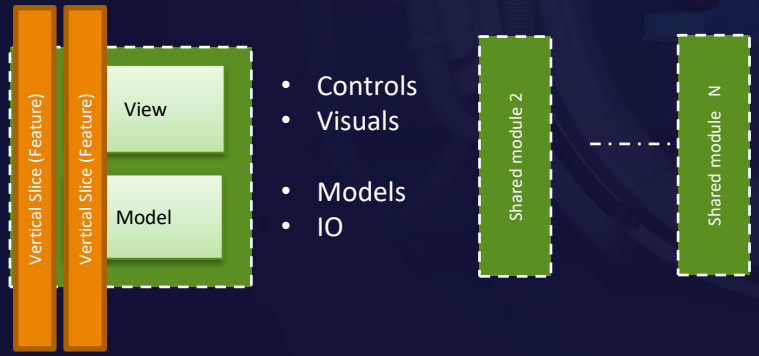
E2E value stream mapping: (external) dependencies

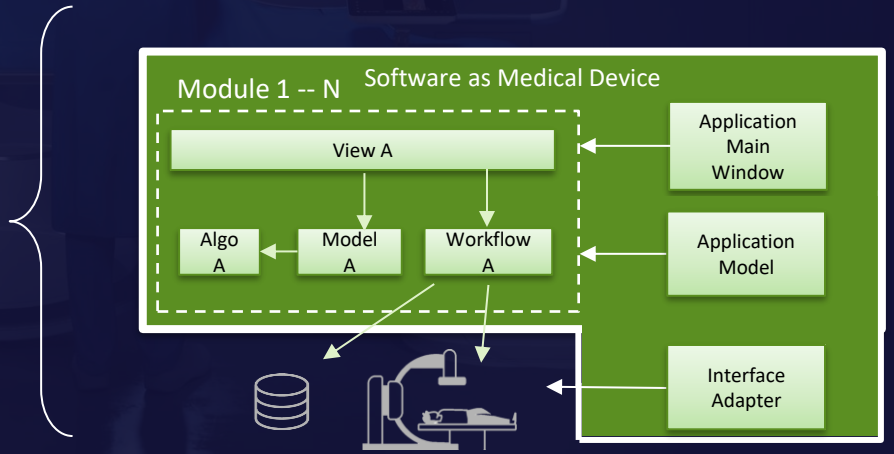


Modular Architecture Design



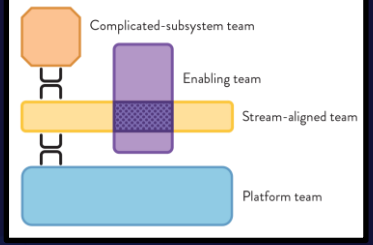


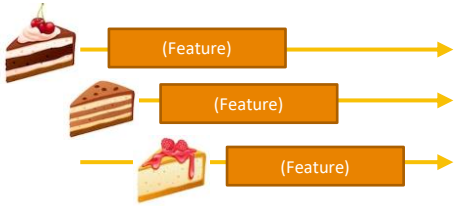


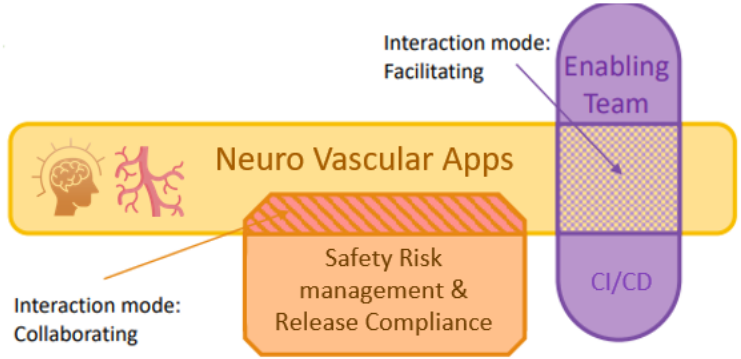


Scaling Engineering for Fast Flow : Team Topologies

Design our teams to match the required software architecture

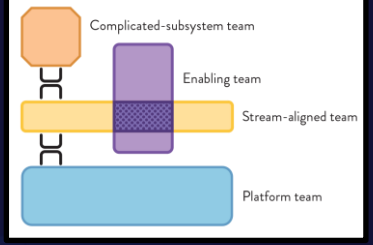


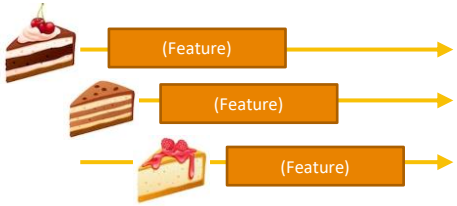


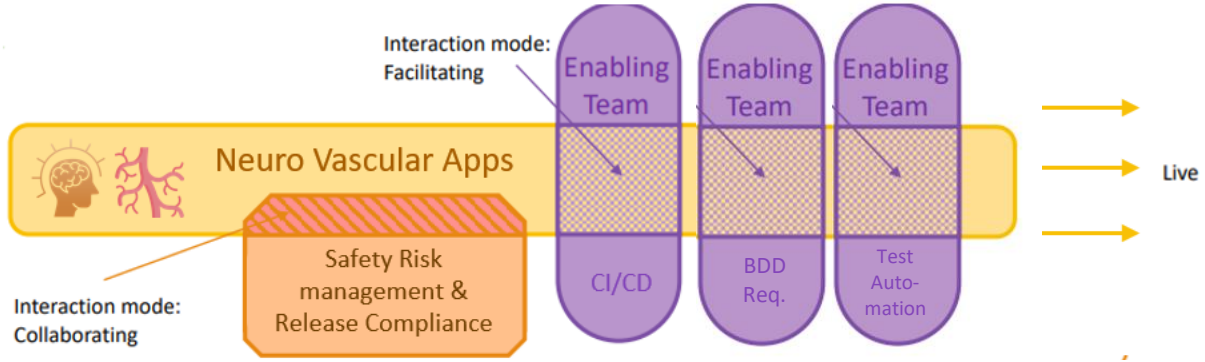


Scaling Engineering for Fast Flow : Team Topologies

Design our teams to match the required software architecture

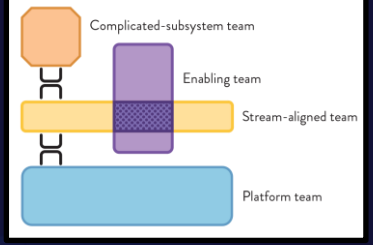


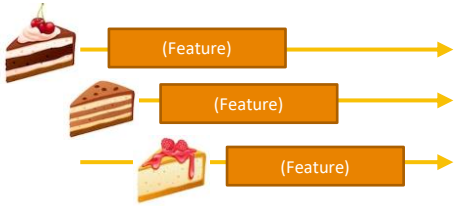


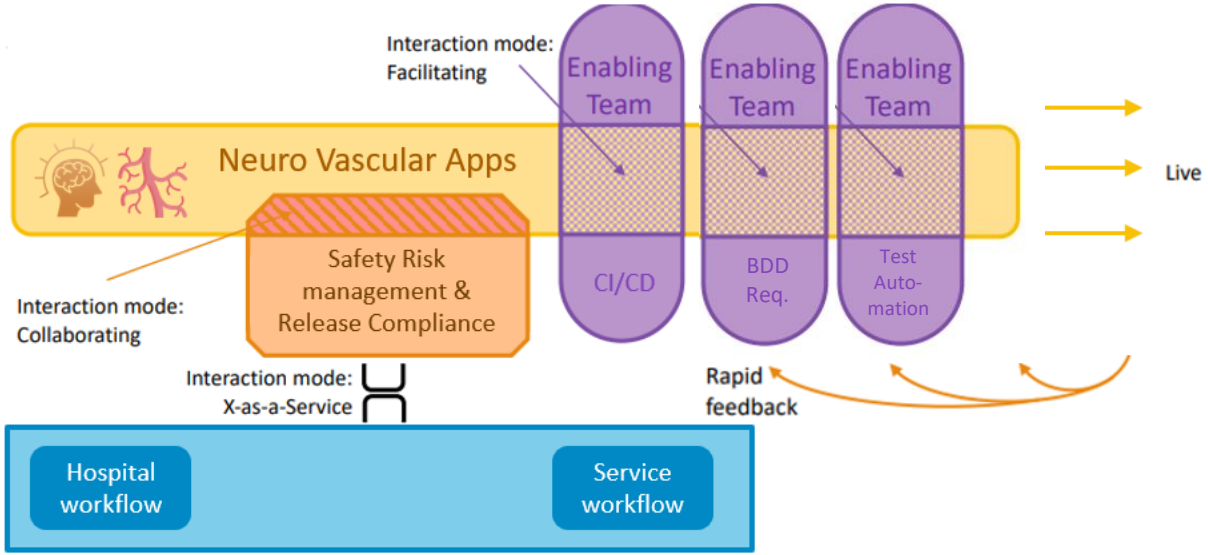


Scaling Engineering for Fast Flow : Team Topologies

Design our teams to match the required software architecture

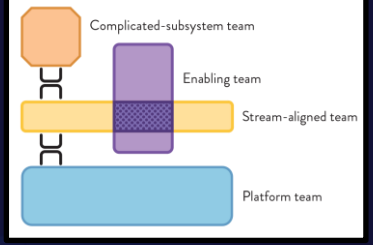


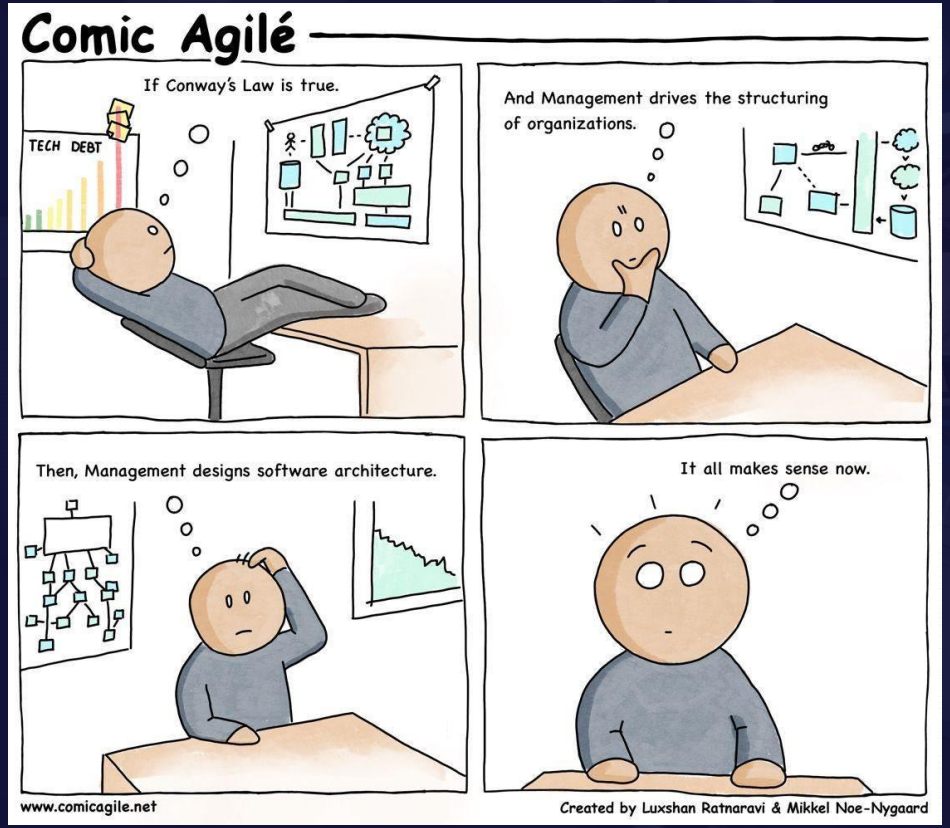




Scaling Engineering for Fast Flow : Team Topologies

Design our teams to match the required software architecture





Learnings so far

VS

Task- switching

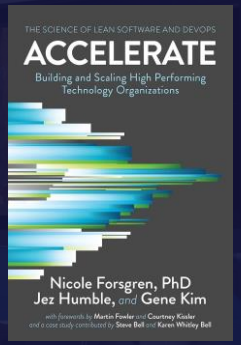
Distributing the (changed) workload across teams

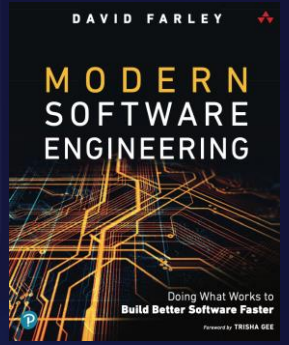
Learnings so far

Scaled Agile on top of traditional project management

VS

Continuous Delivery (CD) engineering discipline





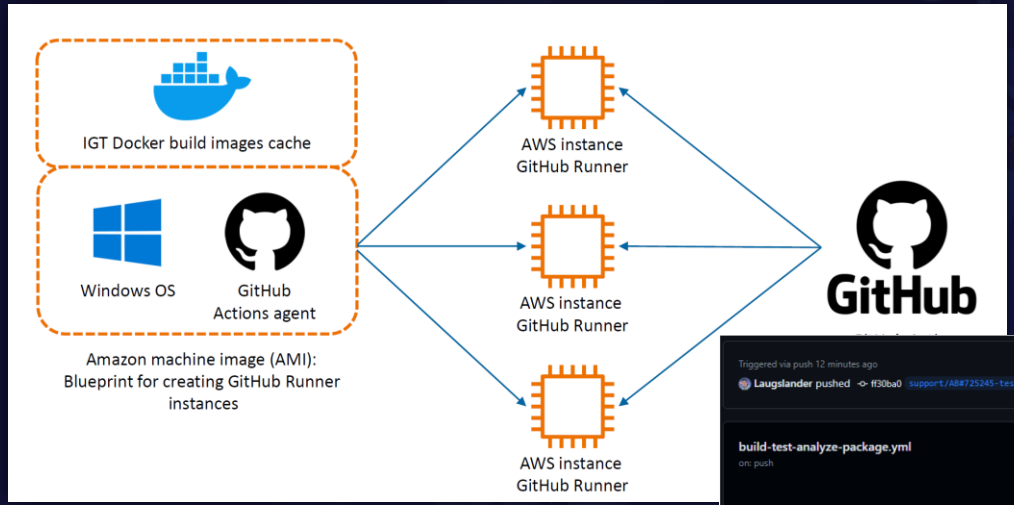
High coordination and alignment cost for predictability

Ambiguous management layers and process roles

Descaling Agile and Decouple for Speed

Continuous Improvement by measuring flow

Continuous Integration



Tools

Technology Radar

Volume 30

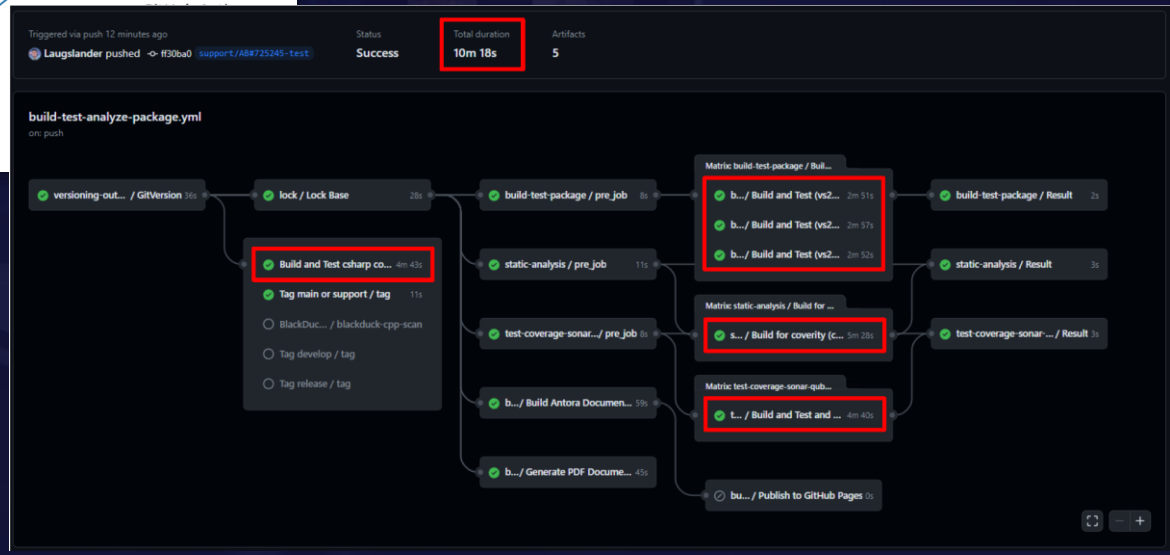
April 2024

An opinionated guide to today's technology landscape

60. Philips's self-hosted GitHub runner

Trial

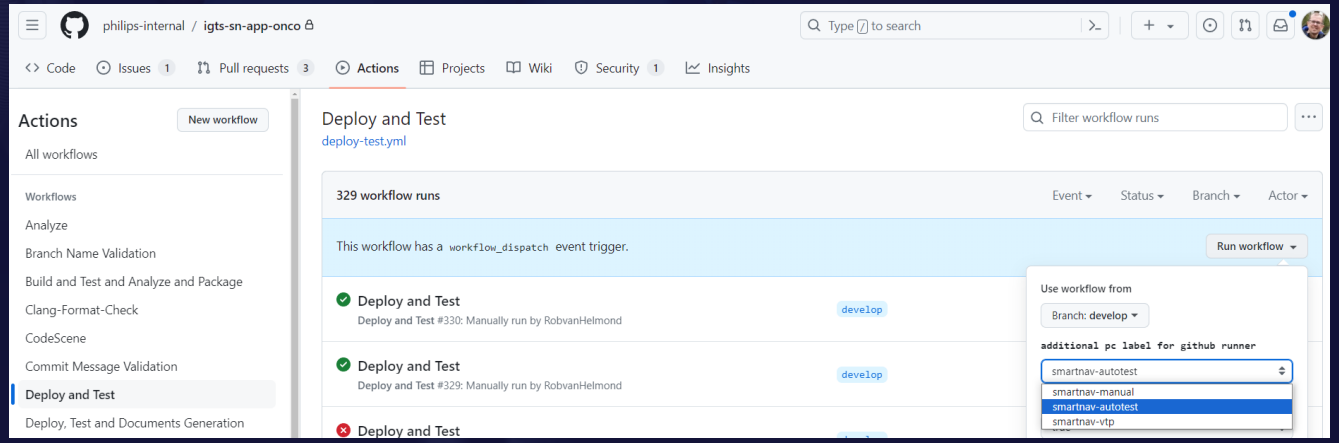
While GitHub Actions runners cover a wide range of the most common run times and are quickest to start with, teams sometimes need to manage self-hosted runners, such as when organizational policy only allows deployments to a privately hosted infrastructure from within the organization's own security perimeter. In such cases, teams can use Philips's self-hosted GitHub runner, a Terraform module that spins up custom runners on AWS EC2 spot instances. The module also creates a set of Lambdas that handles lifecycle management (scaling up and down) for these runners. In our experience, this tool greatly simplifies the provisioning and management of self-hosted GitHub Actions runners. An alternative for teams that use Kubernetes is actions-runner-controller.



DevOps as self-service

Continuous Deployment

- Local development PC
- Target PC + Software simulator
- Target PC + Virtual Azurion
- Target PC + Azurion Lab
  - Biplane
  - Monoplane
  - …

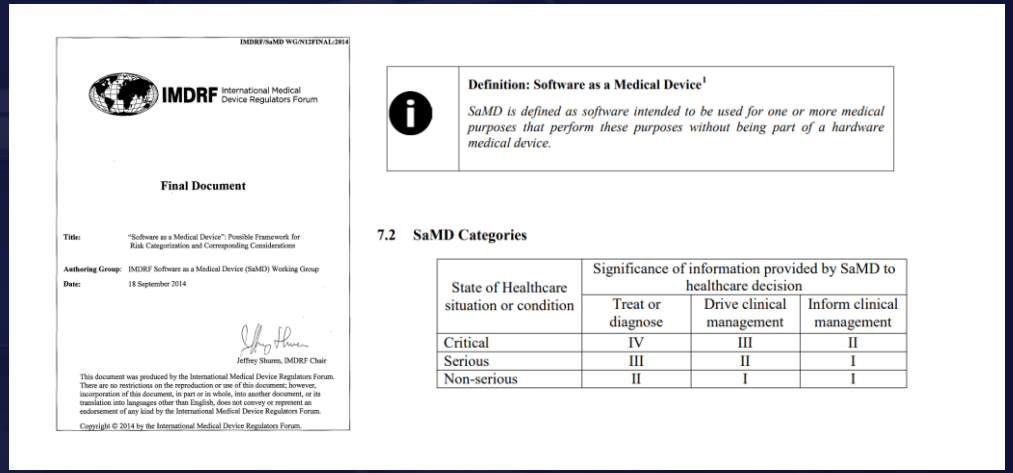


BDD in the regulated medical device industry

From BDD to full and continuous compliance

Regulatory perspectives on medical software.

- Risk Management
- Clinical validation – Safety & Effectiveness
- Usability evaluation – Formative & Summative
- Failure mode and effects analysis
- Algorithms
- Cybersecurity
- IEC62304, IEC 82304, IEC 80001, …



IMDRF/SaMD WG/N12FINAL:2014

IMDRF International Medical Device Regulators Forum

Final Document

Title: "Software as a Medical Device": Possible Framework for Risk Categorization and Corresponding Considerations

Authoring Group: IMDRF Software as a Medical Device (SaMD) Working Group

Date: 18 September 2014

Jeffrey Shuren, IMDRF Chair

This document was produced by the International Medical Device Regulators Forum. There are no restrictions on the reproduction or use of this document; however, incorporation of this document, in part or in whole, into another document, or its translation into languages other than English, does not convey or represent an endorsement of any kind by the International Medical Device Regulators Forum.

Copyright © 2014 by the International Medical Device Regulators Forum.

Definition: Software as a Medical Device[1]

SaMD is defined as software intended to be used for one or more medical purposes that perform these purposes without being part of a hardware medical device.

7.2 SaMD Categories

State of Healthcare situation or condition	Significance of information provided by SaMD to healthcare decision		
	Treat or diagnose	Drive clinical management	Inform clinical management
Critical	IV	III	II
Serious	III	II	I
Non-serious	II	I	I

Why Agile/BDD and why change the way we are working?

- For decades there have been perspective differences between traditional system engineering processes and Agile/iterative software development, for software-only products.[1]

- The software industry has moved to iterative development with quick development cycles.
  - Not only “new” companies, like Google & Facebook, but also legacy companies like Microsoft have moved to this approach.[2,3]
  - Shipping of features moved from a yearly cycle to a 3-week sprint cycle at Microsoft.[2]
  - Agile improved R&D efficiency by 20-30% for medical software development in Abbott.[4]

- With BDD, Agile way of working can be combined with formal requirements & verification management required by regulated industries, while still supporting iterative development cycles.
  - This enables regulated industry to better adopt iterative development methods.
  - High degree of document and process automation are required to implement this successfully.

- FDA & regulatory bodies are starting to recognize Agile as best practice as are becoming more open to supporting this development methodology.[5]

Cheetah from SIT to end of SVER to 10 months.
How to reduce this to 1 week with improved quality?

6.3.1.5 Aligning incremental design validation with regulatory requirements

about alignment with regulations. In the requirements for Design Validation, 21 CFR 820.30(g) specifically requires Design Validation to be performed on “final production units or their equivalent”, and ISO 13485 Section 7.3.7 requires Design Validation to be performed on “representative product,” which includes “initial production units, batches or their

These requirements can be misinterpreted to mean that Design Validation must be near the end of product development, and that Design Validation activities performed earlier in the development cycle do not meet the regulatory requirements. While some Design Validation activities should certainly be done near the end, deferring all

about alignment with regulations. In the requirements for Design Validation, 21 CFR 820.30(g) specifically requires Design Validation to be performed on “final production units or their equivalent”, and ISO 13485 Section 7.3.7 requires Design Validation to be performed on “representative product,” which includes “initial production units, batches or their

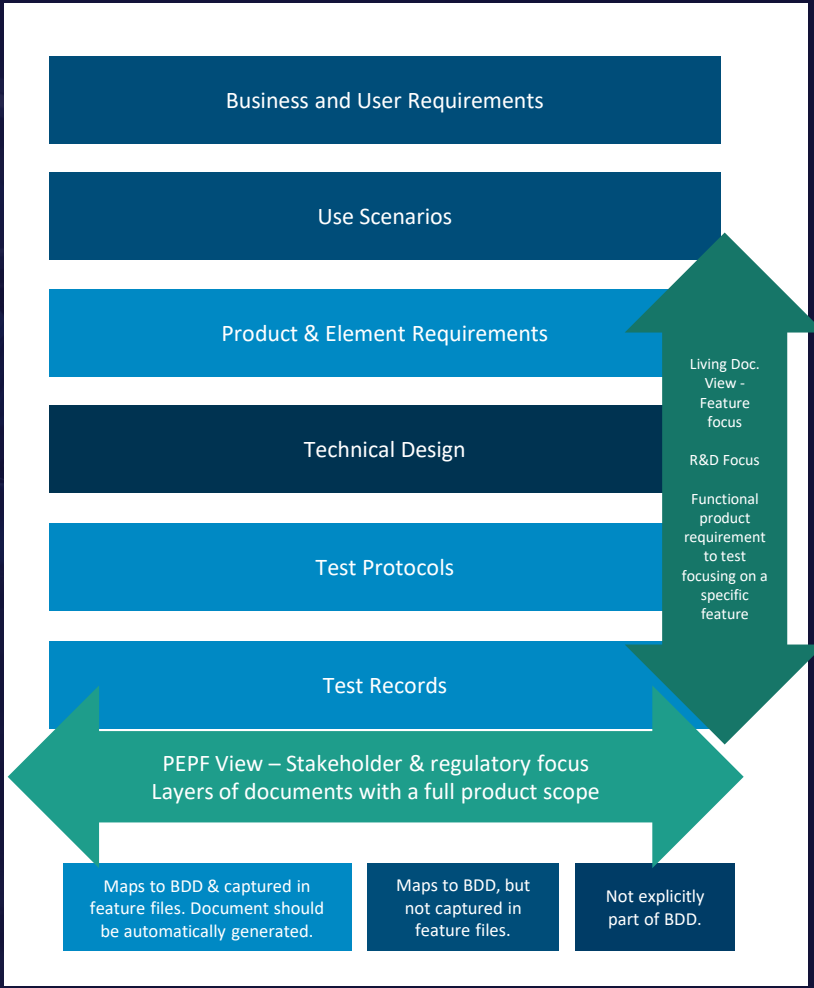
These requirements can be misinterpreted to mean that Design Validation must be near the end of product development, and that Design Validation activities performed earlier in the development cycle do not meet the regulatory requirements. While some Design Validation activities should certainly be done near the end, deferring all

1 = Requirements Engineering in Agile Software Development, De Lucia et al. (2003)
2 = Facebook release cycles
3 = Microsoft iterative development
4 = Adopting Agile in an FDA Regulated Environment
5 = Guidance on the use of Agile in Medical Device Software for FDA compliance

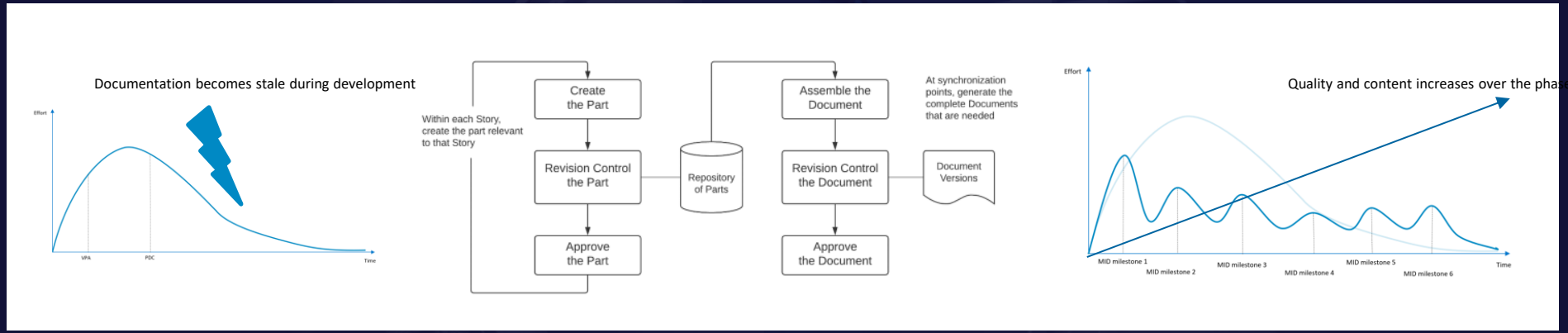
Using BDD to create PEPF deliverables

Single source of truth in Git .feature files to generate PEPF deliverables

- Living Documentation with a **single source of truth** provides the **same content** as PEPF (functional product requirements, element requirements, test specifications & test reports), although typically structured as **different views**.
- Test specification of BDD is an **executable specification** in the Gherkin syntax.
- **Multiple views** ensure that the relevant information is available for at the right moment for the right person, where the BDD view will help **drive consistency over the multiple document layers**.
- **Living Documentation**: At **any moment**, PEPF documents are of **release quality** and can be **automatically generated**.
- **Tooling is needed** to ensure a single source of truth that automates the **different views** both the BDD Living documentation view and PEPF document view and that links to **test driven development**.



How to write, control and approve documents as a sum of its parts?

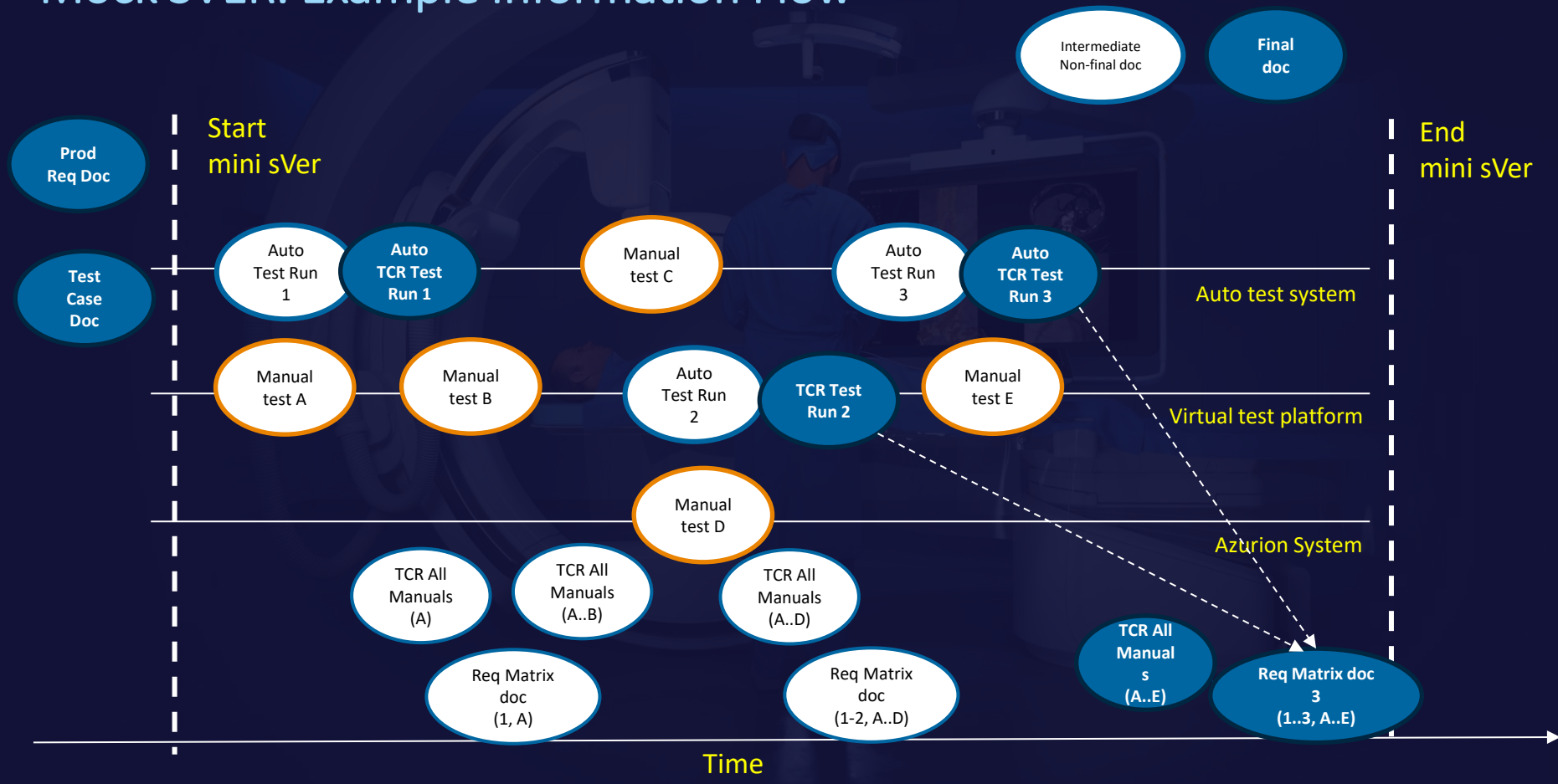


- This should include documents that have dependencies towards the requirements documents, like FMEA, Risk Management, decision logs & traceability documents.

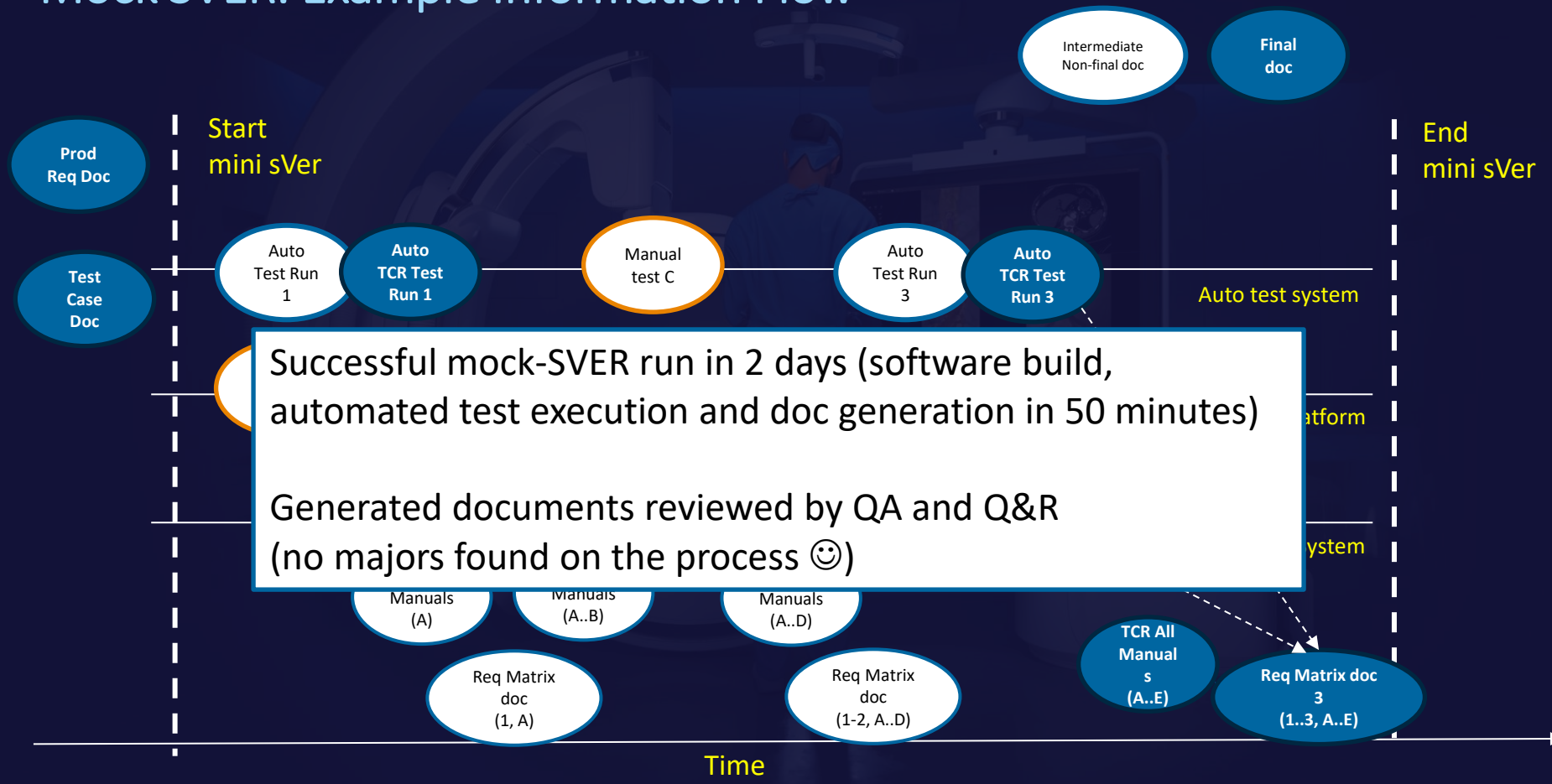
- A cadency should be established where documents are assembled & reviewed.
- High-level documentation is needed e.g. for regulatory submissions.

- Start executing incremental document generation, with multiple complete approval cycles.

Mock SVER: Example Information Flow



Mock SVER: Example Information Flow



Successful mock-SVER run in 2 days (software build, automated test execution and doc generation in 50 minutes)

Generated documents reviewed by QA and Q&R (no majors found on the process ☺)

Release-Readiness Metrics Summary

▲ Trend Improving
▼ Trend Worsening
= Trend Stable

Bugs found by Matrix

5 ▲

Target: < 3

BDD Execution Time

48m ▼

Target: < 120 minutes

DORA Metrics

Deployment Frequency	Lead Time For Changes
On Demand =	< 1 Week =
Target: 1x day - 1x week	Target: < 1 Day
Time To Restore	**Change/Failure Rate**
< 1 Day =	16-20% =
Target: < 1 Day	Target: < 10%

BDD vs Production Code

32 % ▼

Target: < 40%

Requirements Bugs

= 0

Target: < 3

Documents Release Ready

▲ 10 %

Target MS 2: 10%

System Test Cases Automated

▼ 94%

Target: 98%


Requirements as Acceptance tests
Modular Architecture Design
DevOps practices measuring flow
Automation & Living Documentation
Team Topologies
Continuous Compliance
The release time for a Philips Software as Medical Device can be shortened to < 1 week AND with improved quality!

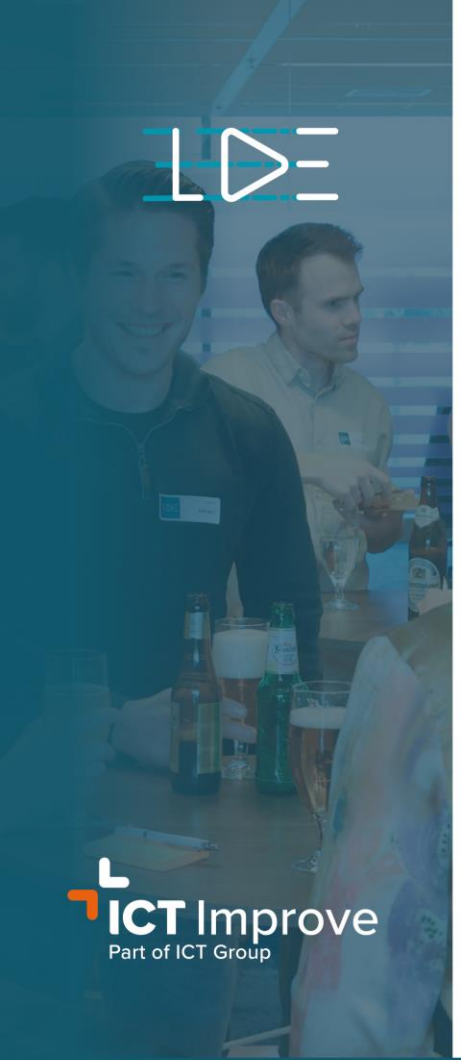
Thank you!

ICT Improve
Part of ICT Group

PHILIPS

Thank you for your attention!

Share your insights using the hashtag **#LDE25** and tag **@ICT Improve**!



PROGRAMME

Living Documentation Event

10 April 2025

14.00	**Walk in**	
14.30	**Opening**	*Auditorium*
14.35 - 15.15	**Keynote Gáspár Nagy** - RAMP up your testing solution: test automation patterns	*Auditorium*
15.25 - 16.10	**Choose between three tracks:**	
	Karl van Heijster Testing: A Philosophical Retrospective	*P083*
	Jennek Geels The journey is the reward	*Auditorium*
15.25 - 17.00	**Workshop Bas Dijkstra & Gáspár Nágy** I know it's only ReqnRoll (but I like it) - Making the most of the Automation phase in BDD (part 1)	*P030*
16.15 - 17.00	**Choose between two tracks:**	
	Rob Albers, Ronald Holthuizen & Martijn van Tienen - BDD, (A)TDD and DevOps practices as a recipe for continuous compliance	*P083*
	Rick Easton Tracy - Castles, not Silos	*Auditorium*
Continuation	**Workshop Bas Dijkstra & Gáspár Nágy** - I know it's only ReqnRoll (but I like it) - Making the most of the Automation phase in BDD (part 2)	*P030*
17.05 - 17.50	**Choose between three tracks:**	
	Jacob Duizer - From Team Topologies to Behavior-Driven Development: Building Teams That Deliver	*P083*
	Pieter Withaar - AI-First BDD, what if we redesign BDD to be AI-first?	*Auditorium*
	Machiel van der Bijl - Model Driven Design (MDD): A new approach to Living Documentation	*P030*
17.55 - 18.50	**Dinner: Beer and pizza's**	
18.55 -19.35	**Keynote: Angelo Hulshout** - GenAI and creativity - threat, or tool	*Auditorium*
19.35 -20.15	**LDE Community + Panel Discussion**	*Auditorium*
20.15 - 21.00	**Drinks**	

ICT Improve
Part of ICT Group