(TYPO3)
FLOW

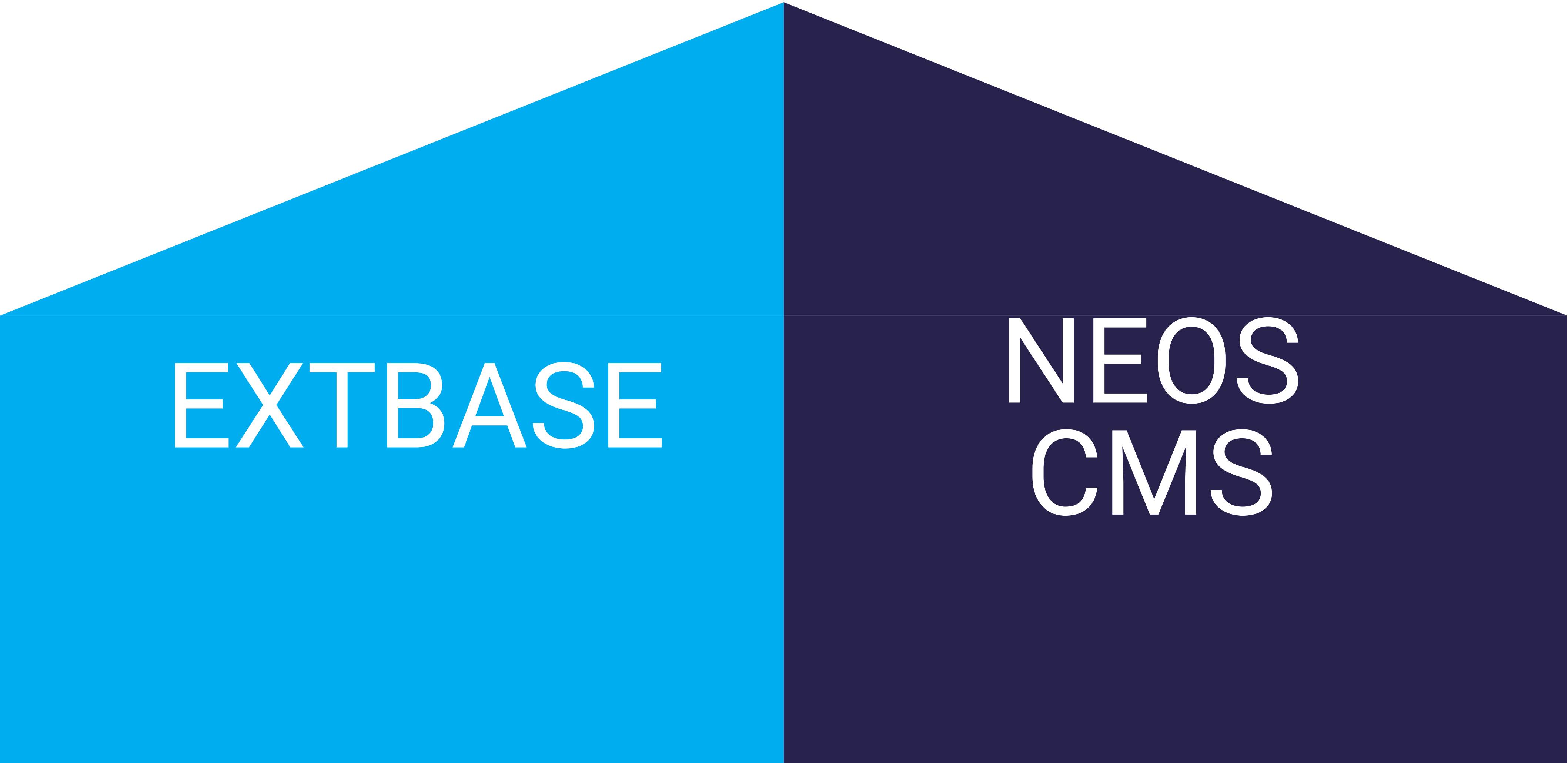
php
Laravel
Symfony
CakePHP
FLOW
yii
ZEND FRAMEWORK
phalcon
CodeIgniter

php
Laravel
Symfony
CakePHP
FLOW
yii
ZEND FRAMEWORK
phalcon
CodeIgniter

php
?
Laravel
Symfony
CakePHP
FLOW
yii
ZEND FRAMEWORK
phalcon
CodeIgniter

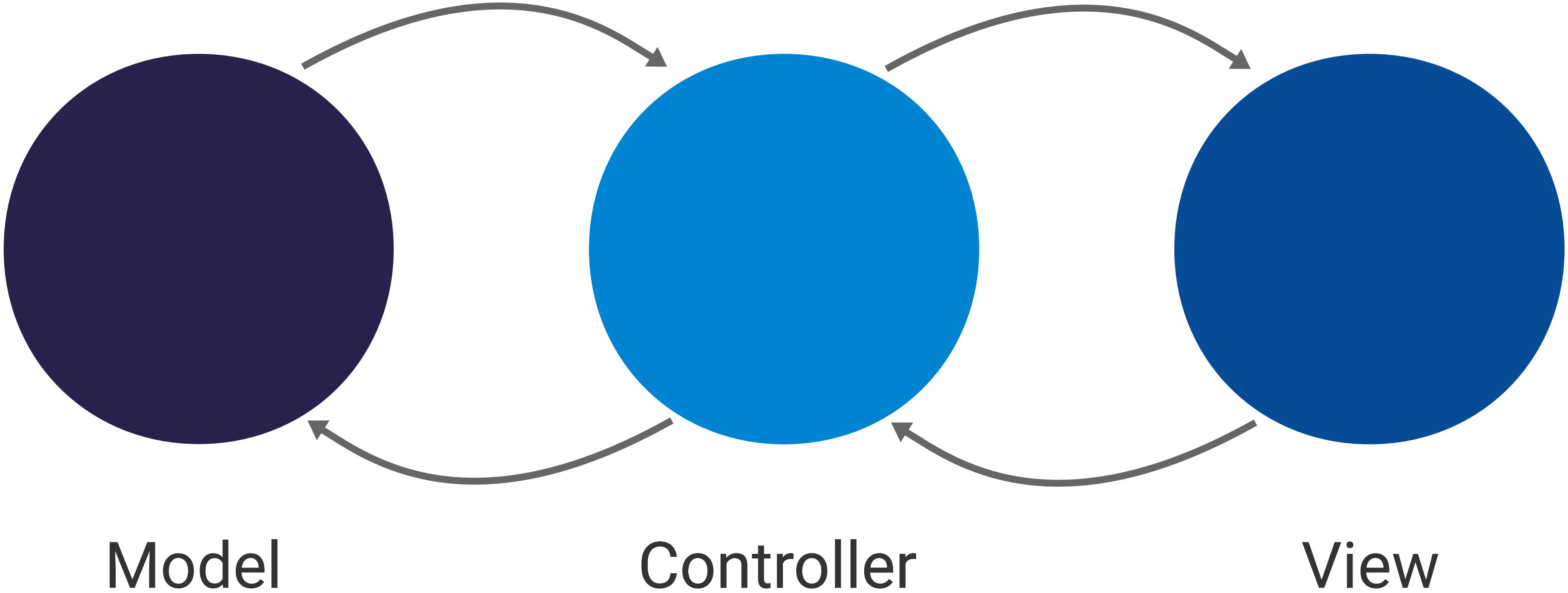
php
Laravel
Symfony
CakePHP
FLOW
yii
ZEND FRAMEWORK
phalcon
CodeIgniter

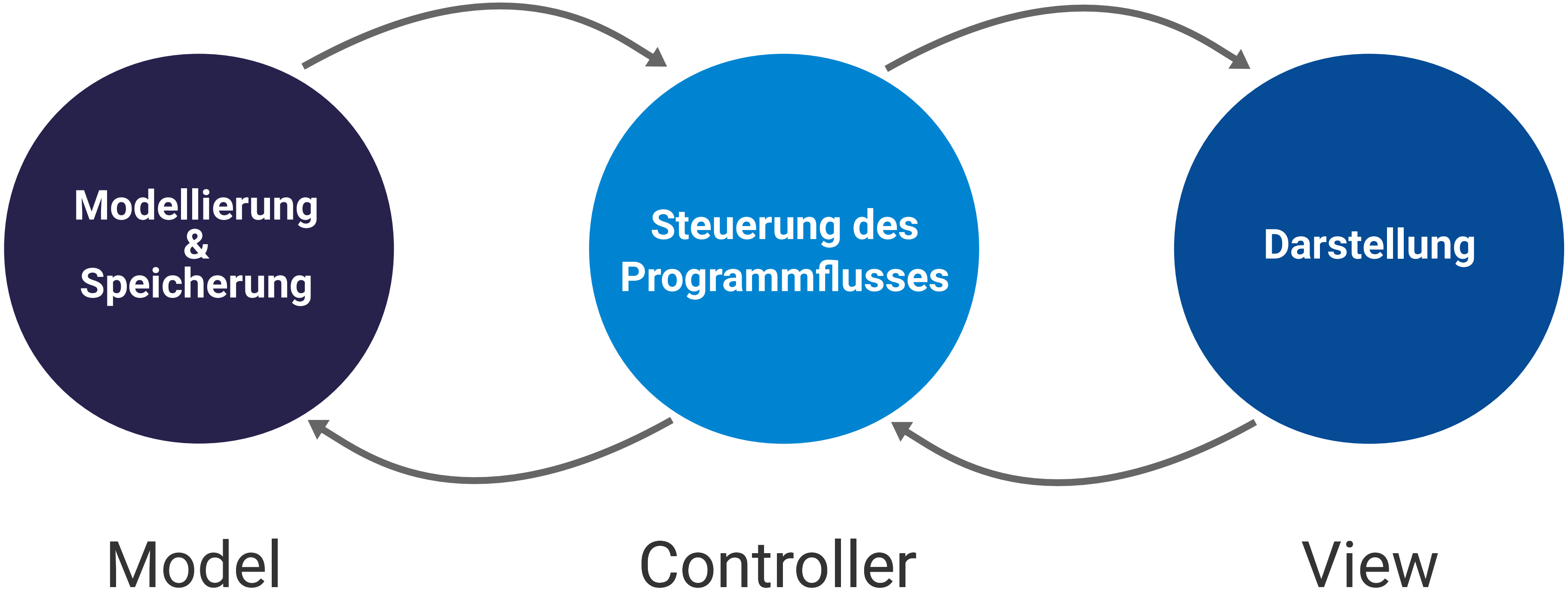
FLOW



EXTBASE

NEOS
CMS


Model
Controller
View



Modellierung & Speicherung

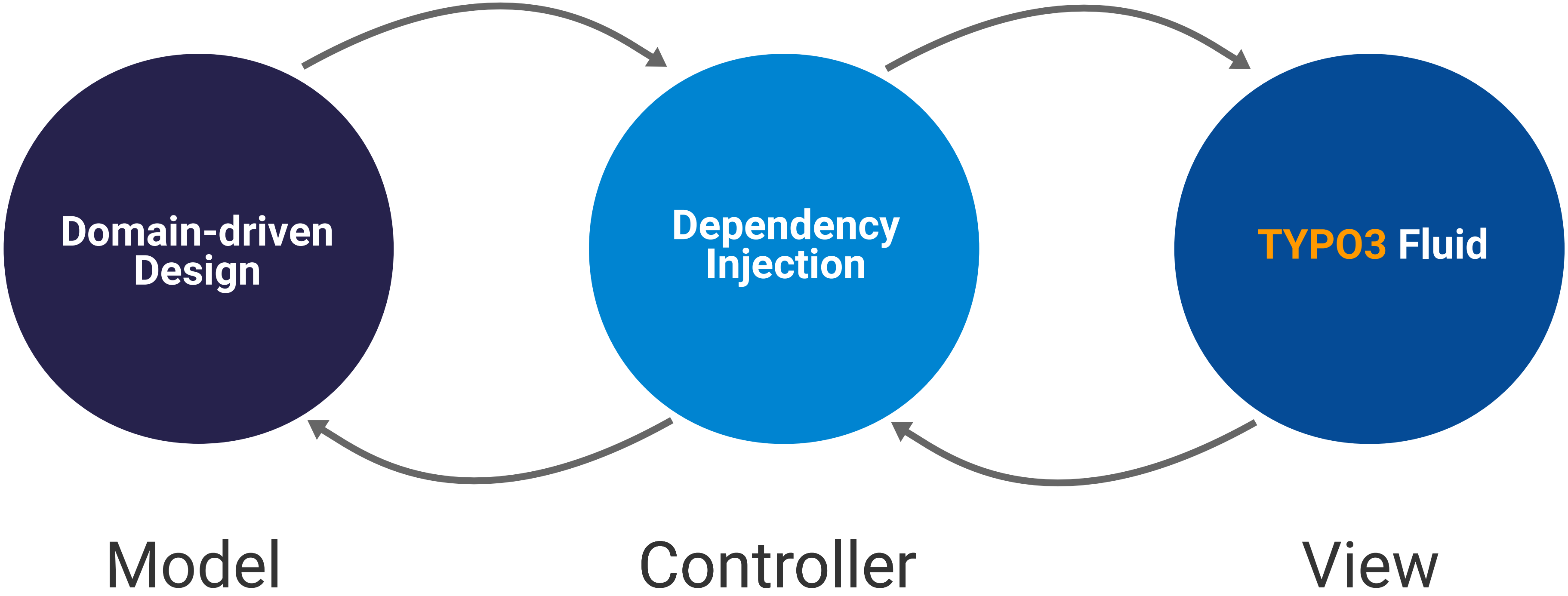
Steuerung des Programmflusses

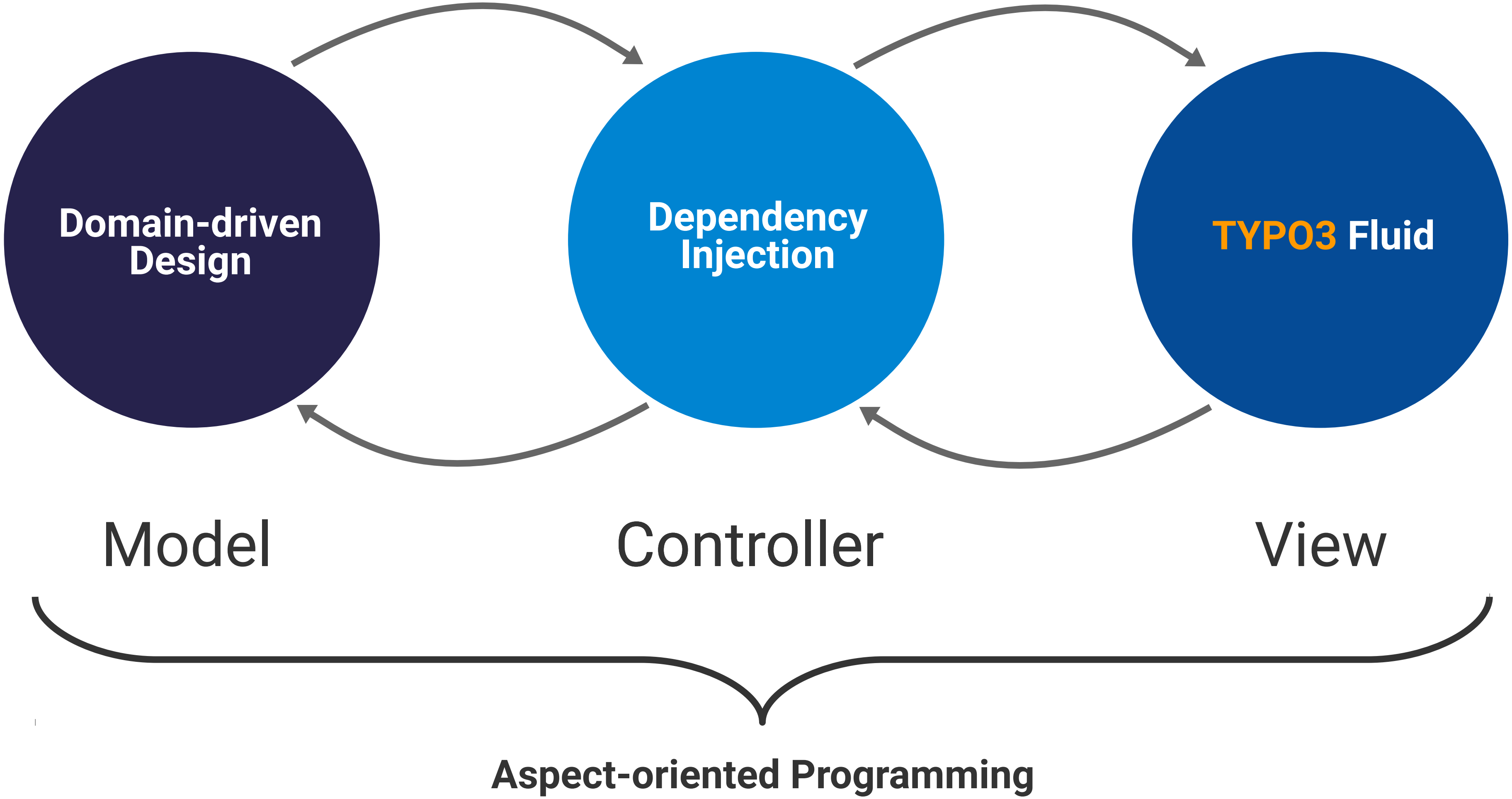
Darstellung

Model

Controller

View


Domain-driven Design
Dependency Injection
TYPO3 Fluid
Model
Controller
View



Domain-driven Design

Dependency Injection

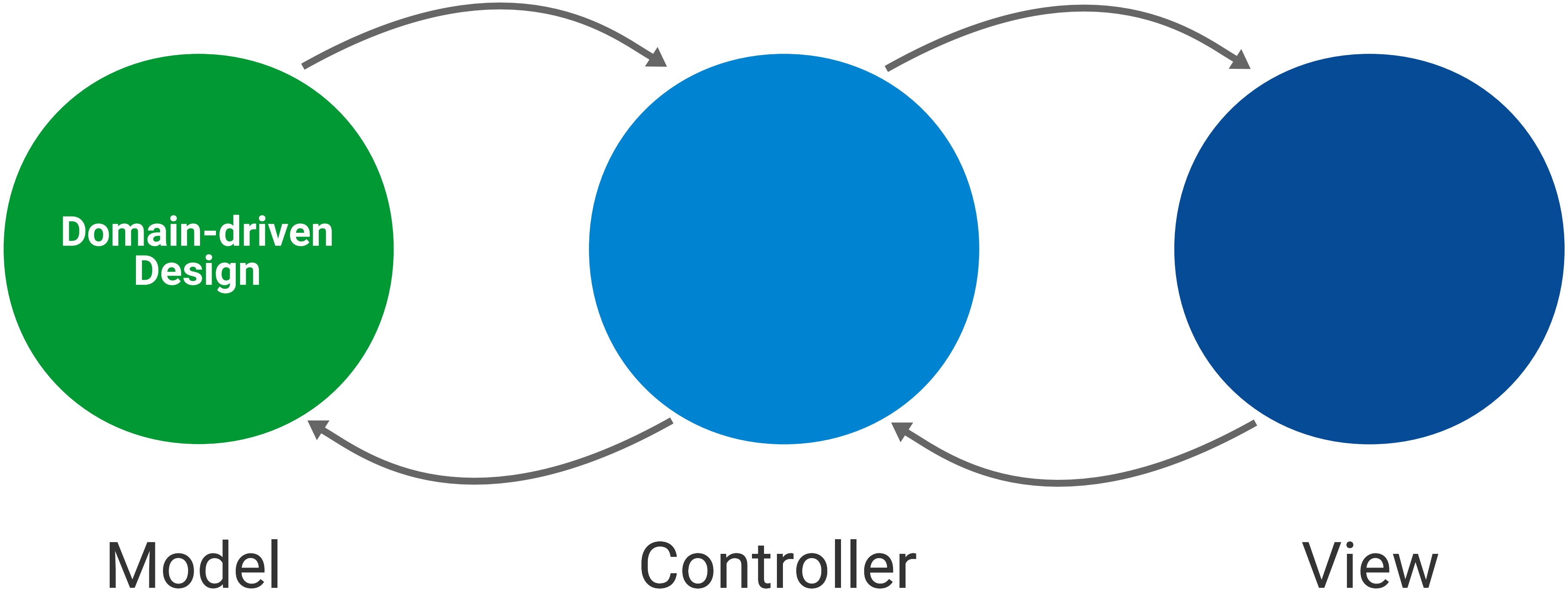
TYPO3 Fluid

Model

Controller

View

Aspect-oriented Programming

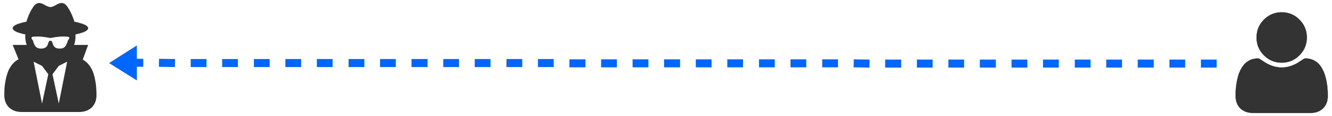


Domain-driven Design

Model

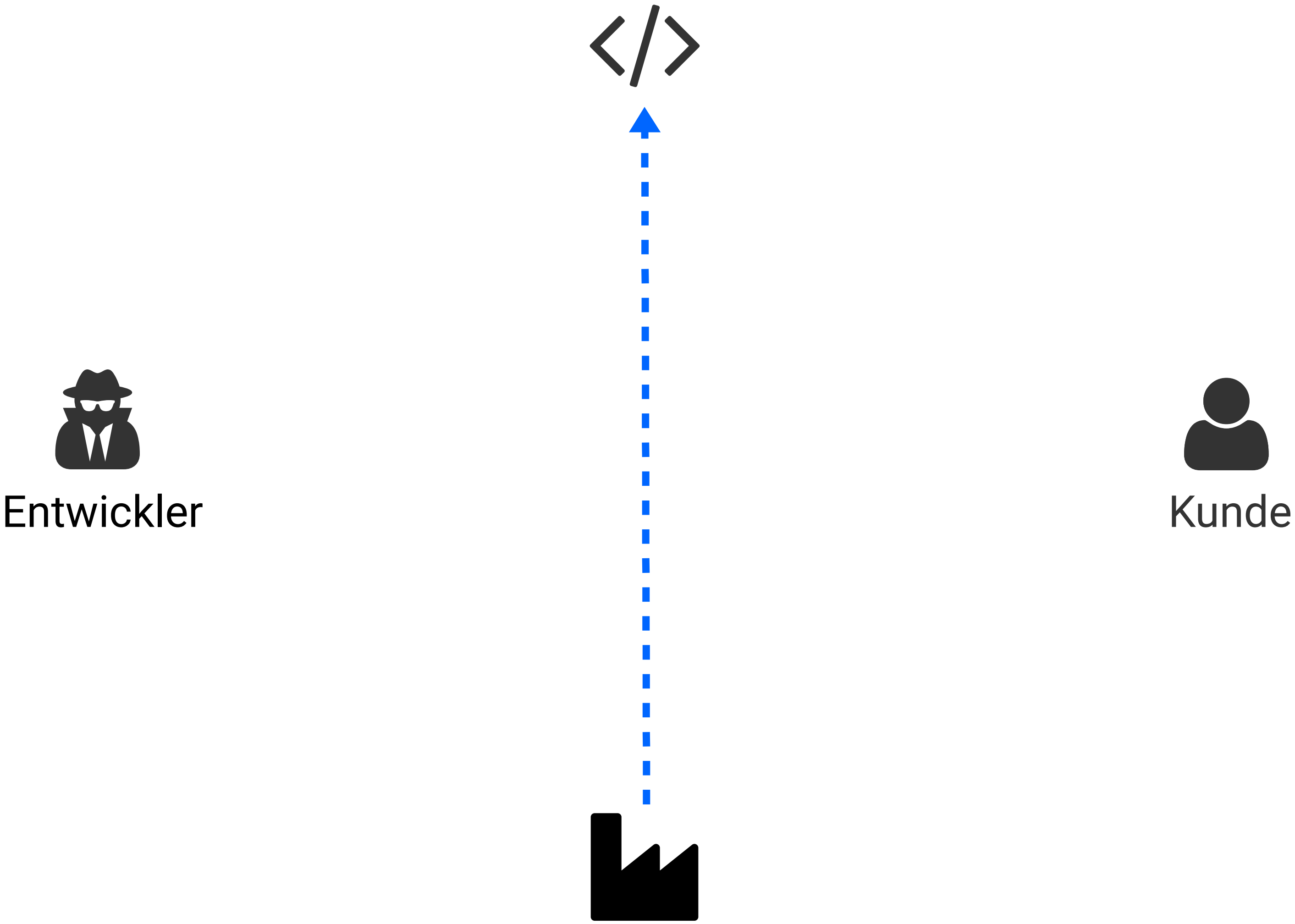
Controller

View



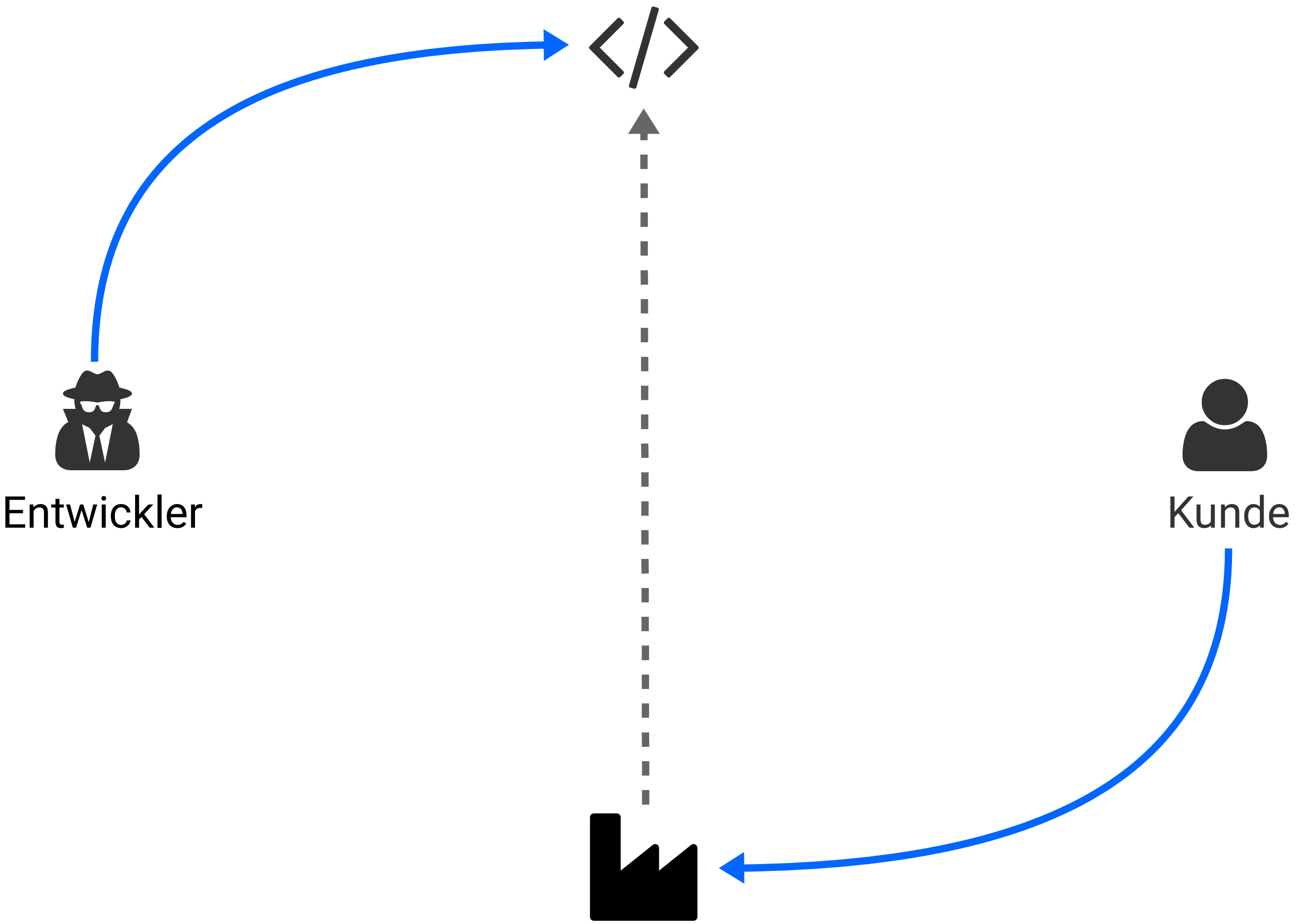
Entwickler

Kunde



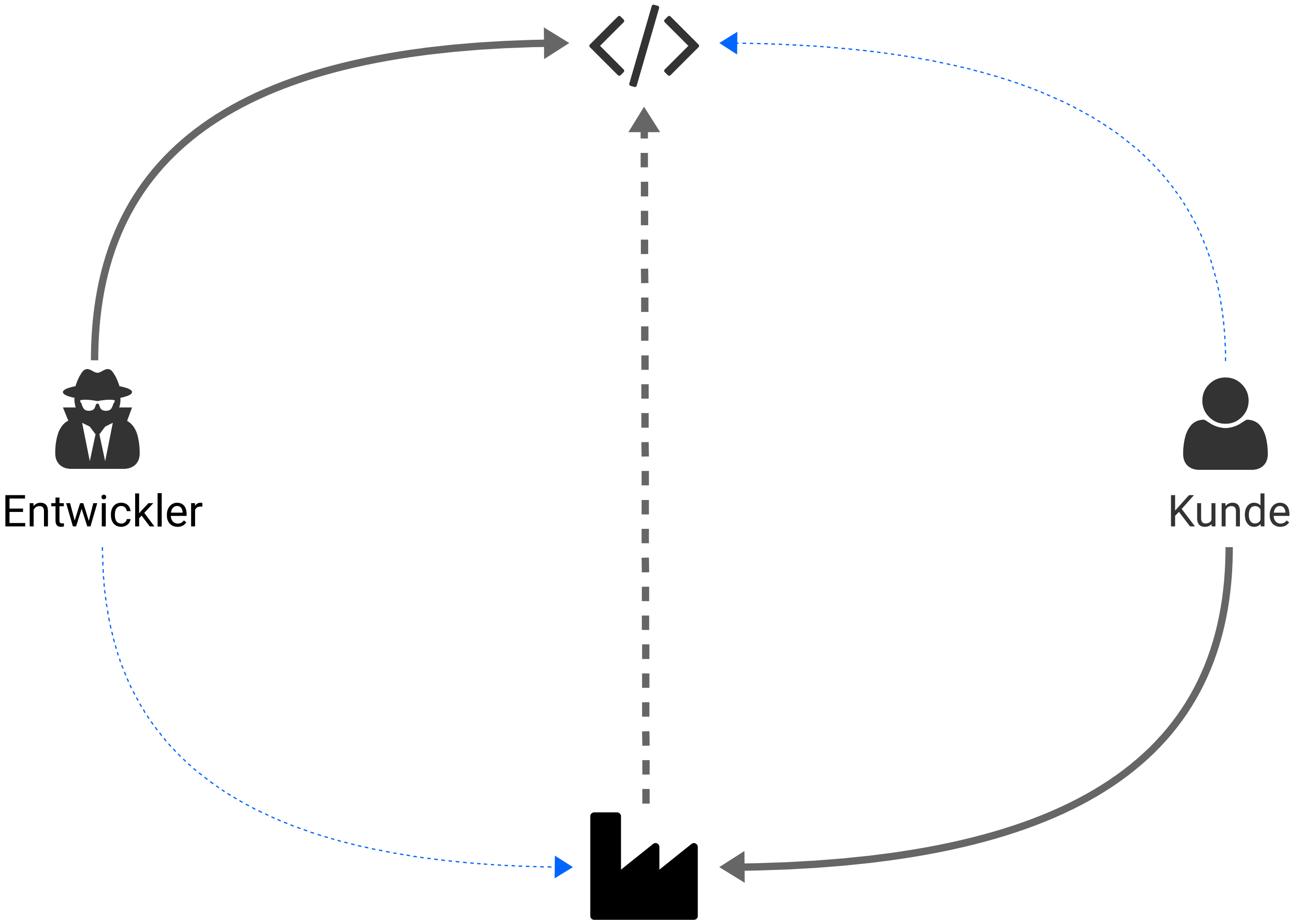
Entwickler

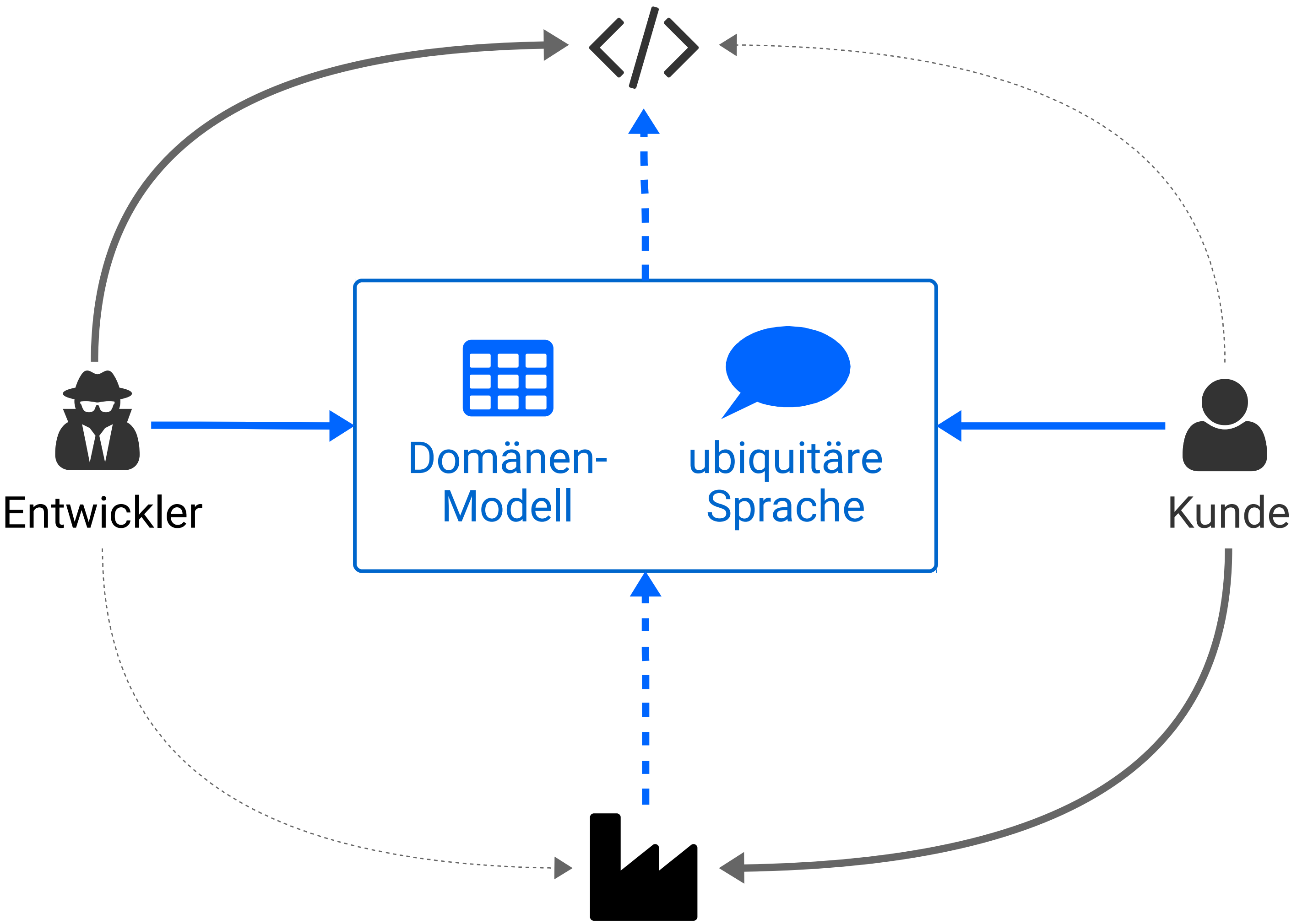
Kunde

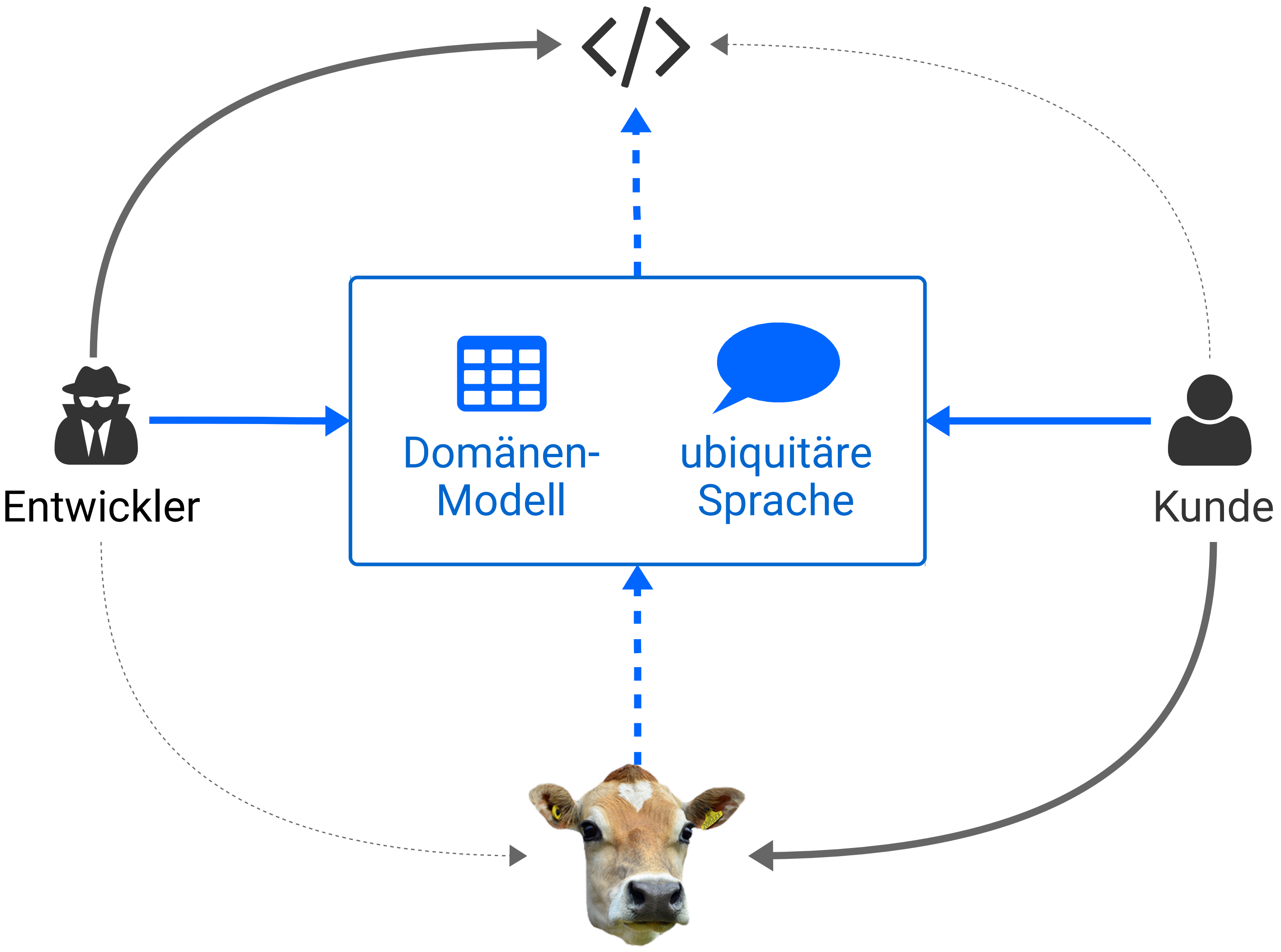


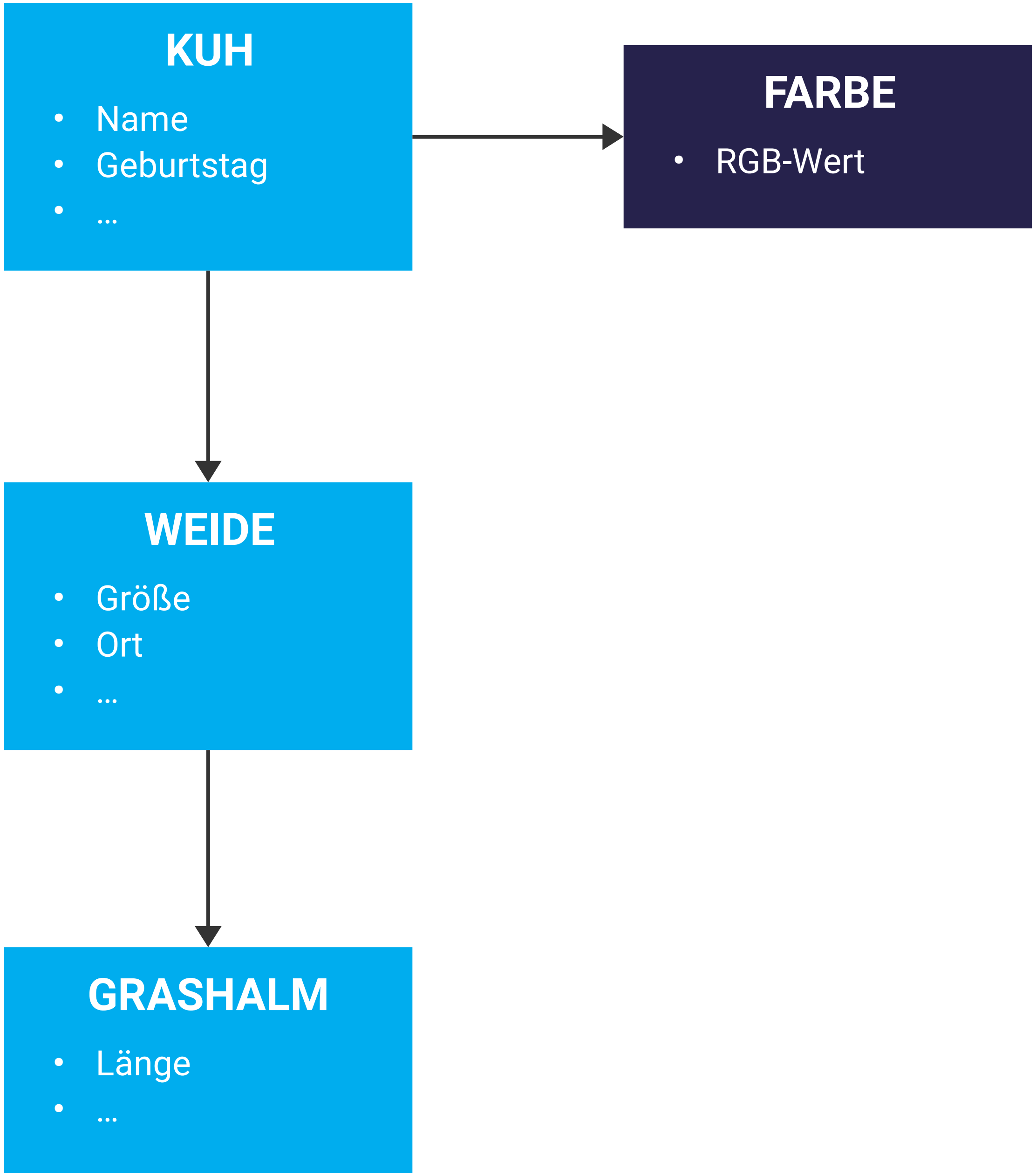
Entwickler

Kunde


Entwickler
Kunde


Entwickler
Domänen-Modell
ubiquitäre Sprache
Kunde


Domänen-Modell
ubiquitäre Sprache
Entwickler
Kunde



KUH

- Name
- Geburtstag
- …

FARBE

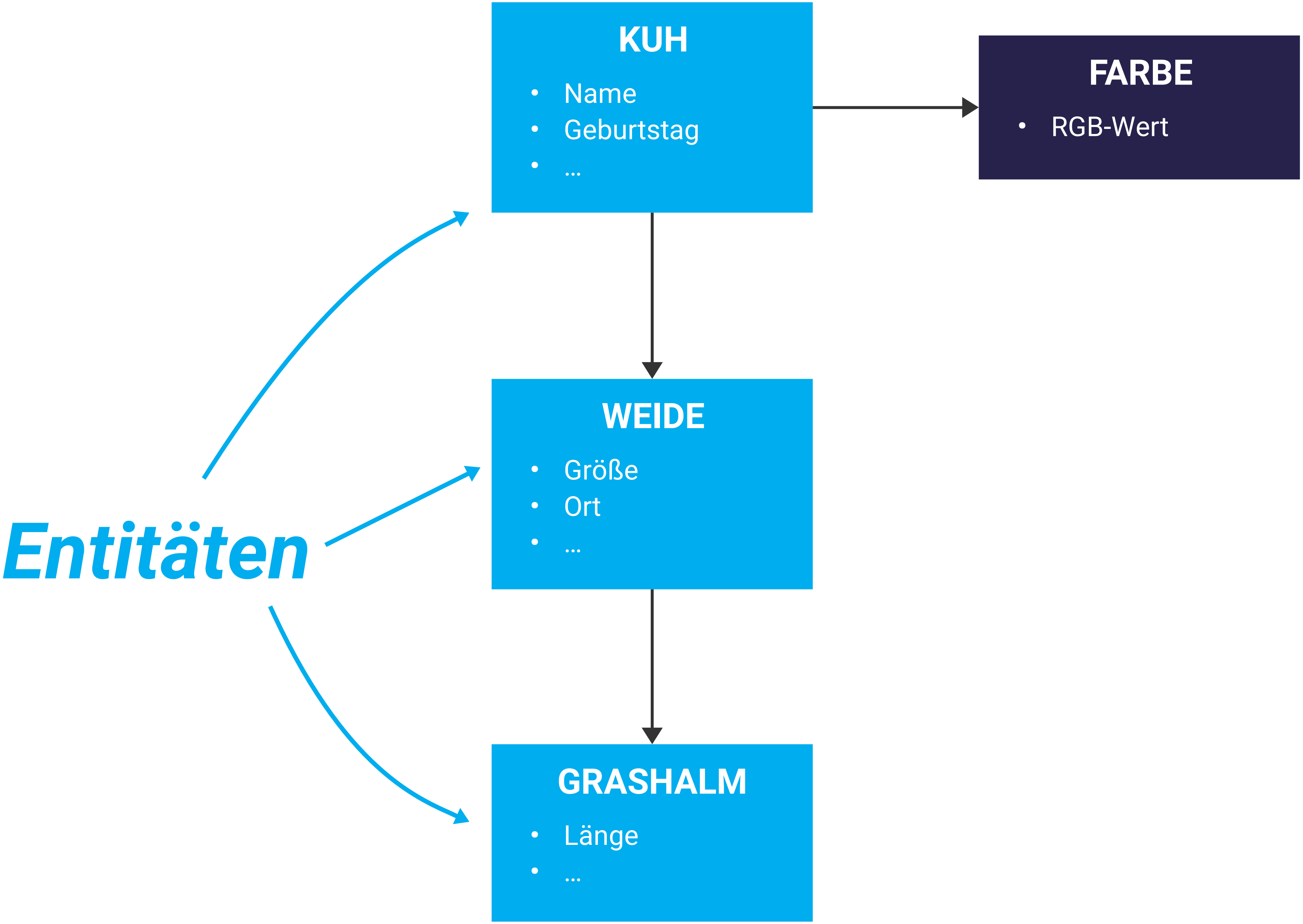
- RGB-Wert

WEIDE

- Größe
- Ort
- …

GRASHALM

- Länge
- …



Entitäten

KUH

- Name
- Geburtstag
- …

FARBE

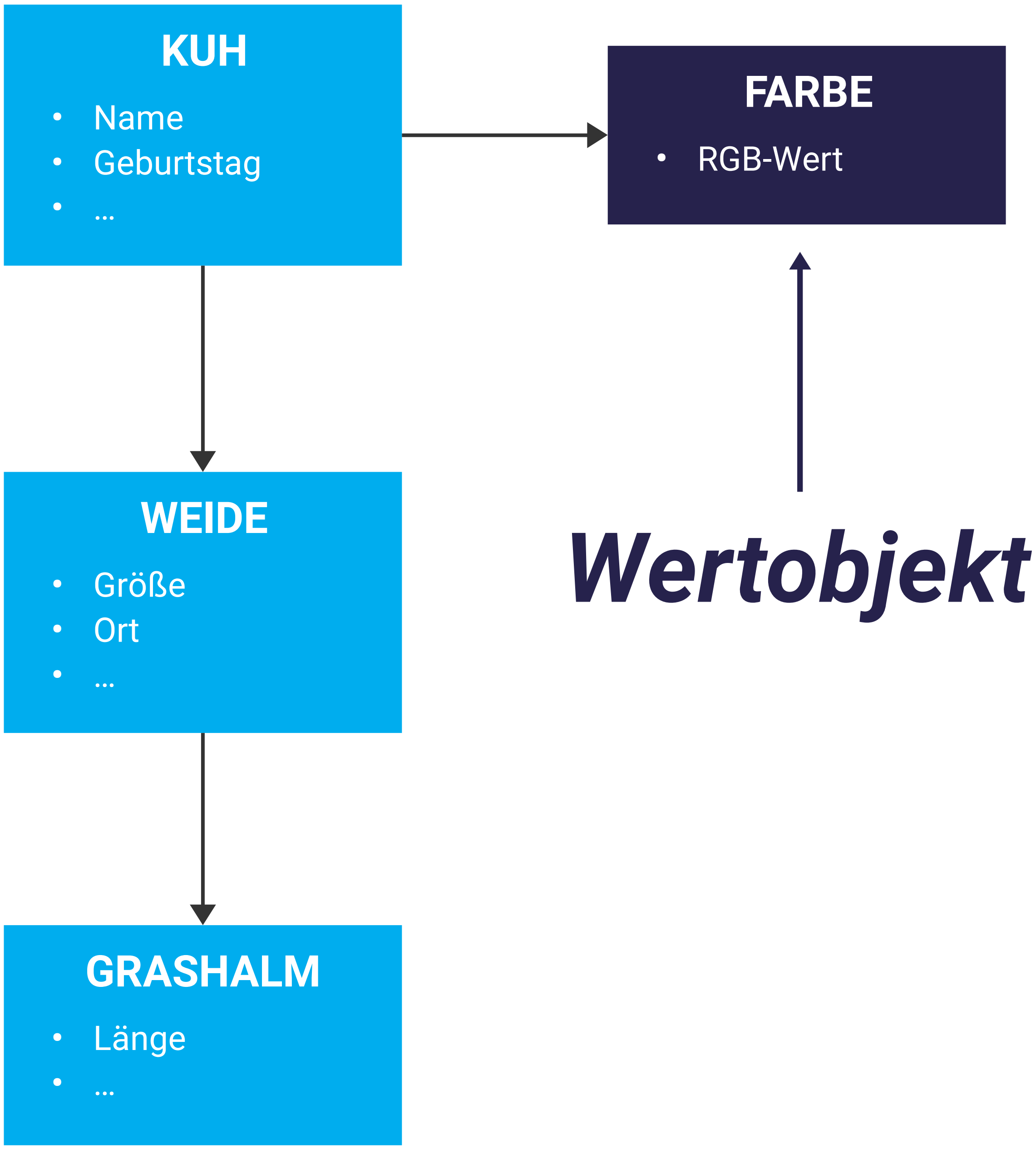
- RGB-Wert

WEIDE

- Größe
- Ort
- …

GRASHALM

- Länge
- …



KUH

- Name
- Geburtstag
- …

FARBE

- RGB-Wert

WEIDE

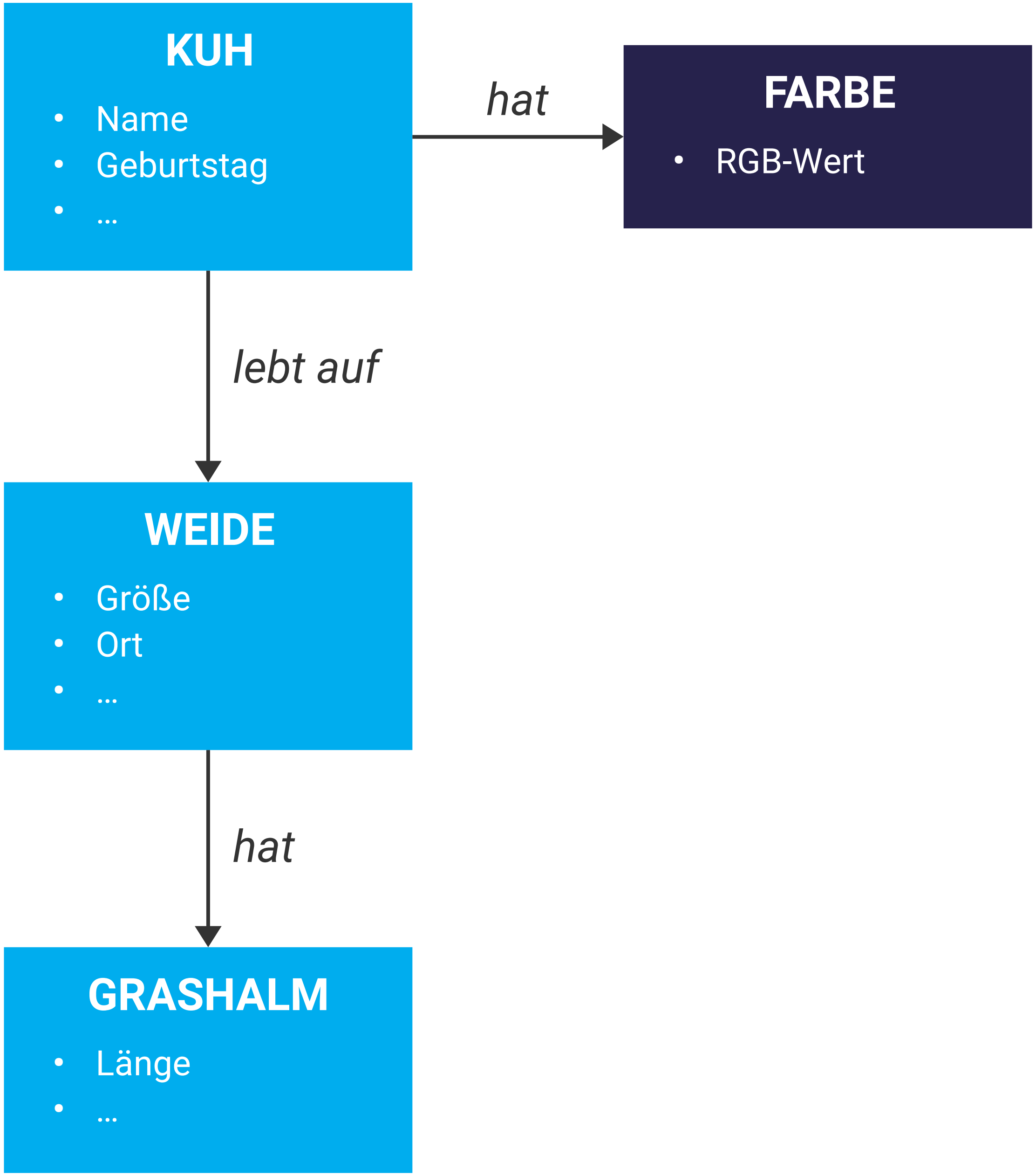
- Größe
- Ort
- …

GRASHALM

- Länge
- …

Wertobjekt

Assoziationen



KUH

- Name
- Geburtstag
- …

hat

FARBE

- RGB-Wert

lebt auf

WEIDE

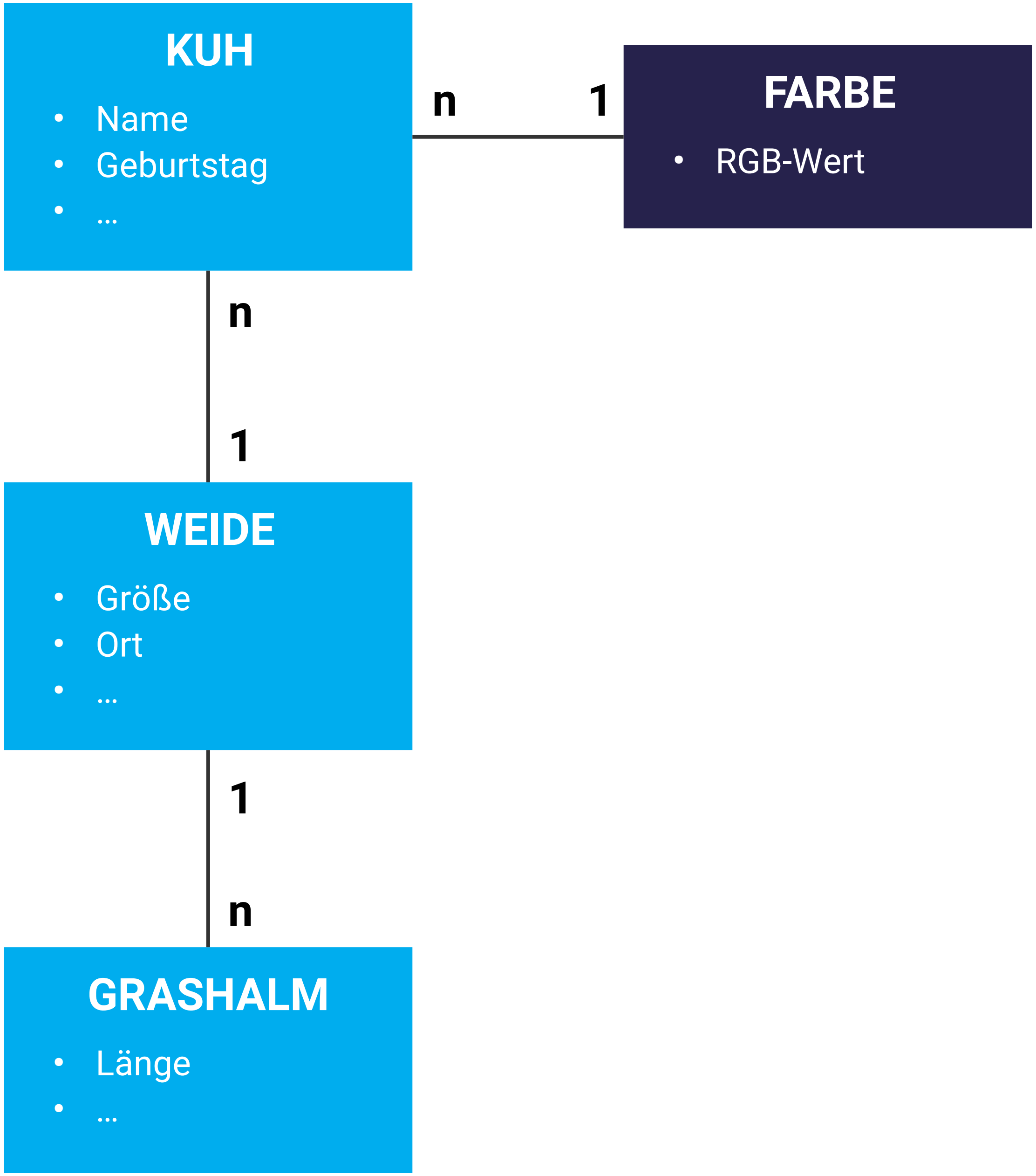
- Größe
- Ort
- …

hat

GRASHALM

- Länge
- …

Assoziationen



KUH

- Name
- Geburtstag
- …

n

1

FARBE

- RGB-Wert

n

1

WEIDE

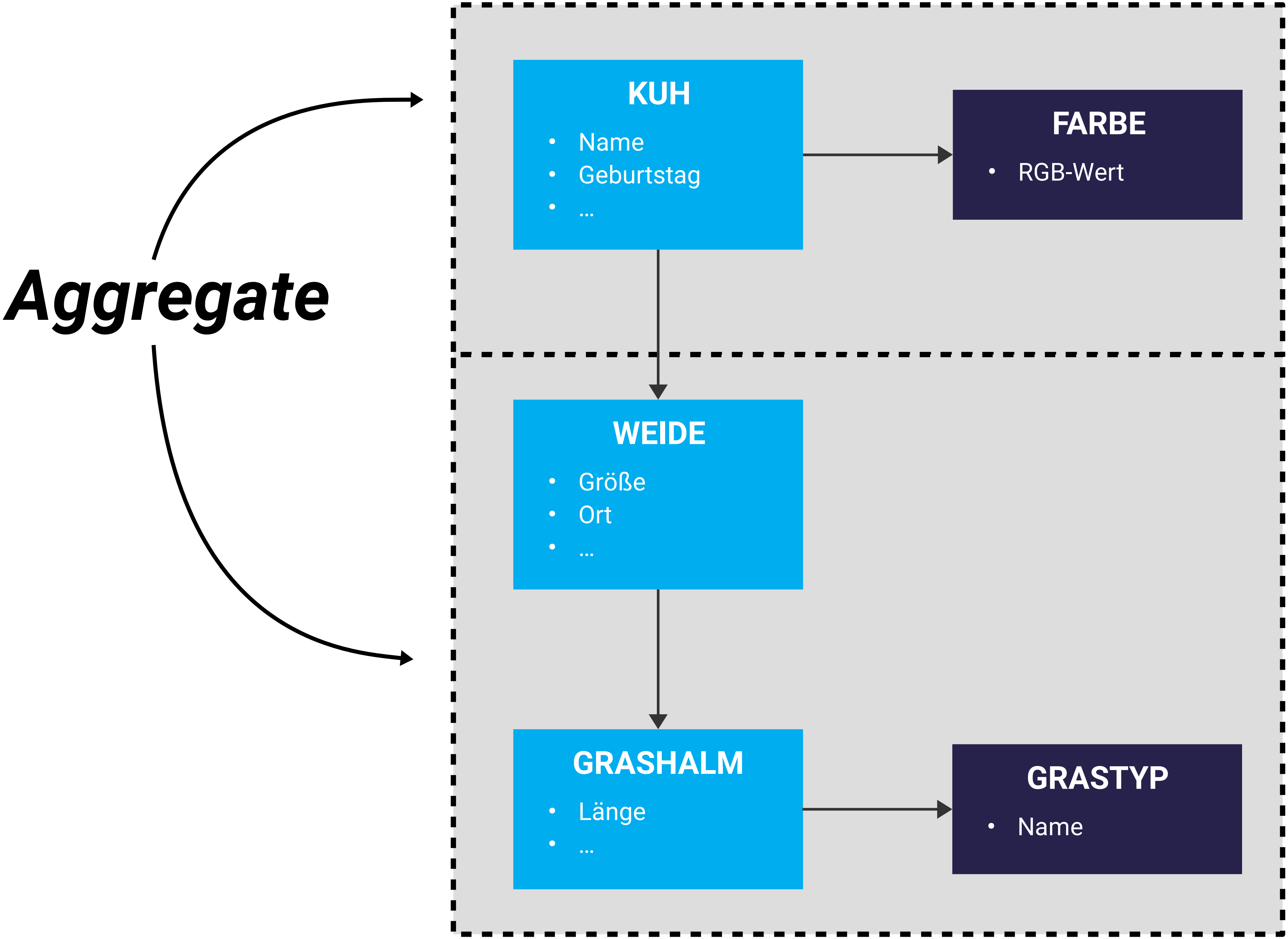
- Größe
- Ort
- …

1

n

GRASHALM

- Länge
- …



Aggregate

KUH

- Name
- Geburtstag
- …

FARBE

- RGB-Wert

WEIDE

- Größe
- Ort
- …

GRASHALM

- Länge
- …

GRASTYP

- Name

KUH
Repository

FARBE

- RGB-Wert

WEIDE
Repository

GRASHALM

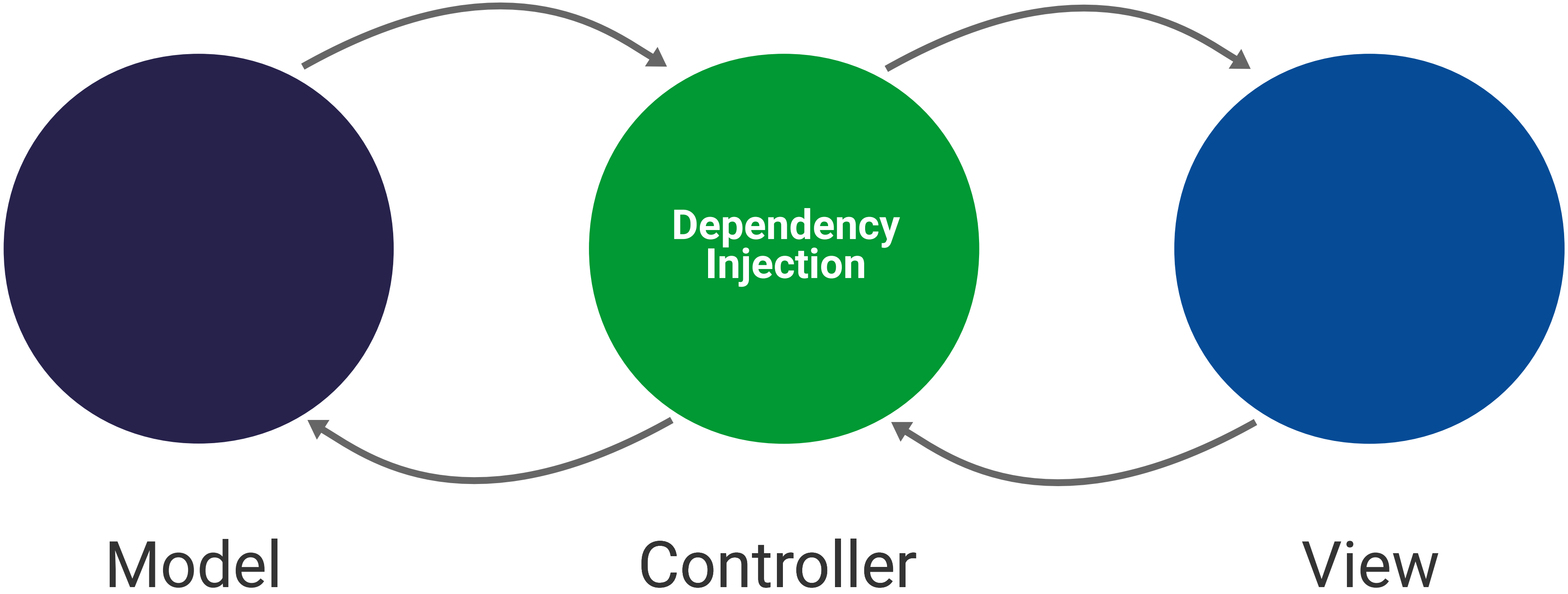
- Länge
- …

GRASTYP

- Name

```
use TYPO3\Flow\Annotations as Flow;
use Doctrine\ORM\Mapping as ORM;

/**
 * @Flow\Entity
 */
class Cow {

    /**
     * @var string
     * @Flow\Identity
     * @ORM\Column(length=80)
     */
    protected $name;

    /**
     * @var \DateTime
     */
    protected $birthday;

    …
```

```php
use TYPO3\Flow\Annotations as Flow;

/**
 * @Flow\Scope(”singleton”)
 */
class CowRepository extends \TYPO3\Flow\Persistence\Repository{}
```


Dependency Injection
Model
Controller
View

http://cows.local/<**PACKAGE**>/<**CONTROLLER**>/<**ACTION**>

http://cows.local/my.package/cow/list

```
class CowController extends \TYPO3\Flow\Mvc\Controller\ActionController {

    public function listAction() {
        $cows = $this->cowRepository->findAll();
        $this->view->assign(”cows”, $cows);
    }

}
```

```
class CowController extends \TYPO3\Flow\Mvc\Controller\ActionController {

    public function listAction() {
        $cows = $this->cowRepository->findAll();
        $this->view->assign("cows", $cows);
    }

}
```

Dependency Injection

```
class CowController extends \TYPO3\Flow\Mvc\Controller\ActionController {

    /**
     * @var MY\PACKAGE\Domain\Repository\CowRepository
     * @Flow\Inject
     */
    protected $cowRepository;


    public function listAction() {

        $cows = $this->cowRepository->findAll();

        $this->view->assign("cows", $cows);

    }


}
```

Dependency Injection

"Dependency Injection" is a 25-dollar term for a 5-cent concept

- **Inversion of Control**
  Kontrolle über den Programmfluss wird an eine (übergeordnete) Komponente abgegeben

- **Hollywood-Prinzip**
  „Don‘t call us, we‘ll call you“

Dependency Injection

mit Dependency Injection

```
/**
 * @var MY\PACKAGE\Interfaces\CowInterface
 * @Flow\Inject
 */
protected $cowInterface;
```

Dependency Injection

mit Dependency Injection

```
/**
 * @var MY\PACKAGE\Interfaces\CowInterface
 * @Flow\Inject
 */
protected $cowInterface;
```

ohne Dependency Injection

```
protected $cowInterface;

public function __construct() {
    $cowInterface = new ConcreteCow();
}
```

```
class CowController extends \TYPO3\Flow\Mvc\Controller\ActionController {

    /**
     * @param MY\PACKAGE\Domain\Model\Cow $cow
     */
    public function showAction(Cow $cow) {
        $this->view->assign("cow”, $cow);
    }


}
```

Property Mapping

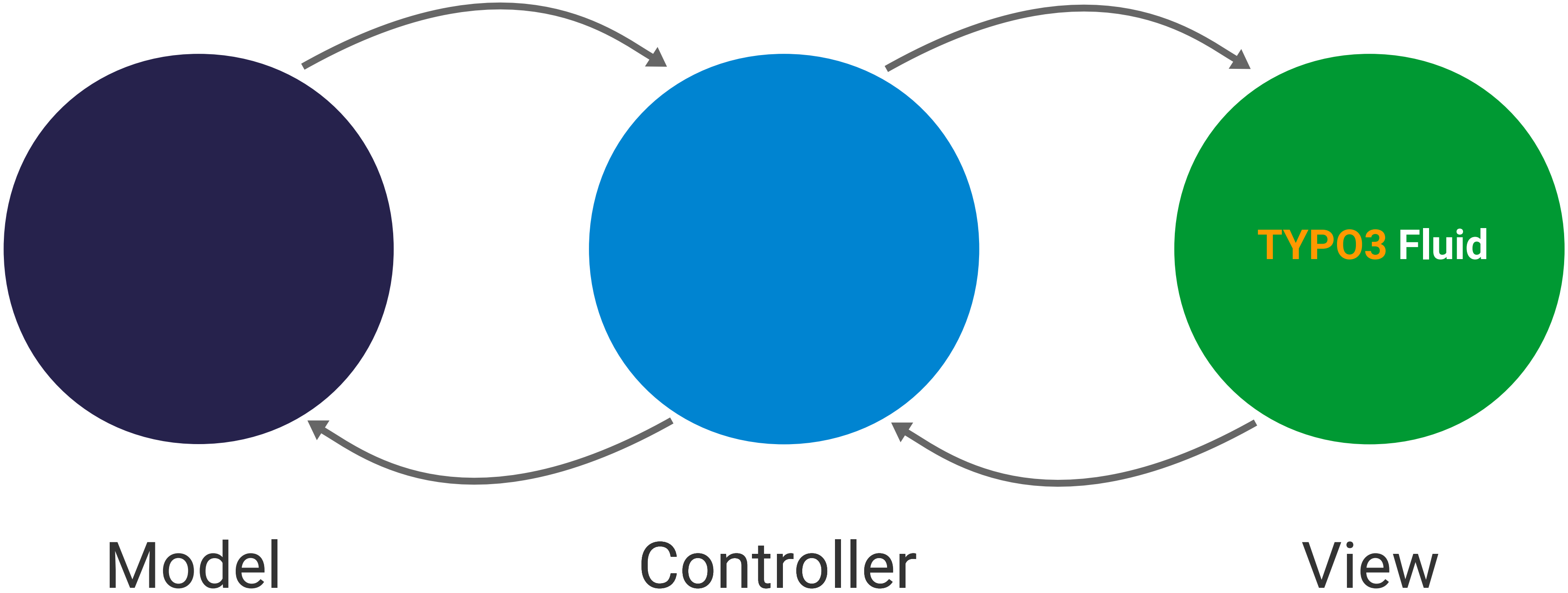
http://cows.local/my.package/cow/show/99ad52e5-61a7-4060-b616-2715c74eaafd

```
class CowController extends \TYPO3\Flow\Mvc\Controller\ActionController {

    /**
     * @param MY\PACKAGE\Domain\Model\Cow $cow
     */
    public function showAction(Cow $cow) {
        $this->view->assign("cow", $cow);
    }


}
```

Property Mapping

http://cows.local/cows/volma

```
class CowController extends \TYPO3\Flow\Mvc\Controller\ActionController {

    /**
     * @param MY\PACKAGE\Domain\Model\Cow $cow
     */
    public function showAction(Cow $cow) {
        $this->view->assign(”cow”, $cow);
    }


}
```



TYPO3 Fluid

Model

Controller

View

Fluid-Template

…/MY.PACKAGE/Resources/Private/Templates/Cow/Show.html

```
<f:layout name="Default” />

<f:section name="Title”>{cow.name}</f:section>

<f:section name="Content”>
    <h1>{cow.name}</h1>
    <f:format.date format="d.m.Y”>{cow.birthday}</f:format.date>
</f:section>
```

Platzhalter

…/MY.PACKAGE/Resources/Private/Templates/Cow/Show.html

```
<f:layout name="Default" />

<f:section name="Title">{cow.name}</f:section>

<f:section name="Content">
    <h1>{cow.name}</h1>
    <f:format.date format="d.m.Y">{cow.birthday}</f:format.date>
</f:section>
```

Fluid-ViewHelper

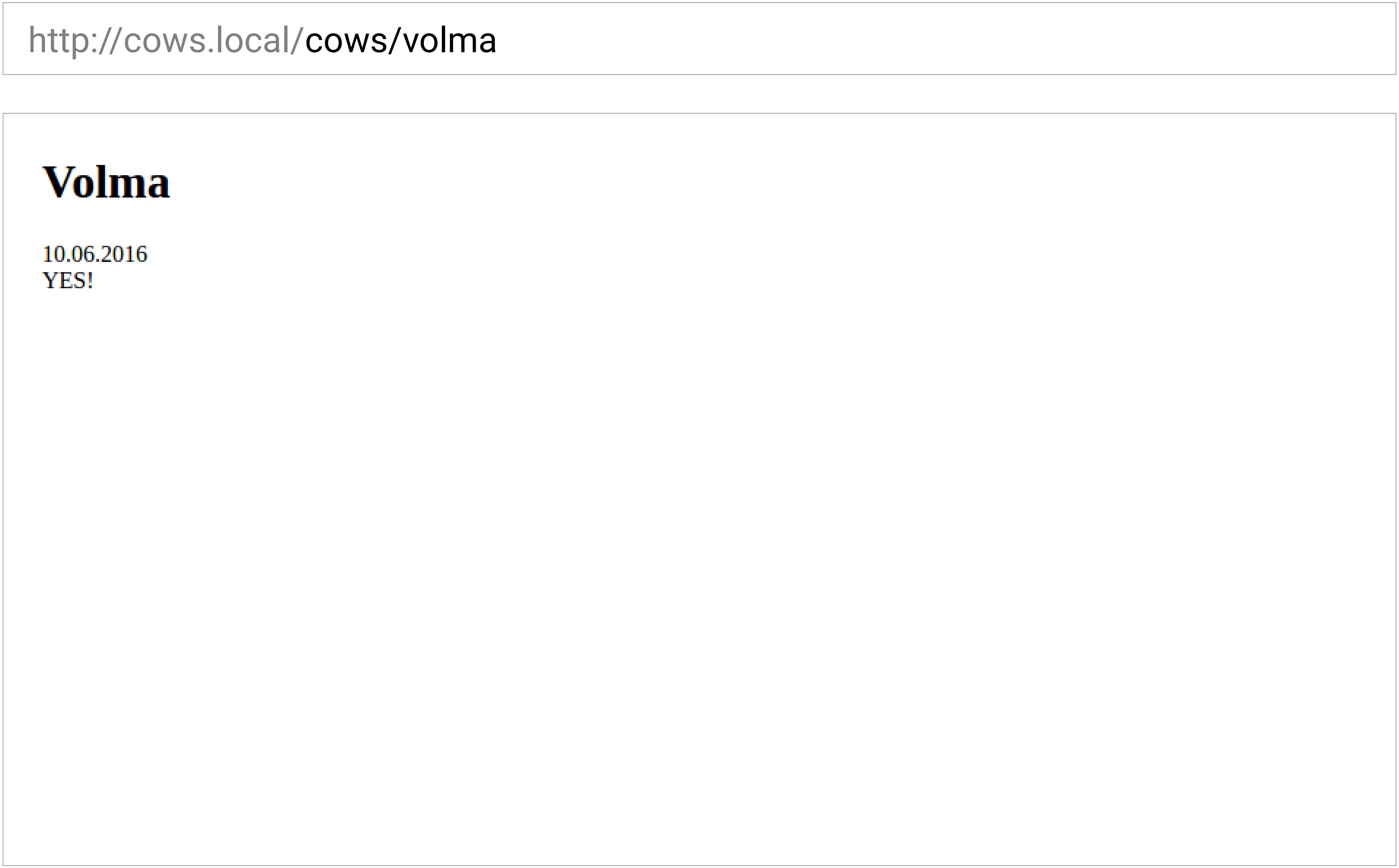
…/MY.PACKAGE/Resources/Private/Templates/Cow/Show.html

```
<f:layout name=”Default” />

<f:section name=”Title”>{cow.name}</f:section>

<f:section name=”Content”>
    <h1>{cow.name}</h1>
    <f:format.date format=”d.m.Y”>{cow.birthday}</f:format.date>
</f:section>
```

eigene ViewHelper

…/MY.PACKAGE/Resources/Private/Templates/Cow/Show.html

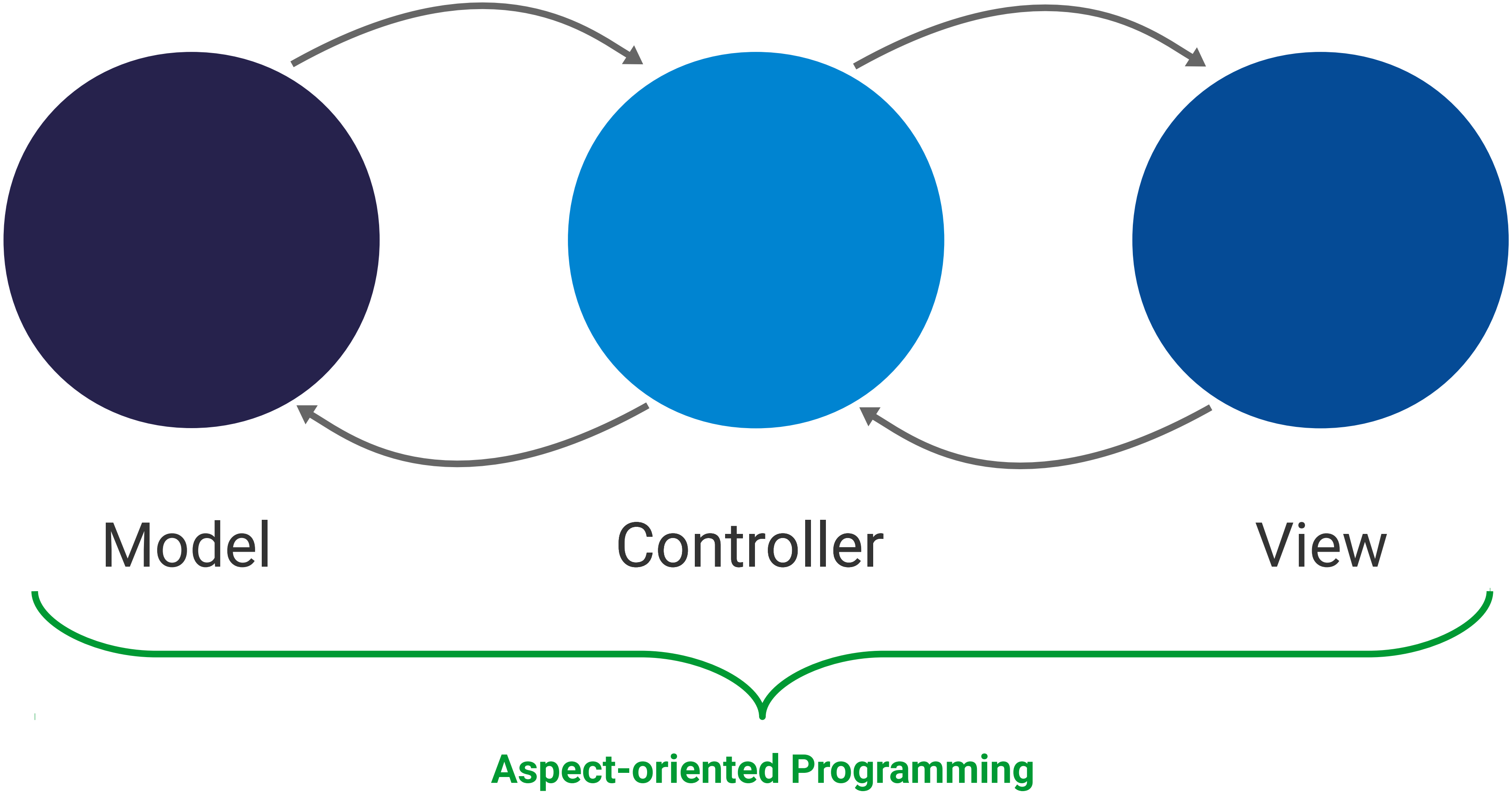
```
{namespace cow=MY\PACKAGE\ViewHelpers}
<f:layout name=”Default” />

<f:section name=”Title”>{cow.name}</f:section>

<f:section name=”Content”>
    <h1>{cow.name}</h1>
    <f:format.date format=”d.m.Y”>{cow.birthday}</f:format.date>
    <cow:isVolma cow=”{cow}” />
</f:section>
```



http://cows.local/cows/volma

Volma

10.06.2016
YES!



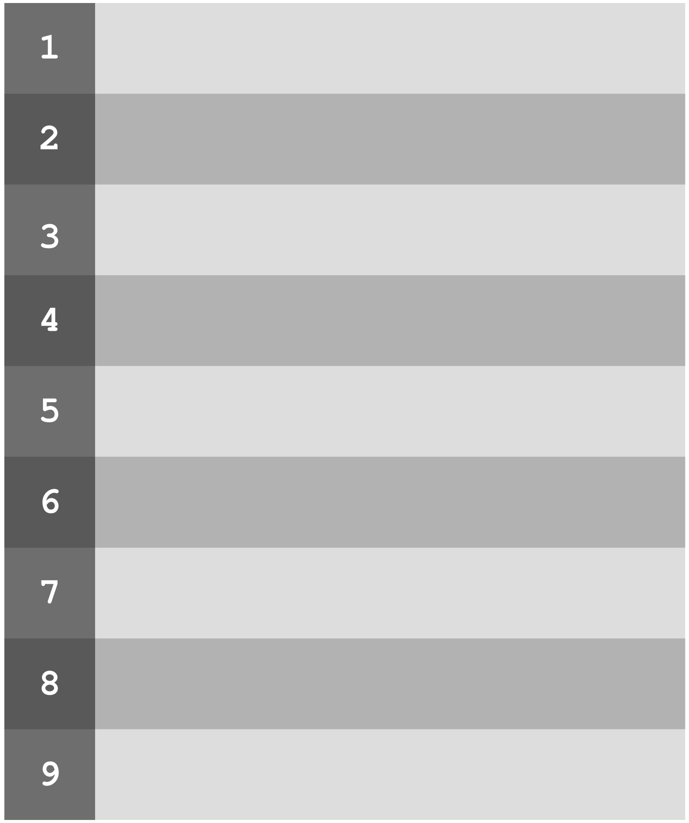
Model

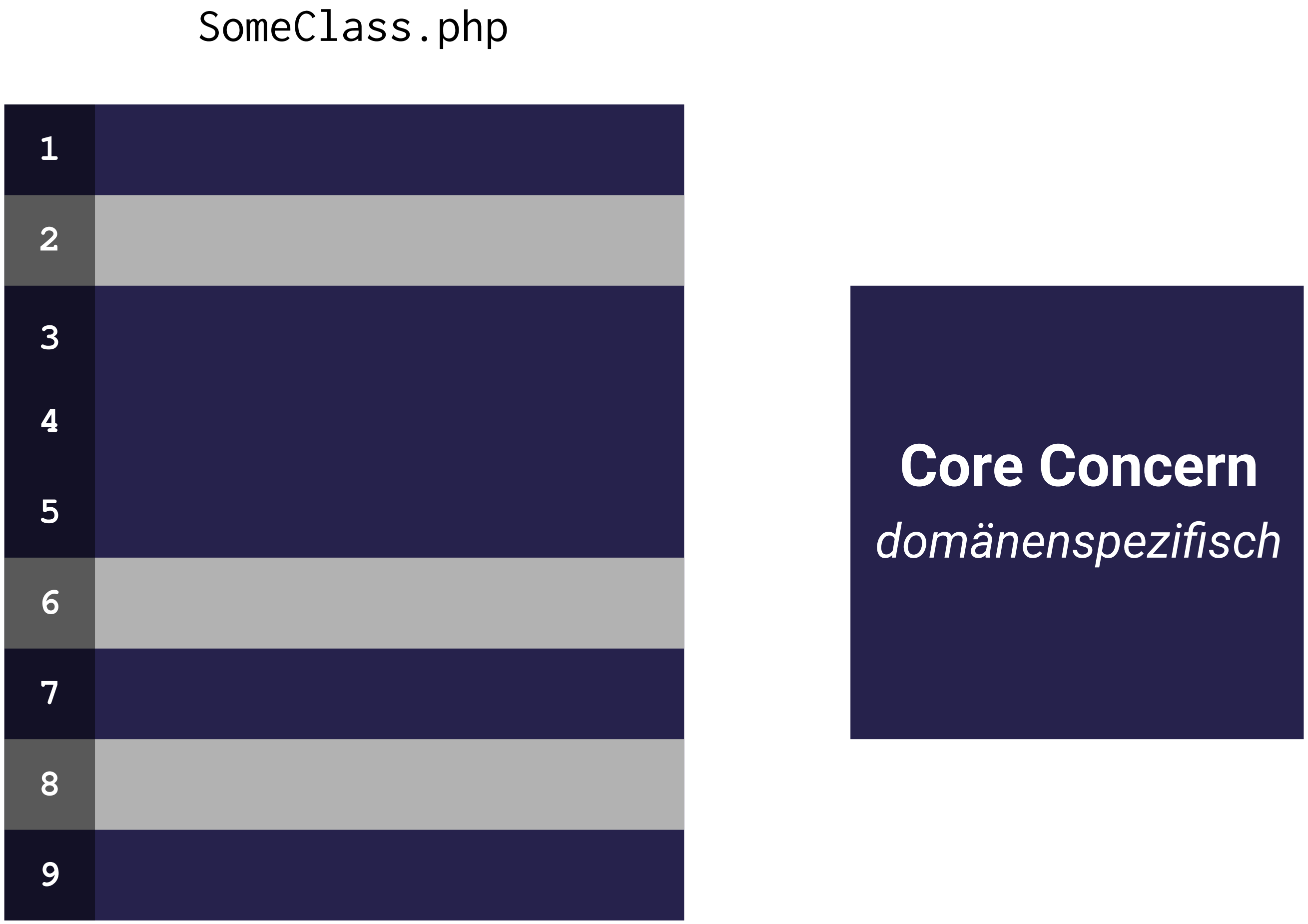
Controller

View

Aspect-oriented Programming

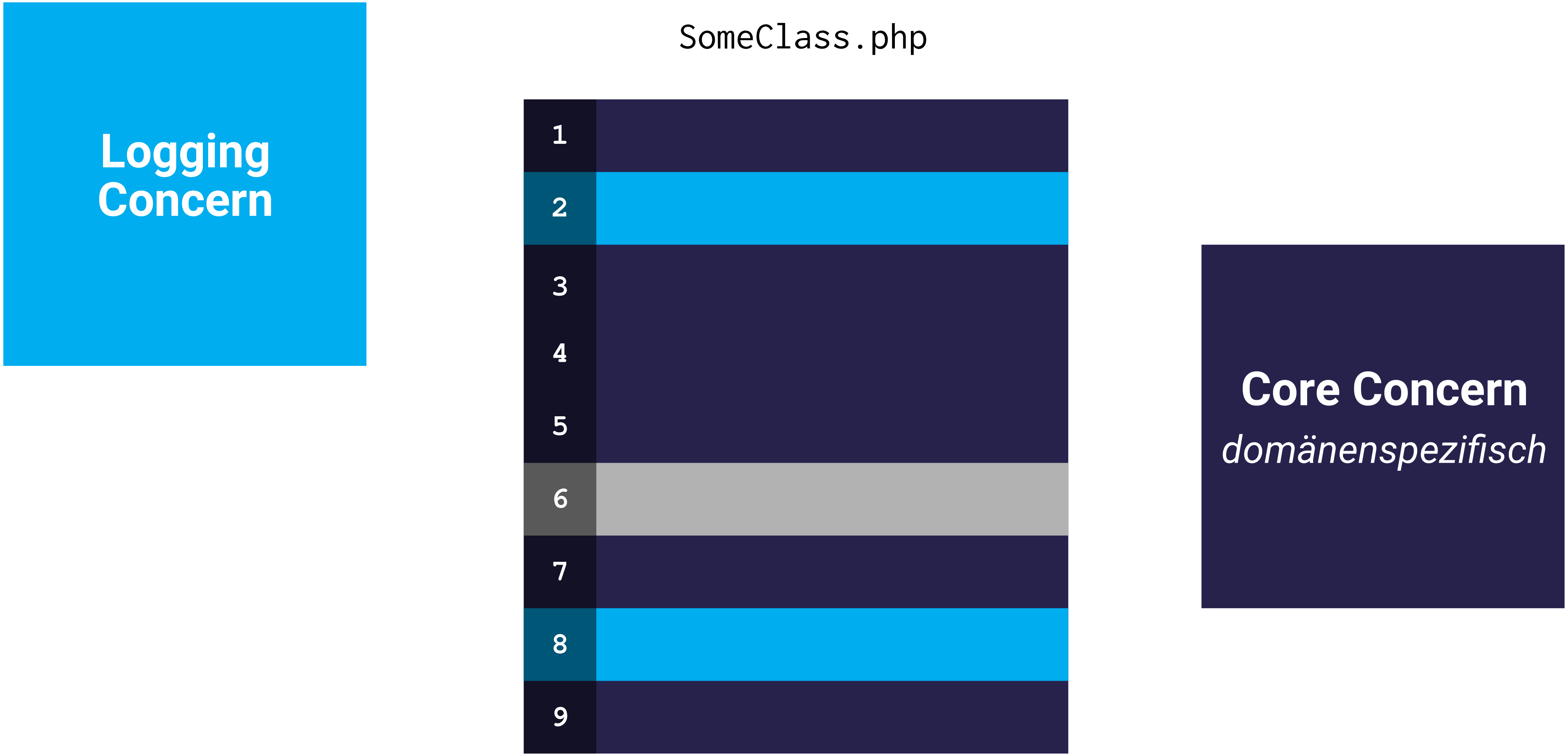
SomeClass.php





Core Concern

domänenspezifisch

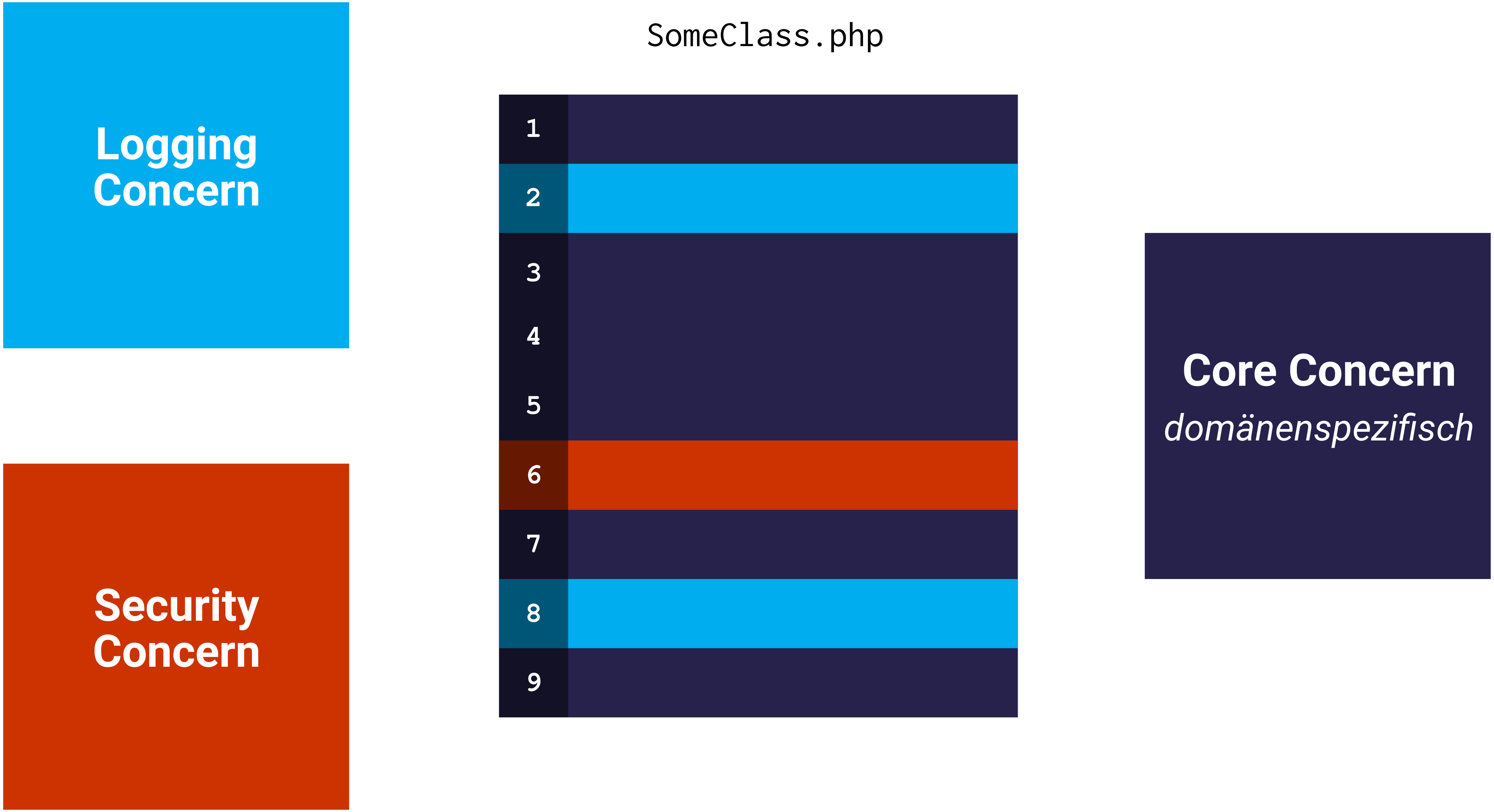


Logging Concern

SomeClass.php

Core Concern

domänenspezifisch



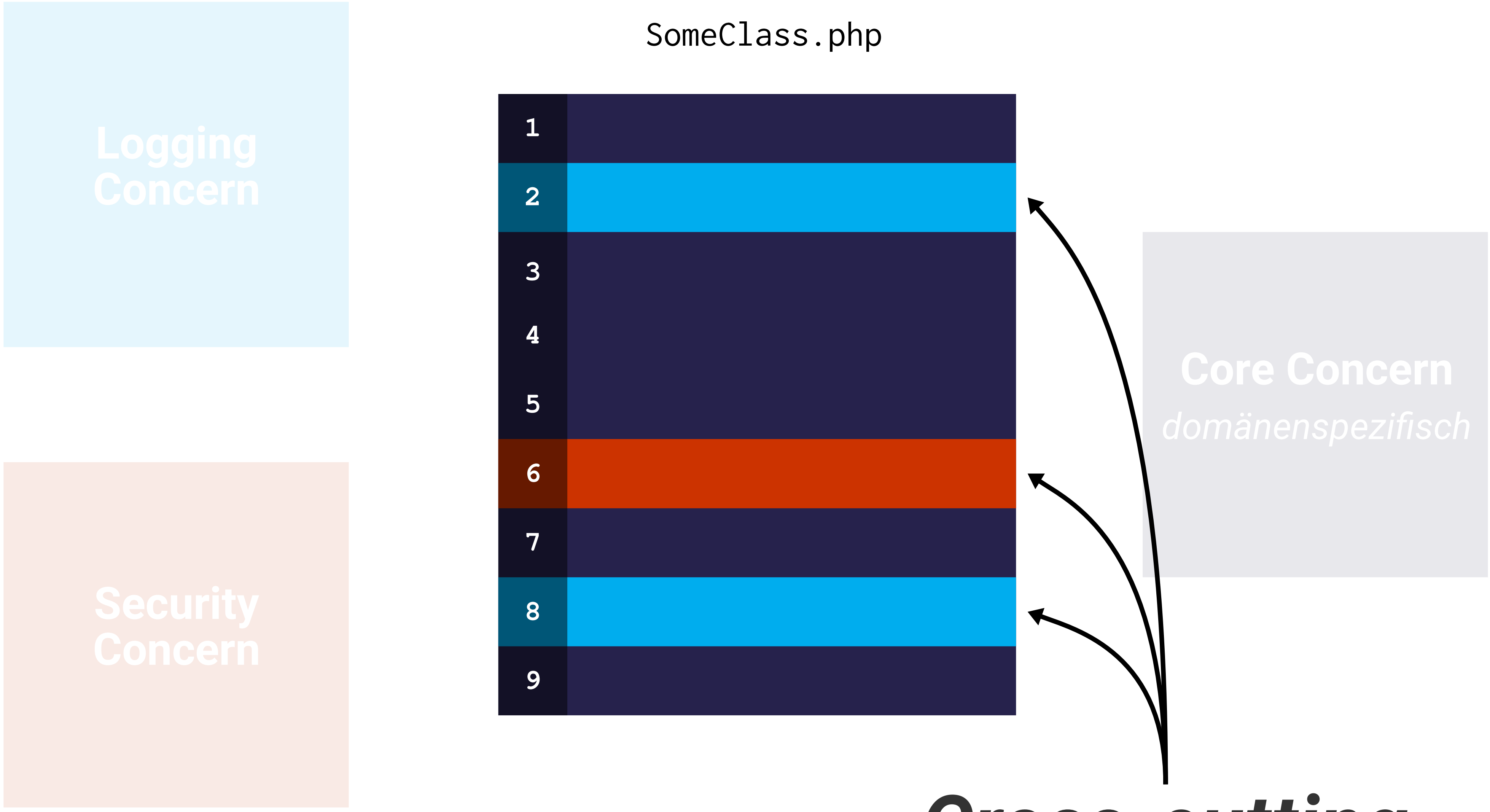
Logging Concern

Security Concern

SomeClass.php

Core Concern

domänenspezifisch



Logging Concern

Security Concern

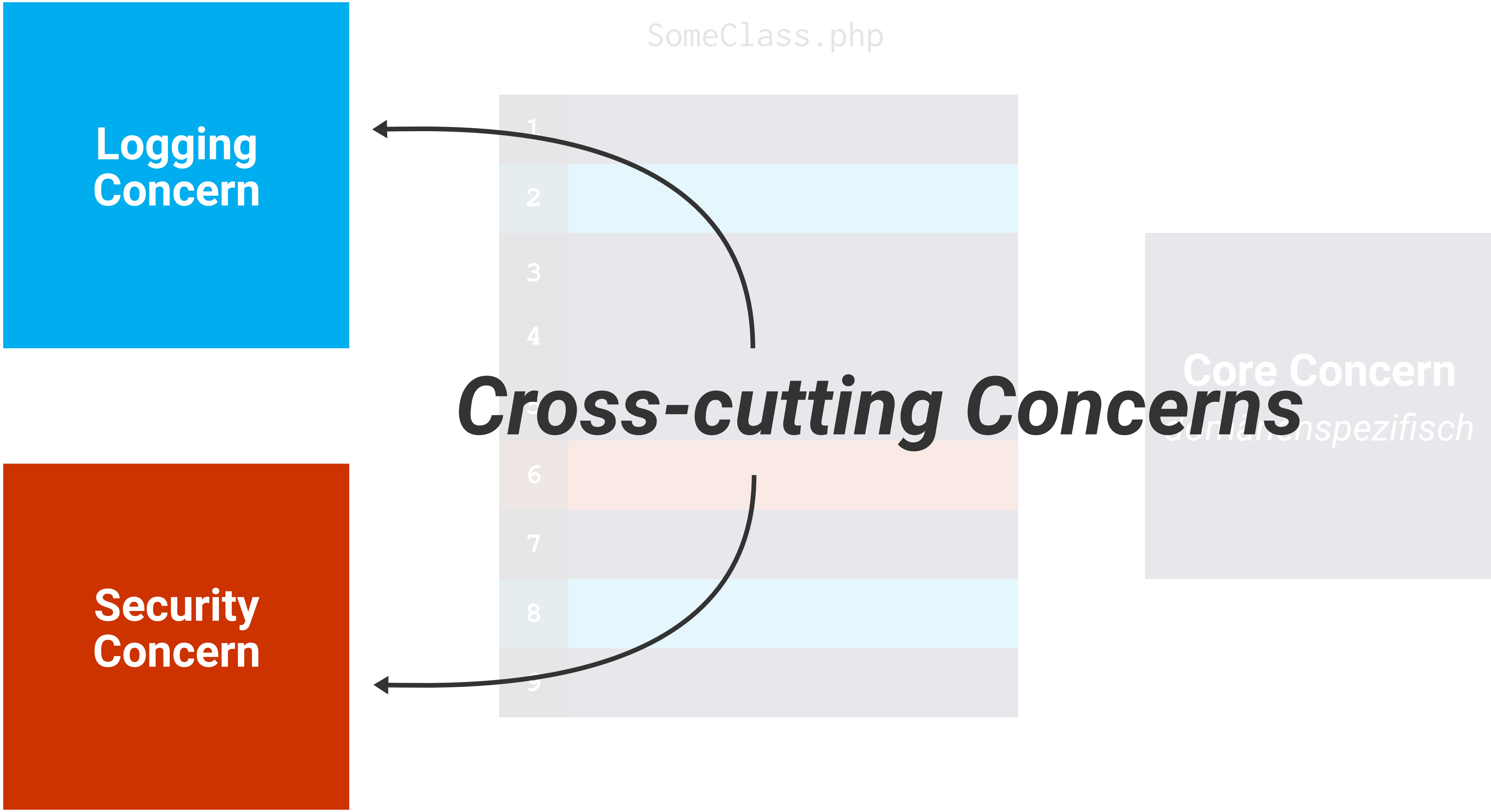
SomeClass.php

Core Concern

domänenspezifisch

Cross-cutting Concerns

Cross-cutting Concerns



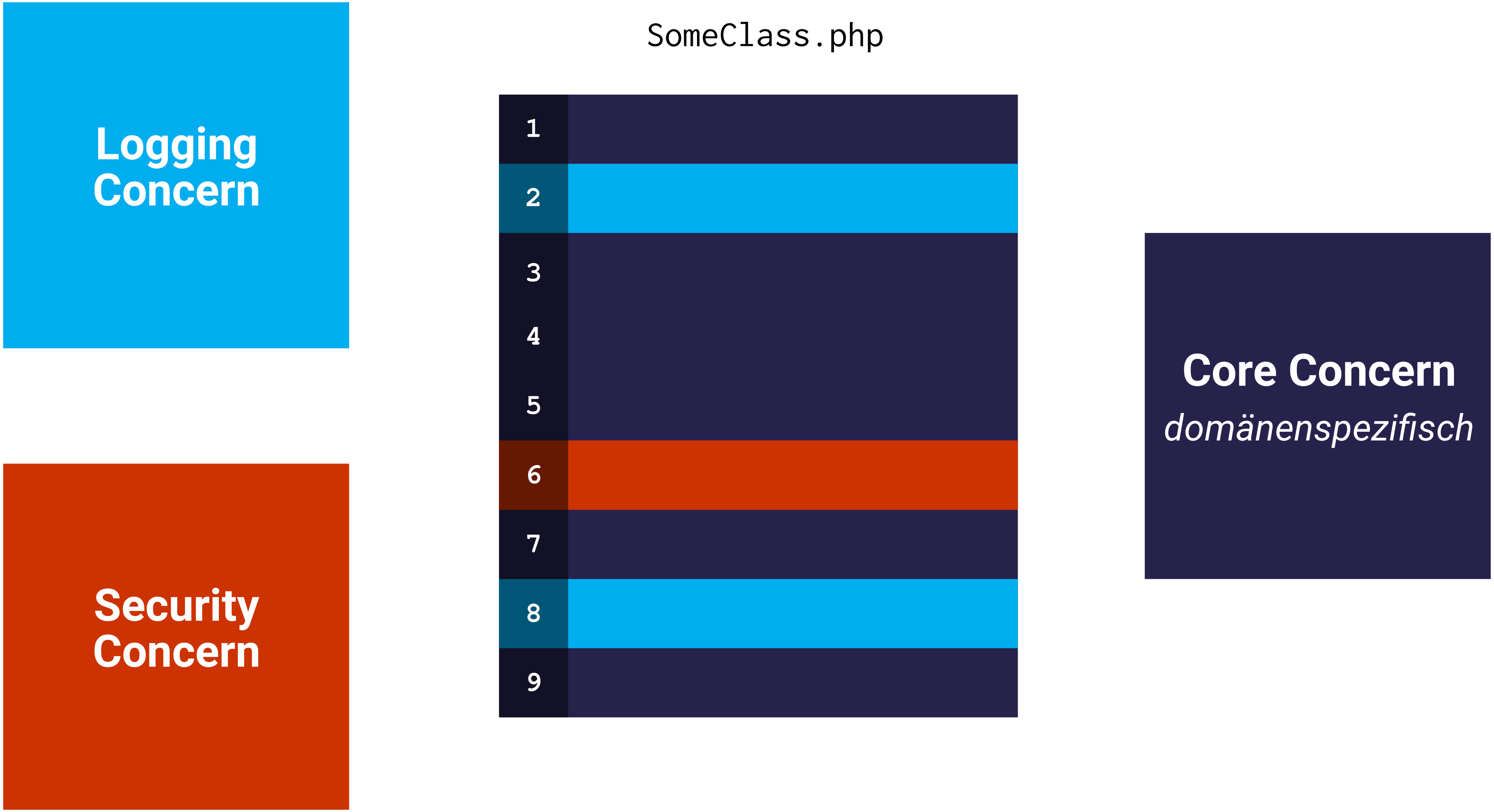
Logging Concern

Security Concern

SomeClass.php

Core Concern

domänenspezifisch



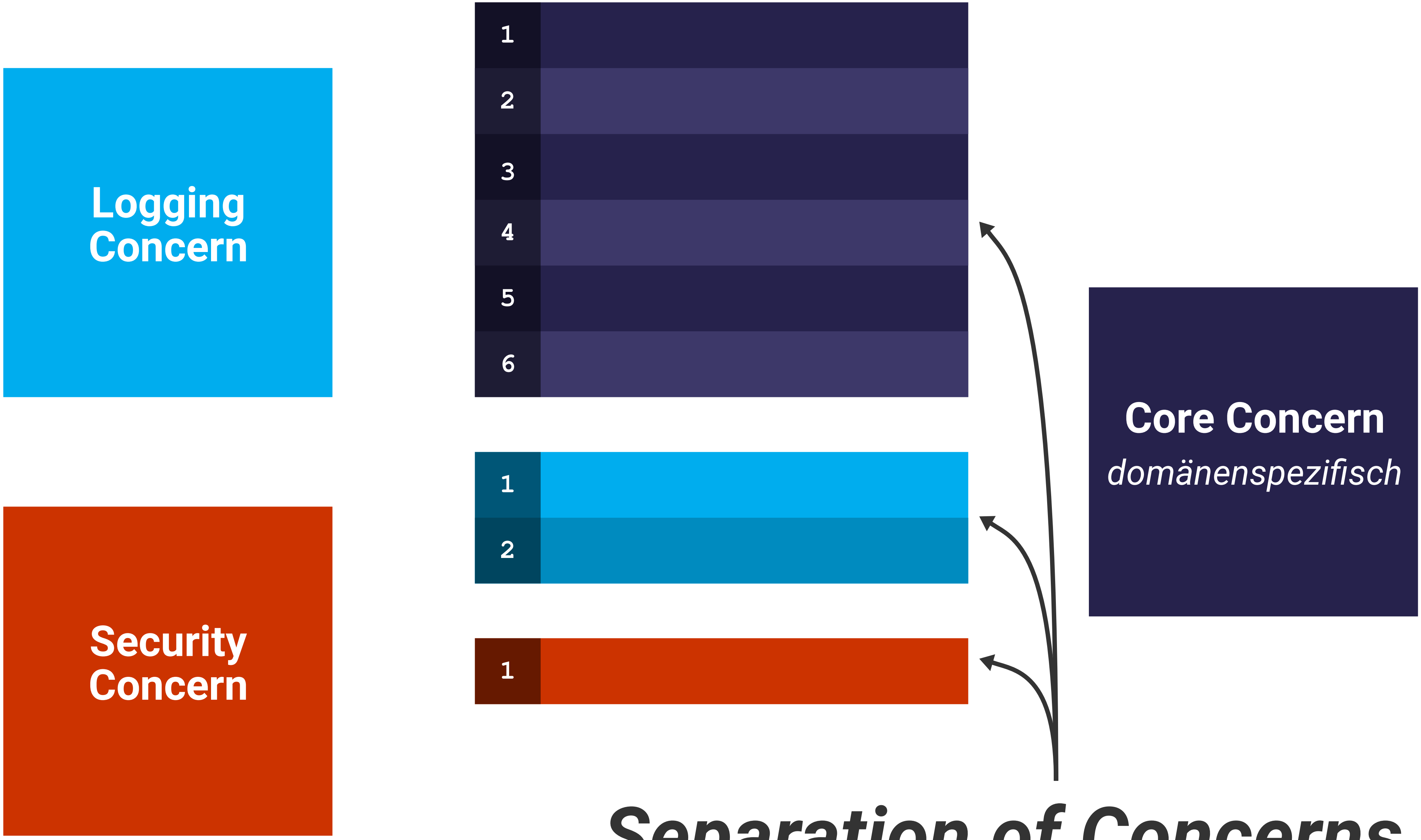
Logging Concern

Security Concern

`SomeClass.php`

Core Concern

domänenspezifisch



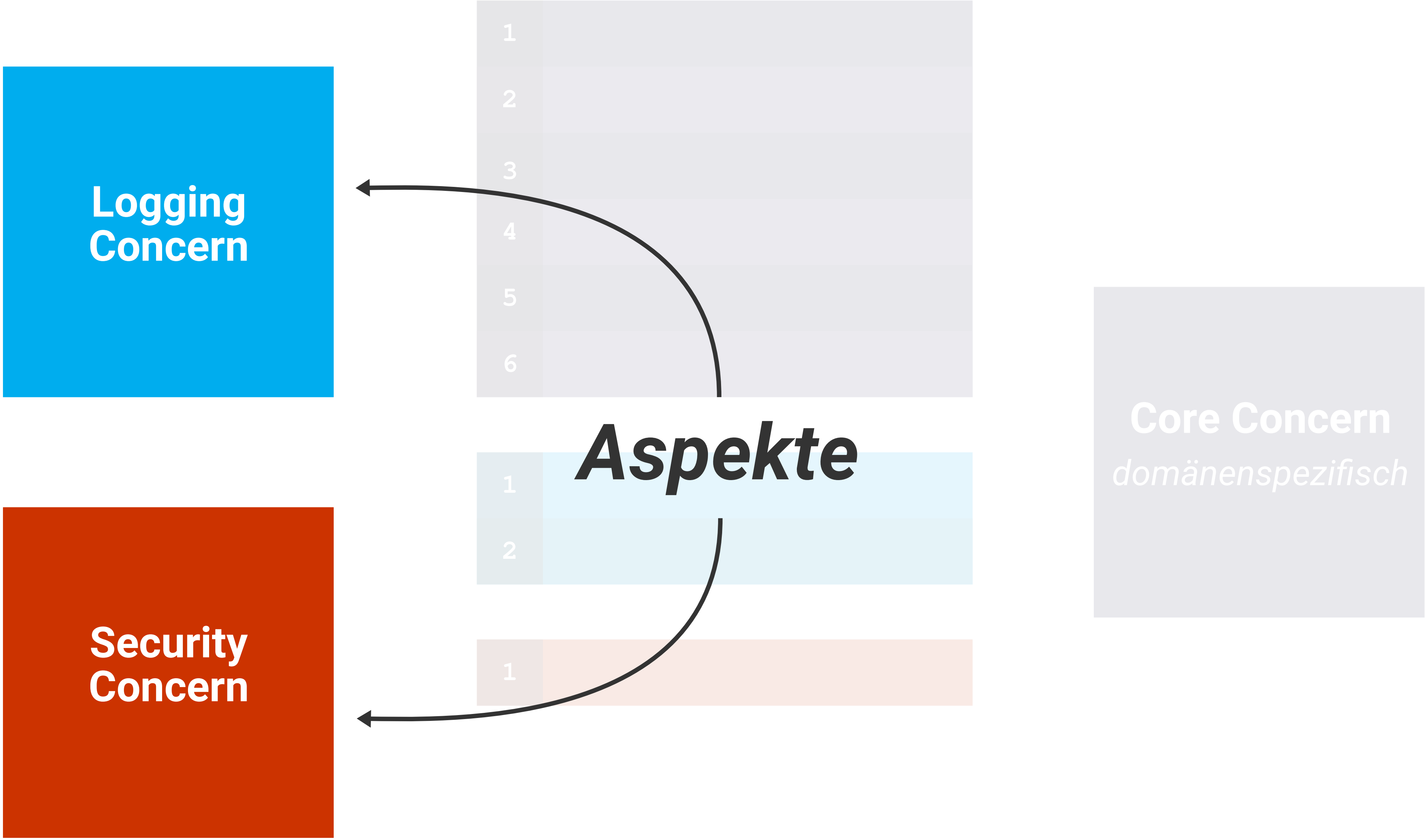
Logging Concern

Security Concern

Core Concern

domänenspezifisch

Separation of Concerns



Logging Concern

Aspekte

Core Concern

domänenspezifisch

Security Concern

```
use TYPO3\Flow\Annotations as Flow;

class CowController extends \TYPO3\Flow\Mvc\Controller\ActionController {

    /**
     * @param MY\PACKAGE\Domain\Model\Cow $cow
     */
    public function showAction(Cow $cow) {
        Logger::log("A cow has been shown");
        $this->view->assign("cow", $cow);
    }


}
```

Aspect

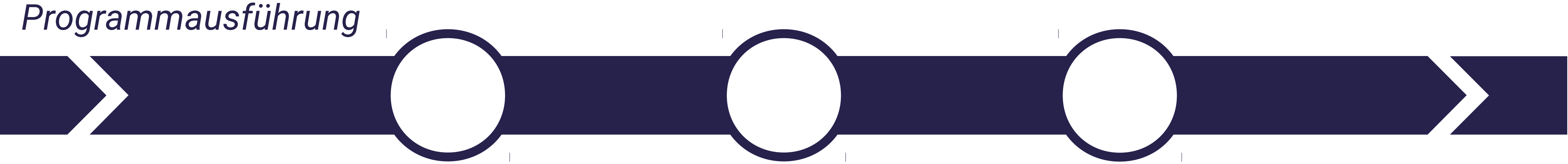
```
/*
 * @Flow\Aspect
 */
class LoggingAspect {

}
```

Advice

```
/*
 * @Flow\Aspect
 */
class LoggingAspect {

    public function logAdvice() {}

}
```

Join point

```
/*
 * @Flow\Aspect
 */
class LoggingAspect {

    /**
     * @param JoinPointInterface $joinPoint
     */
    public function logAdvice(JoinPointInterface $joinPoint) {}

}
```



Programmausführung

Pointcut

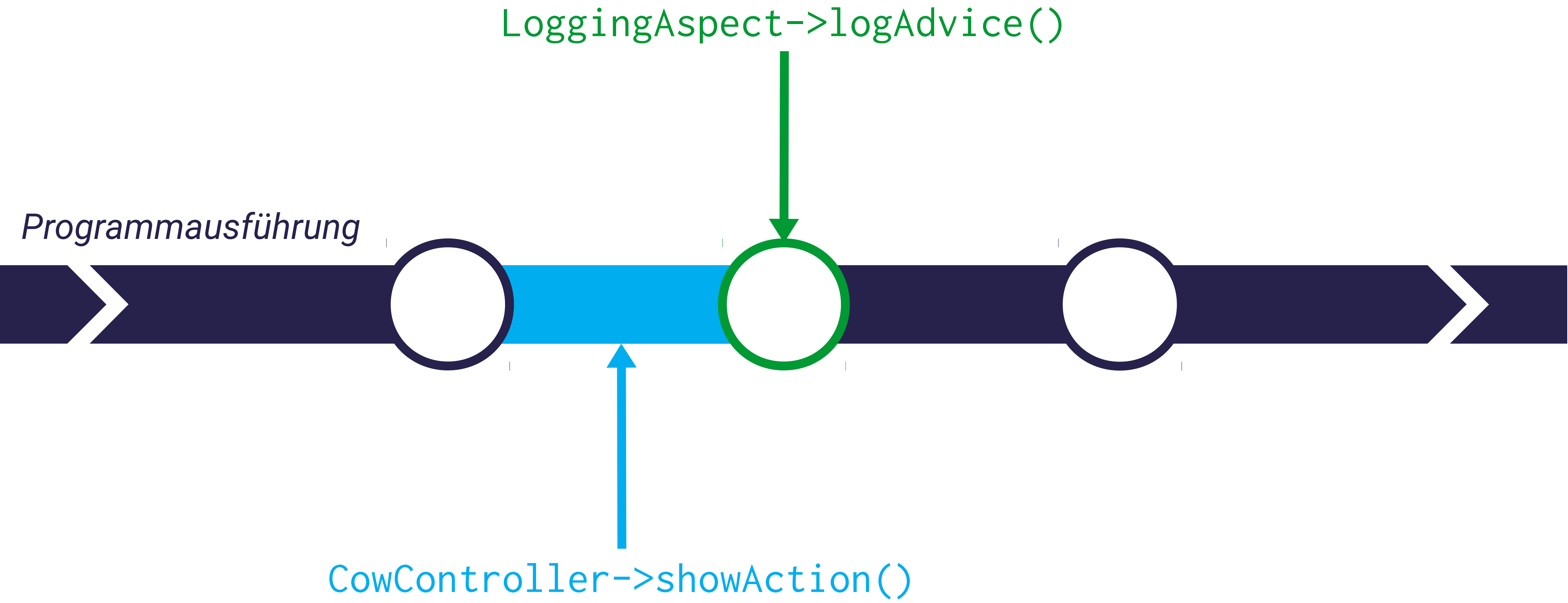
```
/*
 * @Flow\Aspect
 */
class LoggingAspect {

    /**
     * @param JoinPointInterface $joinPoint
     * @Flow\After()
     */
    public function logAdvice(JoinPointInterface $joinPoint) {}

}
```

Target

```
/*
 * @Flow\Aspect
 */
class LoggingAspect {

    /**
     * @param JoinPointInterface $joinPoint
     * @Flow\After(”method()”)
     */
    public function logAdvice(JoinPointInterface $joinPoint) {}

}
```

Target

```
/*
 * @Flow\Aspect
 */
class LoggingAspect {

    /**
     * @param JoinPointInterface $joinPoint
     * @Flow\After(”method(MY\PACKAGE\Controller\.*->.*())”)
     */
    public function logAdvice(JoinPointInterface $joinPoint) {}

}
```

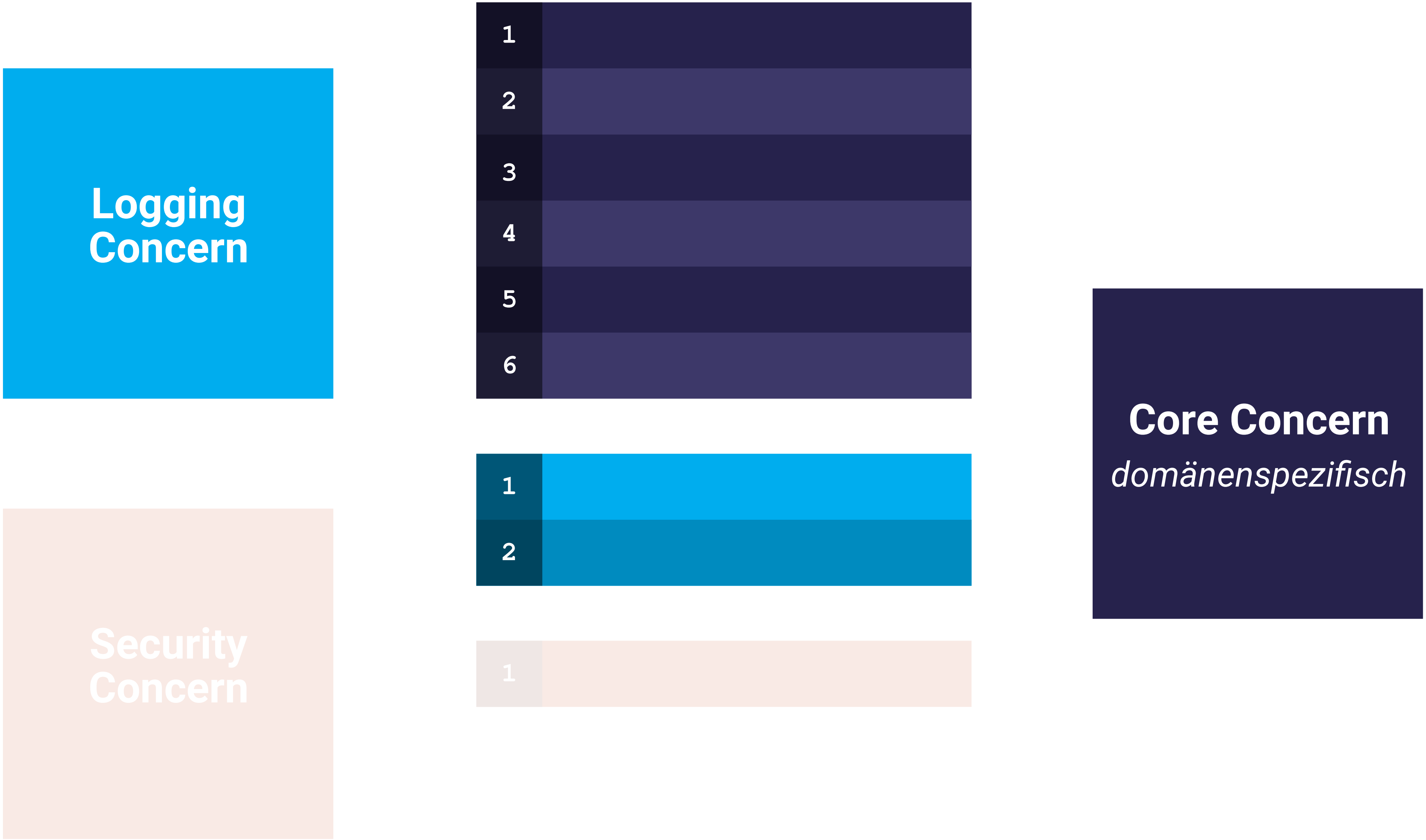


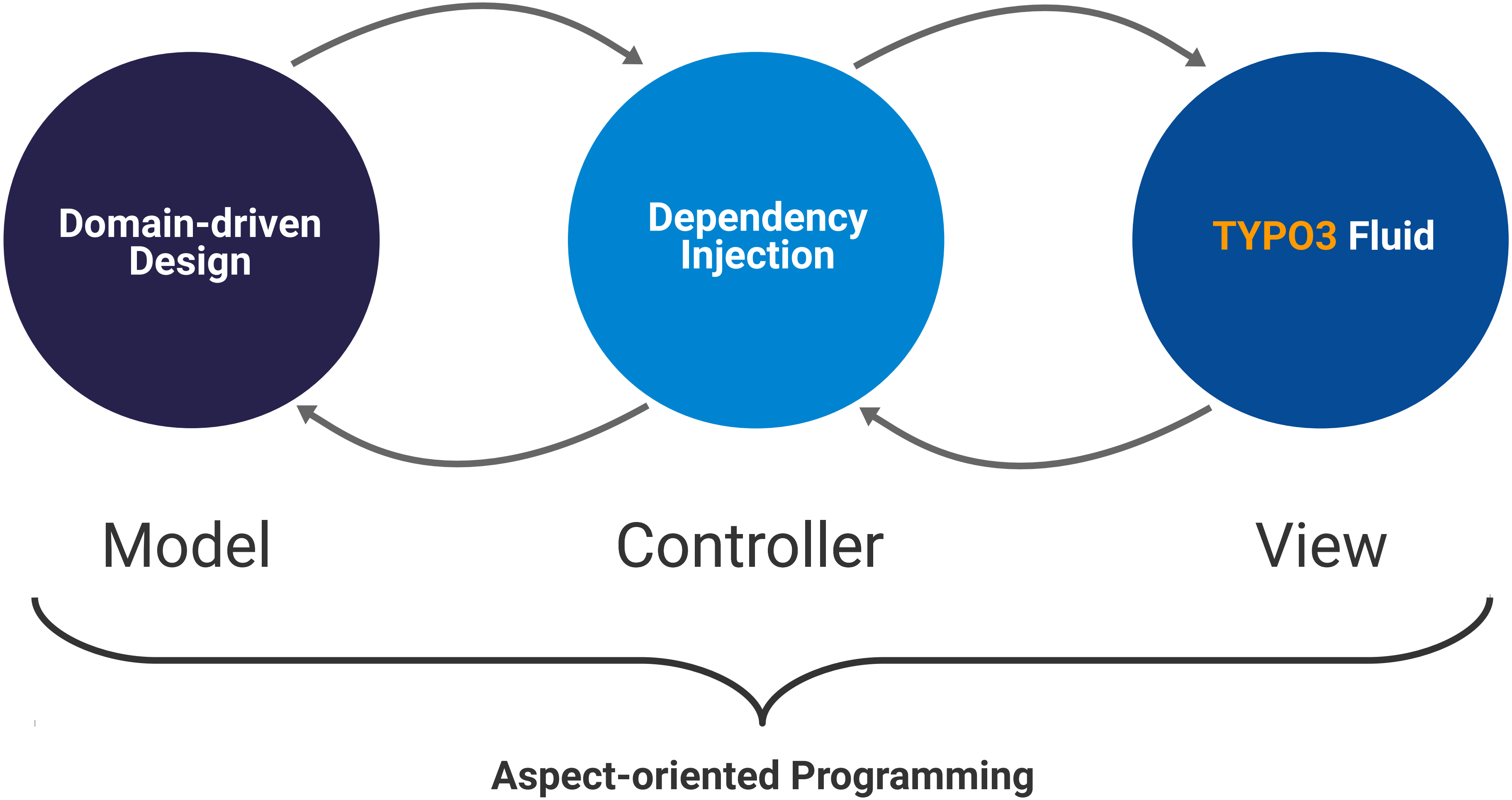
LoggingAspect->logAdvice()

Programmausführung

CowController->showAction()

```
/*
 * @Flow\Aspect
 */
class LoggingAspect {

    /**
     * @param JoinPointInterface $joinPoint
     * @Flow\After(”method(MY\PACKAGE\Controller\.*->.*())”)
     */
    public function logAdvice(JoinPointInterface $joinPoint) {
        Logger::log(
            $joinPoint->getClassName().
            ‘->‘.
            $joinPoint->getMethodName().
            ‘() has been executed‘
        );
    }

}
```


Logging Concern
Security Concern
1
2
3
4
5
6
1
2
1
Core Concern
domänenspezifisch



Domain-driven Design

Dependency Injection

TYPO3 Fluid

Model

Controller

View

Aspect-oriented Programming

FLOW

Security

Test-driven Development

Doctrine ORM

Package Management

Lokalisierung

Object Validation

CLI

Signals & Slots

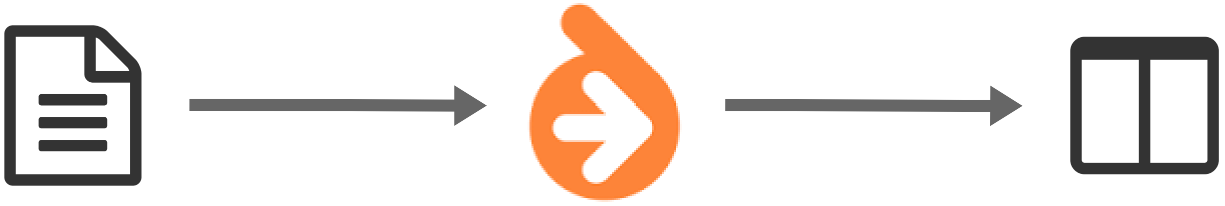
Resource Management

Cache

Vielen Dank für Eure Aufmerksamkeit

Fragen?

```
cowuser> ./flow doctrine:create
cowuser> ./flow doctrine:create
```

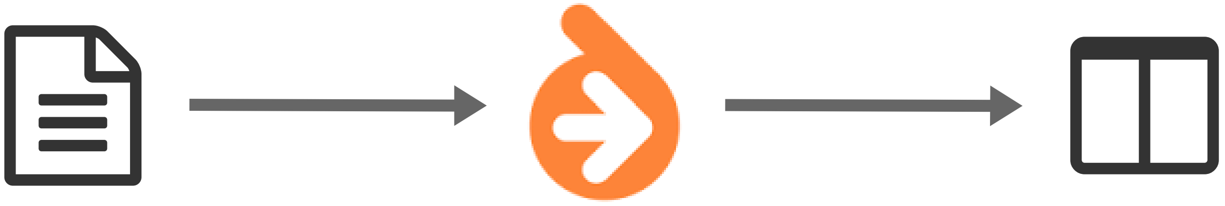


Model-
Klasse

ORM
(Doctrine)

relationales
Schema

```
cowuser> ./flow doctrine:create
```



Model-Klasse

ORM
(Doctrine)

relationales Schema

Routing

http://cows.local/cows/volma

```yaml
-
  name: 'single cow route'
  uriPattern: 'cows/{cow}'
  defaults:
    '@package':     'MY.PACKAGE'
    '@controller':  'Cow'
    '@action':      'show'
  routeParts:
    cow:
      objectType: 'MY\PACKAGE\Domain\Model\Cow'
```

eigene ViewHelper

…/MY.PACKAGE/Resources/Private/Templates/Cow/Show.html

```
{namespace cow=MY\PACKAGE\ViewHelpers}
<f:layout name="Default” />

<f:section name="Title”>{cow.name}</f:section>

<f:section name="Content”>
    <h1>{cow.name}</h1>
    <f:format.date format=”d.m.Y”>{cow.birthday}</f:format.date>
    <cow:isVolma cow="{cow}” />
</f:section>
```

```
class IsVolmaViewHelper extends \TYPO3\Fluid\Core\ViewHelper\AbstractViewHelper {

}
```

```
class IsVolmaViewHelper extends \TYPO3\Fluid\Core\ViewHelper\AbstractViewHelper {

    /**
     * @param \MY\PACKAGE\Domain\Model\Cow $cow
     */
    public function render($cow) {}

}
```

```
class IsVolmaViewHelper extends \TYPO3\Fluid\Core\ViewHelper\AbstractViewHelper {

    /**
     * @param \MY\PACKAGE\Domain\Model\Cow $cow
     */
    public function render($cow) {
        if($cow->getName() === ”Volma”){ return ”YES!”; }
    }

}
```