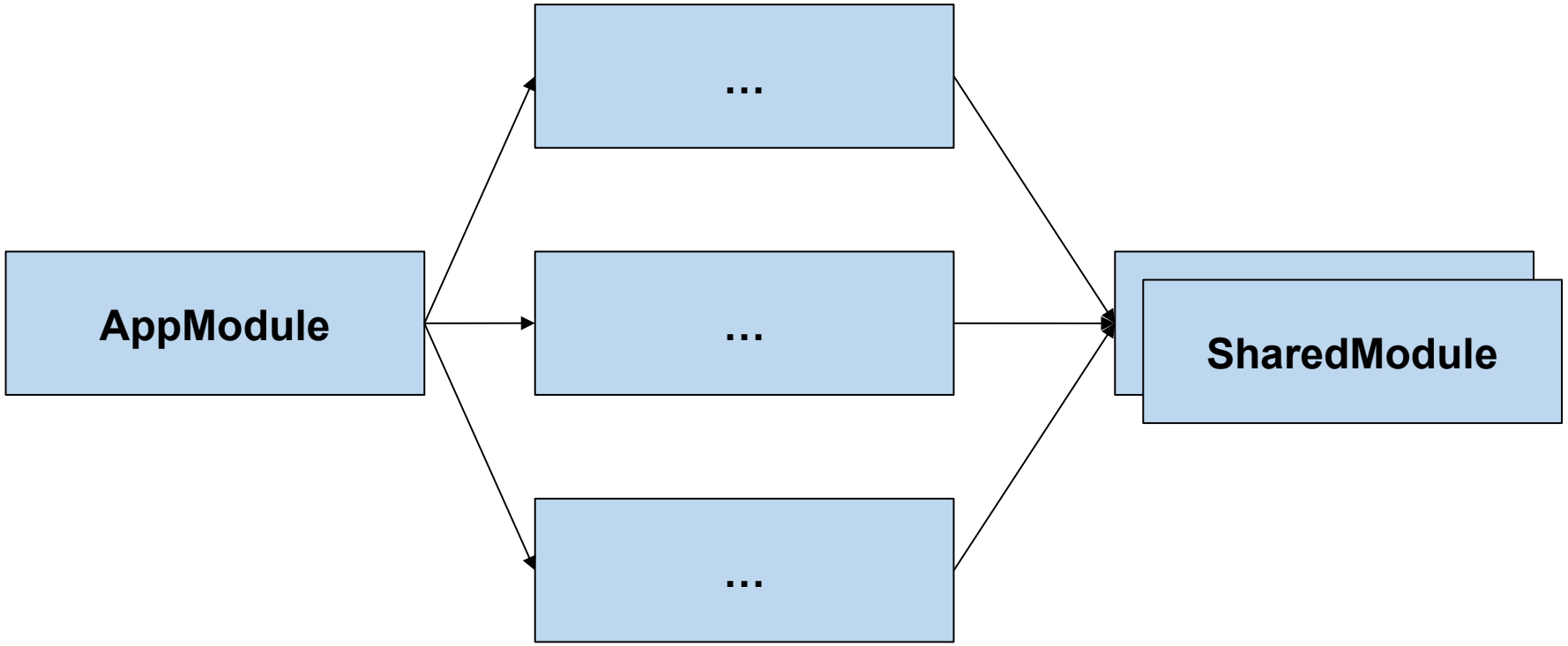
ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE

Angular Architektur-Workshop: Strategic Design and Micro Frontends

ANGULARarchitects.io

Typical Module Structure



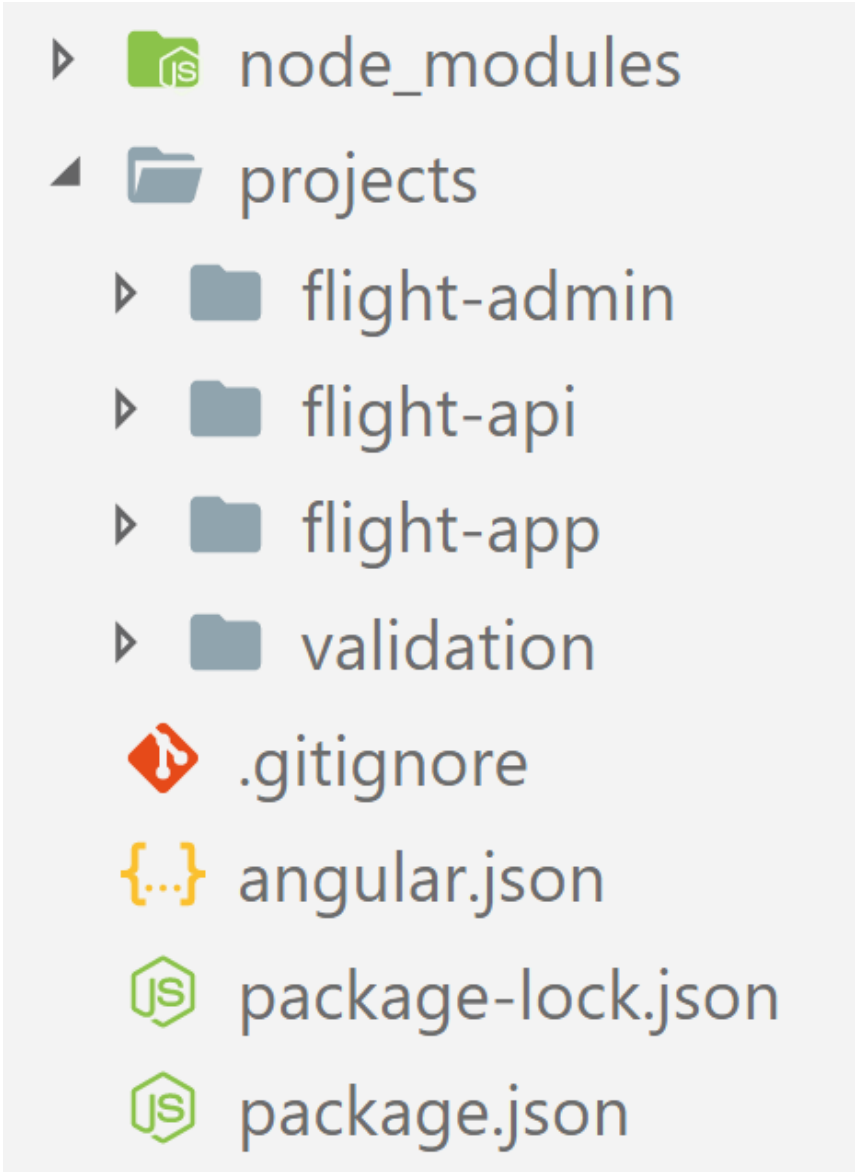
Root Module **Feature Modules** **Shared Modules**

Contents

- (npm-)Packages
- Nx Monorepos
- Strategic Design and DDD
- Microfrontends

Monorepos

Monorepo Structure



Advantages

- Everyone uses the latest versions
- No version conflicts
- No burden with distributing libs

Two Flavors

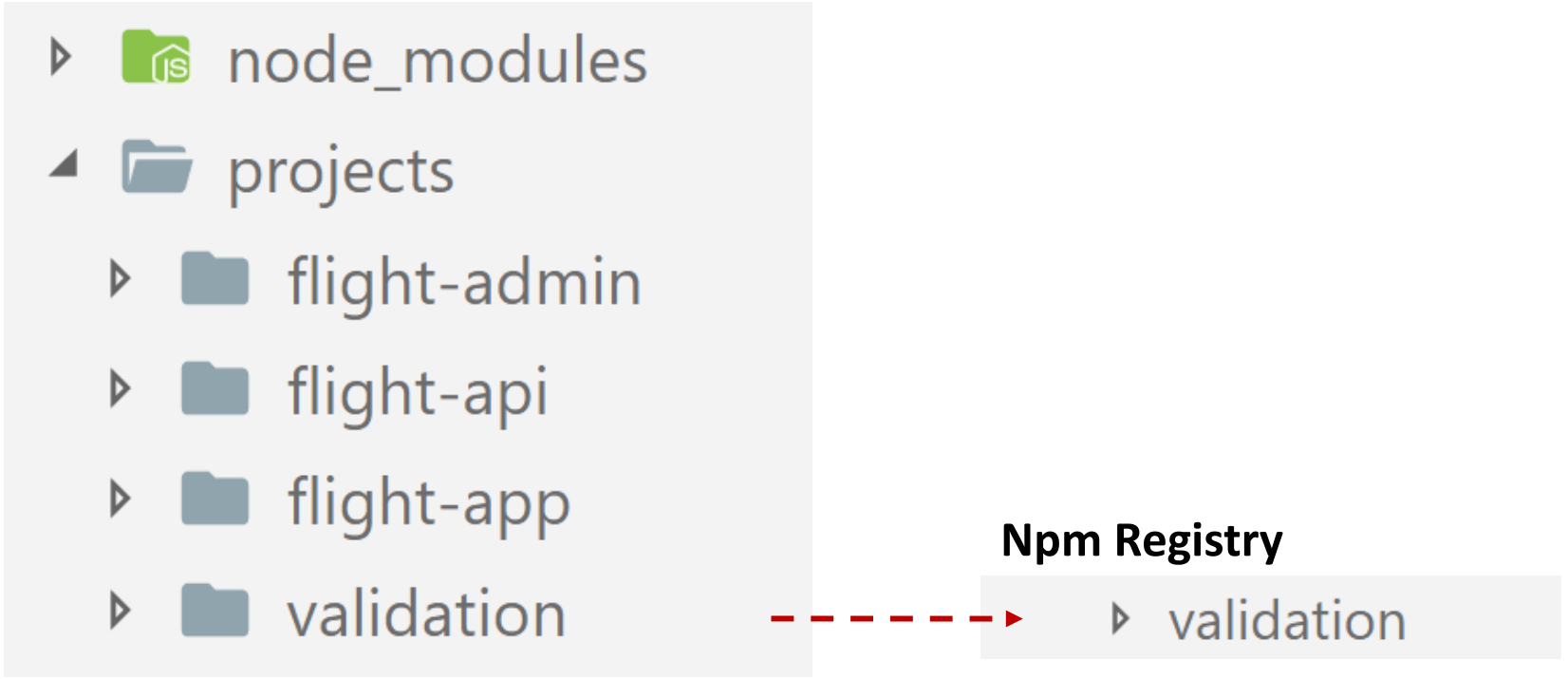
Project Monorepo

- Like Workspaces/Solutions in different IDEs

Company-wide Monorepo

- E. g. used at Google or Facebook

Moving back and forth



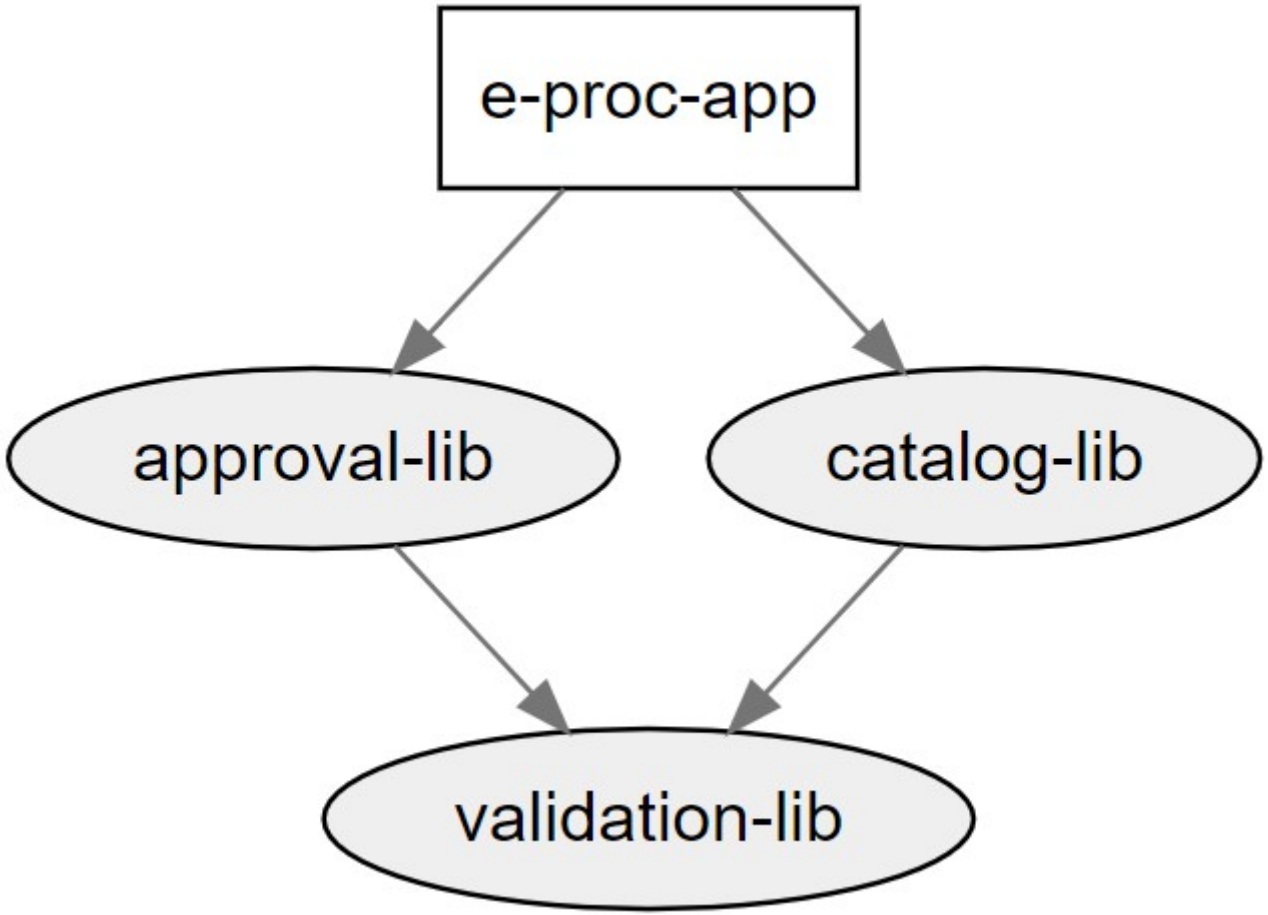
Tooling & Generator

https://nrwl.io/nx

Nrwl Extensions for Angular

An open source toolkit for enterprise Angular applications.

Visualize Module Structure



Creating a Workspace

```
npm install -g @angular/cli

ng new workspace

cd workspace

ng generate app my-app
ng generate lib my-lib

ng serve --project my-app
ng build --project my-app
```

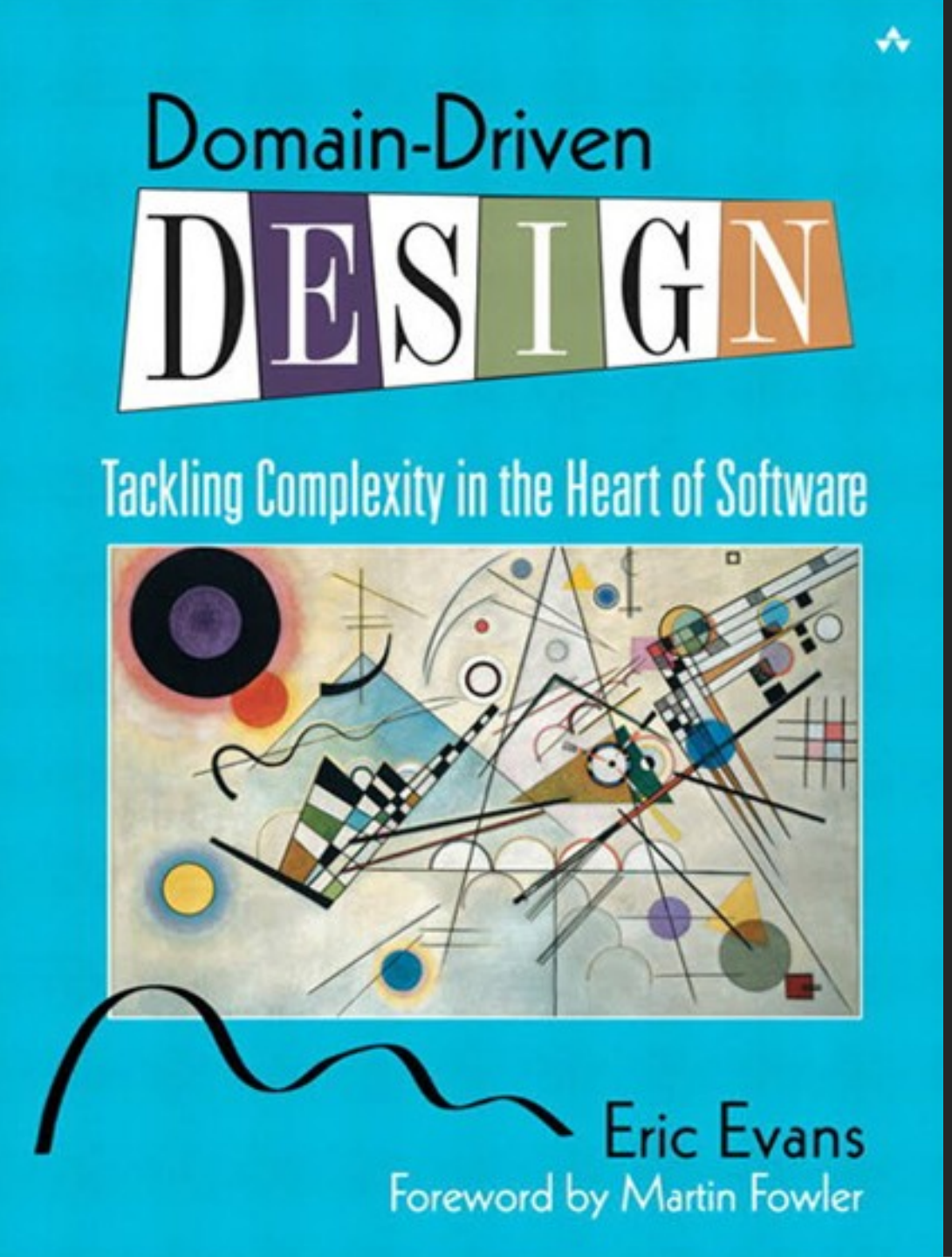
Creating a Workspace

```
npm install -g @angular/cli

npm init nx-workspace workspace

cd workspace

ng generate app my-app
ng generate lib my-lib --buildable

ng serve --project my-app
ng build --project my-app
```

DEMO

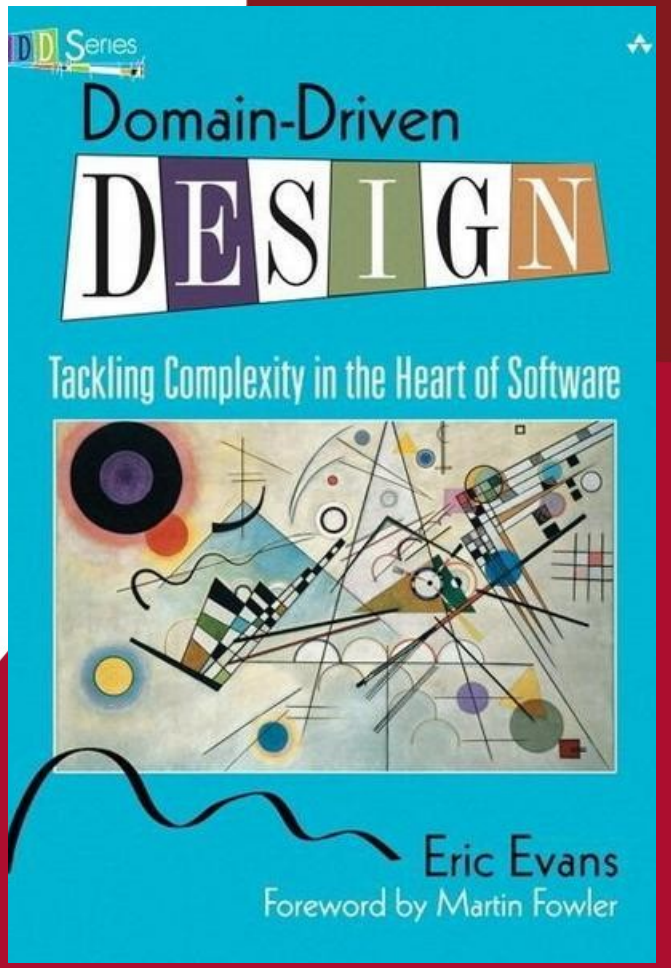
DDD

in a nutshell



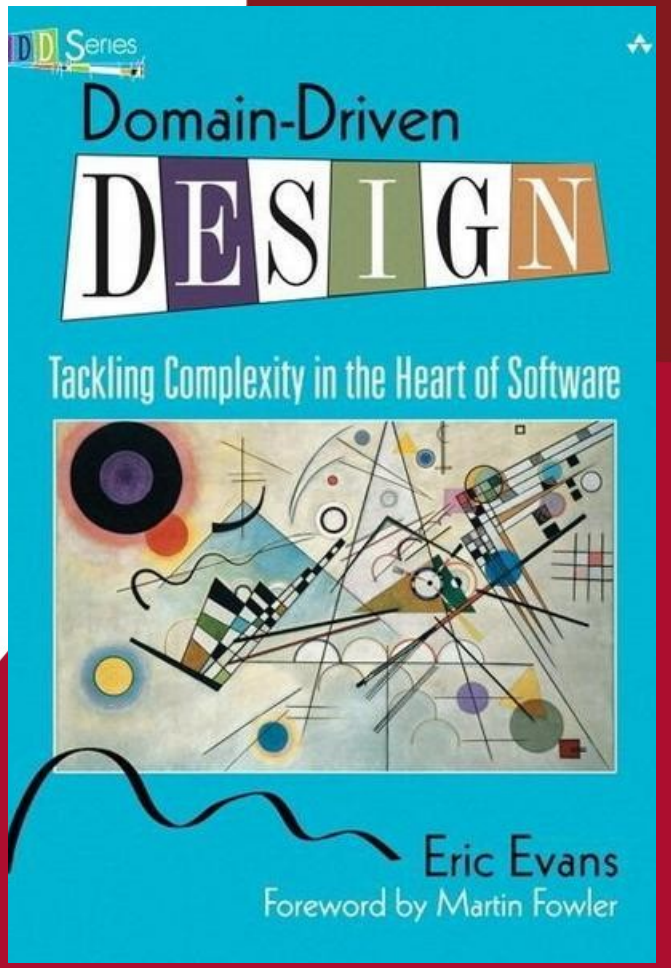
Methodology for bridging the gap b/w requirements and architecture/ design

How to create sustainable frontend architectures with ideas from DDD?



ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE

How to create **sustainable** **frontend** architectures with **ideas from DDD**?

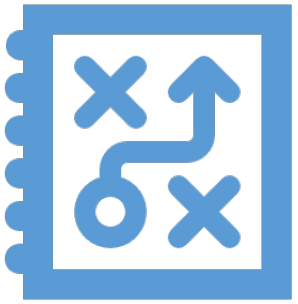


ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE

Domain Driven Design

Decomposing a System

Design Patterns
& Practices



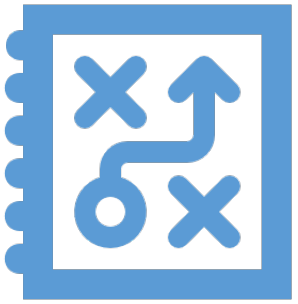
Strategic Design

Tactical Design

Domain Driven Design

Decomposing a System

Design Patterns
& Practices

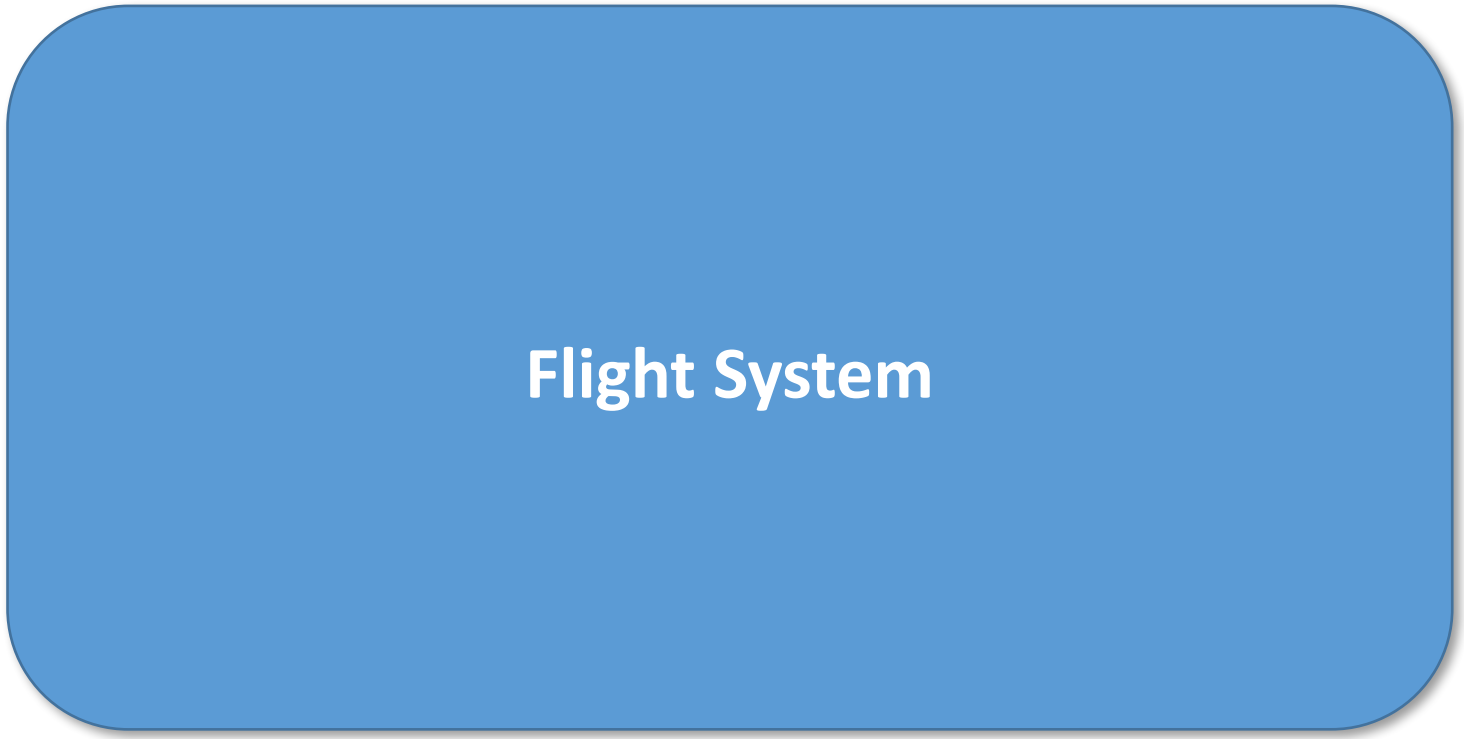


Strategic Design

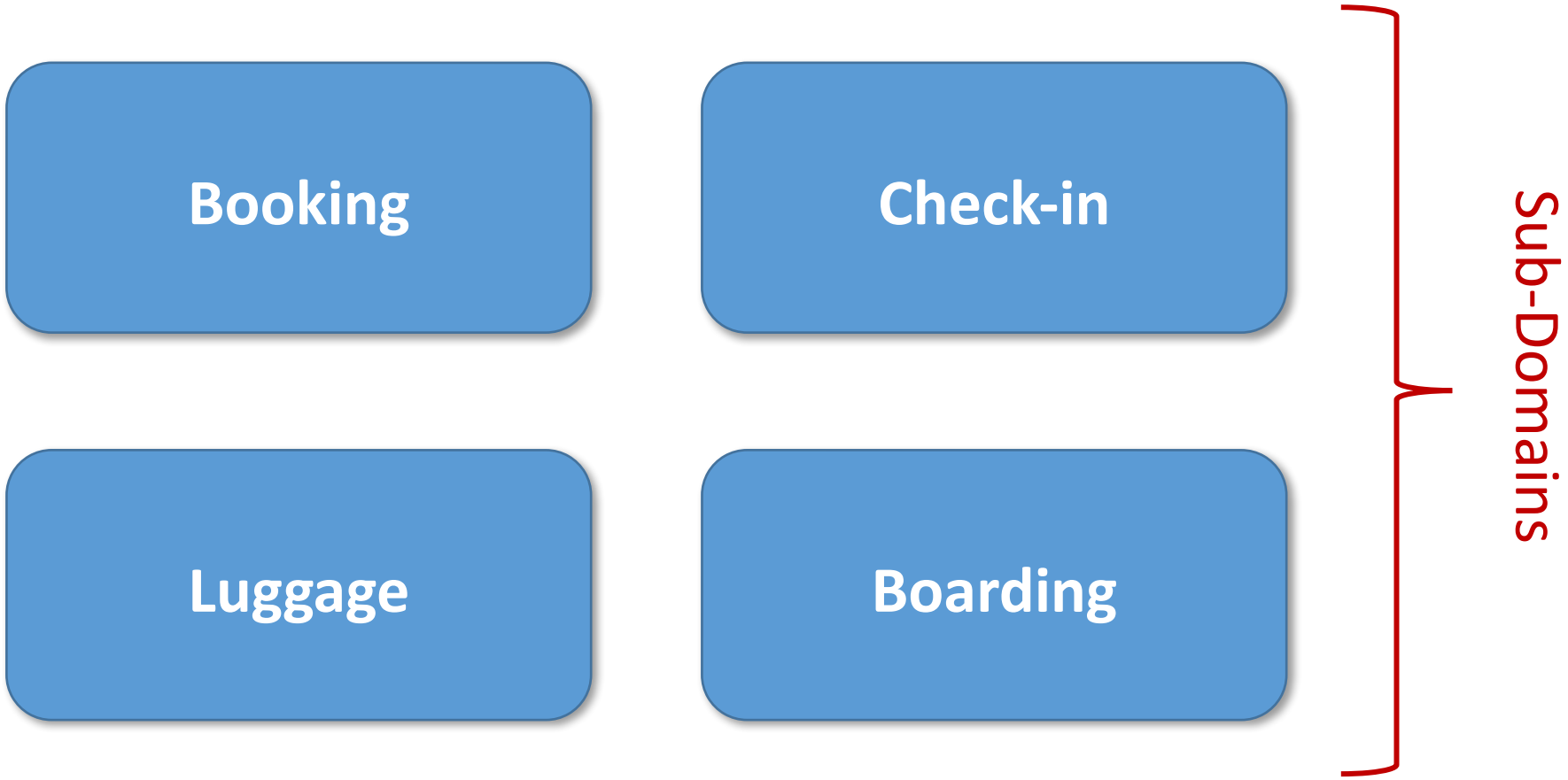
Tactical Design

This is what Strategic DDD prevents

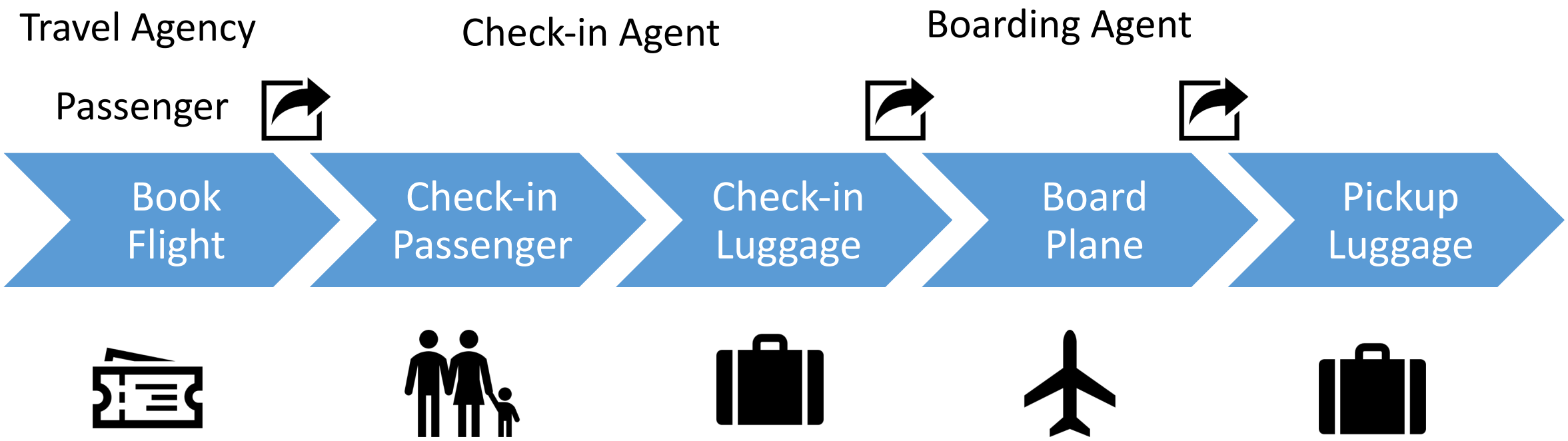
Example

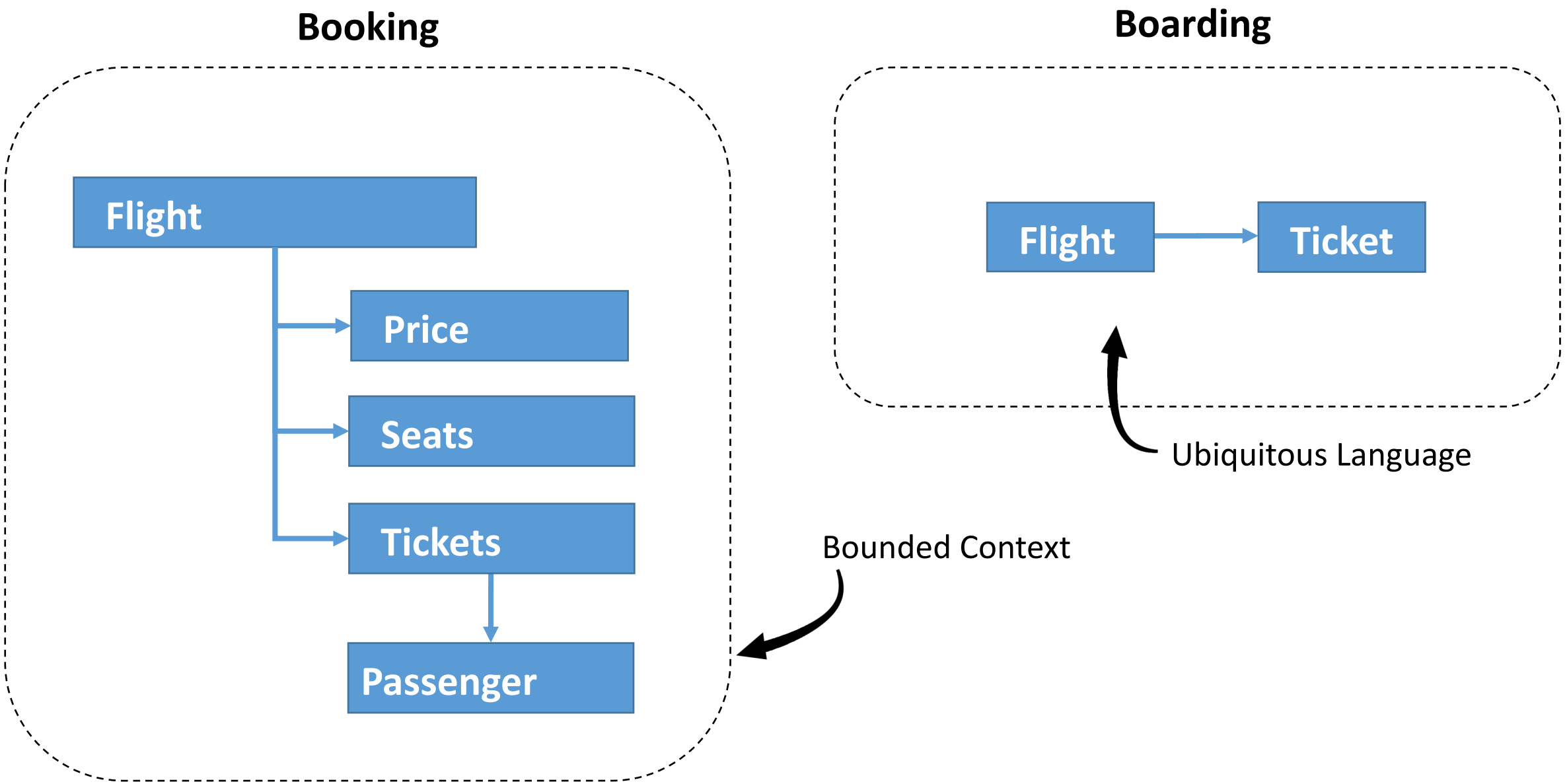


Example



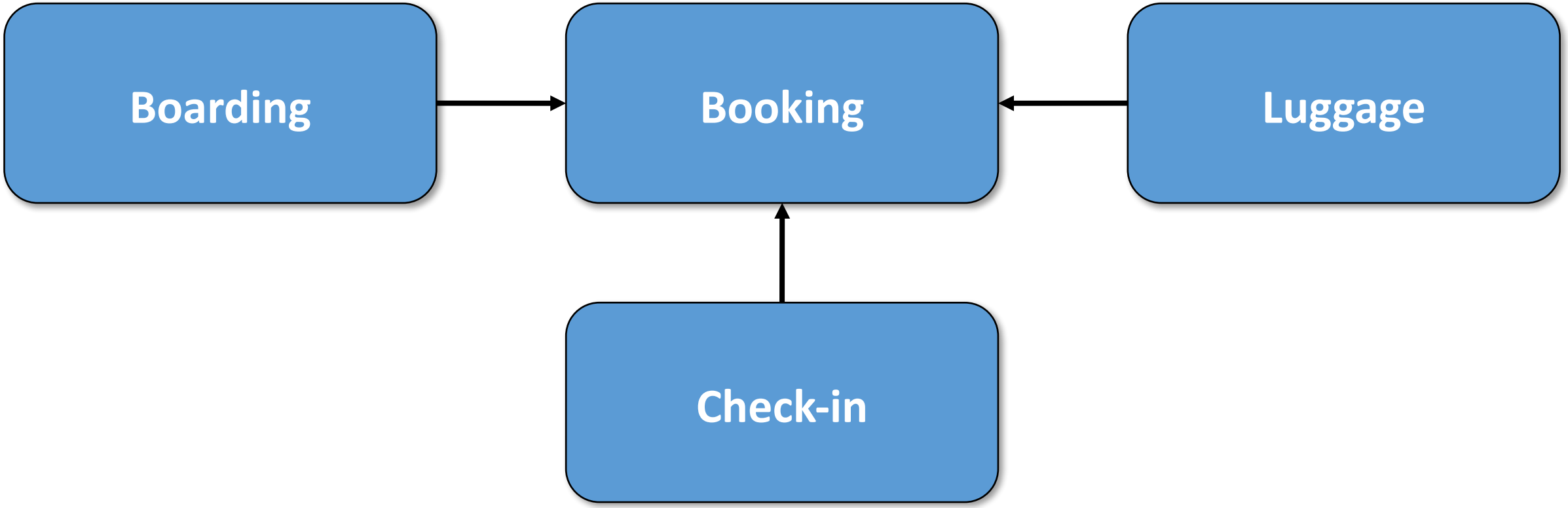
Finding Sub-Domains

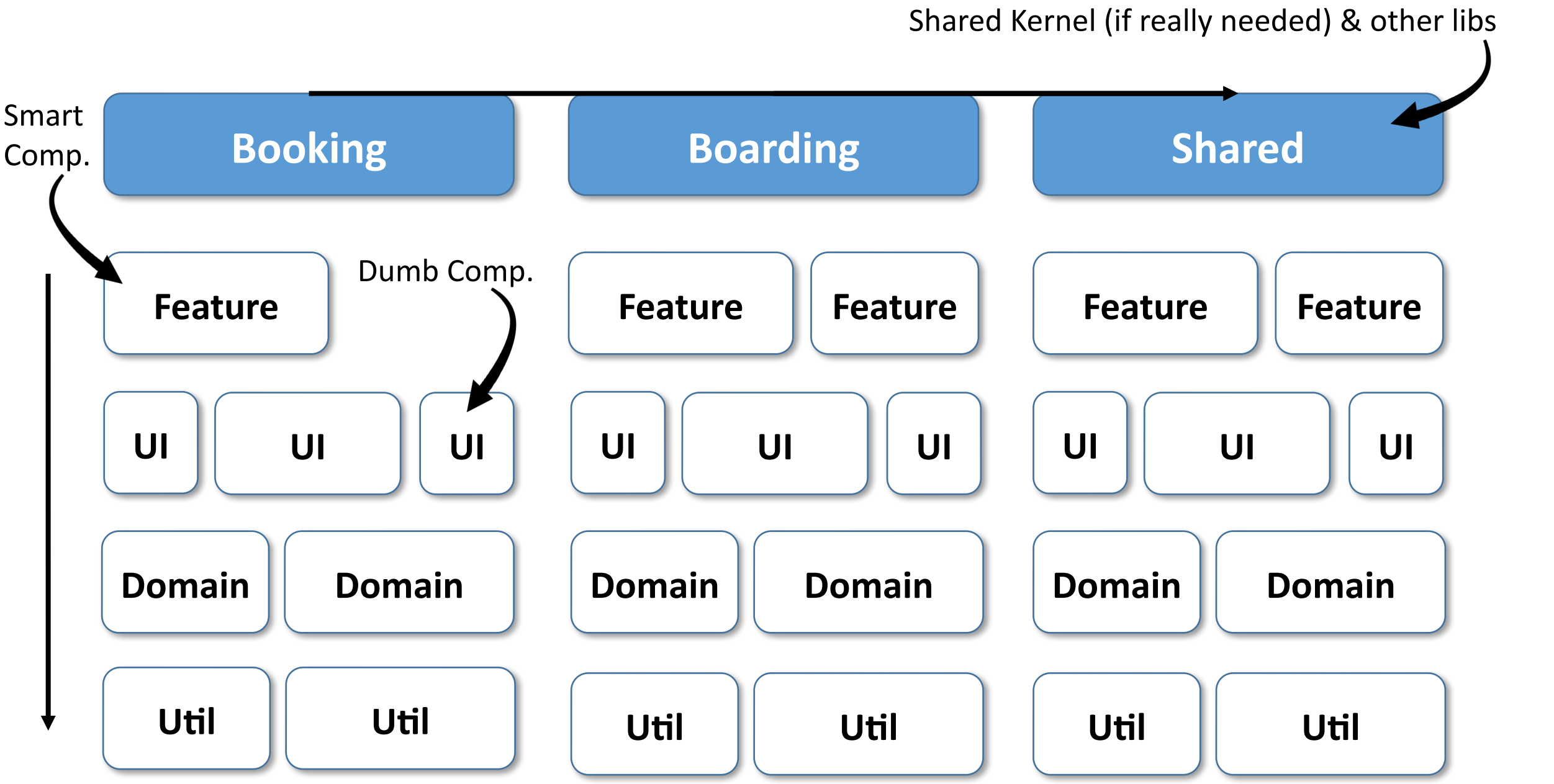




@ManfredSteyer

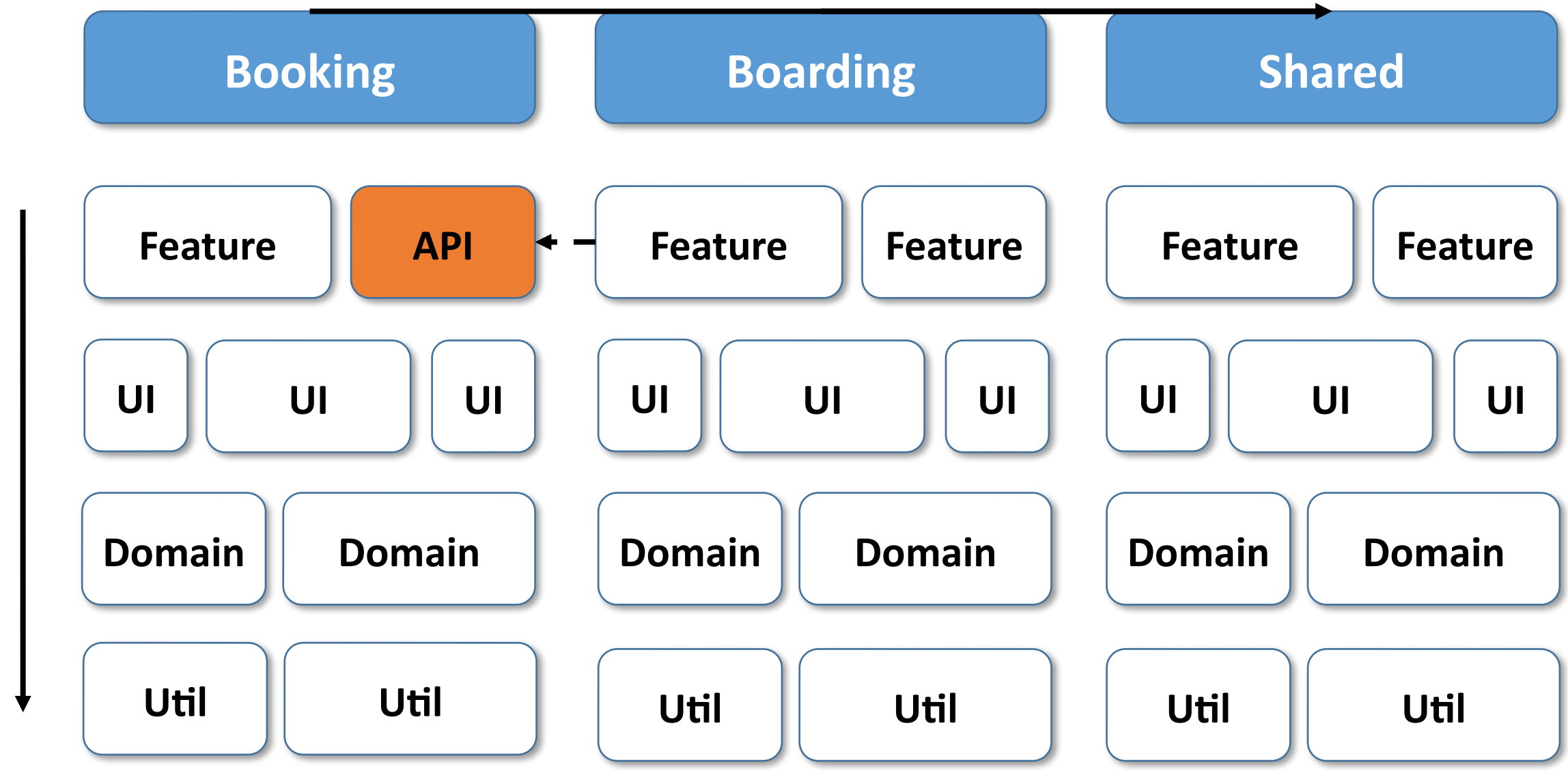
Context Map

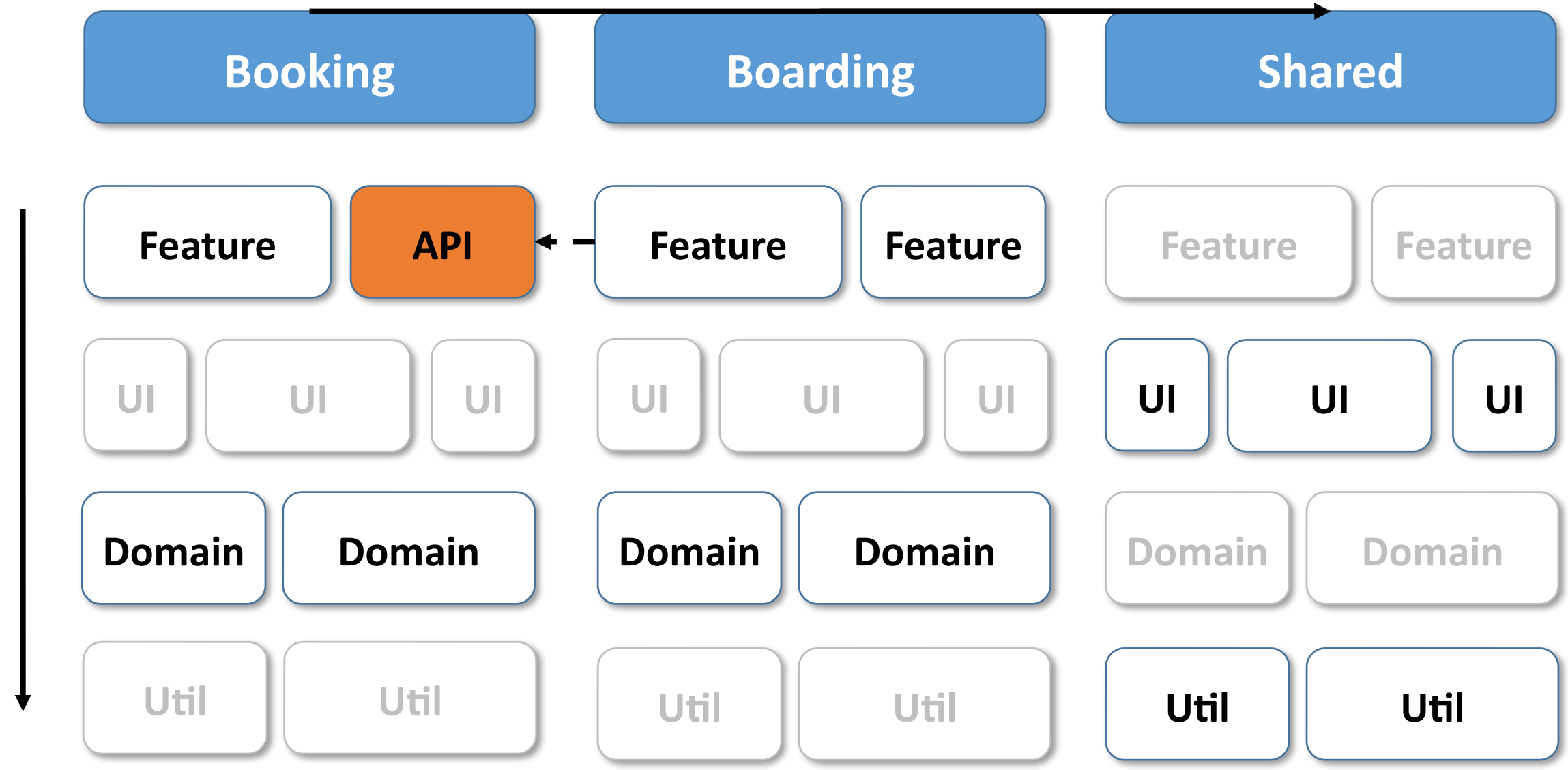




@ManfredSteyer

Enterprise Monorepo Patterns, Nrwl 2018: https://tinyurl.com/y2jjxld7


Booking
Boarding
Shared
Feature
API
Feature
Feature
Feature
Feature
UI
UI
UI
UI
UI
UI
UI
UI
UI
Domain
Domain
Domain
Domain
Domain
Domain
Util
Util
Util
Util
Util
Util


Booking
Boarding
Shared
Feature
API
Feature
Feature
Feature
Feature
UI
UI
UI
UI
UI
UI
UI
UI
UI
Domain
Domain
Domain
Domain
Domain
Domain
Util
Util
Util
Util
Util
Util

DEMO

Build Cache

Best Performance: Several Apps!

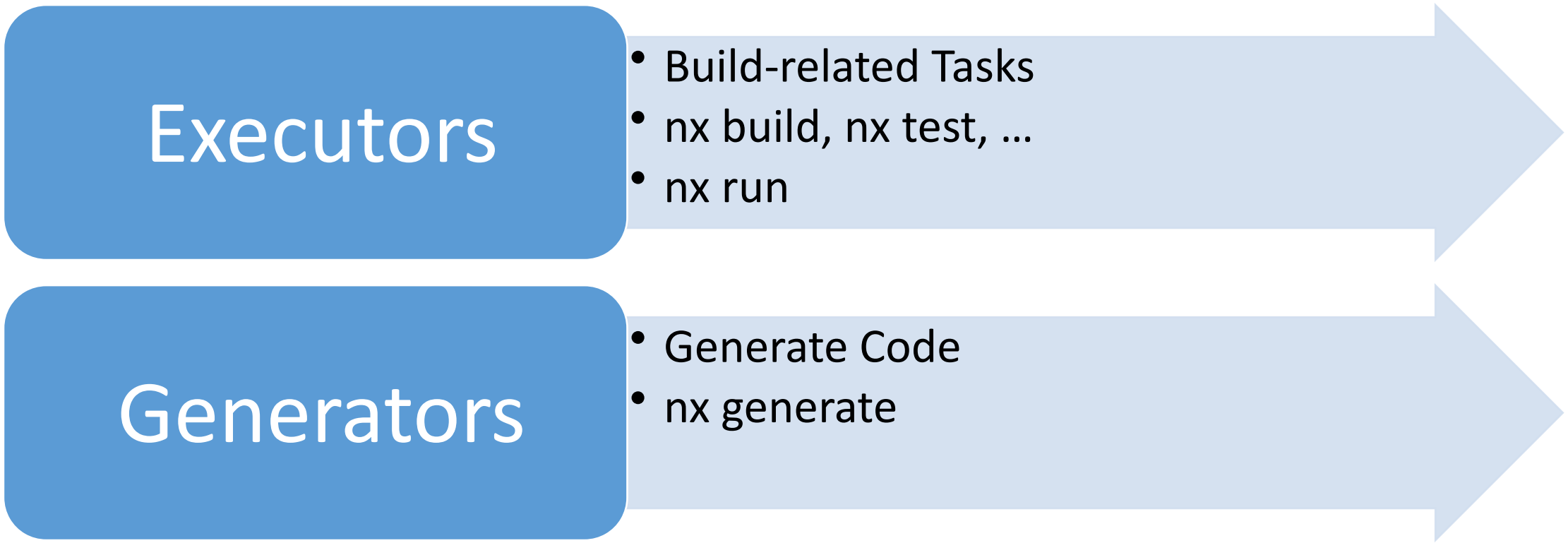
Nx Plugins

ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE

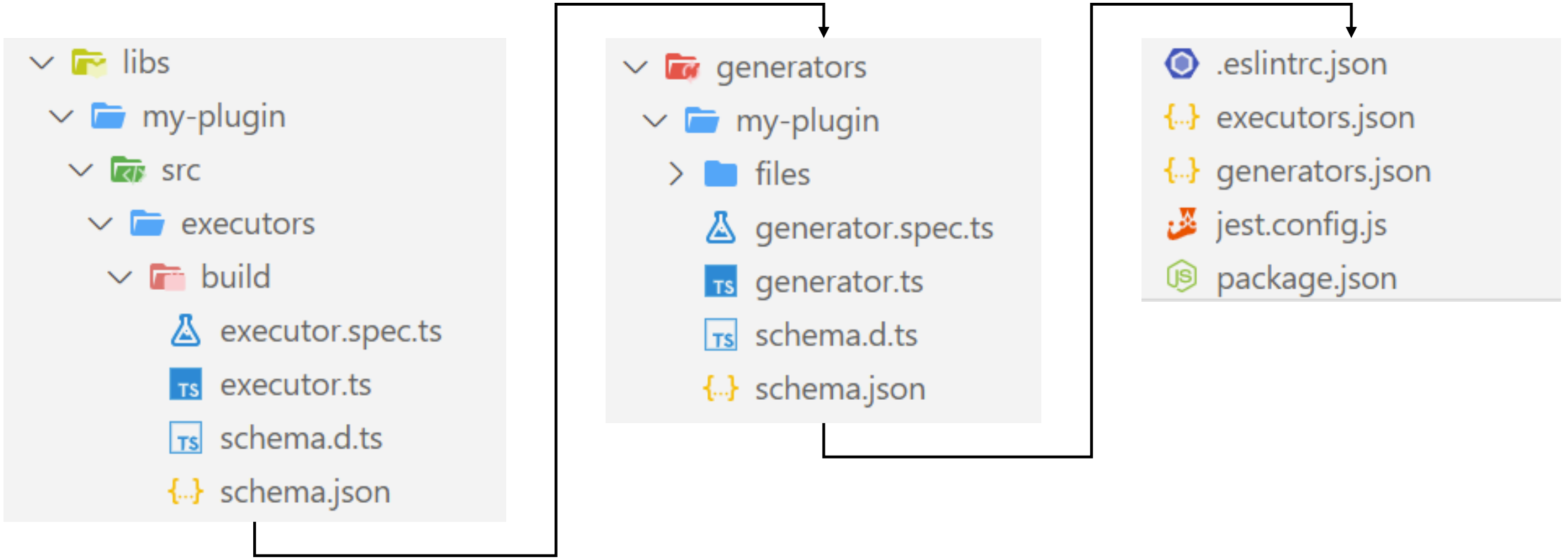
Add Plugin to Workspace

```
npm i @nrwl/nx-plugin

nx generate @nrwl/nx-plugin:plugin my-plugin
```

Building Blocks in Plugins



Folder Structure



Execute Generators

```
nx g @workspace-name/plugin-name:generator-name ...params...

nx g @flight-workspace/my-plugin:data-lib ...params...
```

Execute Generators/ Executors

```
nx run app-or-lib-name:target-name

nx run flight-app:demo-executor
```

target in project.json

DEMO

LAB

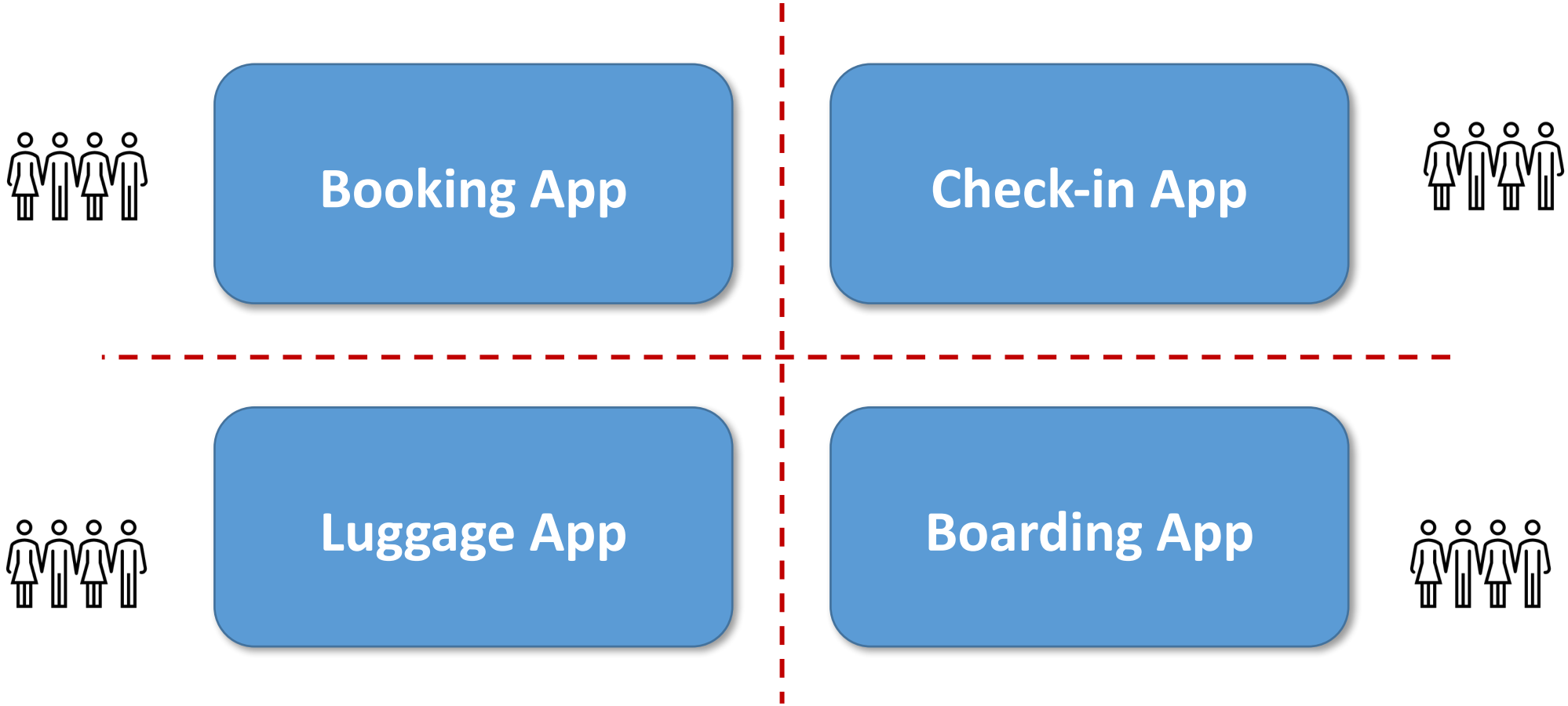
Finegrained Libraries

- Unit of recompilation
- Unit of retesting
- Access restrictions
- Information Hiding
- Easy: Just *ng g lib …*
- Future replacement for NgModules?

Micro Frontends?

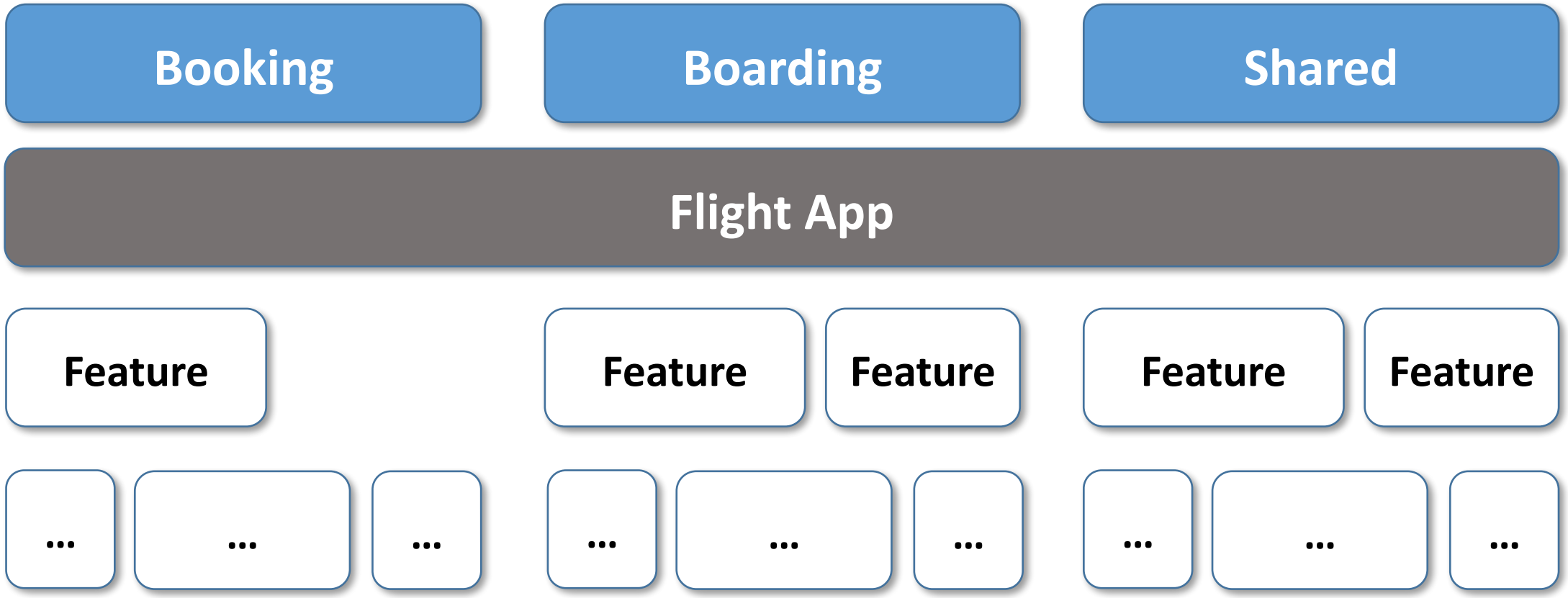
Short outlook

Microfrontends

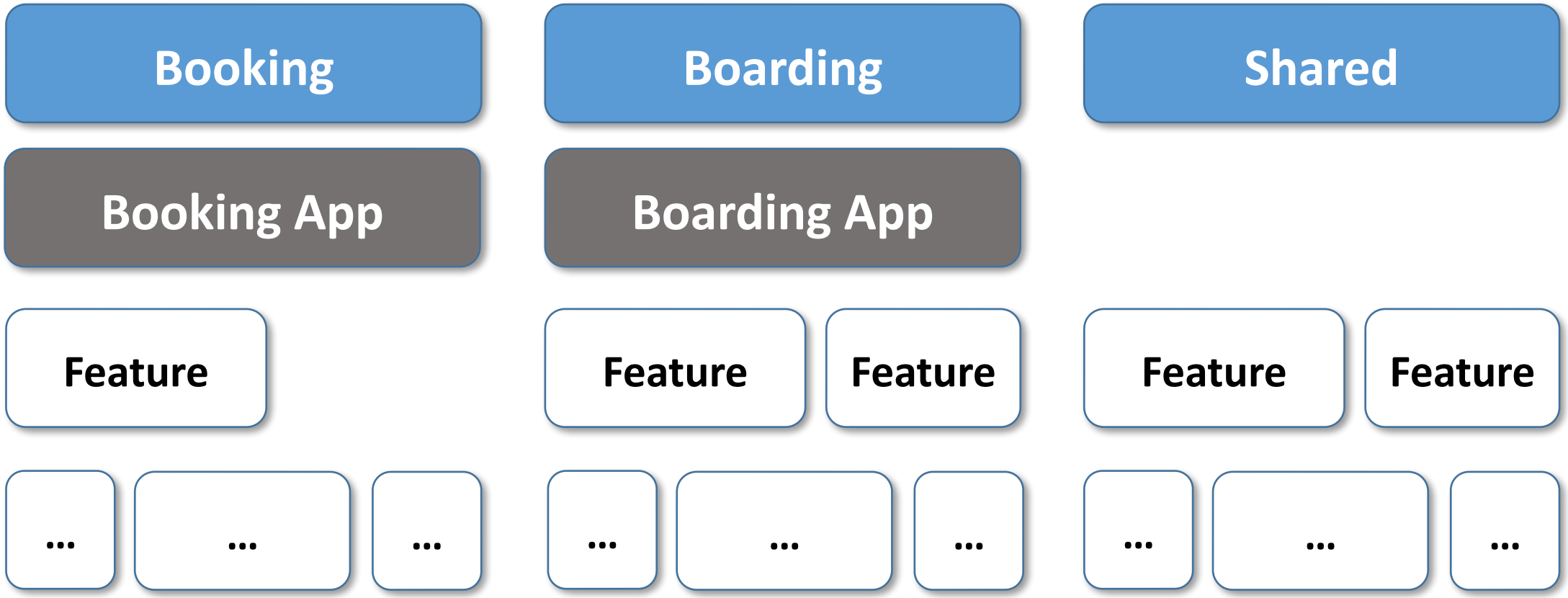


Microfrontends
are first and foremost
about **scaling teams!**

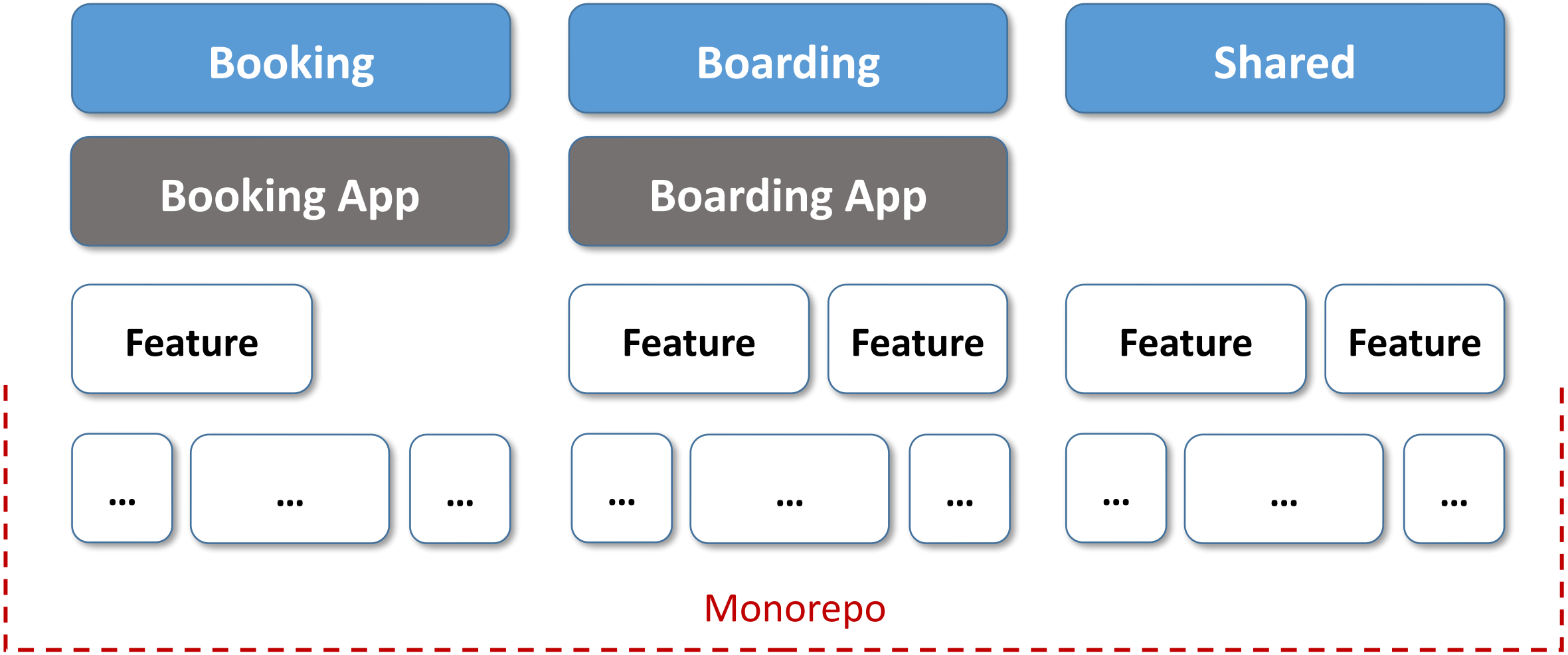
Deployment Monolith



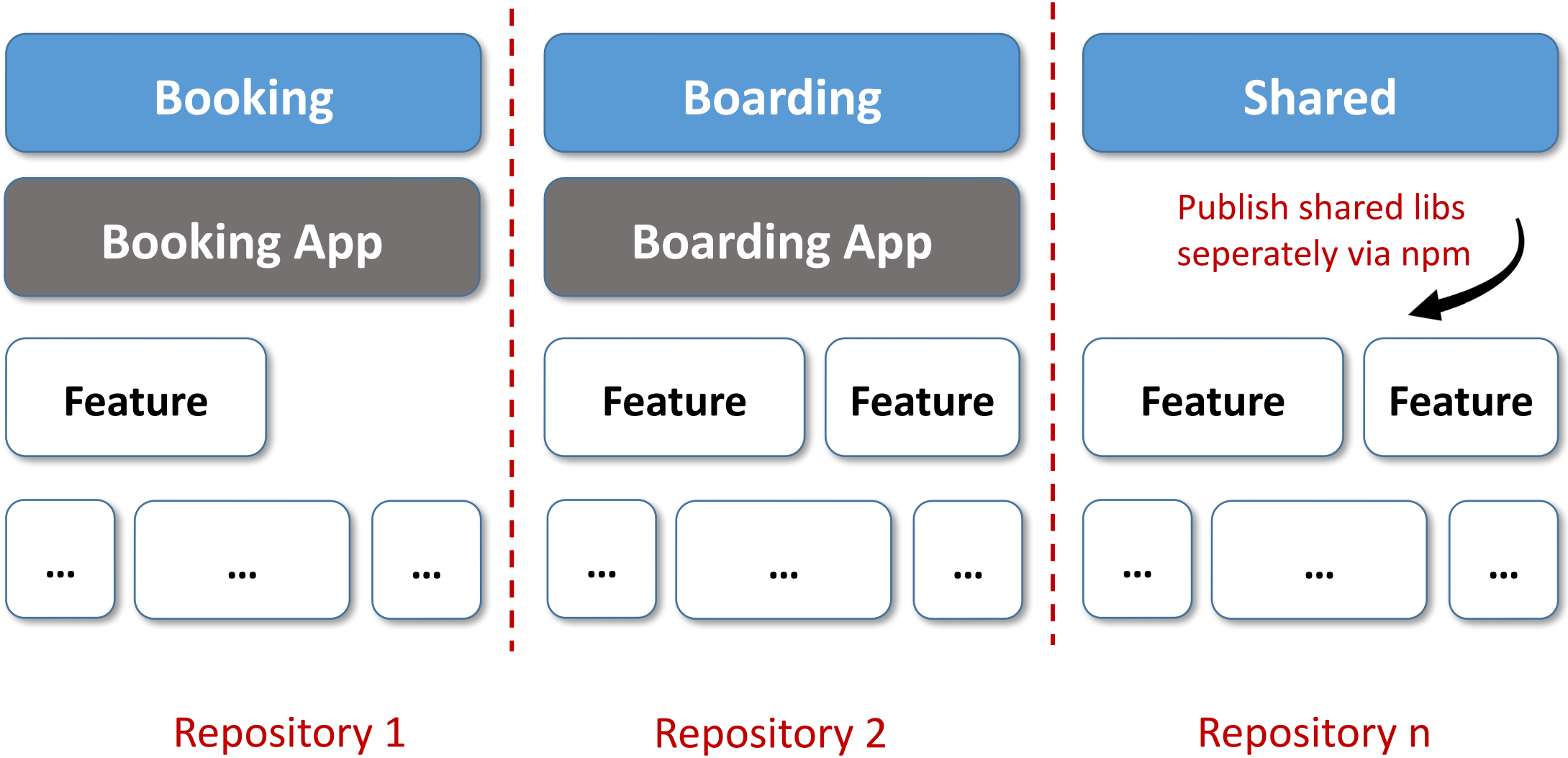
Microfrontends



Option 1: One App per Domain



Option 2: One Monorepo per Domain

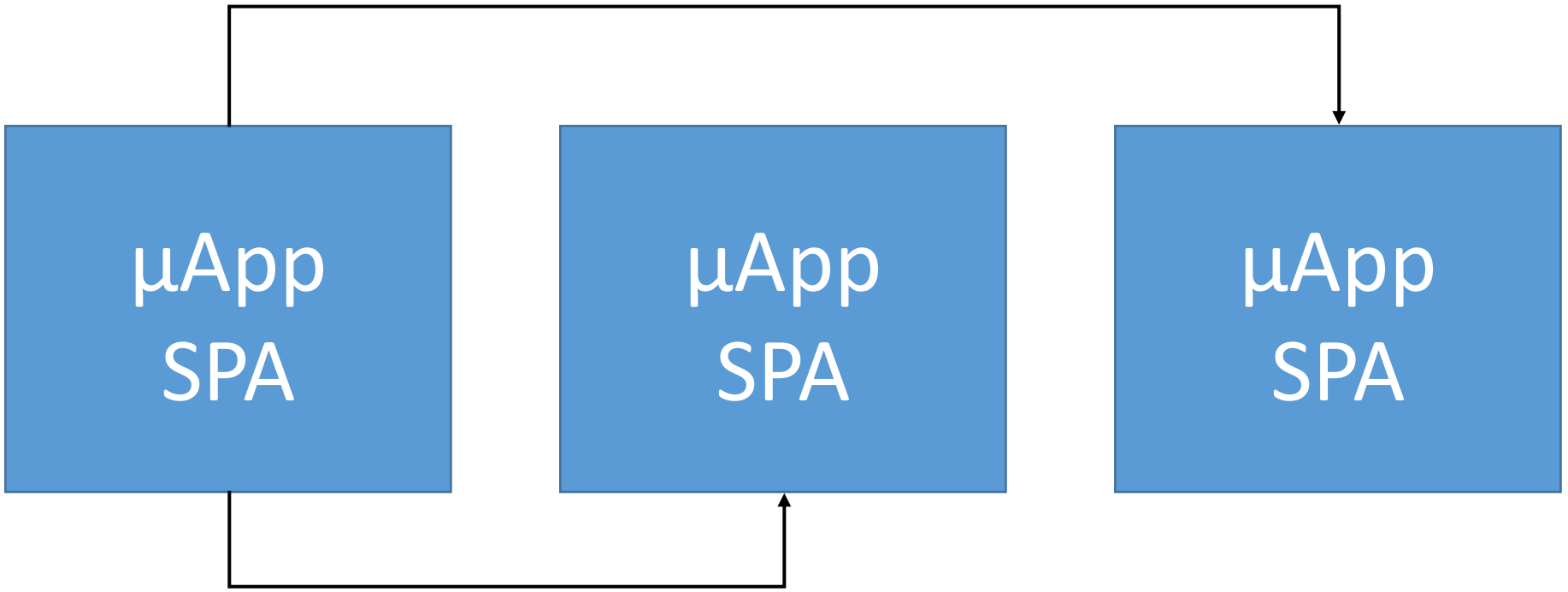


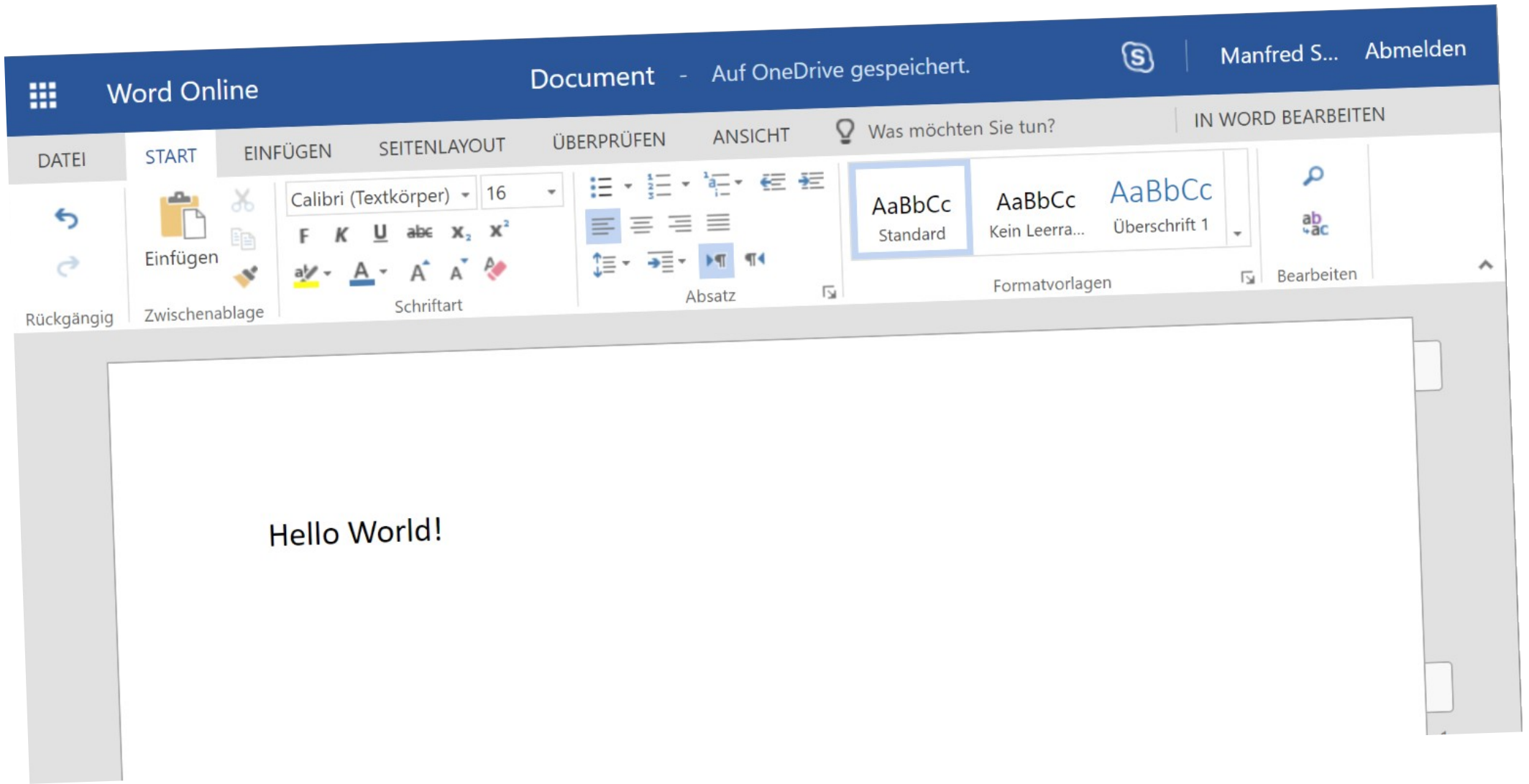
Benefits

- **Autonomous Teams**
- Separate Development
- Separate Deployment
- Own architecture decisions
- Own technology descisions

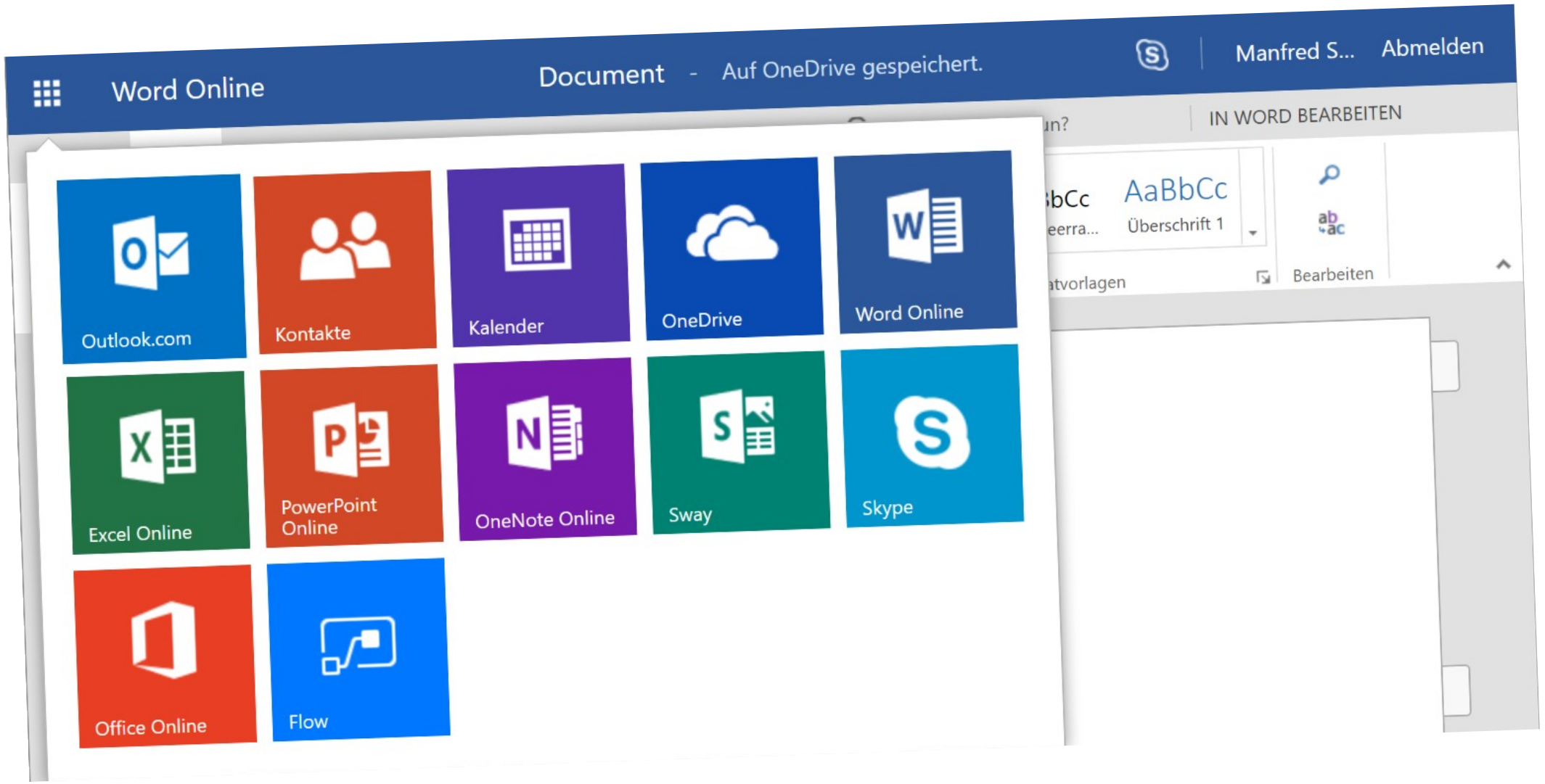
Integration via Hyperlinks

UI Composition w/ Hyperlinks





@ManfredSteyer

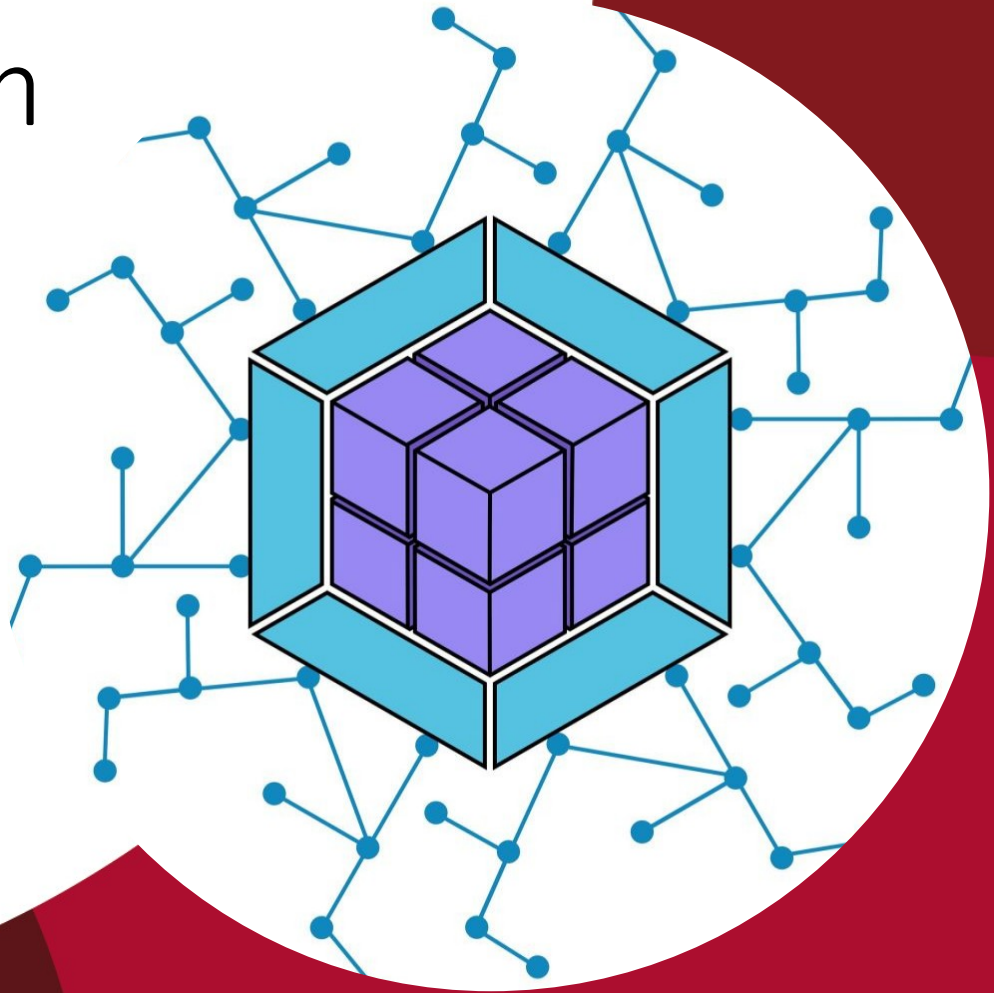


@ManfredSteyer


Integration via
Shell

Providing a (SPA based) Shell

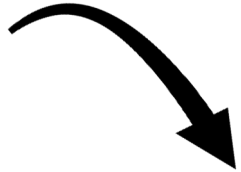
Webpack 5
Module Federation



ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE

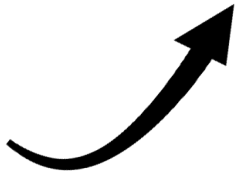
Idea

Does not work with
webpack/ Angular CLI

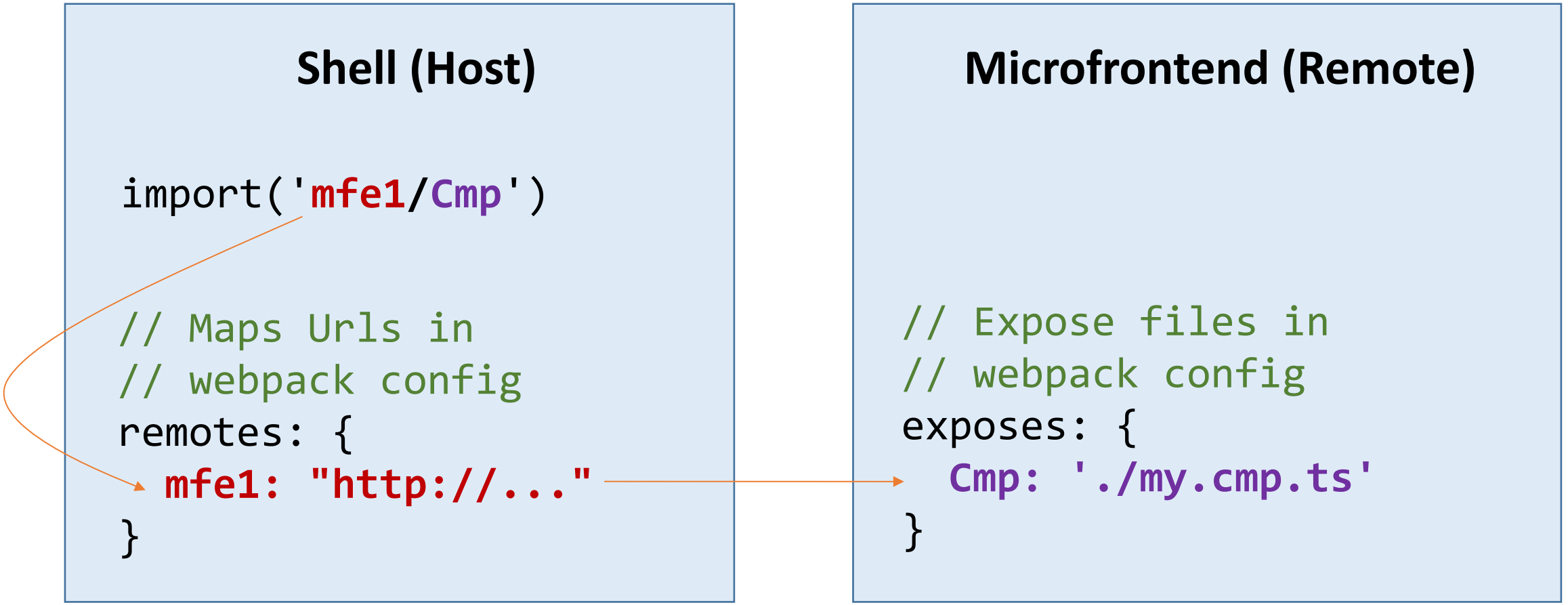


```
const Component = import('http://other-app/xyz')
```

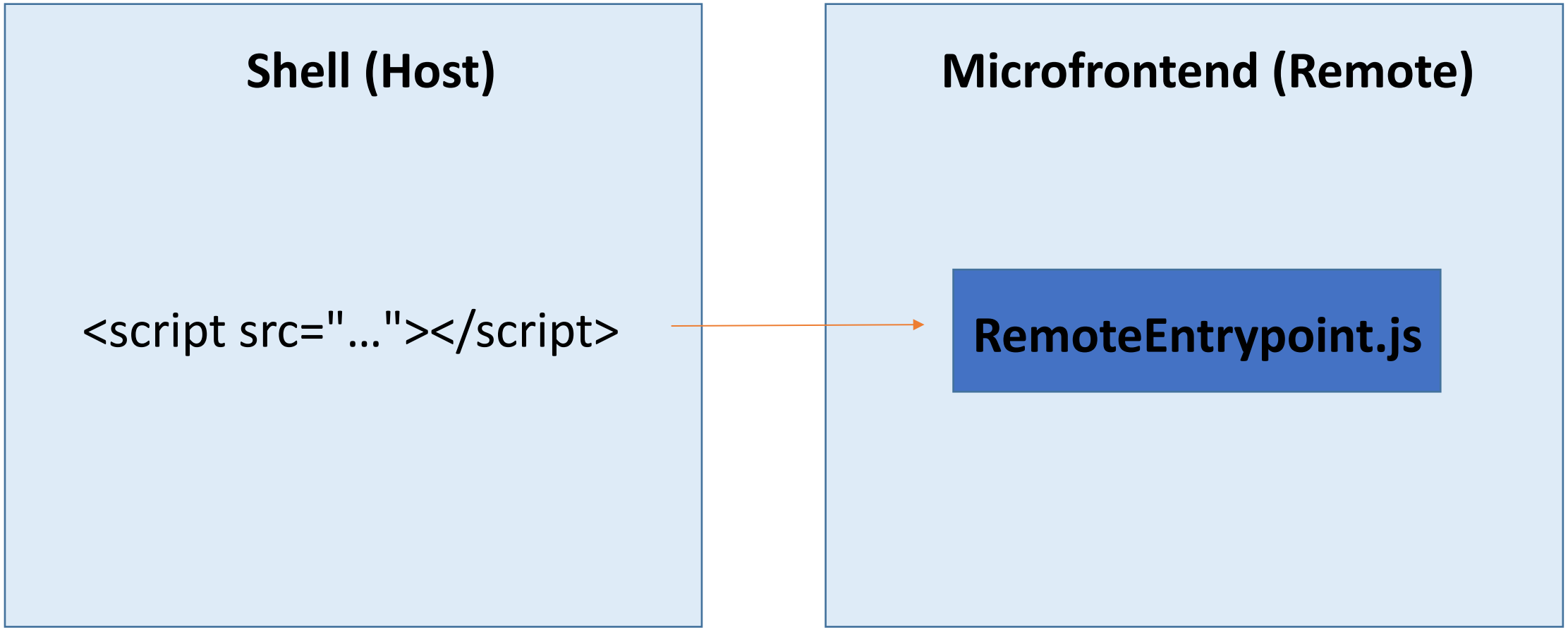
Even lazy parts must be
known at compile time!



Webpack 5 Module Federation



How to Get the Microfrontend's URL?



How to Share Libs?

Shell (Host)

```
shared: [
  "@angular/core", "..."
]
```

Microfrontend (Remote)

```
shared: [
  "@angular/core", "..."
]
```

Dealing with Version Mismatches

ANGULAR ARCHITECTS
INSIDE KNOWLEDGE

Default Behavior

Selecting the highest compatible version

Default Behavior

Conflict: No highest compatible version

Example

- Shell: my-lib: ^10.0
- MFE1: my-lib: ^10.1
- MFE2: my-lib: ^9.0
- MFE3: my-lib: ^9.1

Result:

- Shell and MFE1 share ^10.1
- MFE2 and MFE3 share ^9.1

Configuring Singletons

```
shared: {
  "my-lib": {
    singleton: true
  }
}
```

11.0 ✓ 10.1 ✗

Configuring Singletons

```
shared: {
  "my-lib": {
    singleton: true,
    strictVersion: true // Error instead of warning!
  }
}
```

11.0  10.1 

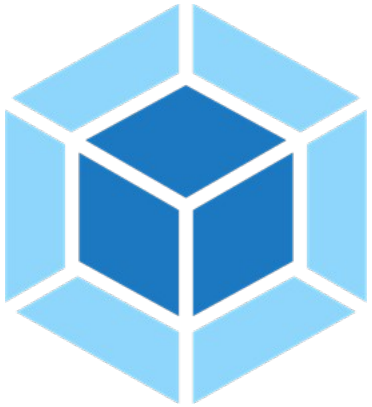
Relaxing Version Requirements

```
shared: {
  "my-lib": {
     requiredVersion: ">=1.0.1 <11.1.1"
  }
}
```

Federated Angular: Angular, CLI, & Module Federation

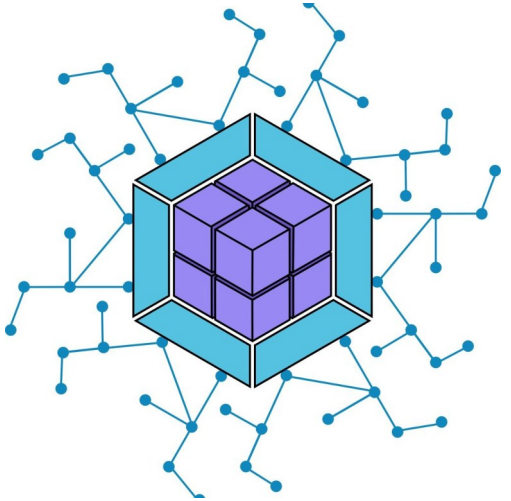
ANGULAR ARCHITECTS
INSIDE KNOWLEDGE

?



webpack

Angular CLI

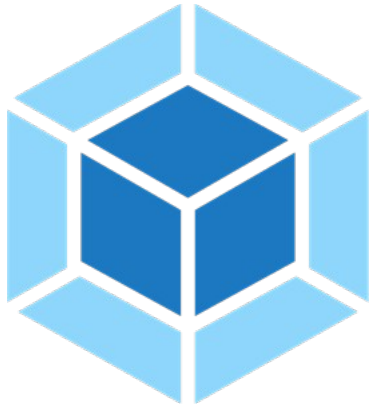


Module Federation
Configuration

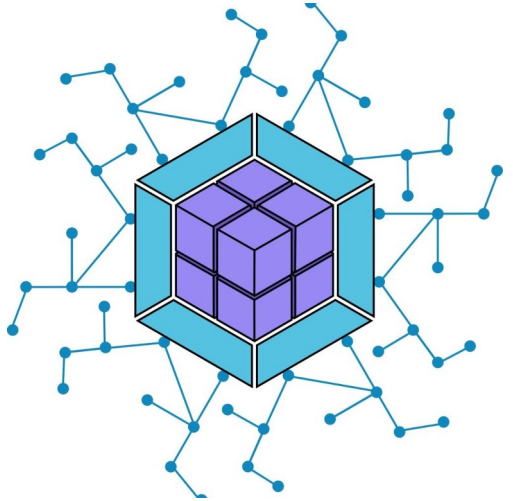
@ManfredSteyer

Custom Builder

Angular CLI



webpack



Module Federation
Configuration

@angular-architects/module-federation

1.0.2 • Public • Published 18 hours ago

Readme | Explore BETA | 3 Dependencies

Features

- ✅ Generates the skeleton for a Module Federation config.
- ✅ Installs a custom builder to enable Module Federation.
- ✅ Assigning a new port to serve (`ng serve`) several projects at once.

@ManfredSteyer

Usage

1) ng add @angular-architects/module-federation
2) Adjust generated configuration
3) ng serve

Usage

1) npm i @angular-architects/module-federation -D
2) ng g @angular-architects/module-federation:init
3) Adjust generated configuration
4) ng serve

DEMO

ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE

Multi Framework/ Version Solutions

ANGULAR ARCHITECTS
INSIDE KNOWLEDGE

Abstracting Differences b/w SPA Frameworks

Wrap them into Web Components

Loading Web Components via Module Federation

ANGULAR ARCHITECTS
INSIDE KNOWLEDGE

Module Federation

```
await import('other-app/web-cmp');
```

Module Federation

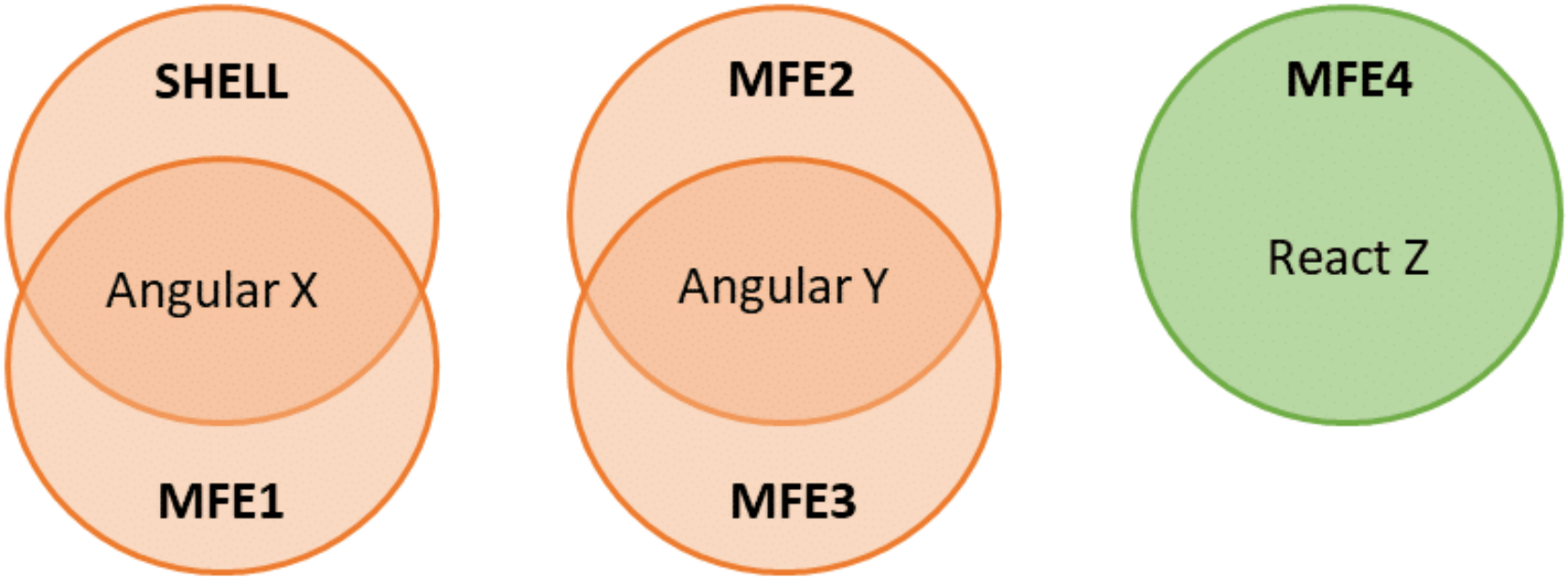
```
const rootElm = document.createElement('web-cmp')
document.body.appendChild(rootElm);

await import('other-app/web-cmp');
```

Routing to Another SPA?

WrapperComponent

```
const rootElm = document.createElement('web-cmp')
document.body.appendChild(rootElm);

await import('other-app/web-cmp');
```

Result



DEMO

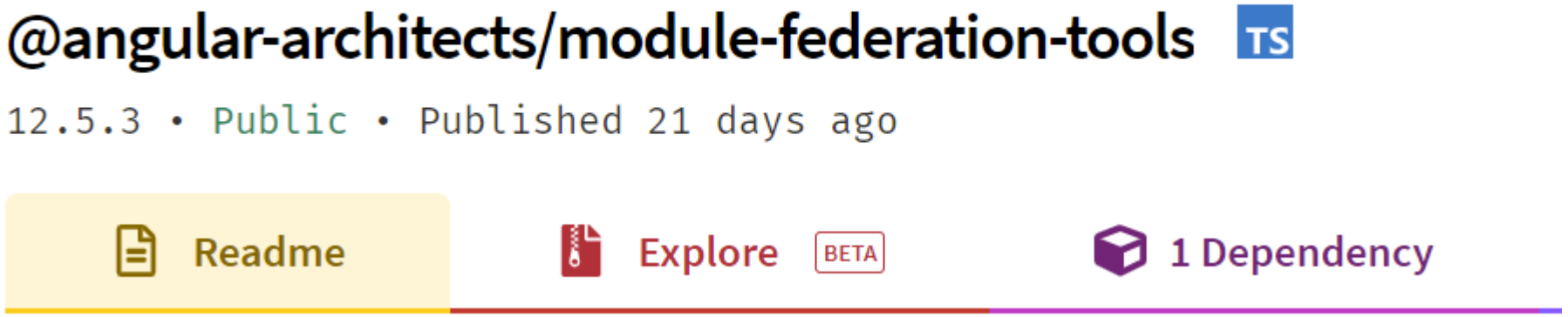
https://red-ocean-0fe4c4610.azurestaticapps.net

ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE

Challanges

- Bundle Size
- Multiple Routers
- Bootstrapping Several Angular Instances
  - Share Platform-Object when same version is reused
  - Share ngZone

Challanges & Solutions




Choosing a Solution

Some General Advice