"I'M HAVING TROUBLE WITH MY PRODUCT ROLL OUT."

An Architecture for Self-Organizing Continuous Delivery Pipelines

1st Vienna Software Seminar

Andreas Steffens
a.steffens@swc.rwth-aachen.de

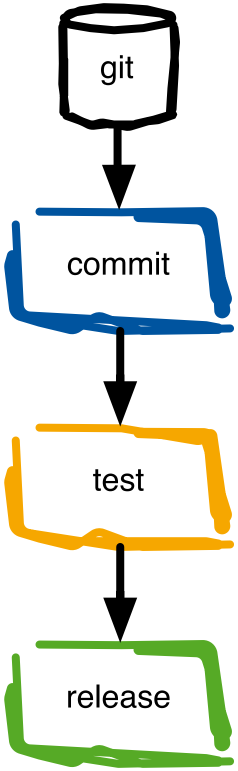
Software Construction

RWTH AACHEN UNIVERSITY

> At an abstract level, a deployment pipeline is an automated manifestation of your process for getting software from version control into the hands of your users.

Humble, Farley

Relation of Architecture and Continuous Delivery



git

commit

test

release

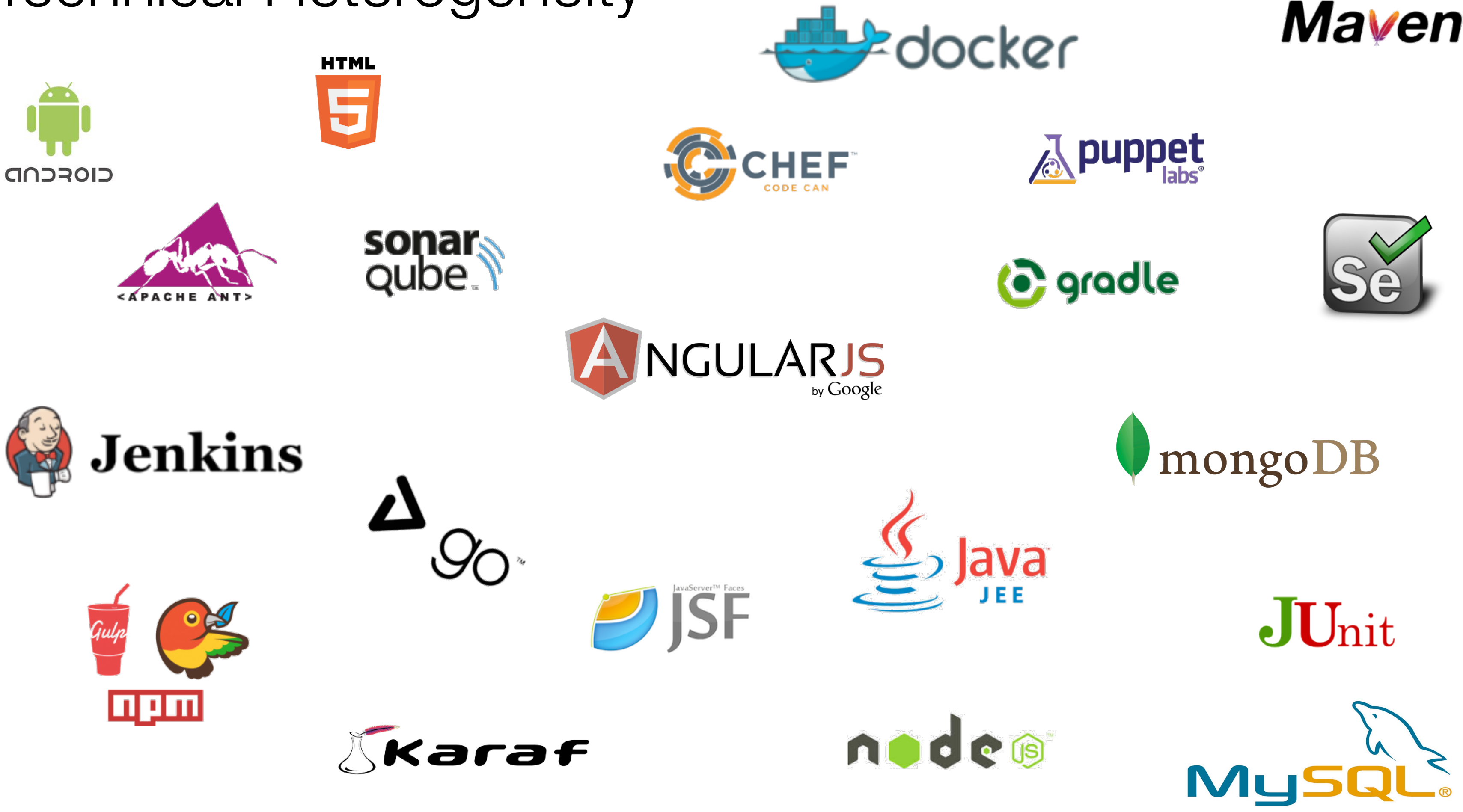
=

Heterogeneity

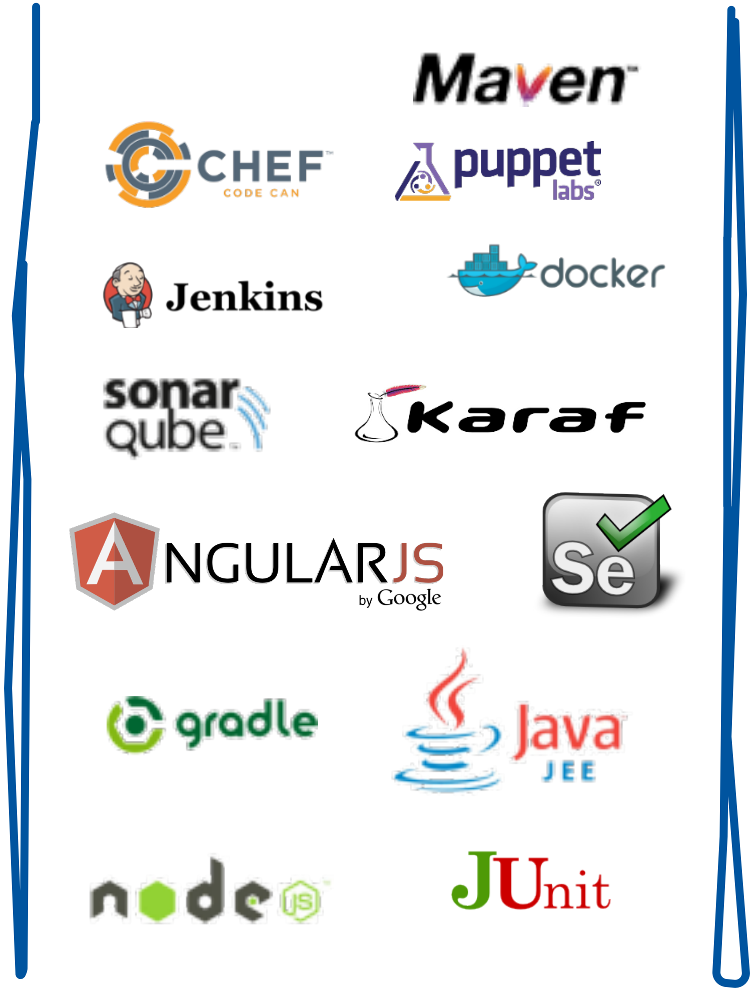
- Commit
- Unit Test
- Performance Test
- Security
- UAT
- Release
- Provision
- Deploy
- Baking
- Static/Dynamic Analysis
- Manual Approval
- Apply Policies
- Monitor
- Operate/Run

More heterogeneity in the software delivery process

Technical Heterogeneity



Build Monolith

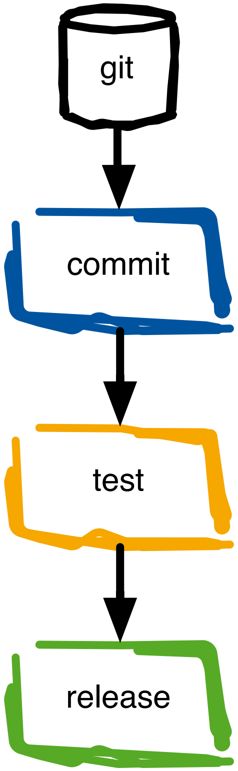


Build Hero

ThoughtWorks® Technology Radar 2016:
Anti-Pattern "A single CI instance for all teams"

Relation of Architecture and Continuous Delivery

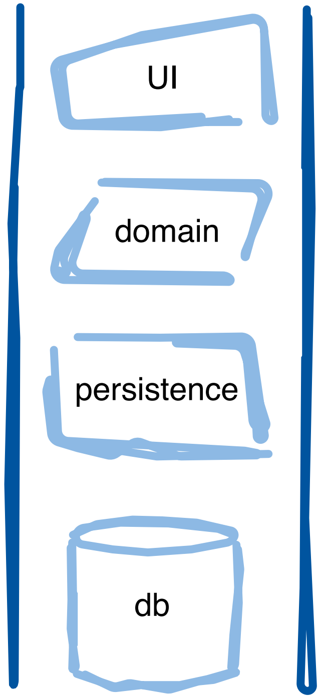
Impact of ~~Architecture~~ Software Development Process



Evolution

- Process steps
- Technologies
- Policies
- Architecture decisions

Requirements



Challenges

- Pipeline model:

  "The build system [..] or scripts are complicated or complex"

- Pipeline system:

  "The build system cannot be modified flexibly"

Laukkanen. Itkonen, Lassenius
"Problems, causes and solutions when adopting continuous delivery—A systematic literature review"
Information and Software Technology (Feb. 2017)

Tackling Variability & Heterogeneity

The pipeline is a *member* of the software development process as the software itself

1. Requirements
2. **Design/Architecture**
3. Construction
4. Test
5. Evolution

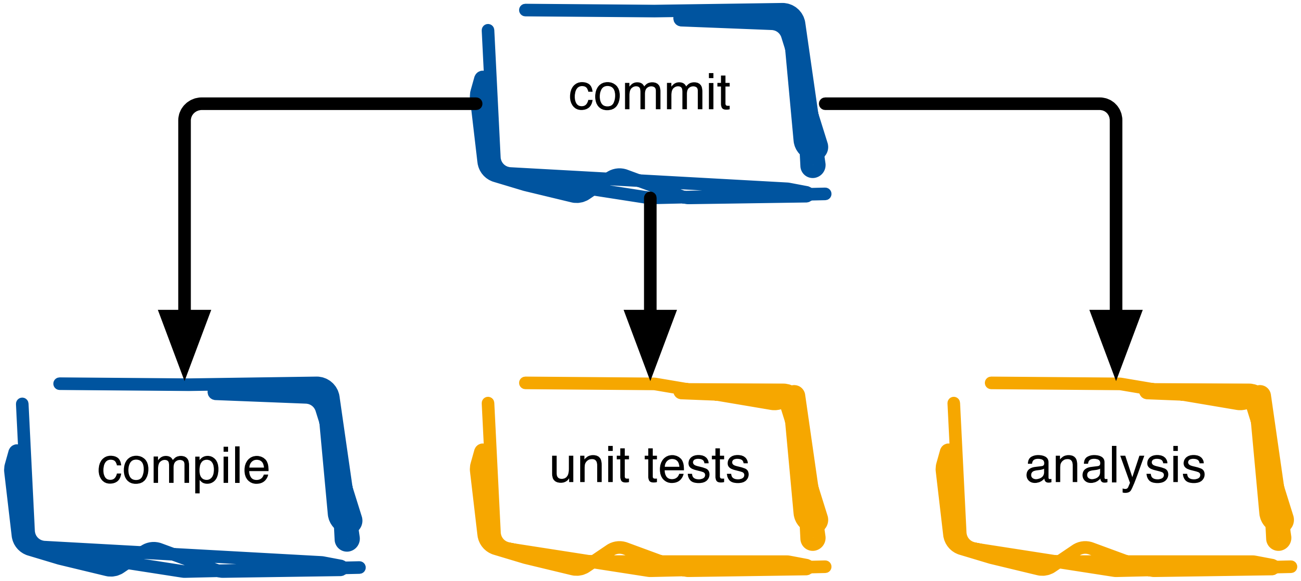
Self-organizing Pipelines

- adapt easily to changes
- Robust to unknown events

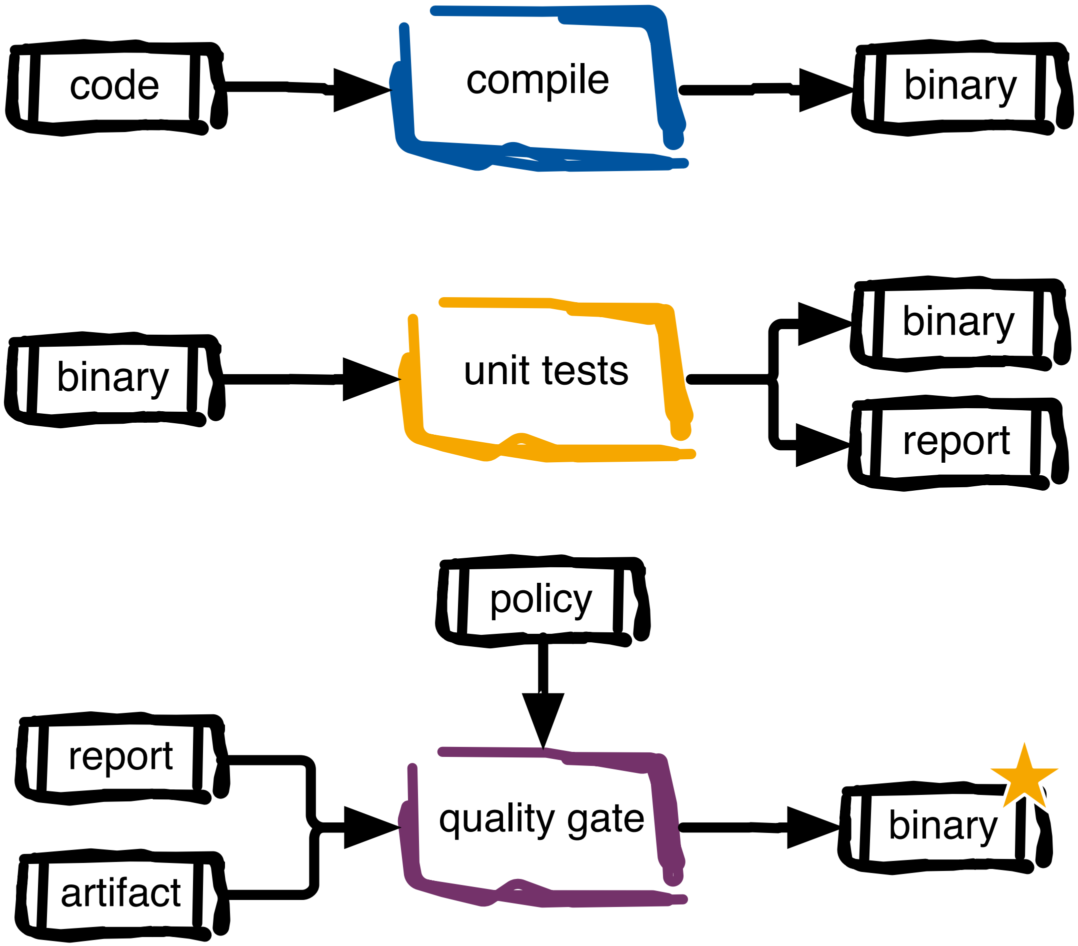
- In Software Engineering Terms
    - Adapt: flexiable & extensible
    - Robust: robust resilient

- “smart”

Commit Stage



Activities inside a pipeline



code → compile → binary

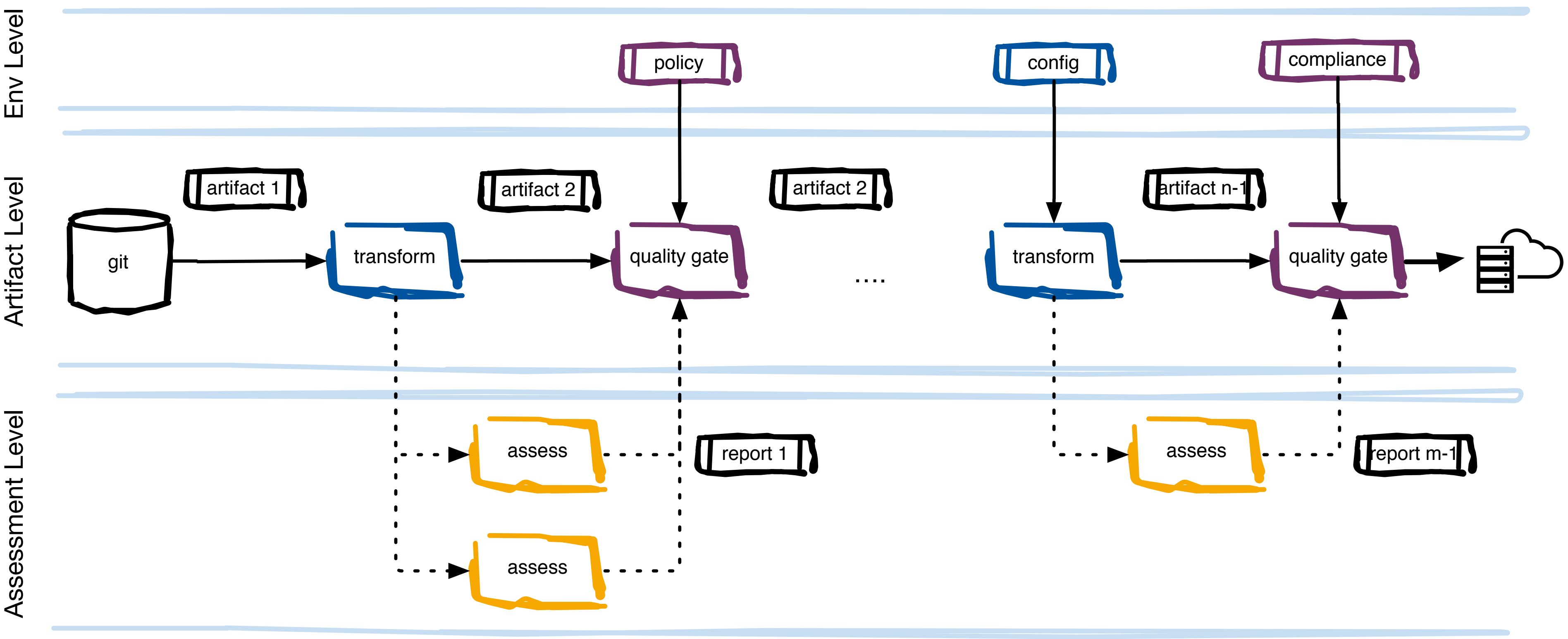
binary → unit tests → binary, report

policy, report, artifact → quality gate → binary

Transformation

Assessment

Decision Gate



- Minimal pipeline: only transformations
- External Information: policies and configuration

- Order of transformations depends on the artifacts

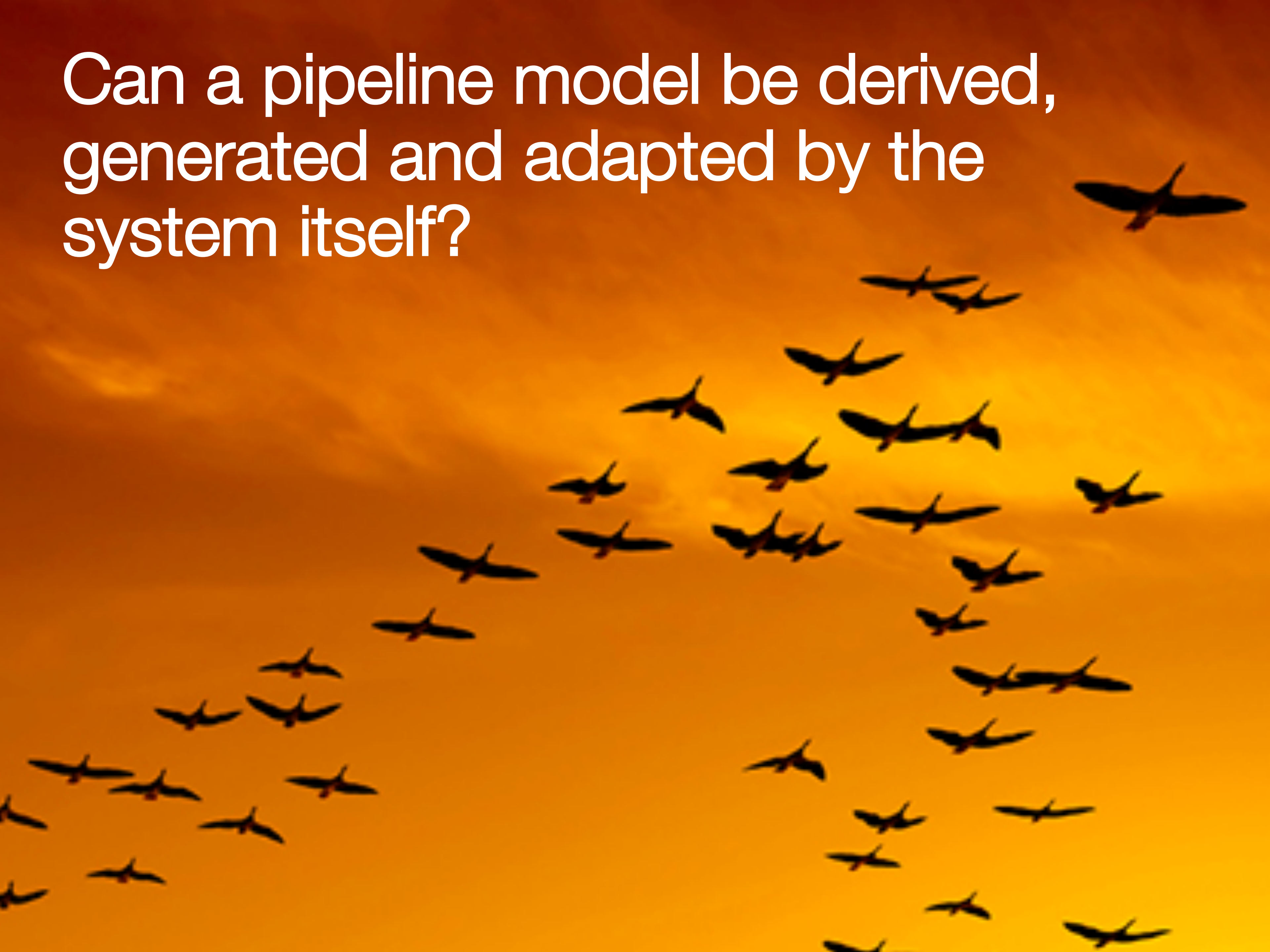
Pipeline Description Languages - PDL

- Lot’s of different models & DSLs

…

- Imperative/declarative style

- Separate pipeline models from pipeline execution



Can a pipeline model be derived, generated and adapted by the system itself?

Beyond Tellerrand

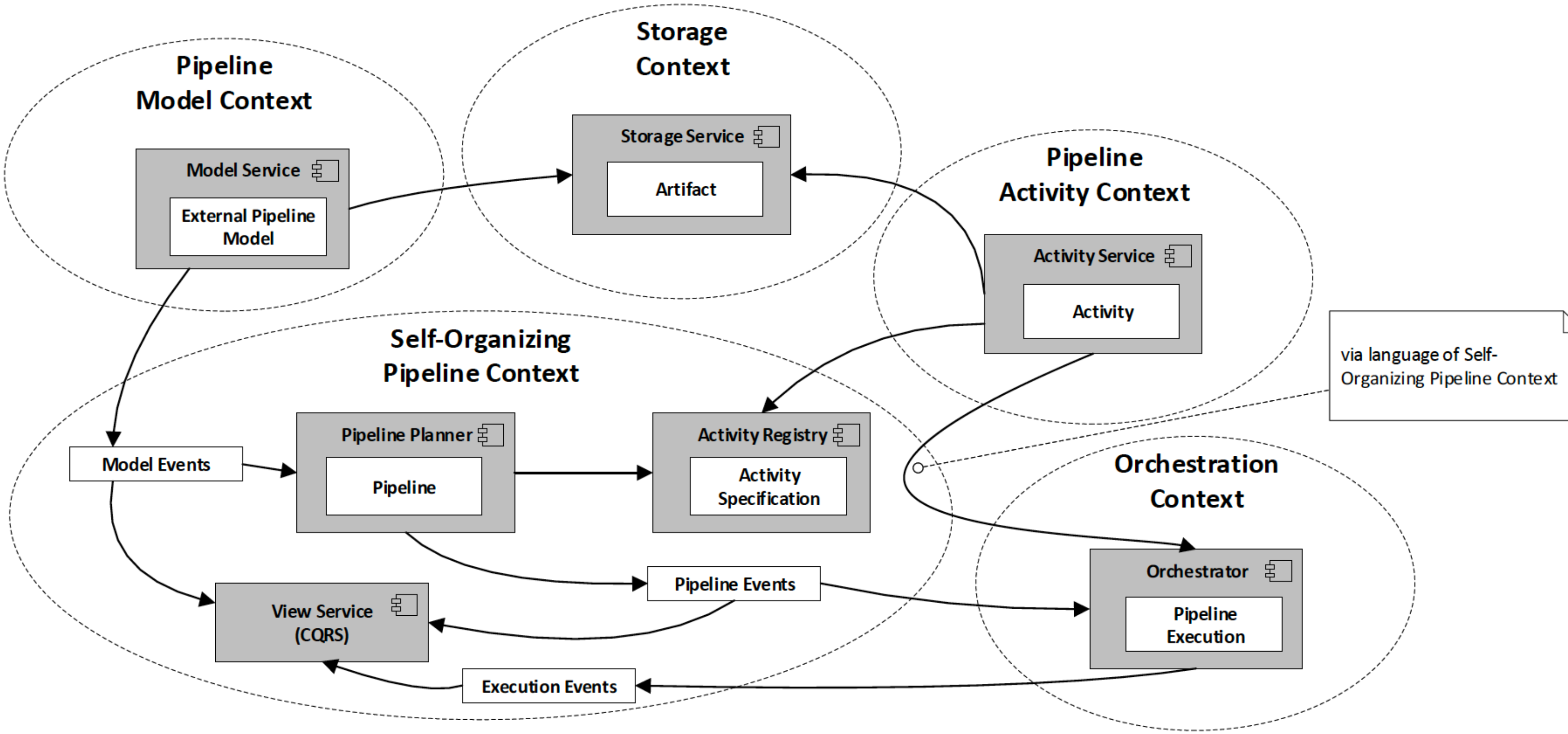
- Netflix Spinnaker:
  - microservices for deployment orchestration

- Pivotal Concourse
  - Isolated execution in Docker
  - Declarative simplified model

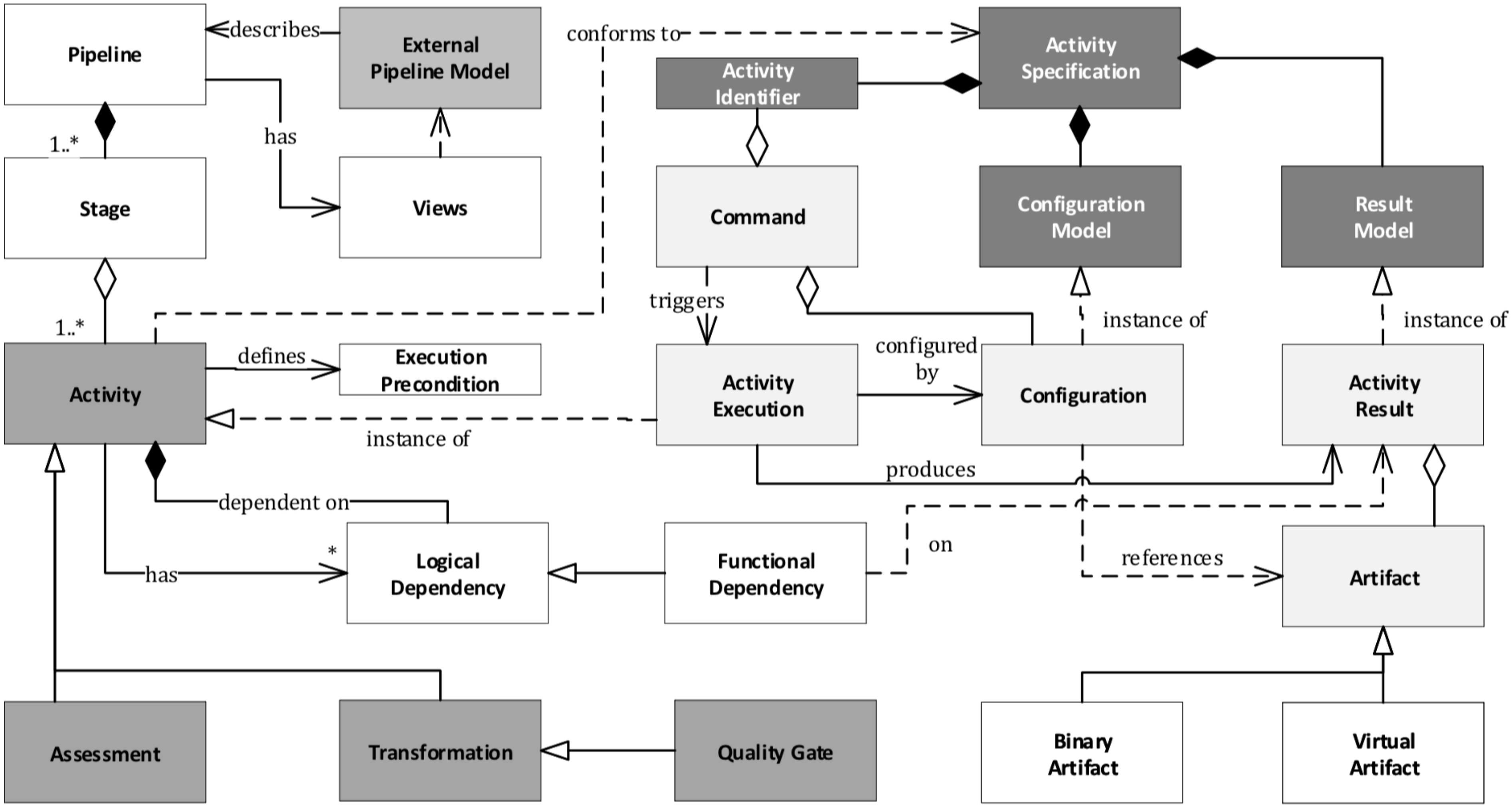
- Gitlab CI
  - Declarative DSL
  - Individual Runner

KEEP
CALM
AND
DO IT
YOURSELF

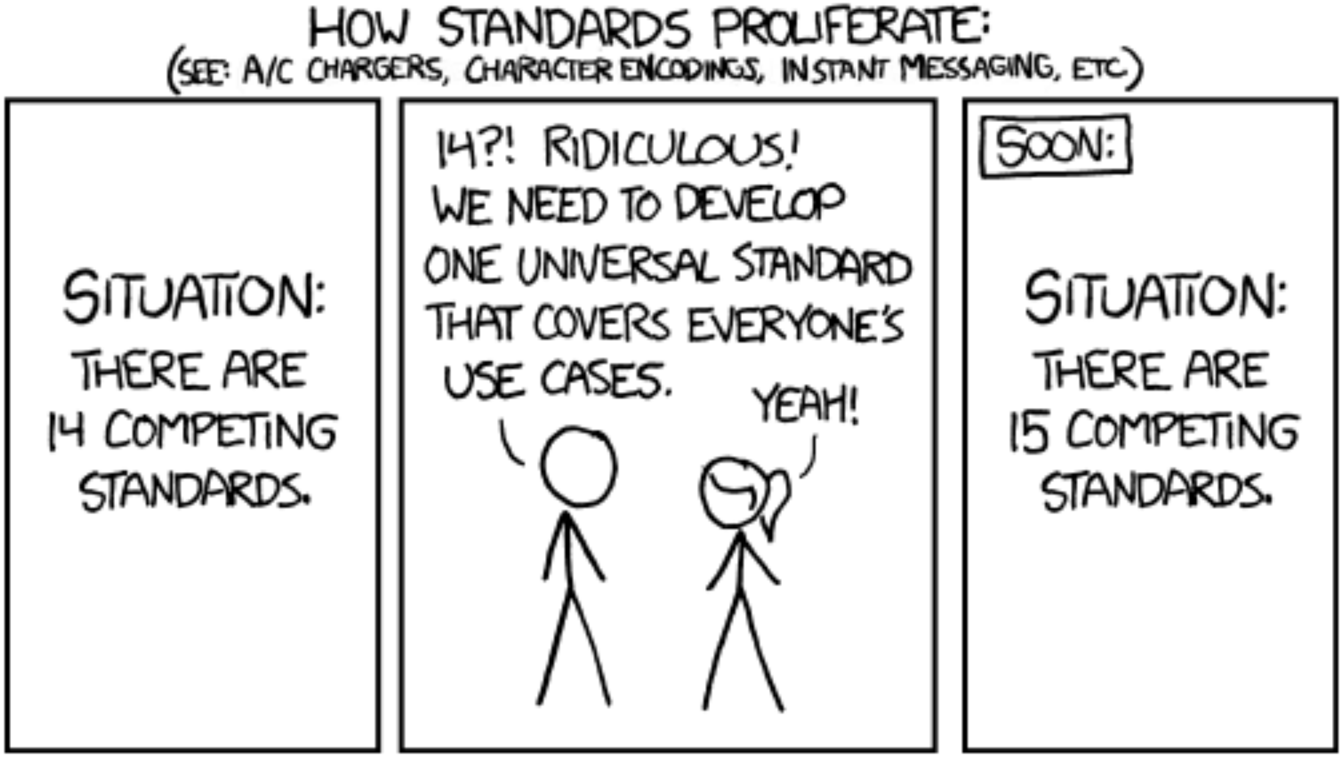
Domain Driven Design



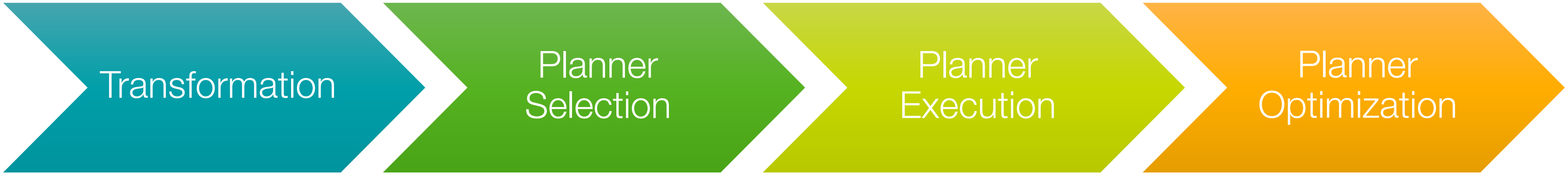
Domain Model



Pipeline DSL


HOW STANDARDS PROLIFERATE:
(SEE: A/C CHARGERS, CHARACTER ENCODINGS, INSTANT MESSAGING, ETC.)
SITUATION: THERE ARE 14 COMPETING STANDARDS.
14?! RIDICULOUS! WE NEED TO DEVELOP ONE UNIVERSAL STANDARD THAT COVERS EVERYONE'S USE CASES.
YEAH!
SOON:
SITUATION: THERE ARE 15 COMPETING STANDARDS.

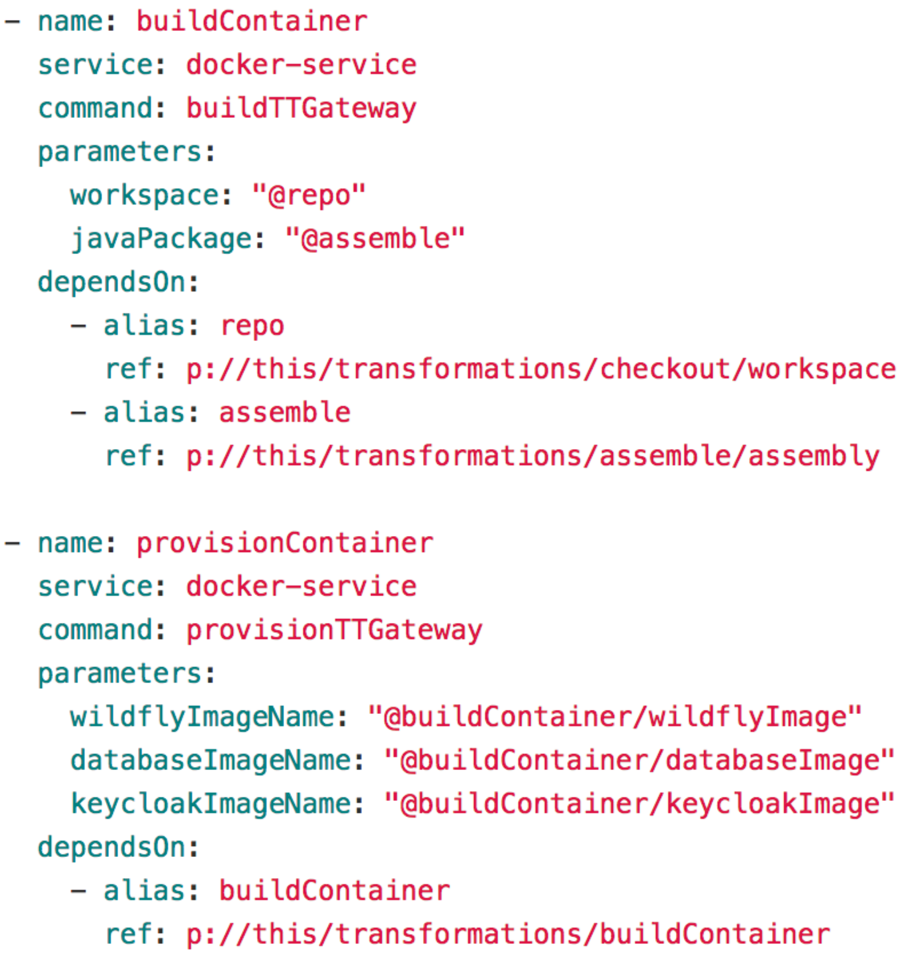
Core Planning Process



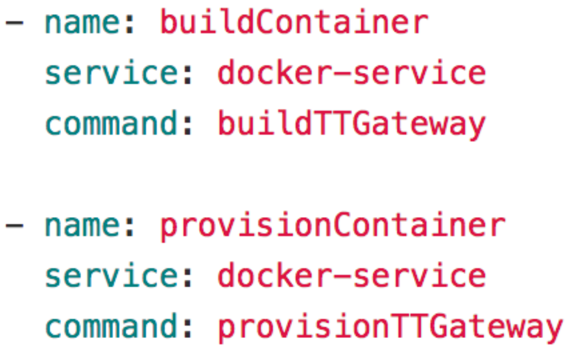
- **Transformation**: M2M from arbitrary PDL
- **Selection**: Which planner to use ?
  - Model level: specified activities & artifacts
  - Project level: use project data to plan (i.e. maven)

- **Execution**: derive order of transformations and assign assessments and gates
- **Optimization**: transform into an optimized equivalent model

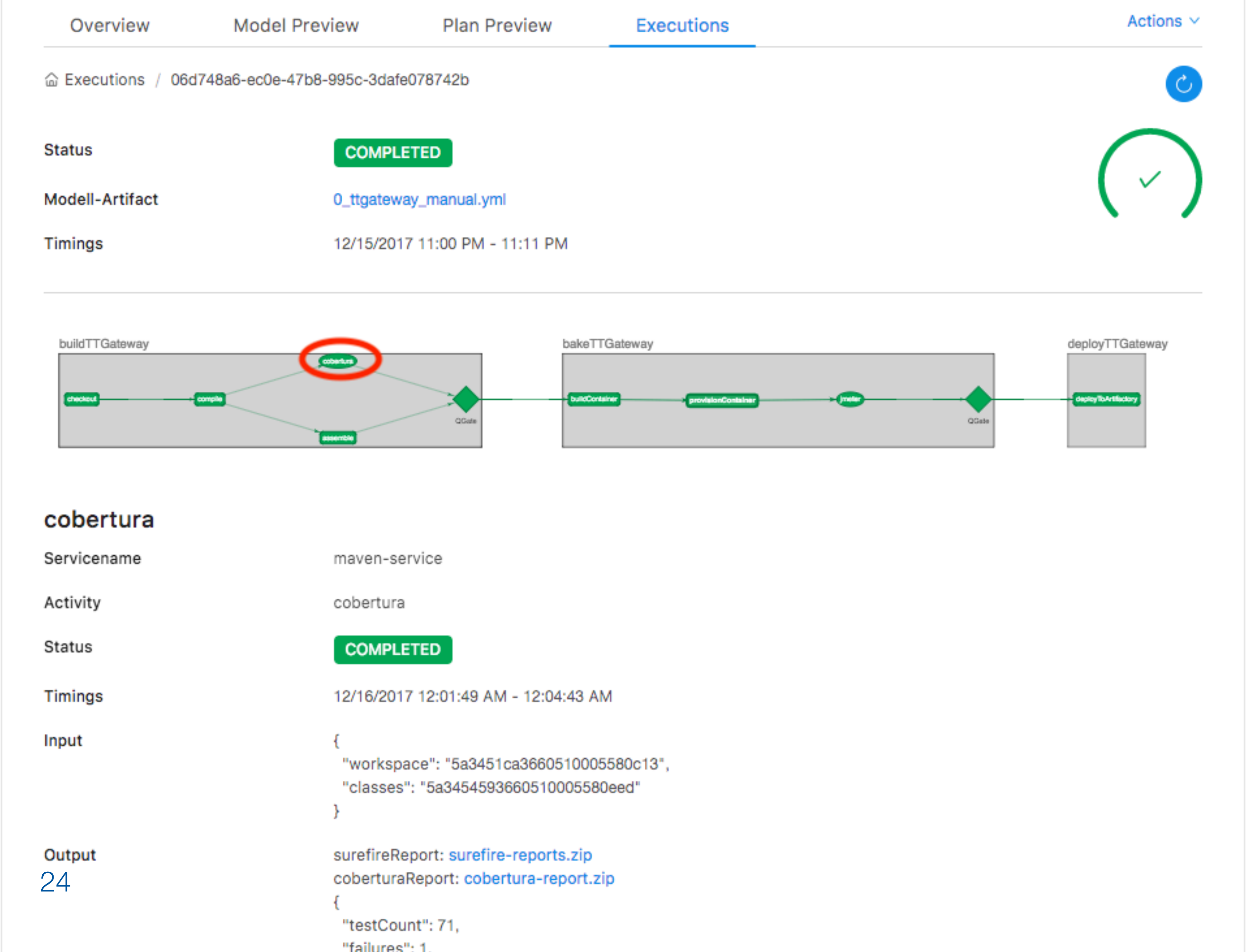
Simplify model by automatic mapping



```
- name: buildContainer
  service: docker-service
  command: buildTTGateway
  parameters:
    workspace: "@repo"
    javaPackage: "@assemble"
  dependsOn:
    - alias: repo
      ref: p://this/transformations/checkout/workspace
    - alias: assemble
      ref: p://this/transformations/assemble/assembly

- name: provisionContainer
  service: docker-service
  command: provisionTTGateway
  parameters:
    wildflyImageName: "@buildContainer/wildflyImage"
    databaseImageName: "@buildContainer/databaseImage"
    keycloakImageName: "@buildContainer/keycloakImage"
  dependsOn:
    - alias: buildContainer
      ref: p://this/transformations/buildContainer
```



```
- name: buildContainer
  service: docker-service
  command: buildTTGateway

- name: provisionContainer
  service: docker-service
  command: provisionTTGateway
```



Executions / 06d748a6-ec0e-47b8-995c-3dafe078742b

Status	COMPLETED
Modell-Artifact	0_ttgateway_manual.yml
Timings	12/15/2017 11:00 PM - 11:11 PM

cobertura

Servicename	maven-service
Activity	cobertura
Status	COMPLETED
Timings	12/16/2017 12:01:49 AM - 12:04:43 AM

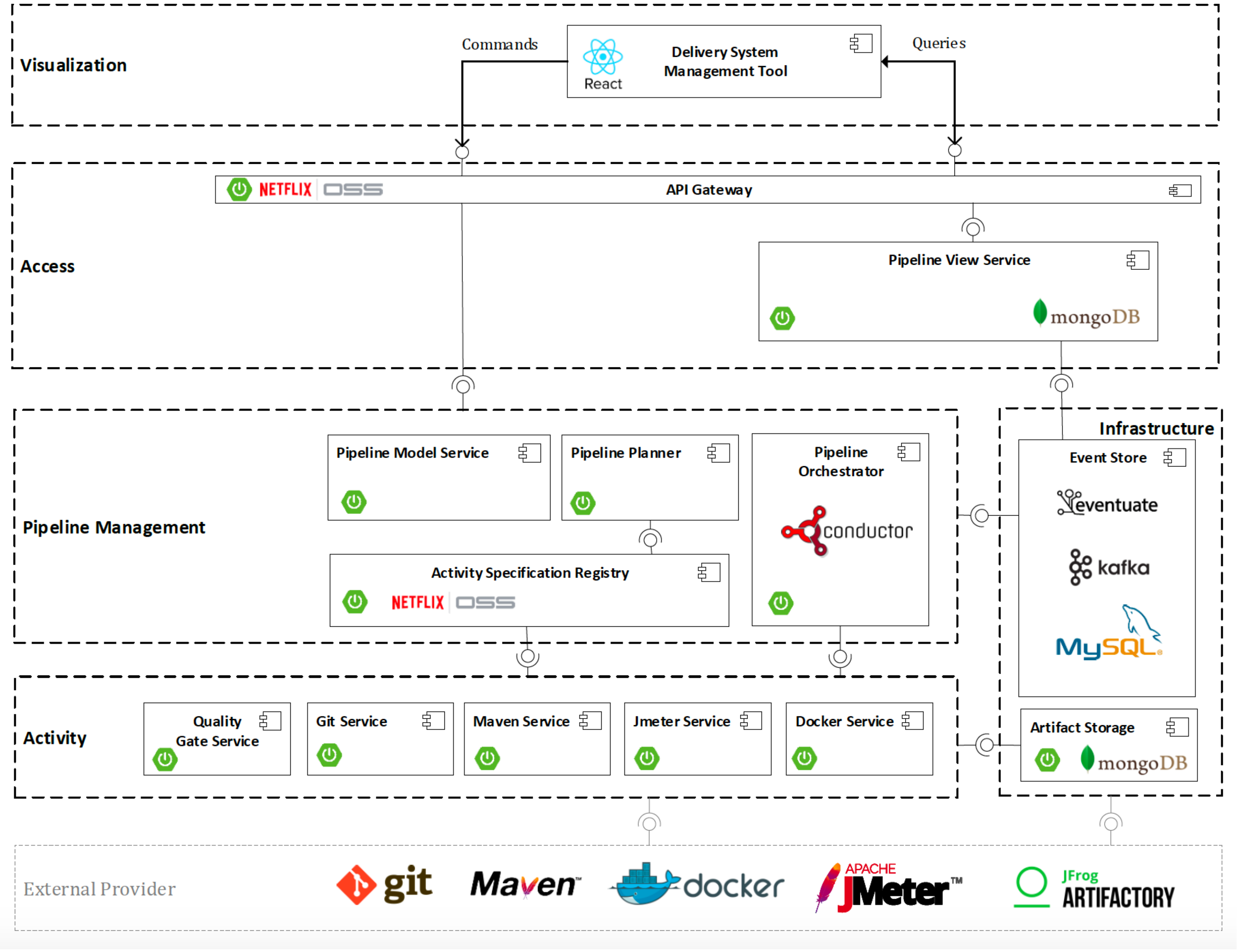
Input

```
{
  "workspace": "5a3451ca3660510005580c13",
  "classes": "5a3454593660510005580eed"
}
```

Output

surefireReport: surefire-reports.zip
coberturaReport: cobertura-report.zip

```
{
  "testCount": 71,
  "failures": 1,
```

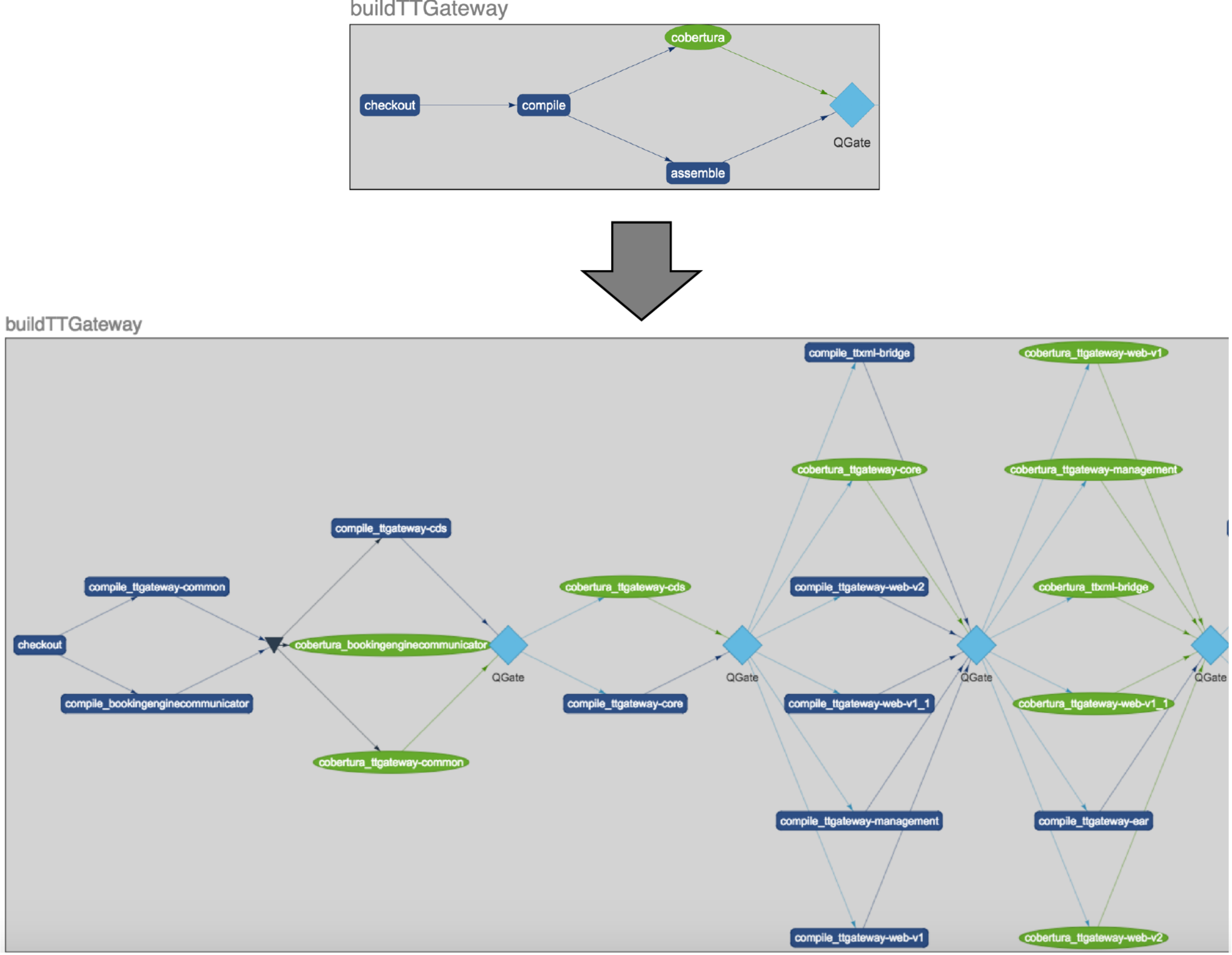

Visualization
Commands
React
Delivery System Management Tool
Queries
NETFLIX OSS
API Gateway
Access
Pipeline View Service
mongoDB
Infrastructure
Pipeline Management
Pipeline Model Service
Pipeline Planner
Pipeline Orchestrator
conductor
Activity Specification Registry
NETFLIX OSS
Event Store
eventuate
kafka
MySQL
Activity
Quality Gate Service
Git Service
Maven Service
Jmeter Service
Docker Service
Artifact Storage
mongoDB
External Provider
git
Maven
docker
APACHE JMeter™
JFrog ARTIFACTORY

Case Study

- Real industry project: API Gateway
- Tech: Java EE, Database, Keycloak
- Tools: JMeter, Docker

- Assessment Focus: Coverage & Performance

- Objective:
  - Is this approach applicable?
  - What is the impact of self-organization?

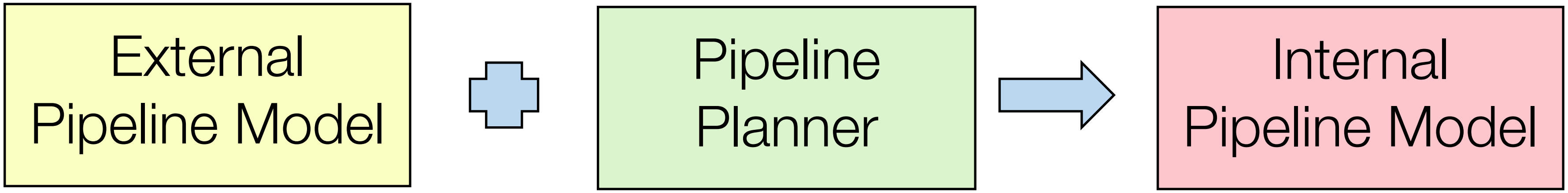

buildTTGateway
checkout
compile
cobertura
assemble
QGate
buildTTGateway
checkout
compile_ttgateway-common
compile_bookingenginecommunicator
compile_ttgateway-cds
cobertura_bookingenginecommunicator
cobertura_ttgateway-common
QGate
cobertura_ttgateway-cds
compile_ttgateway-core
QGate
compile_ttxml-bridge
cobertura_ttgateway-core
compile_ttgateway-web-v2
compile_ttgateway-web-v1_1
compile_ttgateway-management
compile_ttgateway-web-v1
QGate
cobertura_ttgateway-web-v1
cobertura_ttgateway-management
cobertura_ttxml-bridge
cobertura_ttgateway-web-v1_1
compile_ttgateway-ear
cobertura_ttgateway-web-v2
QGate

Some preliminary results

FAIL FAST.
FAIL EARLY.
FAIL OFTEN.

Pipeline Modelling

- Modeling “Power”
- Idea: increase **ease of use**
- Automatic Planning



Collaboration & Platforms

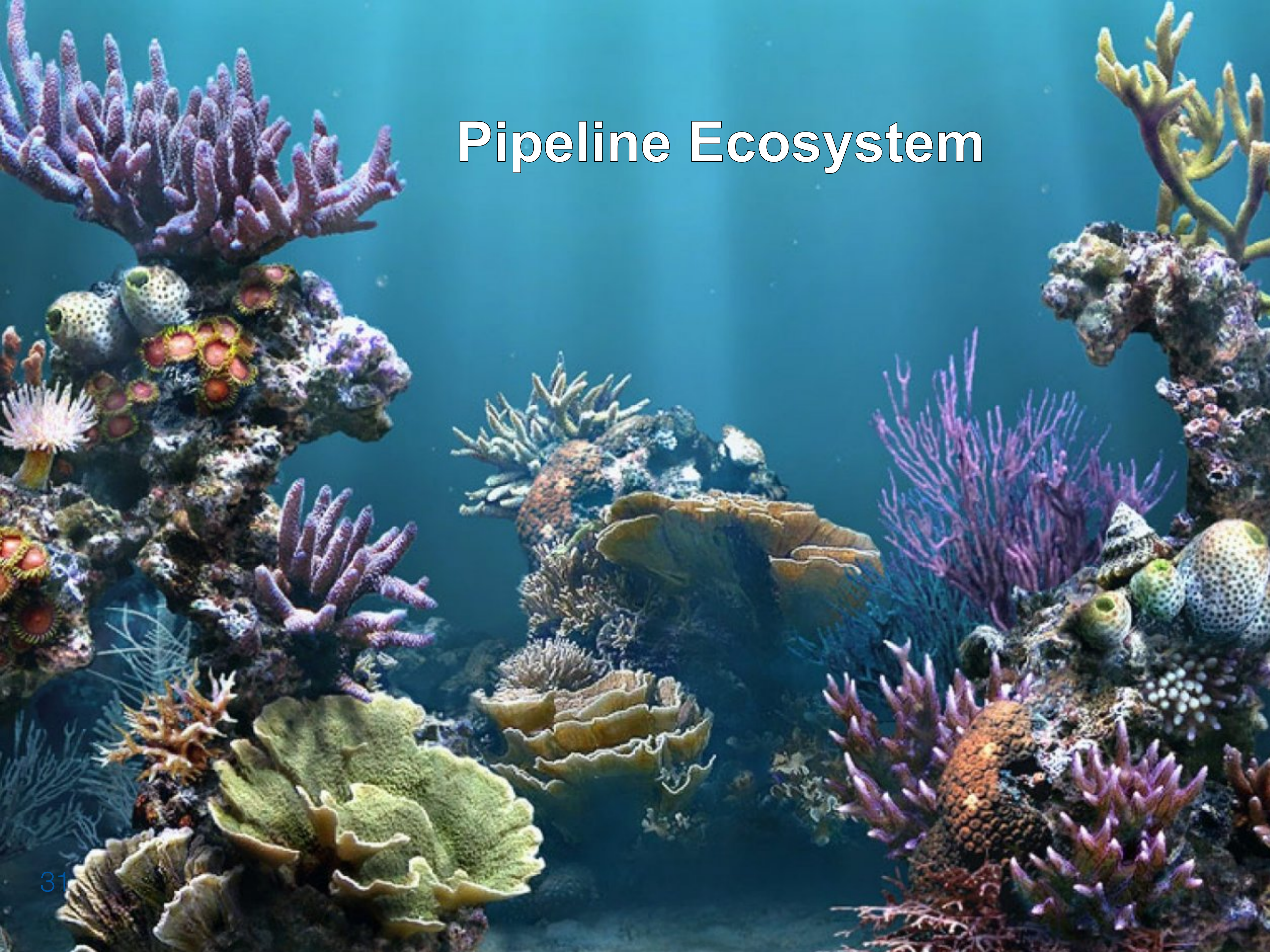
- Tool: CodeFlow
  - Homegrown Code Review
  - Toolbuilder culture

Microsoft

- Platform: Tricorder
  - Ecosystem of Quality Measurement Tools

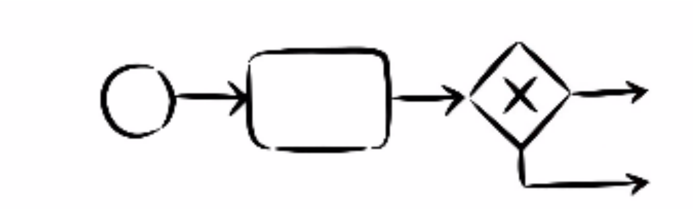
Google

Pipeline Ecosystem



Future Work

- Integrate more pipeline models:
  - currently BPMN (manual approval)



- Make the Pipeline smarter
  - More built-in analytics
  - Learn from build history
    - Test selection
    - Test prioritization
    - Artifact decomposition
  - Learn from operation data
    - Recommend assessment

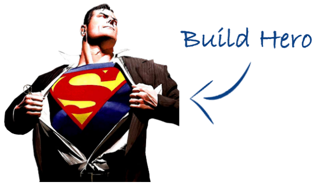
- Validate pipeline against policies
  - Missing assessments

§

- Re-plan during execution

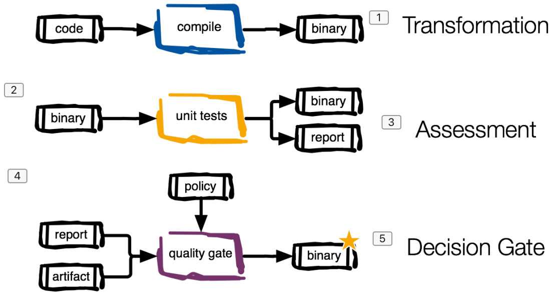
REALITY
CHECK
AHEAD

Monolith



ThoughtWorks Technology Radar 2016:
Anti-Pattern "A single CI instance for all teams"

Activities inside a pipeline

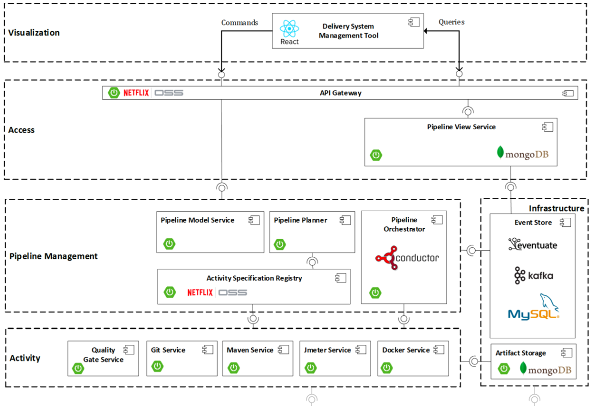


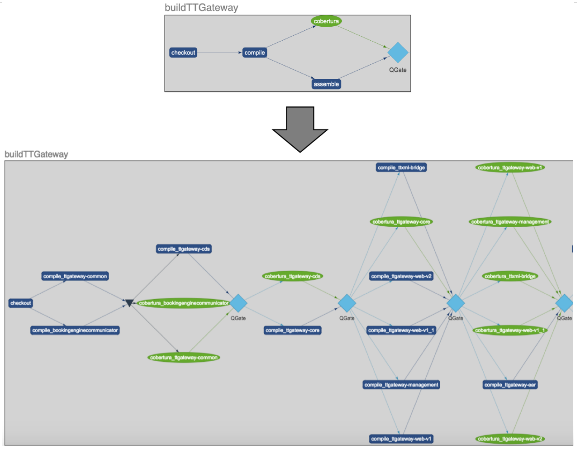
Simplify model by automatic mapping

```
- name: buildContainer
  service: docker-service
  command: buildTTGateway
  parameters:
    workspace: "@repo"
    javaPackage: "@assemble"
  dependsOn:
    - alias: repo
      ref: p://this/transformations/checkout/workspace
    - alias: assemble
      ref: p://this/transformations/assemble/assembly

- name: provisionContainer
  service: docker-service
  command: provisionTTGateway
  parameters:
    wildflyImageName: "@buildContainer/wildflyImage"
    databaseImageName: "@buildContainer/databaseImage"
    keycloakImageName: "@buildContainer/keycloakImage"
  dependsOn:
    - alias: buildContainer
      ref: p://this/transformations/buildContainer
```

1

```
- name: buildContainer
  service: docker-service
  command: buildTTGateway

- name: provisionContainer
  service: docker-service
  command: provisionTTGateway
```





THANKS!

A pipeline is a core/shared software project
of each software organization!

Breakout Groups

- Pipeline Design and Optimization
- DevOps-Supporting Architectures – Only Microservices?
- Challenges for Continuous Delivery in IoT
- Software Complexity – Metrics of Software Landscapes with DevOps

Welcome
to the Present Continuous
It's happening here!
mmarly

German
Informatics Society

Working Group
Microservices & DevOps
ak-msdo.gi.de
22/23.3.2018 Aachen

Communities interested in DevOps

3rd Workshop on CSE
Submission: 10.1.2018