BREDEX
Software-Entwicklung und Beratung

Test automation patterns, workflows and requirements linking

The next steps on the quality journey

Alex Schladebeck, BREDEX GmbH

@alex_schl

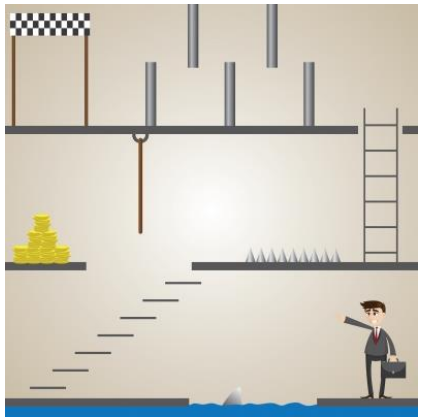
Agenda

[2]

The journey thus far

New challenges

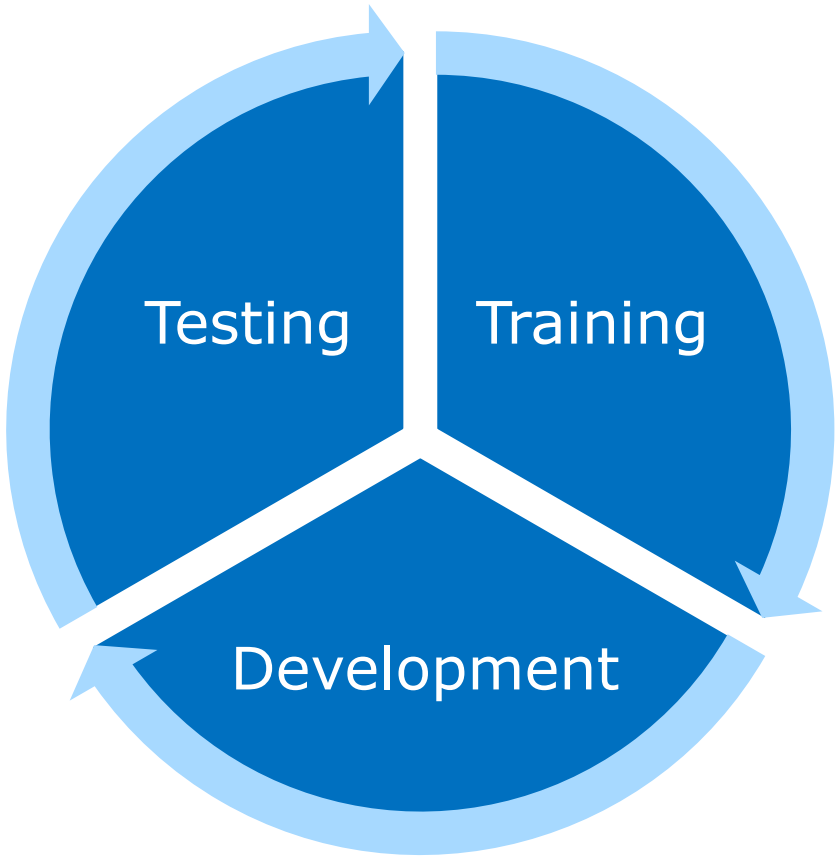
New horizons



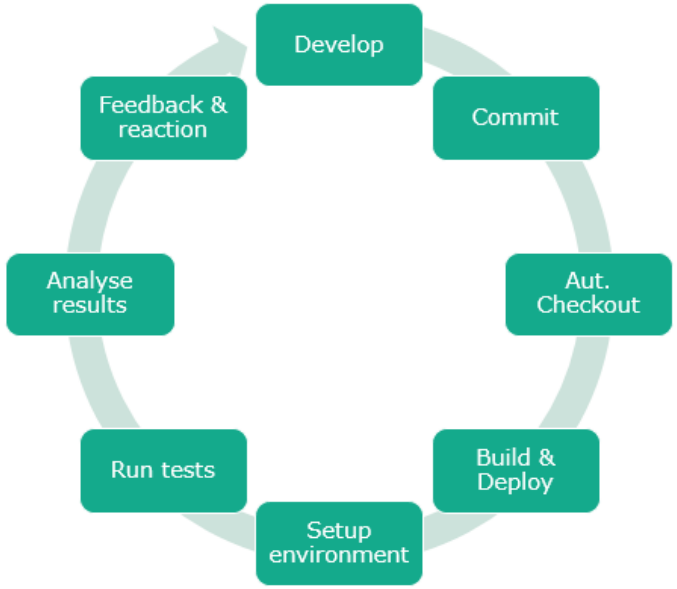
[1]

[3]

Introductions



The journey thus far



Reacting

Continuous integration

[5]

Automated testing

[4]

Preparation

New challenges

- The journey is far from over
- Our latest steps:
  - Improving workflows for test automation
  - Defining and introducting patterns for test automation
  - Integrating test automation better into the team and process

Workflows (1)

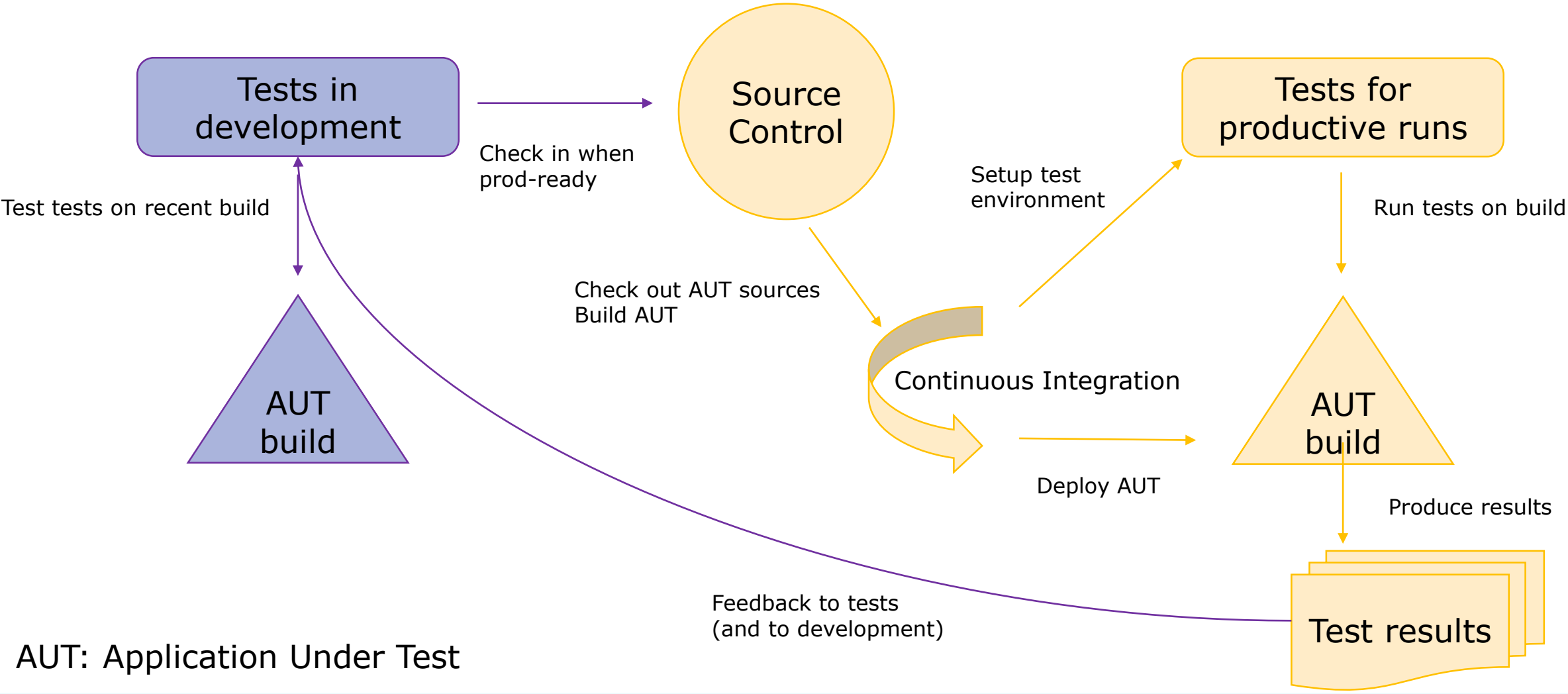
Problem

- Growing test scope
- Changes may be larger and take longer
  - (Wide reaching changes)

Solution

- Split development and productive work

Workflows (1): Split dev and prod



AUT: Application Under Test

Workflows (2)

Problem

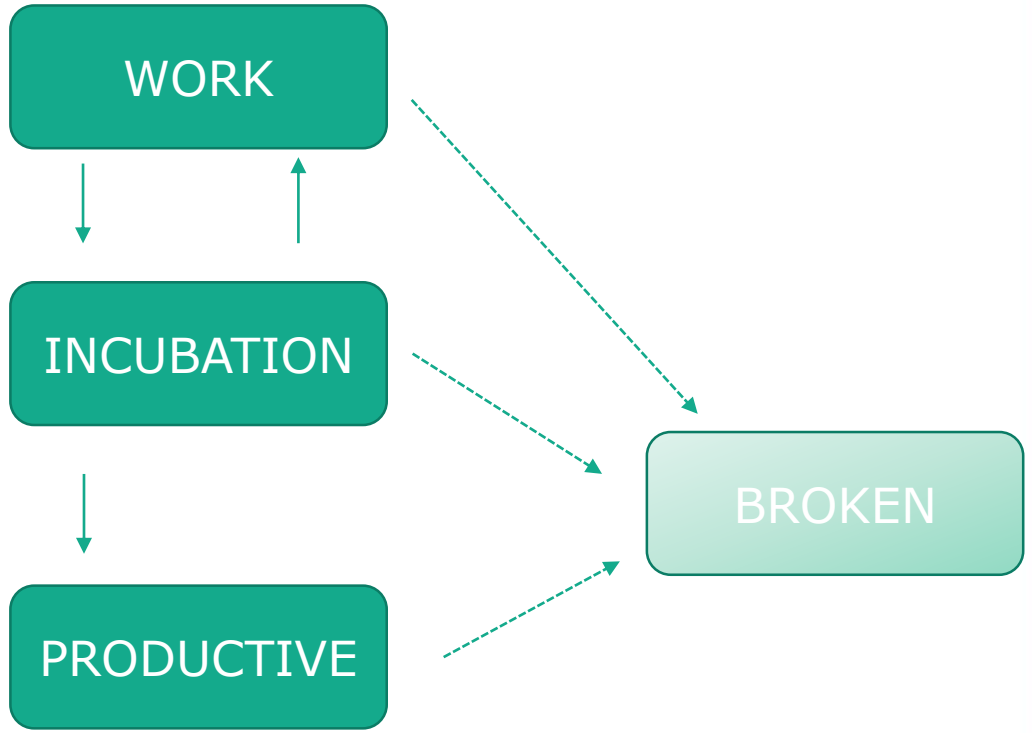
- Accustomed to having successful tests
  - React to regression problems instantly
- New test additions may fail

Solution

- Split test suites according to maturity

Workflows (2): Add an incubation layer

- INCUBATION
  - Tests the test
  - Tests the AUT in correct condition
  - Short lifespan
- PRODUCTIVE
  - Only* regression errors
  - Can be further split
    - Smoke / Full
    - According to AUT area



Test automation patterns

Problem

- Individual preferences for good structure
- Mixing of abstraction levels
- Tests hard to understand / maintain

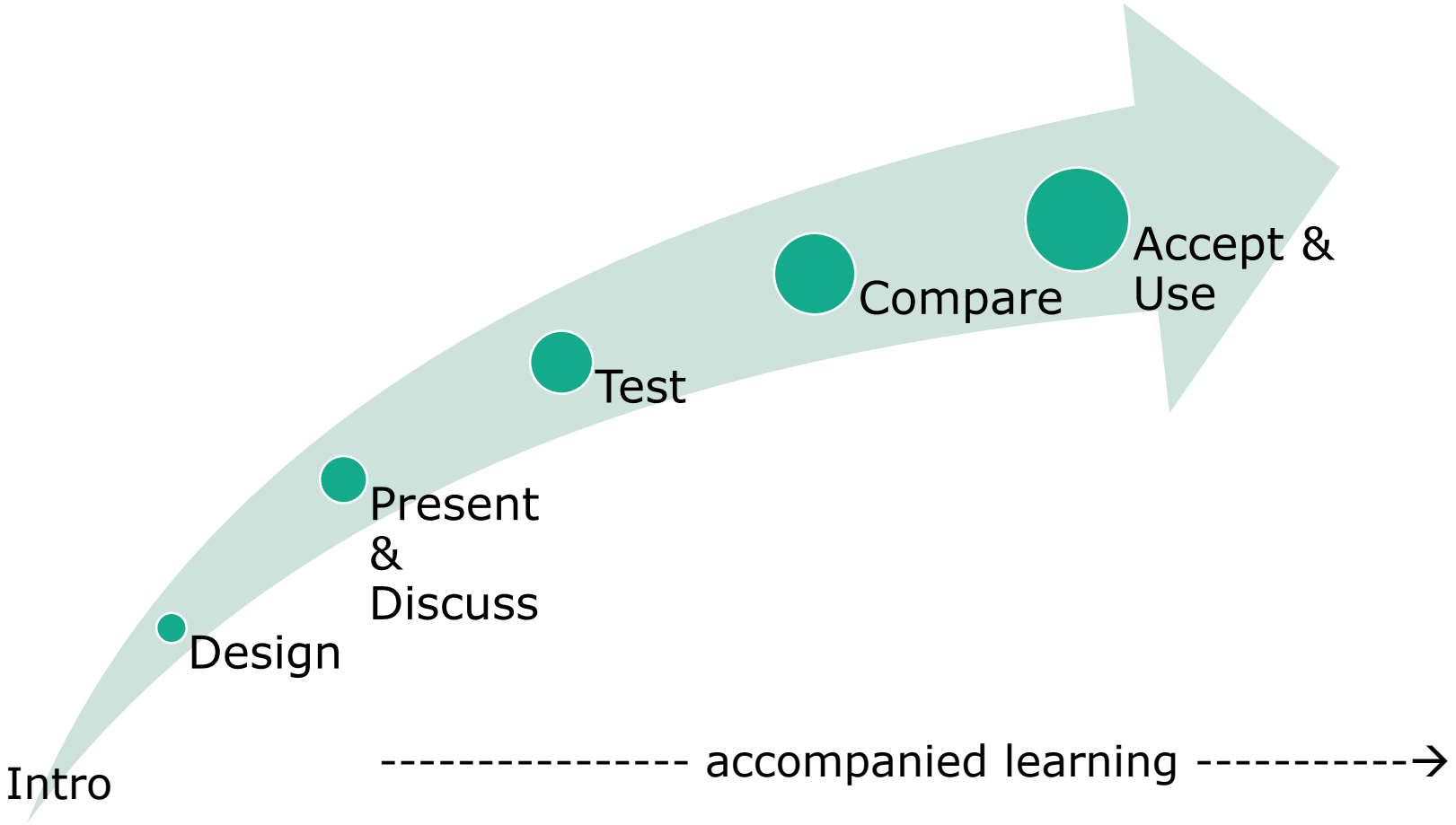
Solution

- Define, introduce and use test automation patterns

A side trip into the world of patterns

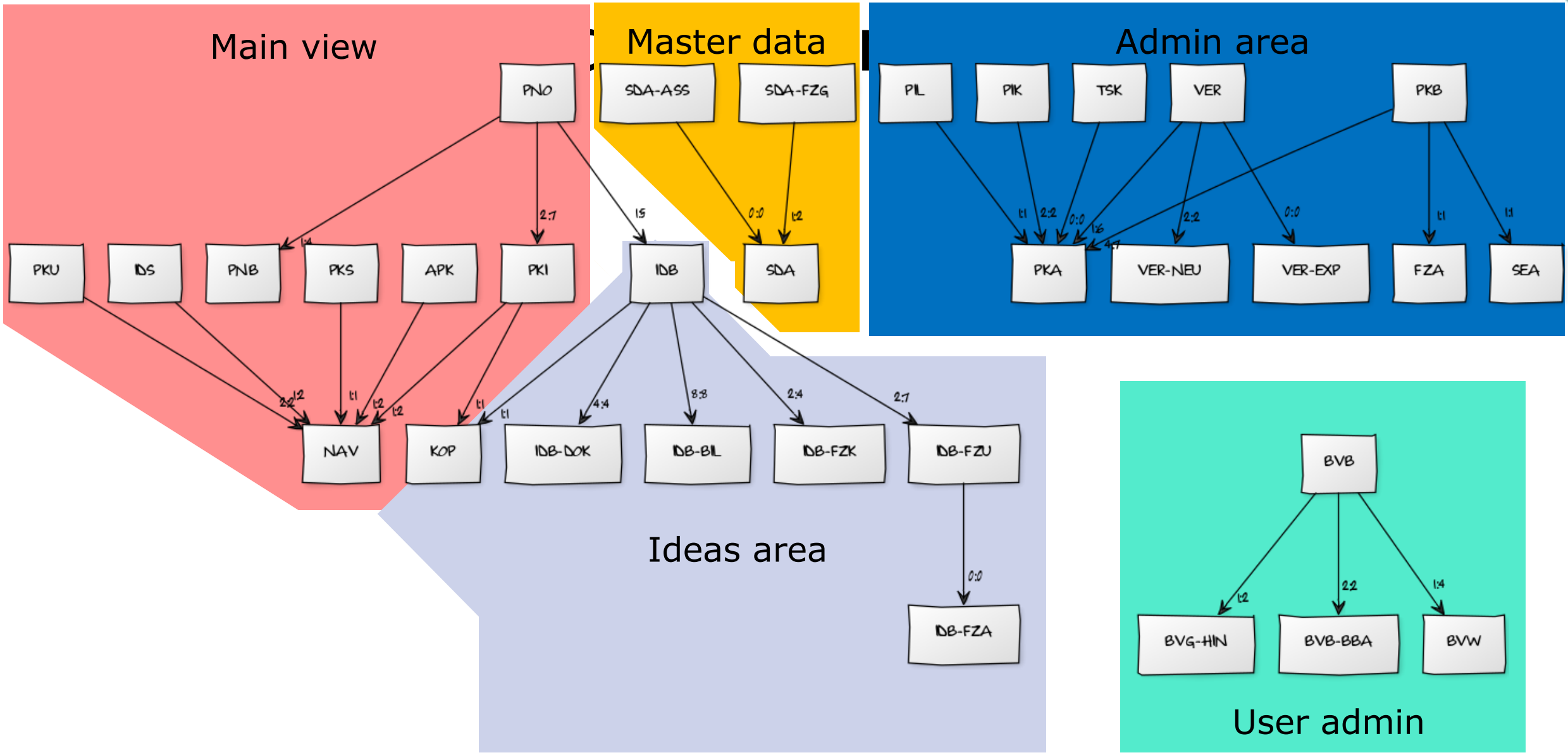
The truth is out there

A side trip into the world of patterns



Main principle: Cohesion & coupling

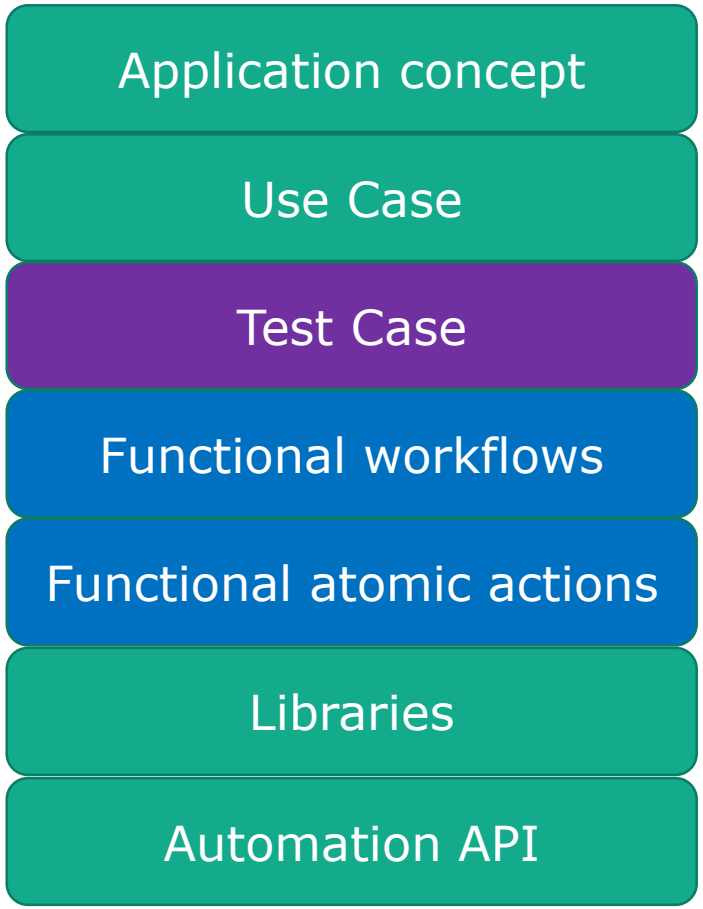
- High internal cohesion
  - Things that belong together are structured together
  - *Contexts*
    - Contain keywords for an area
  - Functional workflows
    - Only contain relevant items
  - Separation of concerns:
    - Navigation
    - Execution
    - Verification
- Minimal coupling
  - Minimal dependencies between keywords across contexts


Main view
PNO
PKU
DS
PNB
PKS
APK
PKI
NAV
KOP
Master data
SDA-ASS
SDA-FZG
SDA
Admin area
PIL
PIK
TSK
VER
PKB
PKA
VER-NEU
VER-EXP
FZA
SEA
IDB
IDB-DOK
IDB-BIL
IDB-FZK
IDB-FZU
IDB-FZA
Ideas area
BVB
BVG-HIN
BVB-BBA
BVW
User admin

1: Cohesion: Abstraction levels

Top Down

Structure tests according to application



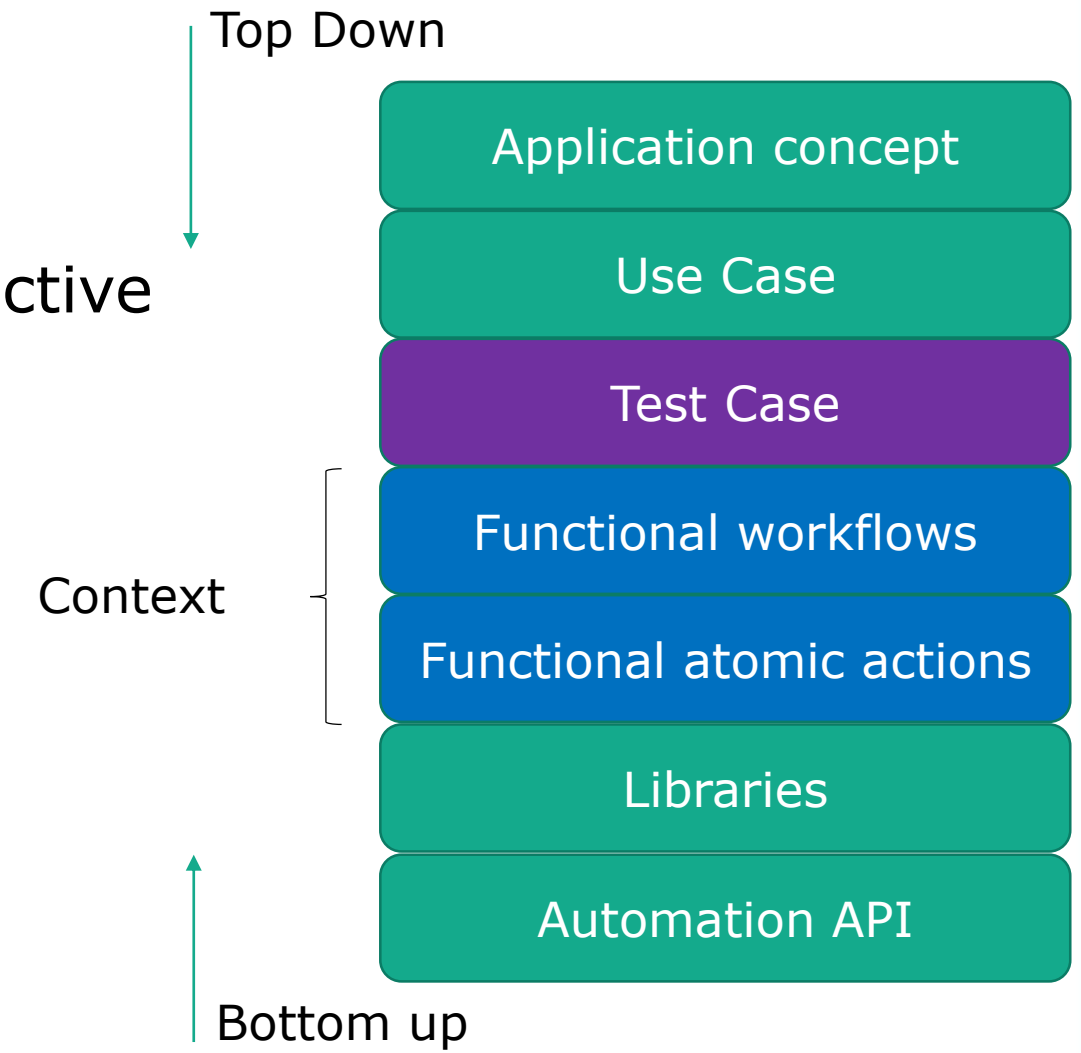
Context (Cohesion)

Bottom up

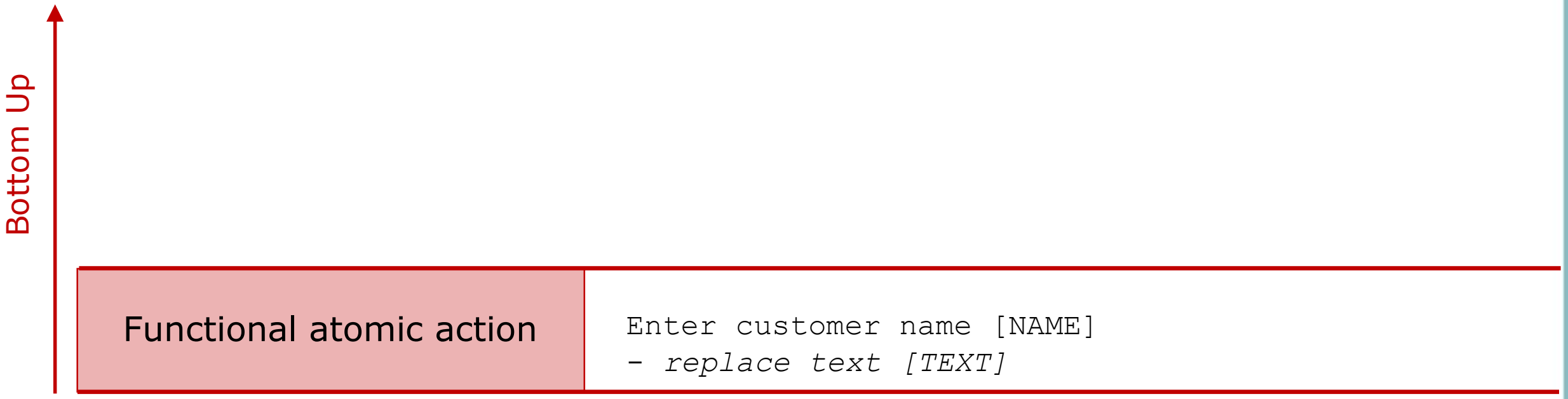
Create keywords for use in automating tests

1: Cohesion: Abstraction levels

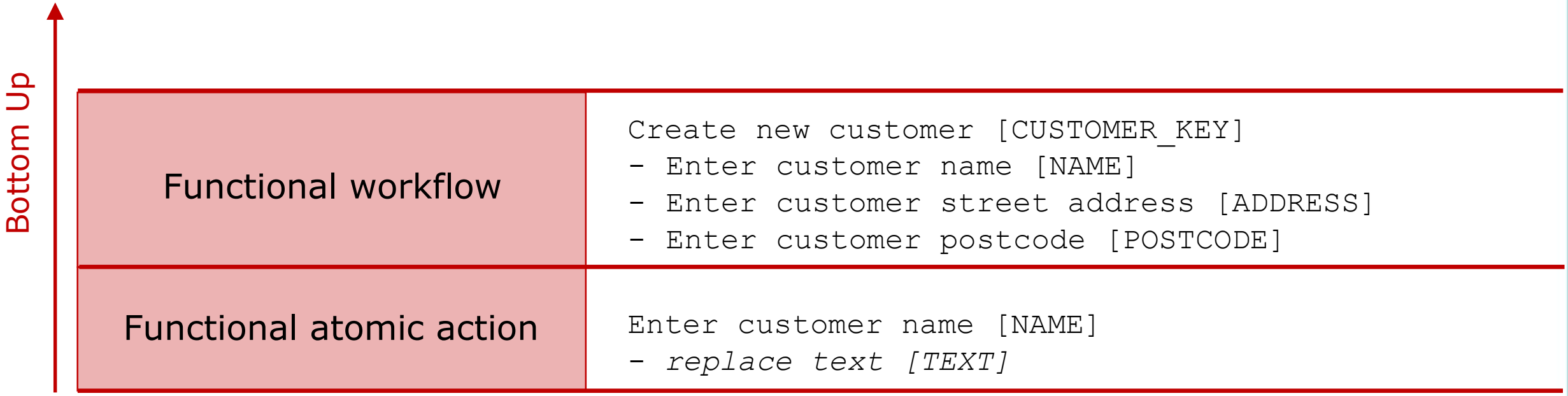
- Structure tests in levels
  - Higher up = user perspective
  - Lower down = automation perspective
  - Test Case is meeting point
    - Readable for team
- Page object pattern (context)
  - One place for
    - Widgets and their operations



1. Cohesion (Contexts)



1. Cohesion (Contexts)



1. Cohesion (Contexts)

Top Down

Bottom Up

Test Case	TC CUS-001:Create new customer - Open new customer dialog - Check that customer does not exist [CUSTOMER_KEY] - Create new customer [CUSTOMER_KEY] - Save new customer - Check that customer exists [CUSTOMER_KEY]
Functional workflow	Create new customer [CUSTOMER_KEY] - Enter customer name [NAME] - Enter customer street address [ADDRESS] - Enter customer postcode [POSTCODE]
Functional atomic action	Enter customer name [NAME] - *replace text [TEXT]*

1: A ‘shining‘ example…

Keyword: Save or open file [MENUPATH; DIALOG_TITLE; PATH_AND_FILE]

Action	Component	Data
Select save or open	Menu	[MENUPATH]
Wait for window	Dialog	[DIALOG_TITLE]
Enter path and file name	File name field	[PATH_AND_FILE]
Click	OK Button	One left click

Problems:
- Very generic
- Uses automation API directly
- Hard to understand
- Hard to find where to edit
- **Different functional workflows**

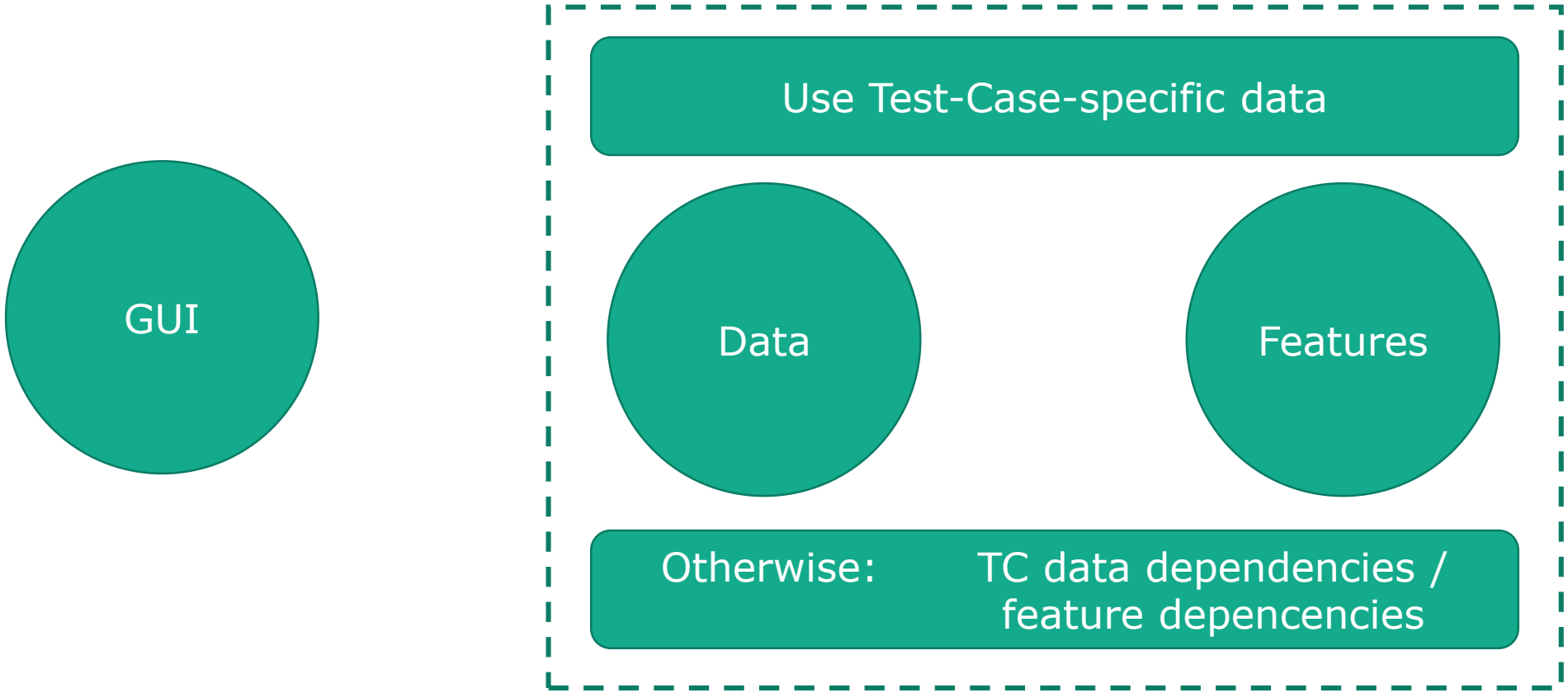
1: A new example

Keyword: Save file (overwrite) [FILENAME]

Action	Component	Data
{Select Save As}	Menu	File/Save
{Wait for Save As Dialog}	Dialog	Save As
{Enter filename in Save As Dialog}	File name field	[FILENAME]
{Click OK in Save As Dialog}	OK Button	One left click
{Confirm overwrite if asked}	----	-----

Advantages:
- Specific to one functional workflow
- Uses functional atomic actions
- Easy to understand without reading details
- Location of keyword is clear

2. Independence of Test Cases



Fresh setup

Symptom
- Test Cases can only be run in a set order

Cause
- Dependencies between Test Cases

Strategy
- Remove dependencies between Test Cases

Pattern
- Fresh setup

- GUI
  - Every Test Case ensures that GUI is in required start state before execution
- Data
  - Should not be used from previous Test Case
- Features
  - Test data availability defines feature-dependency

Pre-defined start state

Symptom
- Test Cases can only be run in a set order

Cause
- Dependencies between Test Cases

Strategy
- Remove dependencies between Test Cases

Pattern
- Pre-defined start state

- GUI
  - Every Test Case ensures that GUI is in required start state after execution
    - Including after error
- Data
  - Should not be used from previous Test Case
- Features
  - Test data availability defines feature-dependency

CRUD

Symptom
- Test Cases can only be run in a set order

Cause
- Dependencies between Test Cases

Strategy
- Accept certain dependencies

Pattern
- CRUD

- GUI
  - Setup before CRUD pattern
- Data
  - Dependent on previous CRUD items being successful
- Features
  - Dependent on previous CRUD items being successful

Setups

```
Test Case 1
- Setup
    - Login if necessary
    - Close any open projects
- <Functional Workflows for Test Case>

Test Case 2
- Setup
    - Login if necessary
    - Close any open projects
- <Functional Workflows for Test Case>
```

Fresh setup

```
Initial setup
Test Case 1
- <Functional Workflows for Test Case>
- Restore state
    - Close any open projects
❖ Event Handling
    ❖ On error, restore state

Test Case 2
- <Functional Workflows for Test Case>
```

Pre-defined start state

```
Setup
Test Case 1 (Create)
- <Functional Workflows for Test Case>
❖ Event Handling
    ❖ On error, restore state

Test Case 2 (Read)
– <Functional Workflows for Test Case>
```

CRUD

VAV: Verify – Act - Verify

Symptom	• Test Case doesn't discover errors
Cause	• Checks are not carried out
Strategy	• Check data before and after action
Pattern	• VAV

- For example
  - Create new customer
    - Check customer does not exist
    - Create customer
    - Check customer exists
- Advantages
  - Separates checks from actions
  - Easy to use
  - Checks are not forgotten

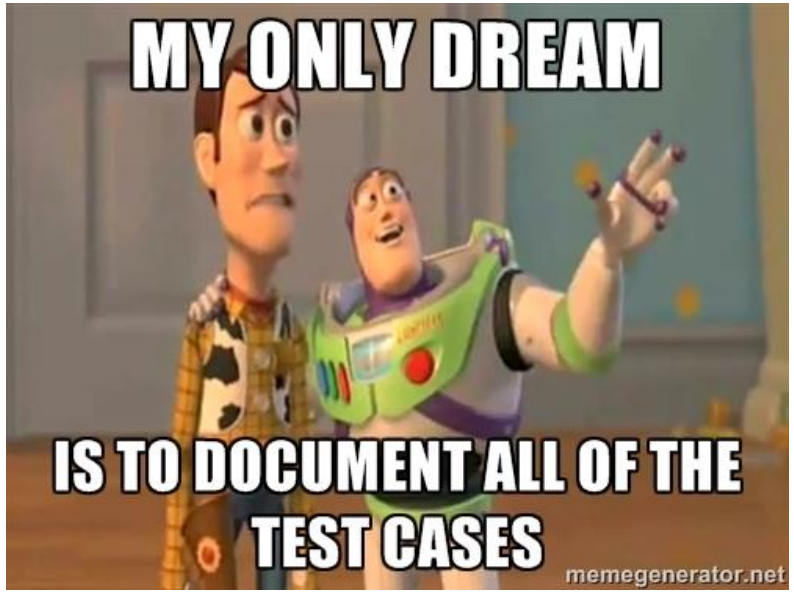
There are more…

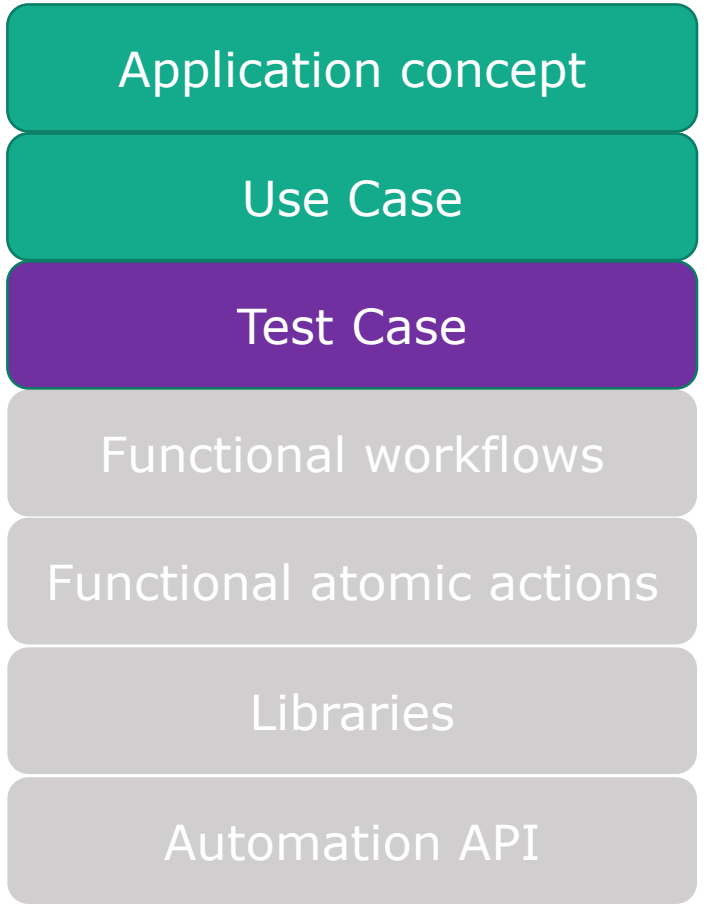
The truth is out there

https://testautomationpatterns.wikispaces.com/

Our tests are lovely. What do they test?

- Linking between Test Cases
  - Which Test Cases exist?
  - Which are automated?

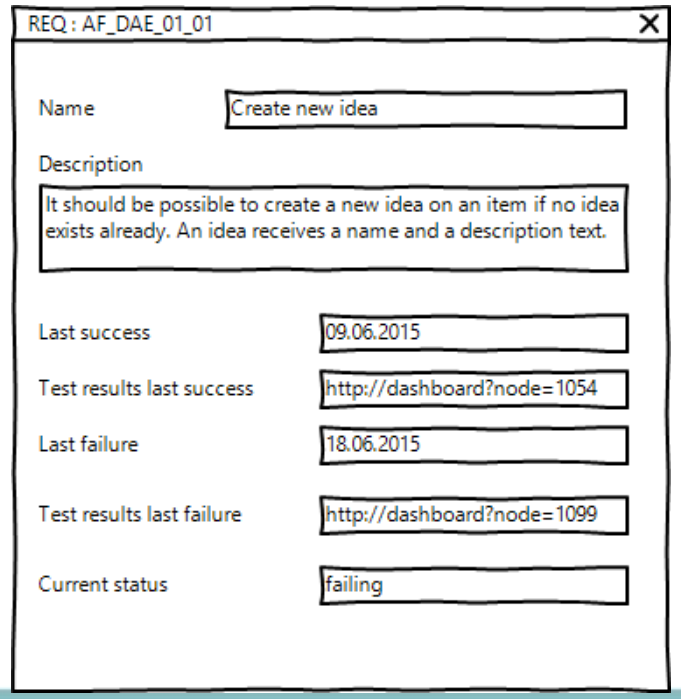




Example 1: ALM-Integration

Requirements in ALM

REQ	
AF_DAE_01_01	Create new idea
AF_DAE_01_02	Edit idea



Generate structure

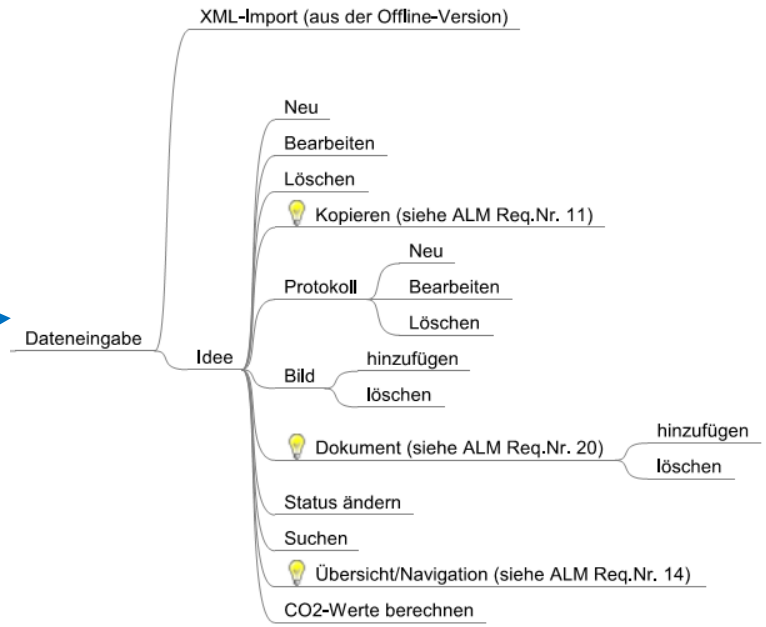
Automatically report

Automation structure

- 10 Testfälle
  - Dateneingabe
    - AF_DAE_01 Ideenblatt
      - AF_DAE_01_01 Neue Idee anlegen
        - TF_DAE_01_01-01: Neue Idee anlegen
      - AF_DAE_01_02 Idee bearbeiten
      - AF_DAE_01_03 Idee kopieren
      - AF_DAE_01_04 Idee löschen
      - AF_DAE_01_05 Idee verschieben
      - AF_DAE_01_06 Bilder
      - AF_DAE_01_07 Fahrzeug auswählen
      - AF_DAE_01_08 Fahrzeugübersicht
      - AF_DAE_01_09 Fahrzeugkennzahlen
      - AF_DAE_01_10 Dokumente
      - AF_DAE_01_11 Status ändern
      - AF_DAE_01_12 Suchen
      - AF_DAE_01_13 Protokollnotizen
    - AF_DAE_02 Übersicht/Navigation

Example 2: Lightweight

Mindmap



User Story → Use Case

User Stories

"Copy idea to another product"

"Create notice for an idea"

Automation structure

- 10 Testfälle
  - Dateneingabe
    - AF_DAE_01 Ideenblatt
      - AF_DAE_01_01 Neue Idee anlegen
        - TF_DAE_01_01-01: Neue Idee anlegen
      - AF_DAE_01_02 Idee bearbeiten
      - AF_DAE_01_03 Idee kopieren
      - AF_DAE_01_04 Idee löschen
      - AF_DAE_01_05 Idee verschieben
      - AF_DAE_01_06 Bilder
      - AF_DAE_01_07 Fahrzeug auswählen
      - AF_DAE_01_08 Fahrzeugübersicht
      - AF_DAE_01_09 Fahrzeugkennzahlen
      - AF_DAE_01_10 Dokumente
      - AF_DAE_01_11 Status ändern
      - AF_DAE_01_12 Suchen
      - AF_DAE_01_13 Protokollnotizen
    - AF_DAE_02 Übersicht/Navigation

Who needs to know?

Detailed information

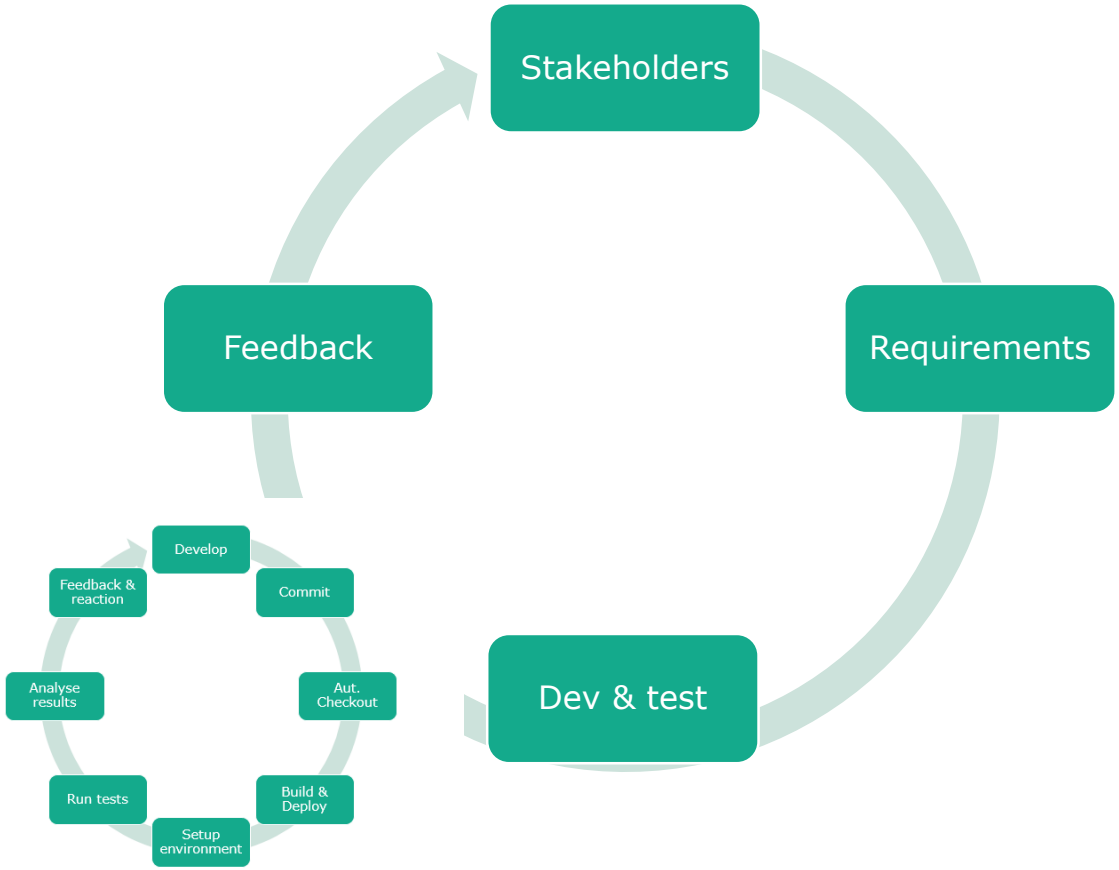
- Test team
- Development team
- Manual test analysis
  - Productive test
    - Error analysis and fixing
  - Incubation test
    - Responsible tester analyses

Trends and results

- Test manager
- Stakeholders
- Reporting
  - Automated reporting
  - Manual status overview
- Long-term graphs

Looking back and new horizons

Business value



- External
  - Easier to release
  - Improved estimation
  - Goodies: manual, videos
- Internal
  - Less time for error analysis
  - Valid across tools/frameworks
- Integration of testing and test automation into whole process

References

Images

- [1] Freedigitalphotos.net, ID 100266055, Iosphere
- [2] Freedigitalphotos.net, ID 100266055, Iosphere
- [3] Freedigitalphotos.net, ID 10010058, marcolm
- [4] Freedigitalphotos.net, ID 10053921, Stuart Miles
- [5] Freedigitalphotos.net, ID 100248430, mapichai