2022

Open Model

GEORGE BROWNBRIDGE (CMCL)

Salisbury House, Station Road, Cambridge CB1 2LA, UK

WP4 - D4.7 BASIC OBJECT BASED MCO WITH PRESELECTED FUNCTIONS

D4.7- BASIC OBJECT BASED MCO WITH PRESELECTED FUNCTIONS

DOCUMENT CONTROL

Document Type	Deliverable Attachment Report
Status	Finalt
Version	1.0
Responsible	George Brownbridge (CMCL)
Author(s)	George Brownbridge (CMCL)
Release Date	2022-04-29, 2023-01-24

ABSTRACT

This report is an attachment to the Deliverable 4.7, “Basic object based MCO with preselected functions” which is due by M15 (April 2022) in OpenModel project. The deliverable itself is a release*, with public dissemination level, and current report is an attachment explaining the development process and access details.

(*) Nature of deliverable: Other

CHANGE HISTORY

Version	Date	Comment
1	28-04-2022	Initial version, by George
1	24-01-2023	Request from EC added

DISSEMINATION LEVEL

PU	Public	X
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

TABLE OF CONTENT

WP4 - D4.7 Basic object based MCO with preselected functions 1

Document Control 1

Abstract 1

Change History 1

Dissemination level 2

Table of Content 3

List of Figures 4

List of Tables 4

1 INTRODUCTION 5

2 Development process 6

2.1 Overview 6

2.2 MoDS servlet 6

2.3 MoDS-CUDS wrapper 7

3 Annex 1 9

4 ACKNOWLEDGMENT 13

LIST OF FIGURES

Figure 2.1 Overview of the developed modules involved in this deliverable. 6

Figure 3.2 Example of a JSON object passed to the MoDS agent from the MoDS CUDS wrapper. 9

Figure 3.3 Screenshot showing the MoDS servlet code deployed in a repository............................... 10

Figure 3.4 Snippet of the YAML file that defines the ontology from which the CUDS object is generated. 11

Figure 3.5 Screenshot showing the code for the MoDS-CUDS wrapper deployed in a repository....... 12

LIST OF TABLES

No table of figures entries found.

D4.7- BASIC OBJECT BASED MCO WITH PRESELECTED FUNCTIONS

1 INTRODUCTION

Current document acts as an attachment to the Deliverable D4.7, “Basic object based MCO with pre-selected functions” in OpenModel project. As the deliverable itself is a release (nature of deliverable is “other”), this document outlines the development process and some more details about access to the deliverable in further details.

2 DEVELOPMENT PROCESS

2.1 OVERVIEW

This deliverable is to produce a demonstrator to show that it is possible to perform Multi-Criteria Optimization (MCO) in a way that can be used to suggest numerical model parameters as part of the model selection process within the Open Innovation Platform (OIP) in OpenModel project.

To facilitate this, two components have been developed as part of this deliverable. These are the "Wrapper" and "MoDS" blocks shown in Figure 2.1. The latter refers to the advanced statistical and data analytics software developed and licensed by CMCL, with the MCO feature that has been used in this deliverable.

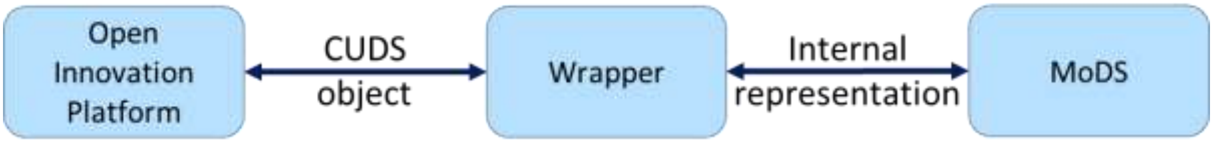


Figure 2.1 Overview of the developed modules involved in this deliverable.

2.2 MODS SERVLET

The "MoDS" block represents a web servlet that can receive an HTTP request and create the appropriate job for CMCL's Model Development Suite (MoDS) software to run. This code has been largely developed from scratch for this deliverable but has been designed so that it can be extended in the future to allow more of the capabilities of the MoDS software to be made available.

A template system has been used to load preconfigured settings for the requested job to which the user specified data and additional configuration settings are added. An example of the user supplied information that can be passed in the initial HTTP request is shown in Figure 3.4.

The response from the initial HTTP request can then be used to request the results of that job. In this case the results are the Pareto front generated by a Multi-Objective Optimization (MOO) algorithm within the MoDS software.

A JSON formatted object was chosen for the "internal representation" accepted by this MoDS servlet because of its common use for passing data in HTTP requests.

A screenshot of the repository containing the code for this component is shown in Figure 3.3.

2.3 MODS-CUDS WRAPPER

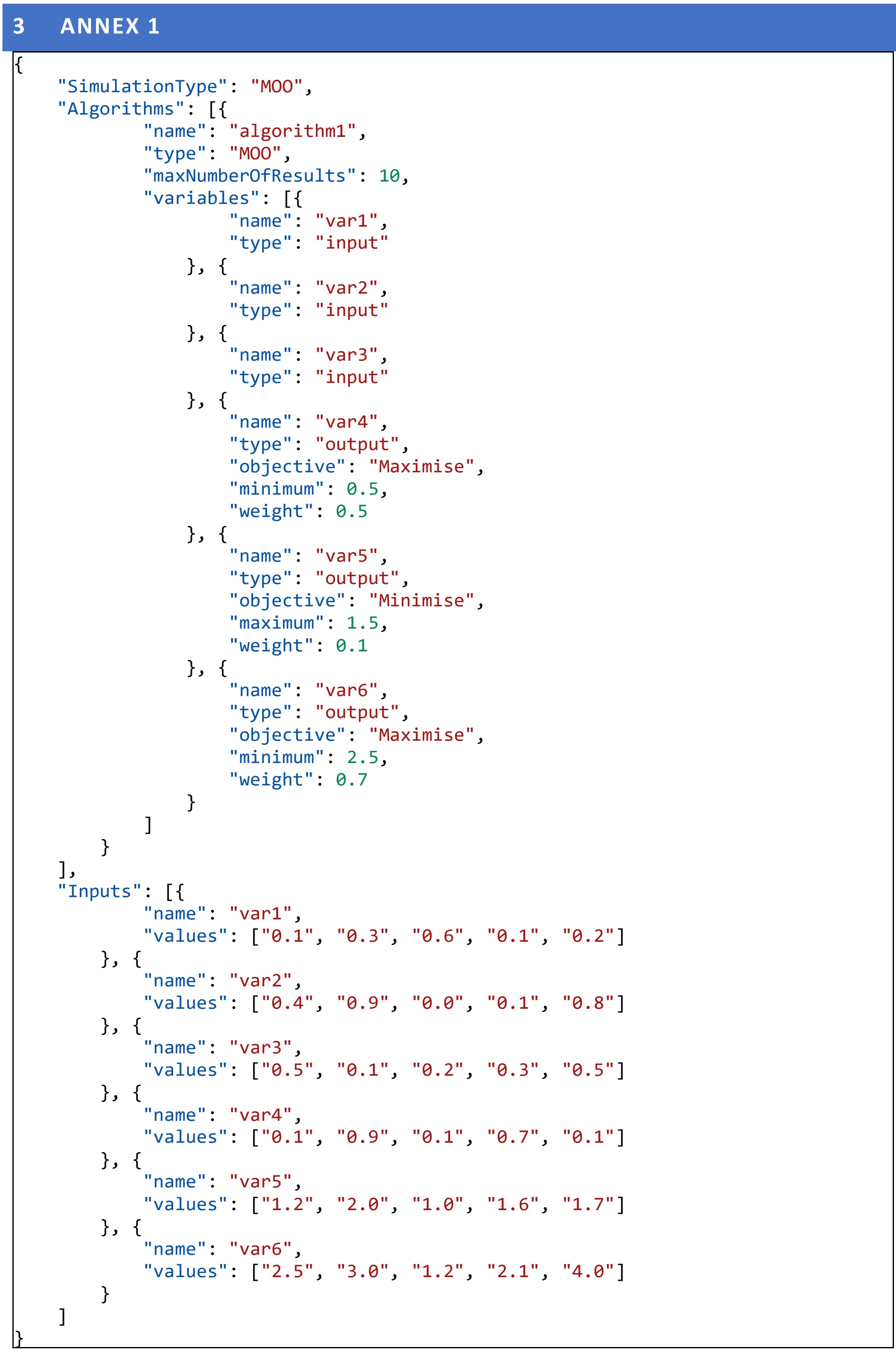
To integrate the MoDS servlet with the Open Innovation Platform an existing wrapper, developed as part of the OntoTrans project (https://ontotrans.eu), was extended and improved upon as part of this deliverable. This wrapper accepts CUDS [1] objects, generated from a simple ontology developed for this purpose. A snippet of the ontology is shown in Figure 3.4.

A screenshot of the repository containing the code for this component is shown in Figure 3.5.

BIBLIOGRAPHY

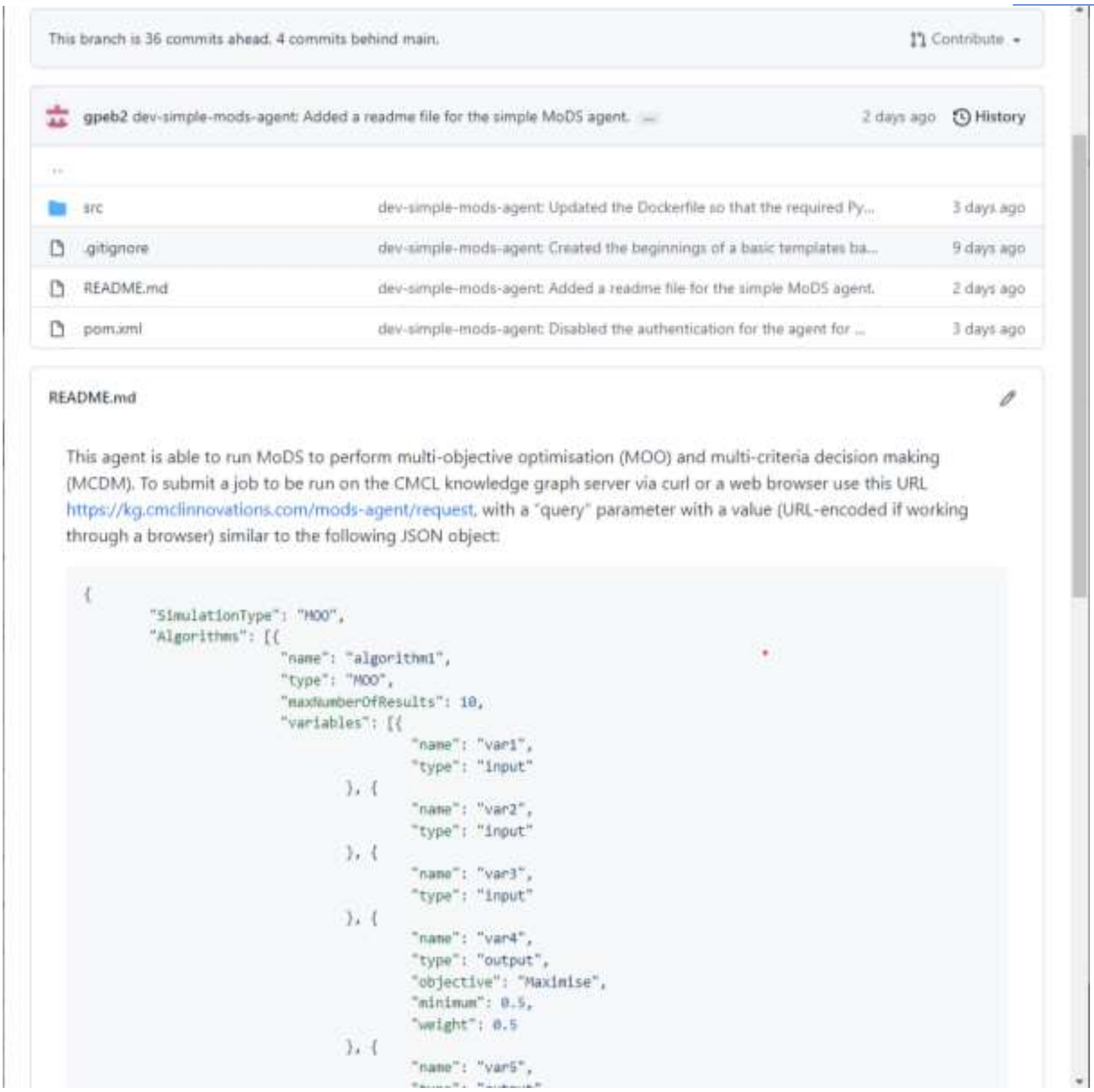
[1] A. Hashibon, T. Rasp, N. Franklin, I. Tziakos, D. Pinte, P. Dadvand, C. Roig, K. Mattila, T. Puurtinen, K. Hiltunen und et al., *Common universal data structures (CUDS) and vocabulary in the SimPhoNy integrated framework,* 2015.

3 ANNEX 1



```
{
    "SimulationType": "MOO",
    "Algorithms": [{
            "name": "algorithm1",
            "type": "MOO",
            "maxNumberOfResults": 10,
            "variables": [{
                    "name": "var1",
                    "type": "input"
                }, {
                    "name": "var2",
                    "type": "input"
                }, {
                    "name": "var3",
                    "type": "input"
                }, {
                    "name": "var4",
                    "type": "output",
                    "objective": "Maximise",
                    "minimum": 0.5,
                    "weight": 0.5
                }, {
                    "name": "var5",
                    "type": "output",
                    "objective": "Minimise",
                    "maximum": 1.5,
                    "weight": 0.1
                }, {
                    "name": "var6",
                    "type": "output",
                    "objective": "Maximise",
                    "minimum": 2.5,
                    "weight": 0.7
                }
            ]
        }
    ],
    "Inputs": [{
            "name": "var1",
            "values": ["0.1", "0.3", "0.6", "0.1", "0.2"]
        }, {
            "name": "var2",
            "values": ["0.4", "0.9", "0.0", "0.1", "0.8"]
        }, {
            "name": "var3",
            "values": ["0.5", "0.1", "0.2", "0.3", "0.5"]
        }, {
            "name": "var4",
            "values": ["0.1", "0.9", "0.1", "0.7", "0.1"]
        }, {
            "name": "var5",
            "values": ["1.2", "2.0", "1.0", "1.6", "1.7"]
        }, {
            "name": "var6",
            "values": ["2.5", "3.0", "1.2", "2.1", "4.0"]
        }
    ]
}
```

Figure 3.2 Example of a JSON object passed to the MoDS agent from the MoDS CUDS wrapper.



This branch is 36 commits ahead, 4 commits behind main.

gpeb2 dev-simple-mods-agent: Added a readme file for the simple MoDS agent. … 2 days ago

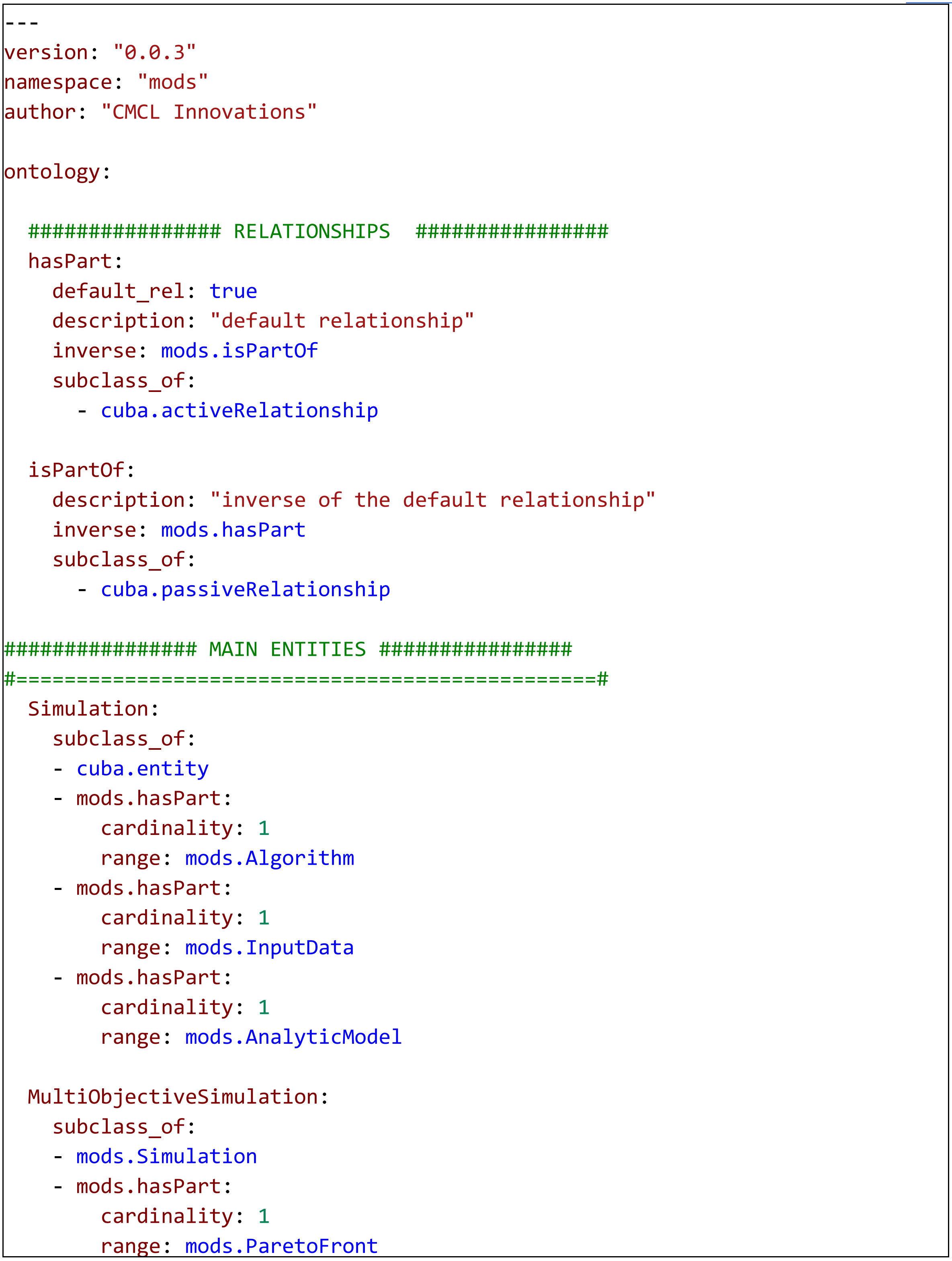
src	dev-simple-mods-agent: Updated the Dockerfile so that the required Py...	3 days ago
.gitignore	dev-simple-mods-agent: Created the beginnings of a basic templates ba...	9 days ago
README.md	dev-simple-mods-agent: Added a readme file for the simple MoDS agent.	2 days ago
pom.xml	dev-simple-mods-agent: Disabled the authentication for the agent for ...	3 days ago

README.md

This agent is able to run MoDS to perform multi-objective optimisation (MOO) and multi-criteria decision making (MCDM). To submit a job to be run on the CMCL knowledge graph server via curl or a web browser use this URL https://kg.cmclinnovations.com/mods-agent/request, with a "query" parameter with a value (URL-encoded if working through a browser) similar to the following JSON object:

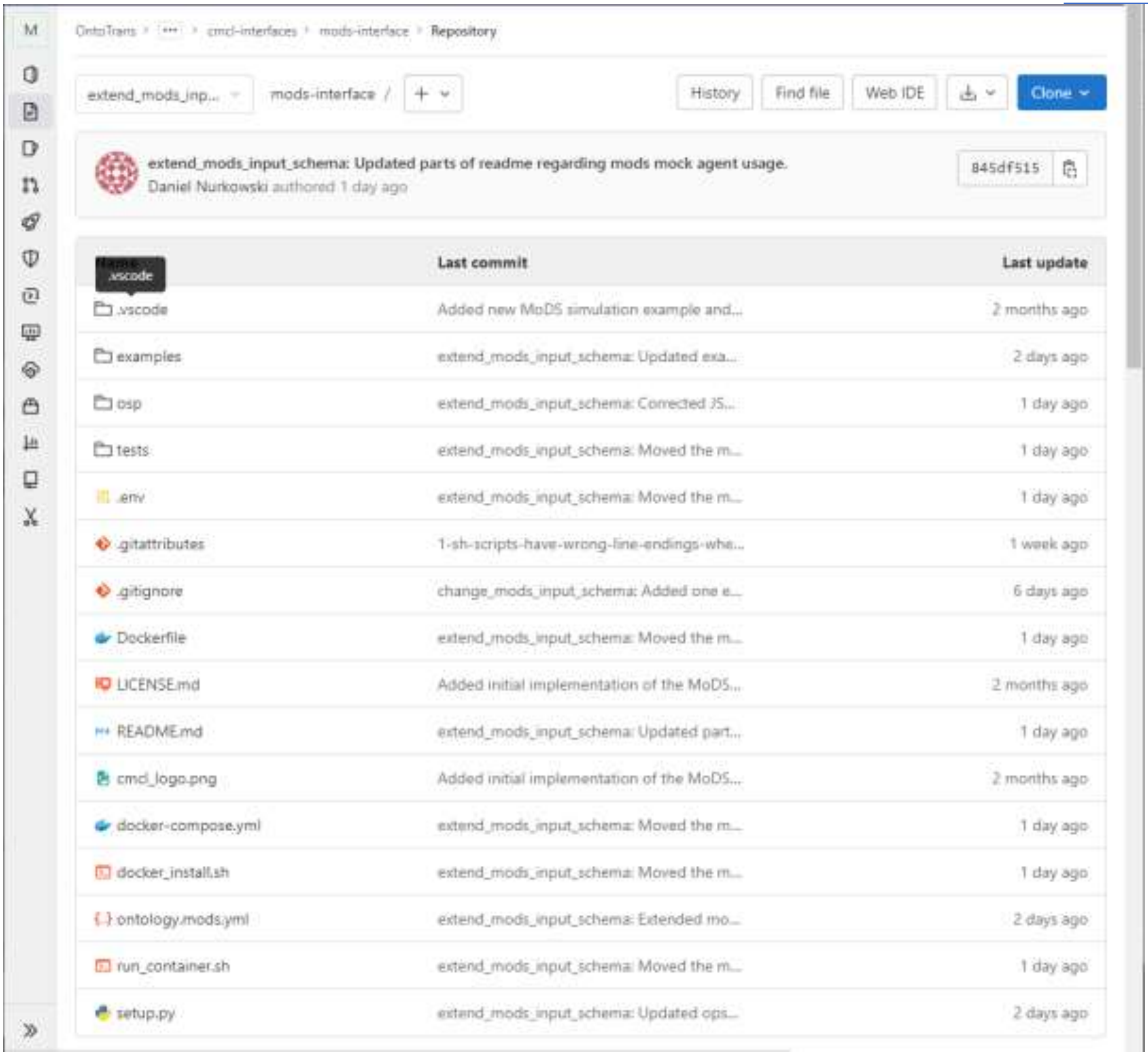
```
{
        "SimulationType": "MOO",
        "Algorithms": [{
                        "name": "algorithm1",
                        "type": "MOO",
                        "maxNumberOfResults": 10,
                        "variables": [{
                                        "name": "var1",
                                        "type": "input"
                                }, {
                                        "name": "var2",
                                        "type": "input"
                                }, {
                                        "name": "var3",
                                        "type": "input"
                                }, {
                                        "name": "var4",
                                        "type": "output",
                                        "objective": "Maximise",
                                        "minimum": 0.5,
                                        "weight": 0.5
                                }, {
                                        "name": "var5",
                                        "type": "output"
```

Figure 3.3 Screenshot showing the MoDS servlet code deployed in a repository.



```yaml
---
version: "0.0.3"
namespace: "mods"
author: "CMCL Innovations"

ontology:

  ################ RELATIONSHIPS  ################
  hasPart:
    default_rel: true
    description: "default relationship"
    inverse: mods.isPartOf
    subclass_of:
      - cuba.activeRelationship

  isPartOf:
    description: "inverse of the default relationship"
    inverse: mods.hasPart
    subclass_of:
      - cuba.passiveRelationship

################ MAIN ENTITIES ################
#================================================#
  Simulation:
    subclass_of:
    - cuba.entity
    - mods.hasPart:
        cardinality: 1
        range: mods.Algorithm
    - mods.hasPart:
        cardinality: 1
        range: mods.InputData
    - mods.hasPart:
        cardinality: 1
        range: mods.AnalyticModel

  MultiObjectiveSimulation:
    subclass_of:
    - mods.Simulation
    - mods.hasPart:
        cardinality: 1
        range: mods.ParetoFront
```

Figure 3.4 Snippet of the YAML file that defines the ontology from which the CUDS object is generated.



extend_mods_input_schema: Updated parts of readme regarding mods mock agent usage.
Daniel Nurkowski authored 1 day ago

845df515

Name	Last commit	Last update
.vscode	Added new MoDS simulation example and...	2 months ago
examples	extend_mods_input_schema: Updated exa...	2 days ago
osp	extend_mods_input_schema: Corrected JS...	1 day ago
tests	extend_mods_input_schema: Moved the m...	1 day ago
.env	extend_mods_input_schema: Moved the m...	1 day ago
.gitattributes	1-sh-scripts-have-wrong-line-endings-whe...	1 week ago
.gitignore	change_mods_input_schema: Added one e...	6 days ago
Dockerfile	extend_mods_input_schema: Moved the m...	1 day ago
LICENSE.md	Added initial implementation of the MoDS...	2 months ago
README.md	extend_mods_input_schema: Updated part...	1 day ago
cmcl_logo.png	Added initial implementation of the MoDS...	2 months ago
docker-compose.yml	extend_mods_input_schema: Moved the m...	1 day ago
docker_install.sh	extend_mods_input_schema: Moved the m...	1 day ago
ontology.mods.yml	extend_mods_input_schema: Extended mo...	2 days ago
run_container.sh	extend_mods_input_schema: Moved the m...	1 day ago
setup.py	extend_mods_input_schema: Updated ops...	2 days ago

Figure 3.5 Screenshot showing the code for the MoDS-CUDS wrapper deployed in a repository.

4 ACKNOWLEDGMENT

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 953167.

This document and all information contained herein is the sole property of the OpenModel Consortium. It may contain information subject to intellectual property rights. No intellectual property rights are granted by the delivery of this document or the disclosure of its content.

Reproduction or circulation of this document to any third party is prohibited without the consent of the author(s).

The content of this deliverable does not reflect the official opinion of the European Union. Responsibility for the information and views expressed herein lies entirely with the author(s).