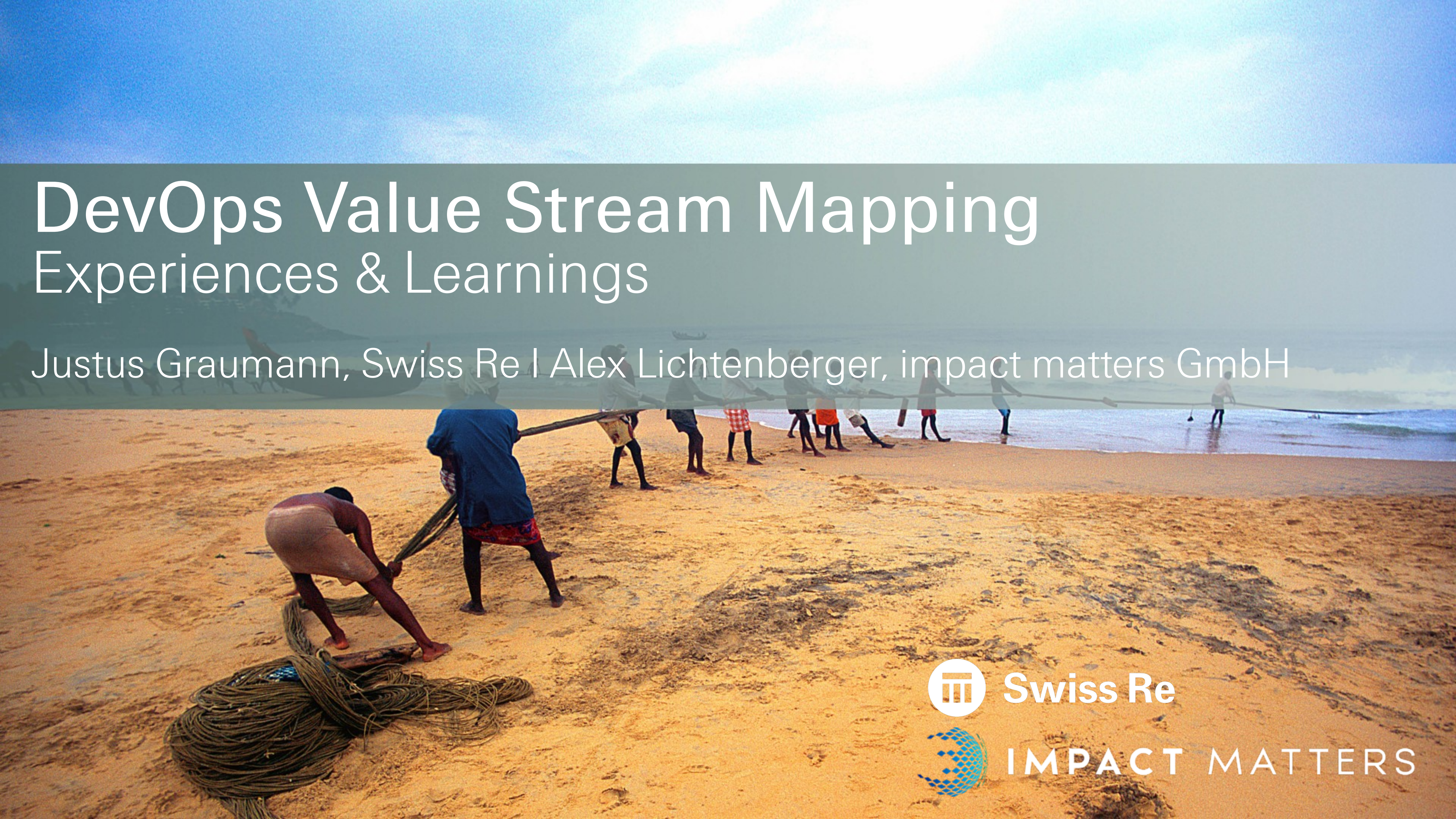


DevOps Value Stream Mapping

Experiences & Learnings

Justus Graumann, Swiss Re I Alex Lichtenberger, impact matters GmbH

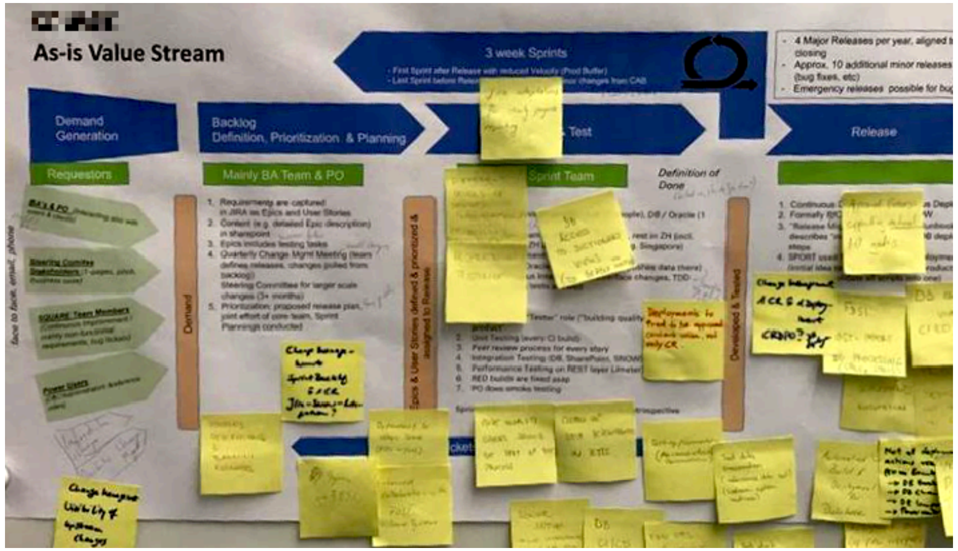
Swiss Re

IMPACT MATTERS

Intro

Topics

- VSM: What it is & approach
- Some reale life examples
- Learnings & benefits



Alex Lichtenberger

- DevOps & Agile Enthusiast
- Founder impact matters GmbH

Justus Graumann

- Application Architect & Engineer at Swiss Re
- Supporting various DevOps initiatives

„Improving daily work is even more important than daily work“

-Gene Kim

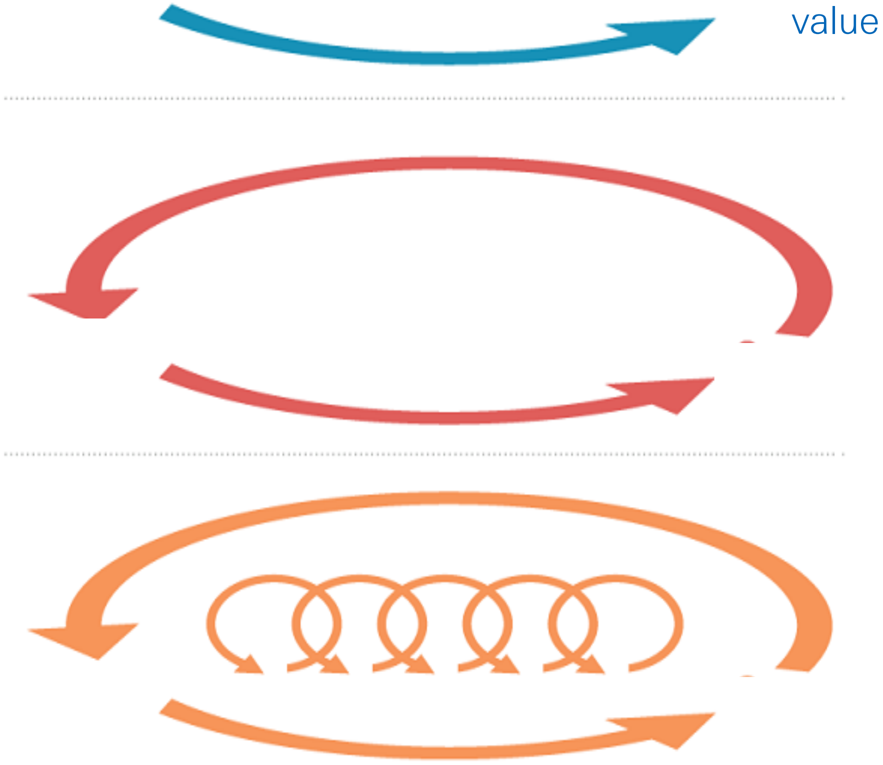
„You can‘t improve what you can‘t see“

DevOps: What we are looking for

Understand & Increase Flow

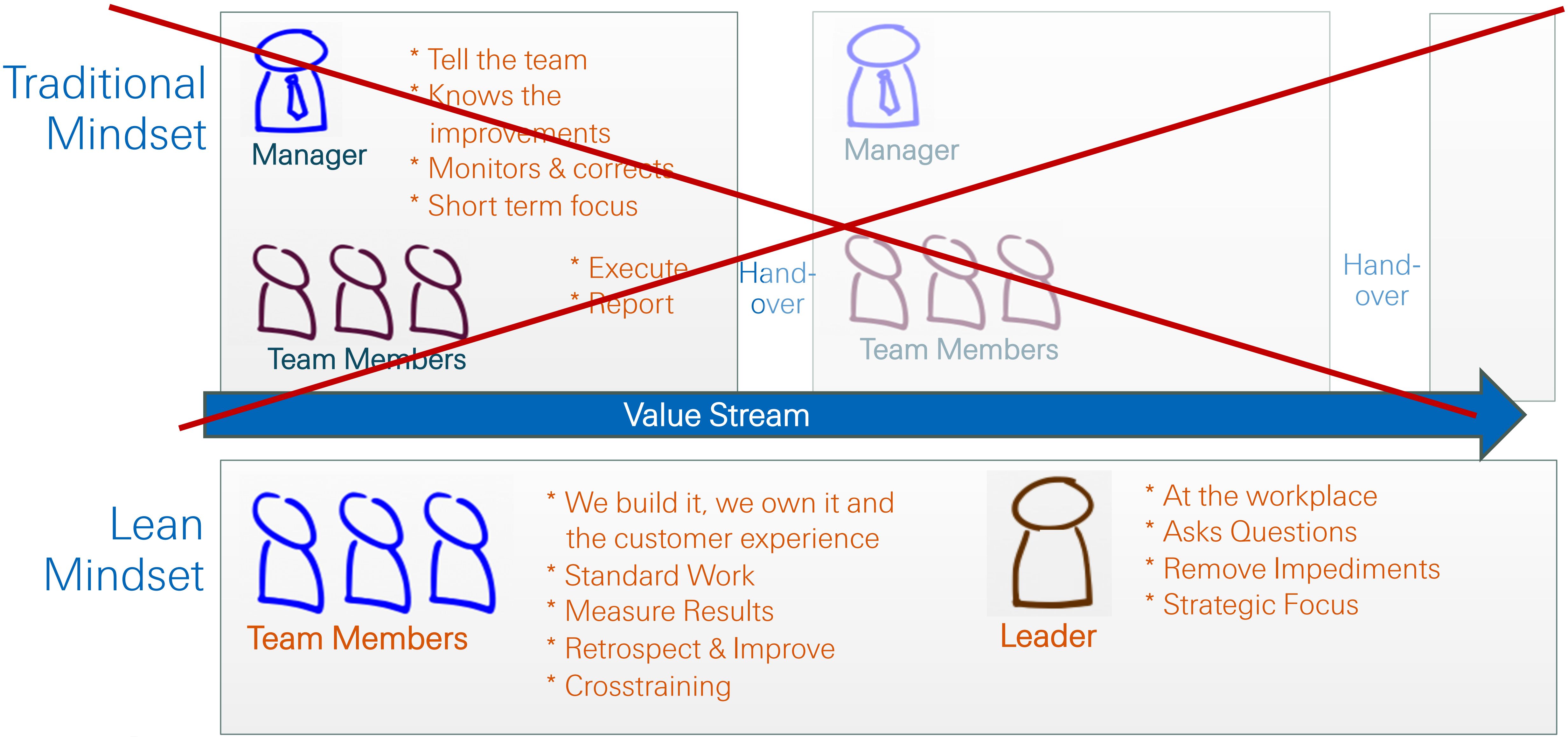
Shorten & Amplify Feedback Loops

Continuous Experimentation & Learning



Various Practices such as CI/CD, Shift Left, IaC and close Collaboration support that.

The Lean Spirit & Mindset



Traditional Mindset

Manager

- Tell the team
- Knows the improvements
- Monitors & corrects
- Short term focus

Team Members

- Execute
- Report

Hand-over

Manager

Team Members

Hand-over

Value Stream

Lean Mindset

Team Members

- We build it, we own it and the customer experience
- Standard Work
- Measure Results
- Retrospect & Improve
- Crosstraining

Leader

- At the workplace
- Asks Questions
- Remove Impediments
- Strategic Focus

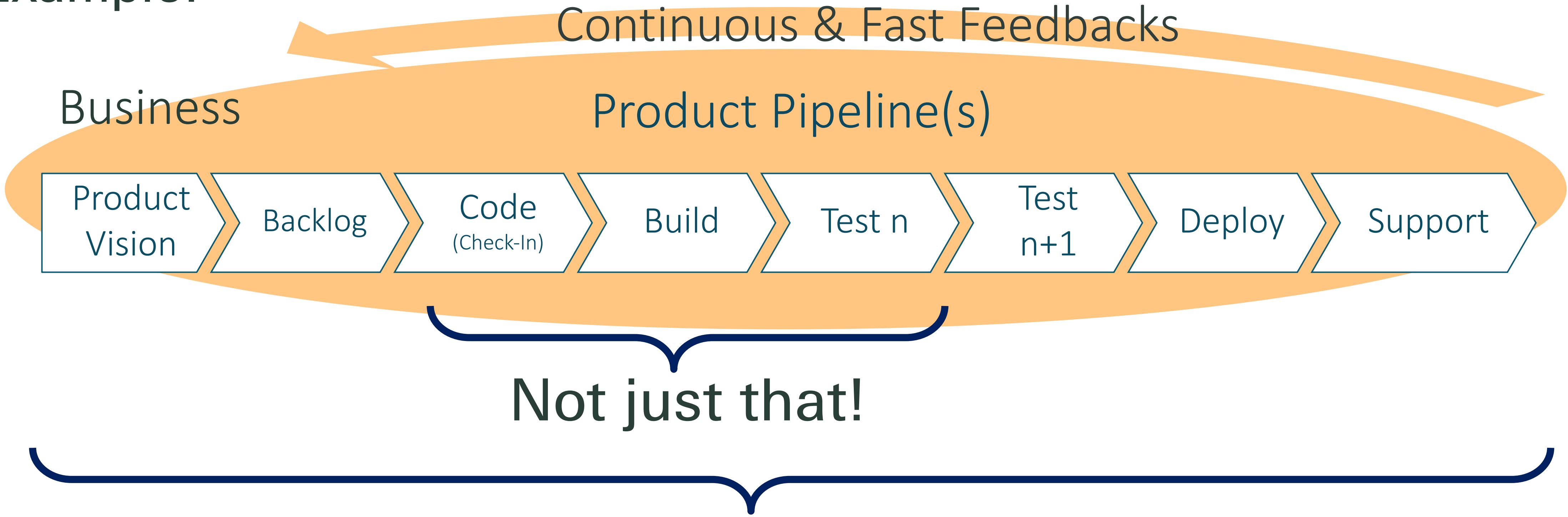
DevOps Value Stream Mapping

Looking at the As-Is Value Stream, end-to-end, identifying improvements potential along the 3 Ways and...

....improve!

Scope: End to End – Inception to Value

Example:

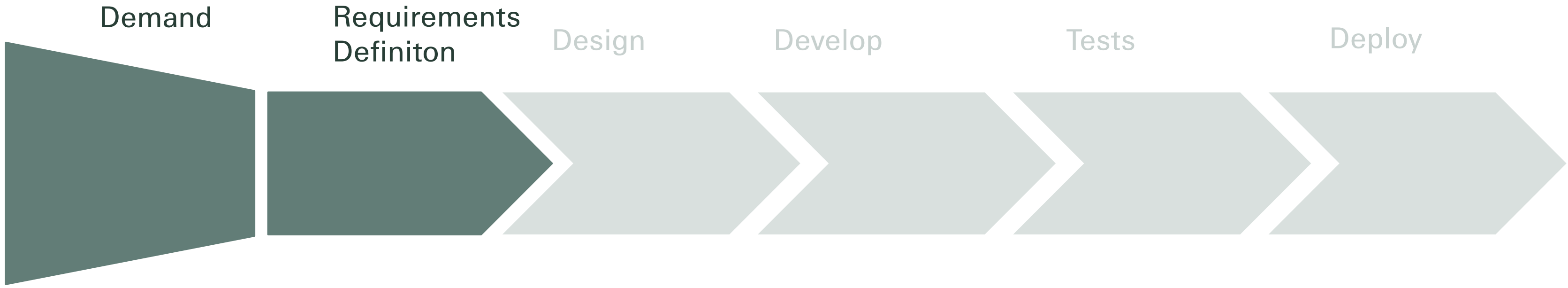


This!

Value Stream Mapping Approach (1/2)

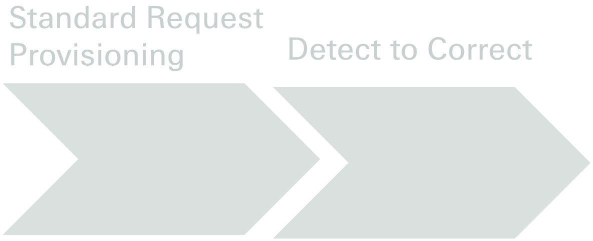
After a couple of iterations we found this approach the most useful:

1. Scoping & preparation As-Is Value Stream Draft (Interviews)
   Gemba – How do people actually work toegther (not how it is documented)



Example questions for one piece in the flow

- Who are typical requestors? (grouping)
- How are requirements gathered & documented?
- How are requirements prioritized?
- Ressources involved?
- Typical lead times?



Value Stream Mapping Approach (2/2)

2. Workshop (usually one day)
   - Verification of «**As Is**» Value Stream
   - Identification of **bottlenecks & improvement potential**
   - **Prioritization** (where do we need to improve first?->Focus)
   - Identification of **improvement** measures
   - Definition of **next steps** & closing
3. **Do it & iterate!**

Typical questions for finding improvement areas?

- Can we eliminate steps or artefacts, which are not adding value?
- Where do we have “idle time”?
- How can we collaborate more closely and avoid unnecessary handovers?
- What is the effort and time needed for each activity?
- Where can be add or shorten feedback loops to shift left?
- Where does it make sense, from a holistic point of view, to optimize?

VSM – What it is not

The electric light did not come from the continuous improvement of the candle.

~ Oren Harari

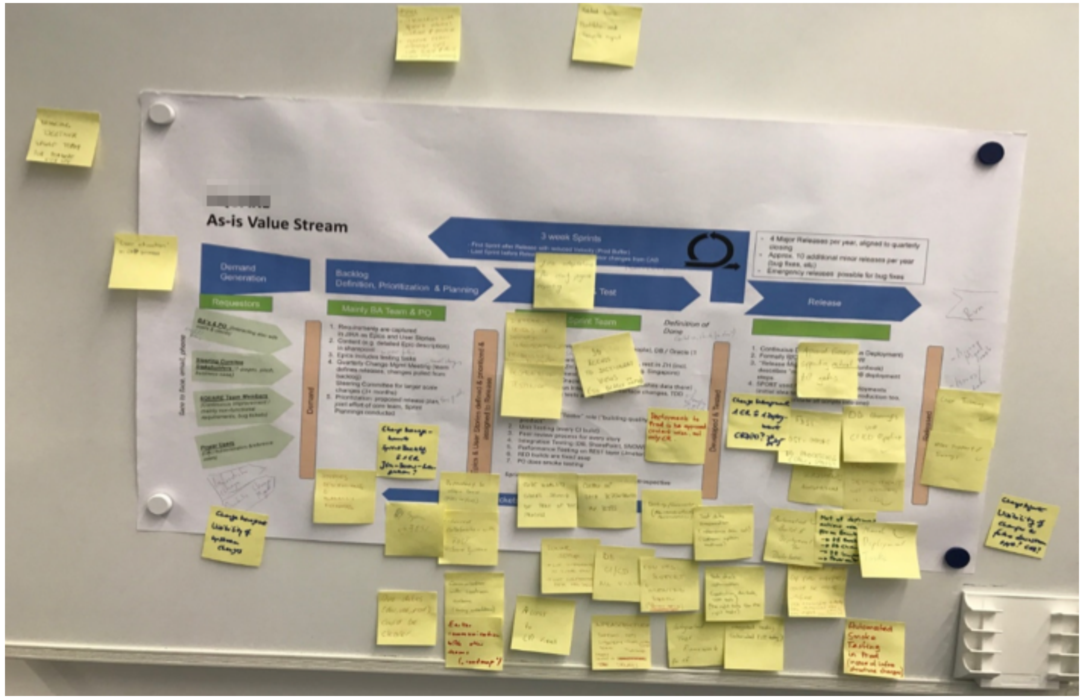
Evolution instead of Revolution!
But you still bring in Innovations!

Example 1 Square

Description of team and scope

Example 1

SQUARE (Software for Qualitative & Quantitative Analysis and Reporting)



- Team is using Agile / Scrum approach

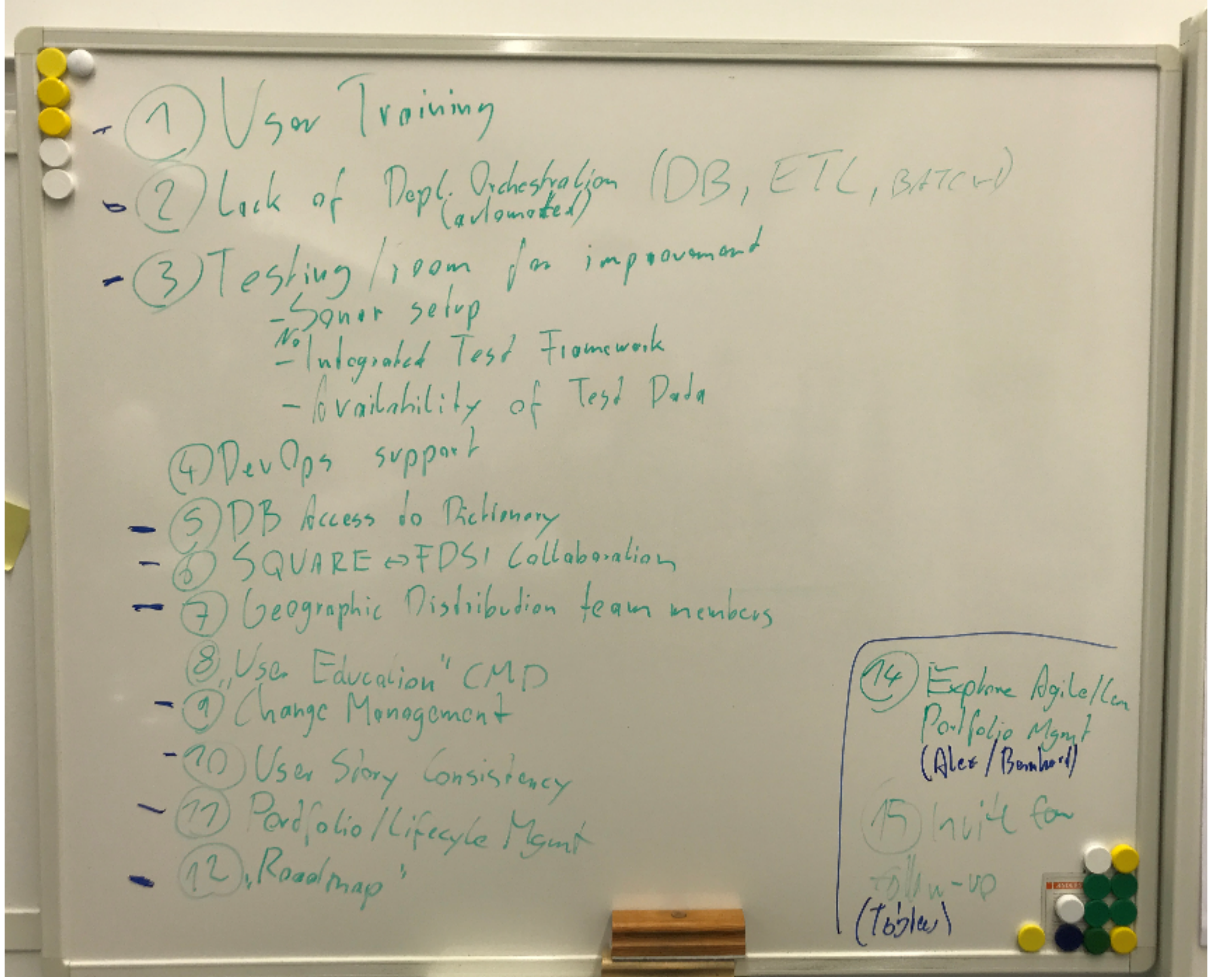
Some Findings out of VSM:

- Collobaration with other teams not optimal
- Lack of deployment orchestration => manual steps
- Missing support in topics like DBaaS
- Missing test data

Some Improvement identified:

- Onboarding on better change management process
- Set up “Trusted Partner Collaboration” Meeting
- Create video tutorials
- and more

Example 1
Improvements areas



1 User Training

2 Lack of Depl. Orchestration (automated) (DB, ETL, BATCH)

3 Testing / room for improvement
- Sonar setup
- No Integrated Test Framework
- Availability of Test Data

4 DevOps support

5 DB Access to Dictionary

6 SQUARE ↔ FDSI Collaboration

7 Geographic Distribution team members

8 „User Education" CMD

9 Change Management

10 User Story Consistency

11 Portfolio/Lifecycle Mgmt

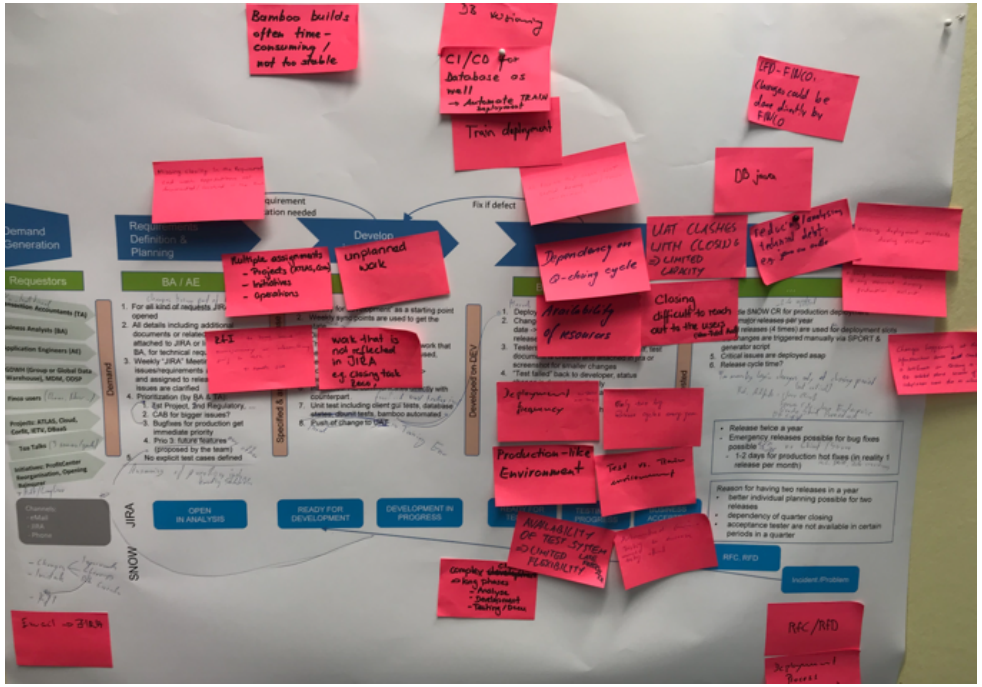
12 „Roadmap"

14 Explore Agile/Lean Portfolio Mgmt (Alex/Bernhard)

15 Invite for follow-up (Tobler)

All involved in one room!

Example 2
Non-Traditional Accounting App



Bamboo builds often time-consuming / not to stable
CI/CD for Database as well → Automate TRAIN deployment
Train deployment
FPD-FINCO. Changes could be done directly by FINCO
DB java
Demand Generation
Requirements Definition & Planning
Develop
Fix if defect
Requestors
BA / AE
Multiple assignments - Projects - Initiatives - Operations
Unplanned work
Dependency on @-change cycle
UAT CLASHES WITH CLOSING → LIMITED CAPACITY
Availability of resources
Closing difficult to reach out to the users
work that is not reflected in JIRA e.g. closing task
Deployment frequency
Production-like Environment
Test vs. Train environment
OPEN IN ANALYSIS
READY FOR DEVELOPMENT
DEVELOPMENT IN PROGRESS
JIRA
SNOW
AVAILABILITY OF TEST SYSTEM → LIMITED FLEXIBILITY
RFC, RFD
Incident Problem
RFC / RFD
Email → JIRA

- Team is using Jira,
  but not a Scrum / Agile approach at all

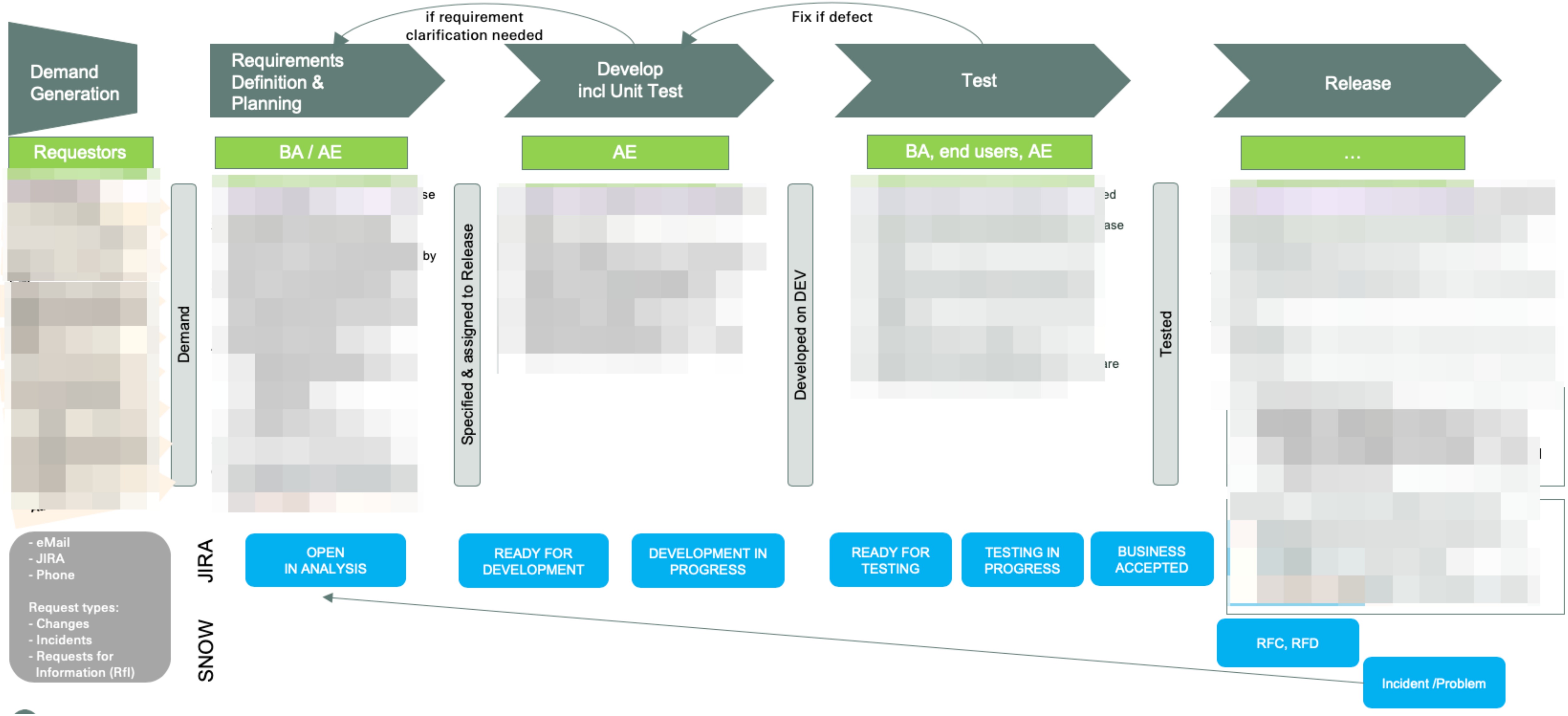
<u>Some Findings out of VSM</u>:

- Change Management process
- work is not reflected in jira (e.g. operational tasks)
- multiple assignments
- availibility of test system

<u>Some Improvement identified</u>:

- Knowledge sharing of new Change Management process
- e2e automation
- enabling early testing approach
- setup production like enviroment in «TRAIN / TEST» environment

As Is Value Stream (Details pixeled out)



if requirement clarification needed

Fix if defect

Demand Generation

Requirements Definition & Planning

Develop incl Unit Test

Test

Release

Requestors

BA / AE

AE

BA, end users, AE

...

Demand

Specified & assigned to Release

Developed on DEV

Tested

- eMail
- JIRA
- Phone

Request types:
- Changes
- Incidents
- Requests for Information (RfI)

JIRA

OPEN IN ANALYSIS

READY FOR DEVELOPMENT

DEVELOPMENT IN PROGRESS

READY FOR TESTING

TESTING IN PROGRESS

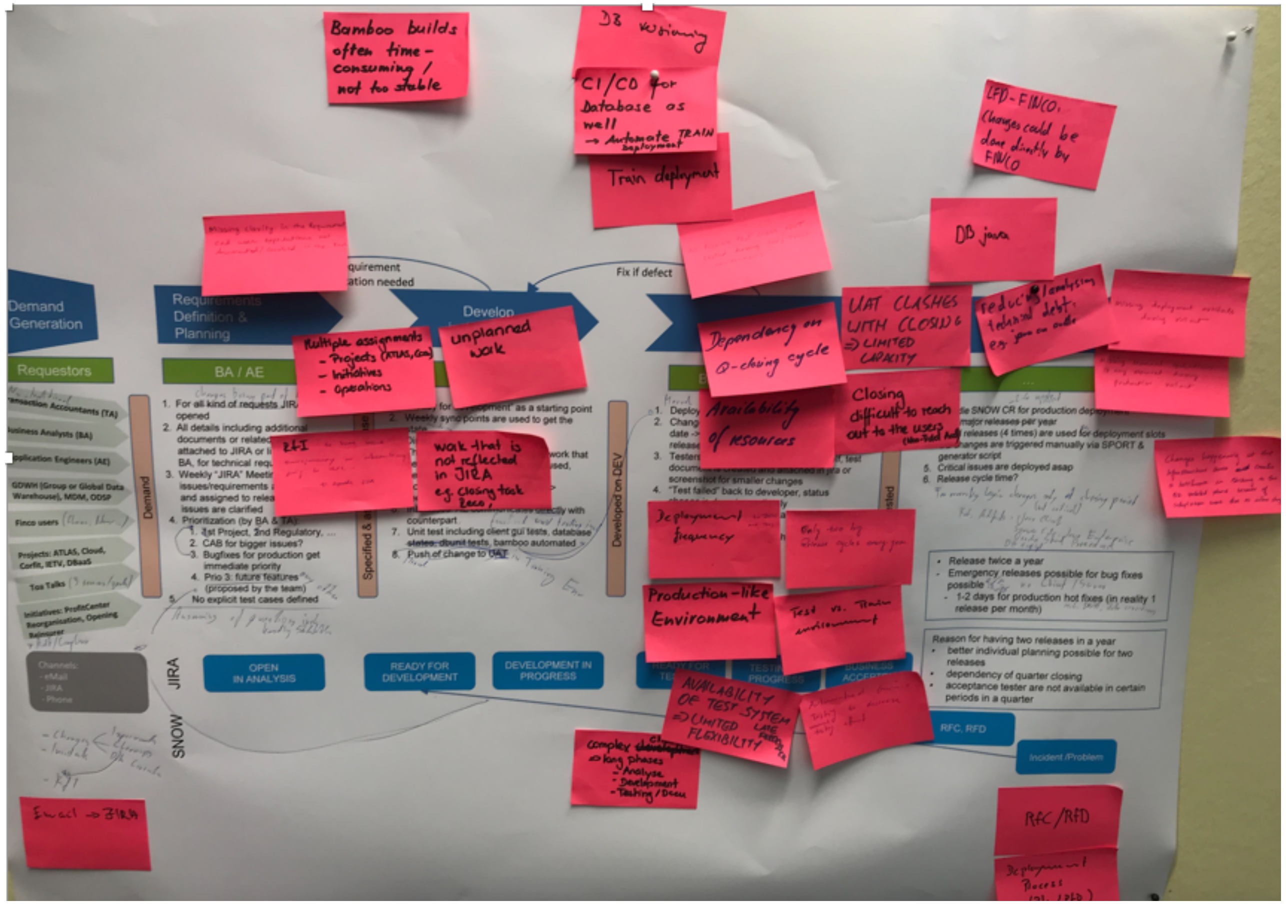
BUSINESS ACCEPTED

SNOW

RFC, RFD

Incident /Problem

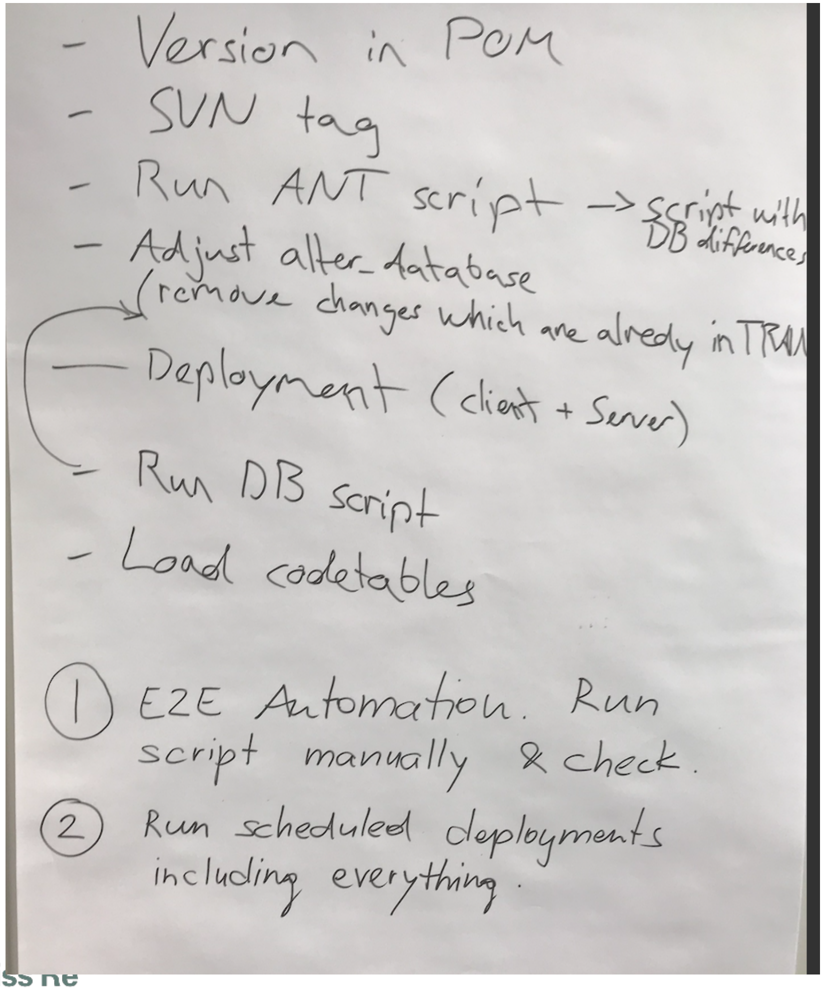
Input from Participants (Improvement Areas)



Bamboo builds often time-consuming / not too stable
DB versioning
CI/CD for Database as well → Automate TRAIN Deployment
Train deployment
LFD - FINCO: Changes could be done directly by FINCO
DB java
Demand Generation
Requirements Definition & Planning
Develop
Fix if defect
Requestors
BA / AE
Multiple assignments - Projects (ATLAS, Cow) - Initiatives - Operations
unplanned work
RtI
work that is not reflected in JIRA e.g. closing task Reco,
Dependency on Q-closing cycle
UAT CLASHES WITH CLOSING ⇒ LIMITED CAPACITY
reducing / analysing technical debt, e.g. java on oracle
Availability of resources
Closing difficult to reach out to the users
Deployment frequency
Production-like Environment
Test vs. Train environment
AVAILABILITY OF TEST SYSTEM ⇒ LIMITED FLEXIBILITY
complex development ⇒ long phases - Analyse - Development - Testing / Docu
Email → JIRA
RfC / RfD
Deployment Process
Transaction Accountants (TA)
Business Analysts (BA)
Application Engineers (AE)
GDWH (Group or Global Data Warehouse), MDM, ODSP
Finco users
Projects: ATLAS, Cloud, Corfit, IETV, DBaaS
Tea Talks
Initiatives: ProfitCenter Reorganisation, Opening Reinsurer
Channels: - eMail - JIRA - Phone
Demand
Specified & ...
Developed on DEV
JIRA
SNOW
OPEN IN ANALYSIS
READY FOR DEVELOPMENT
DEVELOPMENT IN PROGRESS
READY FOR TEST
TESTING IN PROGRESS
BUSINESS ACCEPTANCE
RFC, RFD
Incident /Problem
1. For all kind of requests JIRA opened
2. All details including additional documents or related ... attached to JIRA or ... BA, for technical requ...
3. Weekly "JIRA" Meeting ... issues/requirements ... and assigned to relea... issues are clarified
4. Prioritization (by BA & TA):
1. 1st Project, 2nd Regulatory, ...
2. CAB for bigger issues?
3. Bugfixes for production get immediate priority
4. Prio 3: future features (proposed by the team)
5. No explicit test cases defined
... for development" as a starting point
2. Weekly sync points are used to get the state
... communicates directly with counterpart
7. Unit test including client gui tests, database states, dbunit tests, bamboo automated
8. Push of change to UAT
3. Testers ... test documentation is created and attached in jira or screenshot for smaller changes
4. "Test failed" back to developer, status
... SNOW CR for production deployment
... major releases per year
... releases (4 times) are used for deployment slots
... changes are triggered manually via SPORT & generator script
5. Critical issues are deployed asap
6. Release cycle time?
- Release twice a year
- Emergency releases possible for bug fixes possible
- 1-2 days for production hot fixes (in reality 1 release per month)
Reason for having two releases in a year
- better individual planning possible for two releases
- dependency of quarter closing
- acceptance tester are not available in certain periods in a quarter

Outcomes Group Work

Group A:

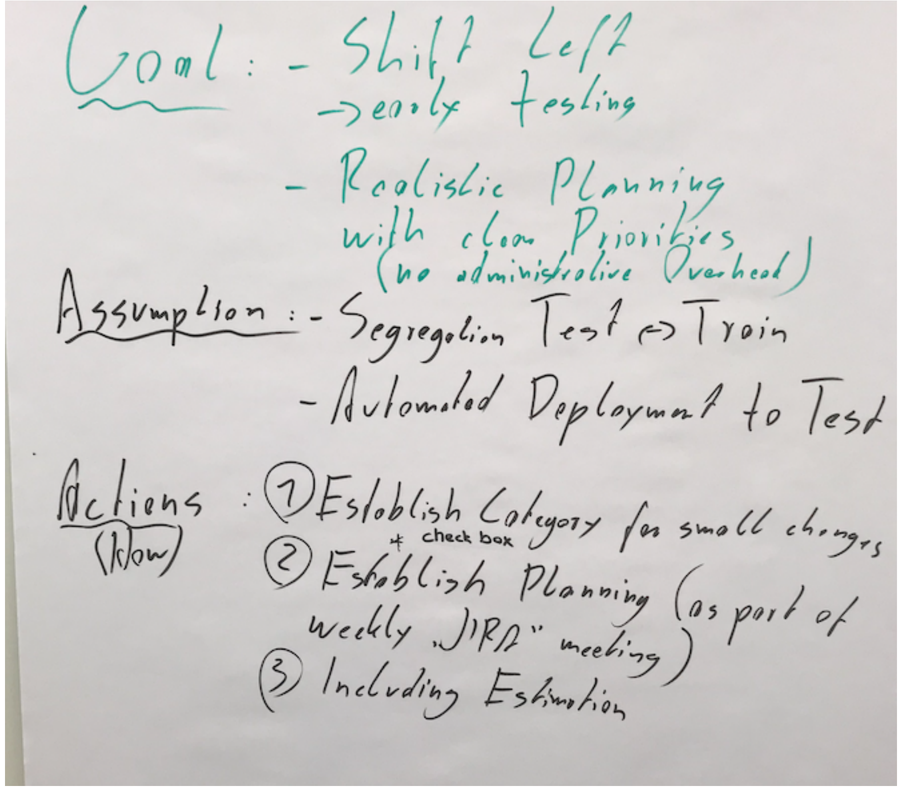


- Version in POM
- SVN tag
- Run ANT script → script with DB differences
- Adjust alter_database (remove changes which are alredy in TRAI
- Deployment (client + Server)
- Run DB script
- Load codetables

① E2E Automation. Run script manually & check.

② Run scheduled deployments including everything.

Group B:



Goal: - Shift Left
→ early testing

- Realistic Planning with clear Priorities (no administrative Overhead)

Assumption: - Segregation Test ↔ Train

- Automated Deployment to Test

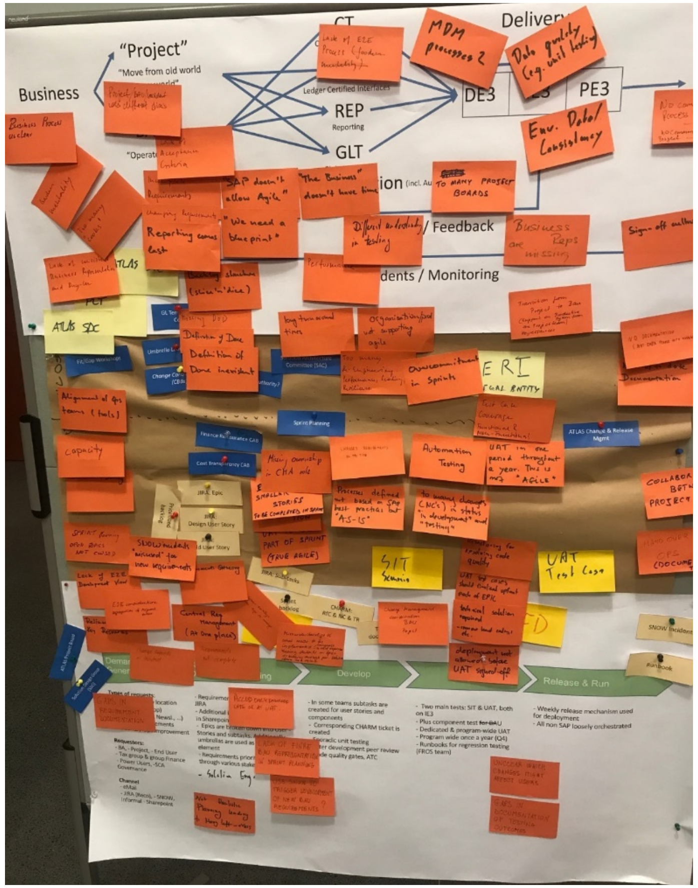
Actions (Now): ① Establish Category for small changes + check box

② Establish Planning (as part of weekly „JIRA" meeting)

③ Including Estimation

Example 2
All involved in a Room!

Example 3
SAP GL Team



- Very complex environment, multiple teams & value streams, hybrid agile.

Some Findings out of VSM:

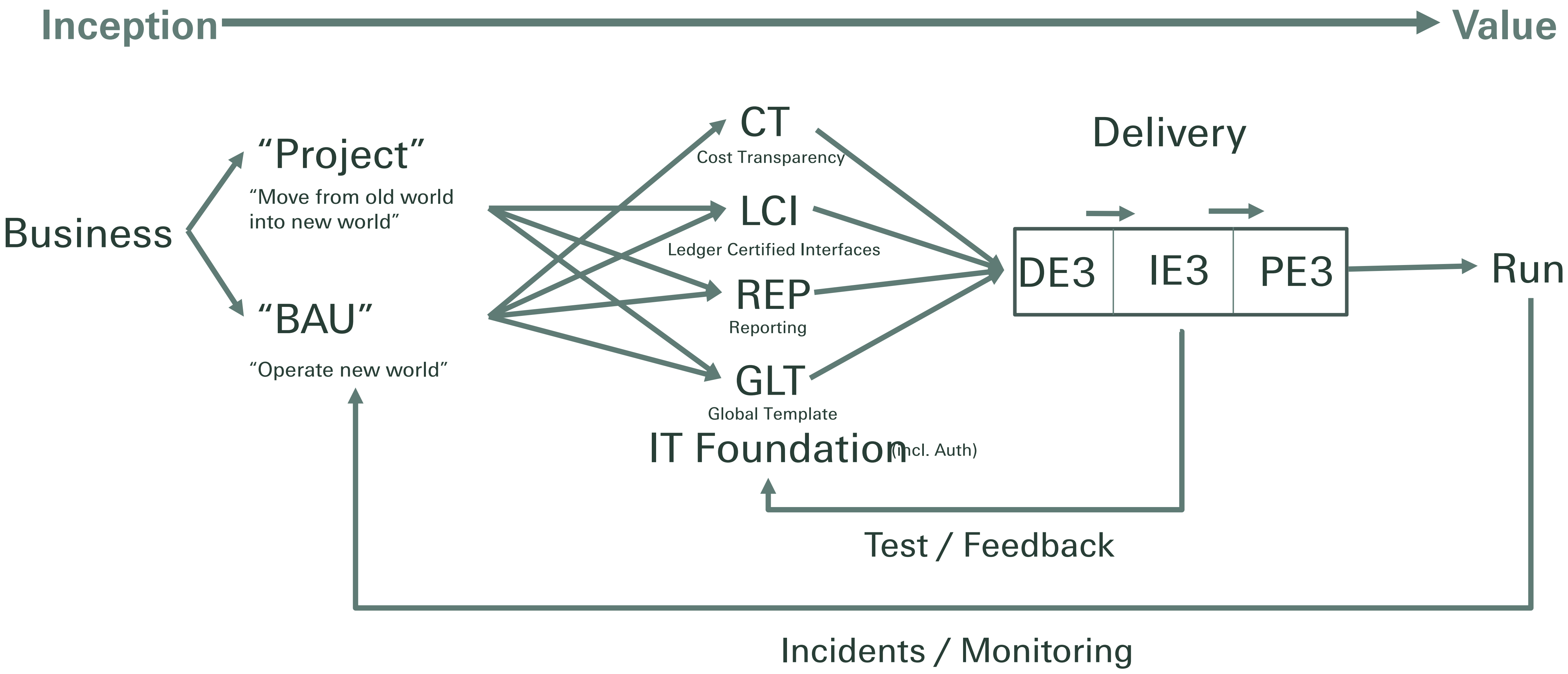
- definition of done / definition of ready is not defined
- e2e accountibility
- Testing automation

Some Improvement identified:

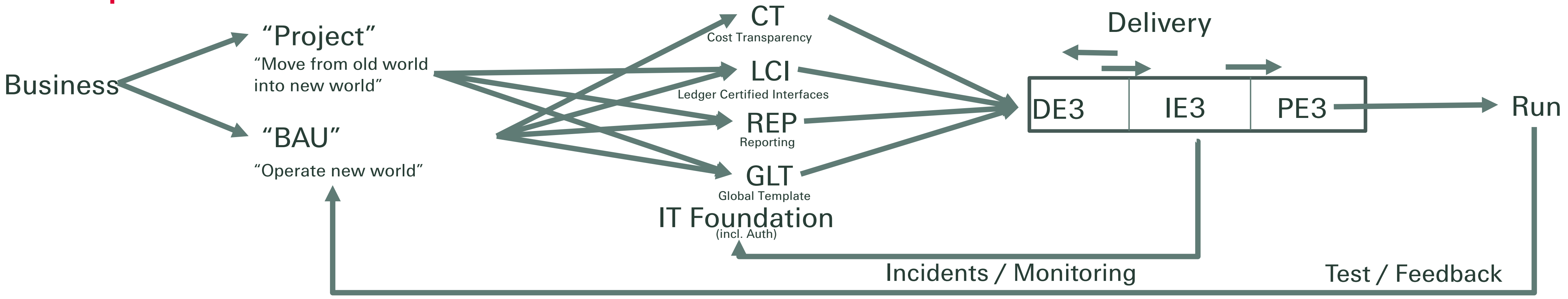
- Testing, Automation, Shift Left
- Collaboration People
- DoD, DoR, Unclear/Changing Requirements

Example 3
Does this work with so many people?

Example 3
Draft As-is Value Stream – High Level



Example 3 - Draft As-is Value Stream – Extended 1



Business

"Project"

"Move from old world into new world"

"BAU"

"Operate new world"

CT

Cost Transparency

LCI

Ledger Certified Interfaces

REP

Reporting

GLT

Global Template

IT Foundation

(incl. Auth)

Delivery

DE3

IE3

PE3

Run

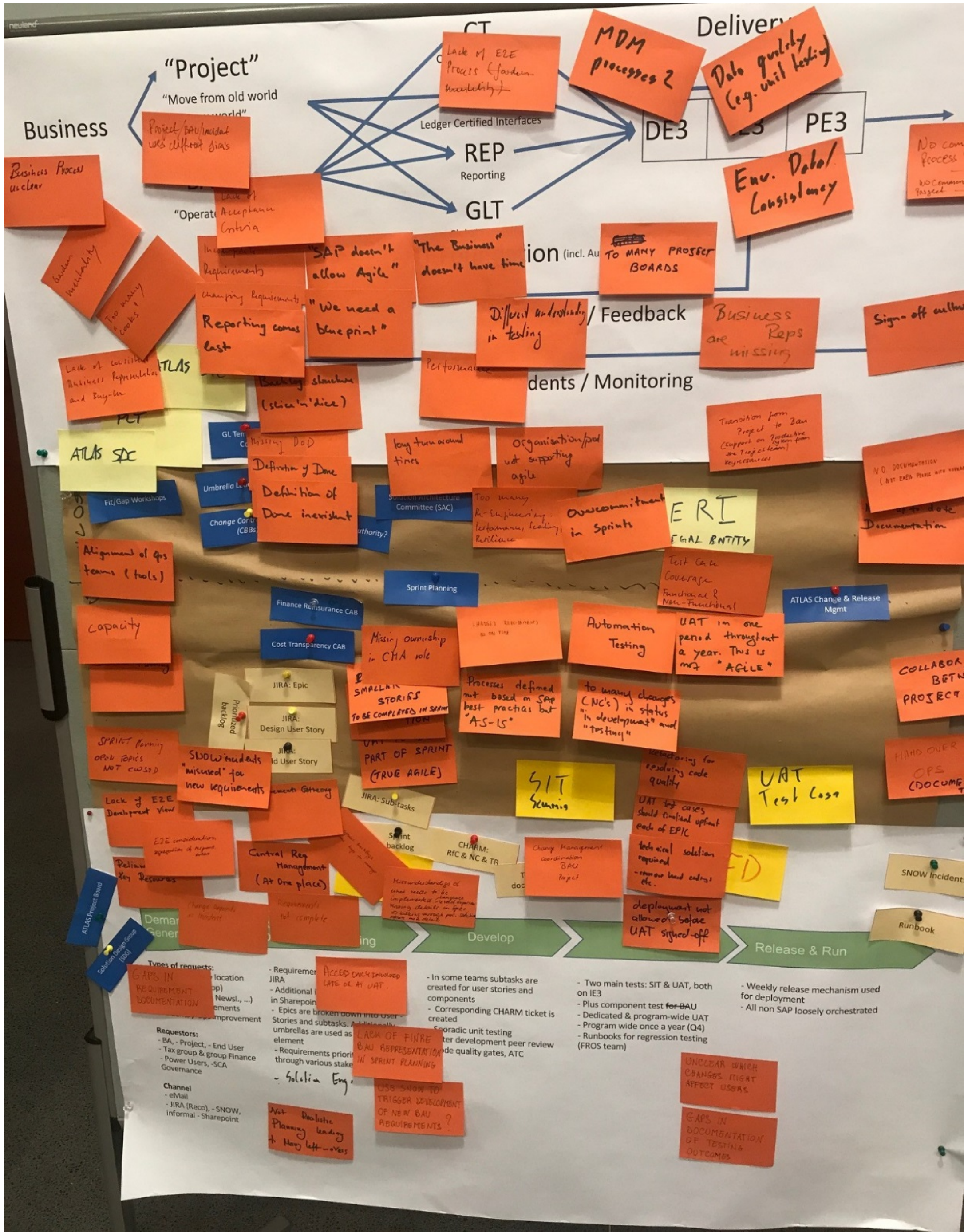
Incidents / Monitoring

Test / Feedback

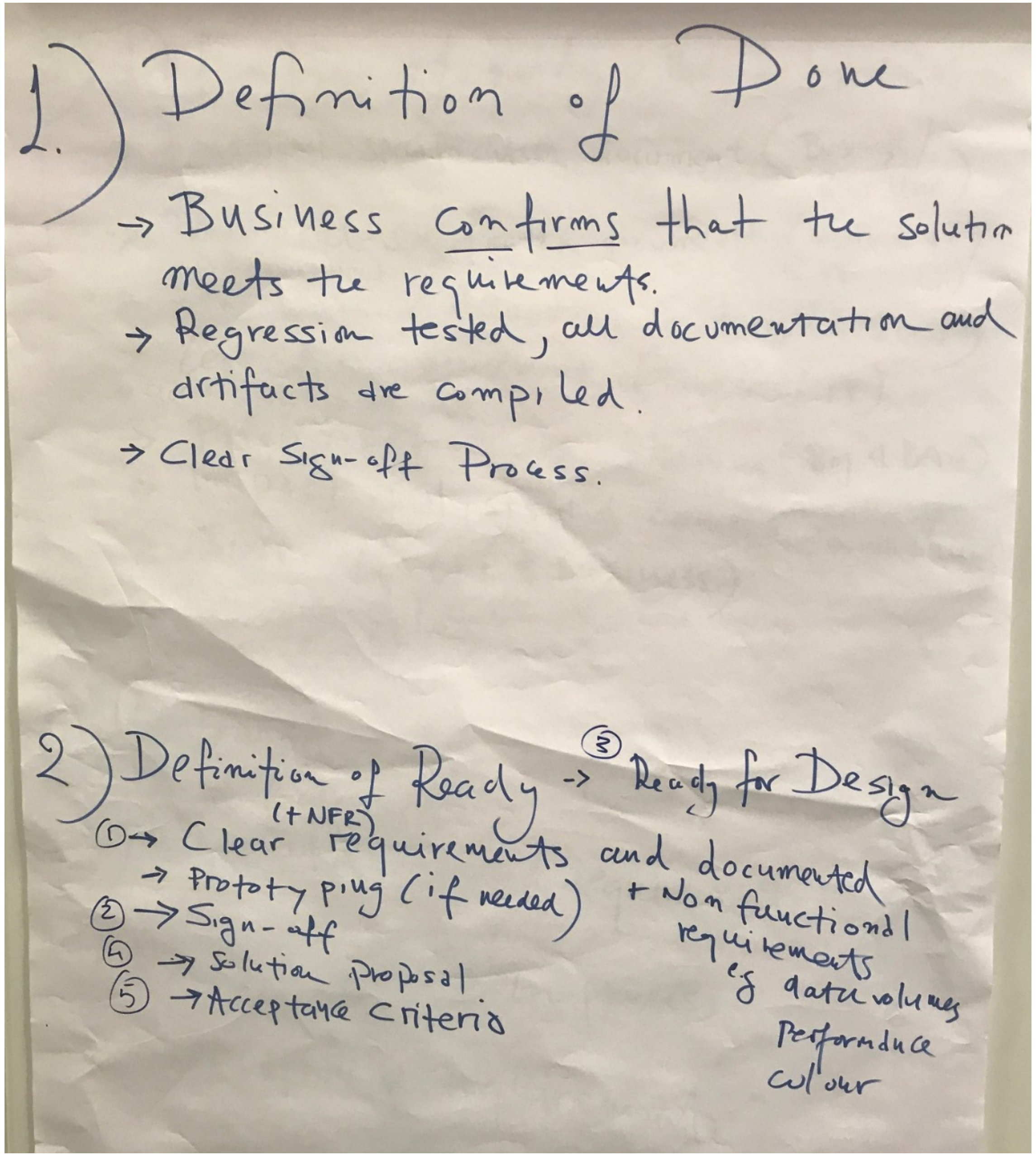
Details (pixeled out)

Demand Generation	Requirements Definition & Planning	Develop	Test	Release & Run
Types of requests: - Onbarding new location (Fit/gap workshop) - Regulatory (FSA Newsl., ...) - End User Requirements - Incidents / Ops improvement **Requestors:** - BA, - Project, - End User - Tax group & group Finance - Power Users, -SCA Governance **Channel** - eMail - JIRA (Reco), - SNOW, informal - Sharepoint	- Requirements are documented in JIRA - Additional information (Design, ...) in Sharepoint, Confluence (Wiki) - Epics are broken down into User - Stories and subtasks. Additionally umbrellas are used as structure element - Requirements prioritization done through various stakeholders - Solution Engineering	- In some teams subtasks are created for user stories and components - Corresponding CHARM ticket is created - Sporadic unit testing - After development peer review - Code quality gates, ATC	- Two main tests: SIT & UAT, both on IE3 - Plus component test for BAU - Dedicated & program-wide UAT - Program wide once a year (Q4) - Runbooks for regression testing (FROS team)	- Weekly release mechanism used for deployment - All non SAP loosely orchestrated

Example 3 -
Identified Improvement Areas
(Red Post It´s)


Business
"Project"
"Move from old world
world"
"Operat
CT
Ledger Certified Interfaces
REP
Reporting
GLT
Delivery
DE3
PE3
ion (incl. Au
/ Feedback
dents / Monitoring
Business Process unclear
Project/BAU/Incident uses different Jira's
Lack of Acceptance Criteria
Lack of E2E Process (fordern traceability)
MDM processes 2
Data quality (e.g. unit testing)
Env. Data/ Consistency
No common Process
Guidance inevitability
Too many "cooks"
Incomplete Requirements
Changing Requirements
Reporting comes last
"SAP doesn't allow Agile"
"We need a blueprint"
"The Business" doesn't have time
TO MANY PROJECT BOARDS
Difficult understanding in testing
Business Reps are missing
Sign-off culture
Lack of consistent Business Representation and Buy-in
ATLAS
PLT
ATLAS SDC
Backlog structure (slice'n'dice)
Performance
Transition from Project to BAU (support on productive system from the Project team) key resources
NO DOCUMENTATION
up to date Documentation
GL Tem
Umbrella
Fit/Gap Workshops
Change Contr (CBBs)
Missing DoD
Definition of Done
Definition of Done inexistent
long turn around times
organisation/prod not supporting agile
Solution Architecture Committee (SAC)
Authority?
Too many Re-Engineering, Performance, Scaling, Resilience
Overcommitment in Sprints
ERI
LEGAL ENTITY
Test Case Coverage Functional & Non-Functional
ATLAS Change & Release Mgmt
Alignment of Ops teams (tools)
Sprint Planning
Finance Reinsurance CAB
Cost Transparency CAB
capacity
CHANGES REQUIREMENTS ALL THE TIME
Automation Testing
UAT in one period throughout a year. This is not "AGILE"
COLLABOR BETW PROJECT
Missing ownership in CMA role
JIRA: Epic
Prioritized backlog
JIRA: Design User Story
JIRA: Build User Story
SMALLER STORIES TO BE COMPLETED IN SPRINT
PART OF SPRINT (TRUE AGILE)
Processes defined not based on SAP best practices but "AS-IS"
to many changes (NC's) in status "in development" and "testing"
SPRINT Planning OPEN TOPICS NOT CLOSED
SNOW incidents "misused" for new requirements
Requirements Gathering
JIRA: Sub-tasks
SIT Scanning
Refactoring for resolving code quality
UAT test cases should finalized upfront each of EPIC
UAT Test Case
HAND OVER OPS (DOCUMENTATION
Lack of E2E Development View
E2E consideration
Central Req Management (At One place)
Sprint backlog
CHARM: RfC & NC & TR
Change Management coordination BAU Project
technical solution required
deployment not allowed before UAT signed-off
SNOW Incident
Runbook
Reliance on Key Resources
Change Requests as Incident
Requirements not complete
Misunderstandings of what needs to be implemented
ATLAS Project Board
Solution Design Group (SDG)
Demand Generation
ing
Develop
Release & Run
Types of requests:
location
Newsl., ...)
ements
improvement
Requestors:
- BA, - Project, - End User
- Tax group & group Finance
- Power Users, -SCA
Governance
Channel
- eMail
- JIRA (Reco), - SNOW,
informal - Sharepoint
GAPS IN REQUIREMENT DOCUMENTATION
- Requiremen
JIRA
- Additional
in Sharepoin
- Epics are broken down into User Stories and subtasks.
umbrellas are used as
element
- Requirements priorit
through various stake
- Solution Eng.
ACCESS ONLY INVOLVED LATE OR AT UAT.
LACK OF FINRE BAU REPRESENTATION IN SPRINT PLANNING
USE SNOW TO TRIGGER DEVELOPMENT OF NEW BAU REQUIREMENTS?
Not Realistic Planning leading to Many left-overs
- In some teams subtasks are created for user stories and components
- Corresponding CHARM ticket is created
- Sporadic unit testing
- After development peer review
- Code quality gates, ATC
- Two main tests: SIT & UAT, both on IE3
- Plus component test ~~for BAU~~
- Dedicated & program-wide UAT
- Program wide once a year (Q4)
- Runbooks for regression testing (FROS team)
- Weekly release mechanism used for deployment
- All non SAP loosely orchestrated
UNCLEAR WHICH CHANGES MIGHT AFFECT USERS
GAPS IN DOCUMENTATION OF TESTING OUTCOMES

Example 3 Results Group 1 (1/2)



1.) Definition of Done

→ Business confirms that the solution meets the requirements.

→ Regression tested, all documentation and artifacts are compiled.

→ Clear Sign-off Process.

2) Definition of Ready (+NFR) → ③ Ready for Design

① → Clear requirements and documented + Non functional requirements e.g data volumes Performance colour

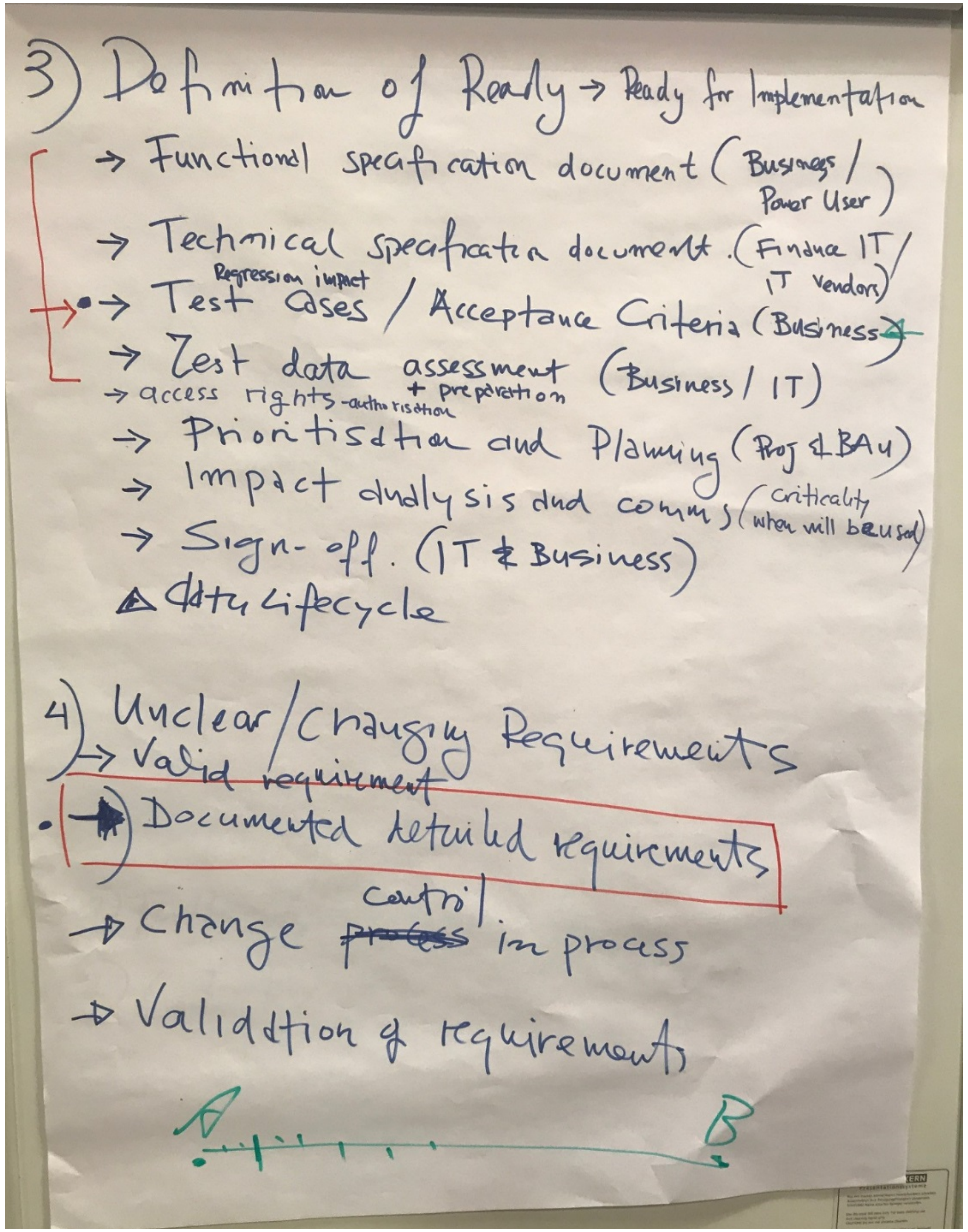
→ Prototyping (if needed)

② → Sign-off

④ → Solution proposal

⑤ → Acceptance criteria

Example 3
Results
Group 1 (2/2)



3) Definition of Ready → Ready for Implementation

→ Functional specification document (Business / Power User)

→ Technical specification document. (Finance IT / IT Vendors)

• → Test Regression impact Cases / Acceptance Criteria (Business)

→ Test data assessment + preparation (Business / IT)

→ access rights-authorisation

→ Prioritisation and Planning (Proj & BAU)

→ Impact analysis and comms (criticality when will be used)

→ Sign-off. (IT & Business)

▲ data Lifecycle

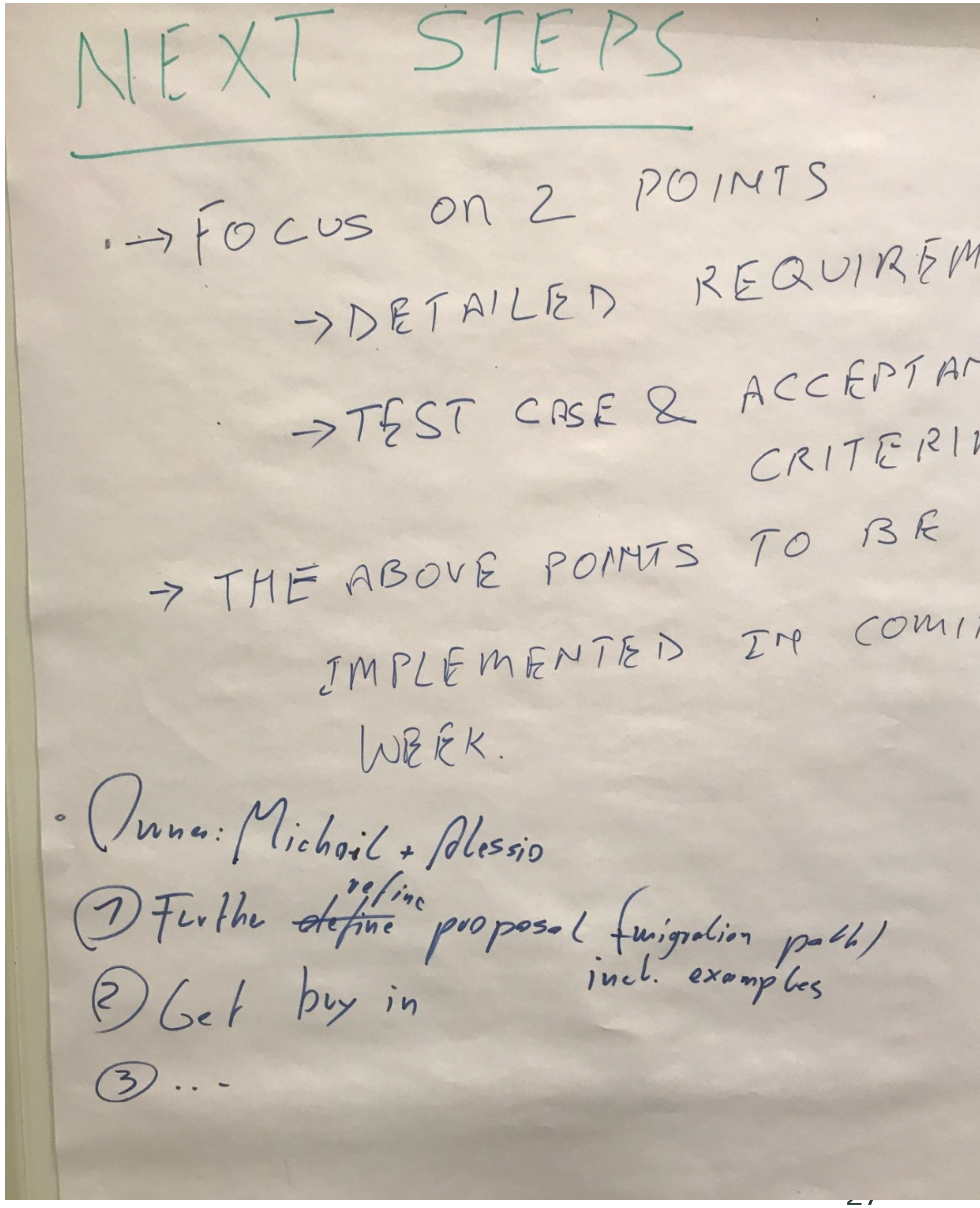
4) Unclear/Changing Requirements

→ Valid requirement

• → Documented detailed requirements

→ Change ~~process~~ control in process

→ Validation of requirements



NEXT STEPS

• → FOCUS on 2 POINTS

→ DETAILED REQUIREM

→ TEST CASE & ACCEPTAN CRITERI

→ THE ABOVE POINTS TO BE IMPLEMENTED IN COMI WEEK.

• Owner: Michail + Alessio

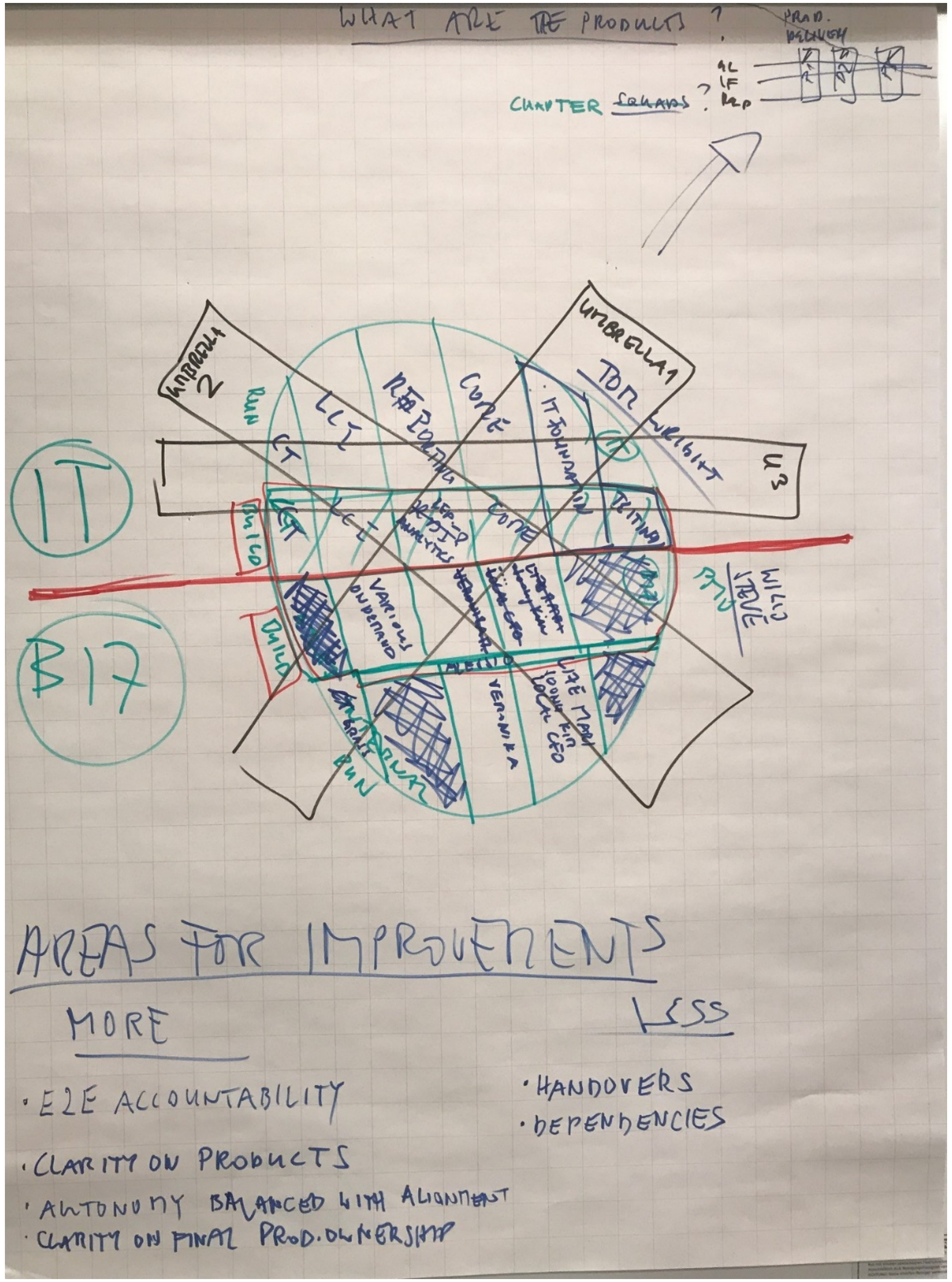
① Further ~~define~~ refine proposal (migration path) incl. examples

② Get buy in

③ ...

Swiss Re

Example 3 Results Group 2



WHAT ARE THE PRODUCTS ?

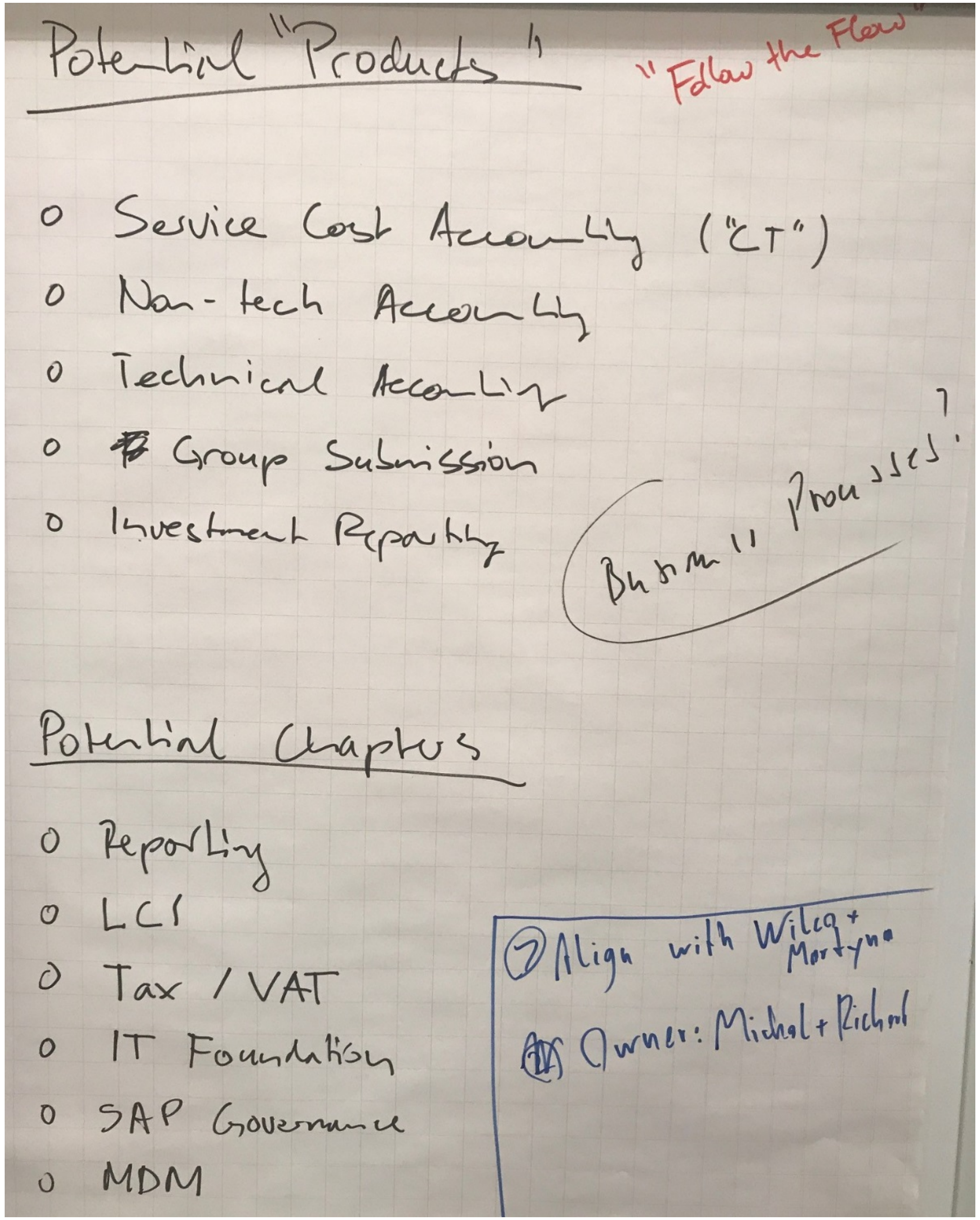
AREAS FOR IMPROVEMENTS

MORE

- E2E ACCOUNTABILITY
- CLARITY ON PRODUCTS
- AUTONOMY BALANCED WITH ALIGNMENT
- CLARITY ON FINAL PROD. OWNERSHIP

LESS

- HANDOVERS
- DEPENDENCIES



Potential "Products"

"Follow the Flow"

- Service Cost Accounting ("CT")
- Non-tech Accounting
- Technical Accounting
- ~~#~~ Group Submission
- Investment Reporting

Business "Processes"?

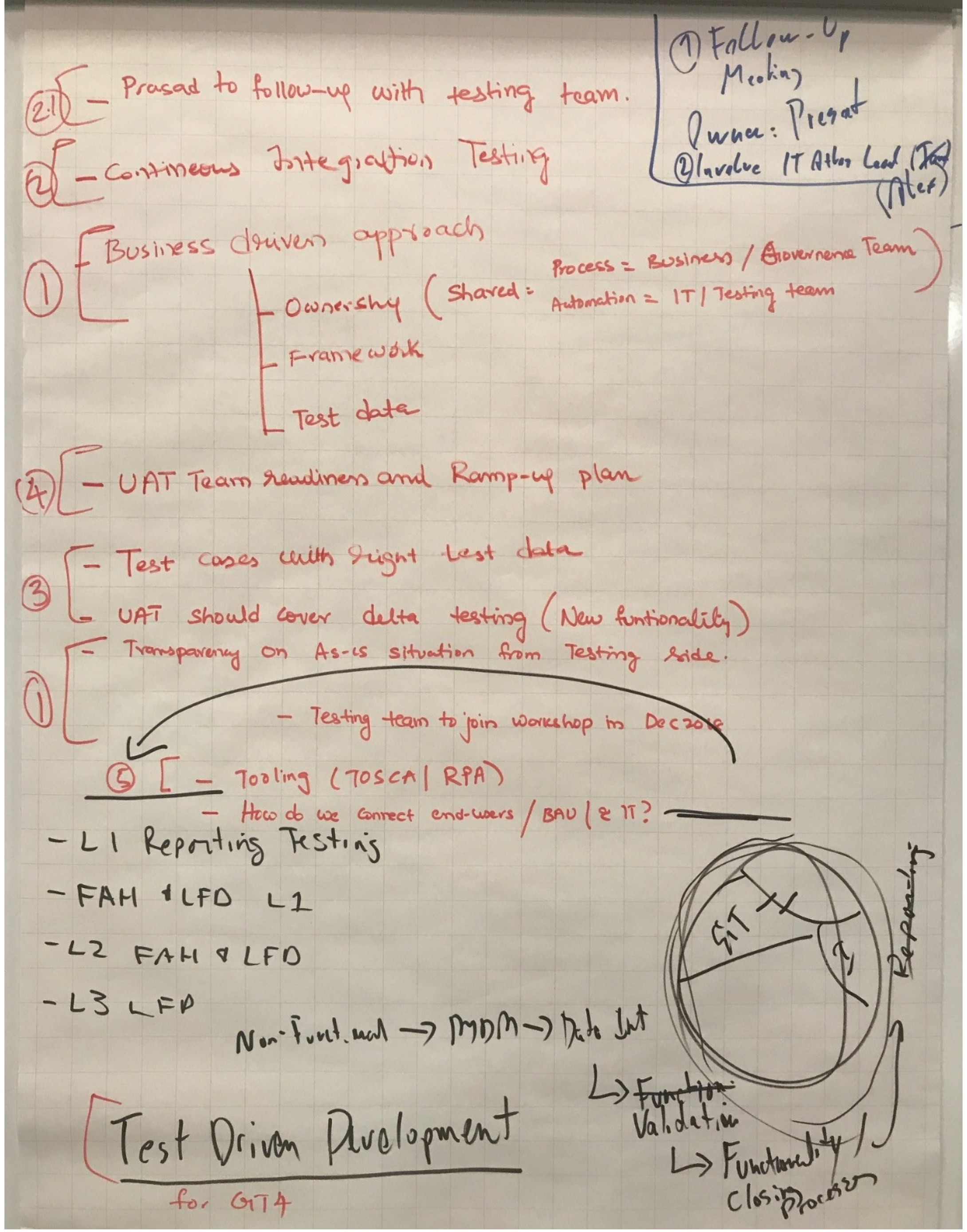
Potential Chapters

- Reporting
- LCI
- Tax / VAT
- IT Foundation
- SAP Governance
- MDM

? Align with Wilco + Martyna

Owner: Michael + Richard

Example 3
Results
Group 3



(2.1) [– Prasad to follow-up with testing team.

(2) [– Continuous Integration Testing

(1) [Business driven approach
- Ownership (Shared : Process = Business / Governance Team, Automation = IT / Testing team)
- Framework
- Test data

(4) [– UAT Team readiness and Ramp-up plan

(3) [– Test cases with right test data
– UAT should cover delta testing (New funtionality)

(1) [– Transparency on As-is situation from Testing side.
– Testing team to join workshop in Dec 2018

(5) [– Tooling (TOSCA | RPA)
– How do we connect end-users / BAU | & IT?

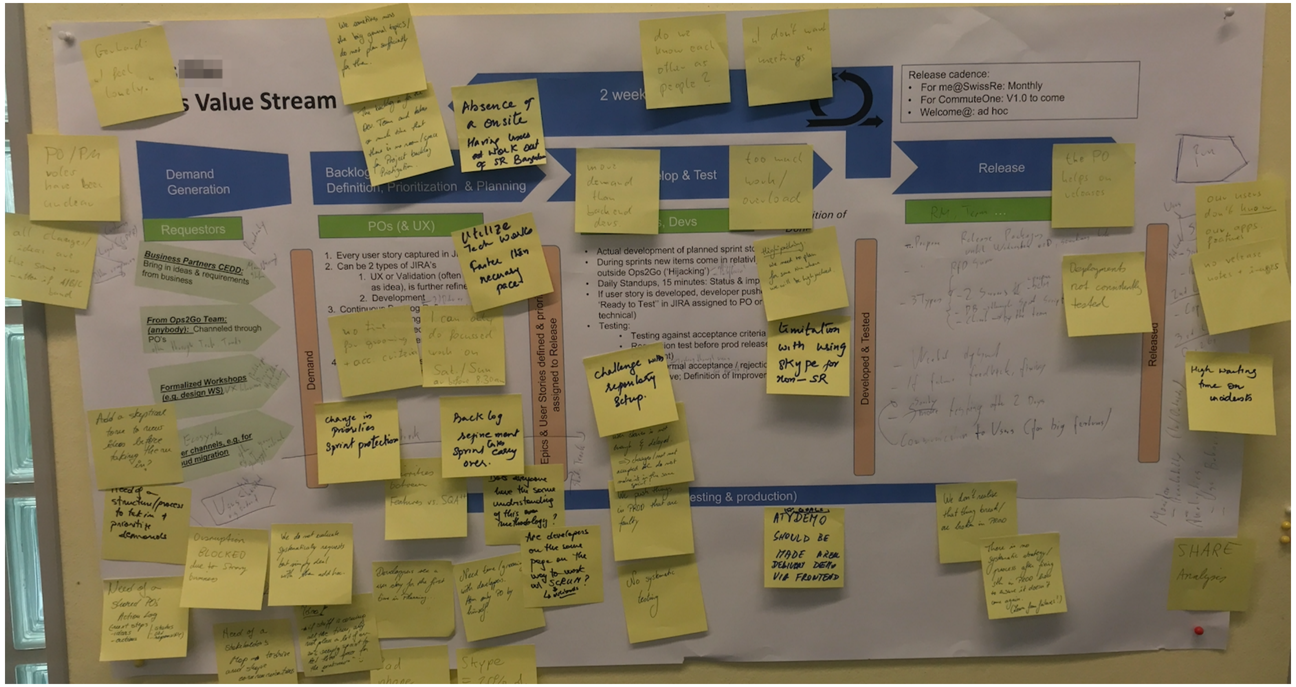
(1) Follow-Up Meeting
Owner: Prasad
(2) Involve IT Athos Lead (~~[illegible]~~ (Atef)

- L1 Reporting Testing
- FAH + LFD L1
- L2 FAH & LFD
- L3 LFD

Non-Functional → MDM → Data Int
↳ ~~Function~~ Validation
↳ Functionality / Closing Processes

[Test Driven Development for GIT4

Example 4
Mobile App Developement Team



Value Stream
Demand Generation
Requestors
Backlog Definition, Prioritization & Planning
POs (& UX)
2 week
Develop & Test
Devs
Release
Release cadence:
For me@SwissRe: Monthly
For CommuteOne: V1.0 to come
Welcome@: ad hoc
Demand
Epics & User Stories defined & prioritized, assigned to Release
Developed & Tested
Released
Testing & production)
Absence of a onsite
Utilize Tech works faster than necessary pace
Back log refinement Sprint carry over
Change in priorities Sprint protection
Challenge with regularly Setup
Limitation with using Skype for Non-SR
ATYDEMO SHOULD BE MADE AVAILABLE VIA FRONTEND
High waiting time on incidents
SHARE Analytics

- Innovation Team
- Scrum, UX, Design Thinking
- High degree of autonomy & decoupling

<u>Some Findings out of VSM</u>:

- No time for incidents
- Quality gates before Release → late feedback
- Inputs from many stakeholders / not clear -> difficult prioritizations

<u>Some Improvement identified</u>:

- **Reserve time for unplanned stuff (ScrumBan)**
- **Shift left (quality gates before release)**
- Create Stakeholder Map

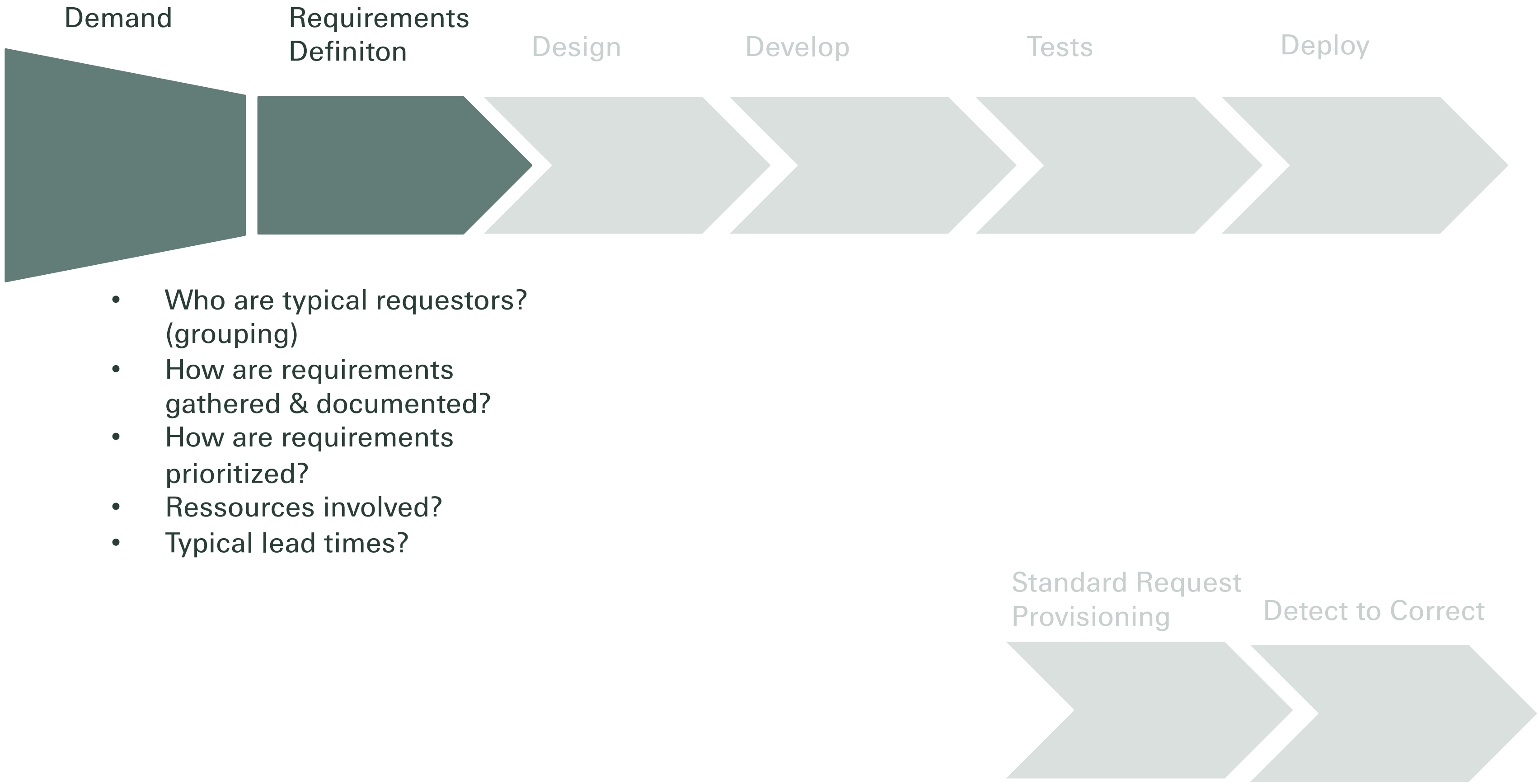
Conclusions & Learnings

- Just making the value Stream visible can already solve many issues !
- As improvements are defined by the team, they usually support it, but it also needs commitment (intrinsic purpose)
- Implement the improvement the agile way -> prioritized backlog, iterations with feedback
- VSM should´t be a one-off exercise, but a continuous effort
- Scoping the Value Stream right at the beginning is key, but sometimes difficult
- Involving business representatives is crucial, but not easy, there is still a business / it silo mindset in place

Appendix
Example questions for identifying As-Is VAlue Stream

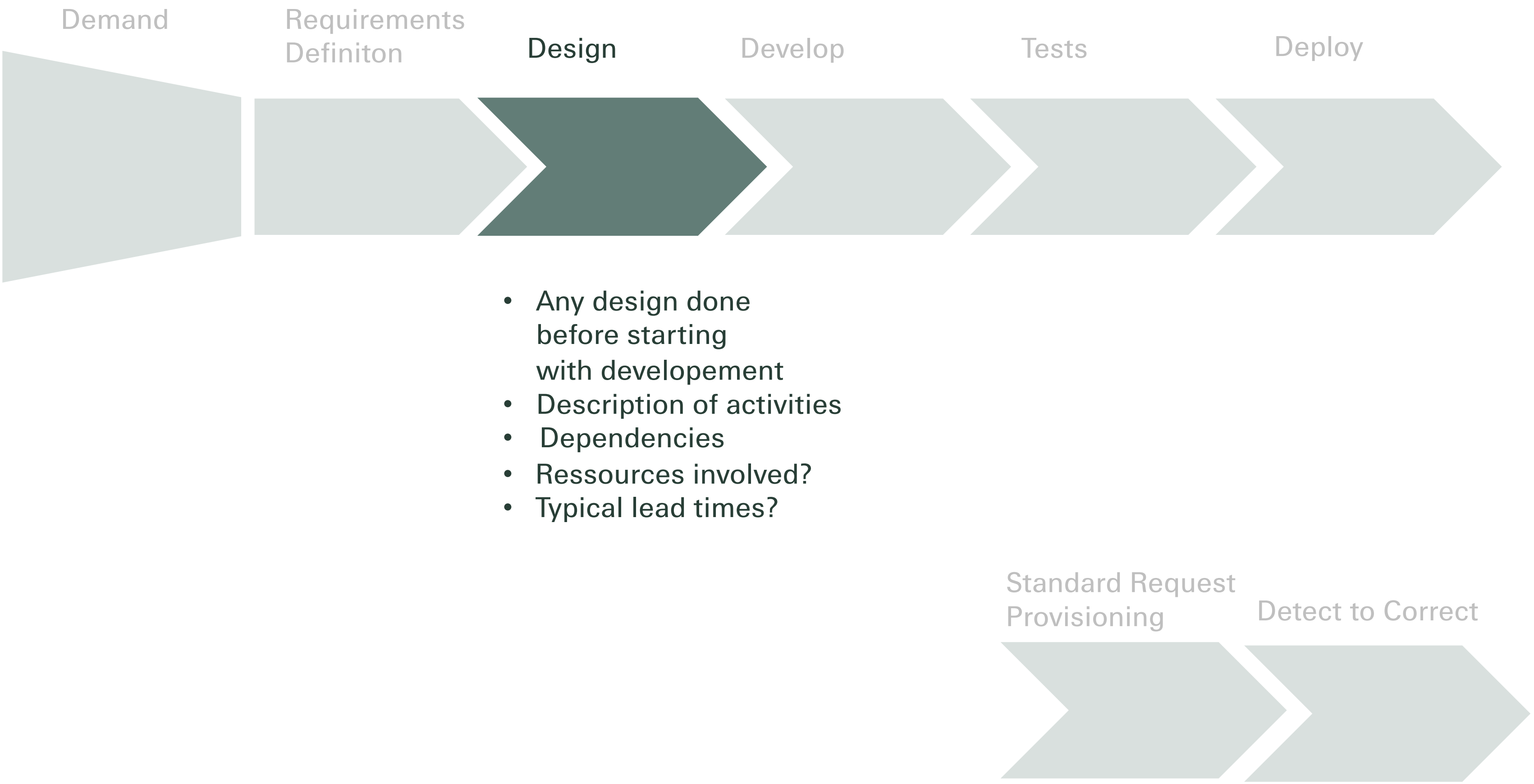
Example Value Stream Questions

This was for a classic waterfall setup



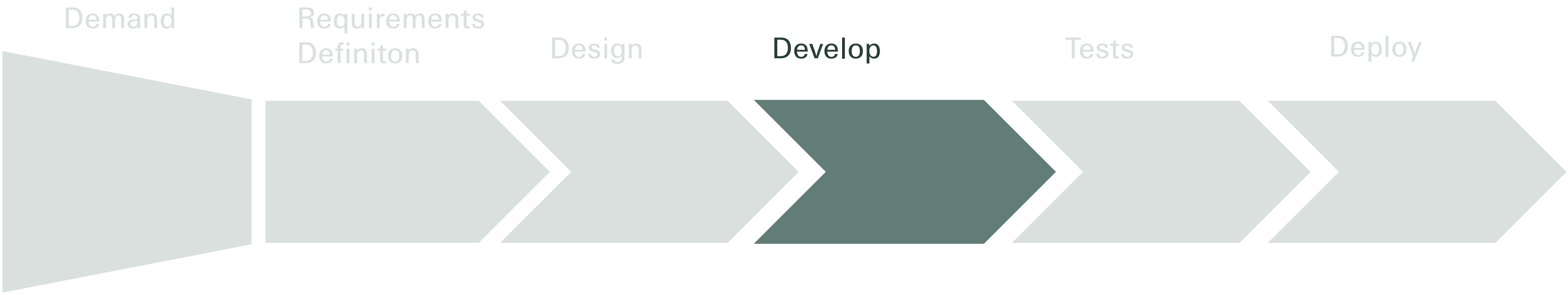
- Who are typical requestors? (grouping)
- How are requirements gathered & documented?
- How are requirements prioritized?
- Ressources involved?
- Typical lead times?

Example Value Stream Questions
This was for a classic waterfall setup



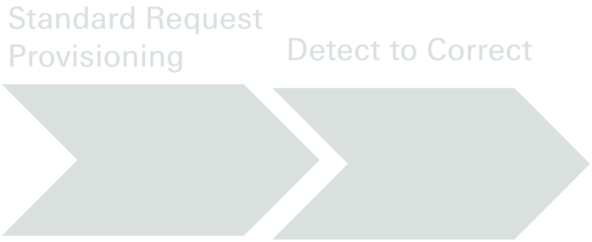
- Any design done before starting with developement
- Description of activities
- Dependencies
- Ressources involved?
- Typical lead times?

Value Example Value Stream Questions
This was for a classic waterfall setup

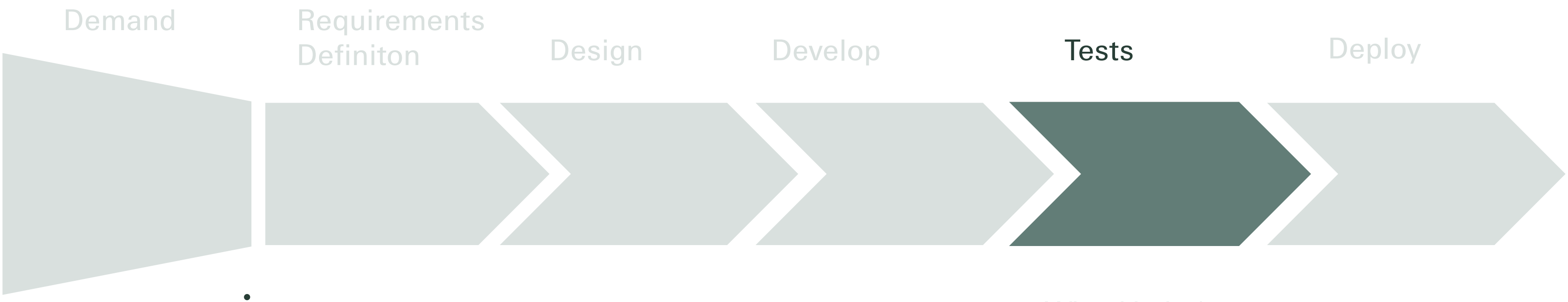


- …

- Description of activities
- Interfaces
- Which tools & plattforms?
- Version control?
- Any pre-flight & unit tests?
- Ressources involved?
- Typical lead times?



Example Value Stream Questions
This was for a classic waterfall setup



Demand

Requirements
Definiton

Design

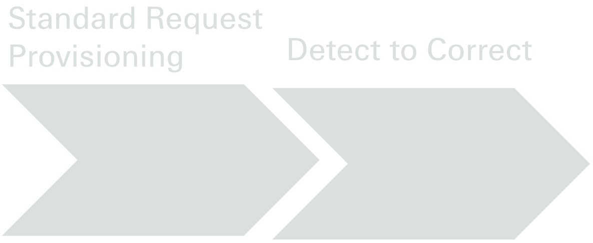
Develop

Tests

Deploy

- …

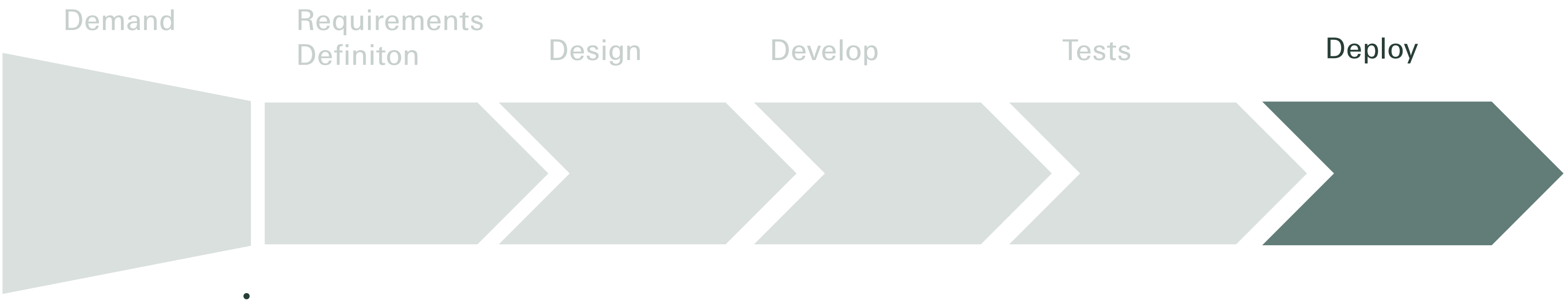
- What kind of test
  are performed?
- Test cases: How defined, performed
- What if tests fail?
- Automated tests?
- Ressources involved?
- Typical lead times??



Standard Request
Provisioning

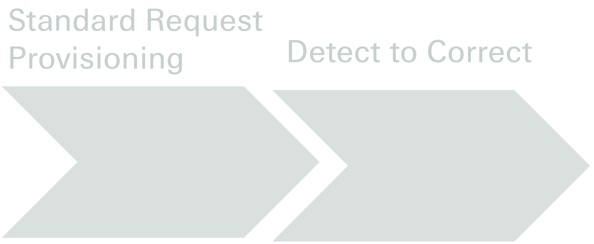
Detect to Correct

Example Value Stream Questions
This was for a classic waterfall setup

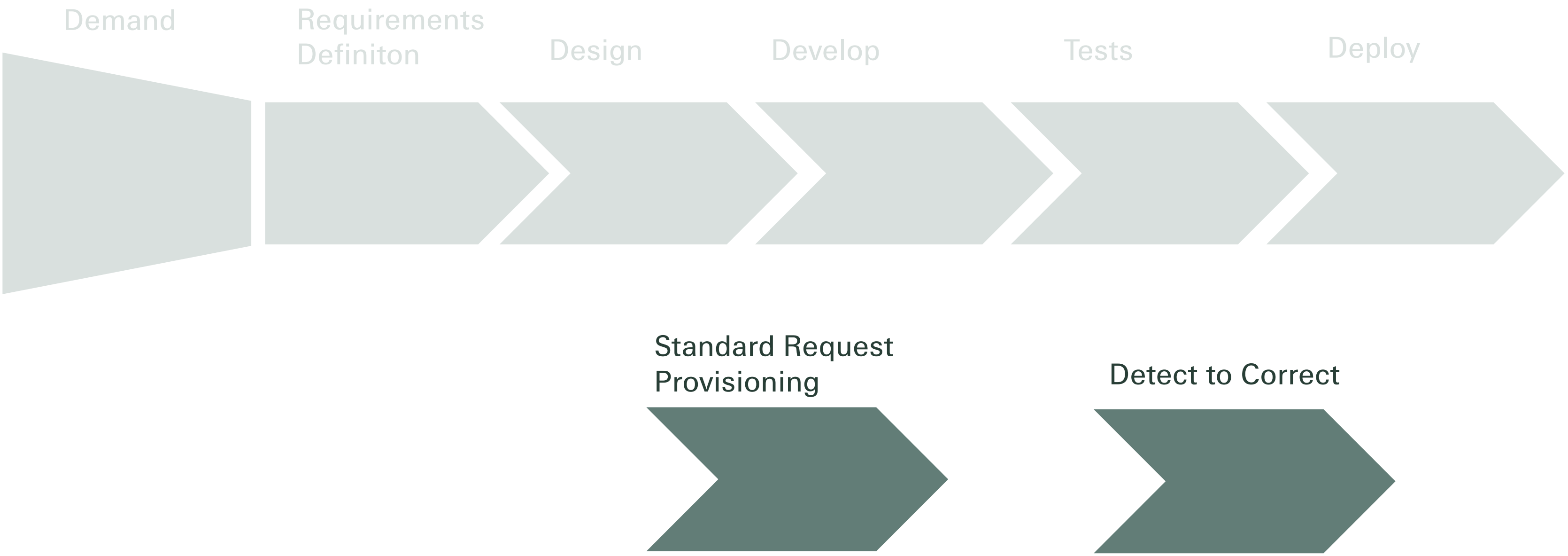


- ...

- Description of activities
- Ressources involved?
- Typical lead times?



Example Value Stream Questions
This was for a classic waterfall setup



Standard Request Provisioning

- Applicable?

Detect to Correct

- To be discussed later:
  - Incident Handling
  - Monitoring