Analyse von Composer-Paketabhängigkeiten in PHP-Projekten

Eine Projektarbeit von Konstantin Tieber

webfactory GmbH

3. Juli 2018

Agenda

- Projektvorstellung
- Analyse
- Entwurf
- Implementierung
- Qualitätssicherung
- Dokumentation
- Fazit

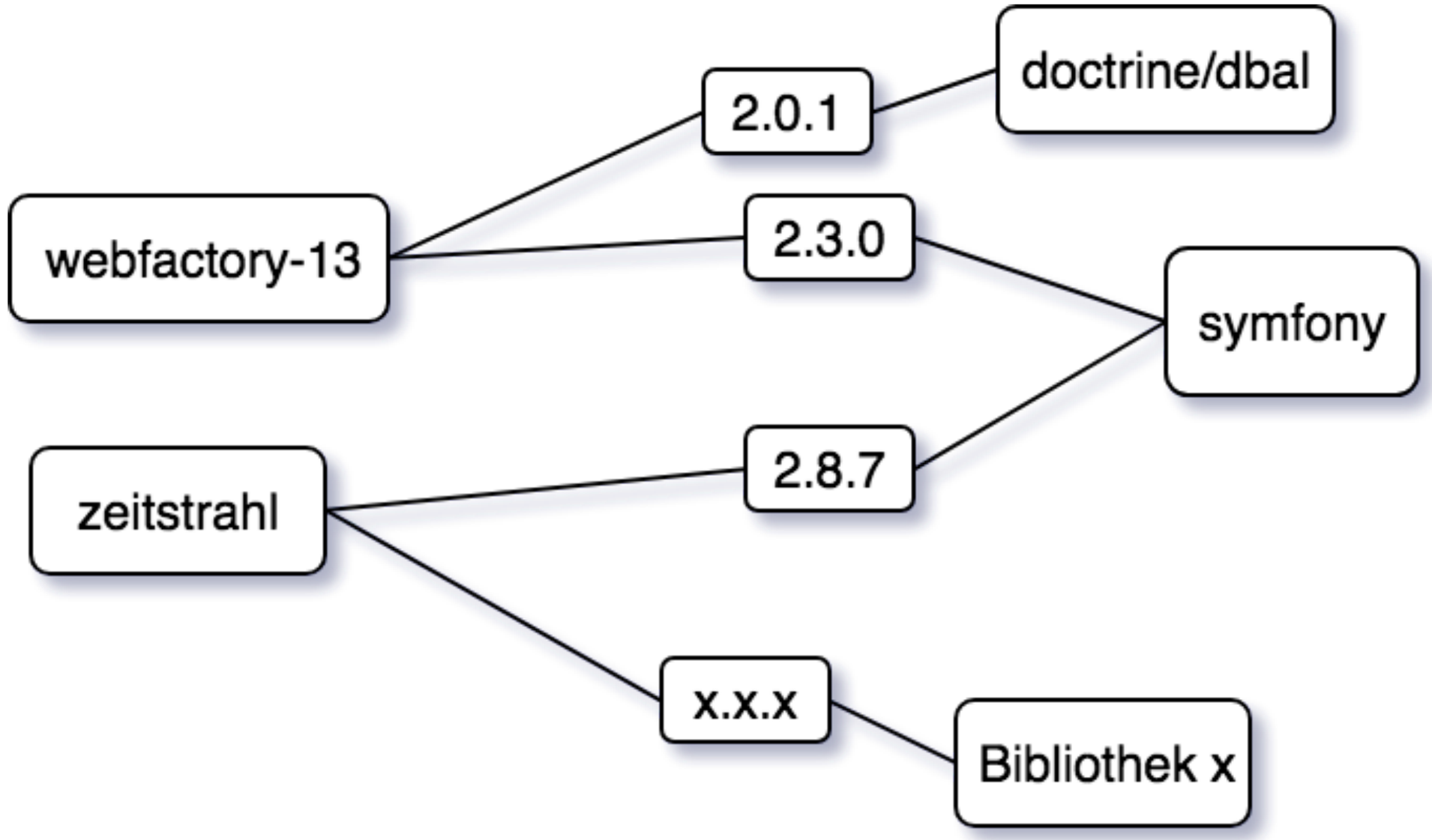
Agenda

- **Projektvorstellung**
- Analyse
- Entwurf
- Implementierung
- Qualitätssicherung
- Dokumentation
- Fazit

Projektumfeld

Projekte

Pakete



Baton

Composer Dependency Analytics Tool

symfony/symfony

<

2.2.1.0

Find projects

Matching Projects

Using symfony/symfony

Version v2.0.10
Used by: g4e-external-wfd-links

Version v2.0.18
Used by: zeitstrahl-jugendpolitik

Version v2.1.8
Used by: webfactory-13-20810, webfactory-13-raster, odl-bonn, wfdynamic-info-website

Projektziel

Ein einfach zu bedienendes Tool soll jederzeit einen aktuellen Überblick darüber geben,

… welche Abhängigkeiten ein Projekt hat.

… welche Projekte eine bestimmte Version eines Pakets einsetzen.

Agenda

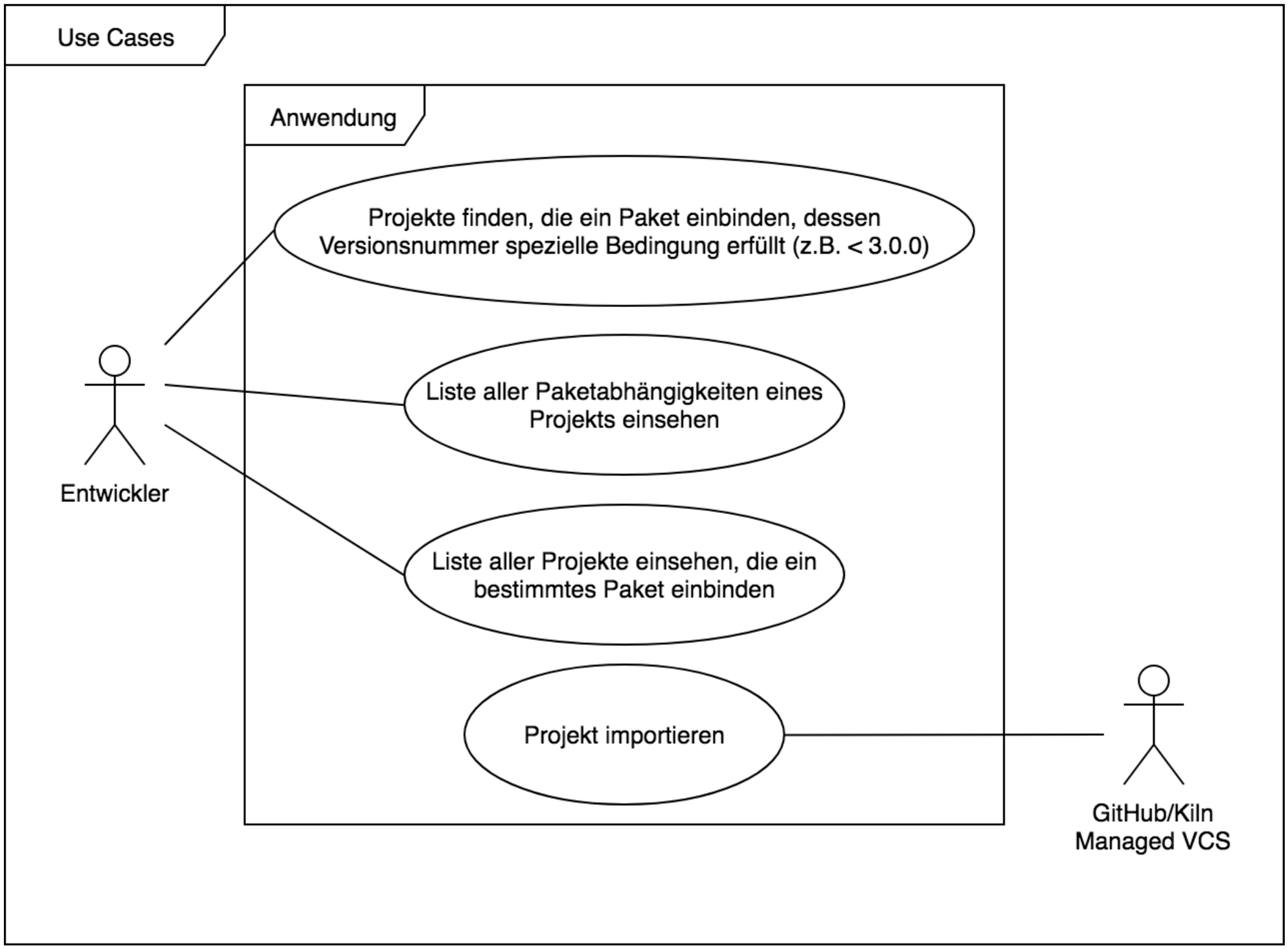
- Projektvorstellung
- **Analyse**
- Entwurf
- Implementierung
- Qualitätssicherung
- Dokumentation
- Fazit

Wirtschaftlichkeit

- Make or Buy: Konkurrenzprodukt Gemnasium

- Amortisationsdauer

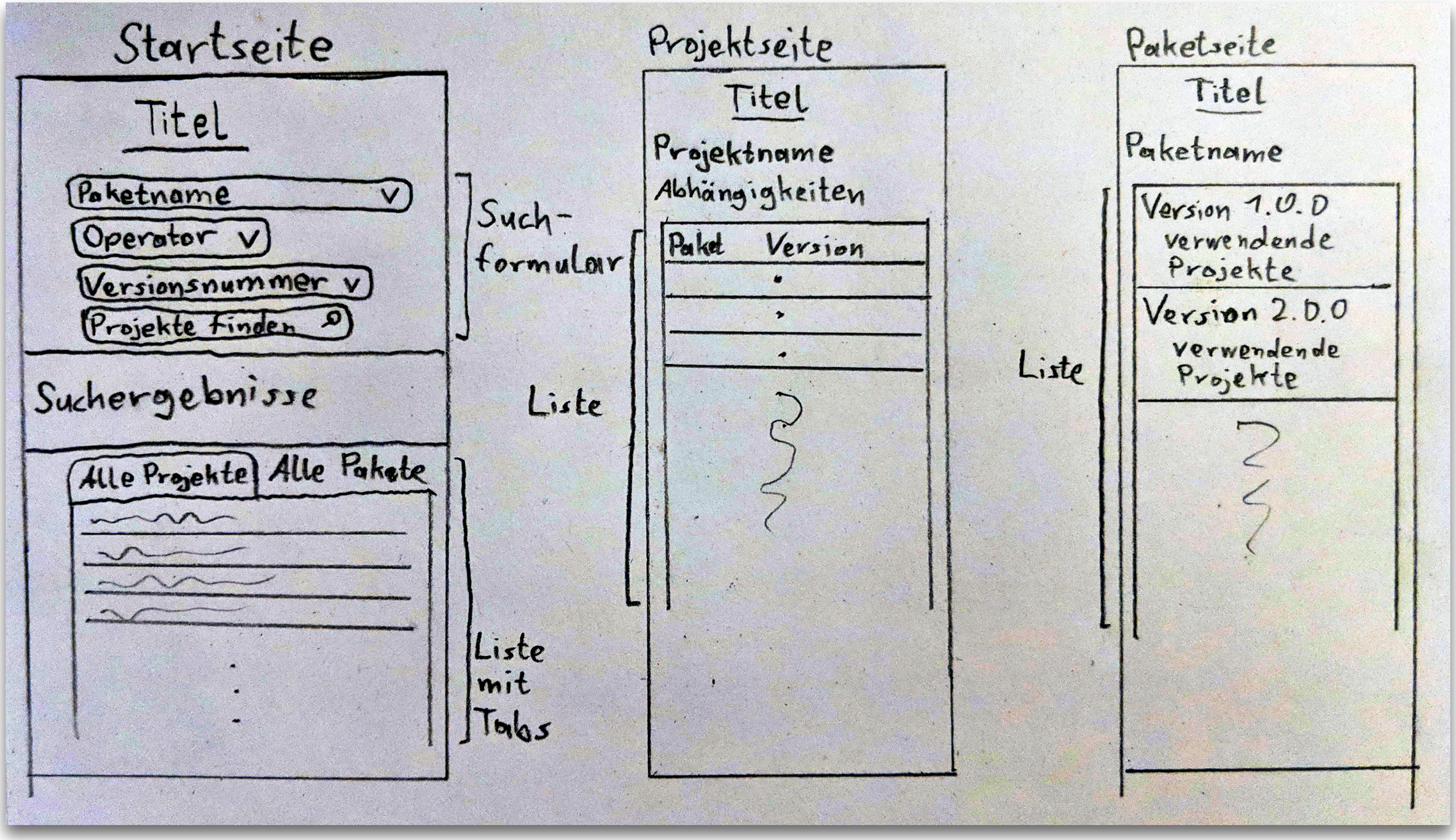
Anwendungsfälle



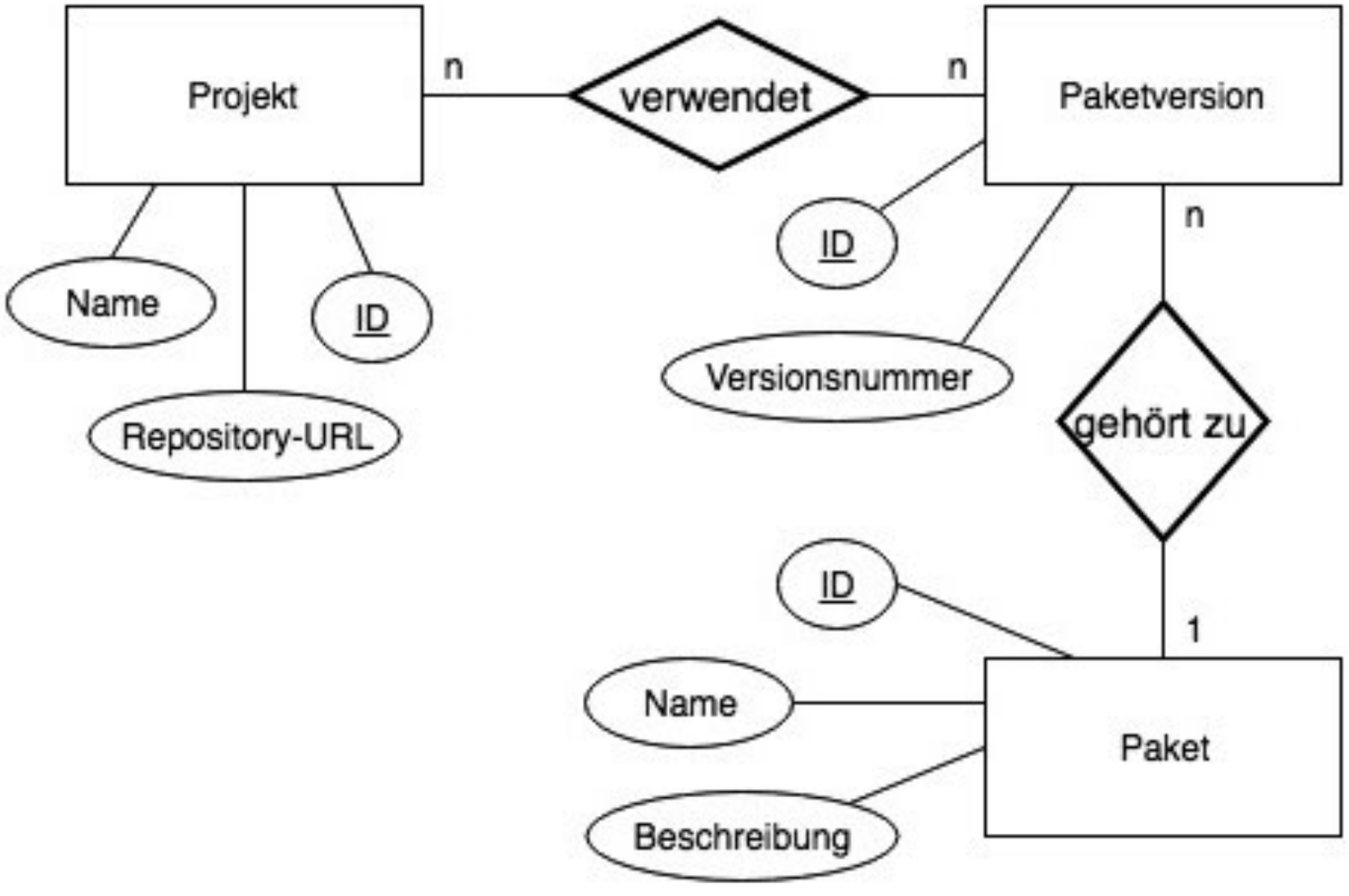
Agenda

- Projektvorstellung
- Analyse
- **Entwurf**
- Implementierung
- Qualitätssicherung
- Dokumentation
- Fazit

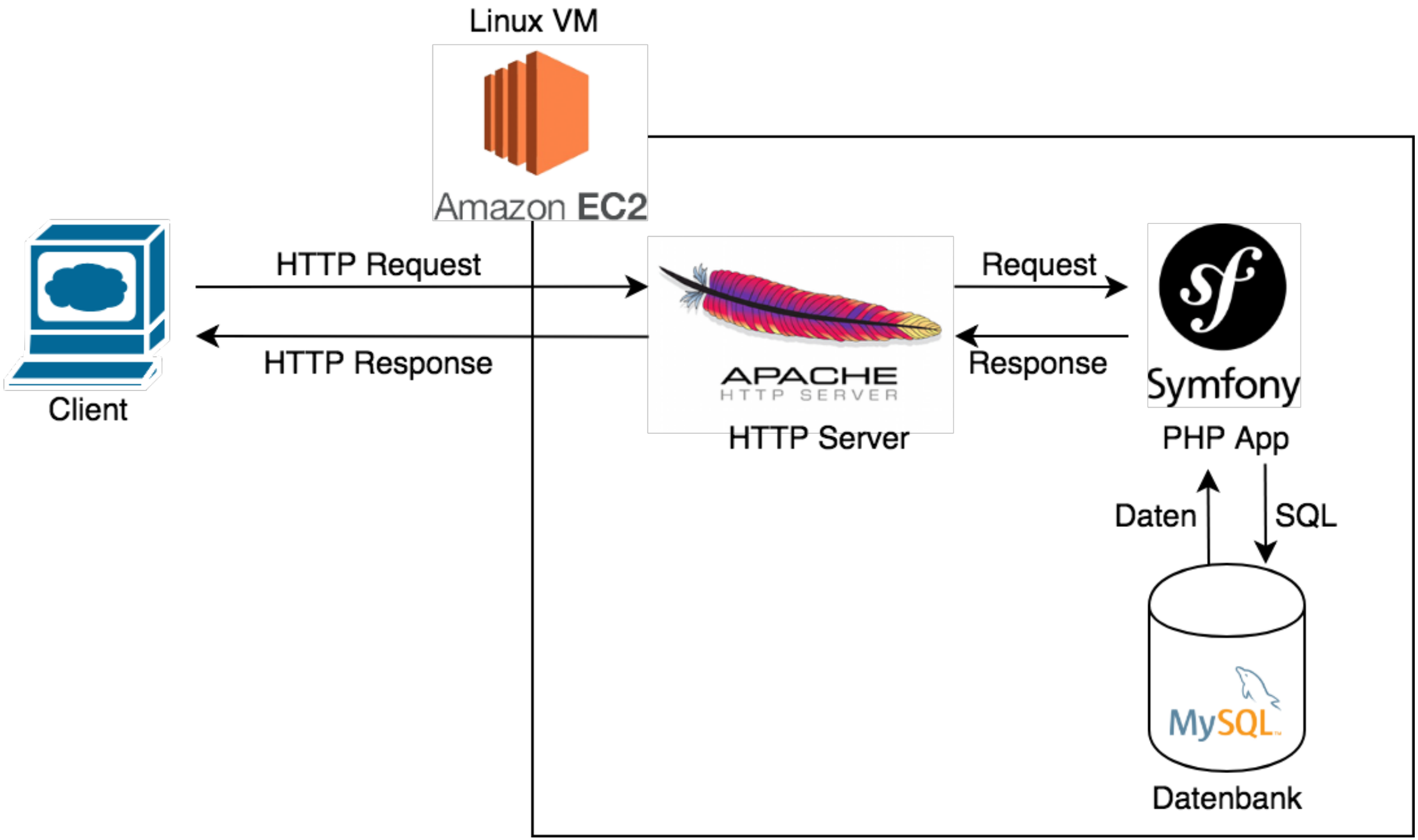
UI Mockups


Startseite
Titel
Paketname
Operator
Versionsnummer
Projekte finden
Suchergebnisse
Alle Projekte
Alle Pakete
Such-formular
Liste
Liste mit Tabs
Projektseite
Titel
Projektname
Abhängigkeiten
Paket
Version
Liste
Paketseite
Titel
Paketname
Version 1.0.0 verwendende Projekte
Version 2.0.0 verwendende Projekte
Liste

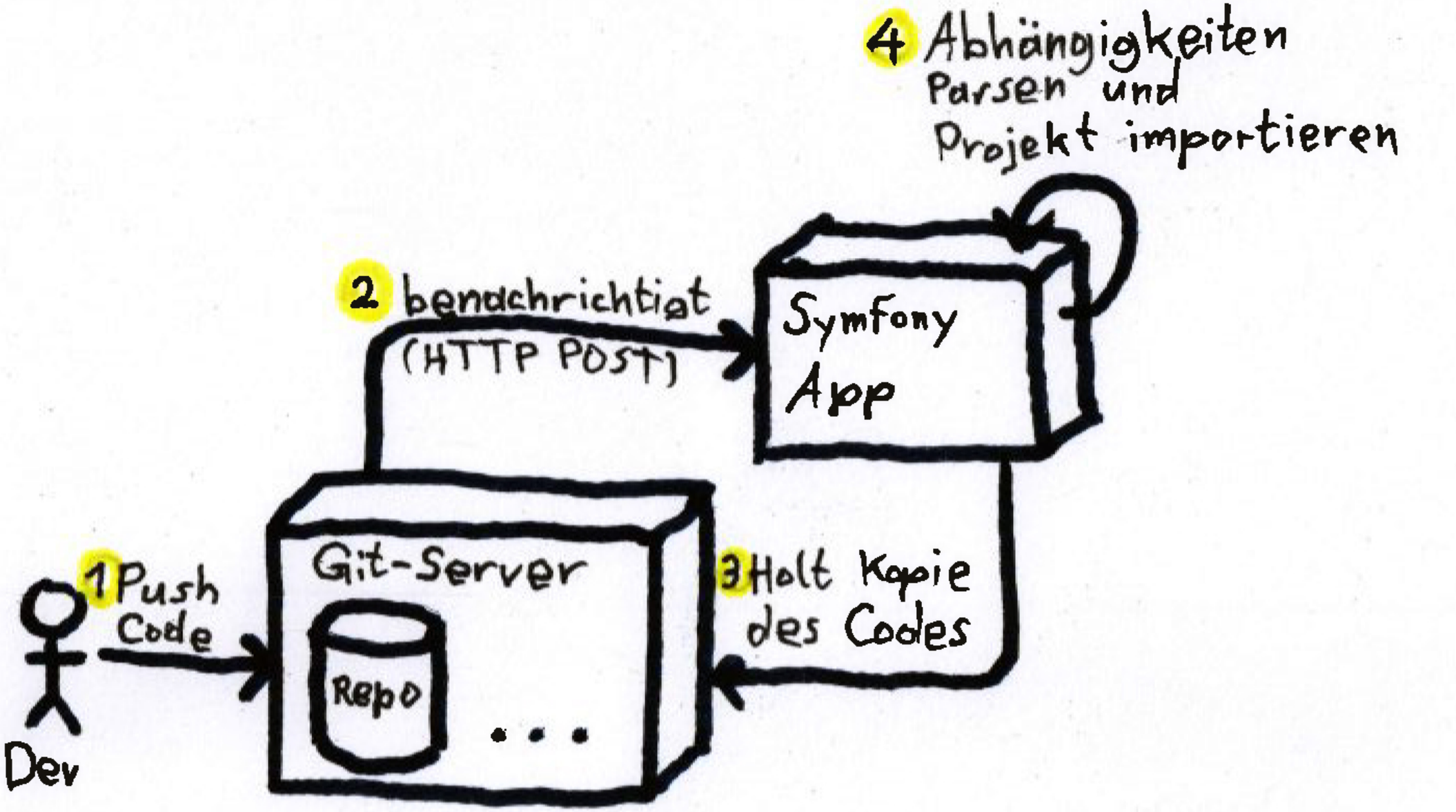
Datenmodell



Architektur



Projektimport

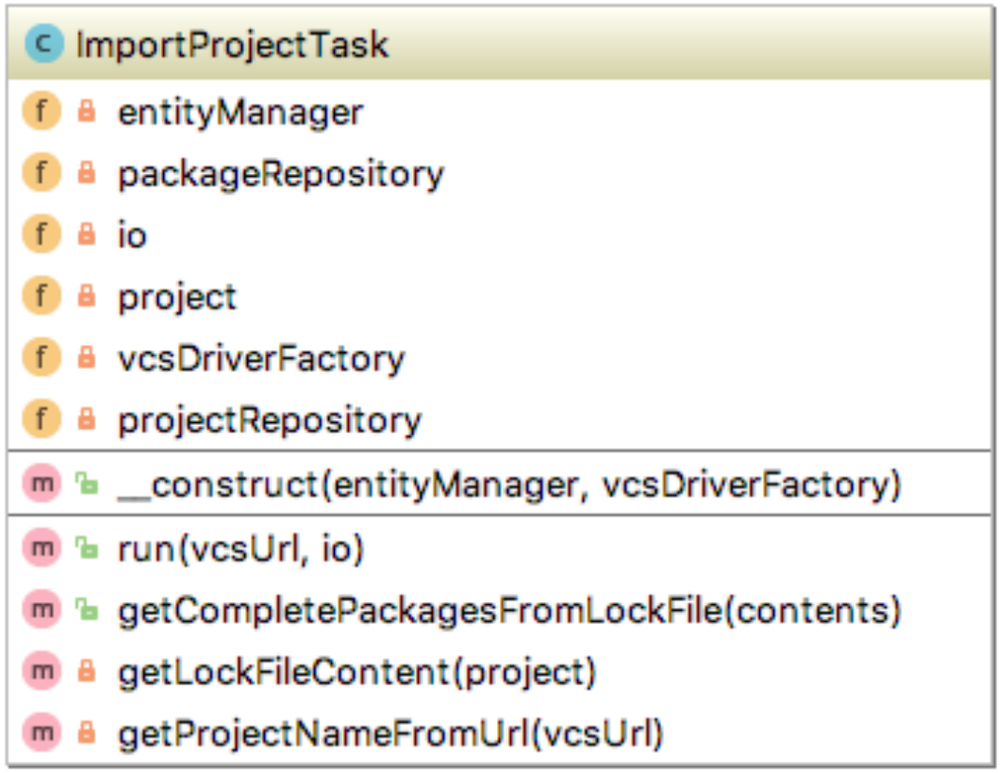


Agenda

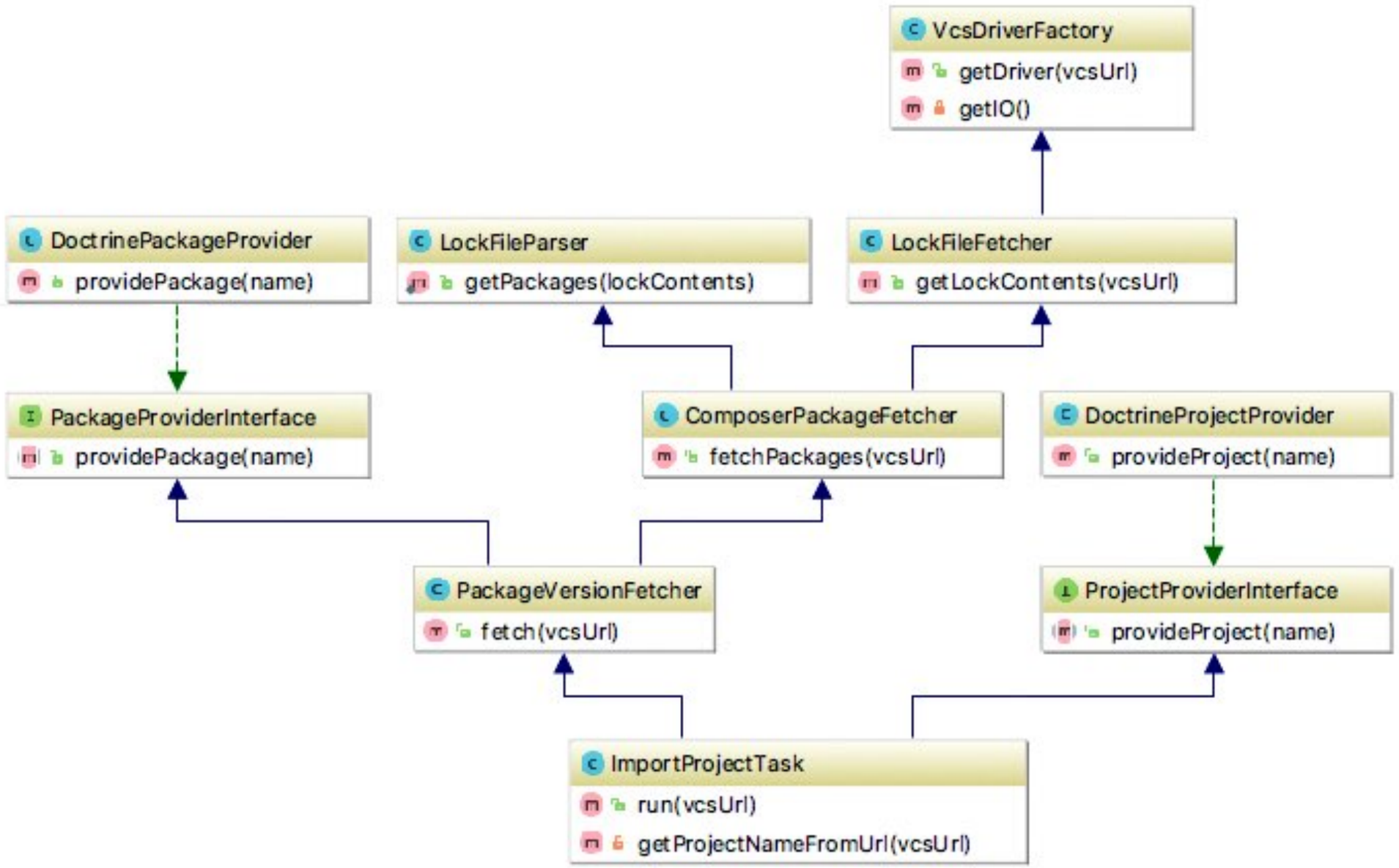
- Projektvorstellung
- Analyse
- Entwurf
- **Implementierung** bin/console app:project import <url>
- Qualitätssicherung
- Dokumentation
- Fazit

```php
class ImportProjectTask
{
    /* ... */
    public function run($vcsUrl, IOInterface $io = null)
    {
        $io !== null ? $this->io = $io : $this->io = new NullIO();
        $this->project = $this->projectRepository->findOneBy(['name' => $this->getProjectNameFromUrl($vcsUrl)]);
        if ($this->project !== null) {
            $this->io->write(["Exisitng repository found, removing current usages"]);
            foreach ($this->project->getUsages() as $usage) {
                $this->project->removeUsage($usage);
            }
        } else {
            $this->io->write(["Importing new project"]);
            $this->project = new Project($this->getProjectNameFromUrl($vcsUrl), $vcsUrl);
        }
        $lockFileContents = $this->getLockFileContent($this->project);
        if($lockFileContents === null) {
            $this->io->writeError('The project '.$this->project->getName().' does not contain a composer.lock file.');
            return false;
        }
        $completeComposerPackages = $this->getCompletePackagesFromLockFile($lockFileContents);
        $this->io->write(["Adding Usages:"]);
        foreach ($completeComposerPackages->getPackages() as $composerPackage) {
            $package = $this->packageRepository->findOneBy(['name' => $composerPackage->getName()]);
            if ($package === null) {
                $package = new Package($composerPackage->getName());
            }
            $this->project->addUsage(new PackageVersion($composerPackage->getPrettyVersion(), $package));
            $this->io->write($composerPackage->getName().", ");
        }
        $this->entityManager->persist($this->project);
        $this->entityManager->flush();
        return true;
    }
    /* ... */
}
```

Eine Klasse für alles



Refactoring



```php
class ImportProjectTask
{
    /* ... */
    public function run($vcsUrl)
    {
        $project = $this->projectProvider->provideProject($this->getProjectNameFromUrl($vcsUrl));
        $project->setVcsUrl($vcsUrl);

        try {
            $project->setUsedPackageVersions($this->packageVersionFetcher->fetch($vcsUrl));
        } /* Error Handling ... */
    }
    /* ... */
}
```

Kapselung

webforum/shared-styles

<=

3.1.0.0

Find projects

1. Paket auswählen
2. VersionConstraint konfigurieren

Matching Projects

Using webforum/shared-styles

Version 2.0.15
Used by: youthreporter

Version 2.1.0
Used by: europeers, yia-lab

Version 3.1.0
Used by: jugendfuereuropa, jugend-in-aktion

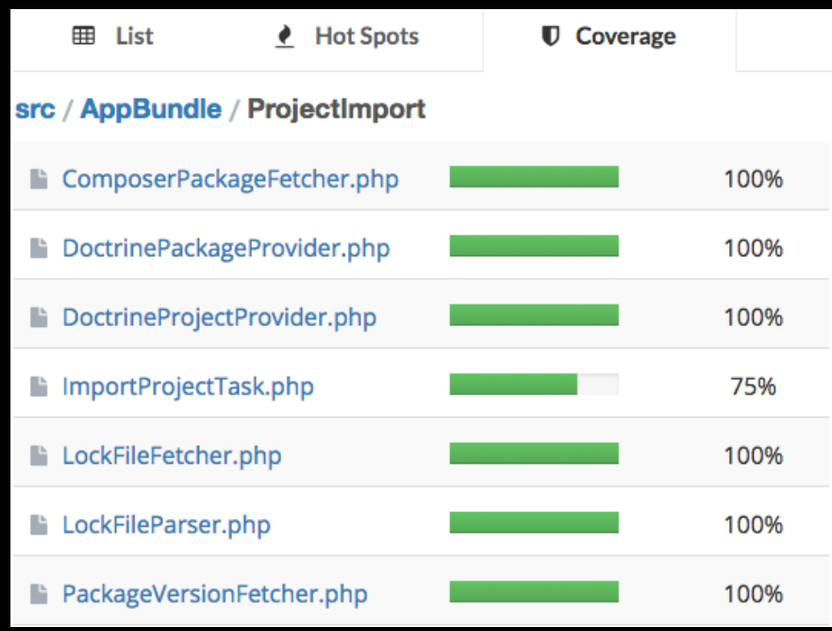
```php
class Package
{
    /* ... */
    public function getMatchingVersions(VersionConstraint $versionConstraint)
    {
        return $this->versions->filter(
            function($packageVersion) use ($versionConstraint) {
                return $versionConstraint->matches($packageVersion);
            }
        );
    }
    /* ... */
}
```

```php
class VersionConstraint
{
    /* ... */
    public function matches(PackageVersion $packageVersion)
    {
        switch($this->operator) {
            case 'all':
                return true;
                break;
            case '<':
                if ($packageVersion->getNormalizedVersion() < $this->normalizedVersionString) {
                  return true;
                }
                break;
            case '<=':
                if ($packageVersion->getNormalizedVersion() <= $this->normalizedVersionString) {
                  return true;
                }
                break;
            /* ... case '>', case '>=', case '==' */
        }
        return false;
    }
}
```

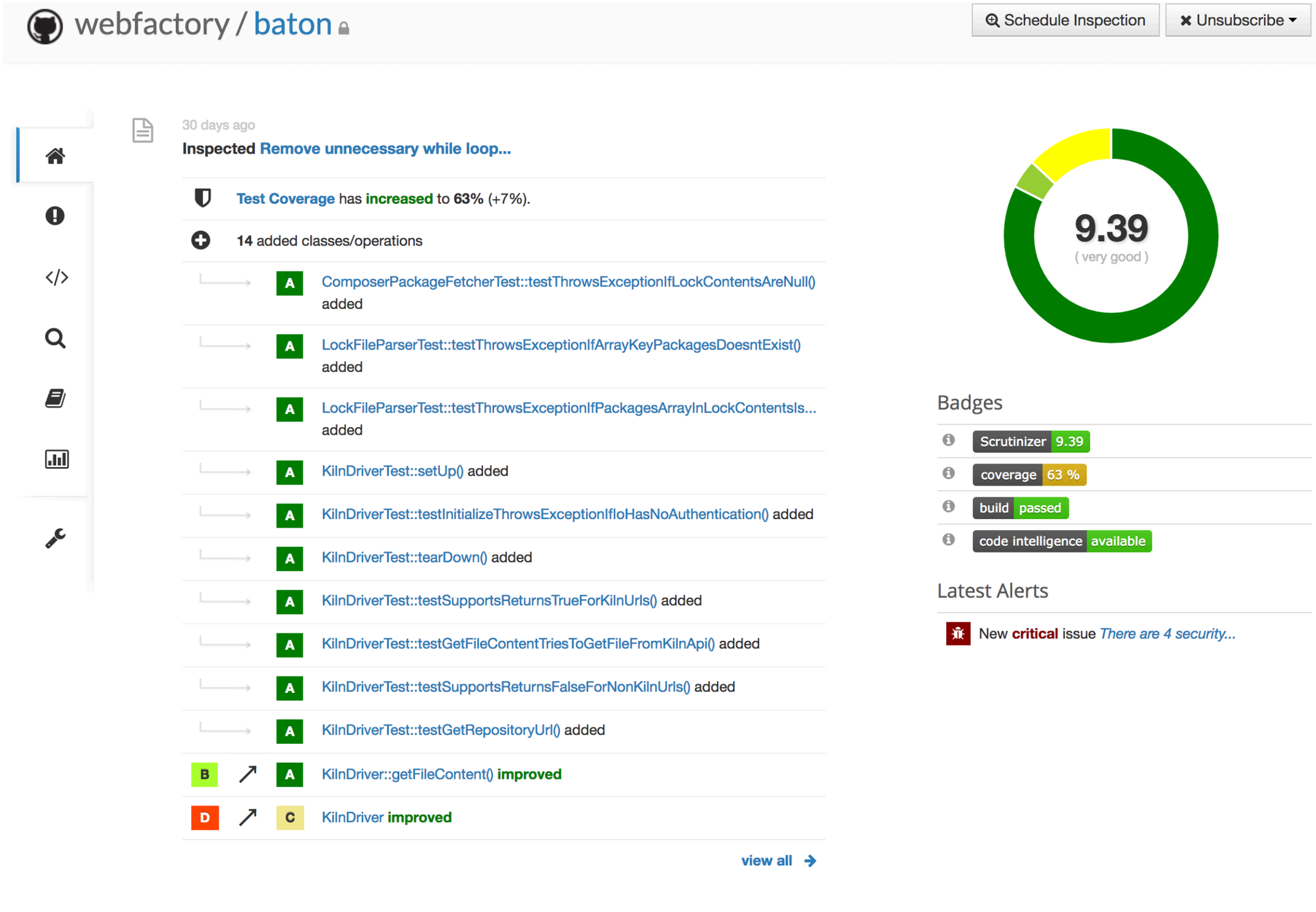
Agenda

- Projektvorstellung
- Analyse
- Entwurf
- Implementierung
- **Qualitätssicherung**
- Dokumentation
- Fazit

```php
class ComposerPackageFetcherTest extends \PHPUnit_Framework_TestCase
{
    public function testFetchPackagesReturnsArrayOfComposerPackages()
    {
        $composerPackageFetcher = new ComposerPackageFetcher(
            $this->getLockFileFetcherMock(file_get_contents(__DIR__ . '/composer_test.lock'))
        );

        $packages = $composerPackageFetcher->fetchPackages("https://foo.git");

        $this->assertInternalType('array', $packages);
        $this->assertInstanceOf(ComposerPackage::class, $packages[0]);
    }

    public function testThrowsExceptionIfLockContentsAreInvalid()
    {
        $composerPackageFetcher = new ComposerPackageFetcher(
            $this->getLockFileFetcherMock(null)
        );

        $this->setExpectedException(ProjectHasNoComposerPackageUsageInfoException::class);

        $composerPackageFetcher->fetchPackages("https://foo.git");
    }

    /** ... */
    private function getLockFileFetcherMock($lockContentsToReturn)
    {
        $projectProviderMock = $this->getMock(LockFileFetcher::class, [], [], '', false);
        $projectProviderMock->expects($this->once())
            ->method('getLockContents')
            ->willReturn($lockContentsToReturn);

        return $projectProviderMock;
    }
}
```



Continuous Integration



Agenda

- Projektvorstellung
- Analyse
- Entwurf
- Implementierung
- Qualitätssicherung
- **Dokumentation**
- Fazit

README.md

README.md

Baton

Baton is a Composer dependency analytics tool which helps you keep track of the Composer dependencies in your PHP projects.

Demo

Visit baton.test.webfactory.de to see Baton in action.

Installing / Getting started

To get the project up and running you simply need to run these commands:

```
phlough install
bin/console doctrine:database:create
bin/console doctrine:schema:create
```

```php
/**
 * Transparently provides Package object, either from the database if it exists or a new one.
 */
interface PackageProviderInterface
{
    /**
     * Provide Package object representing the Package with a given name.
     * If the Package is already known, return the exisitng instance.
     * Otherwise create a new nstance.
     *
     * @param string $name
     * @return Package
     */
    public function providePackage($name);
}
```

Agenda

- Projektvorstellung
- Analyse
- Entwurf
- Implementierung
- Qualitätssicherung
- Dokumentation
- **Fazit**

Danke für die Aufmerksamkeit!
Fragen?