eSAAM 2023

on Cloud to Edge Continuum

# Tool Support for Architectural Pattern Selection and Application in Cloud-Centric Service-Oriented IDEs

Fulya Horozal, Philip Reimer, Sebastian Scholze
ATB – Institut für angewandte Systemtechnik Bremen, Germany
horozal@atb-bremen.de

Oct. 17, 2023

Ludwigsburg, Germany

## Software Architecture Design

- **High-level structure of system components & their interactions**
- **High impact on quality, success & management of software**
- **Architectural patterns & styles**
  - Principles & best practices for software architecture design
  - Guidelines & templates for structuring & organizing software systems
  - Common vocabulary to describe software architecture
  - E.g., event-driven architecture, layered architecture, microservices

## Software Architecture Design

- **Choosing the right architectural pattern**
  - Strengths, drawbacks, technical knowledge
  - Impact on quality attributes (“-abilities”), requirements, constraints
  - Project requirements, constraints & limitations
  - System complexity, scalability needs
  - Team expertise, trade-offs
  - Industry best practices
- **Traditional methods**
  - Architectural pattern catalogs, architectural decision records
  - Architecture tradeoff analysis, decision matrices
  - Expert consultation, reference architectures

# Architectural Decision Tool Support

- **Modeling and diagramming**
  - UML-based software modeling
  - Architectural diagrams
  - E.g., Enterprise Architect (Sparx Systems), IBM Rational Software Architect, Archimate Toolset, Eclipse Papyrus
- **Architectural decision modeling framework (O. Zimmermann)**
- **Quality attribute analysis**
  - Performance & security analysis
  - Scalability & maintainability assessment
  - Cost & change impact analysis
  - E.g., ARIS (Software AG), IBM Rational Rhapsody, QualiWare, Determine
- **ML techniques to learn from architectural decisions (Mahabaleshwar)**
- **Decision studio web tool for technology selection & architectural patterns (Farshidi et al.)**
- **Code generation from architectural models**
  - From UML or other notations
  - Scaffolding & project organization tools

# A Framework for Architectural Pattern Selection and Application

- **Architectural pattern decision support feature for IDE integration**
- **Architectural pattern selection**
  - Knowledge base
    - Application domain
    - Application type
    - Quality attributes / non-functional requirements
    - Development & deployment requirements
    - Architectural features
  - Evaluation & ranking
    - Based on existing literature on pattern analyses (Farshidi et al. 2020, Richards 2022)
    - Scoring system assigning weights to patterns in context of knowledge base
- **Architectural pattern application**
  - GitHub repository templates for project & code organization
- **Integrated into the cloud-native SmartCLIDE IDE**

# Supported Architectural Patterns

- **Layered architecture**
  - Distinct layers for presentation, application logic, data storage
- **Event-driven architecture (EDA)**
  - Systems communicate through events (trigger actions or reactions)
- **Microkernel architecture**
  - Essential core (the microkernel) and various optional modules
- **Microservices architecture**
  - Small independent services that communicate over APIs
- **Service-oriented architecture (SOA)**
  - Loosely coupled, reusable services communicating via interfaces
- **Space-based architecture (SBA)**
  - Distributes data & processing across a network of interconnected, distributed spaces

# Application Domain

| Application Domain | Associated Architectural Patterns |
|---|---|
| Web-based systems | EDA, layered, microservices, SOA, SBA |
| Web services | Microservices, SOA, SBA |
| Service-based systems | Microservices, SOA |
| Distributed systems | EDA, layered, microkernel, microservices, SOA, SBA |
| Cloud computing applications | Microservices, SOA |
| Mobile applications | Layered, microservices, SOA, SBA |
| Compiler design | Layered |
| CASE and related developer tools | EDA, layered, microkernel, microservices, |
| Database systems | EDA, layered, microservices |
| Context-aware systems | EDA, layered, microservices, SOA |
| Adaptable systems | Microkernel, microservices |
| Enterprise application integration | EDA, microservices, SOA |
| Customer relationship management | EDA, layered, microservices, SOA |
| Information management and decision support system | EDA, layered, SOA |
| Transaction processing | EDA, layered, microservices, SOA |

# Application Type

| Application Type | Associated Architectural Patterns |
|---|---|
| Web application / website with small components | Microservices, SOA |
| Large scale web application like e-commerce or social website development | EDA, layered, microservices, SOA, SBA |
| General desktop application | Layered |
| Application with a simple business logic that does not need to scale out | EDA, layered |
| Enterprise or business application with traditional IT departments and processes | Layered, SOA |
| Application with fixed set of core functionalities and a dynamic set of functionalities that need frequent updates | Microkernel, microservices |
| Large, complex, enterprise-wide systems that require integration with many heterogeneous applications | EDA, microservices, SOA |
| Application with many shared components, particularly components across the enterprise | EDA, microservices, SOA |
| Application with immense and rapidly growing data systems | EDA, microservices, SBA |
| Application with different platforms | Microservices, SOA |
| Application that requires strict standards of testability | Layered |

# Quality Attributes / NFRs

| Quality Attributes / Non-functional Requirements | Associated Architectural Patterns |
|---|---|
| Maintainability | All six |
| Performance / Efficiency | EDA, microservices, SOA, SBA |
| Portability | All six |
| Reliability | All six |
| Security | All six |

# Architectural Knowledge

| Development & Deployment Requirements | Associated Architectural Patterns |
|---|---|
| High ease of development / quick development with fewer developers | Layered, microservices |
| Ease of rewriting and updating parts of the application | EDA, microkernel, microservices, SOA |
| Development teams that are spread out | Microservices |
| Adding special functionality, modules or extensions without modifying the original application | Microkernel, microservices |
| High ease of deployment | Microkernel, microservices |
| Rapid, frequent and independent deployment | Microservices |
| Quick response to a constantly changing environment | EDA, microkernel, microservices, SBA |
| Reusability of integrations and components sharing | EDA, microservices, SOA |

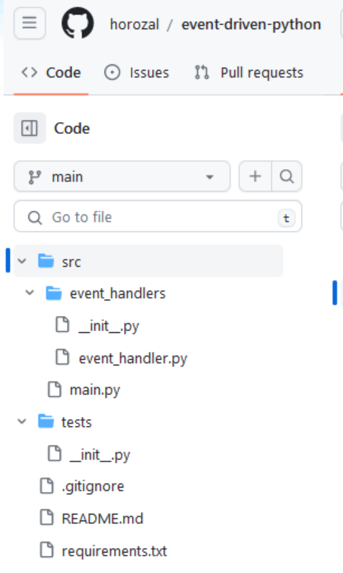
# Architectural Features

| Architectural Features | Associated Architectural Patterns |
|---|---|
| Asynchronous communication / data flow | EDA, layered, microservices, SBA |
| Synchronous communication / data flow | Layered, microkernel, microservices, SOA |
| Loose coupling | EDA, microservices, SOA |
| Independent services | Microservices |
| Separation of concerns | Layered, microkernel, microservices, SOA |
| Plug-in components | Microkernel |
| Dynamic composition | EDA, microkernel, SOA, SBA |
| High volume data | EDA, microservices, SBA |

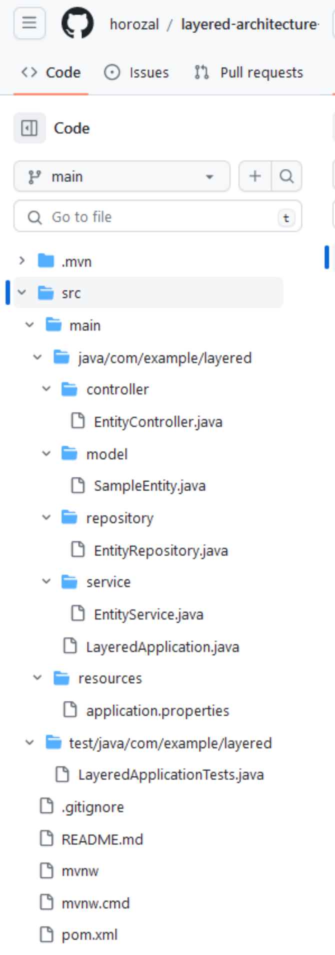
# Architectural Pattern Application

eSAAM 2023
on Cloud to Edge Continuum

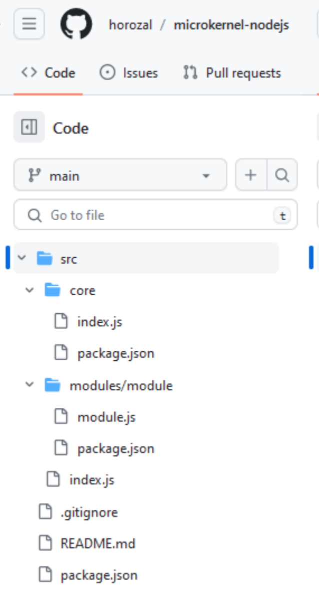
- **18 GitHub repository templates**
  - Frameworks: Java Spring Node.js Python
  - Template for each architectural pattern & framework



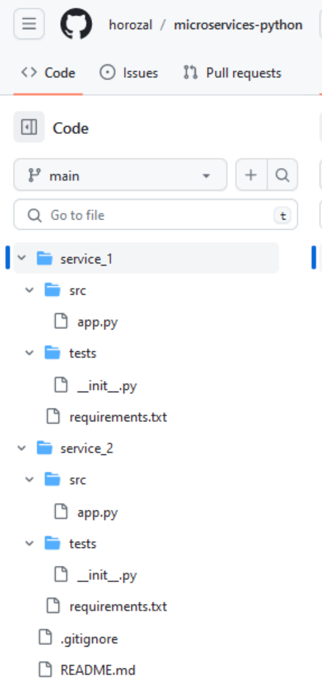
```
horozal / event-driven-python
Code  Issues  Pull requests
Code
main
Go to file
src
  event_handlers
    __init__.py
    event_handler.py
  main.py
tests
  __init__.py
.gitignore
README.md
requirements.txt
```



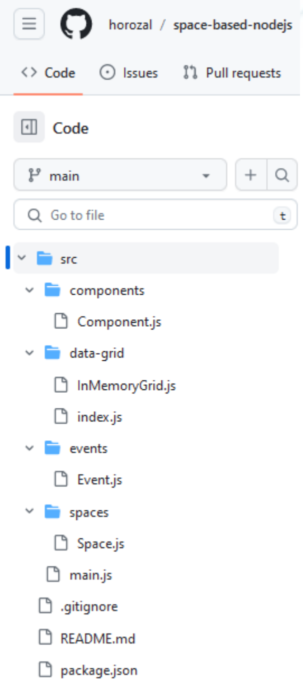
```
horozal / layered-architecture
Code  Issues  Pull requests
Code
main
Go to file
.mvn
src
  main
    java/com/example/layered
      controller
        EntityController.java
      model
        SampleEntity.java
      repository
        EntityRepository.java
      service
        EntityService.java
      LayeredApplication.java
    resources
      application.properties
  test/java/com/example/layered
    LayeredApplicationTests.java
.gitignore
README.md
mvnw
mvnw.cmd
pom.xml
```



```
horozal / microkernel-nodejs
Code  Issues  Pull requests
Code
main
Go to file
src
  core
    index.js
    package.json
  modules/module
    module.js
    package.json
  index.js
.gitignore
README.md
package.json
```

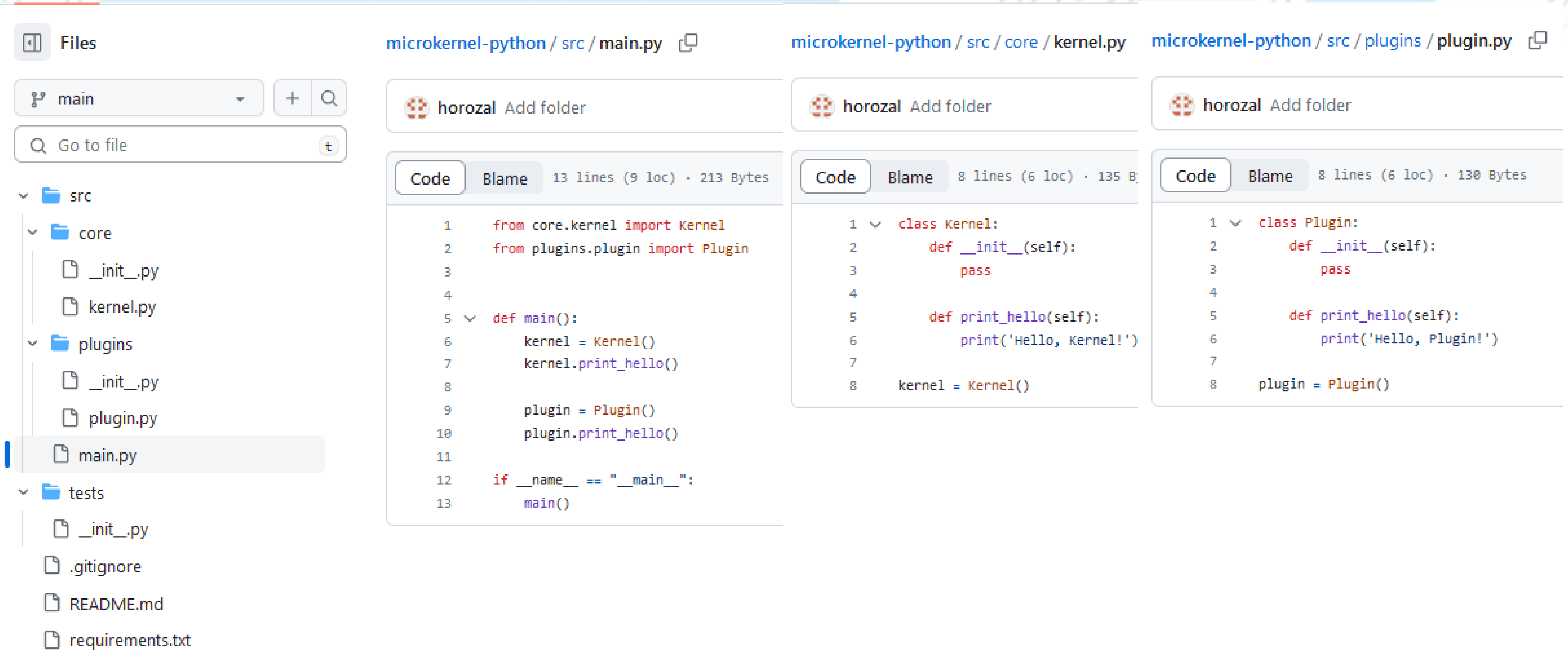


```
horozal / microservices-python
Code  Issues  Pull requests
Code
main
Go to file
service_1
  src
    app.py
  tests
    __init__.py
  requirements.txt
service_2
  src
    app.py
  tests
    __init__.py
  requirements.txt
.gitignore
README.md
```



```
horozal / space-based-nodejs
Code  Issues  Pull requests
Code
main
Go to file
src
  components
    Component.js
  data-grid
    InMemoryGrid.js
    index.js
  events
    Event.js
  spaces
    Space.js
  main.js
.gitignore
README.md
package.json
```

# Repository Templates



microkernel-python / src / main.py

horozal Add folder

Code Blame 13 lines (9 loc) · 213 Bytes

```python
from core.kernel import Kernel
from plugins.plugin import Plugin


def main():
    kernel = Kernel()
    kernel.print_hello()

    plugin = Plugin()
    plugin.print_hello()

if __name__ == "__main__":
    main()
```

microkernel-python / src / core / kernel.py

horozal Add folder

Code Blame 8 lines (6 loc) · 135 By

```python
class Kernel:
    def __init__(self):
        pass

    def print_hello(self):
        print('Hello, Kernel!')

kernel = Kernel()
```

microkernel-python / src / plugins / plugin.py

horozal Add folder

Code Blame 8 lines (6 loc) · 130 Bytes

```python
class Plugin:
    def __init__(self):
        pass

    def print_hello(self):
        print('Hello, Plugin!')

plugin = Plugin()
```

# Implementation

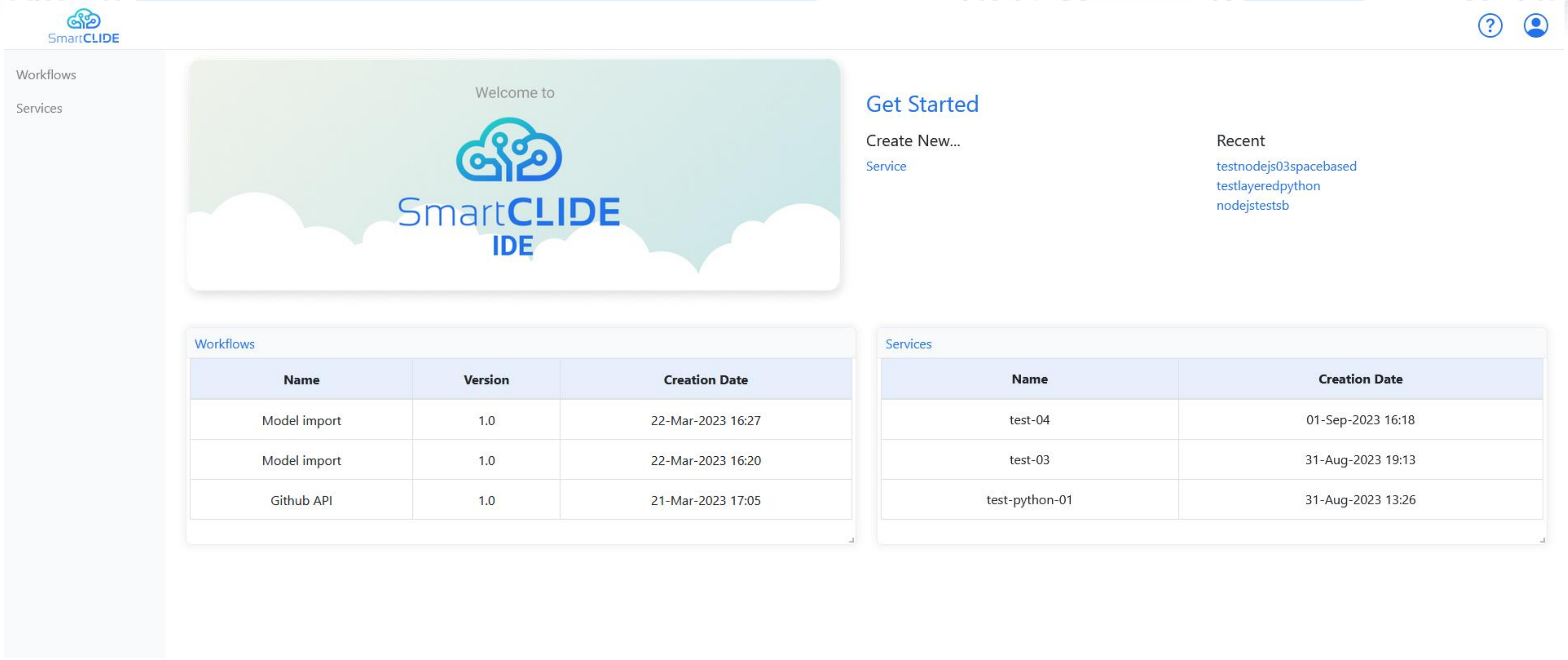
- **Backend REST API in Java Spring**
  - Retrieve survey content
  - Select architectural pattern
  - Select repository template
- **Independent of survey content & evaluation values**
  - JSON format for survey content & evaluation values
  - Reconfigurable

# SmartCLIDE Project

- **H2020 EU-funded project (2020-2023)**
  - https://smartclide.eu/
- **Novel cloud-native IDE**
  - https://ide.che.smartclide.eu/
  - Based on Eclipse Theia
  - Life cycle support (development, testing, deployment, run-time)
  - Collaborative discovery, creation, composition, testing, deployment of services in the cloud
  - Source code monitoring
  - CI/CD integration
- **4 industry pilots for validation & assessment**
  - Real-time communication platform (Wellness Telecom, Spain)
  - Social security application (Netcompany-Intrasoft, Luxembourg)
  - IoT web catalog (Unparallel, Portugal)
  - Project management solution (CONTACT Software, Germany)
- **Open sourced under Eclipse Foundation**
  - Eclipse OpenSmartCLIDE

SmartCLIDE

# SmartCLIDE IDE



Get Started

Create New...

Service

Recent

testnodejs03spacebased

testlayeredpython

nodejstestsb

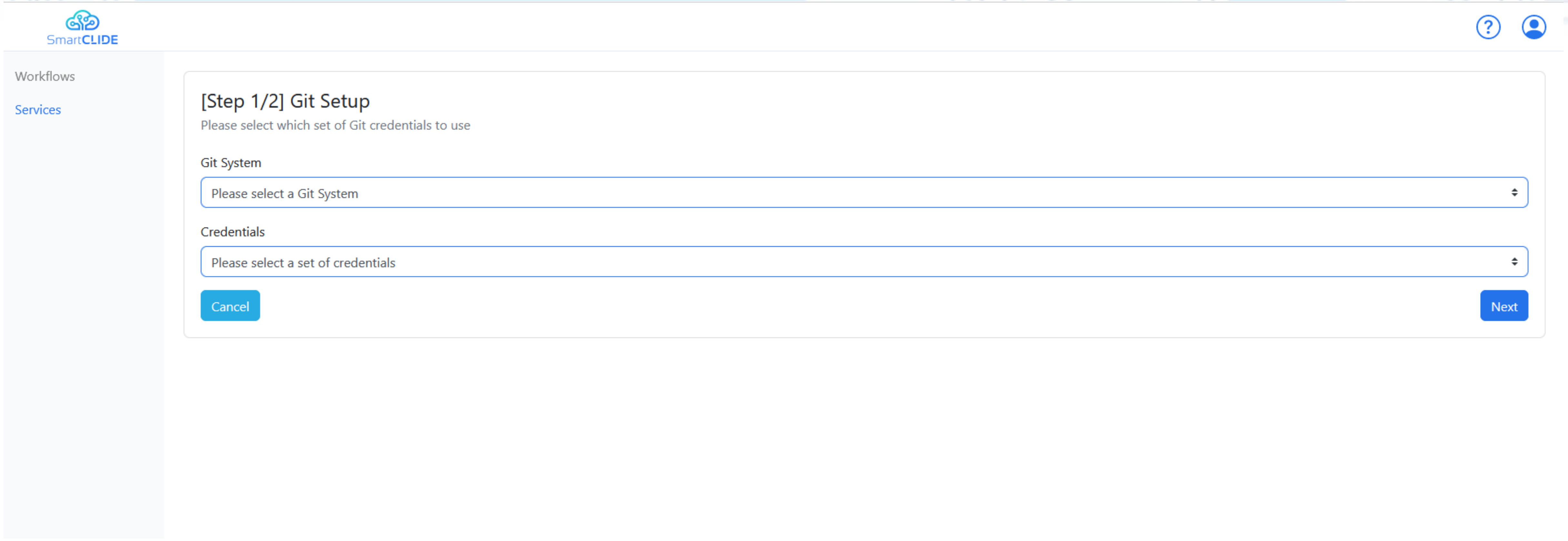
Workflows

| Name | Version | Creation Date |
|---|---|---|
| Model import | 1.0 | 22-Mar-2023 16:27 |
| Model import | 1.0 | 22-Mar-2023 16:20 |
| Github API | 1.0 | 21-Mar-2023 17:05 |

Services

| Name | Creation Date |
|---|---|
| test-04 | 01-Sep-2023 16:18 |
| test-03 | 31-Aug-2023 19:13 |
| test-python-01 | 31-Aug-2023 13:26 |

# SmartCLIDE IDE



[Step 1/2] Git Setup

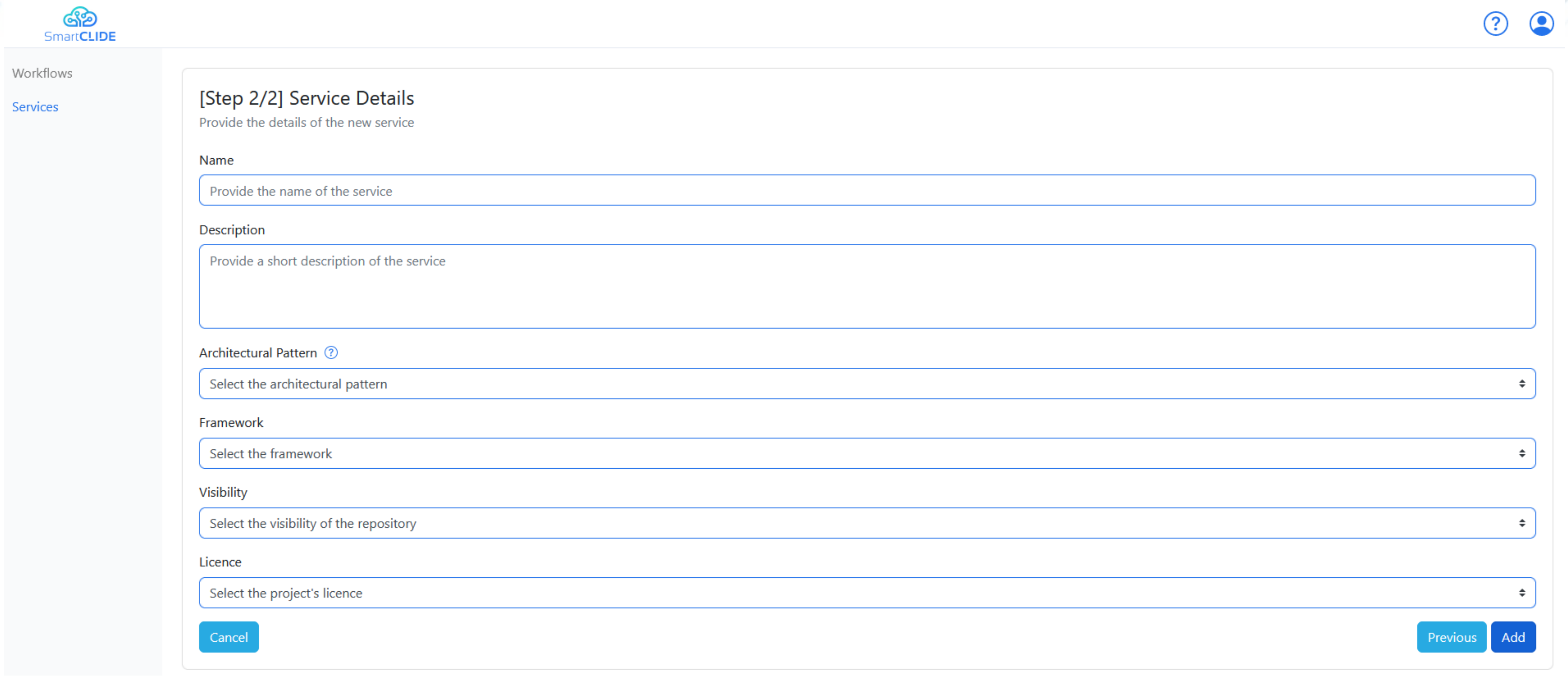
Please select which set of Git credentials to use

Git System: Please select a Git System

Credentials: Please select a set of credentials

# SmartCLIDE IDE

eSAAM 2023
on Cloud to Edge Continuum



SmartCLIDE

Workflows

Services

[Step 2/2] Service Details

Provide the details of the new service

Name

Provide the name of the service

Description

Provide a short description of the service

Architectural Pattern

Select the architectural pattern

Framework

Select the framework

Visibility

Select the visibility of the repository

Licence

Select the project's licence

Cancel

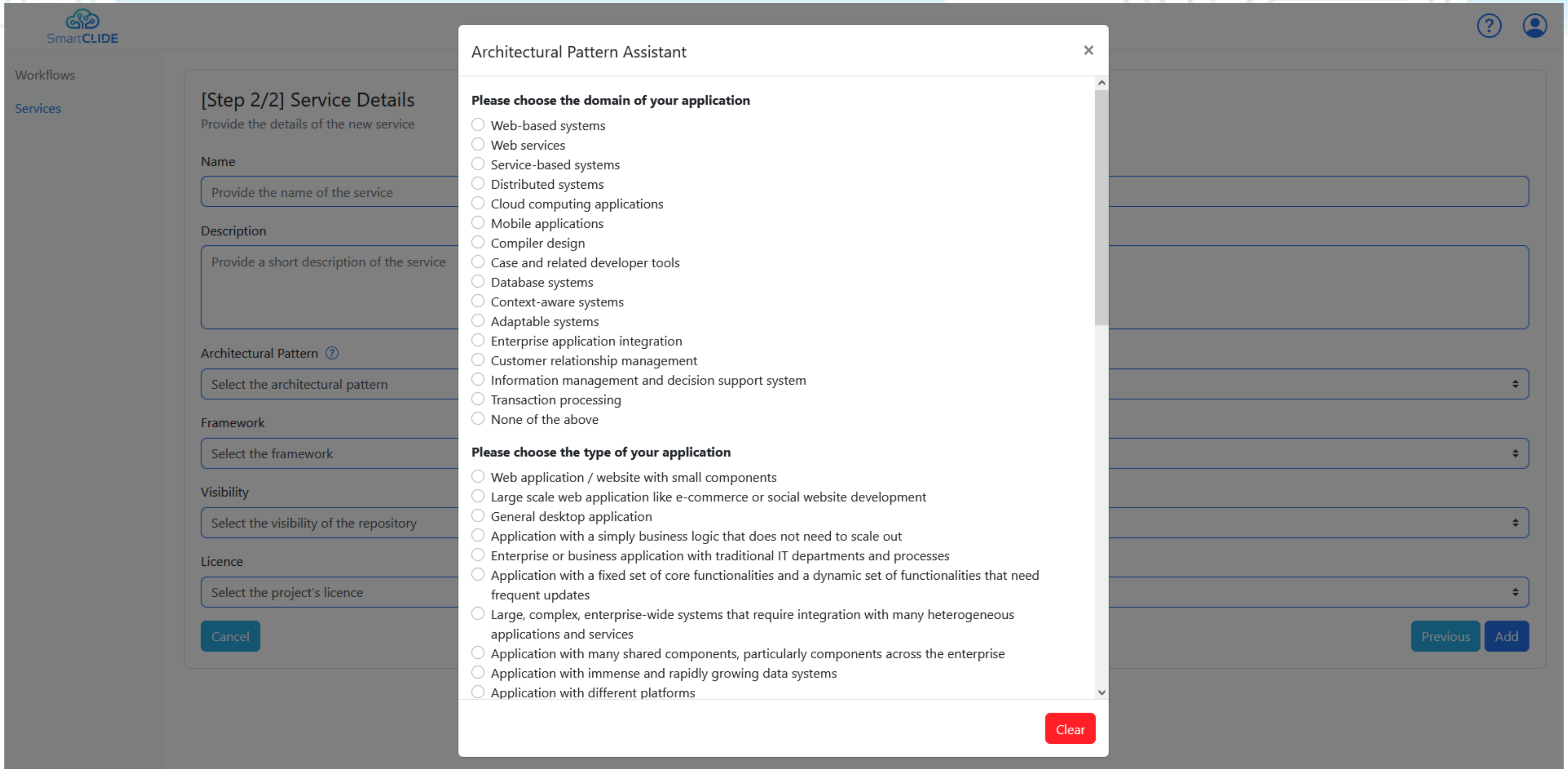
Previous Add

Oct. 17, 2023

Ludwigsburg, Germany

# Architectural Pattern Selection in SmartCLIDE IDE

eSAAM 2023
on Cloud to Edge Continuum


SmartCLIDE
Workflows
Services
[Step 2/2] Service Details
Provide the details of the new service
Name
Provide the name of the service
Description
Provide a short description of the service
Architectural Pattern
Select the architectural pattern
Framework
Select the framework
Visibility
Select the visibility of the repository
Licence
Select the project's licence
Cancel
Previous
Add
Clear

Architectural Pattern Assistant

**Please choose the domain of your application**

- ◯ Web-based systems
- ◯ Web services
- ◯ Service-based systems
- ◯ Distributed systems
- ◯ Cloud computing applications
- ◯ Mobile applications
- ◯ Compiler design
- ◯ Case and related developer tools
- ◯ Database systems
- ◯ Context-aware systems
- ◯ Adaptable systems
- ◯ Enterprise application integration
- ◯ Customer relationship management
- ◯ Information management and decision support system
- ◯ Transaction processing
- ◯ None of the above

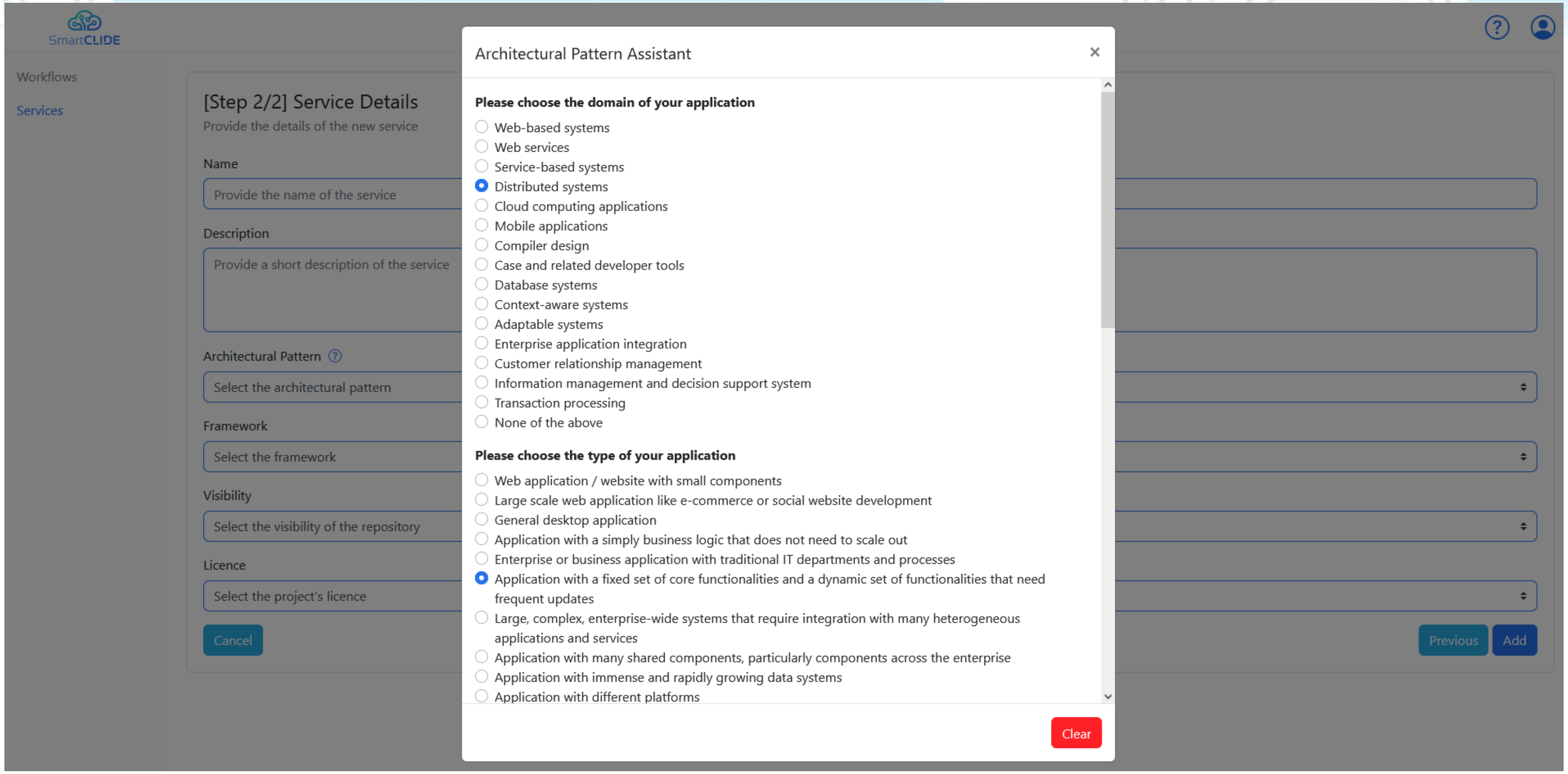
**Please choose the type of your application**

- ◯ Web application / website with small components
- ◯ Large scale web application like e-commerce or social website development
- ◯ General desktop application
- ◯ Application with a simply business logic that does not need to scale out
- ◯ Enterprise or business application with traditional IT departments and processes
- ◯ Application with a fixed set of core functionalities and a dynamic set of functionalities that need frequent updates
- ◯ Large, complex, enterprise-wide systems that require integration with many heterogeneous applications and services
- ◯ Application with many shared components, particularly components across the enterprise
- ◯ Application with immense and rapidly growing data systems
- ◯ Application with different platforms

Oct. 17, 2023 Ludwigsburg, Germany

# Architectural Pattern Selection in SmartCLIDE IDE

eSAAM 2023
on Cloud to Edge Continuum



[Step 2/2] Service Details
Provide the details of the new service

Name: Provide the name of the service

Description: Provide a short description of the service

Architectural Pattern: Select the architectural pattern

Framework: Select the framework

Visibility: Select the visibility of the repository

Licence: Select the project's licence

Architectural Pattern Assistant

**Please choose the domain of your application**

- Web-based systems
- Web services
- Service-based systems
- [✓ Distributed systems]
- Cloud computing applications
- Mobile applications
- Compiler design
- Case and related developer tools
- Database systems
- Context-aware systems
- Adaptable systems
- Enterprise application integration
- Customer relationship management
- Information management and decision support system
- Transaction processing
- None of the above

**Please choose the type of your application**

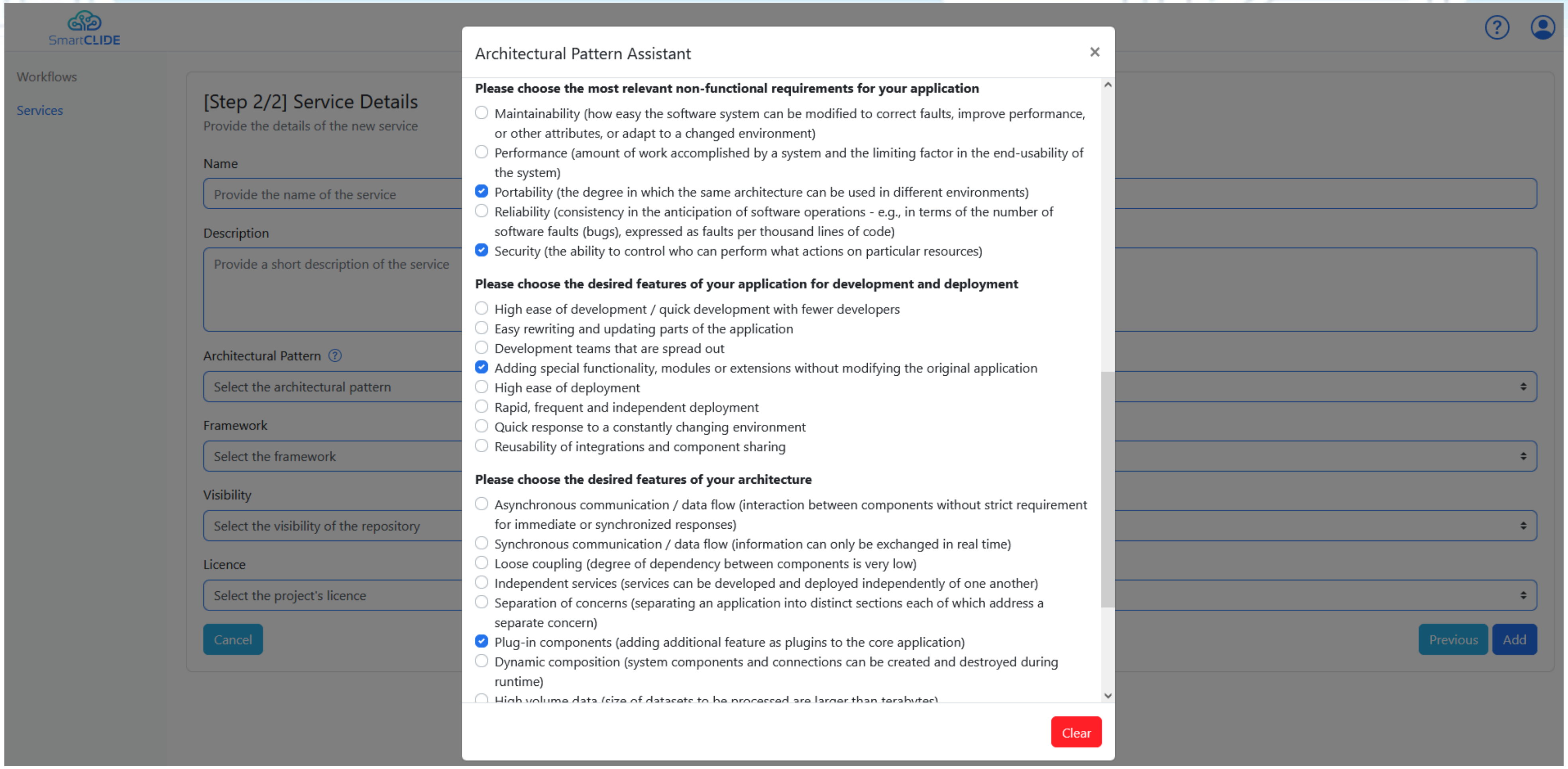
- Web application / website with small components
- Large scale web application like e-commerce or social website development
- General desktop application
- Application with a simply business logic that does not need to scale out
- Enterprise or business application with traditional IT departments and processes
- [✓ Application with a fixed set of core functionalities and a dynamic set of functionalities that need frequent updates]
- Large, complex, enterprise-wide systems that require integration with many heterogeneous applications and services
- Application with many shared components, particularly components across the enterprise
- Application with immense and rapidly growing data systems
- Application with different platforms

Oct. 17, 2023

Ludwigsburg, Germany

# Architectural Pattern Selection in SmartCLIDE IDE

eSAAM 2023
on Cloud to Edge Continuum



[Step 2/2] Service Details
Provide the details of the new service

Name
Provide the name of the service

Description
Provide a short description of the service

Architectural Pattern
Select the architectural pattern

Framework
Select the framework

Visibility
Select the visibility of the repository

Licence
Select the project's licence

Architectural Pattern Assistant

**Please choose the most relevant non-functional requirements for your application**

- ☐ Maintainability (how easy the software system can be modified to correct faults, improve performance, or other attributes, or adapt to a changed environment)
- ☐ Performance (amount of work accomplished by a system and the limiting factor in the end-usability of the system)
- ☑ Portability (the degree in which the same architecture can be used in different environments)
- ☐ Reliability (consistency in the anticipation of software operations - e.g., in terms of the number of software faults (bugs), expressed as faults per thousand lines of code)
- ☑ Security (the ability to control who can perform what actions on particular resources)

**Please choose the desired features of your application for development and deployment**

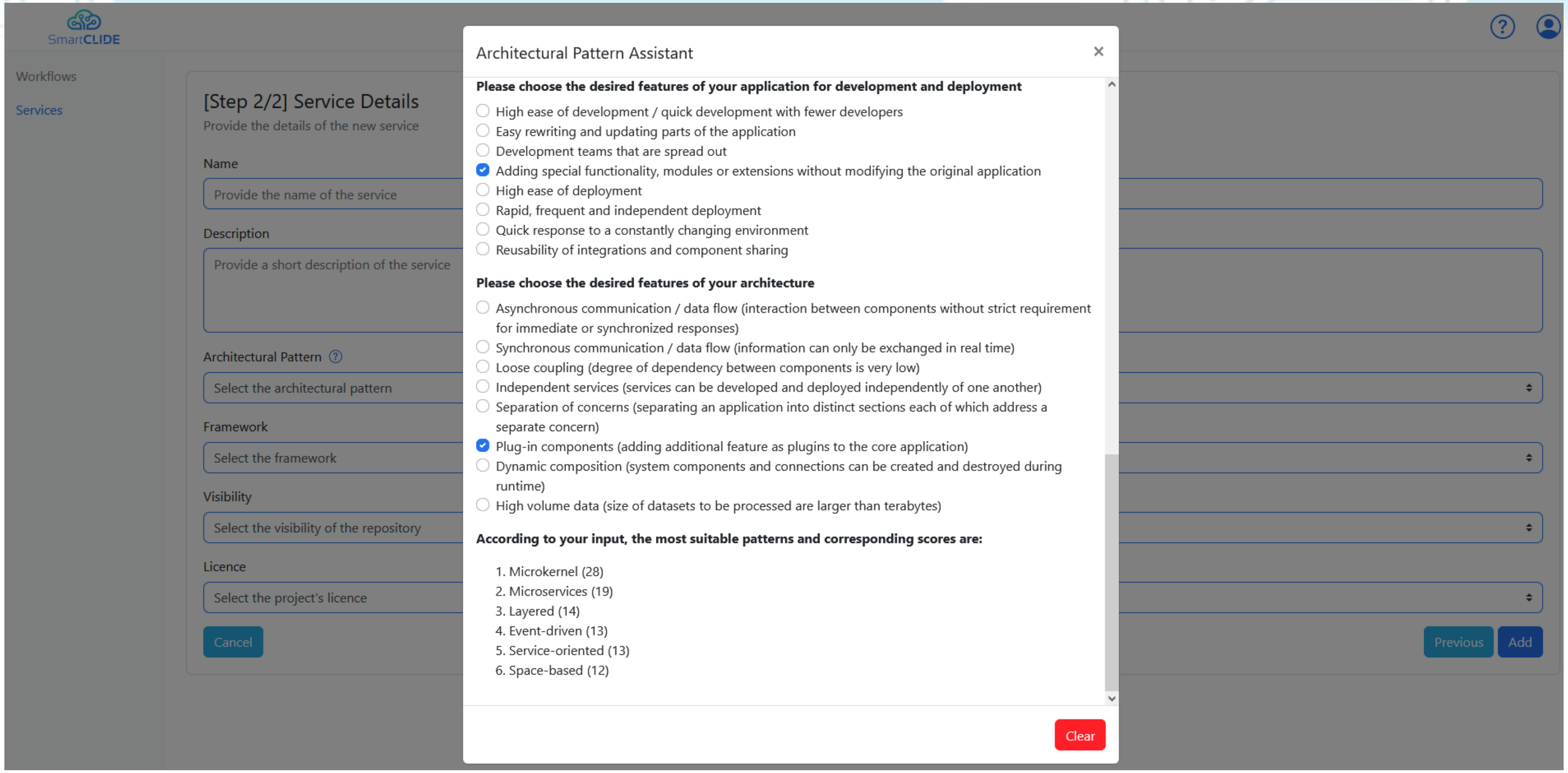
- ☐ High ease of development / quick development with fewer developers
- ☐ Easy rewriting and updating parts of the application
- ☐ Development teams that are spread out
- ☑ Adding special functionality, modules or extensions without modifying the original application
- ☐ High ease of deployment
- ☐ Rapid, frequent and independent deployment
- ☐ Quick response to a constantly changing environment
- ☐ Reusability of integrations and component sharing

**Please choose the desired features of your architecture**

- ☐ Asynchronous communication / data flow (interaction between components without strict requirement for immediate or synchronized responses)
- ☐ Synchronous communication / data flow (information can only be exchanged in real time)
- ☐ Loose coupling (degree of dependency between components is very low)
- ☐ Independent services (services can be developed and deployed independently of one another)
- ☐ Separation of concerns (separating an application into distinct sections each of which address a separate concern)
- ☑ Plug-in components (adding additional feature as plugins to the core application)
- ☐ Dynamic composition (system components and connections can be created and destroyed during runtime)
- ☐ High volume data (size of datasets to be processed are larger than terabytes)

# Architectural Pattern Selection in SmartCLIDE IDE

eSAAM 2023
on Cloud to Edge Continuum


SmartCLIDE
Workflows
Services
[Step 2/2] Service Details
Provide the details of the new service
Name
Provide the name of the service
Description
Provide a short description of the service
Architectural Pattern
Select the architectural pattern
Framework
Select the framework
Visibility
Select the visibility of the repository
Licence
Select the project's licence
Cancel
Previous
Add

Architectural Pattern Assistant

**Please choose the desired features of your application for development and deployment**

- [ ] High ease of development / quick development with fewer developers
- [ ] Easy rewriting and updating parts of the application
- [ ] Development teams that are spread out
- [✓] Adding special functionality, modules or extensions without modifying the original application
- [ ] High ease of deployment
- [ ] Rapid, frequent and independent deployment
- [ ] Quick response to a constantly changing environment
- [ ] Reusability of integrations and component sharing

**Please choose the desired features of your architecture**

- [ ] Asynchronous communication / data flow (interaction between components without strict requirement for immediate or synchronized responses)
- [ ] Synchronous communication / data flow (information can only be exchanged in real time)
- [ ] Loose coupling (degree of dependency between components is very low)
- [ ] Independent services (services can be developed and deployed independently of one another)
- [ ] Separation of concerns (separating an application into distinct sections each of which address a separate concern)
- [✓] Plug-in components (adding additional feature as plugins to the core application)
- [ ] Dynamic composition (system components and connections can be created and destroyed during runtime)
- [ ] High volume data (size of datasets to be processed are larger than terabytes)

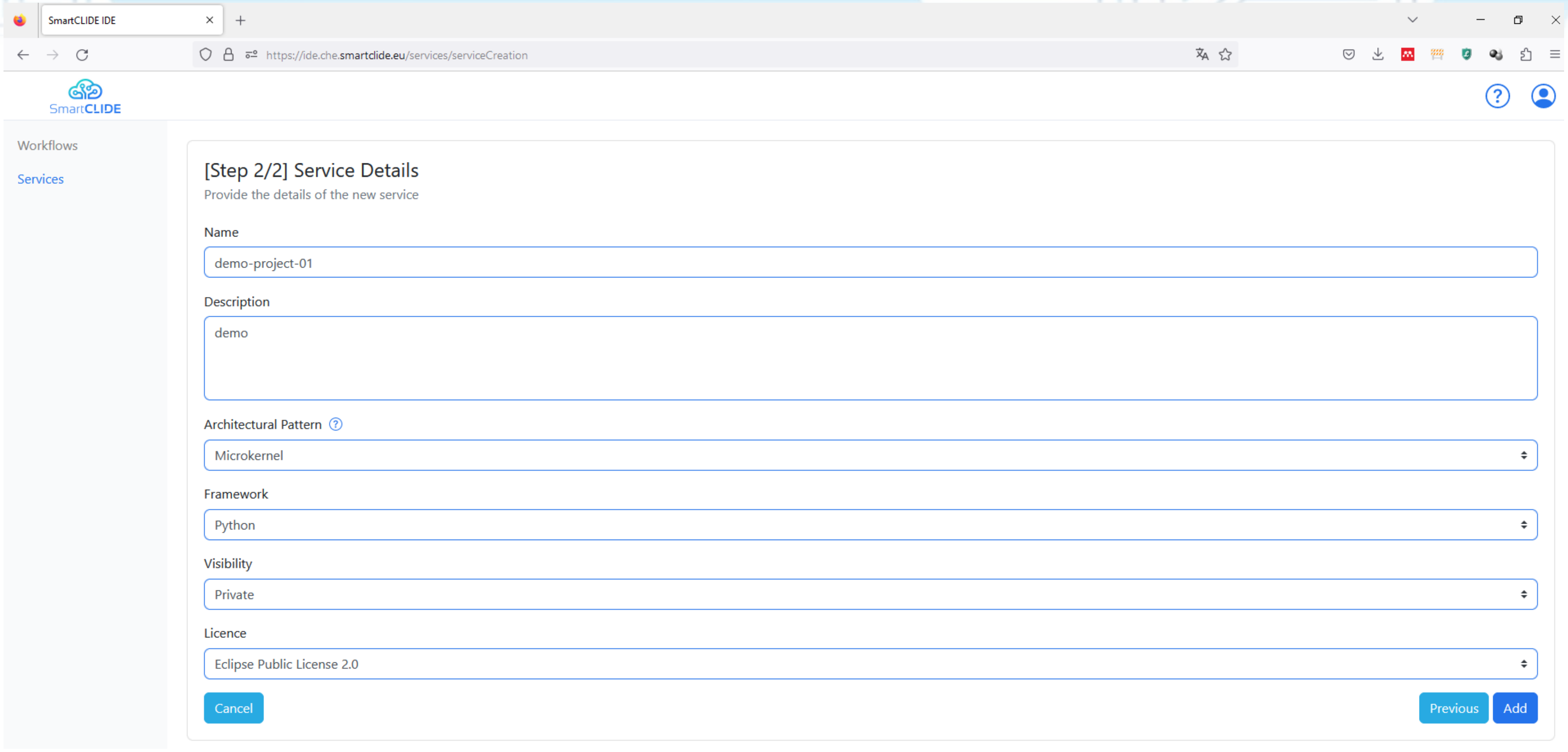
**According to your input, the most suitable patterns and corresponding scores are:**

1. Microkernel (28)
2. Microservices (19)
3. Layered (14)
4. Event-driven (13)
5. Service-oriented (13)
6. Space-based (12)

Clear

# Architectural Pattern Selection in SmartCLIDE IDE

**eSAAM 2023**
**on Cloud to Edge Continuum**



[Step 2/2] Service Details

Provide the details of the new service

Name: demo-project-01

Description: demo

Architectural Pattern: Microkernel

Framework: Python

Visibility: Private

Licence: Eclipse Public License 2.0

Oct. 17, 2023

Ludwigsburg, Germany

## Future Work

- Increase # of patterns supported
- Support pattern combinations
- Improve survey content & evaluation
- Add explanation to pattern suggestions
- Add alternative structures to repository templates

# eSAAM 2023

## on Cloud to Edge Continuum

SmartCLIDE

https://ide.che.smartclide.eu/

Sponsored by:

EUCloudEdgeIoT.eu CODECO NEMO nephele

Organized by:

POLITÉCNICA ECLIPSE FOUNDATION UNIVERSITY OF MACEDONIA