Stefan Tilkov | innoQ

From ad hoc Solutions to Sustainable Platforms

innoQ

innoQ Deutschland GmbH

Krischerstr. 100
40789 Monheim am Rhein
Germany
Phone: +49 2173 3366-0

innoQ Schweiz GmbH

Gewerbestr. 11
CH-6330 Cham
Switzerland
Phone: +41 41 743 0116

http://www.innoq.com

(... and yes, we're hiring)

(... specifically in Berlin)

Initial goals

Simplicity

Speed

Easy development

Maximum productivity

Long-term goals

Stability

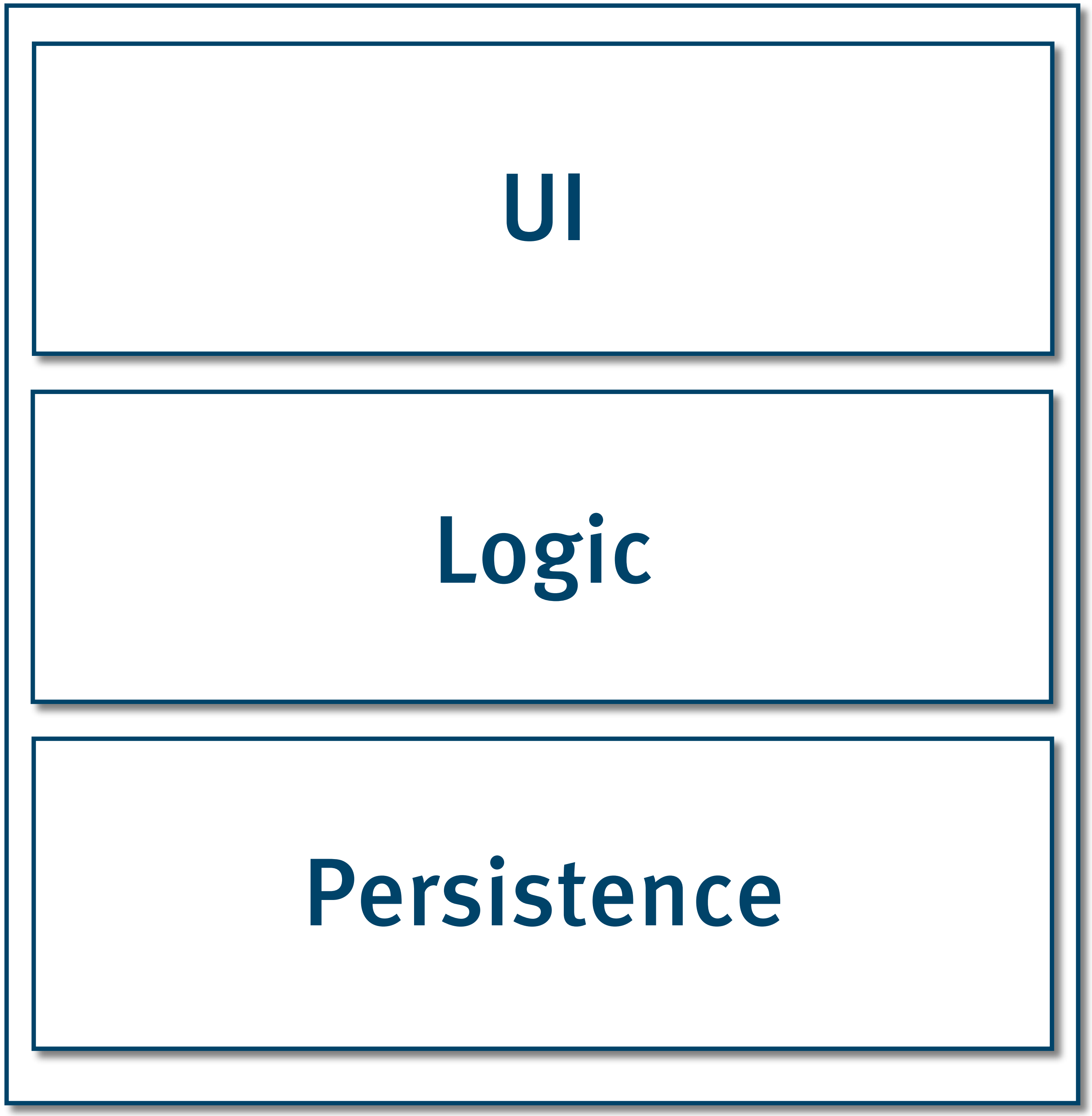
Scalability

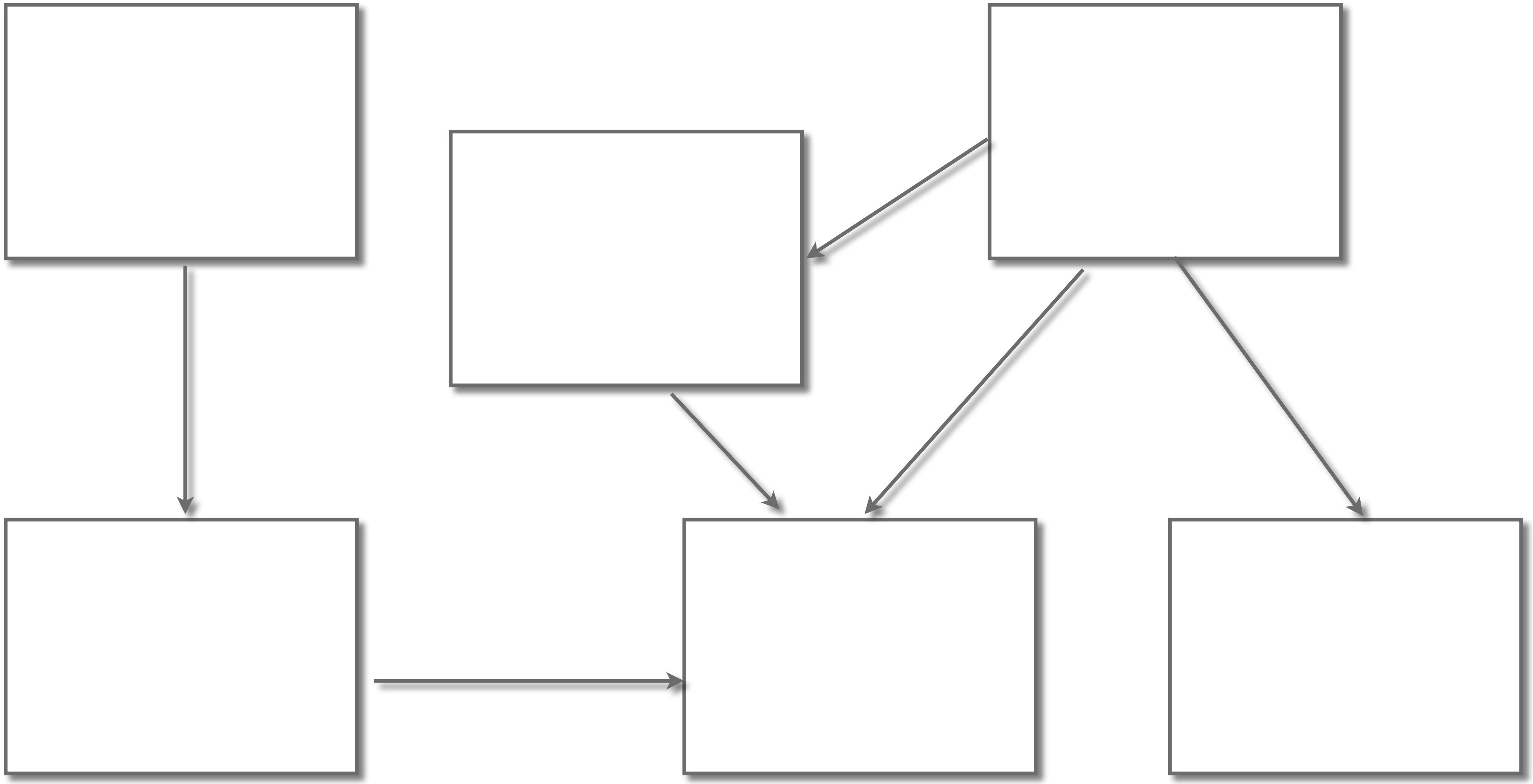
Maintainability

Decoupling

10 Recommendations

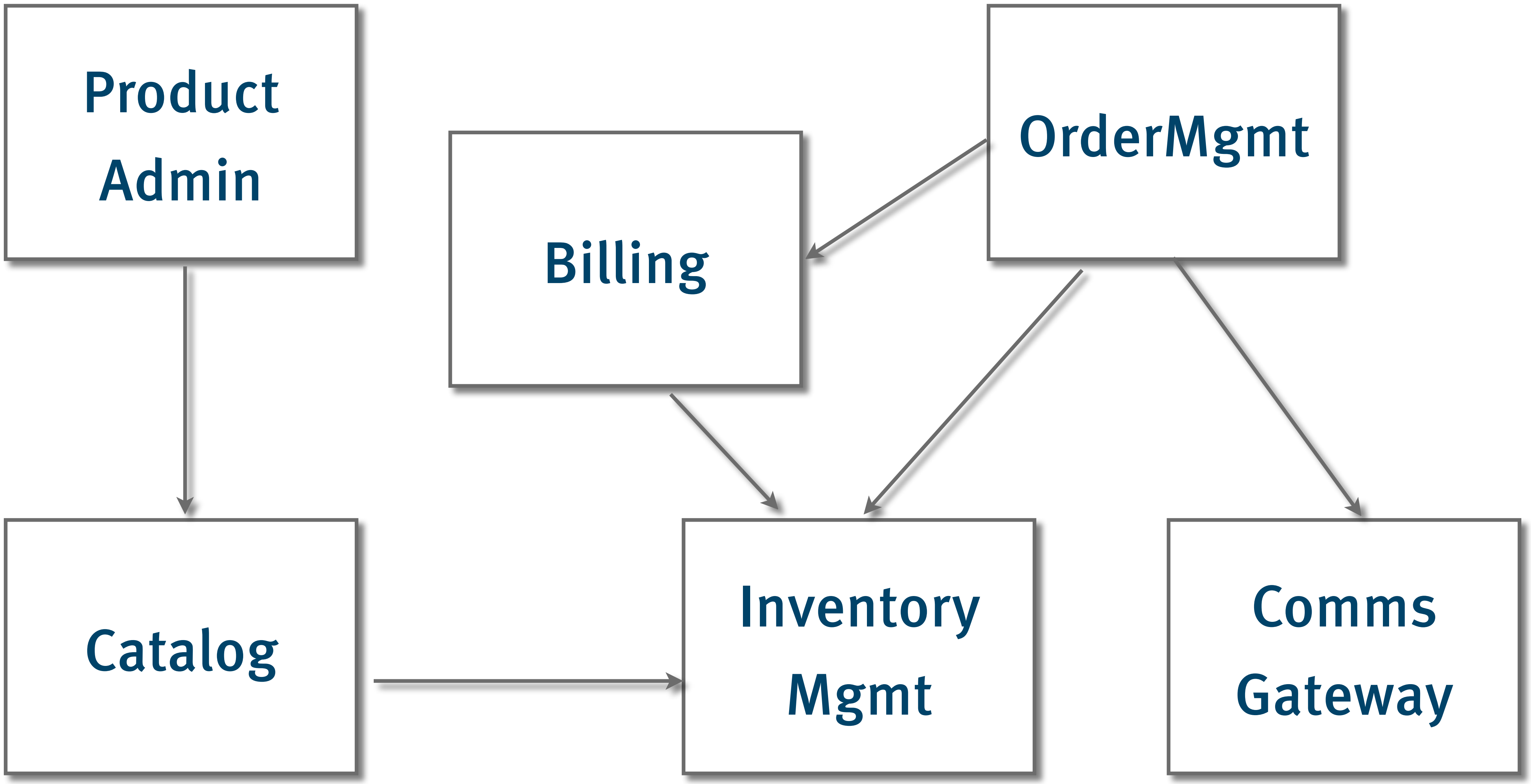
Systematic breakdown


UI
Logic
Persistence



#webinale

@stilkov


Product Admin
Billing
OrderMgmt
Catalog
Inventory Mgmt
Comms Gateway

Modules

Components

Subsystems

Libraries

Classes

Something vague and inexplicable

A system and its parts

How do we come up with system boundaries?

1 project ≡ 1 system?

1 site ≡ 1 system?

Modularization & Size (Ruby/Rails)

Size	Modularization
1-50 LOC*	single file
50-500 LOC	few files, few functions
500-1000 LOC	Library, class hierarchy
1000-2000 LOC	Framework + application
>2000 LOC	multiple applications

* LOC = Lines of Code

System characteristics

Separate (redundant) persistence

Internal, separate logic

Domain models & implementation strategies

Separate UI

Separate development & evolution

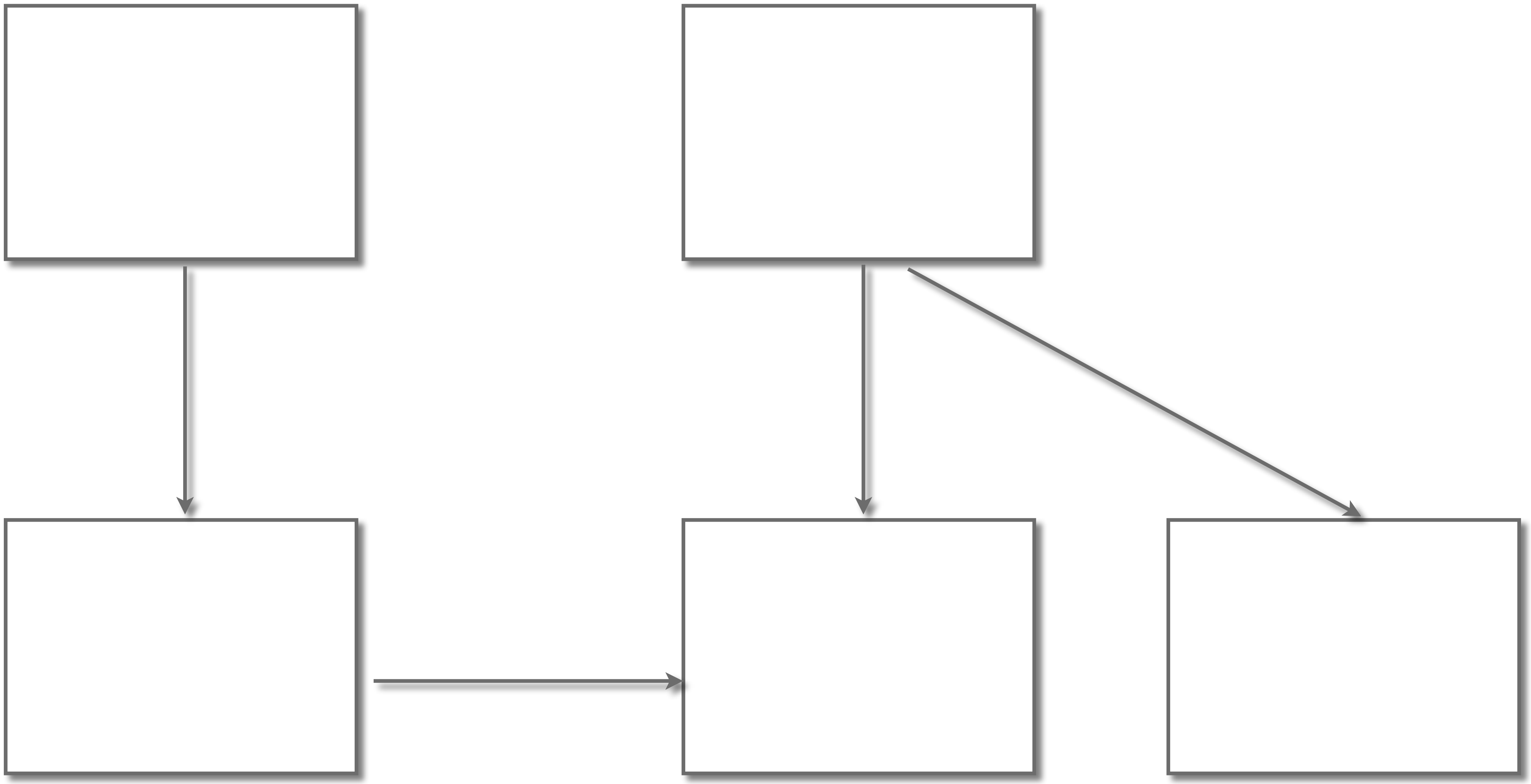
Autonomous operations

Limited interaction with other systems

Recommendation #1:

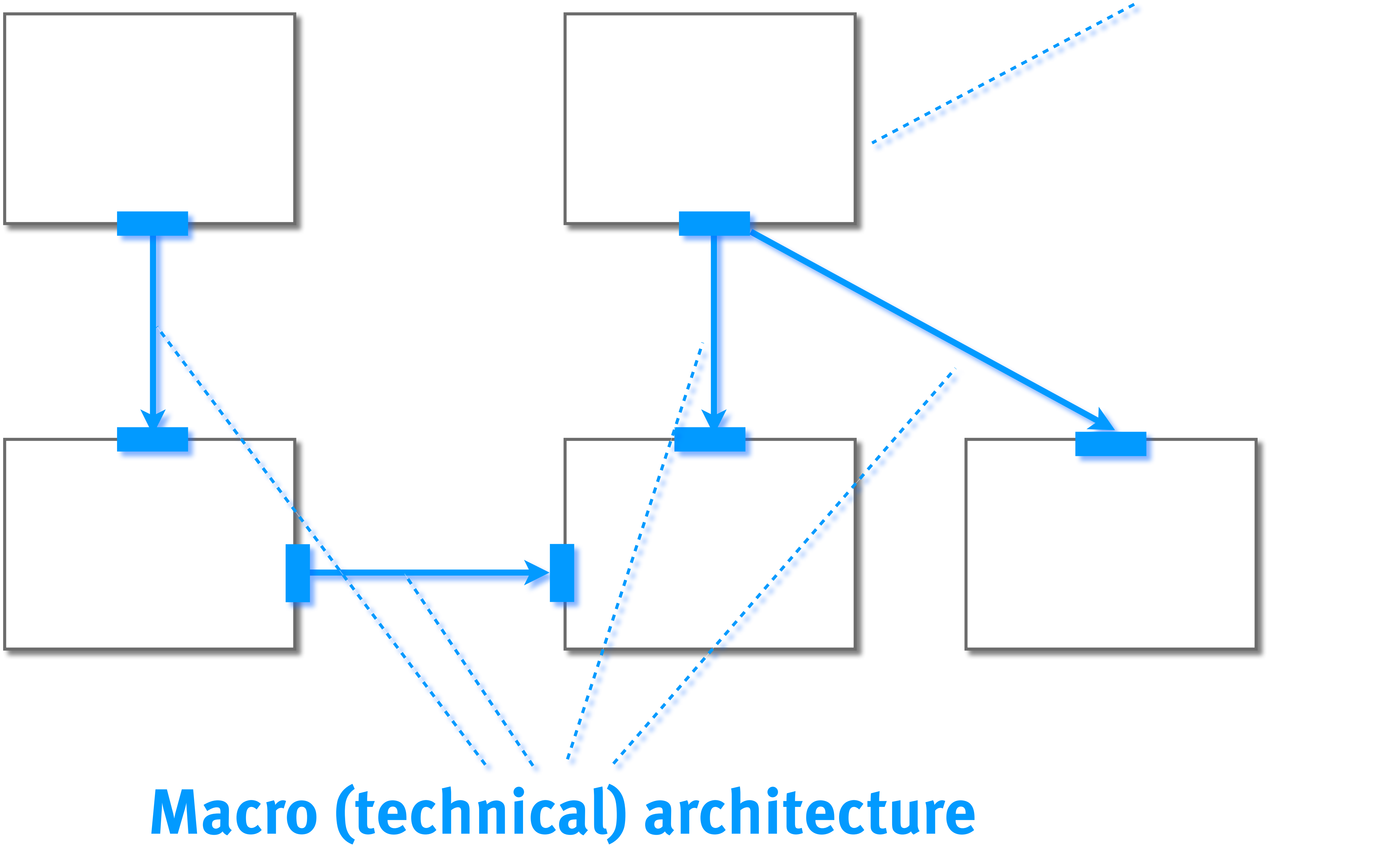
Build the right number of systems.

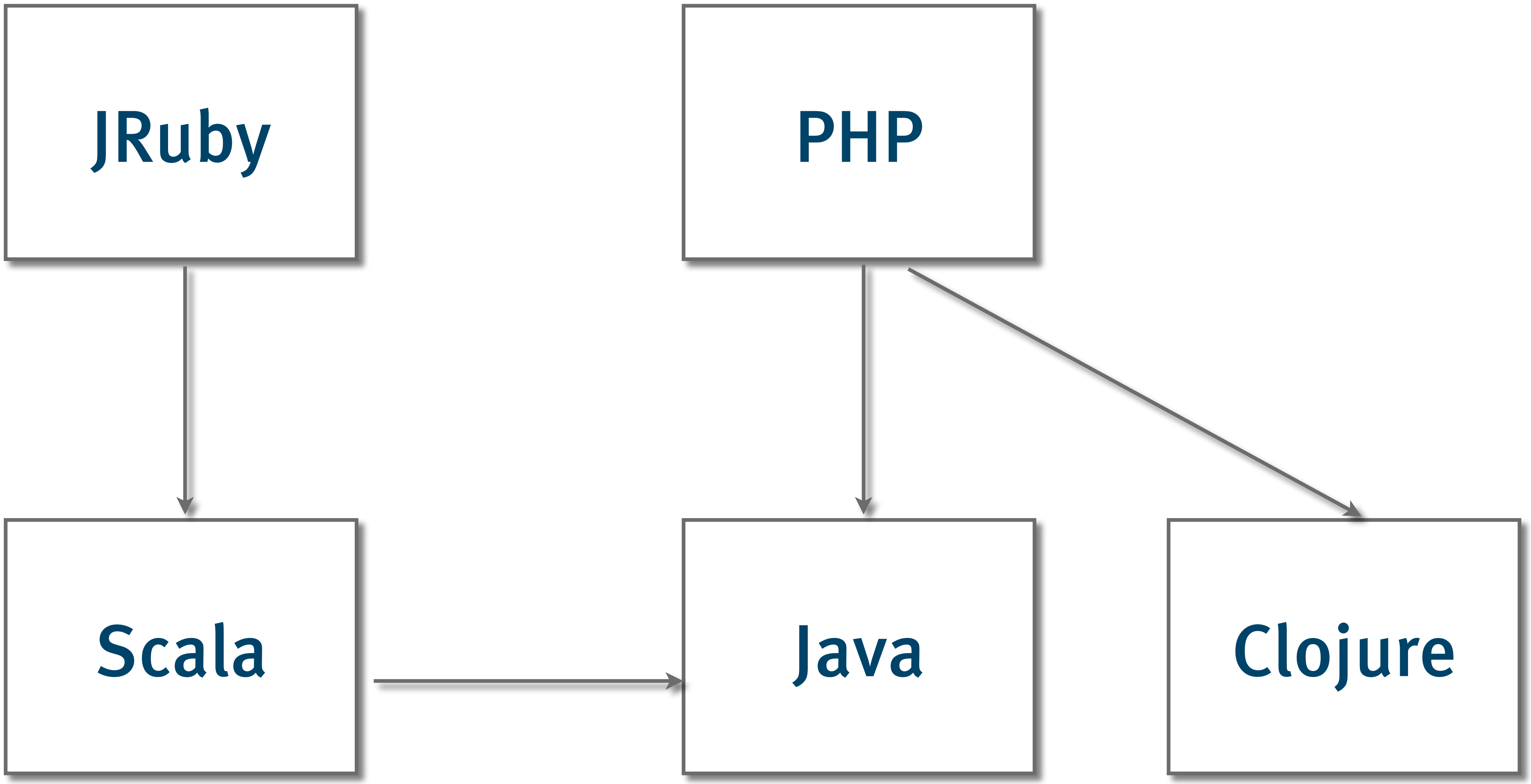
Separate rules, common assets

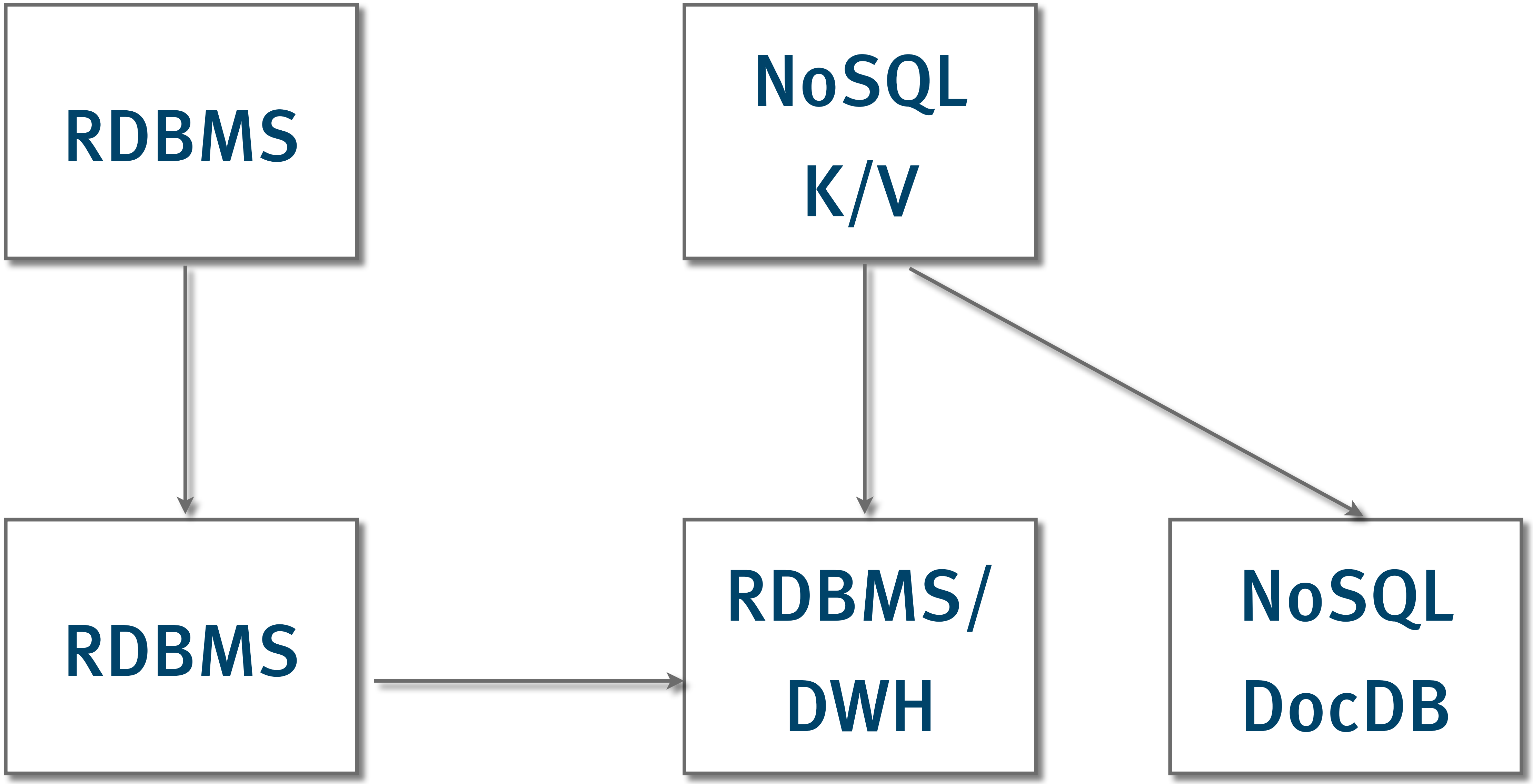


#webinale

@stilkov


Domain architecture
Macro (technical) architecture


JRuby
PHP
Scala
Java
Clojure



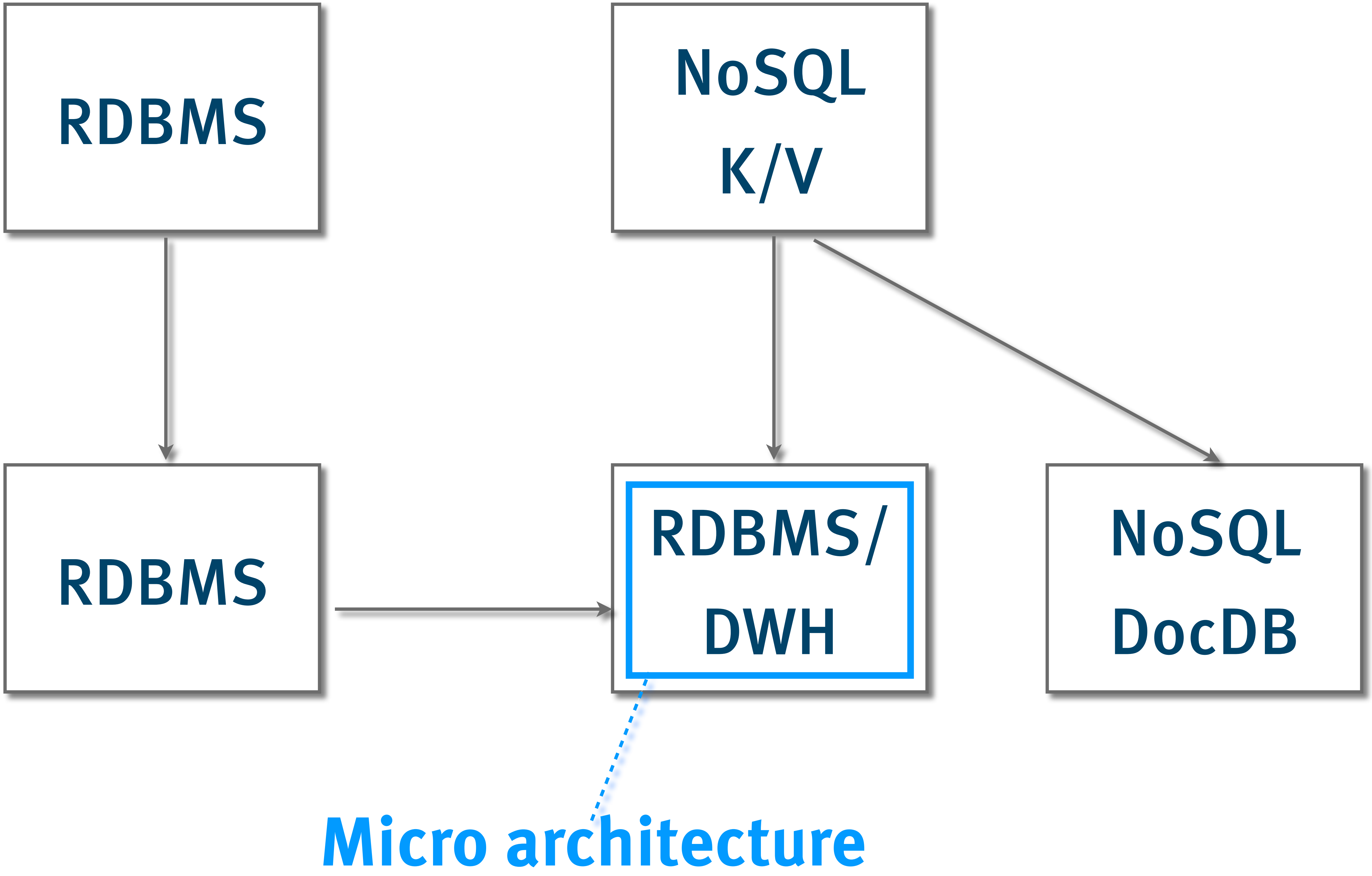
RDBMS

NoSQL K/V

RDBMS

RDBMS/ DWH

NoSQL DocDB



RDBMS
NoSQL K/V
RDBMS
RDBMS/ DWH
NoSQL DocDB
Micro architecture

Rules & Guidelines

Cross-system

Responsibilities

UI integration

Communication protocols

Data formats

Redundant data

BI interfaces

Logging, Monitoring

System-internal

Programming languages

Development tools

Frameworks

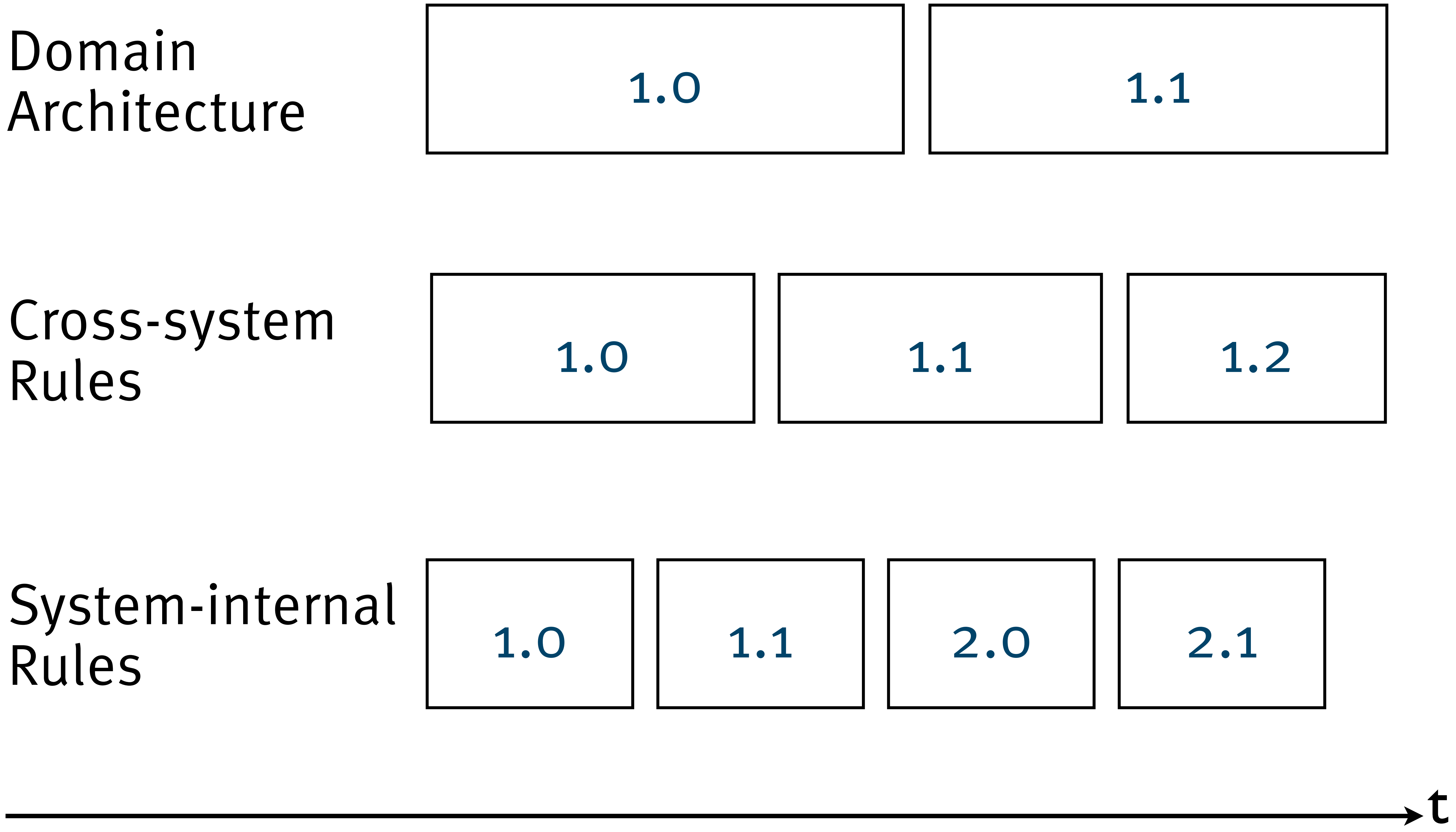
Process/Workflow control

Persistence

Design patterns

Coding guidelines

(Deployment, Operations)



Domain Architecture

1.0

1.1

Cross-system Rules

1.0

1.1

1.2

System-internal Rules

1.0

1.1

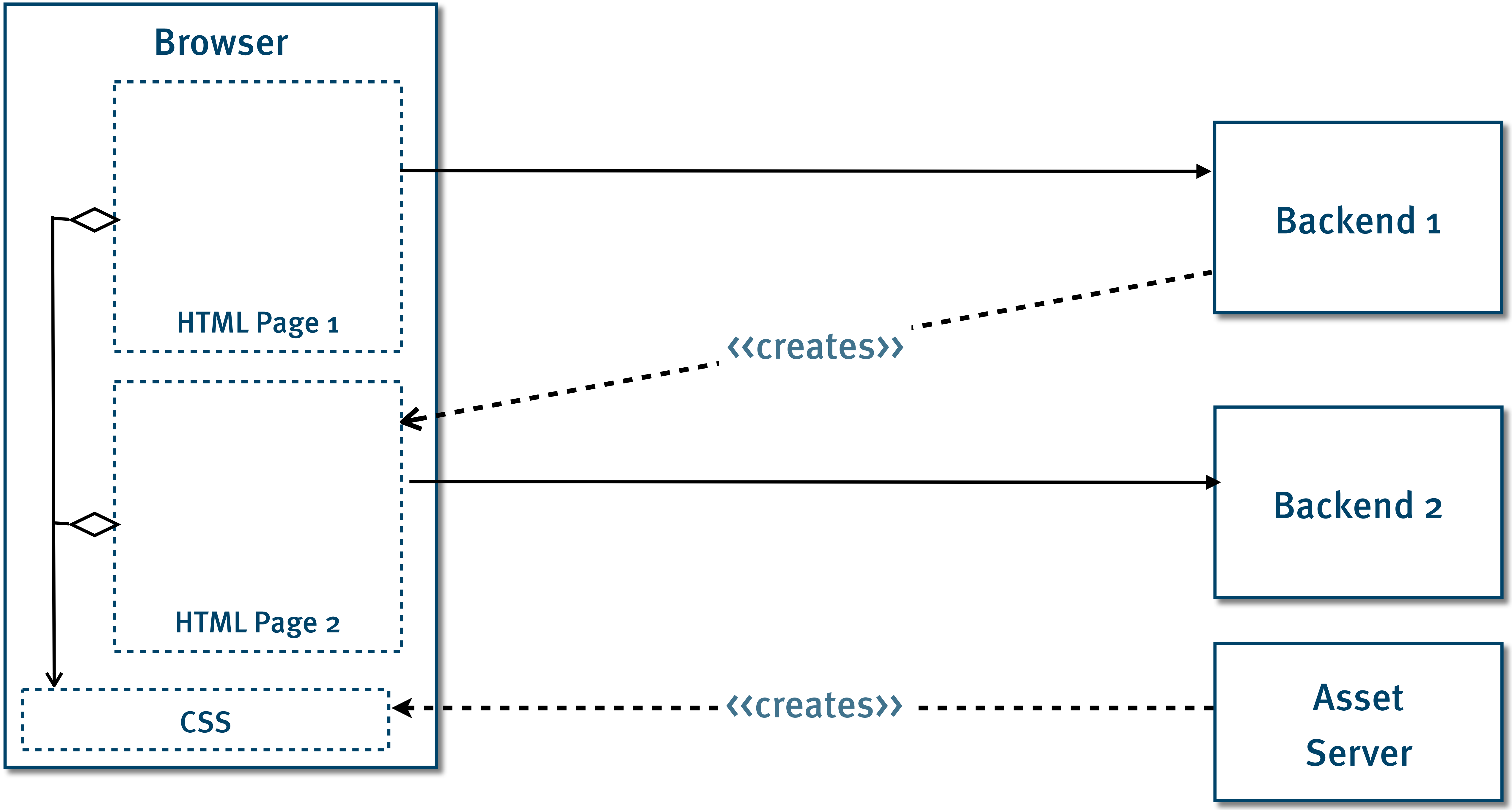
2.0

2.1

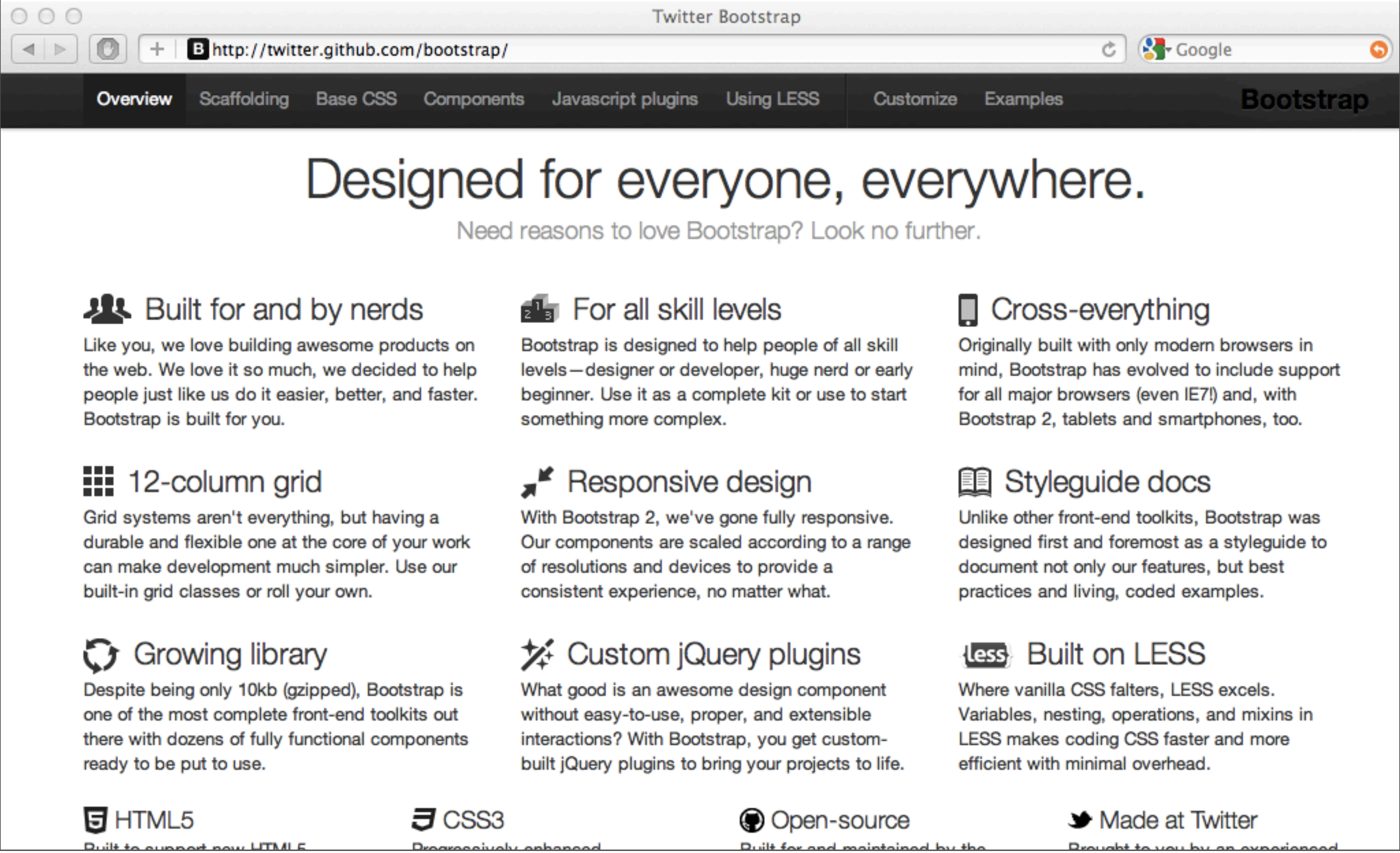
t

Common Assets

Links


Browser
HTML Page 1
HTML Page 2
CSS
Backend 1
Backend 2
Asset Server
«creates»
«creates»

Asset Example



Designed for everyone, everywhere.

Need reasons to love Bootstrap? Look no further.

Built for and by nerds

Like you, we love building awesome products on the web. We love it so much, we decided to help people just like us do it easier, better, and faster. Bootstrap is built for you.

For all skill levels

Bootstrap is designed to help people of all skill levels—designer or developer, huge nerd or early beginner. Use it as a complete kit or use to start something more complex.

Cross-everything

Originally built with only modern browsers in mind, Bootstrap has evolved to include support for all major browsers (even IE7!) and, with Bootstrap 2, tablets and smartphones, too.

12-column grid

Grid systems aren't everything, but having a durable and flexible one at the core of your work can make development much simpler. Use our built-in grid classes or roll your own.

Responsive design

With Bootstrap 2, we've gone fully responsive. Our components are scaled according to a range of resolutions and devices to provide a consistent experience, no matter what.

Styleguide docs

Unlike other front-end toolkits, Bootstrap was designed first and foremost as a styleguide to document not only our features, but best practices and living, coded examples.

Growing library

Despite being only 10kb (gzipped), Bootstrap is one of the most complete front-end toolkits out there with dozens of fully functional components ready to be put to use.

Custom jQuery plugins

What good is an awesome design component without easy-to-use, proper, and extensible interactions? With Bootstrap, you get custom-built jQuery plugins to bring your projects to life.

Built on LESS

Where vanilla CSS falters, LESS excels. Variables, nesting, operations, and mixins in LESS makes coding CSS faster and more efficient with minimal overhead.

HTML5

Built to support new HTML5

CSS3

Progressively enhanced

Open-source

Built for and maintained by the

Made at Twitter

Brought to you by an experienced

Recommendation #2:

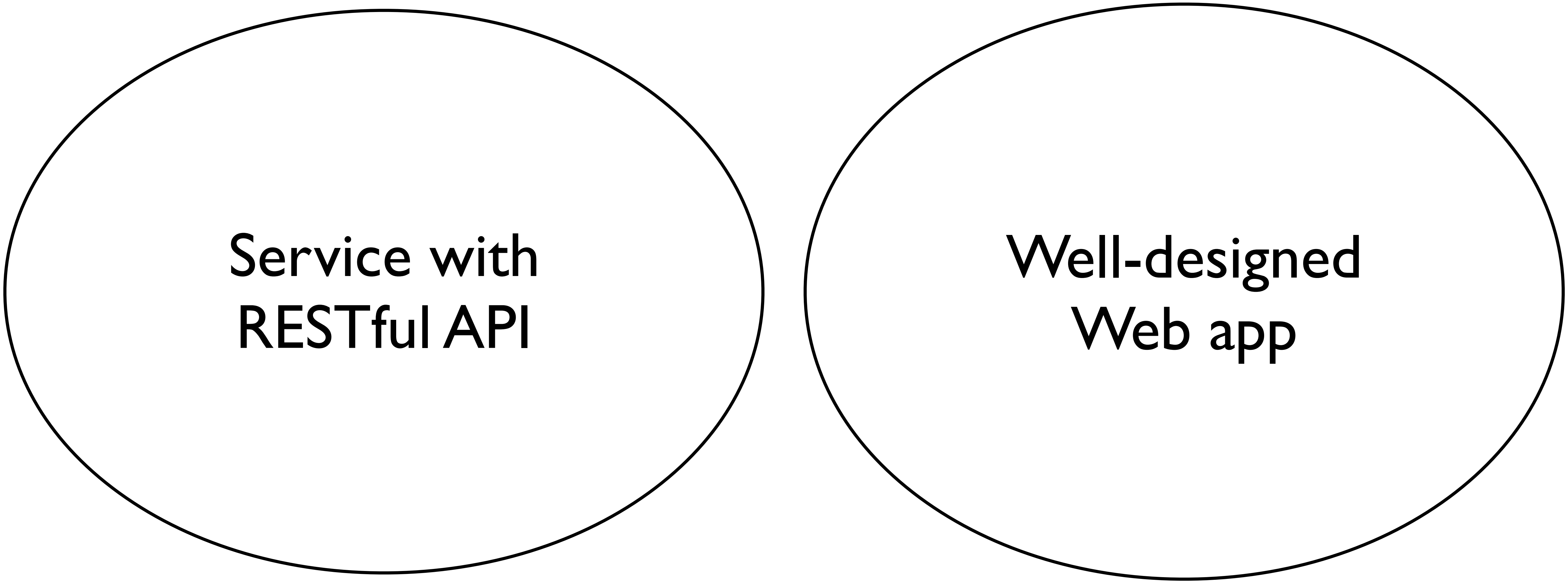
Avoid both chaos and standardization paralyzation.

Services & APIs

The Bezos Mandate*

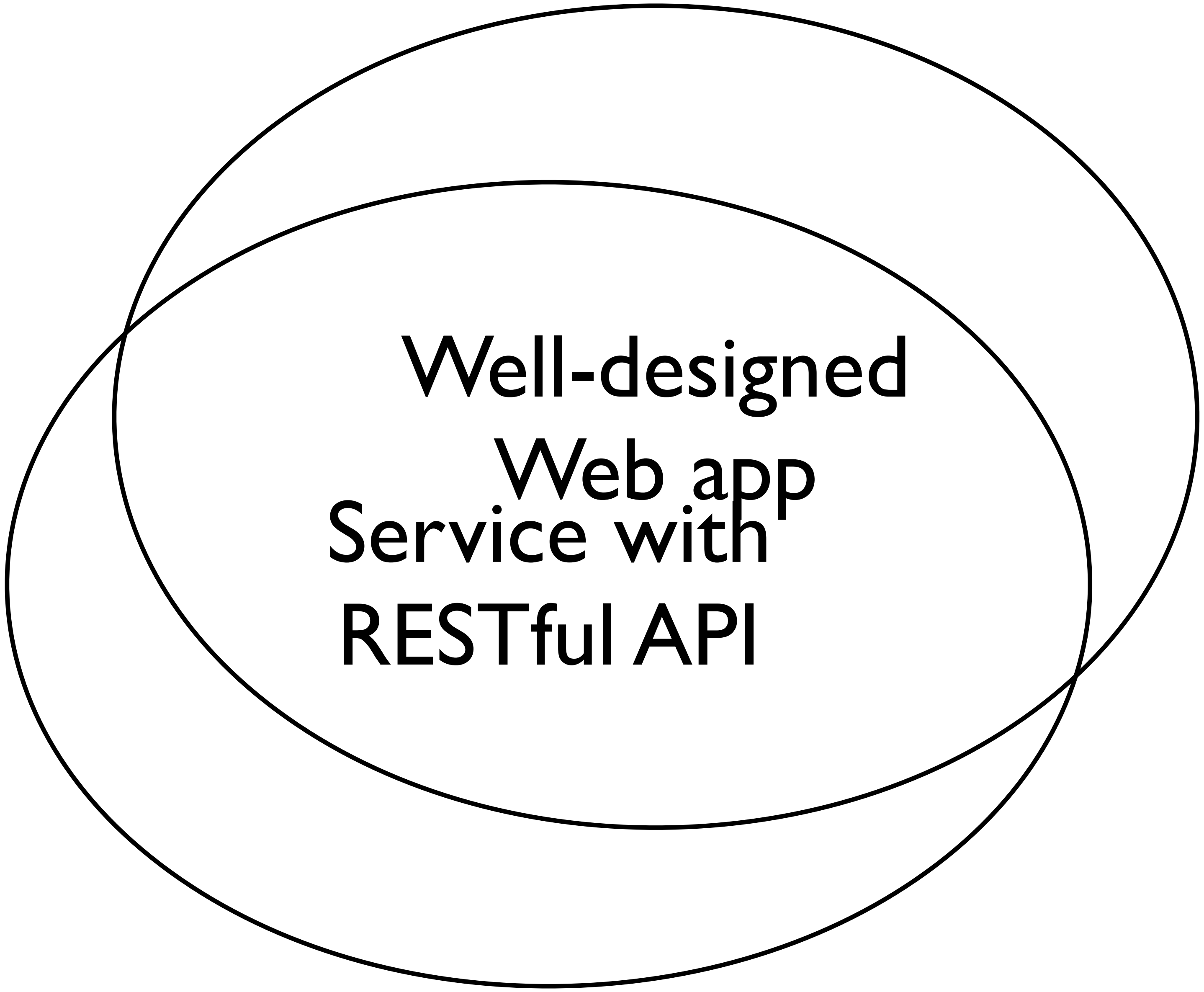
1. All teams will expose **data and functionality** through **service interfaces**.
2. Teams must communicate with each other through these interfaces.
3. No other form of interprocess communication allowed, **only service calls over the net**.
4. The protocols used are unimportant.
5. All service interfaces to be designed from the ground up to be **externalizable**. No exceptions.
6. Anyone who doesn't do this will be fired.
7. Thank you; have a nice day!

***Steve Yegge, https://plus.google.com/112678702228711889851/posts/eVeouesvaVX**



Service with RESTful API

Well-designed Web app



Well-designed Web app

Service with RESTful API

Recommendation #3:

Eat your own API dog food.

Identity

Users

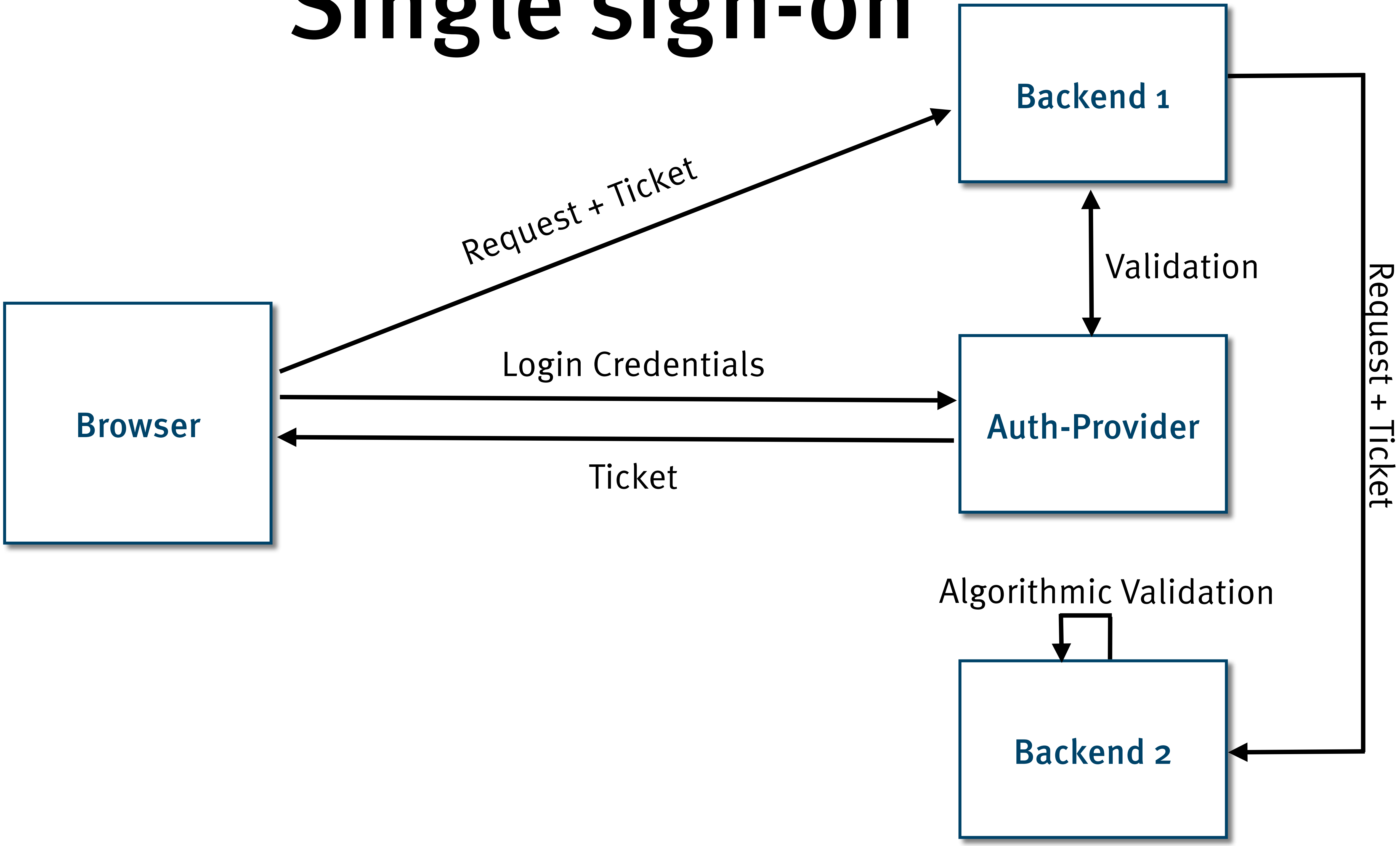
Products

Identity

Users

Products

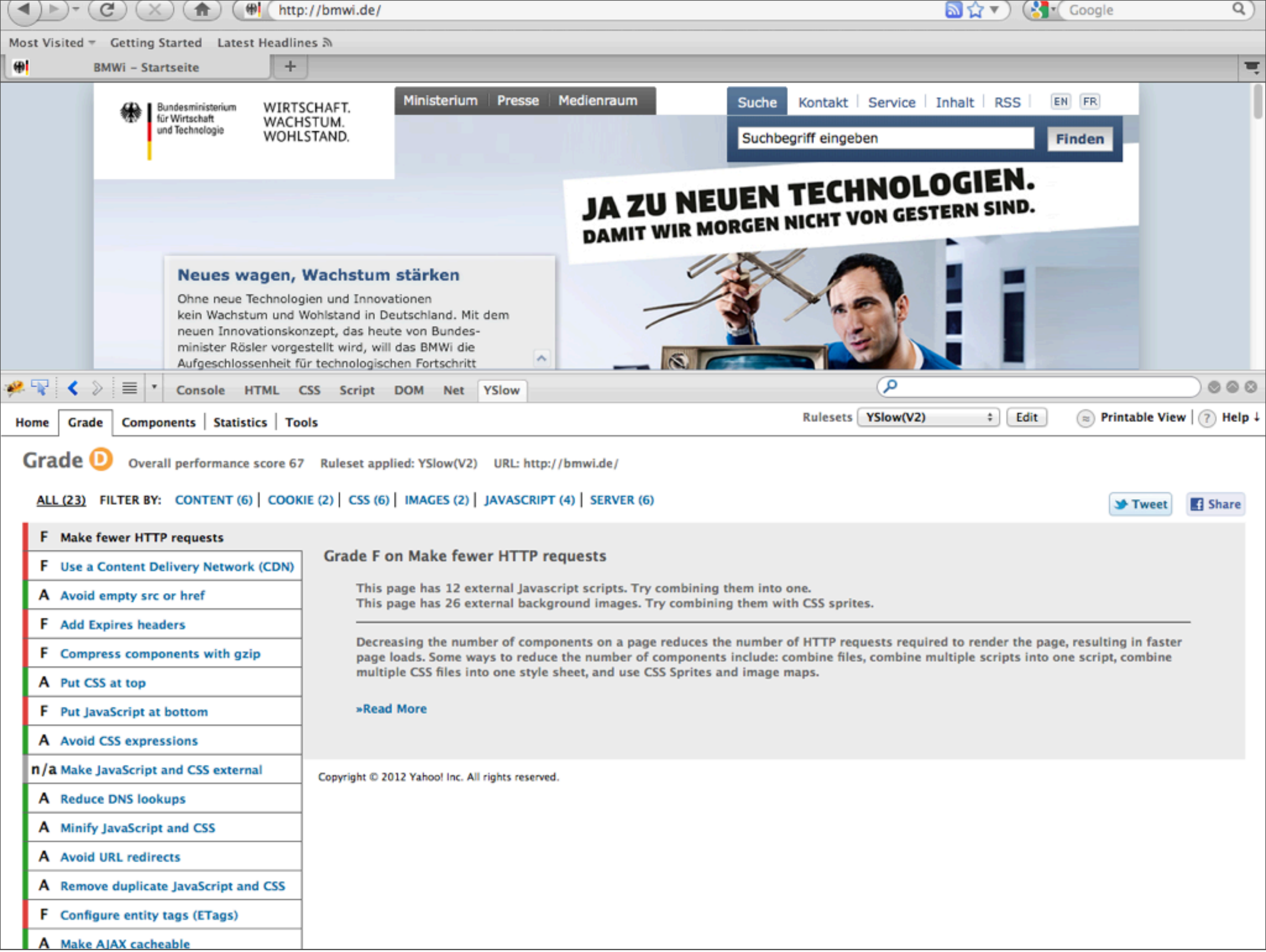
Single sign-on



Recommendation #4:

Separate user identity, sign-up and self-care from product dependencies.

Understand your requests



Neues wagen, Wachstum stärken

Ohne neue Technologien und Innovationen kein Wachstum und Wohlstand in Deutschland. Mit dem neuen Innovationskonzept, das heute von Bundesminister Rösler vorgestellt wird, will das BMWi die Aufgeschlossenheit für technologischen Fortschritt

Grade D Overall performance score 67 Ruleset applied: YSlow(V2) URL: http://bmwi.de/

ALL (23) FILTER BY: CONTENT (6) | COOKIE (2) | CSS (6) | IMAGES (2) | JAVASCRIPT (4) | SERVER (6)

F	Make fewer HTTP requests
F	Use a Content Delivery Network (CDN)
A	Avoid empty src or href
F	Add Expires headers
F	Compress components with gzip
A	Put CSS at top
F	Put JavaScript at bottom
A	Avoid CSS expressions
n/a	Make JavaScript and CSS external
A	Reduce DNS lookups
A	Minify JavaScript and CSS
A	Avoid URL redirects
A	Remove duplicate JavaScript and CSS
F	Configure entity tags (ETags)
A	Make AJAX cacheable

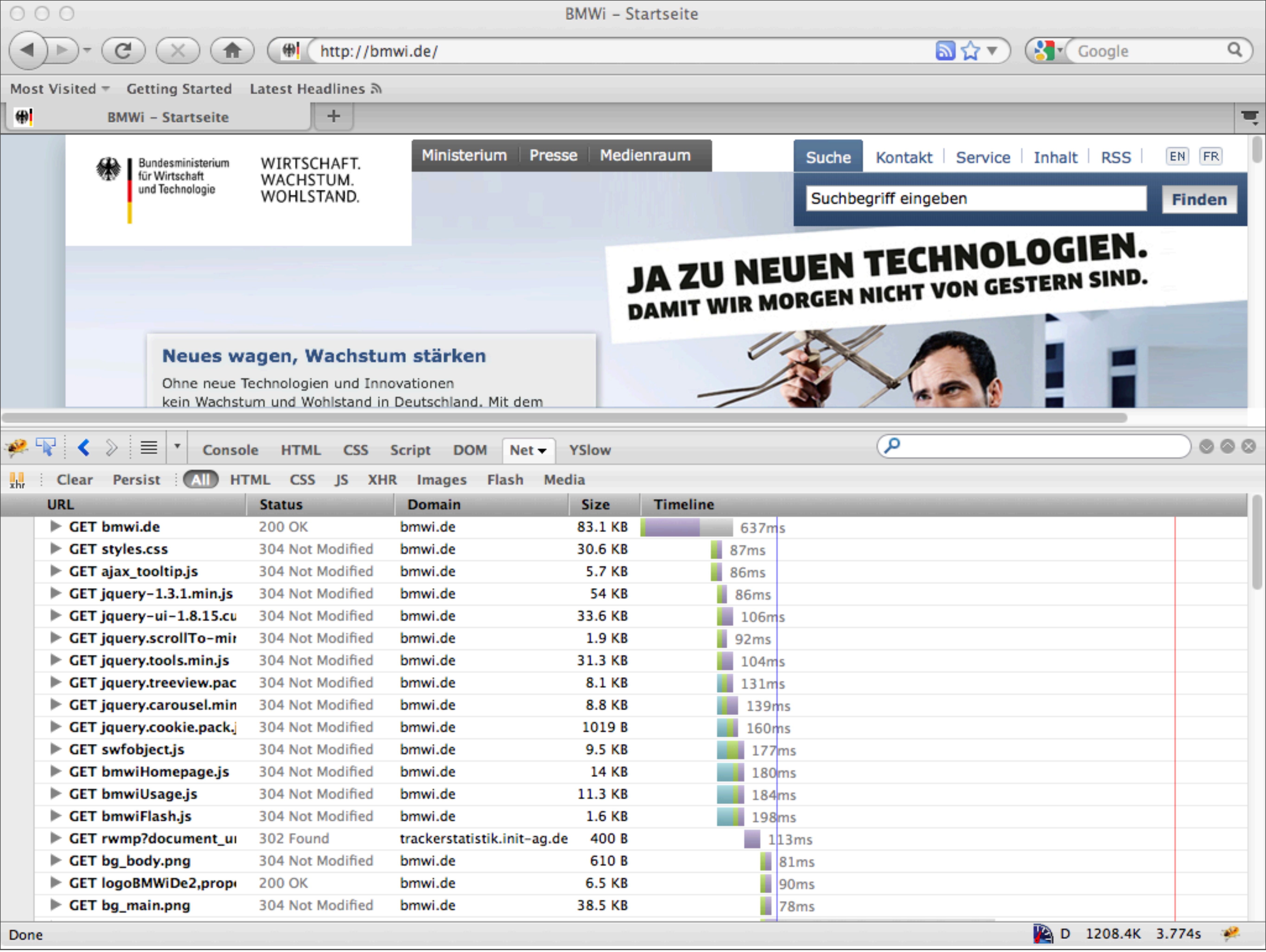
Grade F on Make fewer HTTP requests

This page has 12 external Javascript scripts. Try combining them into one.
This page has 26 external background images. Try combining them with CSS sprites.

Decreasing the number of components on a page reduces the number of HTTP requests required to render the page, resulting in faster page loads. Some ways to reduce the number of components include: combine files, combine multiple scripts into one script, combine multiple CSS files into one style sheet, and use CSS Sprites and image maps.

»Read More

Copyright © 2012 Yahoo! Inc. All rights reserved.



Neues wagen, Wachstum stärken

Ohne neue Technologien und Innovationen
kein Wachstum und Wohlstand in Deutschland. Mit dem

URL	Status	Domain	Size	Timeline
GET bmwi.de	200 OK	bmwi.de	83.1 KB	637ms
GET styles.css	304 Not Modified	bmwi.de	30.6 KB	87ms
GET ajax_tooltip.js	304 Not Modified	bmwi.de	5.7 KB	86ms
GET jquery-1.3.1.min.js	304 Not Modified	bmwi.de	54 KB	86ms
GET jquery-ui-1.8.15.cu	304 Not Modified	bmwi.de	33.6 KB	106ms
GET jquery.scrollTo-mi	304 Not Modified	bmwi.de	1.9 KB	92ms
GET jquery.tools.min.js	304 Not Modified	bmwi.de	31.3 KB	104ms
GET jquery.treeview.pac	304 Not Modified	bmwi.de	8.1 KB	131ms
GET jquery.carousel.min	304 Not Modified	bmwi.de	8.8 KB	139ms
GET jquery.cookie.pack.j	304 Not Modified	bmwi.de	1019 B	160ms
GET swfobject.js	304 Not Modified	bmwi.de	9.5 KB	177ms
GET bmwiHomepage.js	304 Not Modified	bmwi.de	14 KB	180ms
GET bmwiUsage.js	304 Not Modified	bmwi.de	11.3 KB	184ms
GET bmwiFlash.js	304 Not Modified	bmwi.de	1.6 KB	198ms
GET rwmp?document_u	302 Found	trackerstatistik.init-ag.de	400 B	113ms
GET bg_body.png	304 Not Modified	bmwi.de	610 B	81ms
GET logoBMWiDe2,prop	200 OK	bmwi.de	6.5 KB	90ms
GET bg_main.png	304 Not Modified	bmwi.de	38.5 KB	78ms

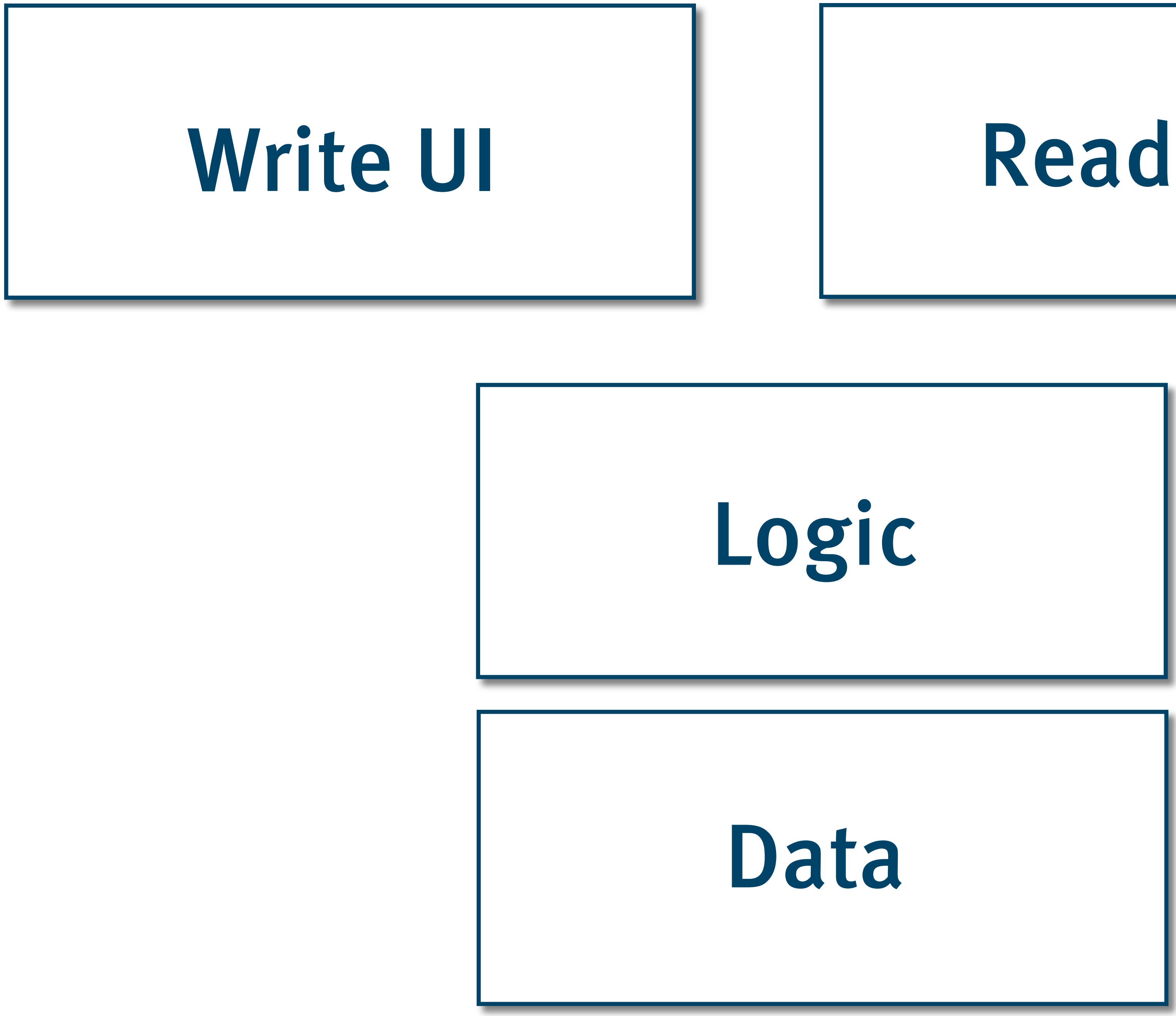
Recommendation #5:

Pick the low-hanging fruit of front-end performance optimizations.

Separate Command & Query Paths

CQRS

(Command Query Responsibility Segregation)



Write UI

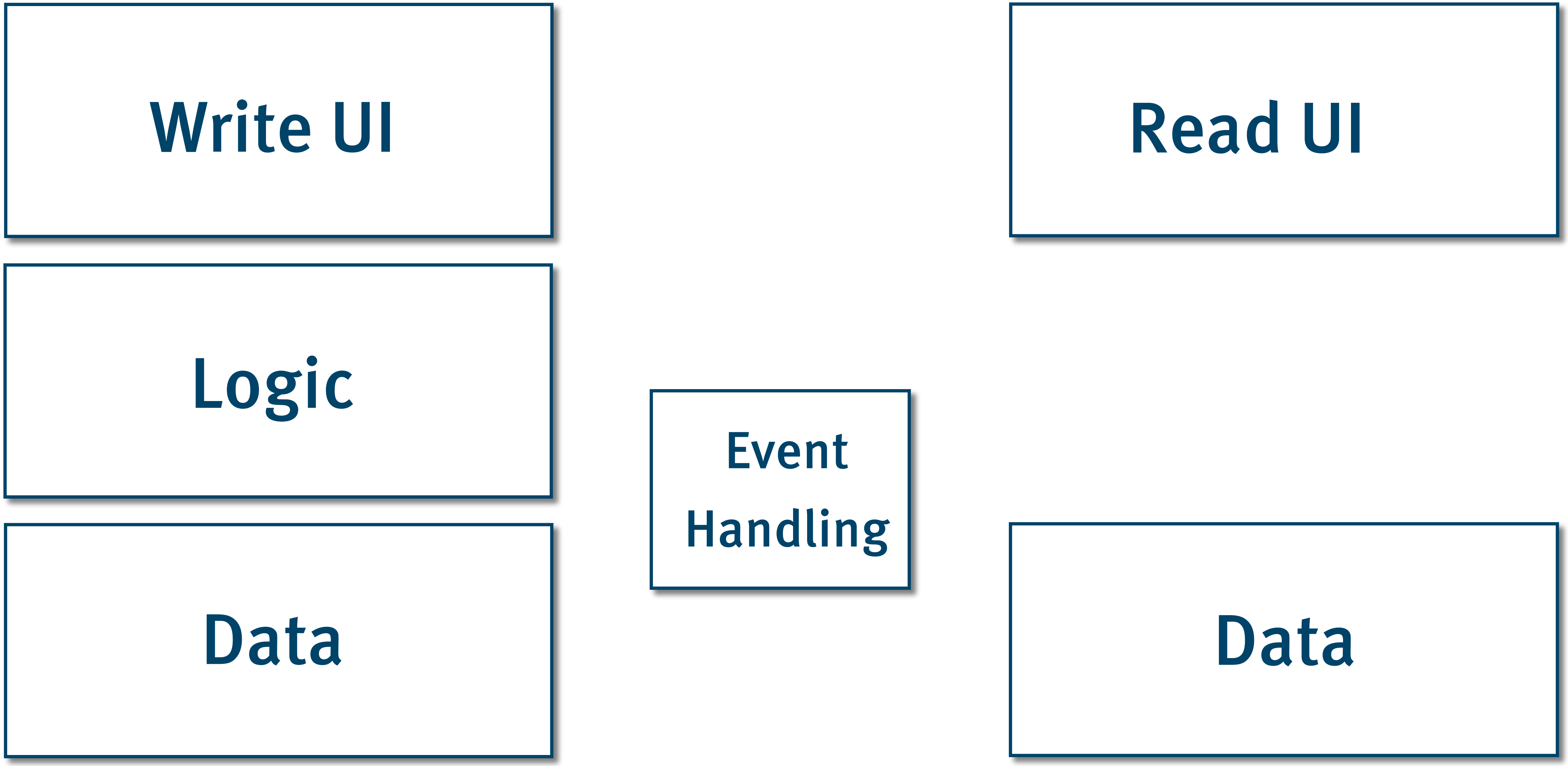
Read UI

Logic

Data

CQRS

(Command Query Responsibility Segregation)



Write UI

Logic

Data

Event Handling

Read UI

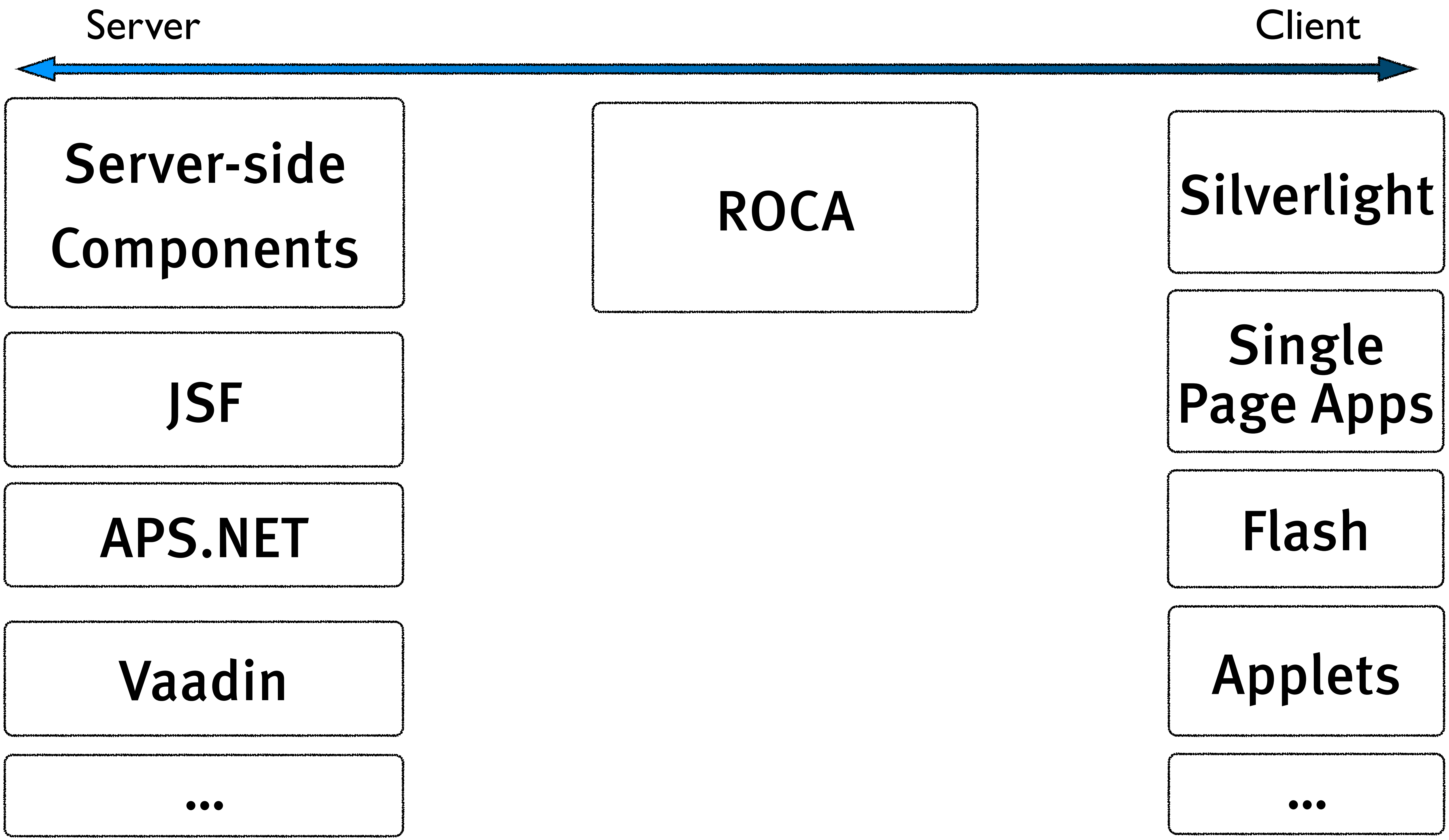
Data

Recommendation #6:

Don’t bother readers with write complexity.

Learn how to build Web UIs

From server to client



Server

Client

Server-side Components

JSF

APS.NET

Vaadin

...

ROCA

Silverlight

Single Page Apps

Flash

Applets

...

ROCA

http://roca-style.org/

Do this:

- REST principles
- Server-side application logic
- Server-side, structured HTML generation
- Unobtrusive JavaScript
- Progressive enhancement

ROCA

http://roca-style.org/

Avoid this:

- Broken accessibility
- Broken back/forward/refresh
- No bookmarks/links
- JS “engines” in client
- Hashbangs et al.

Recommendation #7:

Follow "pure" Web principles to get performance, scale and longevity.

Automate. Seriously.

Testing

Delivery

Operations

Back-office tasks

Recommendation #8:

Automate everything or you will be hurt.

Design for scale with care

NoSQL

mongoDB

{name: "mongo", type:"DB"}

APACHE HBASE

CouchBase

riak

redis

Cassandra

Servers and Caches

Cloud Platforms

Transient

Sharded

Unshared

Horizontally Scalable

Ye Good Ol’ SQL RDBMS

Recommendation #9:

You can use new-fangled stuff, but you might not have to.

Agility

#webinale

@stilkov

Recommendation #10:

Don’t become too rigid –

retain the right to be chaotic.

Thank you!

Q&A

innoQ

innoQ Deutschland GmbH

Krischerstr. 100
40789 Monheim am Rhein
Germany
Phone: +49 2173 3366-0
http://www.innoq.com

innoQ Schweiz GmbH

Gewerbestr. 11
CH-6330 Cham
Switzerland
Phone: +41 41 743 0116
info@innoq.com