

Prêt-à-LLOD

Teanga

Bernardo Stearns



HORIZON 2020¹

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 825182

Introduction

- Contributions to the pret-a-lod project, a project that aim to increase the uptake of language technologies by exploiting the combination of linked data, language technologies and software development standards
- Specifically the talk is about teanga, which is command line interface that launches a local web application and enables users to create, run and inspect (workflows of dockerized rest apis)
- How employing both software development standards and linked data can benefit a platform for creating nlp workflows



NUI Galway
OÉ Gaillimh



Universidad
Zaragoza



Politécnica
Universität Bielefeld

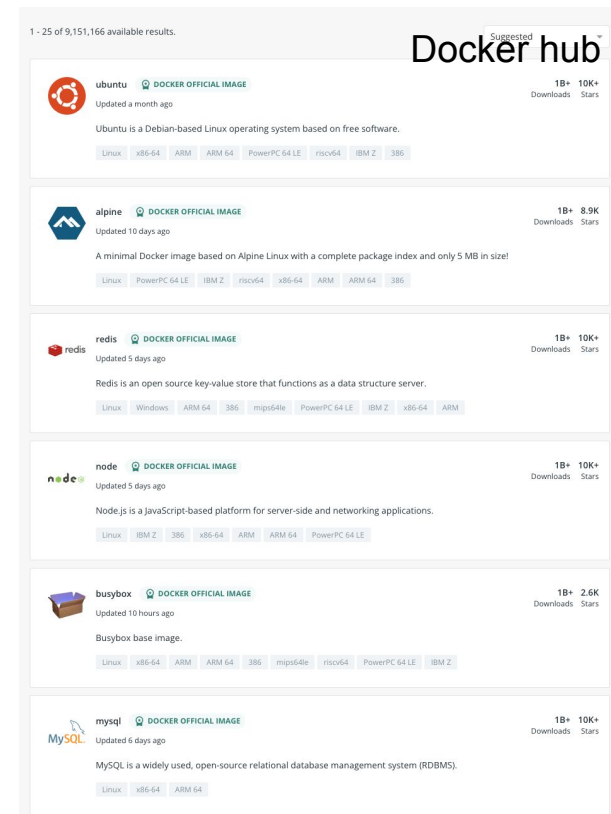


semalytix



Motivation

- Software Engineering wide adoption of microservices against monolithic services due to success of containerization technologies for isolating applications environments.



Motivation

- Software Engineering wide adoption of microservices against monolithic services due to success of containerization technologies for isolating applications environments.
- delivery of many services that can be used in a range of different applications, such as language/nlp technologies.

DKPro

DKPro - Welcome

DKPro is a community of projects focussing on re-usable Natural Language Processing software.

dkpro-core-api-featurepath-asl
dkpro-core-api-segmentation-asl
dkpro-core-arktools-gpl
dkpro-core-berkeleyparser-gpl
dkpro-core-castransformation-asl
dkpro-core-cleanlp-asl
dkpro-core-commonscodec-asl
dkpro-core-corenlp-gpl
dkpro-core-decompounding-asl
dkpro-core-dictionaryannotator-asl
dkpro-core-frequency-asl
dkpro-core-gate-asl
dkpro-core-hunpos-asl
dkpro-core-io-text-asl
dkpro-core-io-tgrep-gpl
dkpro-core-io-web1t-asl
dkpro-core-ixa-asl
dkpro-core-jazzy-asl
dkpro-core-langdetect-asl



NUI Galway
OÉ Gaillimh



Universidad
Zaragoza



Politécnica
Universität Bielefeld



semalytix



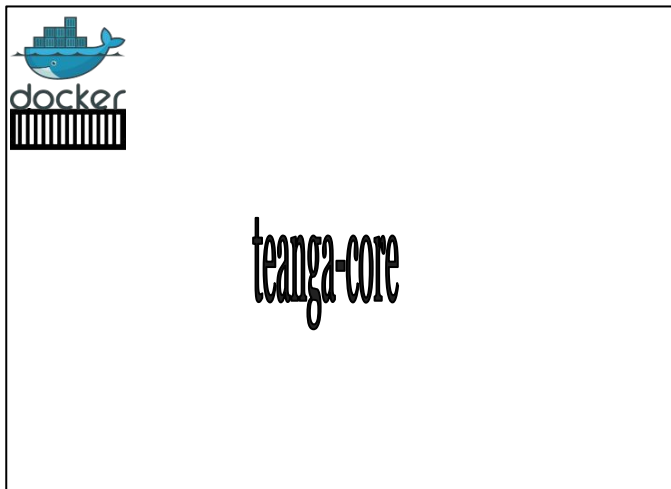
Motivation

- Software Engineering wide adoption of microservices against monolithic services due to success of containerization technologies for isolating applications environments.
- delivery of many services that can be used in a range of different applications, such as language/nlp technologies.
- Still there is a high entry barrier for using and discovering services and workflows.
- So how can we bridge the gap between a high availability of services and the difficulty of using and integrating those services ?

Motivation

- Software Engineering wide adoption of microservices against monolithic services due to success of containerization technologies for isolating applications environments.
- delivery of many services that can be used in a range of different applications, such as language/nlp technologies.
- Still there is a high entry barrier for using and discovering services and workflows.
- So how can we bridge the gap between a high availability of services and the difficulty of using and integrating those services in processing workflows we might have ?
- With solid technologies standards that describes services, guarantees portability across different systems.
- With linked data approaches that enables us to tell the compatibility between services

Technologies



Technologies

Teanga installation
and start



NUI Galway
OÉ Gaillimh



Universidad
Zaragoza



Politécnica

GOETHE
UNIVERSITÄT
FRANKFURT AM MAIN



semalytix

OXFORD
UNIVERSITY PRESS



SEMANTIC WEB COMPANY
linking data to knowledge



Derilinx
DRIVE DECISION-MAKING. INSPIRE CHANGE



Command Line Interface

- We use sh a Primitive shell command language
- The standard Unix Shell
- Most of others shell command languages follows sh standard
- Don't need to rely on having some programming language installed



Docker

- Docker is a technology for isolating applications environments, analogous to virtual machines
 - Lower effort to guarantee a reliable environment
 - This guarantees portability, if you have docker you potentially can run any docker container (if your machine can support the application size)
 - This guarantees standardization, you can run ubuntu even if you are using mac or windows

Docker

Show simple ubuntu pull



NUI Galway
OÉ Gaillimh



Universidad
Zaragoza



Politécnica



GOETHE
UNIVERSITÄT
FRANKFURT AM MAIN



semalytix



OXFORD
UNIVERSITY PRESS



SEMANTIC WEB COMPANY
linking data to knowledge



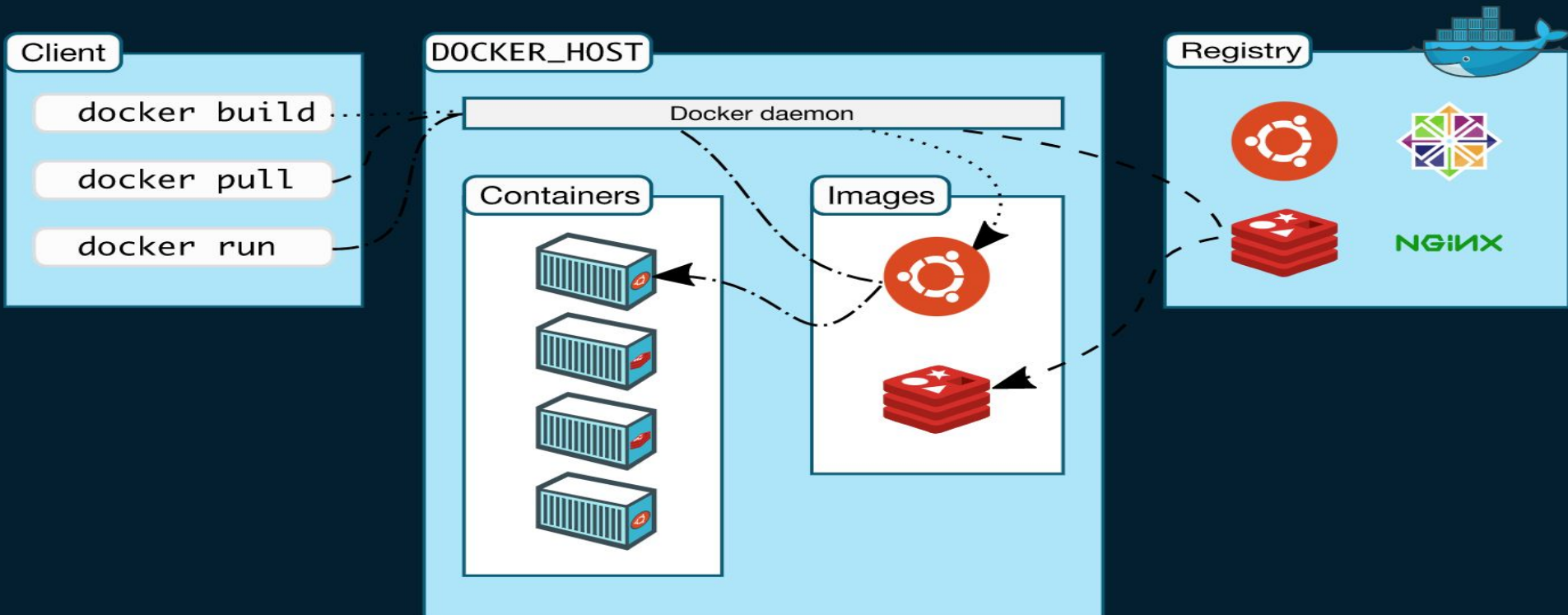
Derilinx
DRIVE DECISION-MAKING. INSPIRE CHANGE



Docker

Docker architecture

Docker uses a client-server architecture. The Docker *client* talks to the Docker *daemon*, which does the heavy lifting of building, running, and distributing your Docker containers. The Docker client and daemon *can* run on the same system, or you can connect a Docker client to a remote Docker daemon. The Docker client and daemon communicate using a REST API, over UNIX sockets or a network interface. Another Docker client is Docker Compose, that lets you work with applications consisting of a set of containers.



Airflow

- Airflow is a platform to programmatically author, schedule and monitor workflows.
- Use Airflow to author workflows as Directed Acyclic Graphs (DAGs) of tasks
- Airflow has a scheduler which executes your tasks on an array of workers defined by the user

Airflow

Backend - workflow executor



Dynamically defining execution graph

```
dag = DAG('tutorial',
          default_args=default_args,
          description='A simple tutorial DAG',
          schedule_interval=timedelta(days=1),)

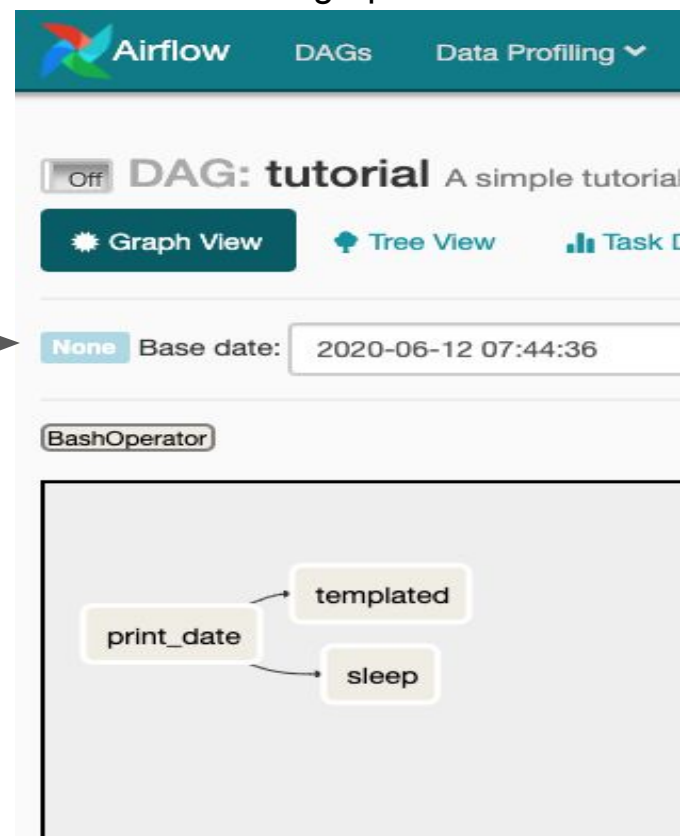
# t1, t2 and t3 are examples of
# tasks created by instantiating operators
t1 = BashOperator(
    task_id='print_date',
    bash_command='date',
    dag=dag,
)

t2 = BashOperator(
    task_id='sleep',
    depends_on_past=False,
    bash_command='sleep 5',
    retries=3,
    dag=dag,
)

t3 = BashOperator(
    task_id='templated',
    depends_on_past=False,
    bash_command=templated_command,
    params={'my_param': 'Parameter I passed in'},
    dag=dag,
)

t1 >> [t2, t3]
```

Visualizing instances of execution graph



Airflow

Providers packages [↗](#)

Providers packages include integrations with third party projects. They are updated independently of the Apache Airflow core. [Read the documentation >>](#)

- Airbyte
- Alibaba
- Amazon
- Apache Beam
- Apache Cassandra
- Apache Drill
- Apache Druid
- Apache HDFS
- Apache Hive
- Apache Kylin
- Apache Livy
- Apache Pig
- Apache Pinot
- Apache Spark
- Apache Sqoop
- ArangoDB
- Asana
- Celery
- IBM Cloudant
- Kubernetes
- Databricks
- Datadog
- DBT cloud
- Dingding
- Discord
- Docker
- Elasticsearch
- Exasol
- Facebook
- File Transfer Protocol (FTP)
- Github
- Google
- gRPC
- Hashicorp
- Hypertext Transfer Protocol (HTTP)
- Influx DB
- Internet Message Access Protocol (IMAP)
- Java Database Connectivity (JDBC)
- Jenkins
- Jira
- Microsoft Azure
- Microsoft PowerShell Remoting Protocol (PSRP)
- Microsoft SQL Server (MSSQL)
- Windows Remote Management (WinRM)
- MongoDB
- MySQL
- Neo4J
- ODBC
- OpenFaaS
- Opsgenie
- Oracle
- Pagerduty
- Papermill
- Plexus
- PostgreSQL
- Presto
- Qubole
- Redis
- Salesforce
- Samba
- Segment
- Sendgrid
- SFTP
- Singularity
- Slack
- Snowflake
- SQLite
- SSH
- Tableau
- Telegram
- Trino
- Vertica
- Yandex
- Zendesk



NUI Galway
OÉ Gaillimh



Universidad
Zaragoza



POITÉCNICA
Universität Bielefeld



GOETHE
UNIVERSITÄT
FRANKFURT AM MAIN



semalytix



OXFORD
UNIVERSITY PRESS



SEMANTIC WEB COMPANY
linking data to knowledge



Derilinx
DRIVE DECISION-MAKING. INSPIRE CHANGE



Airflow



Show simple airflow execution example

Airflow



- Dynamic: Airflow pipelines are configuration as code (Python), allowing for dynamic pipeline generation. This allows for writing code that instantiates pipelines dynamically.
- Extensible: Easily define your own operators, executors and extend the library so that it fits the level of abstraction that suits your environment.
- Elegant: Airflow pipelines are lean and explicit. Parameterizing your scripts is built into the core of Airflow using the powerful Jinja templating engine.
- Scalable: Airflow has a modular architecture and uses a message queue to orchestrate an arbitrary number of workers. Airflow is ready to scale to infinity.



NUI Galway
OÉ Gaillimh



Universidad
Zaragoza



Politécnica



GOETHE
UNIVERSITÄT
FRANKFURT AM MAIN



semalytix



OXFORD
UNIVERSITY PRESS



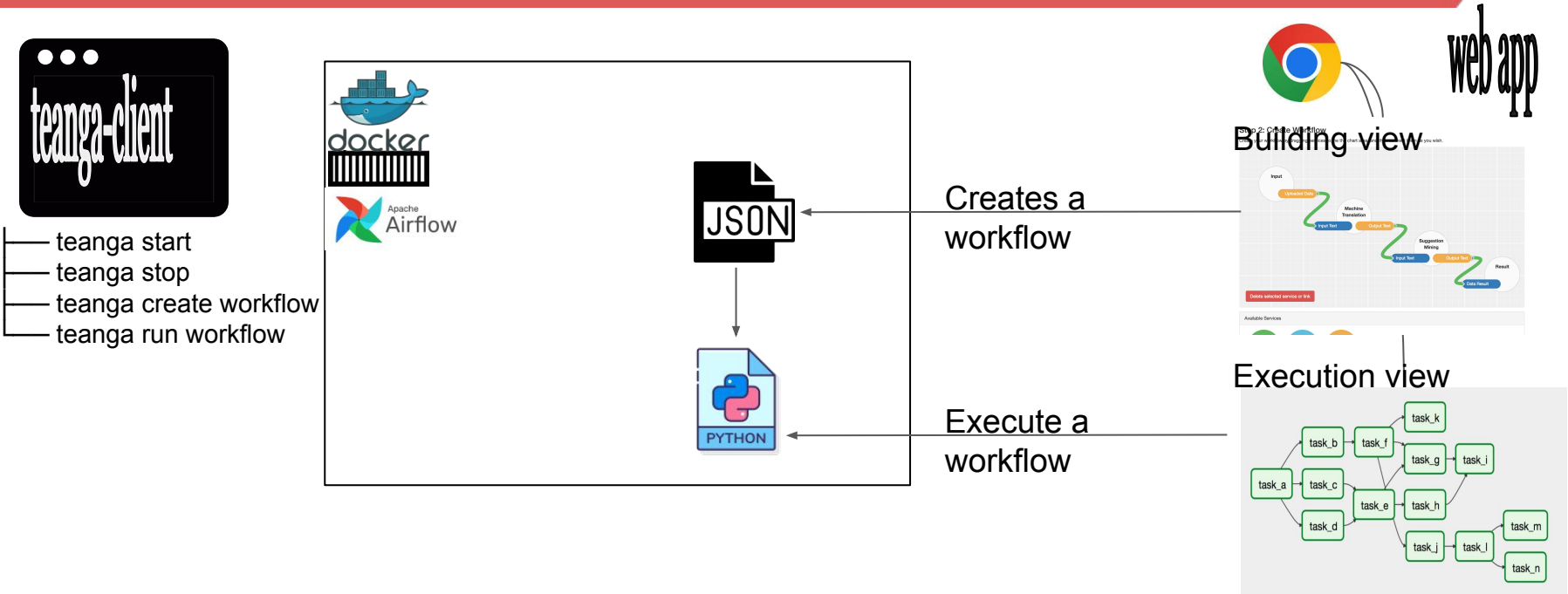
SEMANTIC WEB COMPANY
linking data to knowledge



Derilinx
DRIVE DECISION-MAKING. INSPIRE CHANGE



End users Usage



End users Usage

Show building view

End users Usage

The screenshot shows the Docker Hub search results for the term 'pretalod'. The interface includes a search bar at the top with 'pretalod' entered, and navigation links for 'Explore', 'Pricing', 'Sign In', and 'Register'. On the left, there are filters for 'Products' (Images, Plugins), 'Trusted Content' (Docker Official Image, Verified Publisher, Open Source Program), 'Operating Systems' (Linux, Windows), and 'Architectures' (ARM, ARM 64, IBM POWER, IBM Z, PowerPC 64 LE, x86, x86-64). The search results show 1 - 12 of 12 results for 'pretalod', sorted by 'Best Match'. The results list several Docker images:

- pretalod/link-otic**: By pretalod • Updated 10 months ago. "One Time Inverse Consultation" algorithm <https://hub.docker.com/r/pretalod/link-otic>. 20 Downloads, 0 Stars. Linux, x86-64.
- pretalod/lex-terminup**: By pretalod • Updated 10 months ago. TerminUp is a tool for terminology enrichment. <https://www.github.com/Pret-a-LLOD/terminup>. 18 Downloads, 0 Stars. Linux, x86-64.
- pretalod/link-cider-em**: By pretalod • Updated 10 months ago. Context and Inference baseD ontology alignER - EMbedding <https://github.com/Pret-a-LLOD/CIDER-EM-api>. 12 Downloads, 0 Stars. Linux, x86-64.
- pretalod/link-cycles**: By pretalod • Updated 10 months ago. It uses a cycles algorithm to infer translation pairs. <https://github.com/Pret-a-LLOD/Cycles-api>. 10 Downloads, 0 Stars. Linux, x86-64.
- pretalod/teanga-core**: By pretalod • Updated a day ago. 5 Downloads, 0 Stars. Linux, x86-64.
- pretalod/lex-cbl**: By pretalod • Updated 10 months ago. corpus-based lexicalization. <https://github.com/Pret-a-LLOD/ontology-lexicalization>. 6 Downloads, 0 Stars. Linux, x86-64.

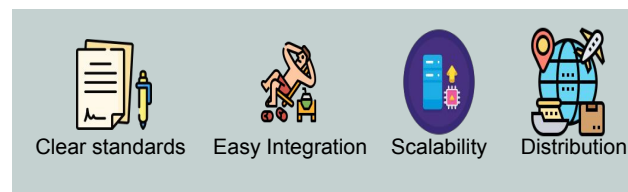
Overview: Who uses Teanga ?



End user

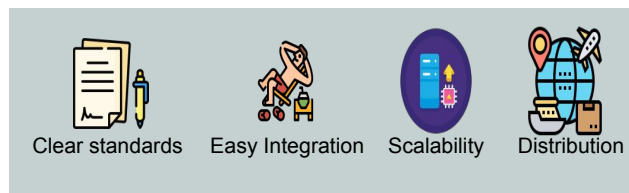


Service Developers



Teanga for service developers

Service Developers



Open API
Specification

Swagger



Teanga for service developers



```
class Notification(Resource):
    def get(self, id):
        self.abort_if_notification_not_found(id)
        return notification_manager.get_notification(id)

    def delete(self, id):
        self.abort_if_notification_not_found(id)
        notification_manager.delete_notification(id)
        return '', HttpStatus.no_content_204.value

    def patch(self, id):
        self.abort_if_notification_not_found(id)
        notification = notification_manager.get_notification(id)
        parser = reqparse.RequestParser()
        parser.add_argument('message', type=str)
        parser.add_argument('ttl', type=int)
        parser.add_argument('displayed_times', type=int)
        parser.add_argument('displayed_once', type=bool)
        args = parser.parse_args()
        print(args)

    def post(self):
        parser = reqparse.RequestParser()
        parser.add_argument('message', type=str, required=True,
            help='Message cannot be blank!')
        parser.add_argument('ttl', type=int, required=True,
            help='Time to live cannot be blank!')
        parser.add_argument('notification_category', type=str,
            required=True, help='Notification category cannot be blank!')
        args = parser.parse_args()
        notification = NotificationModel(
            message=args['message'],
            ttl=args['ttl'],
            creation_date=datetime.now(utc),
            notification_category=args['notification_category']
        )
        notification_manager.insert_notification(notification)
        return notification, HttpStatus.created_201.value
```



NUI Galway
OÉ Gaillimh



Universidad
Zaragoza



Universitat
Zaragoza



GOETHE
UNIVERSITÄT
FRANKFURT AM MAIN



semalytix

OXFORD
UNIVERSITY PRESS



SEMANTIC WEB COMPANY
linking data to knowledge



Derilinx
DRIVE DECISION-MAKING. INSPIRE CHANGE



Teanga for service developers



Open API
Specification



Swagger

- A document (or set of documents) that defines or describes an API
- allows both humans and computers to discover and understand the capabilities of the service without access to source code, documentation, or through network traffic inspection.
- Once an application is well described in OpenAPI it can benefit from automation tools. (i.e. swagger UI, show pets example)



NUI Galway
OÉ Gaillimh



Universidad
Zaragoza



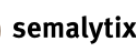
Politécnica
Universität Bielefeld



GOETHE
UNIVERSITÄT
FRANKFURT AM MAIN



DFK



semalytix



OXFORD
UNIVERSITY PRESS



SEMANTIC WEB COMPANY
linking data to knowledge



Derilinx
DRIVE DECISION-MAKING. INSPIRE CHANGE



Teanga for service developers



```
/naisc/{config}/block:
  get:
    summary: Find a blocking between two datasets
    operationId: block
    parameters:
      - name: left
        in: query
        description: The ID of the left dataset to block as uploaded to upload
        required: true
        schema:
          type: string
      - name: right
        in: query
        description: The ID of the right dataset to block as uploaded to upload
        required: true
        schema:
          type: string
      - name: config
        in: path
        description: The configuration to be used for matching
        required: true
        schema:
          type: string
    responses:
      '200':
        description: The blocking succeeded
        content:
          application/json:
            schema:
              type: array
              items:
                $ref: "#/components/schemas/Blocking"
```



NUI Galway
OÉ Gaillimh



Universidad
Zaragoza



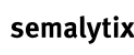
POITÉCNICA
Universität Bielefeld



GOETHE
UNIVERSITÄT
FRANKFURT AM MAIN



DFK



semalytix



OXFORD
UNIVERSITY PRESS



SEMANTIC WEB COMPANY
linking data to knowledge



Derilinx
DRIVE DECISION-MAKING. INSPIRE CHANGE



European
Commission

Teanga for service developers



- 1) Importance when developing services as rest apis.
Bridging the description-implementation gap

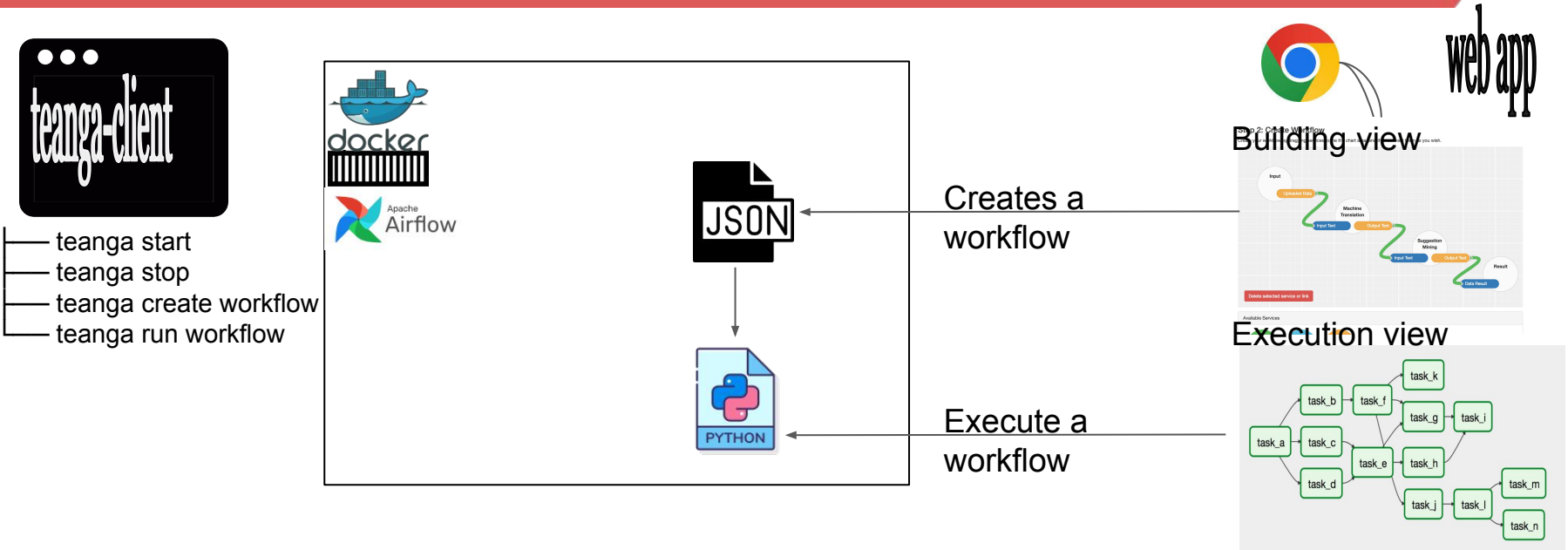
```
/naisc/{config}/block:
  get:
    summary: Find a blocking between two datasets
    operationId: block
    parameters:
      - name: left
        in: query
        description: The ID of the left dataset to block as uploaded to upload
        required: true
        schema:
          type: string
      - name: right
        in: query
        description: The ID of the right dataset to block as uploaded to upload
        required: true
        schema:
          type: string
      - name: config
        in: path
        description: The configuration to be used for matching
        required: true
        schema:
          type: string
    responses:
      '200':
        description: The blocking succeeded
        content:
          application/json:
            schema:
              type: array
              items:
                $ref: "#/components/schemas/Blocking"
```

```
@webserver.route("/naisc/{config}/block", methods=["GET"])
def block(config):
    """
    print(request.args)
    print(request.json)
    print(request.data)
    if request.data: print(request.data)
    else: raise Exception("request body is empty")
    blockings = []
    for line in json.loads(request.data):
        blockings.append(calculate_blocking(line))
    return jsonify(blockings)
```

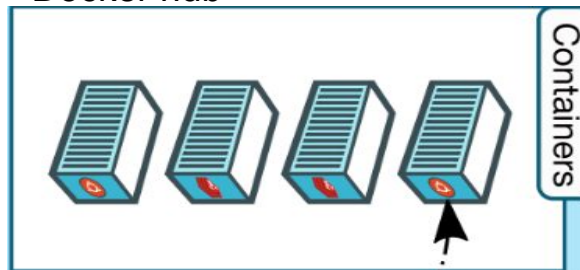
Teanga for service developers

- Developers can follow examples of already existing services
- They can test it locally
- Ideally with more adoption, there will be better standards defined

Overview



Docker hub



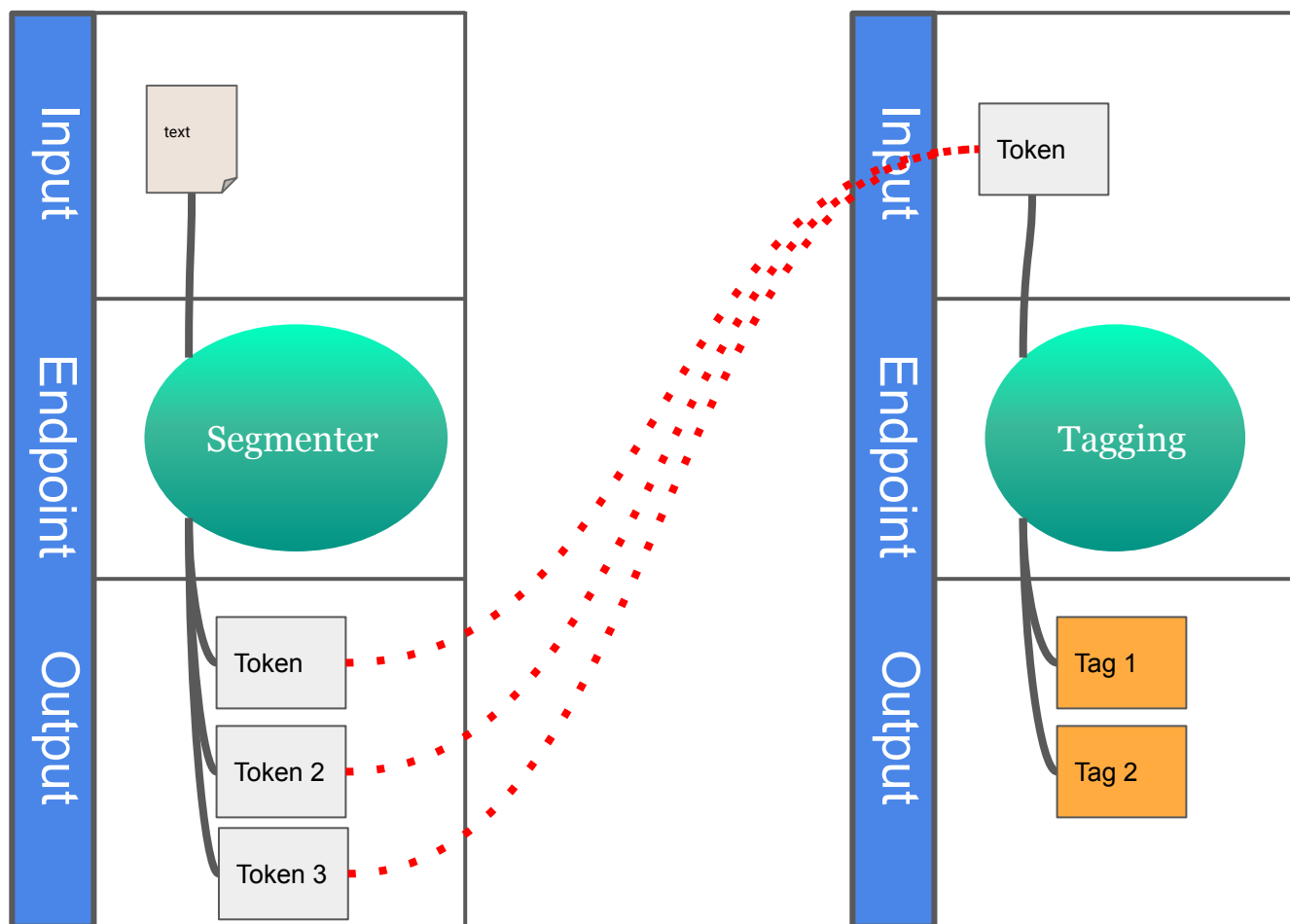
Workflows description and linking

```
{
  "1": {
    "input": {
      "EmptyCas": { "documentText": "this is a test", "language": "en" }
    },
    "operation_id": "segmenter",
    "repo": "berstearns",
    "image_id": "teanga-dkpro-wrapper",
    "image_tag": "032021",
    "host_port": 8001,
    "container_port": 8080,
    "dependencies": []
  },
  "2": {
    "input": {
    },
    "operation_id": "tagger",
    "repo": "berstearns",
    "image_id": "teanga-dkpro-wrapper",
    "image_tag": "032021",
    "host_port": 8001,
    "container_port": 8080,
    "dependencies": [ { "operator": "pass", "steps": [ "1" ] } ]
  }
}
```

Workflows description and linking

Show workflow instantiation

Case Study: Naisc



Matching Scenario Diagram

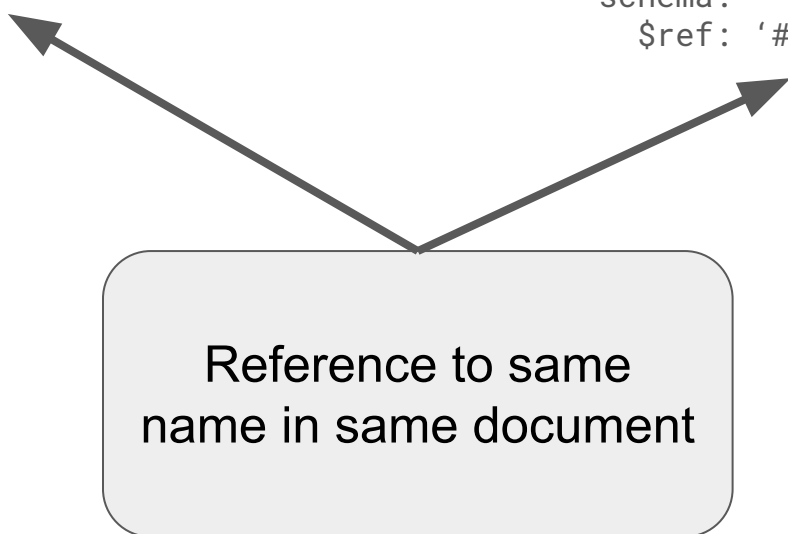
Service required input

Service output		Single Value	Array of values	Single schema	Array of schemas
	Single value	Type matching			
	Array of values	Item matching	Item matching		
	Single schema			Schema matching	
	Array of schemas			Schema matching	Schema matching

Level 0: Schema Name

```
requestBody:  
  content:  
    application/json:  
      schema:  
        $ref: '#/name'
```

```
responses:  
  '200':  
    description: Success  
    content:  
      application/json:  
        schema:  
          $ref: '#/name'
```



Reference to same
name in same document

Level 1: Schema Match

```
name:  
  type: object  
properties:  
  partOfSpeech:  
    type: string
```

```
differentName:  
  type: object  
properties:  
  partOfSpeech:  
    type: string
```

Schema contain same
property set

Level 1.5: Schema Match (inexact)

```
name:
  type: object
properties:
  partOfSpeech:
    type: string
  someOtherProperty:
    type: string
```

```
differentName:
  type: object
properties:
  partOfSpeech:
    type: string
```

Extra properties can be
dropped from output of
one service to input of
next

Level 2: RDF Matching

```
name:  
  type: object  
properties:  
  partOfSpeech:  
    type: string
```

```
differentName:  
  type: object  
properties:  
  pos:  
    type: string
```

```
{  
  "partOfSpeech":  
  "http://www.lexinfo.net/ontology/3.0/partOfSpeech"  
}
```

```
{  
  "pos":  
  "http://www.lexinfo.net/ontology/3.0/partOfSpeech"  
}
```

Services can be
matched because
JSON-LD context uses
same property

Level 3: Linking

```
name:
  type: object
  properties:
    partOfSpeech:
      type: string
```

```
differentName:
  type: object
  properties:
    pos:
      type: string
```

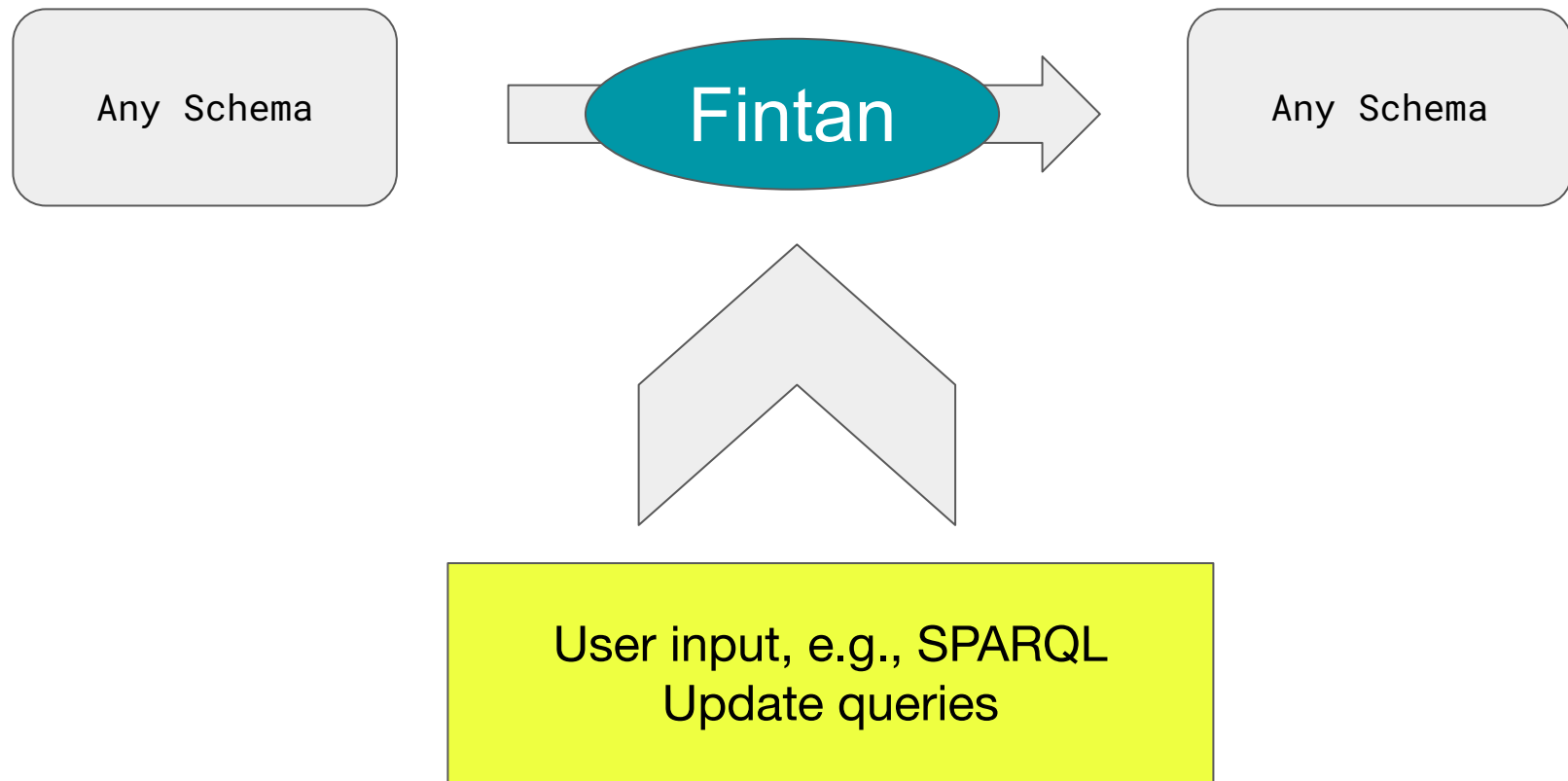
```
{
  "partOfSpeech":
    "http://www.lexinfo.net/ontology/3.0/partOfSpeech"
}
```

```
{
  "pos":
    "http://www.example.com/ontology/partOfSpeech"
}
```

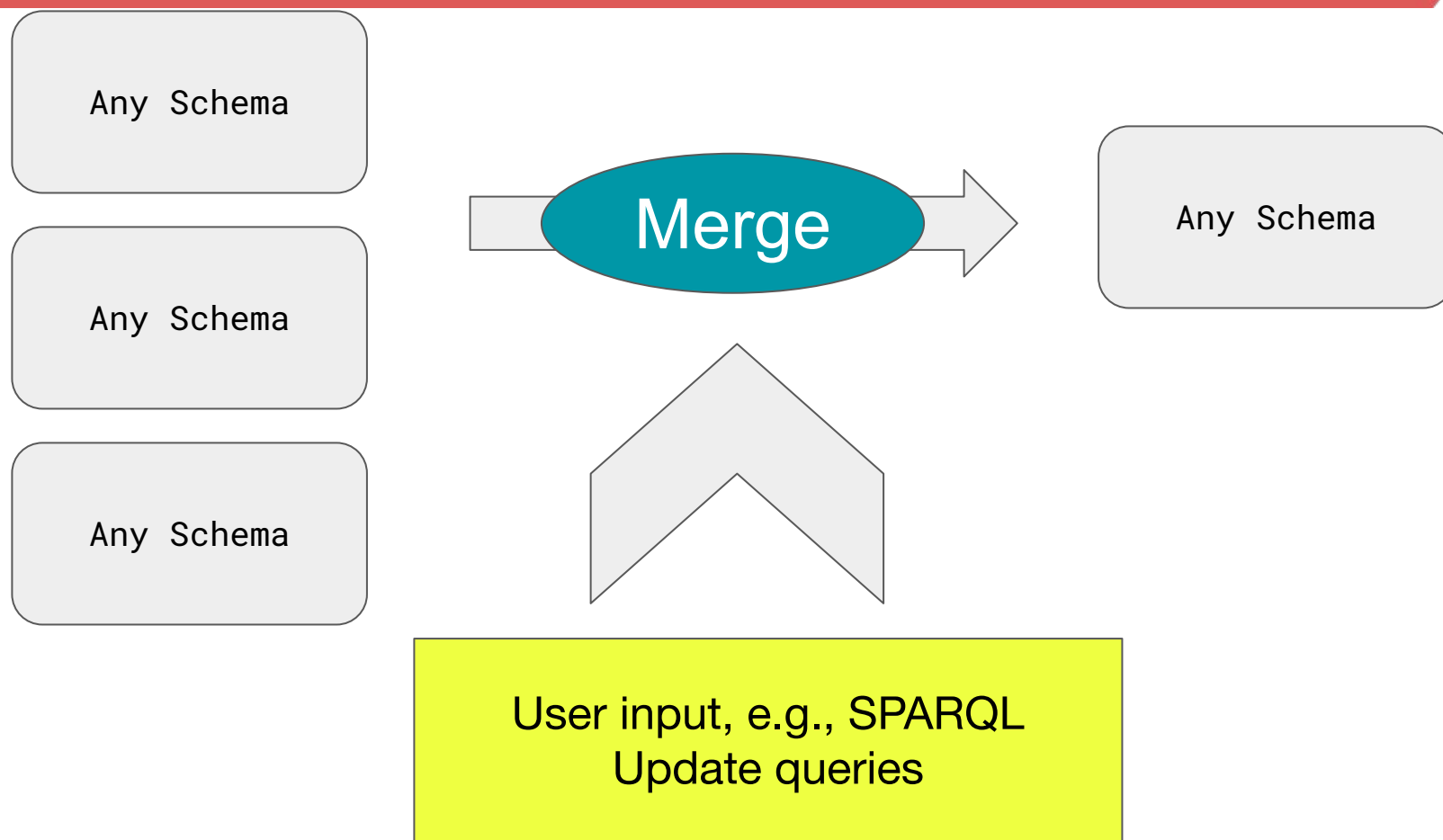
Naisc infers that the
properties are equivalent

<<http://www.lexinfo.net/ontology/3.0/partOfSpeech>> skos:exactMatch <<http://www.example.com/ontology/partOfSpeech>>

Level 4: User-driven transformation



Level 4: User-driven transformation

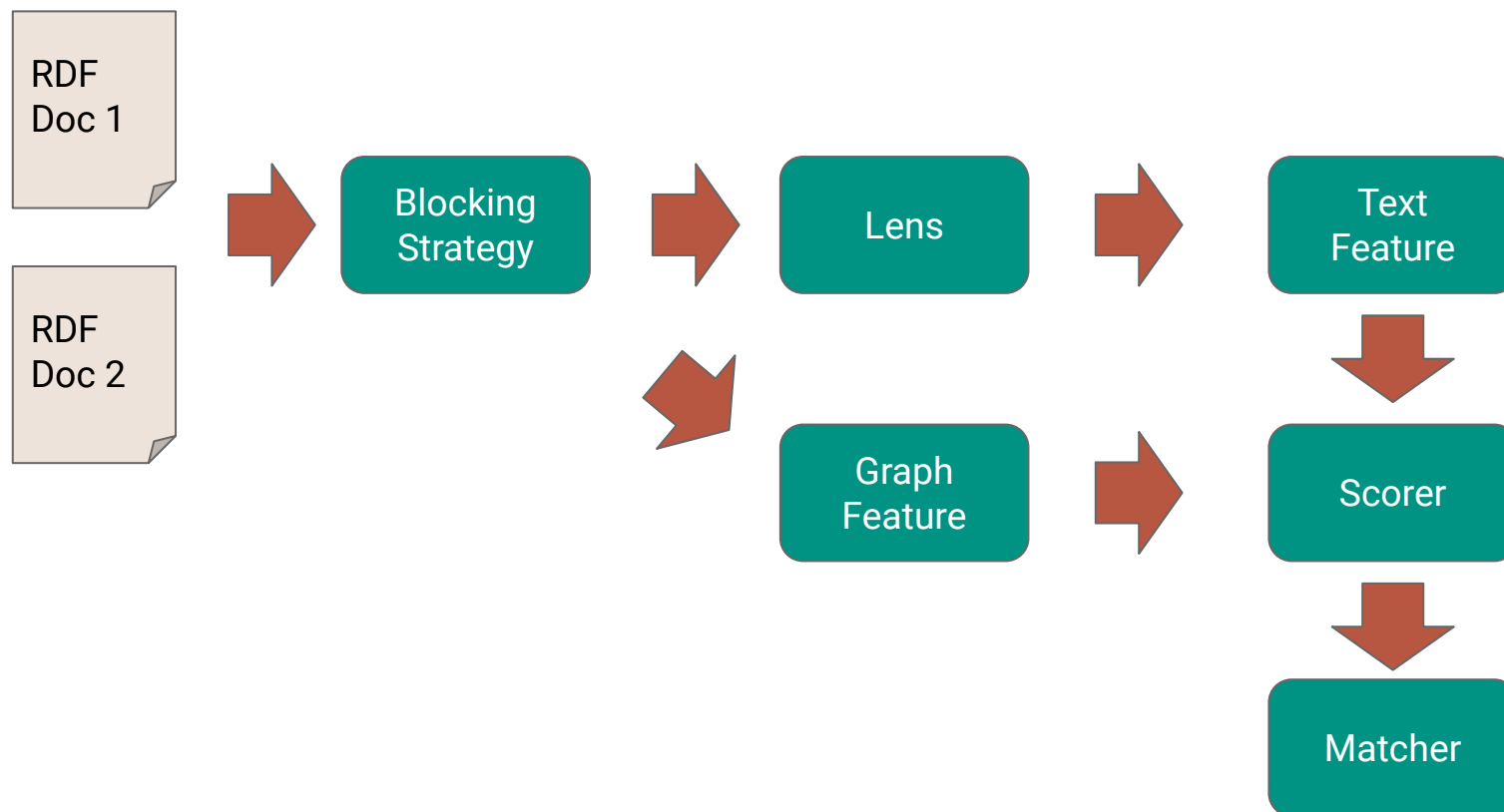


Workflows description and linking

Through workflow descriptions we can define matching operators in the workflow runtime

We can check service compatibility on building time in the UI

Case Study: Naisc



Architecture: Technologies



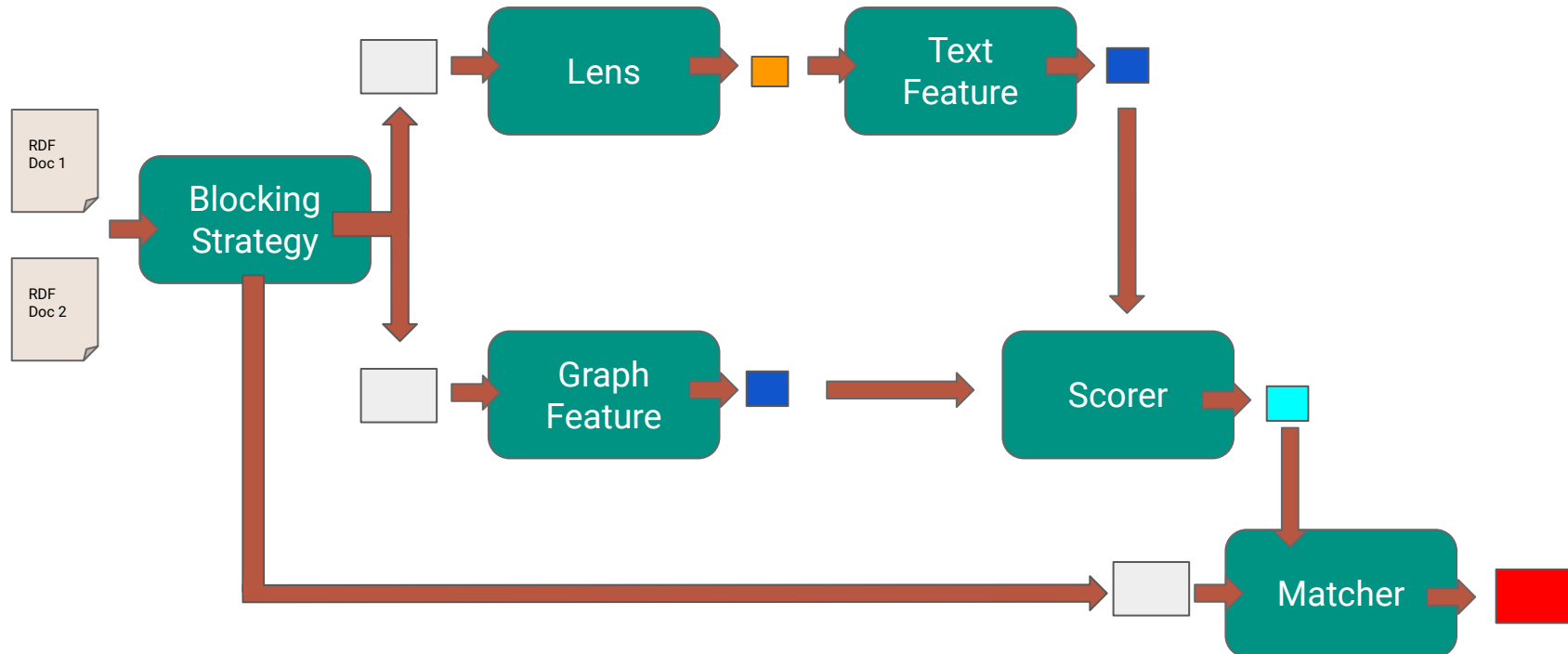
- 2) Importance when matching services.
Infer relationships between services

```
/naisc/{config}/block:
get:
  summary: Find a blocking between two datasets
  operationId: block
  parameters:
    - name: left
      in: query
      description: The ID of the left dataset to block as uploaded to upload
      required: true
      schema:
        type: string
    - name: right
      in: query
      description: The ID of the right dataset to block as uploaded to upload
      required: true
      schema:
        type: string
    - name: config
      in: path
      description: The configuration to be used for matching
      required: true
      schema:
        type: string
  responses:
    '200':
      description: The blocking succeeded
      content:
        application/json:
          schema:
            type: array
            items:
              $ref: "#/components/schemas/Blocking"
```

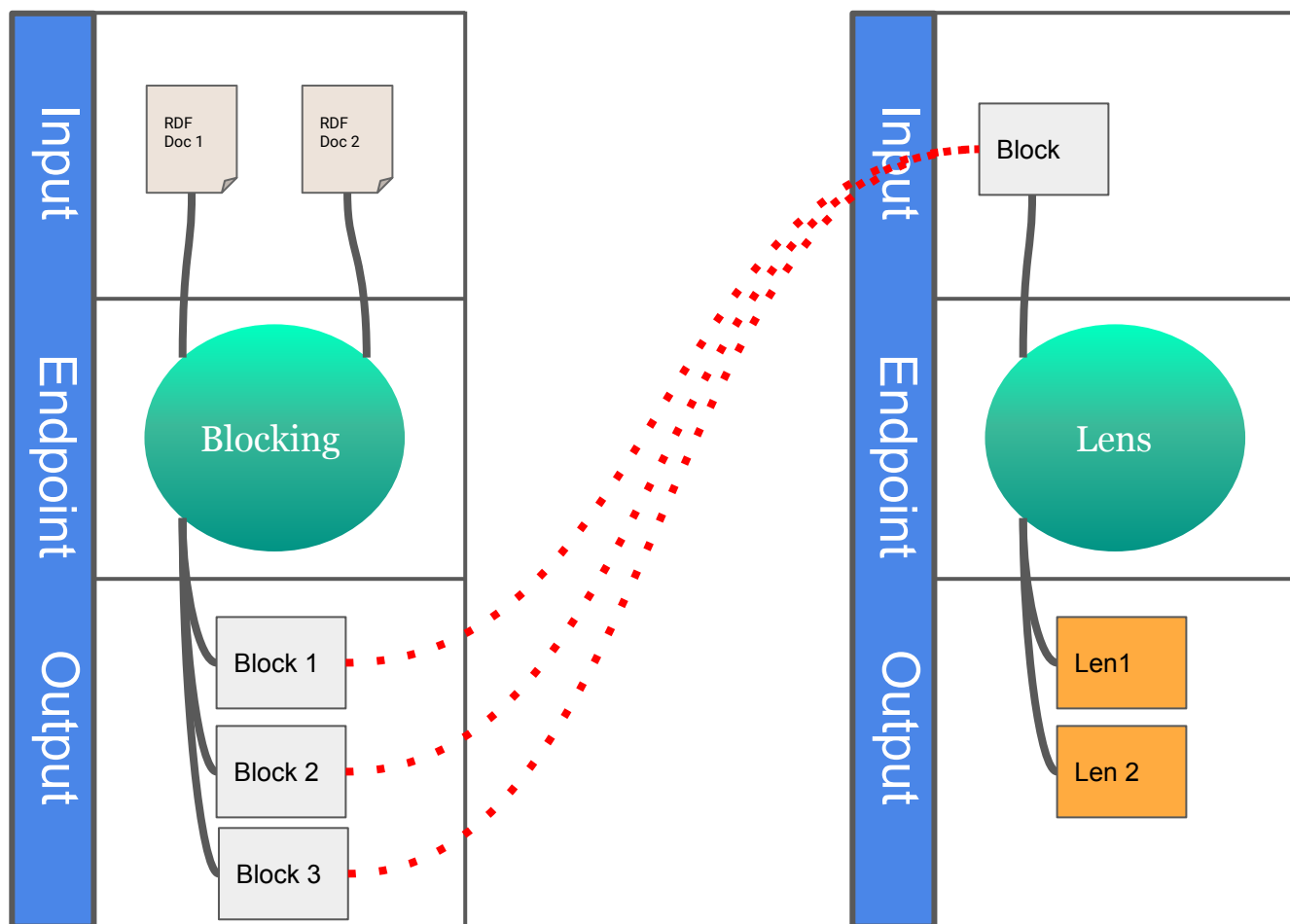


```
/naisc/{config}/extract_text:
post:
  summary: Extract text with a lens
  operationId: extract_text
  requestBody:
    content:
      application/json:
        schema:
          $ref: '#/components/schemas/Blocking'
  parameters:
    - name: config
      in: path
      description: The configuration to be used for matching
      required: true
      schema:
        type: string
  responses:
    '200':
      description: Success
      content:
        application/json:
          schema:
            type: array
            items:
              $ref: "#/components/schemas/LangStringPair"
```

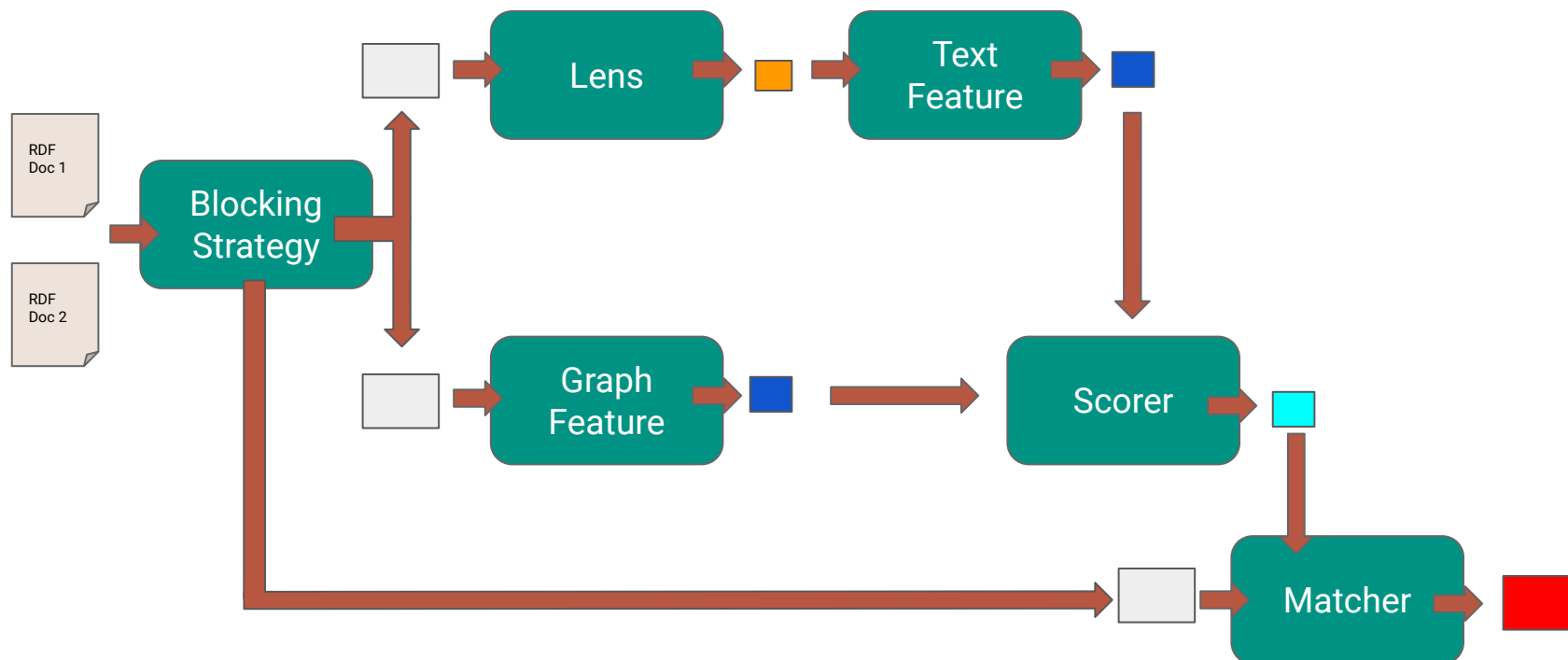
Case Study: Naisc



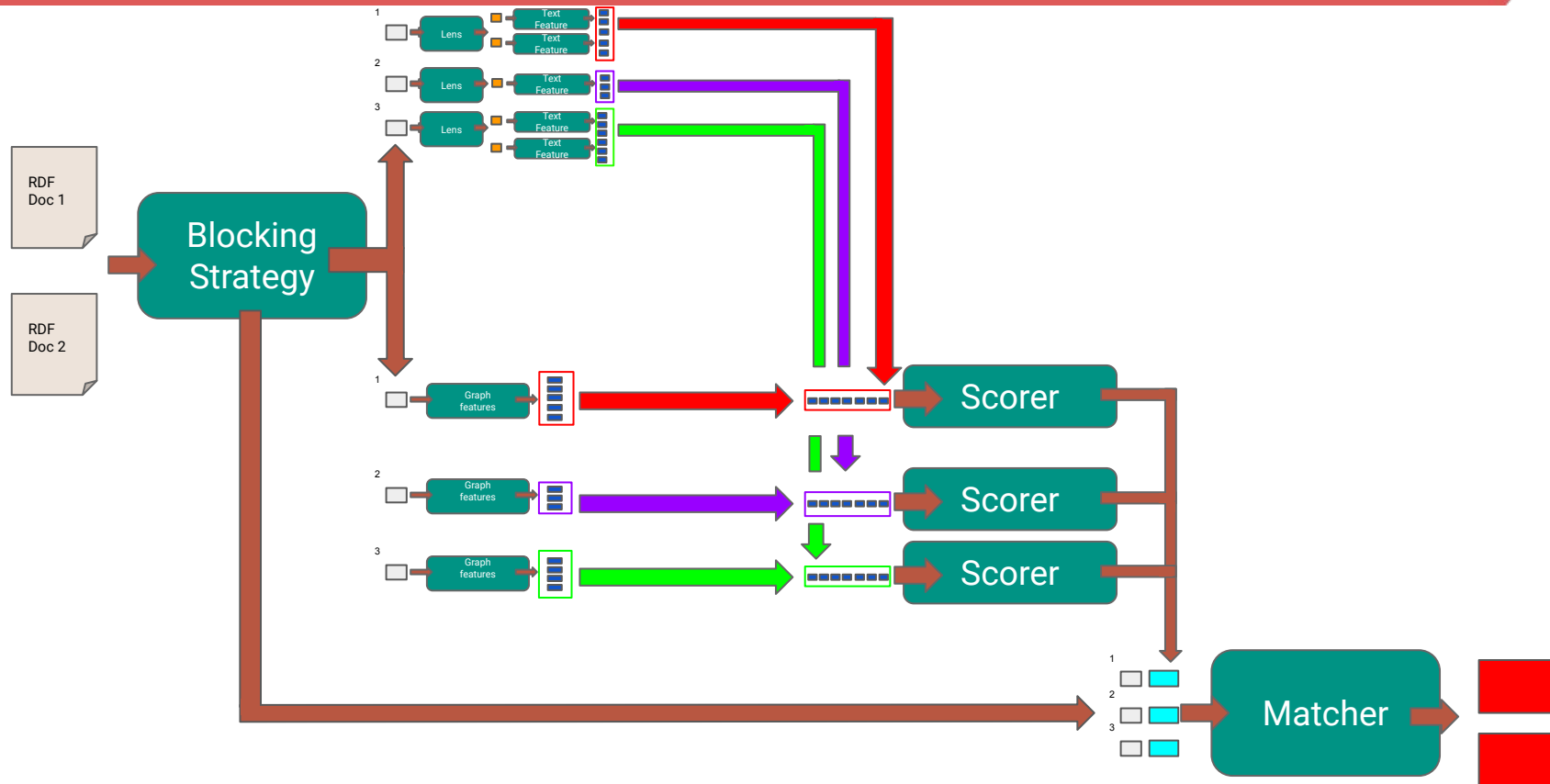
Case Study: Naisc



Case Study: Naisc



Case Study: Naisc



Frontend

The screenshot displays a web browser window with the address bar showing `localhost:8080/admin/workflow/build`. The page title is "React App" and the browser's address bar shows the URL. The application header features the "Teanga MVP" logo. The main heading is "Build a workflow from the Flow Diagram", accompanied by a "Create Workflow" button. Below the heading is a search bar labeled "Search :". Under the search bar, three components are listed: "dummy teanga", "dkpro", and "naisc". Each component has an "Add to Workflow" button. The "naisc" component is highlighted with a red border. Below the list is a grid area containing a flow diagram. The diagram shows two circular nodes labeled "Naisc" connected by a green arrow labeled "Data". The first node is connected to an orange rectangle, and the second node is connected to a blue rectangle.

Teanga Services

- Most of recently added services were instances of DKPRO
 - 44 DKPRO services + 6 others

Summary

- After setting up a reliable architecture that is easy to use we open room for external people integrating and test their services.
- As the number of services increase there lot of potential to analyse match and compatibility of services through services descriptions
- A visual workflow creation is important but a difficult part of the process is discovering new services and exploring new possibilities of workflow.