

# **FAST AND RESILIENT INTEGRATION TESTING**

## **CONTINUOUS LIFECYCLE 2015**

   Dr. Thomas Schank

  Max F. Albrecht

<http://drtom.ch/talks/2015/CL>



Version 1.0.0

# Digitales Areal ZHdK

Ideen, Projekte, Werke – künstlerisch und wissenschaftlich: Das Medienarchiv der Künste ist die Plattform der ZHdK zum gemeinschaftlichen Arbeiten mit Medien und Teilen von Inhalten.

ZHdK-Login

Externe

Alle Funktionen nutzen und auf mehr Inhalte zugreifen.

Anmelden

Erkunden

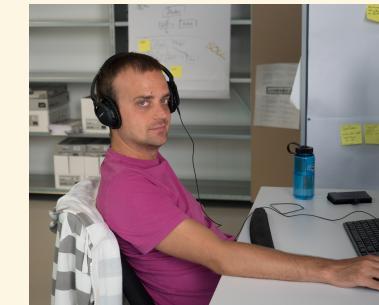
Suche

Support

ZHdK-Katalog [Alle anzeigen →](#)

# MADEK TEAM & US

 ZURICH UNIVERSITY OF THE ARTS



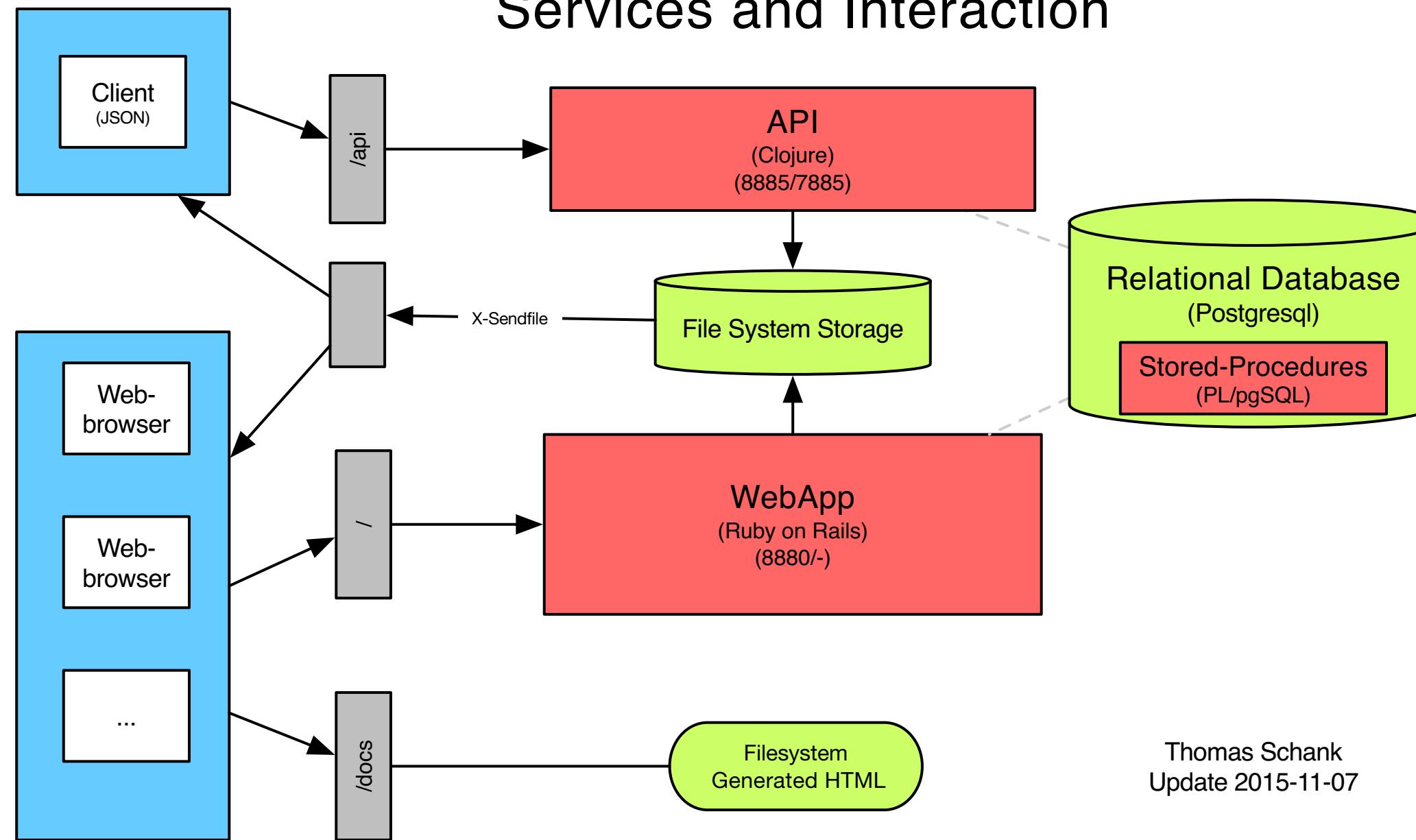
- Thomas: Software-Architect, Developer, CI-Infrastructure
- Max: Frontend Software-Engineer, Meta-Data Concepts

# MADEK - MEDIENARCHIV DER KÜNSTE

## ARCHITECTURE & TECHNOLOGIES

- Ruby on Rails, Clojure
- React with progressive enhancement
- 3-tier web-application
- towards micro-services
- deployment via Ansible to private cloud

# Madek 3.0 Architecture Services and Interaction



Thomas Schank  
Update 2015-11-07

# MADEK TESTING

"specification by example"

- integration testing
- components interaction

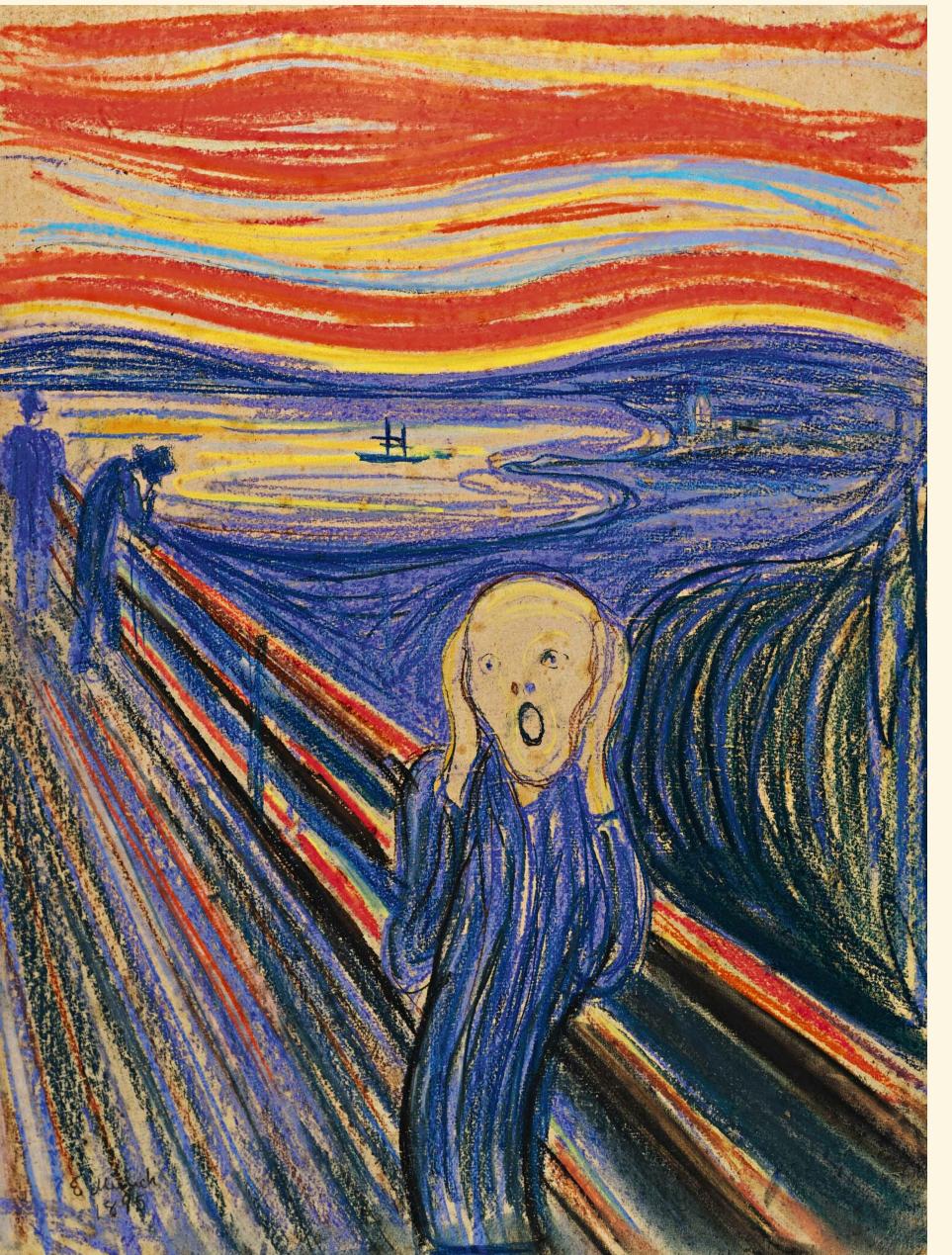
# 1. THE PROBLEM



# MADEK PROJECT 2012

many new features, many new tests

- testing time 1 1/2 - 2 hours, increasing
- more and more failing tests: **false negatives**
- 1/8 builds pass



## TRY TO IMPROVE TESTS

- very time and resources consuming
- improvement for some time
- new features and new tests made efforts futile

# **MANUAL RETRYING**

automated tests, local retries

automatic → semi automatic testing

## 2. COMPREHENSION



# PROBABILITY OF A FALSE NEGATIVE FOR A WHOLE TEST-SUITE

	Expression	Example
probability false negative single test	$p_f$	3%
probability "success"	$p_s = 1 - p_f$	0.97
number of tests	$n$	100
probability "success" whole suite	$P_s = p_s^n = (1 - p_f)^n \approx 5\%$	

→ only one out of 20 will pass as it should

"succes" = true positive

# WHY RETRYING WORKS SO WELL

let k number of independent retries per test

$$P_s(n) = (1 - p_f)^n \Rightarrow P'_s(n, k) = (1 - p_f^k)^n$$

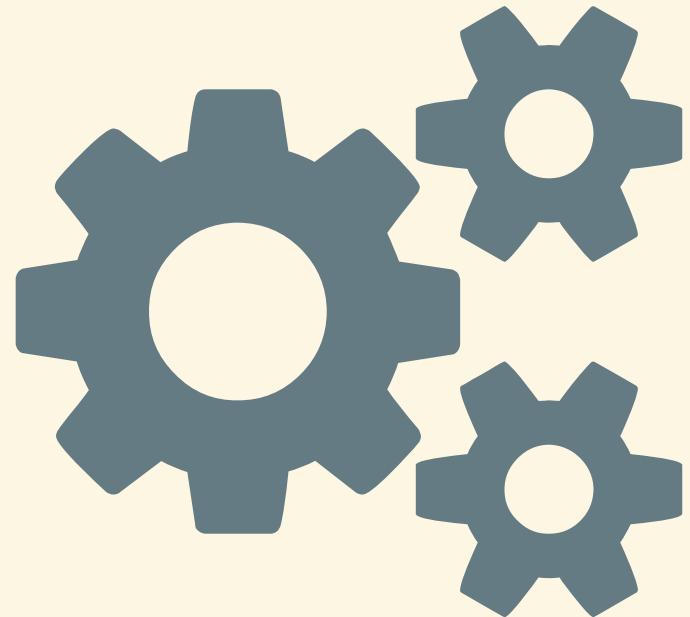
Expected successful outcome for n = 100 and p<sub>f</sub> = 0.03

k	P'_s
1	5%
2	91%
3	99.7%

## 2. COMPREHENSION - CONCLUSION

- more tests → **exponential** increase of likeliness for false negatives
- compensate by **retrying** single tests just a **few times**
  - **retrying is not an anti-pattern**
  - **it can be a necessity**

# **3. IMPLEMENTATION**



## Projekt MAdeK\_AT\_\_next\_\_AGGREGATOR



Arbeitsbereich



Letzte Änderungen

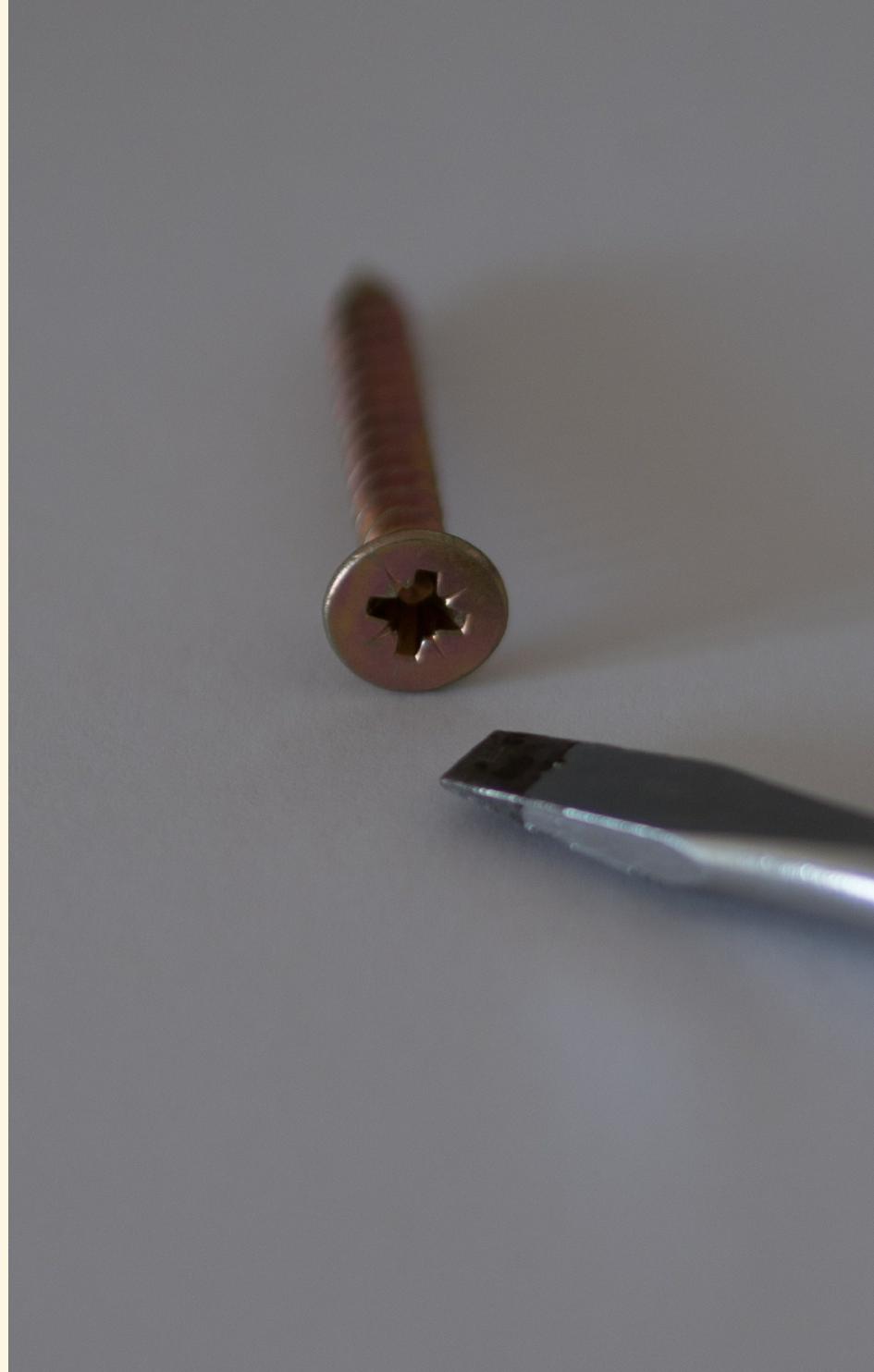
### Vorgelagerte Projekte

- [MAdeK AT next feature admin\\_media\\_sets](#)
- [MAdeK AT next feature admin\\_people](#)
- [MAdeK AT next feature admin\\_previews](#)
- [MAdeK AT next feature admin\\_specify\\_links\\_and\\_logo](#)
- [MAdeK AT next feature admin\\_users](#)
- [MAdeK AT next feature apply\\_to\\_all](#)
- [MAdeK AT next feature batch\\_edit\\_permissions](#)
- [MAdeK AT next feature browse](#)
- [MAdeK AT next feature configurable\\_contexts\\_in\\_lists](#)
- [MAdeK AT next feature context](#)
- [MAdeK AT next feature dashboard\\_page](#)
- [MAdeK AT next feature delete](#)
- [MAdeK AT next feature explore\\_page](#)
- [MAdeK AT next feature filter\\_media\\_resource\\_types](#)
- [MAdeK AT next feature filter\\_panel](#)
- [MAdeK AT next feature filter\\_set](#)
- [MAdeK AT next feature graph\\_visualization](#)
- [MAdeK AT next feature guest\\_user](#)
- [MAdeK AT next feature import](#)
- [MAdeK AT next feature import\\_01](#)
- [MAdeK AT next feature import\\_02](#)
- [MAdeK AT next feature import\\_encoding](#)
- [MAdeK AT next feature import\\_metadata](#)
- [MAdeK AT next feature import\\_via\\_dropbox](#)
- [MAdeK AT next feature inheritance\\_of\\_contexts](#)
- [MAdeK AT next feature login](#)
- [MAdeK AT next feature metadata](#)
- [MAdeK AT next feature permissions](#)
- [MAdeK AT next feature permissions\\_01](#)
- [MAdeK AT next feature permissions\\_02](#)
- [MAdeK AT next feature preview](#)
- [MAdeK AT next feature search](#)
- [MAdeK AT next feature showing\\_media\\_resources](#)
- [MAdeK AT next feature uberadmin](#)
- [MAdeK AT next feature welcome\\_page](#)
- [MAdeK AT next feature workgroups](#)
- [MAdeK AT next spec admin](#)
- [MAdeK AT next spec controllers](#)
- [MAdeK AT next spec controllers\\_02](#)
- [MAdeK AT next spec misc](#)
- [MAdeK AT next spec models](#)
- [MAdeK AT next spec requests](#)

# JENKINS

- fall 2012
- build creates other builds via the Jenkins API
- last build aggregates
- solved false negative problem (partly)
- testing time: 15 - 25 minutes

→ it worked



- frequent code pushes interfere
  - "REST-like style API" → not much like REST
  - considerable effort and maintenance
- Jenkins and "CI-X" just aren't made for this

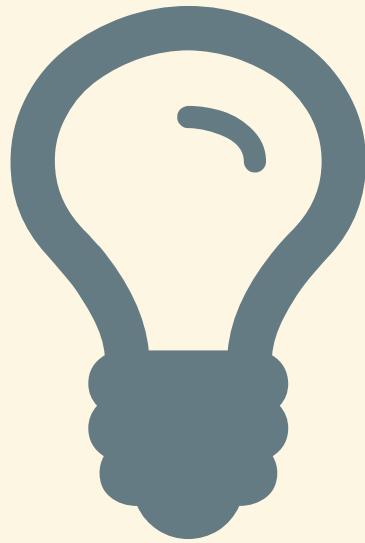
# CIDER-CI

homegrown solution, started in spring 2013

- inherent support for **retries** and **parallelization**
- **test reproducibility**
- **tight integration with source code**
- **manage services** while testing
- support everything from **testing** to **deployment**

ready to use in fall 2013, never looked back

# **4. CONCEPTS IN CONTEXT**

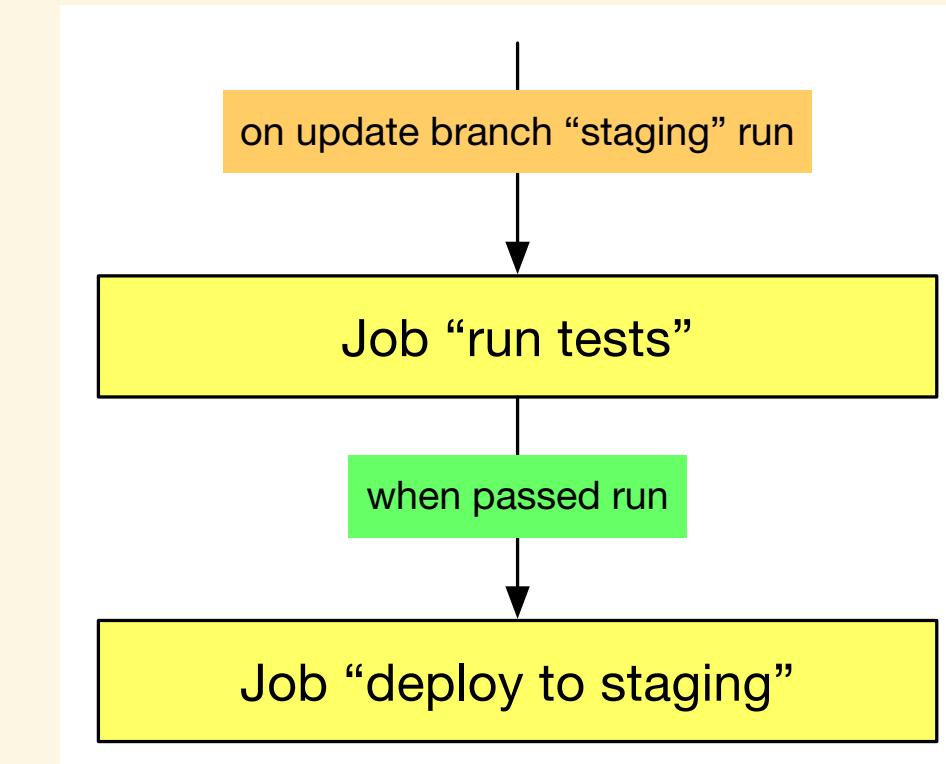


# JOBS

## EXAMPLES

- run test-suite
- perform static code checks
- build
- deploy

jobs can be triggered and can depend on each other



# PROJECT CONFIGURATION

cider-ci.yml file in the project

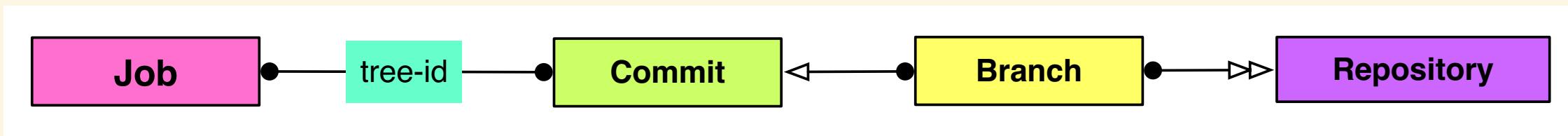
```
jobs:  
  deploy_test:  
    name: Deploy to test  
  
    depends-on:  
      - type: job  
        job: integration-tests  
        states: [passed]  
  
    run-on:  
      - type: branch  
        include-match: ^master$  
  
    # specify tasks etc
```

The source is the truth.

configuration: reproducible, reviews, audits ???

# CIDER-CI AND THE SOURCE CODE

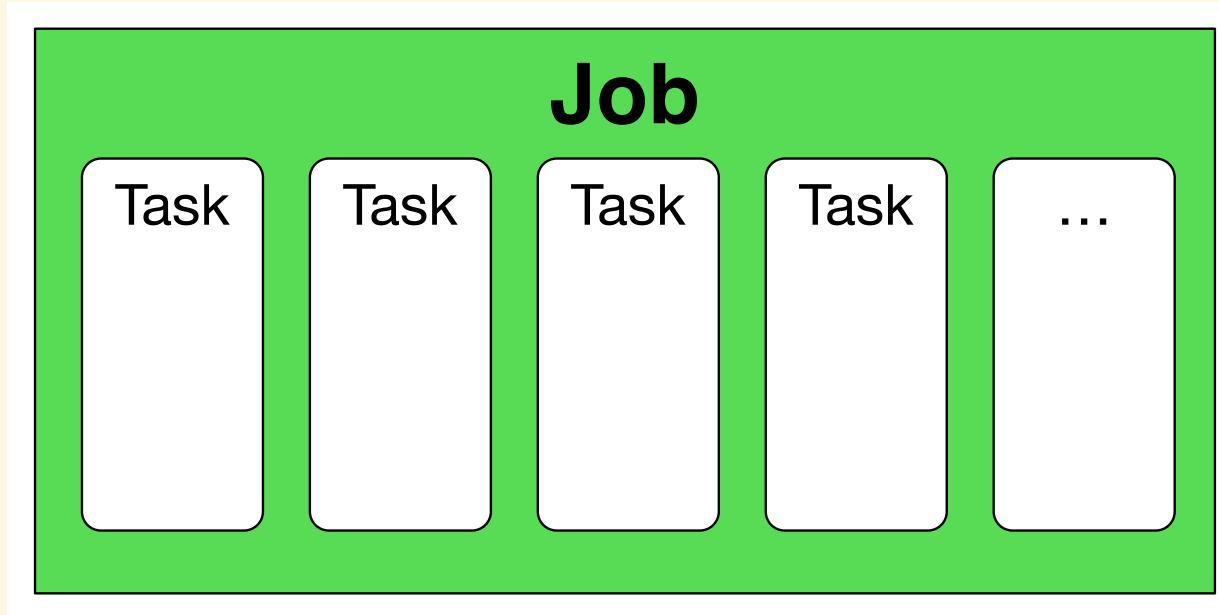
Cider-CI "knows" about commits, branches, submodules,...



**tree-id:** fingerprint of your source code

- **reproducibility**
- jobs can be run at any time (later)
- **binary search** for "bad" commits
- commit amends, squashing: **existing job remains valid**

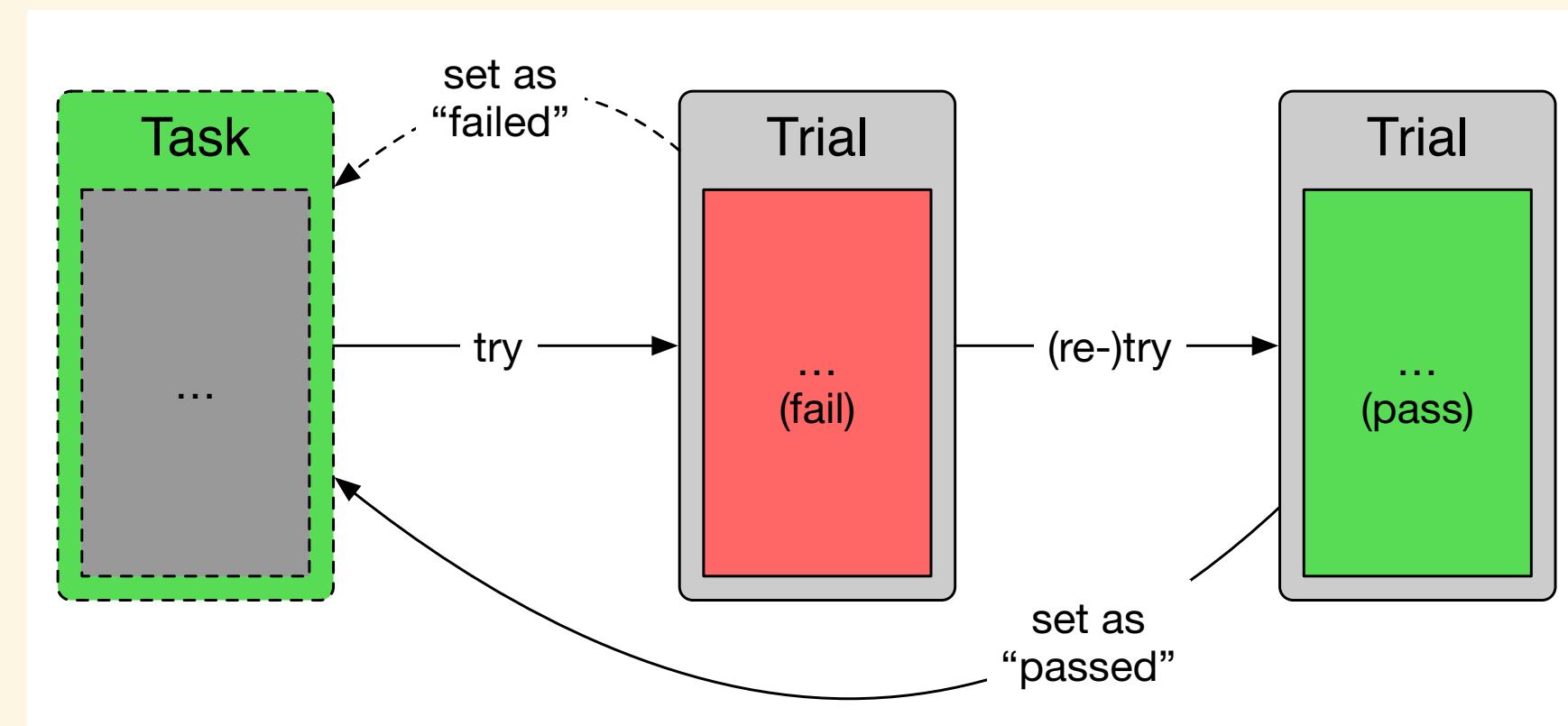
# JOB & TASKS



job: container and state aggregate for tasks

→ parallelization

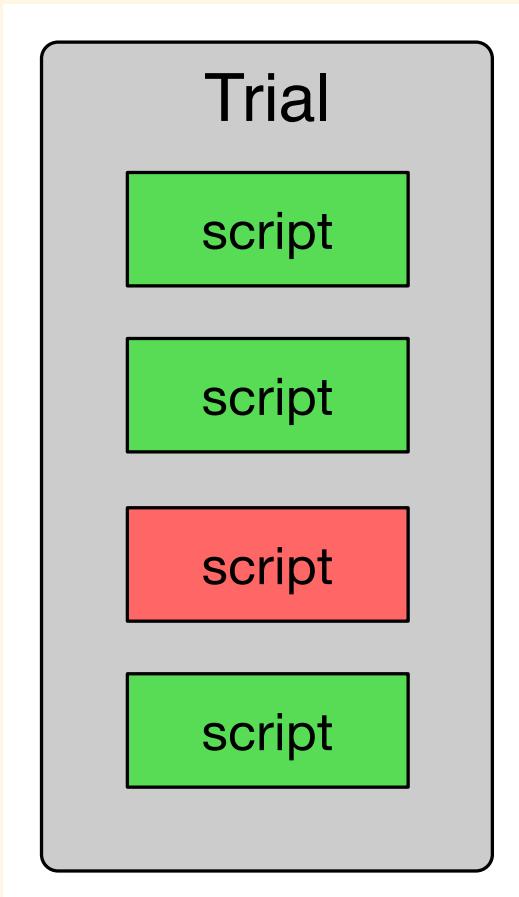
# TASKS & TRIALS



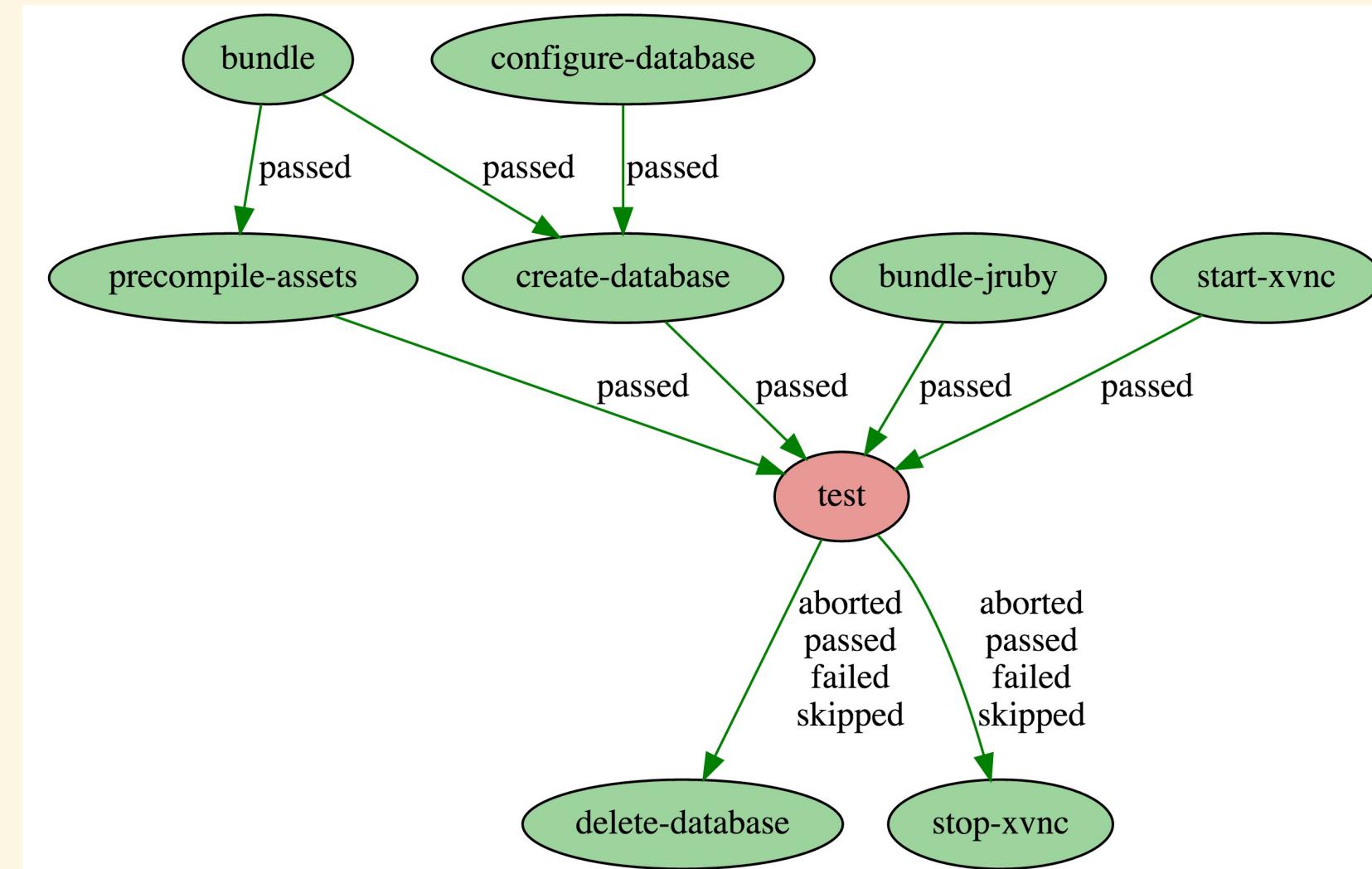
- **blueprint**
- container and state aggregate for trials  
→ resilience

# TRIAL & SCRIPTS

- actual unit of execution
- executed in the **same context**
- depend on each other

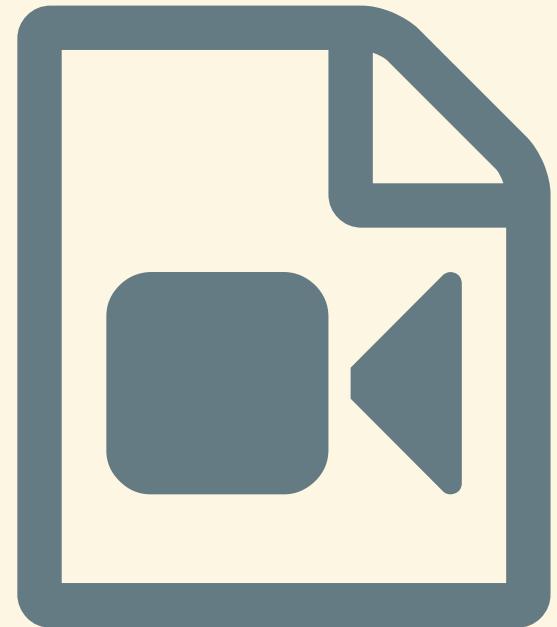


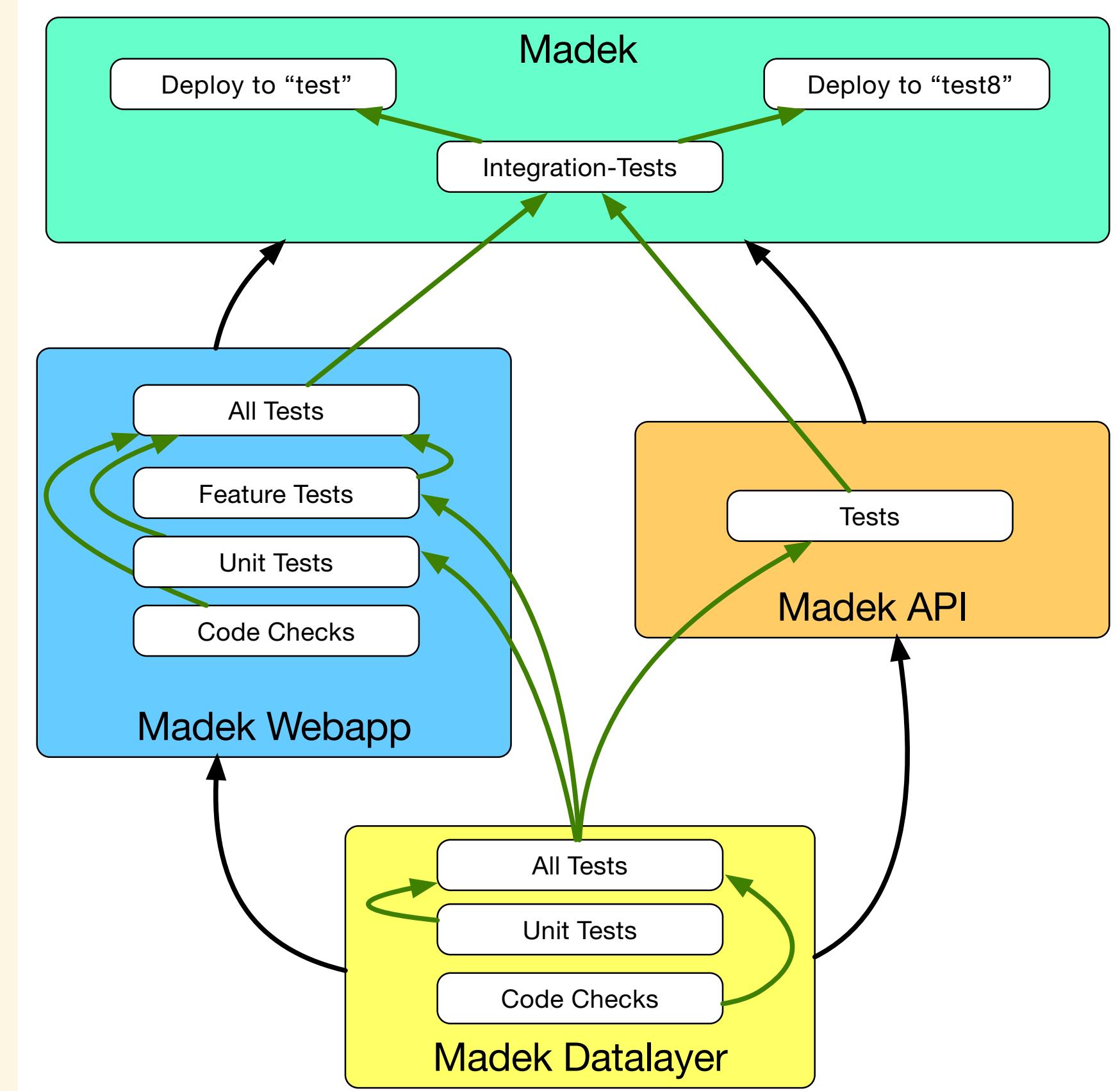
# SCRIPT DEPENDENCIES



- traditional CI: one "build"  $\Leftrightarrow$  one script
- more modern: one main script + before and after "hooks"
- Cider-CI: **scripts with dependencies**

# 4. DEMOS





The screenshot shows the Cider-CI web interface with the title "My Workspace". At the top, there are input fields for "Repository name 1, ...", "Branch name 1, ...", and "Git-ref (tree-id, commit-id)". Below this is a search bar with "Text search (subject, author, committer, ...)" and buttons for "Reset" and "Filter". A large banner in the center reads "Workflow and Job Dependencies Demo". The main content area displays a list of recent commits:

Commit	Author	Time Ago	Description	Job	Status	Run ...
4ffd50	Leihs	2 minutes ago	hide plus icon	fs/procurement	0	Run ...
ecea56	Madek Webapp	8 minutes ago	Perform fragment caching in dev mode ...	ts_madek-v3	0	Run ...
2e2339	Cider-Cl	41 minutes ago	Fix UI caching in trials#show	tmp	0	Run ...
2cc8f5	Cider-Cl User-Int...	42 minutes ago	Fix caching in trials#show and depend...	tmp	0	Run ...
03e359	Cider-Cl Integrat...	10 hours ago	Adjust to normalized scripts	tmp	0	Run ...

```
thomas@nx-18122:~/...madek_v3/webapp$ git la -2
* 6509e57 6e28620 (HEAD -> ts_madek-v3) Remove `pending` declaration from JS Devtools test Thomas Schank, 21 seconds ago, committed 21 seconds ago
* ecea561 30d1f5e (github/ts_madek-v3) Perform fragment caching in dev mode when env var RAILS_CACHE is set Thomas Schank, 8 minutes ago, committed 8 minutes ago

thomas@nx-18122:~/...madek_v3/webapp$
```



Session: madek\_v3.webapp 1 1      1:shell\* 2:rails\_server 3:rlwrap 4:vim#- 7:test 8:datalayer      09 Nov 08:31

ScreenFlow File Edit Mark Insert Font Actions View Window Help

Mo. 21:36 DrTom

Cider-CI

ci.zdhk.ch/cider-ci/ui/workspace/jobs/9056ff32-3f14-4b4f-8ab6-92ea91935405

Cider-CI Cardano 3.8.0-21 Workspace API Browser Documentation Administration drtom

Home / Workspace / Job

Analytics Specification

## Job Integration-Tests

passed created 13 hours ago

This job runs the top level integration tests. It must be run from the master project.

Commit

5a4326 546a67 13 hours ago Madek

Upgrade webpack: remove pending declar... 0 master  
0 ts\_master  
1 mfa\_master Run ...

## Tasks

All substr OR substr 12 Per page Reset Filter

spec/features/basic_spec.rb	Retry
spec/features/media-file_up-and-download_spec.rb	Retry
spec/features/traverse-webapp2api-via-session_spec.rb	Retry
spec/unit/datalayer-consistency_spec.rb	Retry

# **5. ADDENDUM**



# MANAGING FALSE POSITIVES

Retrying randomized tests can hide problems!

"Generative Testing" e.g.

## SOLUTION:

- reproducibility by initializing the pseudo random generator (we use the `tree_id` e.g.)
- statistics

# GIT SCM AND GIT ONLY

- don't compromise
- can't support everything with reasonable effort

# SECURITY & TRUST

- Cider-CI server itself never runs any code from projects
- "blessed" executors only accept trials for a particular project (repository)

# MATCHING TRIALS TO EXECUTORS

- task specifies **required traits**, e.g: [bash, ruby-2.2 ]
- executors advertise **available traits**, e.g. [bash, maven, postgresql, ruby-2.1, ruby-2.2, ...]

Cider-CI will determine a suitable executor.

# DEPLOYMENT

- Ansible
- Cider-CI deploy project, SCM managed, reproducible

# CIDER-CI IS AN EXPERT SYSTEM

it is about **making the hard possible**, and not not about  
making the simple easy\*

- for professionals
- no compromises
- steep learning curve
- high rewards

→ swiss army knife for devops

\*see "Simple Made Easy" by Rich Hickey

# CONCLUSION

- A false negative outcome becomes likely with an increasing number of tests.
- The problem must be solved by retrying single tests.
- Consider to build your own pipeline.
- Try Cider-CI, open source, installs with two commands:  
<http://docs.cider-ci.info/introduction/quick-start/>



*Thank You!*