IPB Leibniz-Institut für Pflanzenbiochemie

How software architecture defines boundaries and APIs bridge them

Fabian Mauz

WissenschaftsCampus Pflanzenbasierte Bioökonomie Halle

Roadmap

Architecture

- Definition and Motivation
- Example : Microservices
- Example : Ports and Adapters

Roadmap

Architecture

- Definition and Motivation
- Example : Microservices
- Example : Ports and Adapters

APIs

- A definition from reality
- Examples : Java Interface & REST
- Open API

Roadmap

Architecture

- Definition and Motivation
- Example : Microservices
- Example : Ports and Adapters

APIs

- A definition from reality
- Examples : Java Interface & REST
- Open API

Tools & Tipps

- Security
- Testing APIs with good architecture
- Github Actions

What is software architecture?

The software architecture of a program or computing system is the **structure or structures** of the system, which comprise software **components**, the externally visible **properties** of those components, and the **relationships** among them [1]

[1] L. Bass, P. Clements and R. Kazman. *Software Architecture in Practice.* Addison Wesley, **1999**, ISBN 0-201-19930-0.

What is software architecture?

The software architecture of a program or computing system is the **structure or structures** of the system, which comprise software **components**, the externally visible **properties** of those components, and the **relationships** among them [1]

[1] L. Bass, P. Clements and R. Kazman. *Software Architecture in Practice.* Addison Wesley, **1999**, ISBN 0-201-19930-0.

What is software architecture?

Martin Fowler

*„The goal of software architecture is to minimize the human resources required to **build** and **maintain** the required system“*

Robert C. Martin, Clean Architecture, 2017

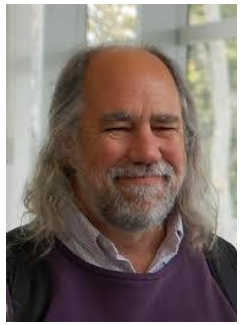
Robert C. Martin

„Software architecture is those decisions which are both **important** and **hard to change** “

Martin Fowler

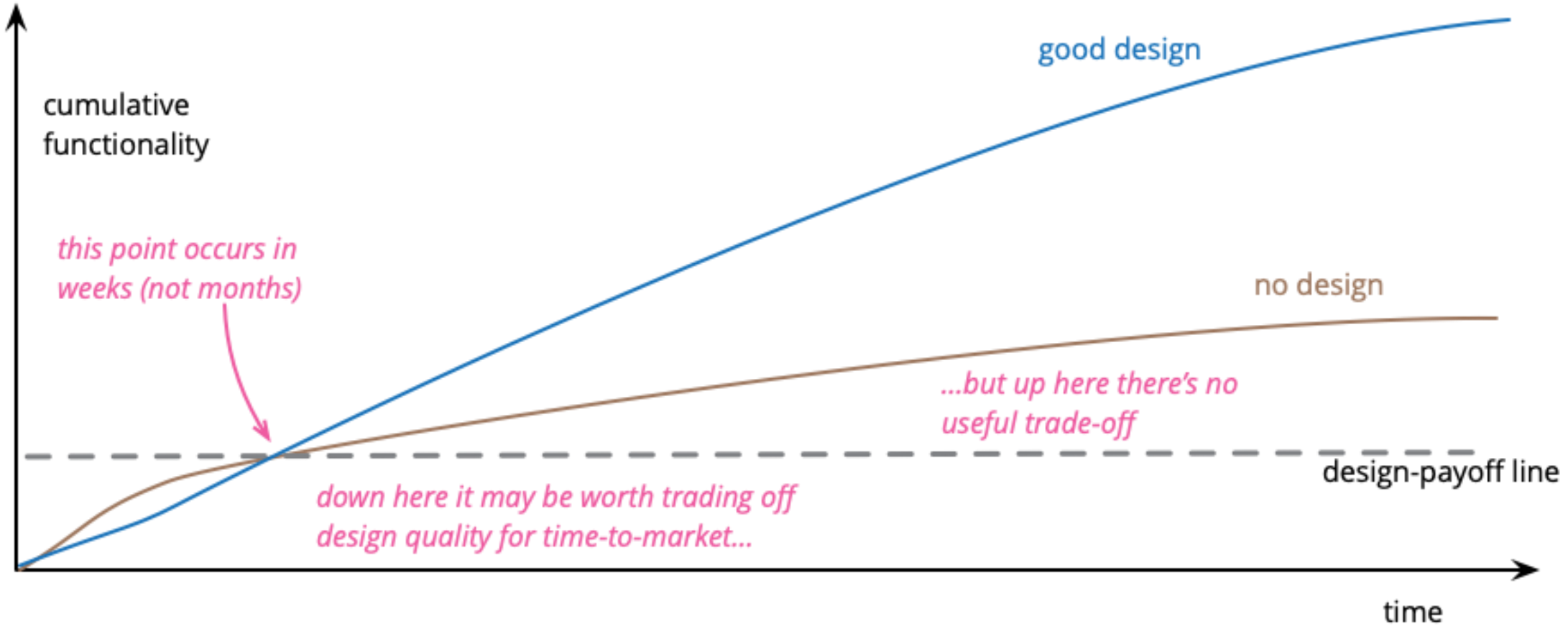
„Architecture represents the significant design decisions that **shape a system**, where **significant** is measured by **cost of change** “

Grady Booch



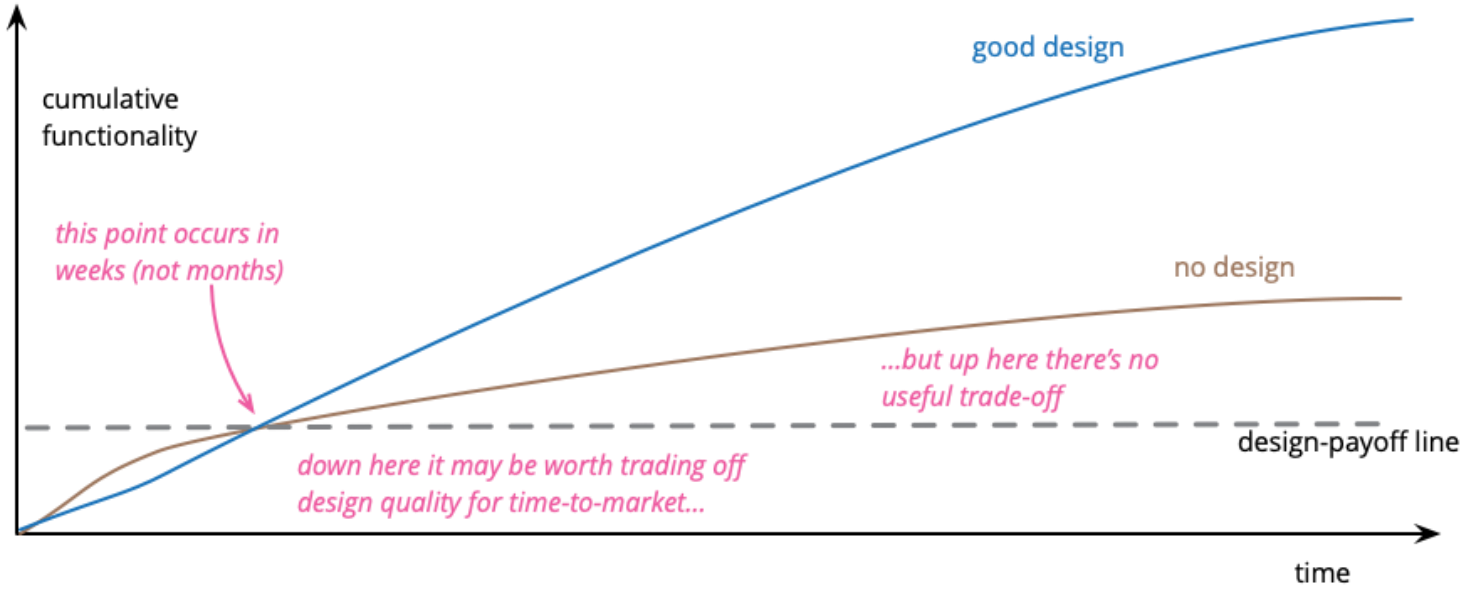
Grady Booch

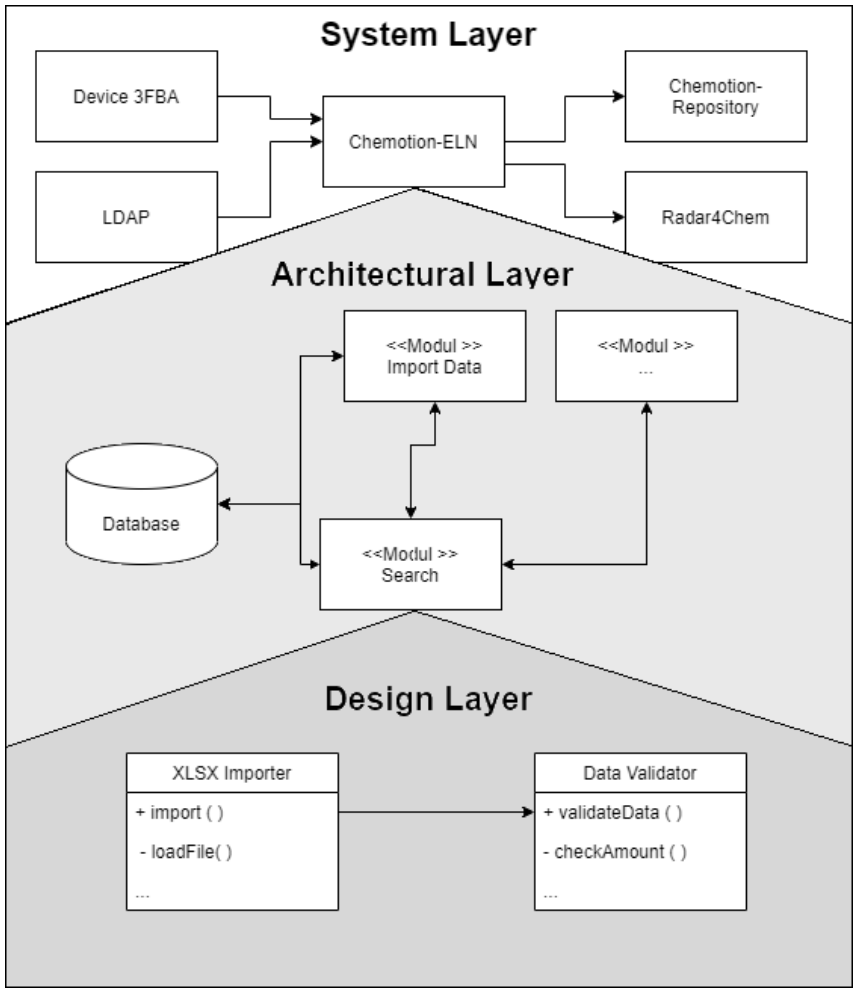
Why thinking about architecture?



M. Fowler, *Design Stamina Hypothesis*, **2007**

Why thinking about architecture?



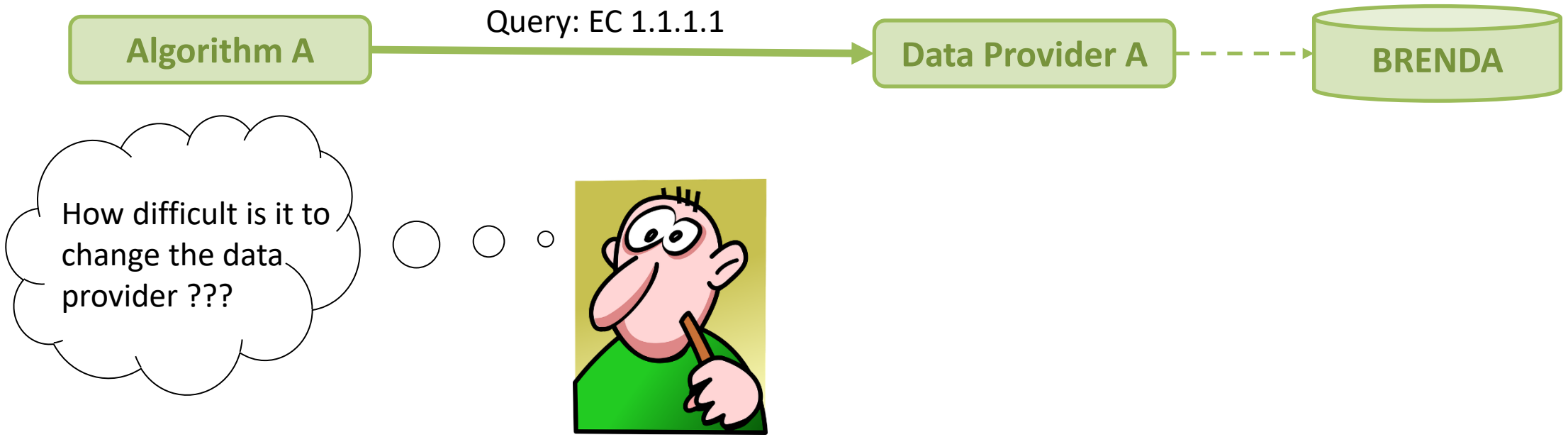


System ≠ *Architecture* ≠ Design ???

M. Fowler, *Design Stamina Hypothesis*, **2007**

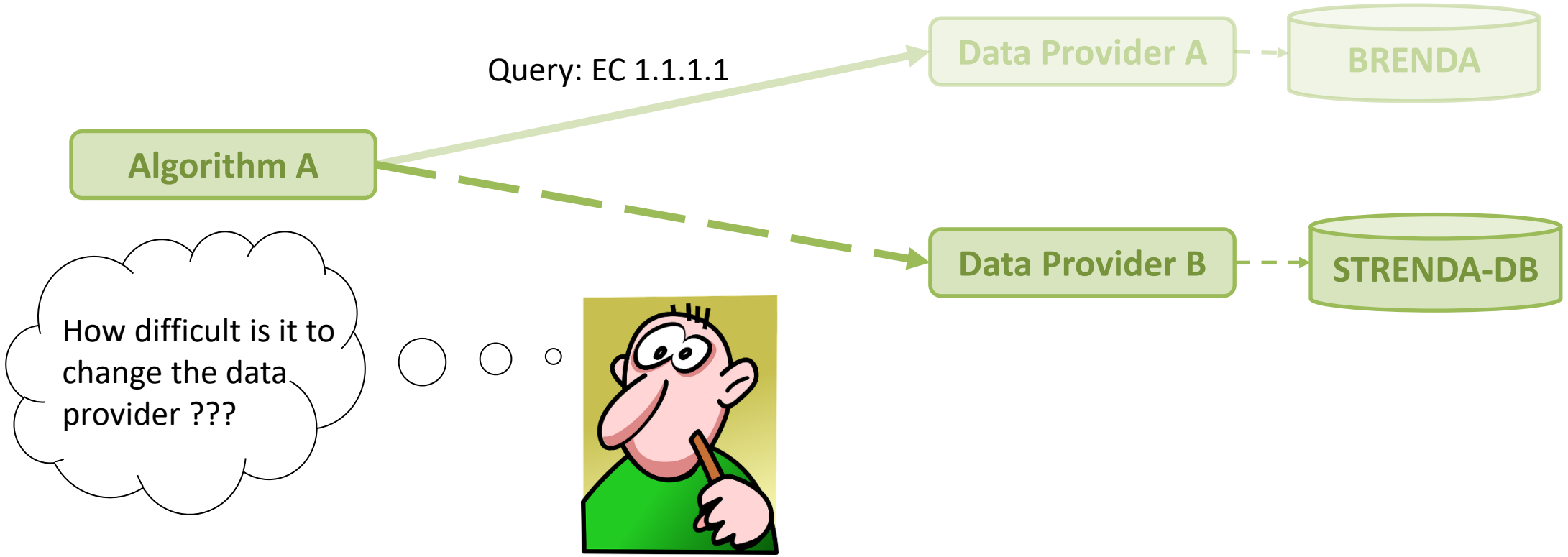
Dependencies

One main problem to change software are **dependencies**



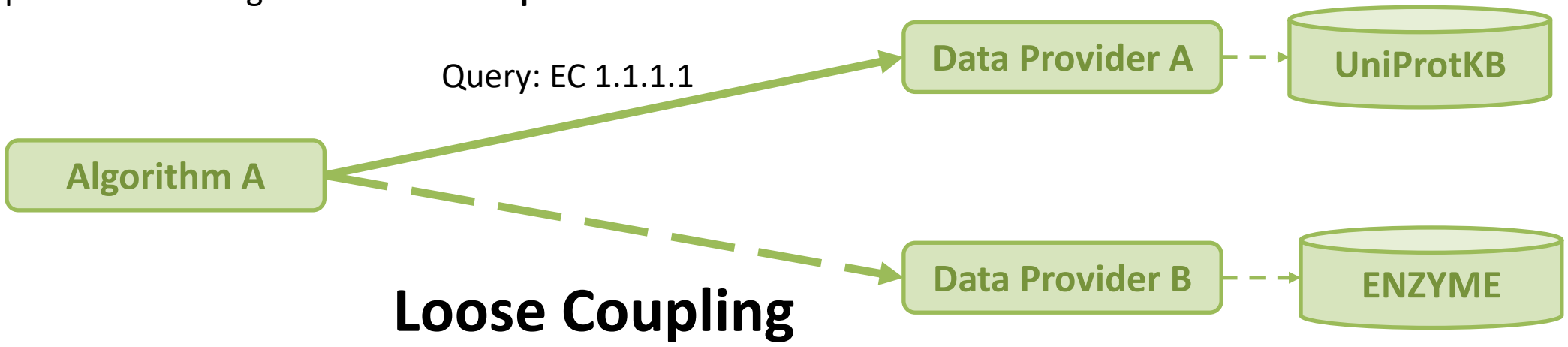
Dependencies

One main problem to change software are **dependencies**

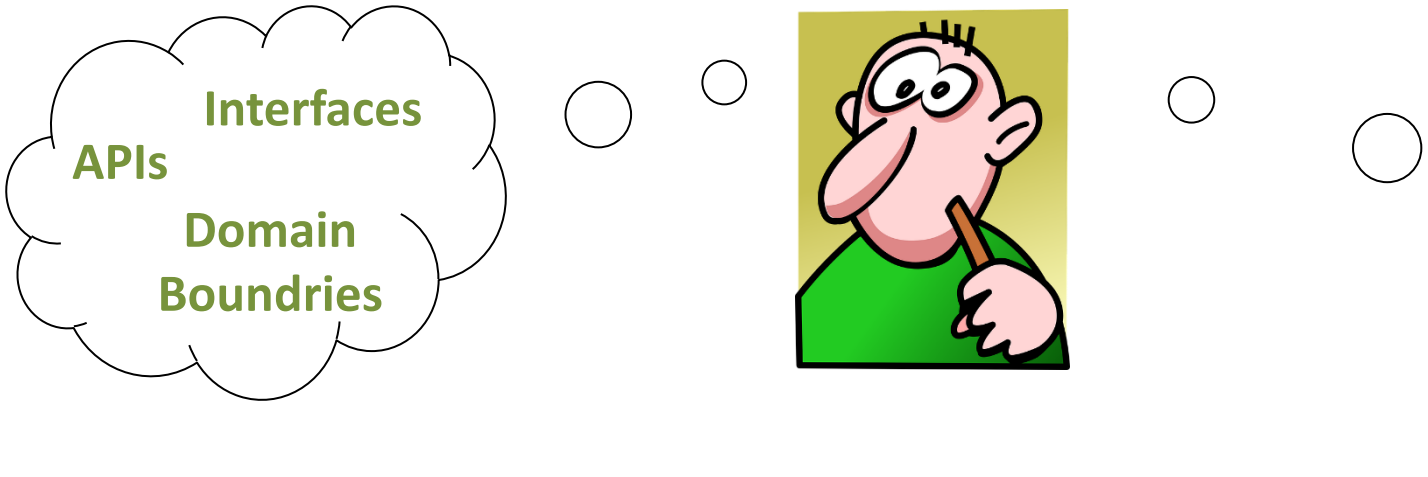


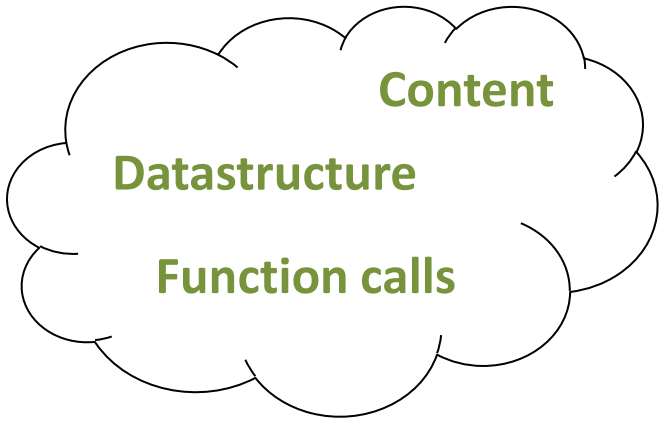
Coupling

One main problem to change software are **dependencies**



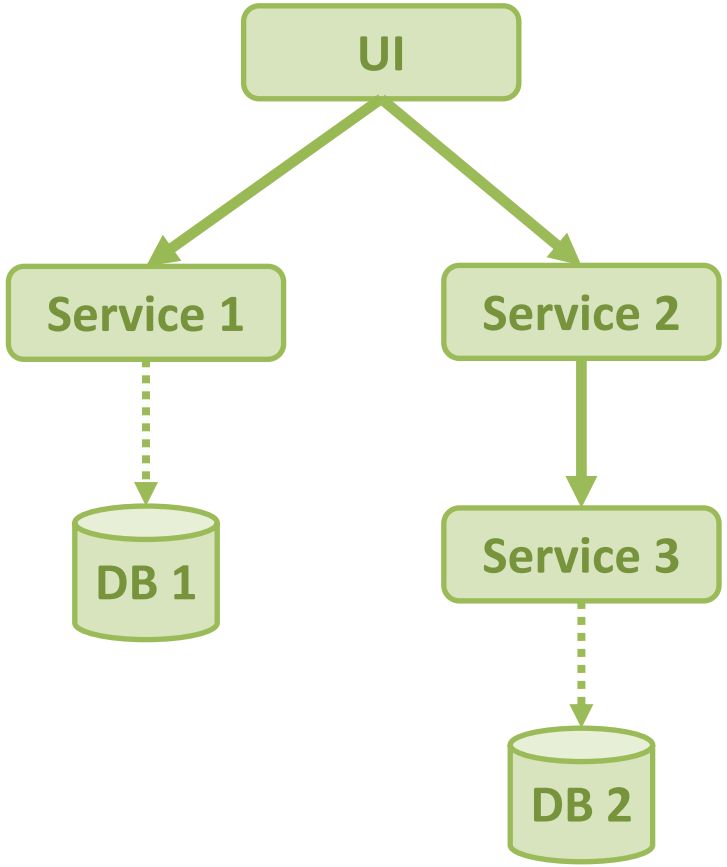
Loose Coupling





Microservice architecture

- Small independent, specialized services
- Own isolated data persistence
- Communication takes place via well-defined APIs



The good

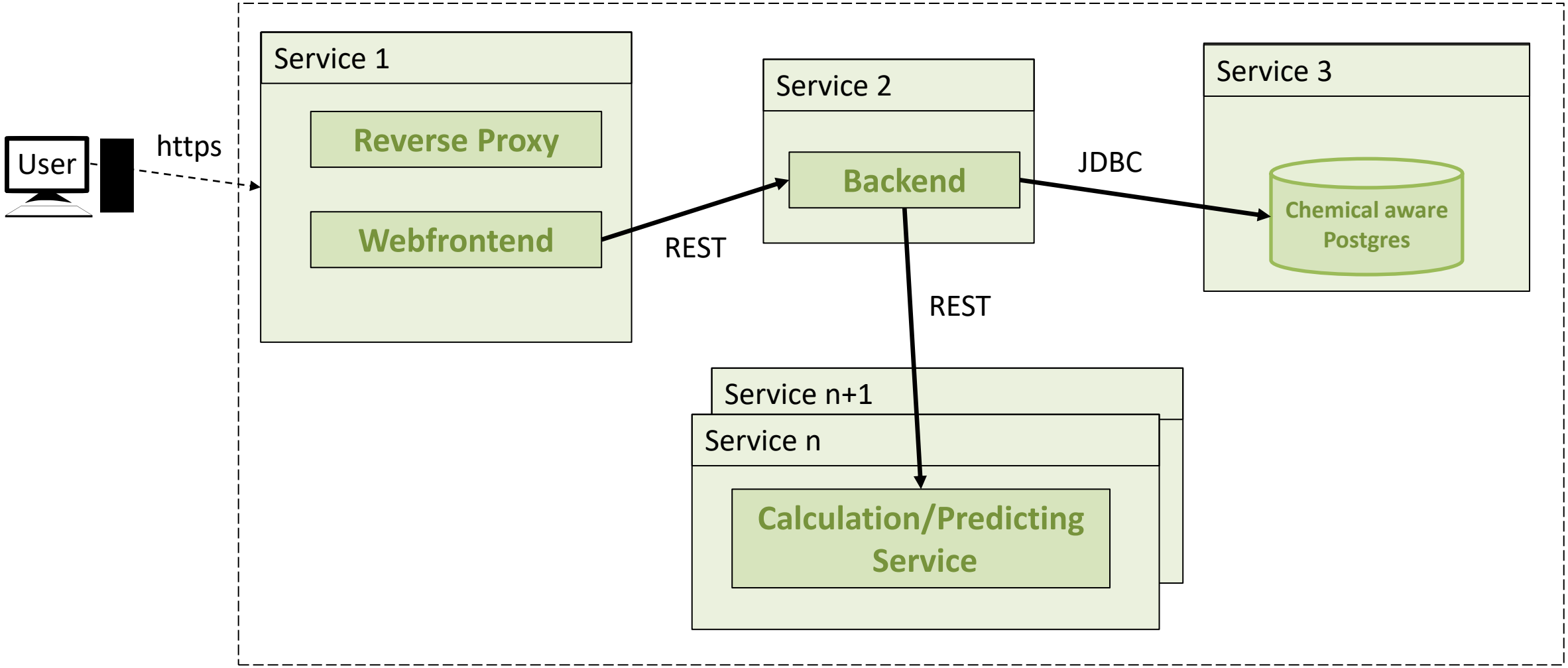
- „Small“ codebase, can be implemented independently
- High scalability
- Technological flexibility
- Reliability

The bad

- More complex than monolith
- Deployment more complex
- Data Consistency
- Communication slower

O.Al-Debagy, P.Martinek, *A Comparative Review of Microservices and Monolithic Architectures, 2018*

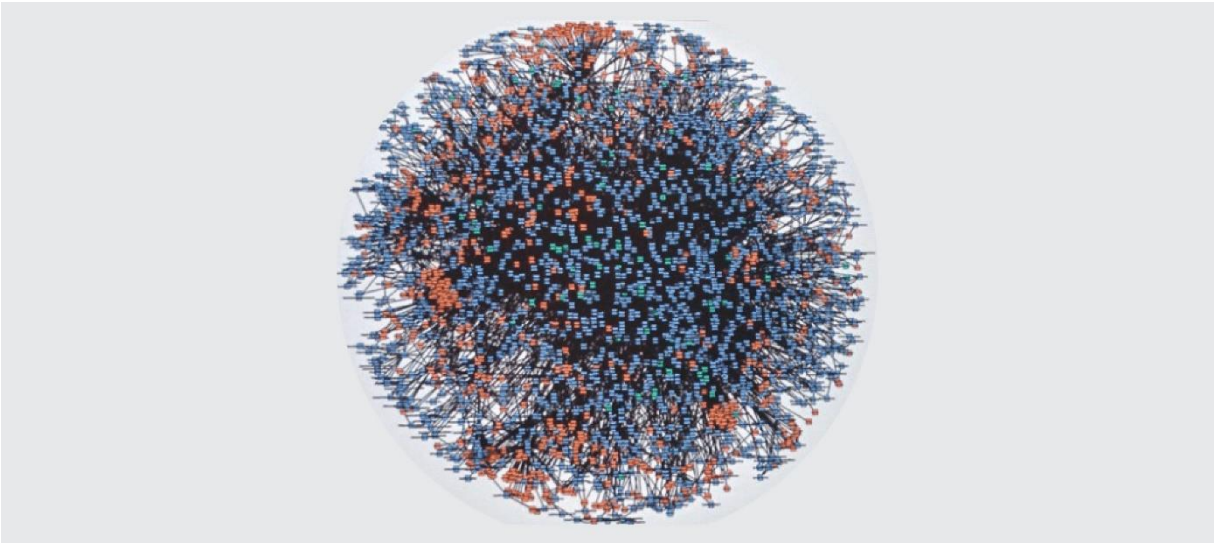
Microservice architecture – Examples



Microservice architecture – Examples - Netflix & Amazon

Netflix-Death-Star Architecture

Amazon claims to have reduced the cost of its video streaming service by 90% by switching from a microservice/serverless architecture to a monolith

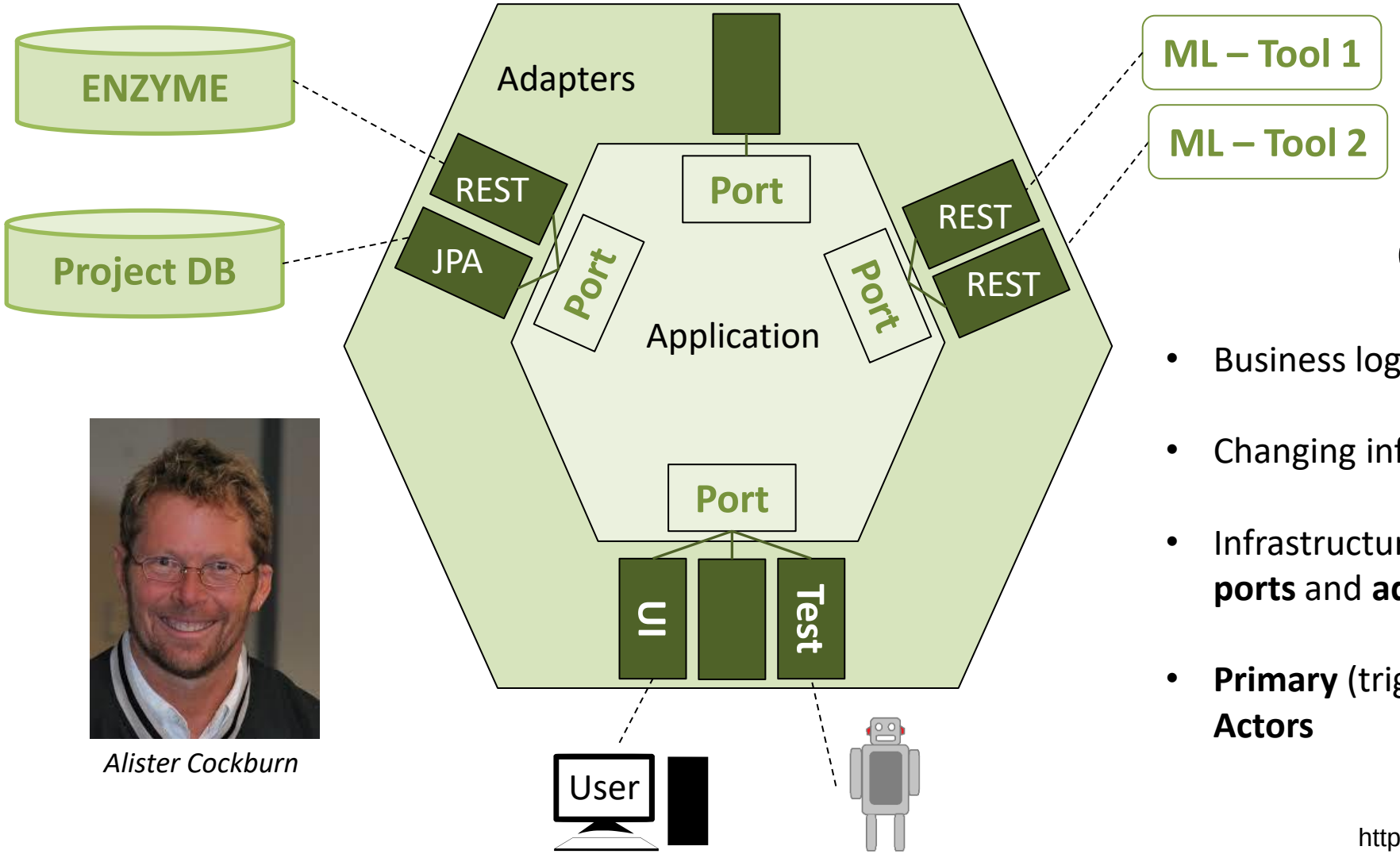


Amazon, 2000

„Microservices and serverless components are tools that do work at high scale, but whether to use them over monolith has to be made on a case-by-case basis"

Marcin Kolney, Amazon Prime Senior Software Development Engineer

Ports and Adapters/Hexagonal architecture



Alister Cockburn

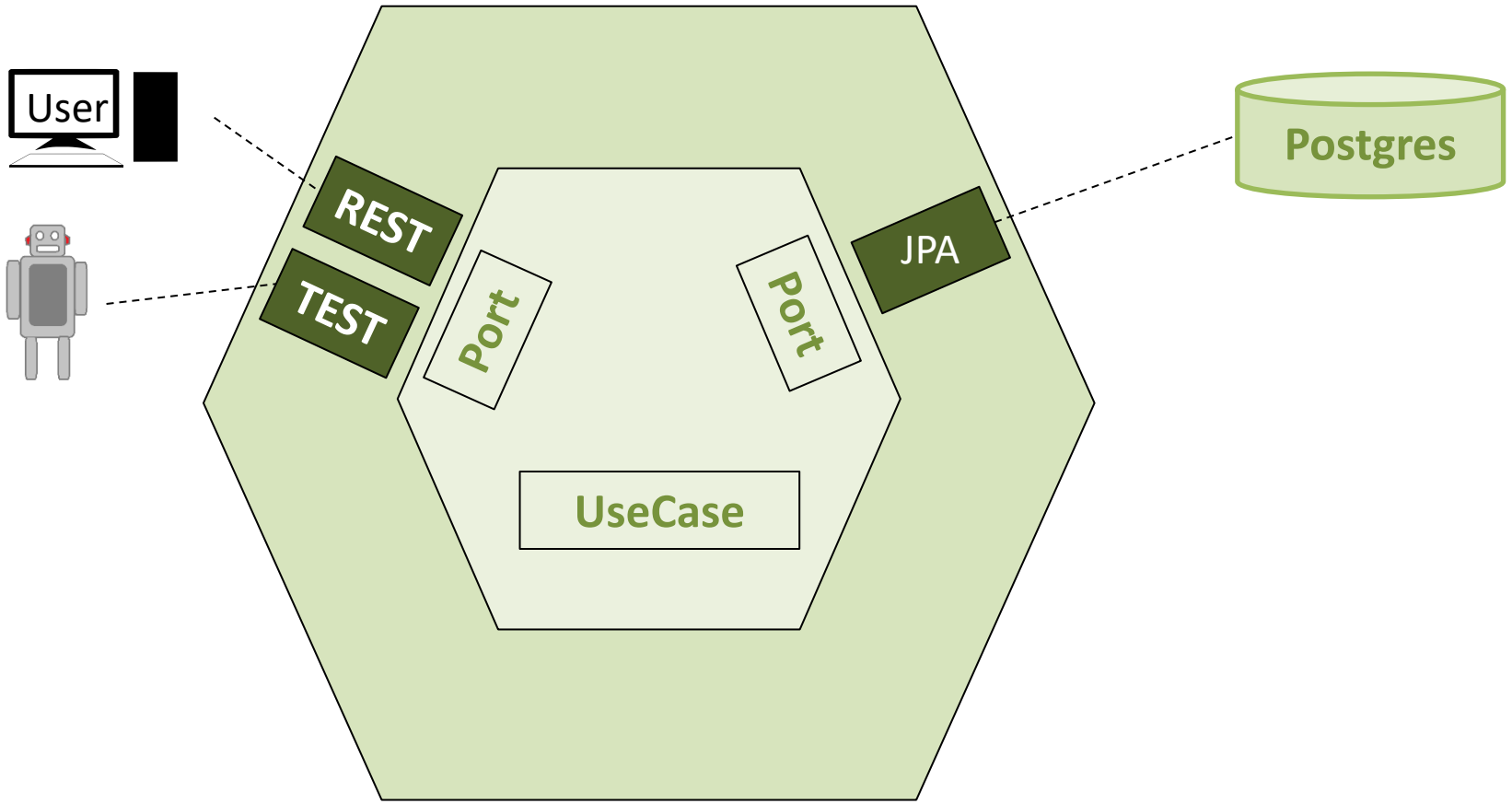
Core Ideas

- Business logic is **isolated** from infrastructure
- Changing infrastructure does not effect BL
- Infrastructure and BL communicate with **ports** and **adapters** (loose coupling)
- **Primary** (triggers an usecase) and **Secondary Actors**

https://alistair.cockburn.us/hexagonal-architecture/, **2005**

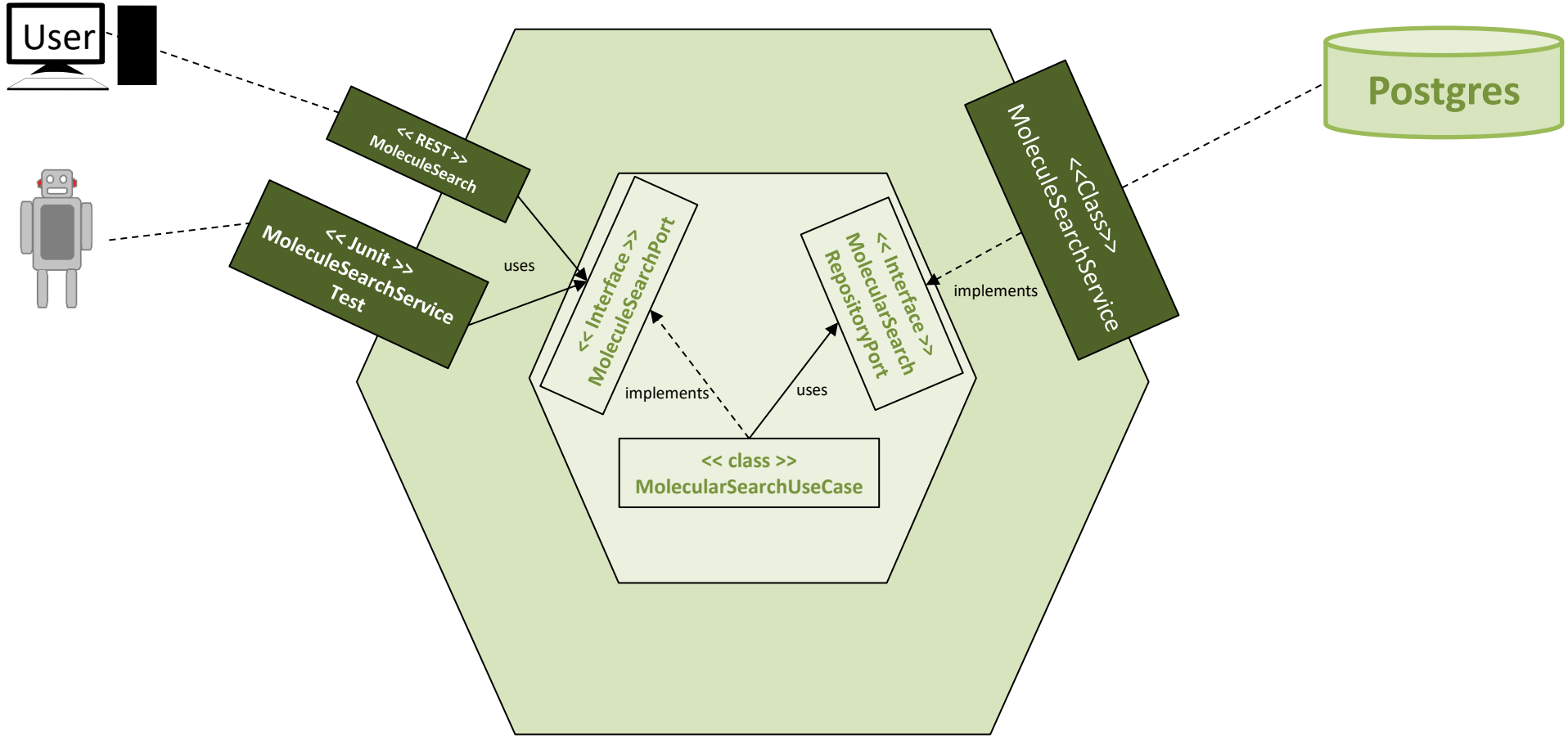
Ports and Adapters – Example

UseCase : Search for molecule with substructure search



Ports and Adapters – Example

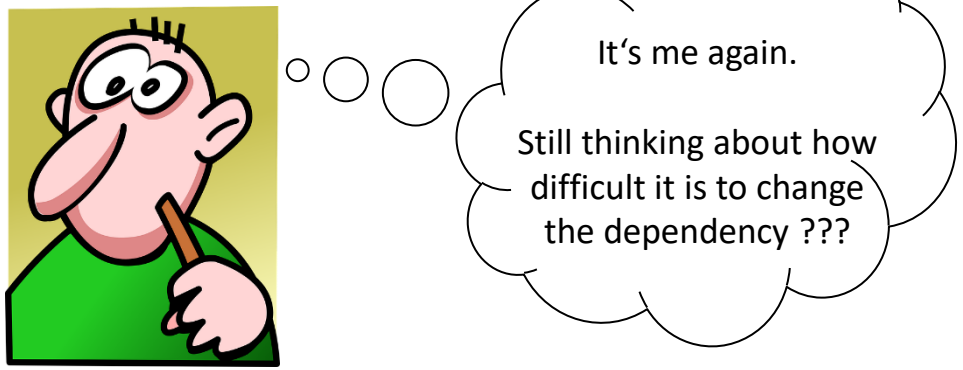
UseCase : Search for molecule with substructure search



Ports and Adapters – Example

Adaper – Junit Test

```java
@ExtendWith(PostgresqlContainerExtension.class)
@ExtendWith(ArquillianExtension.class)
public class MolecularSearchUseCaseTest extends TestBase {

    @Inject
    private MoleculeSearchUseCasePort port;

    @Test
    public void find_Reaction_with_Valid_Smile() {
        List<Molecule> reactions = port.getMoleculesBySubstructure("O=Cc1ccccc1");
        Assert.assertEquals(1, reactions.size());

    }

    @Deployment
    public static WebArchive createDeployment() {
        return prepareDeployment("MolecularSearchUseCaseTest.war")
                .addClass(MolecularSearchServiceStub.class)
                .addClass(MolecularSearchUseCase.class);
    }
}
```



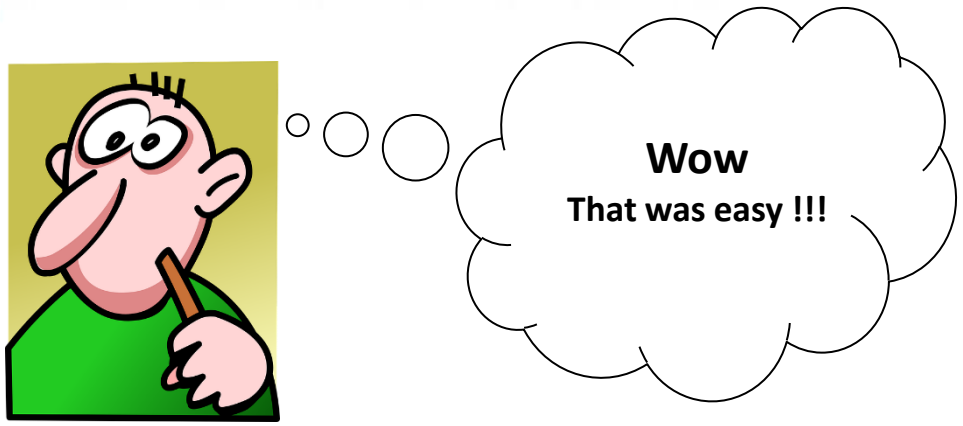
Adaper – REST

```java
@Path("/molecules")
@Stateless
public class MoleculeSearch {

    @Inject
    private MoleculeSearchUseCasePort port;

    private static final Logger LOGGER = LogManager.getLogger();
    ObjectMapper jsonSerializer = new ObjectMapper();

    @GET
    @Produces("text/json")
    public Response getMolecule(
            @QueryParam("smarts") String smiles,
            @QueryParam("search_type") String searchType,
            @QueryParam("similarity_value") float similarityValue
    ) throws Exception {
        try {
            MoleculeSearchType.valueOf(searchType);
        } catch (Exception e) {
            return Response
                    .status(500)
                    .header("error-message", "param 'search_type' " + searchType
                    .header("Access-Control-Allow-Origin", "*")
                    .build();
        }
        List<Molecule> molecules = port.getMoleculesBySubstructure(smiles);

        return Response
                .ok(jsonSerializer.writeValueAsString(molecules))
                .build();
    }
}
```

Ports and Adapters – Example

Adaper – Junit Test

```java
@ExtendWith(PostgresqlContainerExtension.class)
@ExtendWith(ArquillianExtension.class)
public class MolecularSearchUseCaseTest extends TestBase {

    @Inject
    private MoleculeSearchUseCasePort port;

    @Test
    public void find_Reaction_with_Valid_Smile() {
        List<Molecule> reactions = port.getMoleculesBySubstructure("O=Cc1ccccc1");
        Assert.assertEquals(1, reactions.size());

    }

    @Deployment
    public static WebArchive createDeployment() {
        return prepareDeployment("MolecularSearchUseCaseTest.war")
                .addClass(MolecularSearchServiceStub.class)
                .addClass(MolecularSearchUseCase.class);
    }
}
```



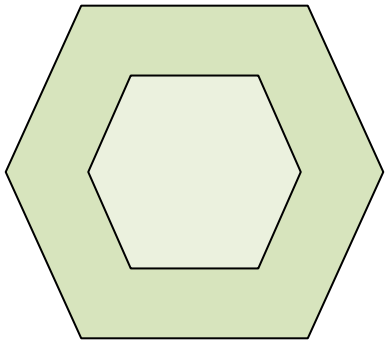
Adaper – REST

```java
@Path("/molecules")
@Stateless
public class MoleculeSearch {

    @Inject
    private MoleculeSearchUseCasePort port;

    private static final Logger LOGGER = LogManager.getLogger();
    ObjectMapper jsonSerializer = new ObjectMapper();

    @GET
    @Produces("text/json")
    public Response getMolecule(
            @QueryParam("smarts") String smiles,
            @QueryParam("search_type") String searchType,
            @QueryParam("similarity_value") float similarityValue
    ) throws Exception {
        try {
            MoleculeSearchType.valueOf(searchType);
        } catch (Exception e) {
            return Response
                    .status(500)
                    .header("error-message", "param 'search_type' " + searchType
                    .header("Access-Control-Allow-Origin", "*")
                    .build();
        }
        List<Molecule> molecules = port.getMoleculesBySubstructure(smiles);

        return Response
                .ok(jsonSerializer.writeValueAsString(molecules))
                .build();
    }
}
```

API – A practical definition

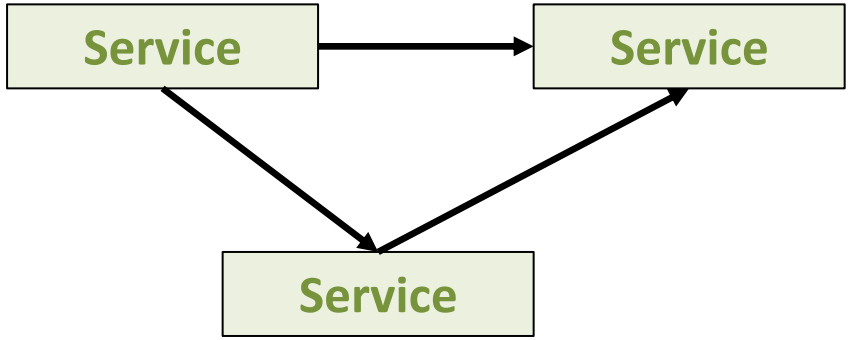
Both architectures define **boundaries**

Adapters and application are conceptually separated from each other



Communication over a function call defined by an interface

Services are conceptually and often physically separated from each other

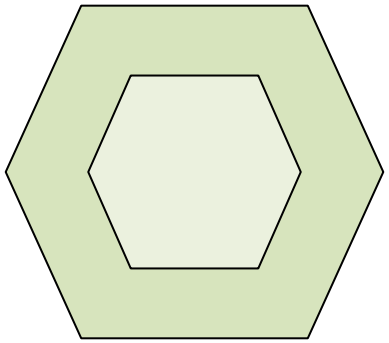


Communication over a network based protocol like REST or JDBC

API – A practical definition

Both architectures define **boundaries**

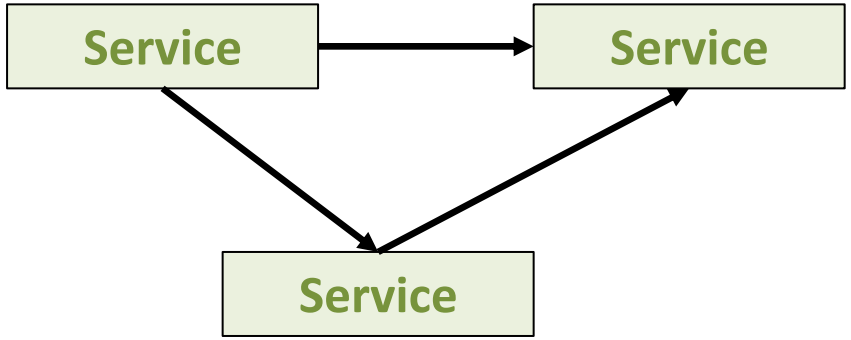
Adapters and application are conceptually separated from each other



Communication over a function call defined by an interface

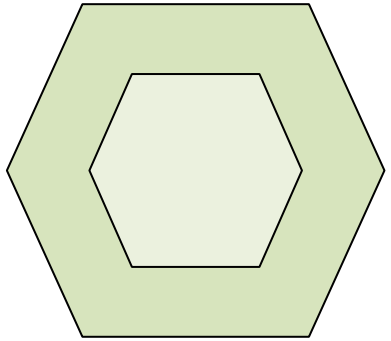
Can we find a definition that does justice to both
??

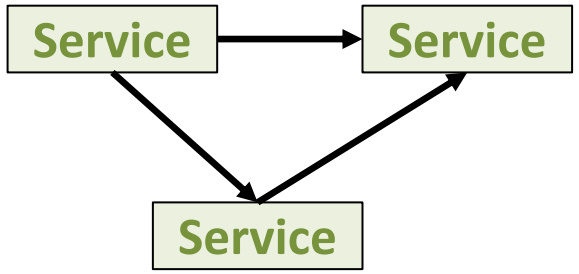
Services are conceptually and often physically separated from each other



Communication over a network based protocol like REST or JDBC

API – Application programming interface





- both represent an abstraction of the underlying implementation → Domain
- both introduce a specification of the required and supplied data → Signature
- both define a semantics or a behavior of the exchange → Protocol

J. Gough, D Bryant, M Auburn, *Mastering API Architecture*,**2023,** S. XXV

API - Java function call defined by an interface

The keywords and the environment (java) defines the protocol

The naming of the method and parameters indicates it‘s functionallity

Protocol

Domain

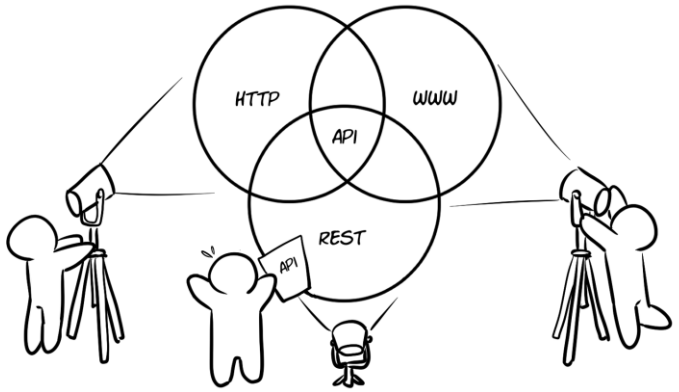
```
public interface MolecularSearchRepositoryPort {
    List<Molecule> getMoleculesBySubstructure(String smarts);
```

Signature

The method name,input and output arguments and their types

- There is no statement about **how** something is done only **what**
- There is no indication of whether the request is idempotent.(Idempotent calls would return each time the same result. This can be defined in part by the protocol type)

API – REST Definition



Set of rules/constraints

- Client-server
- **Stateless systems**
- Cache
- **Uniform interface**
- Layered system
- ***Code on demand***

REST : Representational State Transfer

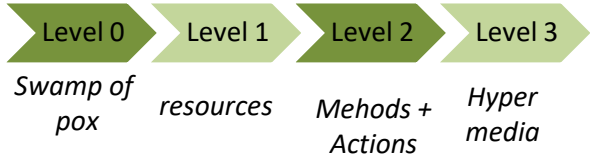
Not a standard or protocol, like HTTP but more of a style

Protocol: Not strictly defined but in reality, 99% http(s)

Domain: domain specific resources

Signature: Query parameters, server address+path and Response

Maturity levels of REST



R. Fielding, *Architectural Styles and the Design of Network-based Software Architectures*,**2000**

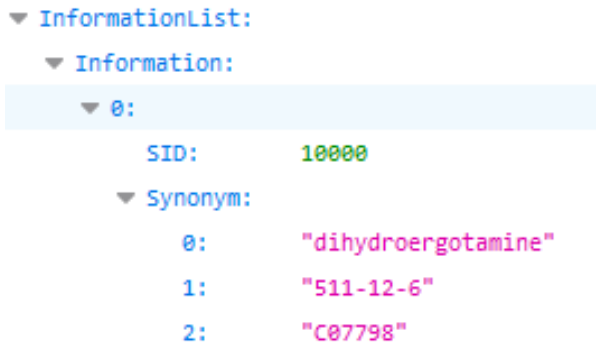
M. Amundsen, Design and Build Great Web APIs, **2020**

API – REST Example - Browser

pubchem - searching for synonyms

https://pubchem.ncbi.nlm.nih.gov/rest/pug/substance/sid/10000/synonyms/json

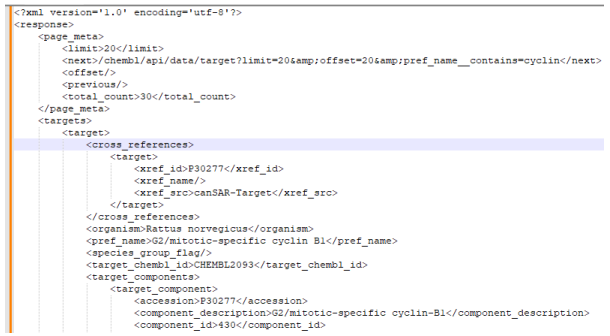
protocol server path



ChEMBL - searching for molecules with name pattern

https://www.ebi.ac.uk/chembl/api/data/target?pref_name__contains=cyclin

protocol server path parameter



https://pubchem.ncbi.nlm.nih.gov/docs/pug-rest-tutorial

https://www.ebi.ac.uk/chembl/ws

API – REST Example javascript and python

Contains information like IP address, server, accepted data formats, …

context specific information like parameters

The request method is also defined in the header

- **GET** : Used to receive information
- **POST**: send information to server
- **PUT**: update information on server
- **DELETE**: remove information on server

API – REST Example javascript and python

Contains information like IP address, server, accepted data formats, …

context specific information like parameters

The request method is also defined in the header

- **GET** : Used to receive information
- **POST**: send information to server
- **PUT**: update information on server
- **DELETE**: remove information on server

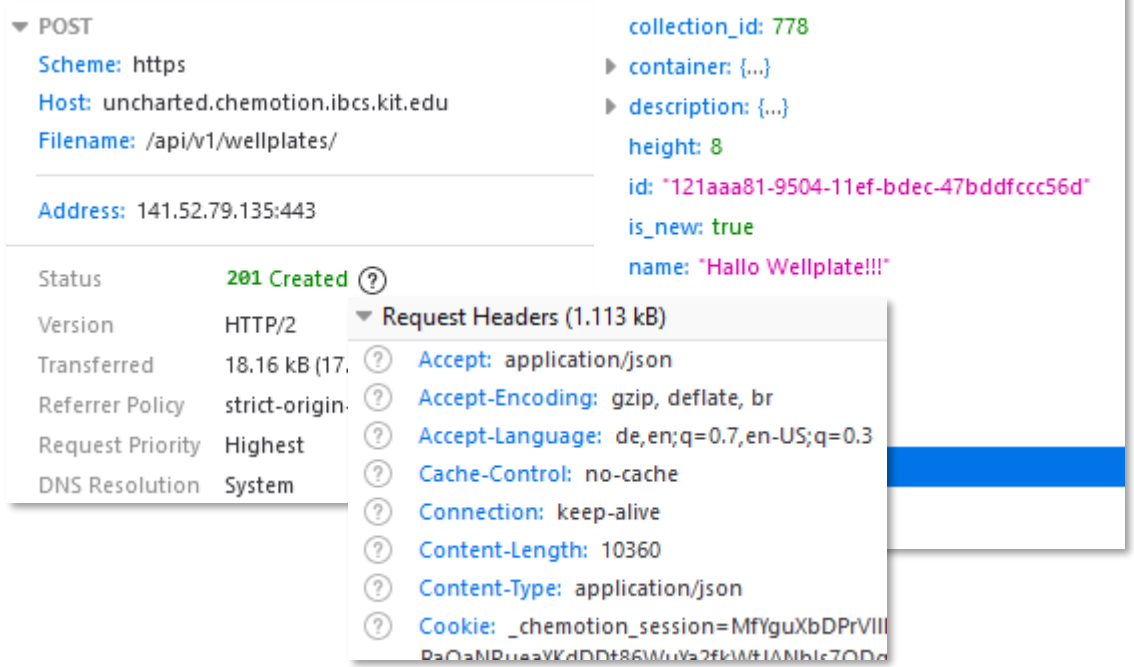
```
const promise = () => fetch('/api/v1/wellplates/', {
  credentials: 'same-origin',
  method: 'post',
  headers: {
    Accept: 'application/json',
    'Content-Type': 'application/json'
  },
  body: JSON.stringify(wellplate.serialize())
})
```

Example java script

```
import requests

requests.post(
  '/api/v1/wellplates/',
  data=json.dumps({'wellplate': ' … ' }),
  headers={'Content-type':'application/json'}
);
```

Example python

Both create



API – REST Example

https://pubchem.ncbi.nlm.nih.gov/rest/pug/substance/sid/10000/synonyms/json

https://www.ebi.ac.uk/chembl/api/data/target?pref_name__contains=cyclin

But how do i know what paths or parameters i have to enter ???

API – Open API

https://pubchem.ncbi.nlm.nih.gov/rest/pug/substance/sid/10000/synonyms/json

https://www.ebi.ac.uk/chembl/api/data/target?pref_name__contains=cyclin

But how do i know what paths or parameters i have to enter ???

Many but not all web services offer sufficient API documentation and they vary in form and quality

https://www.openapis.org/

API – Open API

https://pubchem.ncbi.nlm.nih.gov/rest/pug/substance/sid/10000/synonyms/json

https://www.ebi.ac.uk/chembl/api/data/target?pref_name__contains=cyclin

But how do i know what paths or parameters i have to enter ???

OpenAPI

The OpenAPI Specifications provides a formal standard for describing HTTP APIs

https://www.openapis.org/

API – Open API Specification 3.0


```yaml
openapi: 3.0.3
info:
  title: Molecule Search API
  version: 0.1.0
paths:
  /molecules/:
    get:
      summary: Search for a molecule
      description: 'Search for a molecule'
      operationId: moleculeSearchID
      parameters:
        - in: query
          name: smarts
          example: CCO
          schema:
            type: string
        - in: query
          name: search_type
          schema:
            type: string
            enum:
            - SUBSTRUCTURE
            - EXACT
            - SIMILARITY
            default: EXACT
        - in: query
          name: similarity_value
          schema:
            type: number
            format: float
            minimum: 0
            maximum: 1
            example: 0.5
      responses:
        '200':
          description: successful operation
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/Molecule'
            application/xml:
              schema:
                $ref: '#/components/schemas/Molecule'
        '404':
          description: Molecule not found
```

```yaml
components:
  schemas:
    Molecule:
      type: object
      properties:
        id:
          type: integer
          format: int64
          example: 10
        molecule:
          type: string
          description: Molecule description as sdf or mol file
        iupac:
          type: string
        smiles:
          type: string
        chebiid:
          type: string
        inchi:
          type: string
```

API – Open API Specification 3.0


```yaml
openapi: 3.0.3
info:
  title: Molecule Search API
  version: 0.1.0
paths:
  /molecules/:
    get:
      summary: Search for a molecule
      description: 'Search for a molecule
      operationId: moleculeSearchID
      parameters:
        - in: query
          name: smarts
          example: CCO
          schema:
            type: string
        - in: query
          name: search_type
          schema:
            type: string
            enum:
            - SUBSTRUCTURE
            - EXACT
            - SIMILARITY
            default: EXACT
        - in: query
          name: similarity_value
          schema:
            type: number
            format: float
            minimum: 0
            maximum: 1
            example: 0.5
      responses:
        '200':
          description: successful operation
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/Molecule'
            application/xml:
              schema:
                $ref: '#/components/schemas/Molecule'
        '404':
          description: Molecule not found
```

```java
@Path("/molecules")
@Stateless
public class MoleculeSearch {

    @Inject
    private MoleculeSearchPort port;

    private static final Logger LOGGER = LogManager.getLogger();
    ObjectMapper jsonSerializer = new ObjectMapper();

    @GET
    @Produces("text/json")
    public Response getMolecule(
            @QueryParam("smiles") String smiles,
            @QueryParam("search_type") String searchType,
            @QueryParam("similarity_value") float similarityValue
    ) throws Exception {
        List<Molecule> molecules = port.getMoleculesBySubstructure(smiles);

        return Response
                .ok(jsonSerializer.writeValueAsString(molecules))
                .header("Access-Control-Allow-Origin", "*")
                .build();
    }
}
```

```yaml
components:
  schemas:
    Molecule:
      type: object
      properties:
        id:
          type: integer
          format: int64
          example: 10
        molecule:
          type: string
          description: Molecule description as sdf or mol file
        iupac:
          type: string
        smiles:
          type: string
        chebiid:
          type: string
        inchi:
          type: string
```

API – Open API Specification 3.0


```yaml
openapi: 3.0.3
info:
  title: Molecule Search API
  version: 0.1.0
paths:
  /molecules/:
    get:
      summary: Search for a molecule
      description: 'Search for a molecule'
      operationId: moleculeSearchID
      parameters:
        - in: query
          name: smarts
          example: CCO
          schema:
            type: string
        - in: query
          name: search_type
          schema:
            type: string
            enum:
            - SUBSTRUCTURE
            - EXACT
            - SIMILARITY
            default: EXACT
        - in: query
          name: similarity_value
          schema:
            type: number
            format: float
            minimum: 0
            maximum: 1
            example: 0.5
      responses:
        '200':
          description: successful operation
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/Molecule'
            application/xml:
              schema:
                $ref: '#/components/schemas/Molecule'
        '404':
          description: Molecule not found
```

What's in here for me???

```yaml
components:
  schemas:
    Molecule:
      type: object
      properties:
        id:
          type: integer
          format: int64
          example: 10
        molecule:
          type: string
          description: Molecule description as sdf or mol file
        iupac:
          type: string
        smiles:
          type: string
        chebiid:
          type: string
        inchi:
          type: string
```

Automatic code generation

- Automatic code generation for clients and servers for various languages
- Automatic documentation generation for different styles
- Even database shemas could be generated

```java
@Path("/molecules/")
@Api(value = "/", description = "")
public interface DefaultApi  {

    /**
     * Search for a molecule
     *
     * Search for a molecule
     *
     */
    @GET

    @Produces({ "application/json", "application/xml" })
    @ApiOperation(value = "Search for a molecule", tags={  })
    @ApiResponses(value = {
        @ApiResponse(code = 200, message = "successful operation", response = Molecule.class),
        @ApiResponse(code = 404, message = "Molecule not found") })
    public Molecule moleculeSearchID(
        @QueryParam("smiles") String smiles,
        @QueryParam("search_type") @DefaultValue("EXACT")String searchType,
        @QueryParam("similarity_value") @DecimalMin("0") @DecimalMax("1") Float similarityValue);
}
```

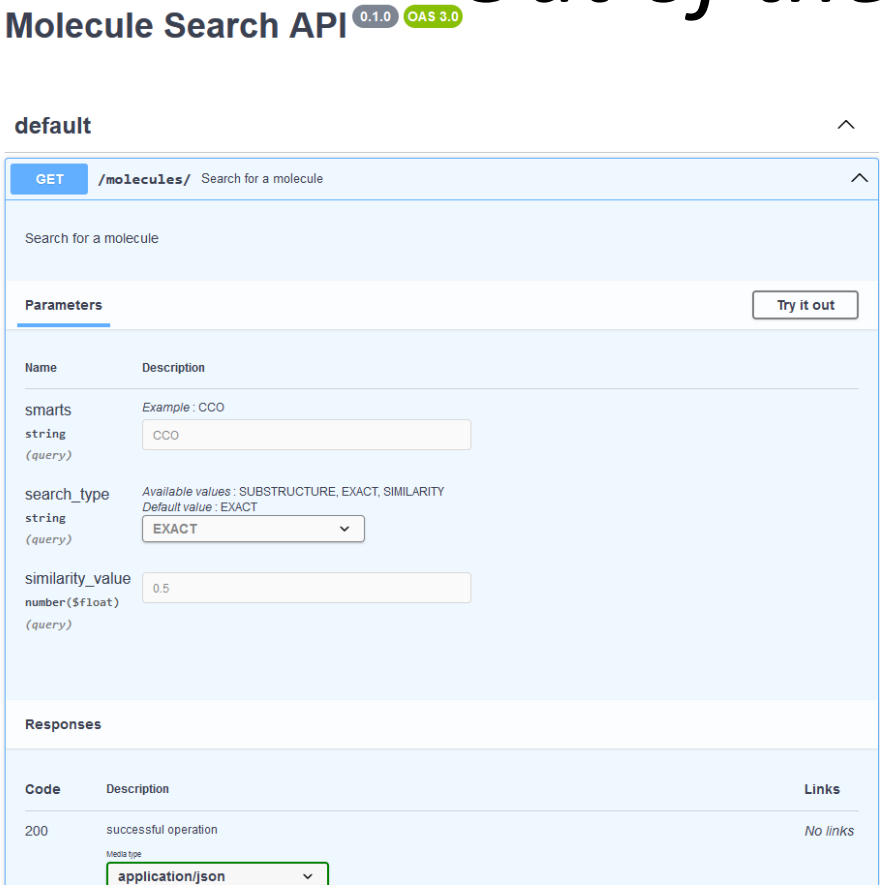
```java
public class Molecule  {

  @ApiModelProperty(example = "10", value = "")
  private Long id;

  @ApiModelProperty(value = "Molecule description as sdf or mol file")
 /**
   * Molecule description as sdf or mol file
   **/
  private String molecule;

  @ApiModelProperty(value = "")
  private String iupac;

  @ApiModelProperty(value = "")
  private String smiles;

  @ApiModelProperty(value = "")
  private String chebiid;

  @ApiModelProperty(value = "")
  private String inchi;
```

Automatically generated java REST server endpoint

https://openapi-generator.tech/

API – Open API – documentation

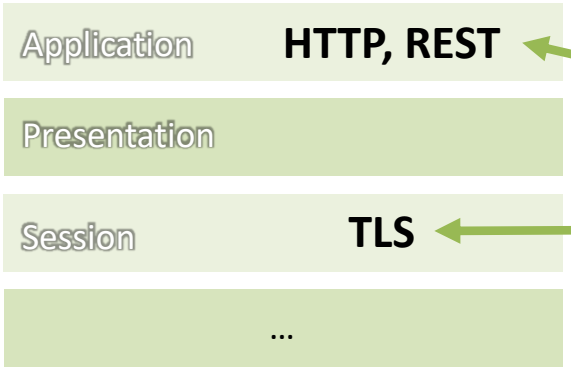
Out of the box documentation



- Extensive documentation of the API routes
- Detailed description of the data resources
- Playground for trying out the API
- Ways to simulate various security mechanisms

https://editor.swagger.io/

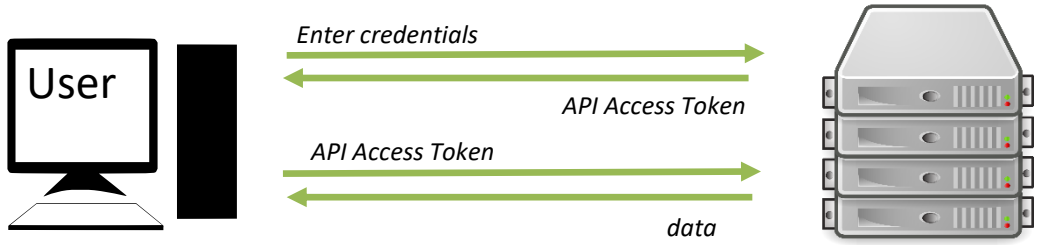
API – Securing API endpoints



OSI Layer Model

Authentication and Authorization

Encryption



JSON Web Token (JWT)

Compact, URL-safe representation of claims between 2 parties

xxxx.yyyy.zzzz

Header: *defines type of token and signing algorithm*

Payload: *contains the claims*

Signature: *Signature for validation*

```
{
  "sub": "1234567890",
  "name": "John Doe",
  "admin": true
}
```

Authentification < > Authorisation

Who **What**

More sophisticated mechanisms:
OAuth2.0 , OpenID connect, …

M.Jones, J. Bradley,N. Sakimura, *https://datatracker.ietf.org/doc/html/rfc7519*, **2015**

M. Amundsen, Design and Build Great Web APIs, **2020**, p. 207ff

API – Testing of APIs

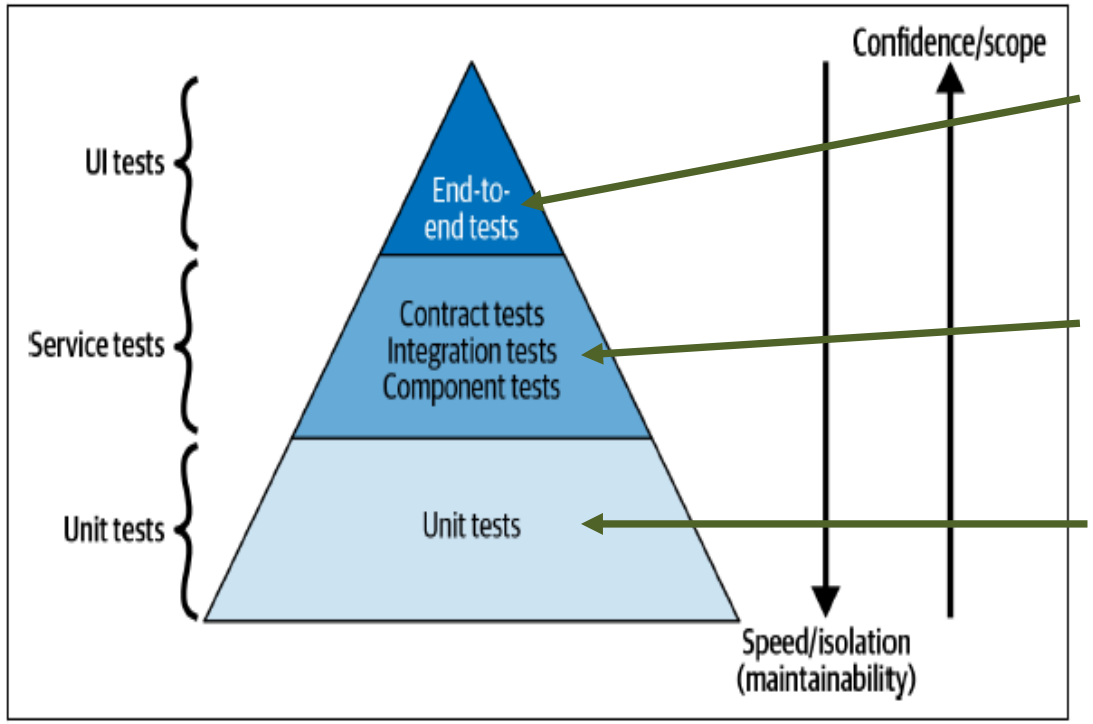


Figure 2-3. The test pyramid, showing the proportion of tests desired

- *Simulates a client and a hole usecase*
- *Frameworks: Cypress, Playwright, …*

- *By using the Hexagon architecture, components could be mocked*
- *Using frameworks could help: testcontainers, virtual archives, …*

- *Usually APIs are tested in upper layers*

!! All tests in all layers must be executed automatically regardless of the environment !!

J. Gough, D Bryant, M Auburn, *Mastering API Architecture*,**2023,** S. 50

Testing with github actions

"… Automate, customize and execute your software development workflows directly in your repository."

- **Workflow**: configurable automated sequence of tasks
- **Event**: specific occurrence in your repository that triggers a workflow
- **Job**: collection of steps that are executed in a workflow
- **Action**: individual building blocks of a workflow
- **Runner**: computer on which your workflows are executed
- **Step**: smallest executable unit in a job

```yaml
name: BioCasNavi - frontend CI

on:
  push:
    branches: [ "main" ]
  pull_request:
    branches: [ "main" ]

jobs:
  test-job:
    runs-on: ubuntu-latest
    steps:
    - name: Checkout
      uses: actions/checkout@v3

    - name: Use Node 20.x
      uses: actions/setup-node@v3
      with:
        node-version: '20.x'

    - name: Install dependencies
      working-directory: ./frontend
      run: npm install


    - name: Test (jest and react-testing-library)
      working-directory: ./frontend
      run: npm test

```

https://docs.github.com/de/actions

Github actions - examples

Hundreds of provided actions for many areas

cypress-io/github-action

```yaml
name: End-to-end tests
on: push
jobs:
  cypress-run:
    runs-on: ubuntu-22.04
    steps:
      - name: Checkout
        uses: actions/checkout@v4
      # Install npm dependencies, cache them correctly
      # and run all Cypress tests
      - name: Cypress run
        uses: cypress-io/github-action@v6
```

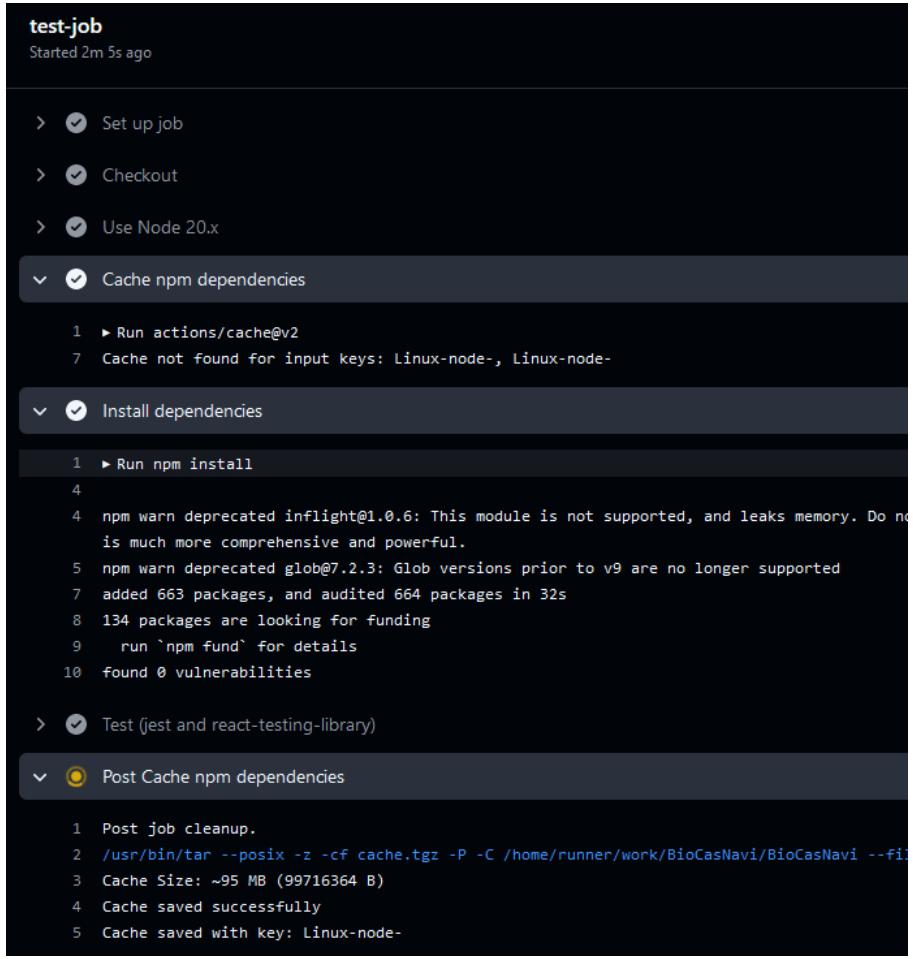
codecov-action

```yaml
steps:
- uses: actions/checkout@main
- uses: codecov/codecov-action@v4
  with:
    fail_ci_if_error: true # optional (default = false)
    files: ./coverage1.xml,./coverage2.xml # optional
    flags: unittests # optional
    name: codecov-umbrella # optional
    token: ${{ secrets.CODECOV_TOKEN }} # required
    verbose: true # optional (default = false)
```

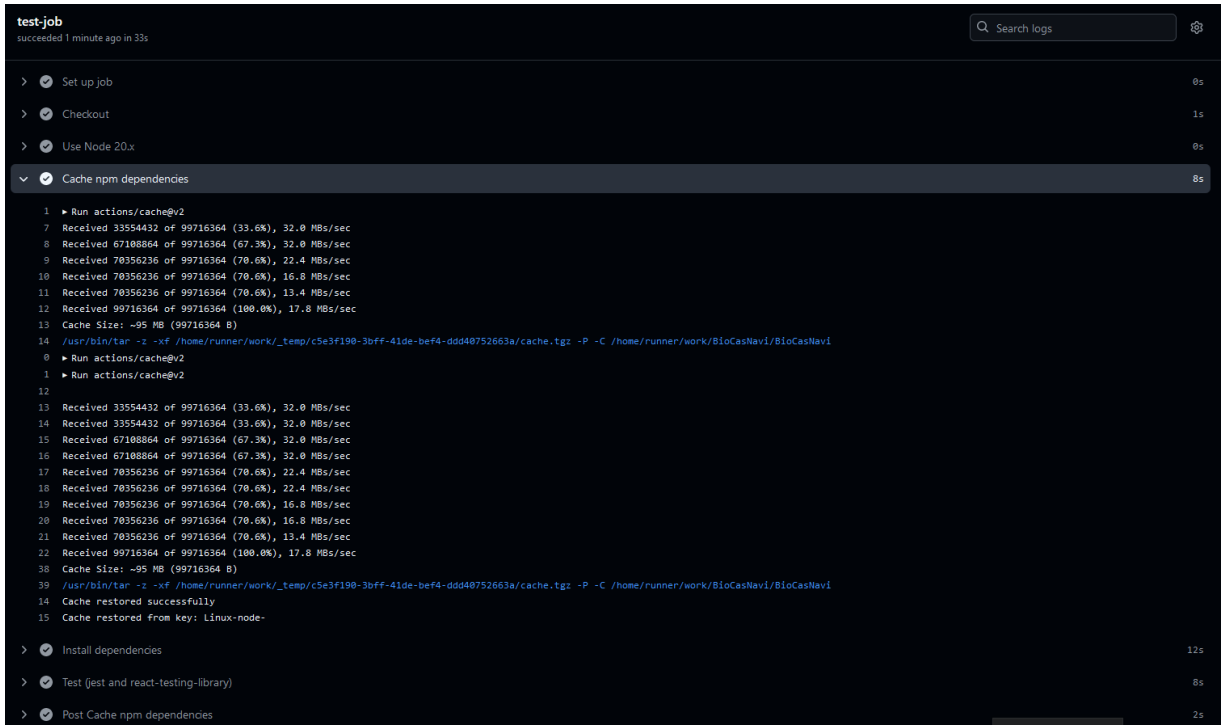
```yaml
test-job:
  runs-on: ubuntu-latest
  steps:
  - name: Checkout
    uses: actions/checkout@v3

  - name: Use Node 20.x
    uses: actions/setup-node@v3
    with:
      node-version: '20.x'

  - name: Cache npm dependencies
    uses: actions/cache@v2
    with:
      path: '~/.npm'
      key: ${{ runner.os }}-node-${{ hashFiles('**/package-lock.json') }}
      restore-keys: |
        ${{ runner.os }}-node-

  - name: Install dependencies
    working-directory: ./frontend
    run: npm install


  - name: Test (jest and react-testing-library)
    working-directory: ./frontend
    run: npm test
```

https://github.com/sdras/awesome-actions

Github action - caching



Execution without cached dependencies



Execution with cached dependencies

Workflow execution time : **33s** vs. **3,6s**

Thank you

I would like to thank

Prof. Ludger Wessjohann

Dr. Mehdi Davari

Dr. Frank Broda

NWC department & Dr. Steffen Neumanns group

Funded by dip