Grant Agreement: 872698

Digital innovation HUBs and CollAborative Platform for cyber-physical systems

HUBCAP

The Initial HUBCAP Model-Based Services

Technical Note Number: D6.2

Version: 1.0

Date: September 2020

Public Document

http://hubcap.au.dk

Editors

Adrian Pop (RISE)

Tools Contributors

Stefano Tonetta (FBK)
Pietro Braghieri (FBK)
Simon Barner (FOR)
Adrian Pop (RISE)
Hugo Daniel Macedo (AU)
Felix Schaller (VAL)
Dario Pietraroia (TTS)
Bas Gunnink (CLP)
Jörg Brauer (VSI)

Catalogues Support

Rosamaria Maniaci

Reviewers

Jalil Boudjadar (AU)
Marco Bozzano (FBK)

Consortium

Aarhus University	AU	Newcastle University	UNEW
Fortiss GmbH	FOR	Virtual Vehicle Research GmbH	VV
Fondazione Bruno Kessler	FBK	KTH Royal Institute of Technology	KTH
University “Lucian Blaga” of Sibiu	ULBS	Engineering Ingegneria Informatica S.p.A.	ENGIT
Research Institutes of Sweden AB	RISE	F6S Network Limited	F6S
Politecnico di Milano	POLIMI	Unparallel Innovation	UNP
Controllab Products	CLP	BEIA Consult	BEIA
Verified Systems International GmbH	VSI	Validas	VAL
Technology Transfer Systems srl	TTS		

Document History

Ver.	Date	Author	Description
0.1	10-07-2020	Adrian Pop	Structure
0.2	31-08-2020	Adrian Pop	Final draft before review
1.0	30-09-2020	Adrian Pop	Final

Abstract

The **HUBCAP Collaborative Platform** will provide **catalogues of models and tools** for Model-Based Design (MBD) for trial experiments to design and develop innovative CPS solutions with the support of MBD technology.

D6.2 consists of an **initial set of tools** provided by HUBCAP partners that are uploaded, installed and tested inside the HUBCAP Collaborative Platform **sandbox** to constitute the contents of the first catalogue of tools. The catalogue will continuously be enriched along the project by the HUBCAP partners, together with the winners of the Open Calls, and in the future by the HUBCAP ecosystem.

D6.2 is defined as of **type "OTHER"**, given by the actual catalogue of models and the tools that are installed in the sandbox. The catalogues can be browsed by accessing the platform. This document summarizes the contents of the **tools** catalogue for the reviewer's convenience.

List of Abbreviations

DIH	Digital Innovation Hub
CMS	Catalogues Management System
HTTP	HyperText Transfer Protocol
IdM	Identity Management
JSON	JavaScript Object Notation
KMS	Knowledge Management System
REST	Representational State Transfer
SSO	Single Sign On
FMI	Functional Mock-up Interface
FMU	Functional Mock-up Unit
IDE	Integrated Development Environment
VDM	Vienna Development Method
UML	Unified Modeling Language
SysML	Systems Modeling Language
DSE	Design Space Exploration
GUI	Graphical User Interface
CLI	Command Line Interface
RCP	Rich Client Platform
FTA	Fault Tree Analysis
MCS	Minimal Cut Sets
FMEA	Failure Modes and Effects Analysis
TFPG	Timed Failure Propagation Graphs
CCA	Common Cause Analysis
SAT	Boolean Satisfiability Problem
SMT	SAT Modulo Theories
SME	Small and Medium-sized Enterprises
COE	Co-simulation Orchestration Engine
MBT	Model Based Testing
CPU	Central Processing Unit
GPU	Graphics Processing Unit
DMA	Direct Memory Access
UAV	Unmanned Aerial Vehicle

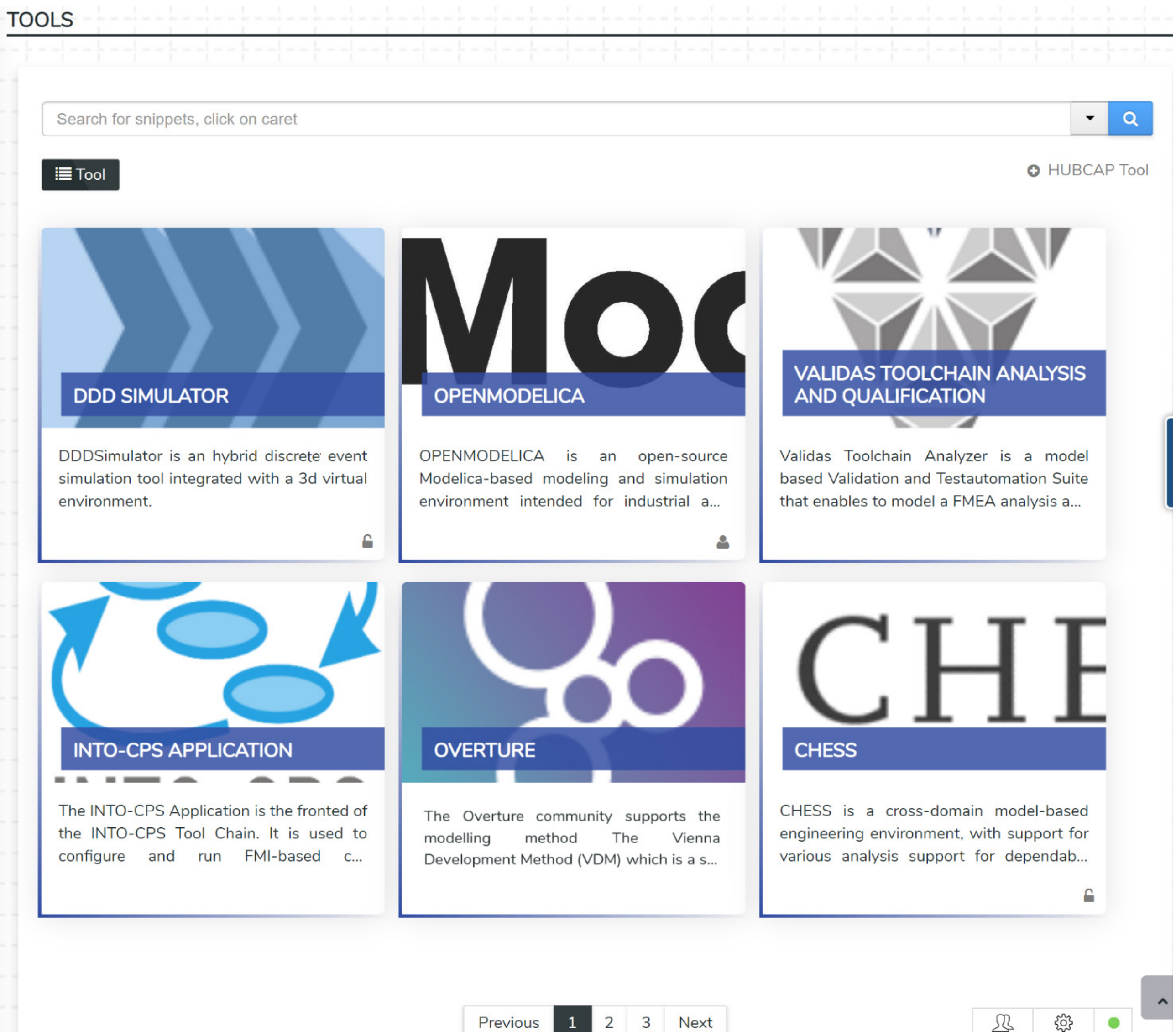
Contents

1. Introduction

Model-Based Design (MBD) is an approach to address the design of complex systems with models. It prescribes the use of models through the development process in order to represent system structure and behaviours, providing a basis for machine-assisted analysis of system properties, and supporting design decisions through processes of refinement into implementation. The purpose is 1) to **reduce the complexity** of design **by abstraction**; 2) to **ensure the quality** of the system **by rigorous analysis** of its properties; 3) to **reduce the cost** of the development **by detecting issues** in the **early** phases of the development. MBD is quite standard in software engineering and is becoming more and more relevant in systems engineering, where it must integrate methods for control engineering and safety engineering.

MBD appears largely to be applied in domains such as aerospace where the return on investment can take decades. By contrast, SMEs require considerable flexibility to change processes to adopt MBD and may lack in-house expertise. In addition, the selection, procurement, training and deployment costs for some methods and tools can be discouragingly high. In general, it is difficult for SMEs to invest in acquiring the necessary background for example because of the high license fees from commercial vendors of MBD assets.

The **HUBCAP Collaborative Platform** will lower such barriers by providing **catalogues of models and tools** for Model-Based Design (MBD) to support trial experiments that can design and develop innovative CPS solutions with the support of MBD technology. D6.2 consists of an **initial set of tools** provided by HUBCAP partners that are uploaded, installed and tested inside the HUBCAP Collaborative Platform **sandbox** to constitute the contents of the first catalogue of tools. The tools described in this deliverable can use the **initial set of models** described in D6.1 and the new models added since the D6.1 deliverable which are presented in Section 4. This document summarizes the contents of the **tools catalogue** for the reviewer's convenience.


TOOLS
Search for snippets, click on caret
Tool
HUBCAP Tool
DDD SIMULATOR
DDDSimulator is an hybrid discrete event simulation tool integrated with a 3d virtual environment.
OPENMODELICA
OPENMODELICA is an open-source Modelica-based modeling and simulation environment intended for industrial a...
VALIDAS TOOLCHAIN ANALYSIS AND QUALIFICATION
Validas Toolchain Analyzer is a model based Validation and Testautomation Suite that enables to model a FMEA analysis a...
INTO-CPS APPLICATION
The INTO-CPS Application is the fronted of the INTO-CPS Tool Chain. It is used to configure and run FMI-based c...
OVERTURE
The Overture community supports the modelling method The Vienna Development Method (VDM) which is a s...
CHESS
CHESS is a cross-domain model-based engineering environment, with support for various analysis support for dependab...
Previous
1
2
3
Next


TOOLS
Search for snippets, click on caret
Tool
HUBCAP Tool
AUTOFOCUS3
AutoFOCUS3 is a model-based tool and research platform for safety-critical embedded systems. It builds on a gener...
OCRA
A command-line tool for the verification of
DRAFT
XSAP
xSAP is a tool for safety assessment of
DRAFT
NUXMV
nuXmv is a symbolic model checker for the
DRAFT
20-SIM
20-sim is modeling and simulation
DRAFT
UNITY FMU PACKAGE
Bring your physics to life and create state-
DRAFT
Previous 1 2 3 Next

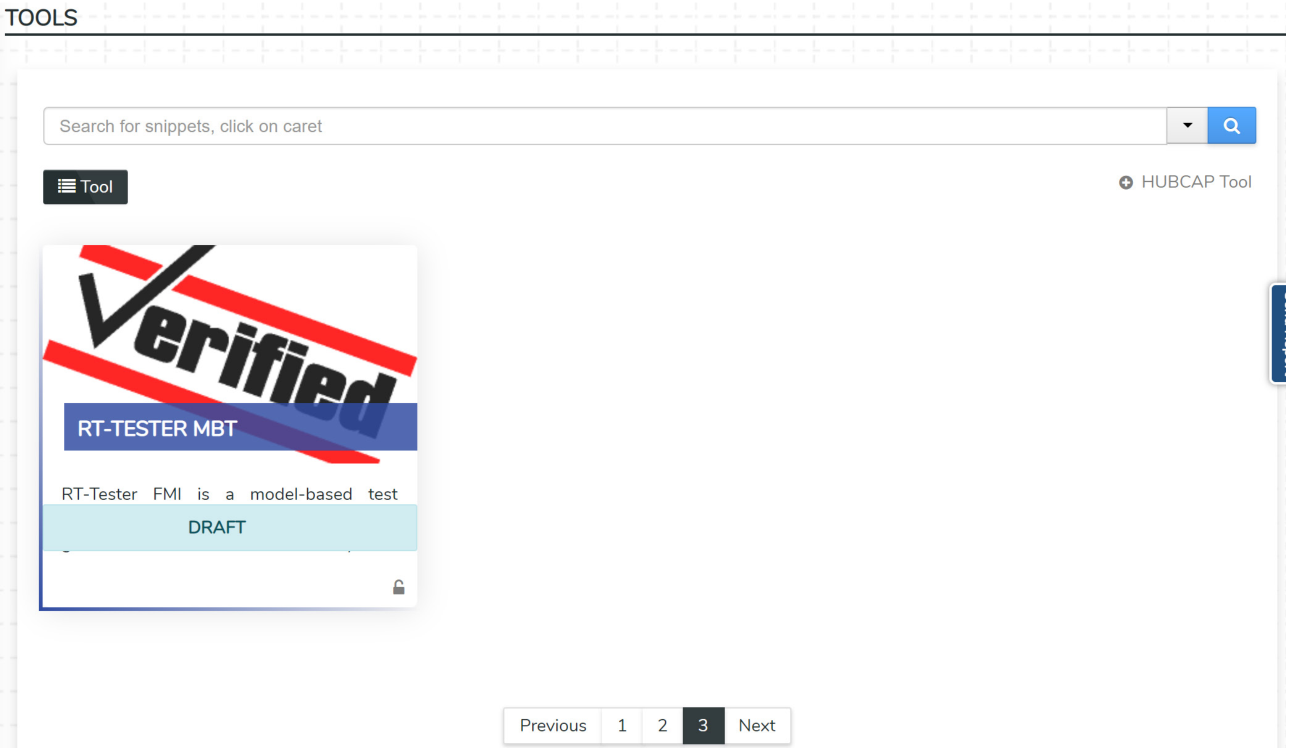


Figure 1. Screenshot of the tools catalogue

2. The Tools from HUBCAP partners

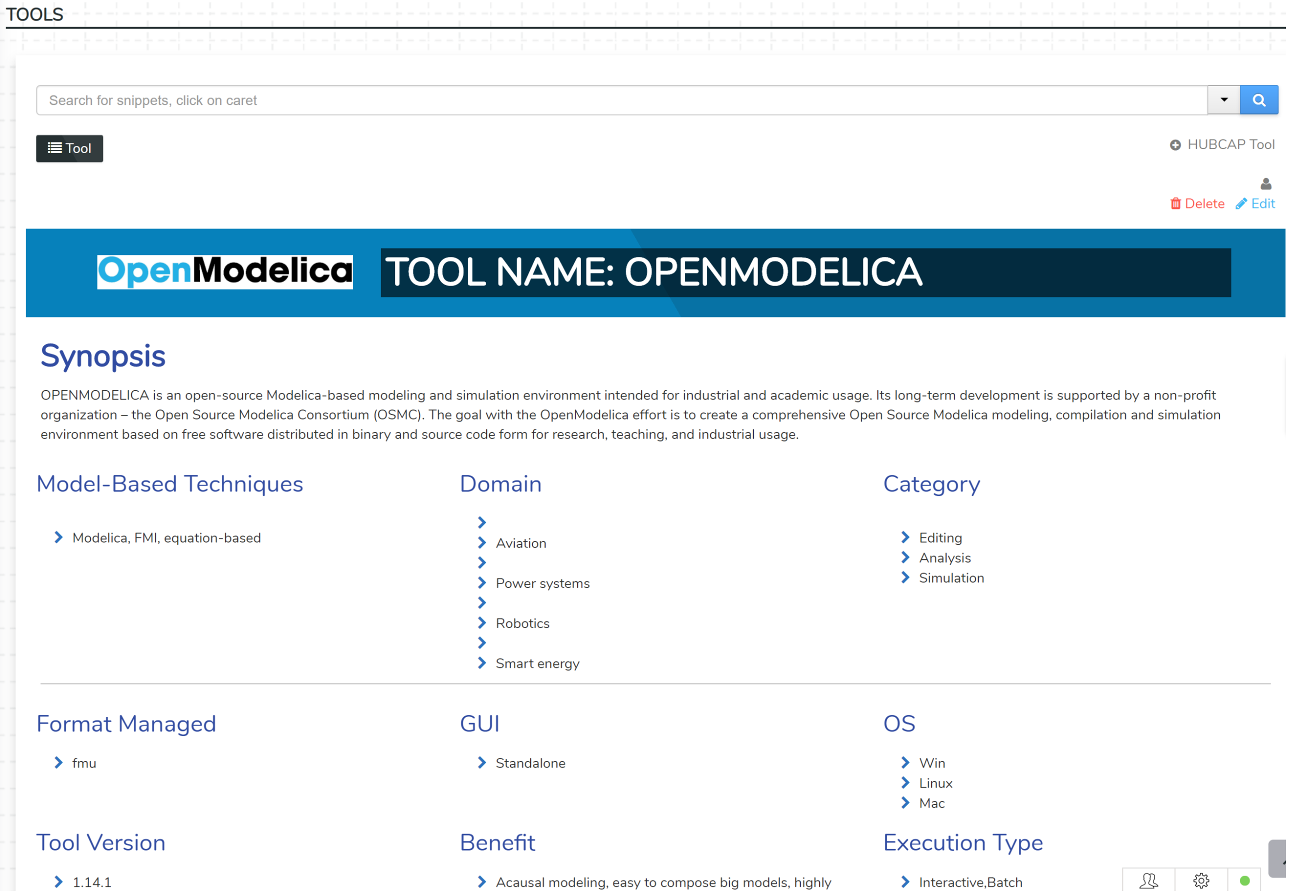
The following sections report the list of tools that are currently present on the platform, with a snapshot of the related platform page.
The URL for the platform is: https://hubcap-portal.eng.it/

Currently the tools catalogue is disconnected from the sandbox but ENGIT is working on integrating them. One should be able to launch the sandbox directly from the tools or model catalogues.

2.1. OpenModelica

Provided by: **RISE**
Sandbox integration status: **Integrated**

OpenModelica is an open-source Modelica-based modelling and simulation environment intended for industrial and academic usage. Its long-term development is supported by a non-profit organization – the Open Source Modelica Consortium (OSMC). The goal with the OpenModelica effort is to create a comprehensive Open Source Modelica modelling, compilation and simulation environment based on free software distributed in binary and source code form for academic and industrial usage.

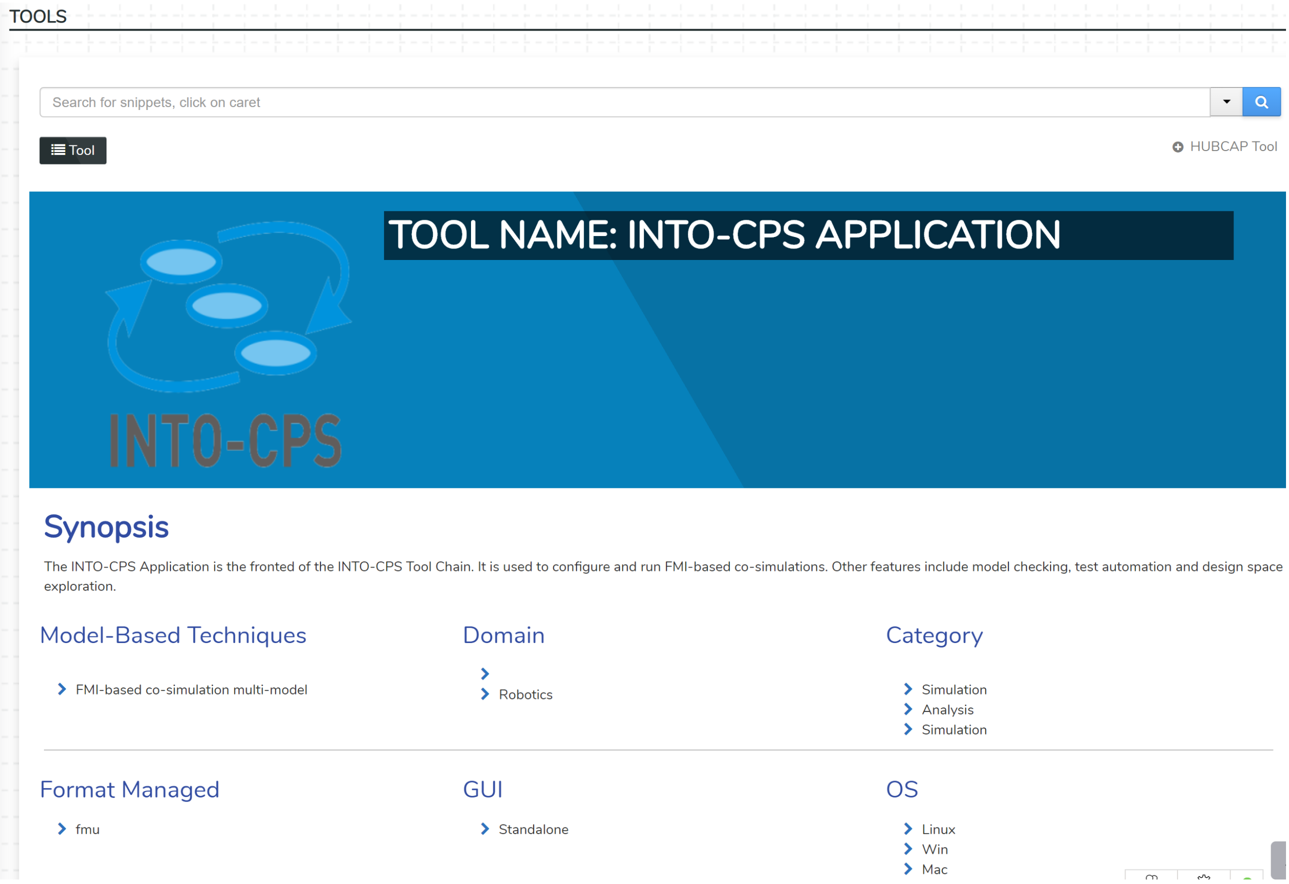


2.2. INTO-CPS Application

Provided by: **AU**

Sandbox integration status: **Integrated**

The INTO-CPS Application is the frontend of the INTO-CPS Tool Chain. It is used to configure and run FMI-based co-simulations. Other features include model checking, test automation and design space exploration.

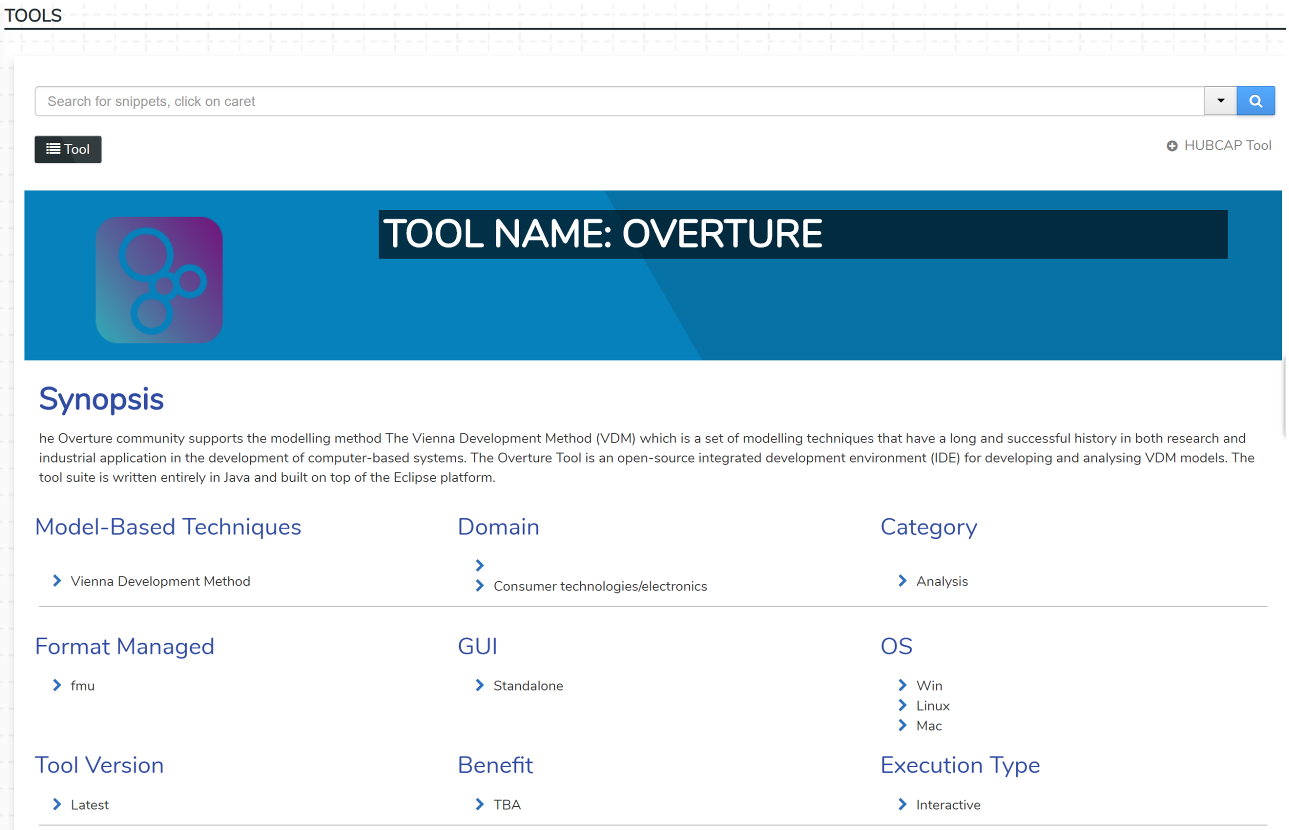


2.3. Overture

Provided by: **AU**

Sandbox integration status: **Integrated**

The Overture community supports the modelling method The Vienna Development Method (VDM) which is a set of modelling techniques that have a long and successful history in both research and industrial application in the development of computer-based systems. The Overture Tool is an open-source integrated development environment (IDE) for developing and analysing VDM models. The tool suite is written entirely in Java and built on top of the Eclipse platform.

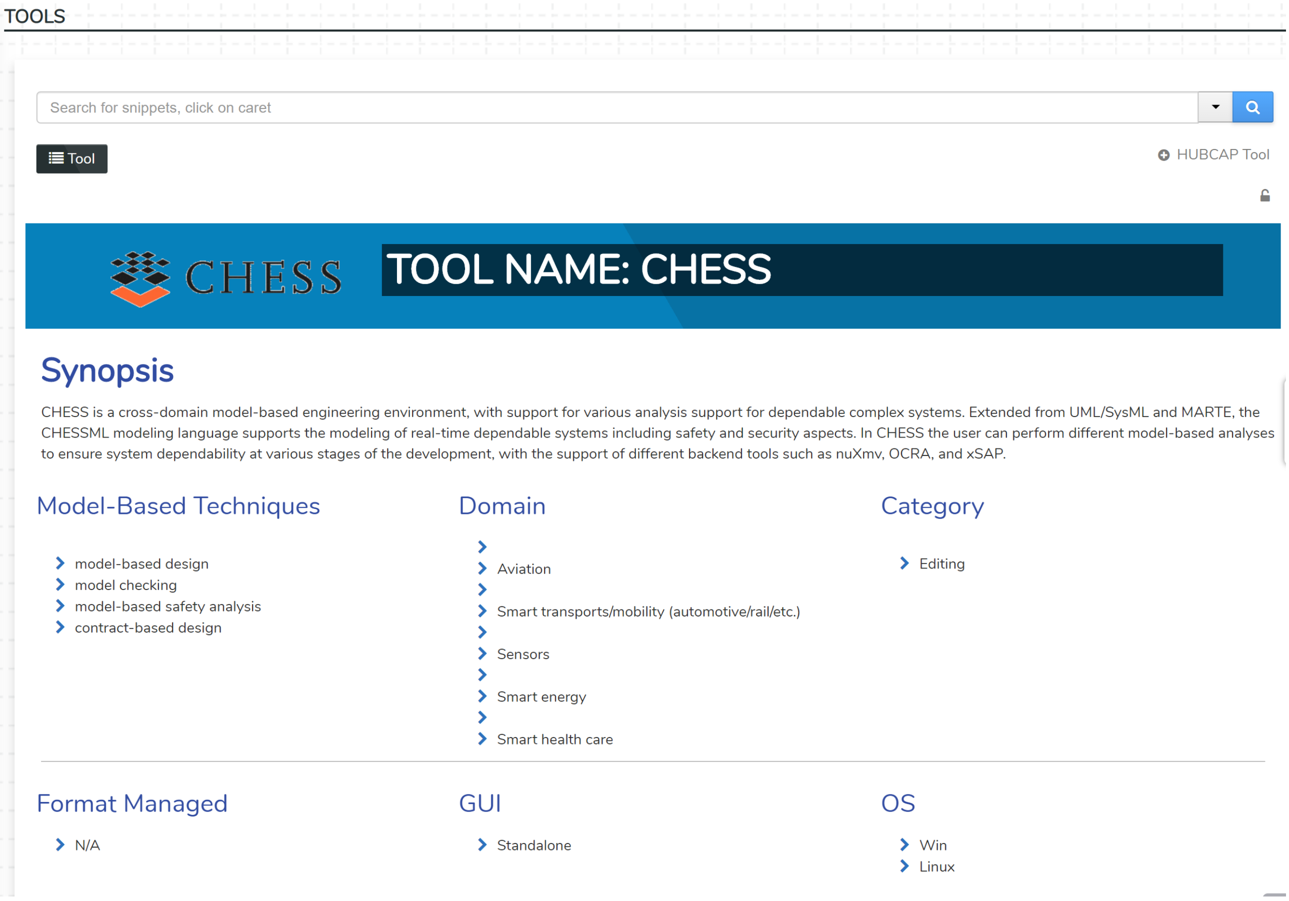


2.4. Chess

Provided by: **FBK**

Sandbox integration status: **Integrated**

CHESS is a cross-domain model-based engineering environment, with support for various analysis capabilities for dependable complex systems. Extended from UML/SysML and MARTE, the CHESSML modelling language supports the modelling of real-time dependable systems including safety and security aspects. In CHESS the user can perform different model-based analyses to ensure system dependability at various stages of the development, with the support of different backend tools such as nuXmv, OCRA, and xSAP.



2.5. AutoFOCUS3

Provided by: **FOR**
Sandbox integration status: **Integrated**

AutoFOCUS3 (AF3) is a model-based tool and research platform for safety-critical embedded systems. It builds on a generic Eclipse-based tooling framework, is open source (Apache 2.0 license), and is released biannually for the Windows, Linux, and macOS 64-bit platforms.

AutoFOCUS3 supports the design, development and validation of safety-critical embedded systems in many development phases, including architecture design, implementation, hardware/software integration, and safety argumentation based on formal models. The integration in the HUBCAP platform and sandbox focusses on the Design Space Exploration (DSE) capabilities of the tool to synthesize hardware/software integration models that allow for an efficient operation of the resulting system. Efficiency is defined by a user of the tool through the definition of objectives for the DSE.

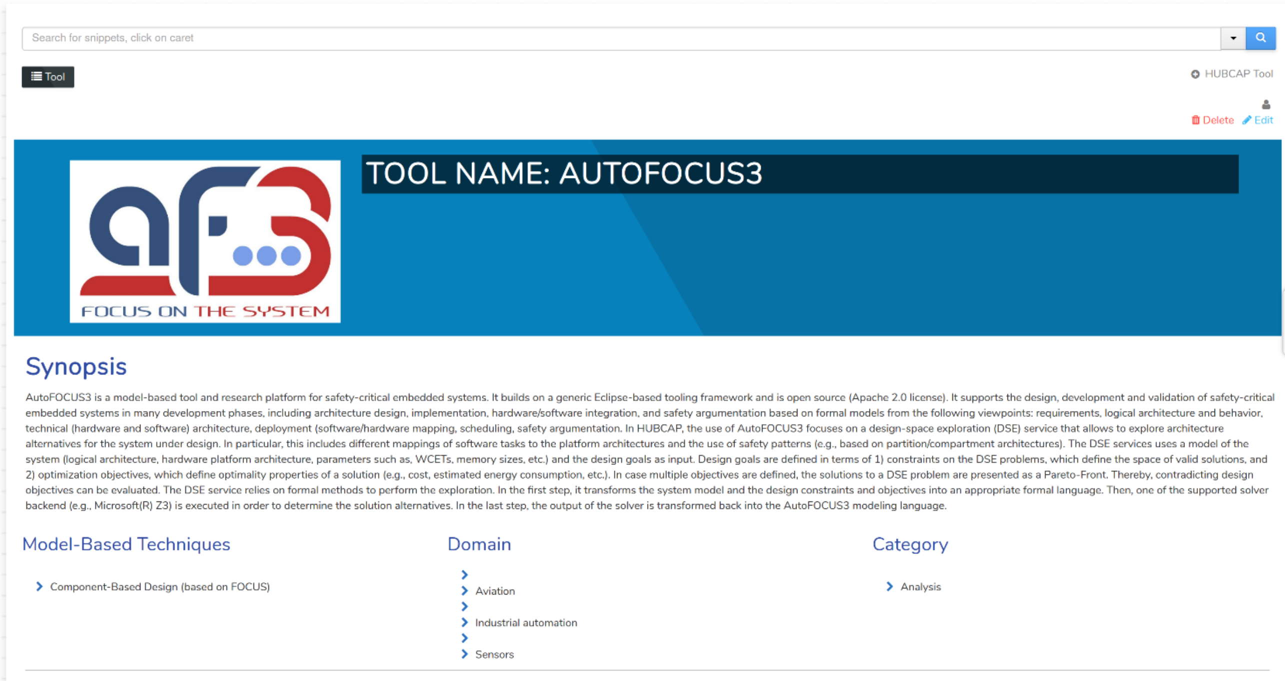


Figure 2: AutoFOCUS3 in the HUBCAP catalogue.

2.5.1. DSE Service

The DSE service is provided as a sandbox tool that can load AF3 models from the GUI. A Command Line Interface (CLI) allows the automatic import of these models from a sandbox model input folder, given the models are provided as raw files. The following description shows how to use the DSE service.

First, the AF3 sandbox has to be selected and one of its example models must be added from the Web GUI (see Figure 3). It is also possible to create a model from scratch using the AF3 modelling capabilities, but we will assume the most relevant case for HUBCAP starting from example models. For an in depth explanation of the modelling capabilities of AF3, refer to the online documentation.

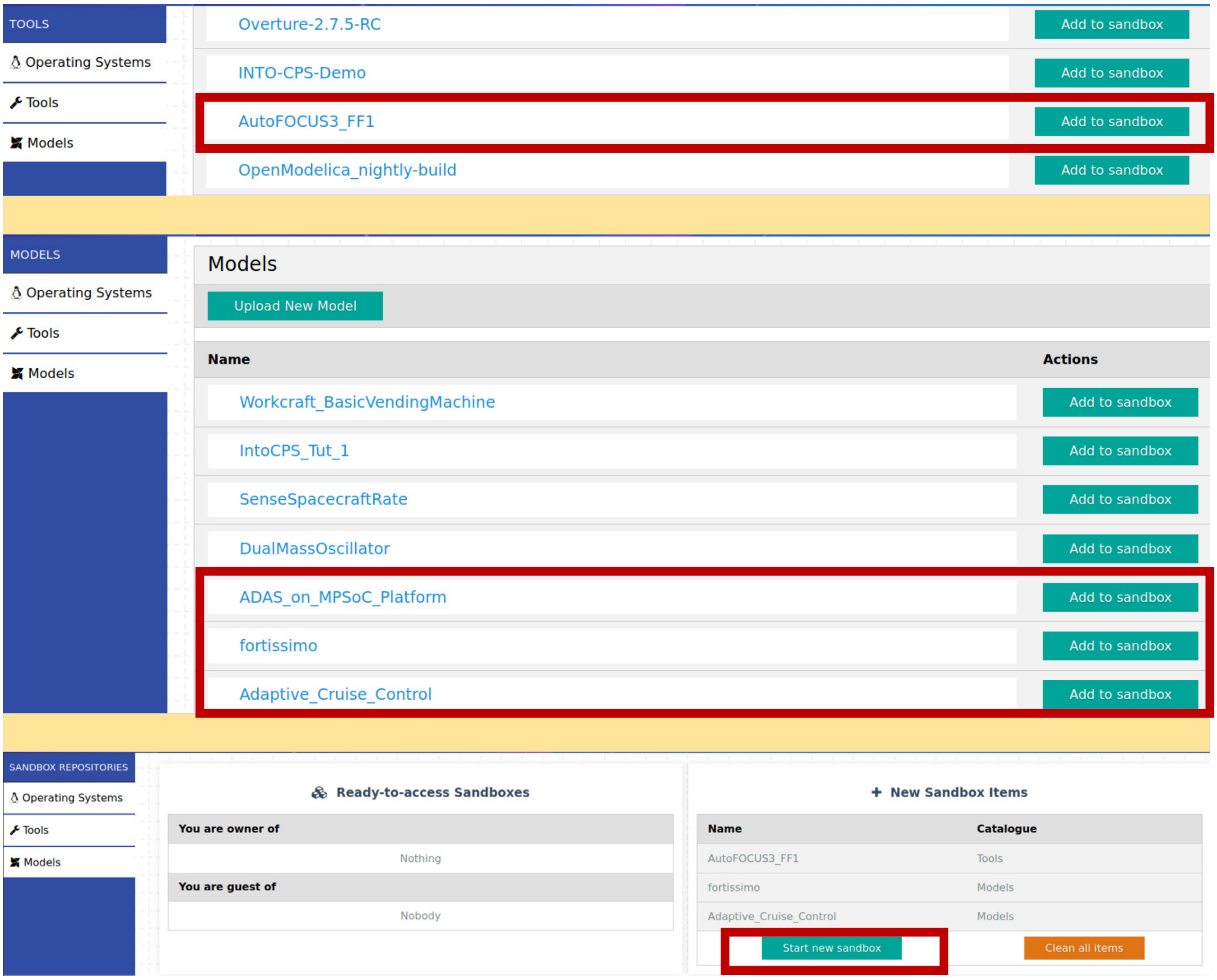


Figure 3: AutoFOCUS3 sandbox and models from the HUBCAP sandbox platform.

Once the sandbox is started, AF3 can be started by double-clicking on the AF3 icon on the desktop of the platform. By that, the AF3 Eclipse Rich Client Platform (RCP) is launched with the DSE perspective preloaded (see Figure 4).

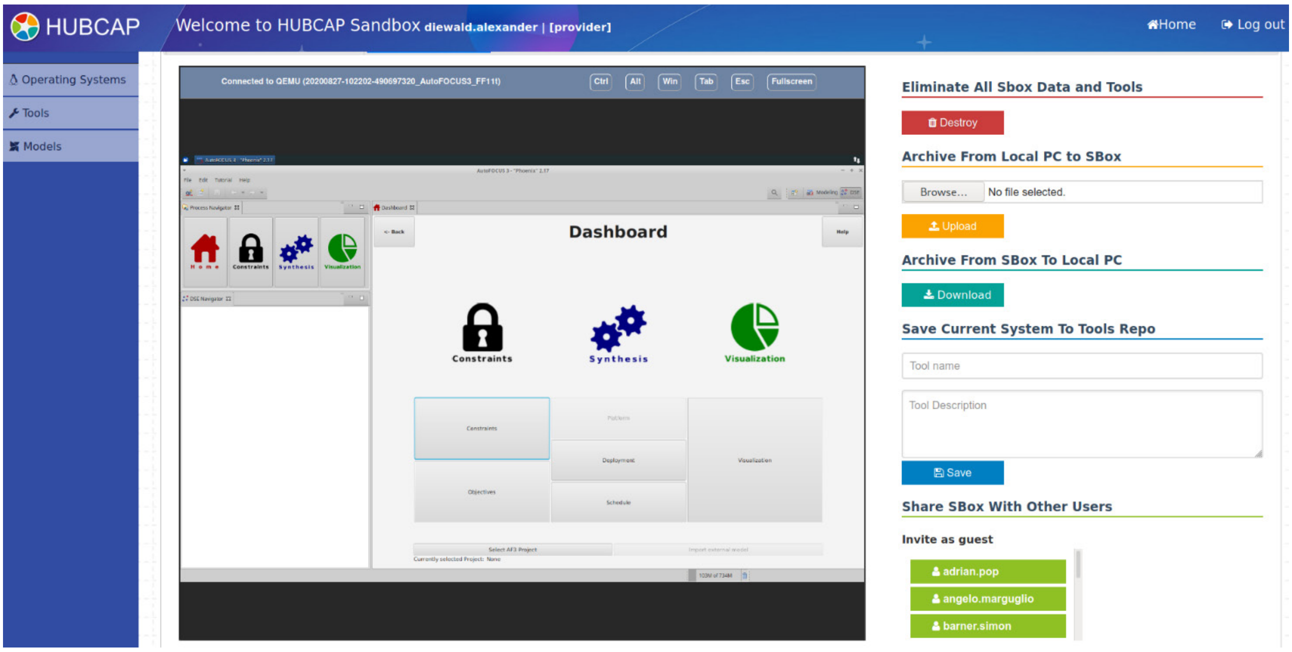


Figure 4: AutoFOCUS3 DSE service executed in the HUBCAP sandbox platform.

In order to load a model in the AF3 application, the import wizard for AF3 projects must be launched from the menu bar (see Figure 5). In the file selection dialog the desired example models from the catalogue or those uploaded by a user can be imported. Catalogue models configured with the sandbox reside in the folder */nfs/toolsdata/<username>* (Note: *<username>* must be substituted with the user name of the sandbox platform, not the UNIX user name of the sandbox).

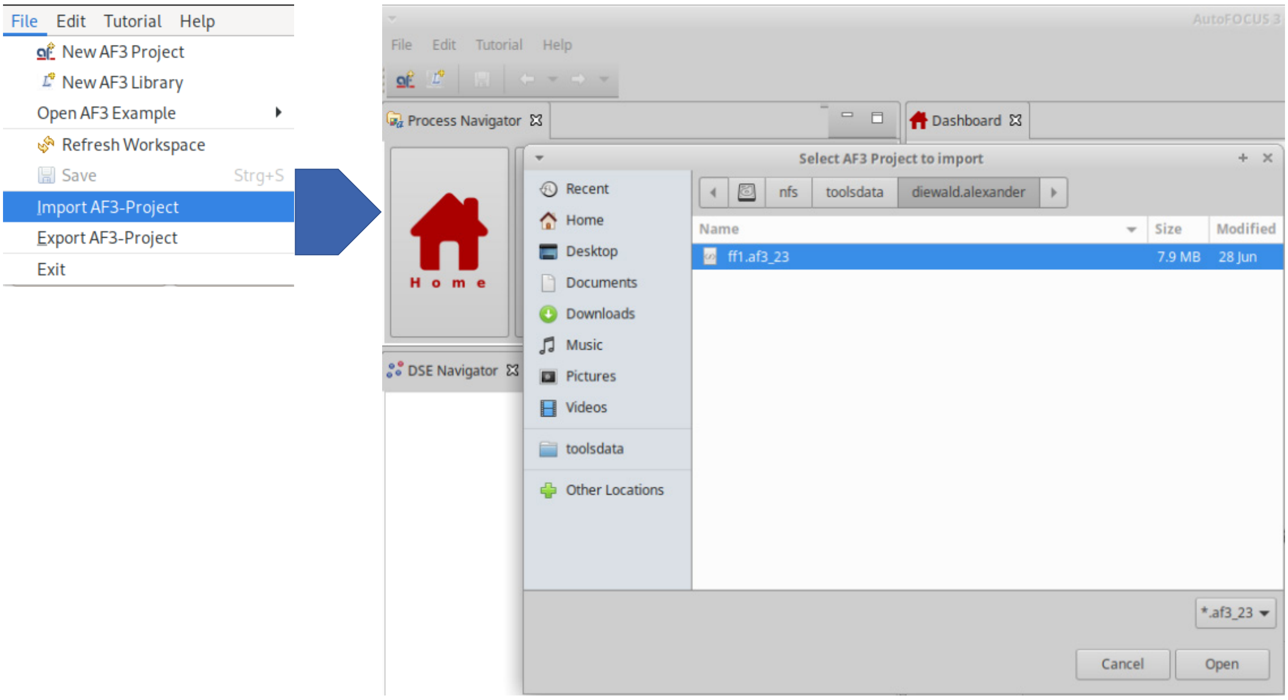


Figure 5: Importing an AF3 project.

After the model was imported into AF3, the contained DSE project must be imported to the DSE perspective (see Figure 6). This step is needed to support multiple DSE projects within an AF3 model. Therefore, the “Import from AF3” must be clicked to open the DSE import wizard. After selecting the imported AF3 model, its DSE project can be imported or a new DSE project can be created by selecting the required input artifacts.

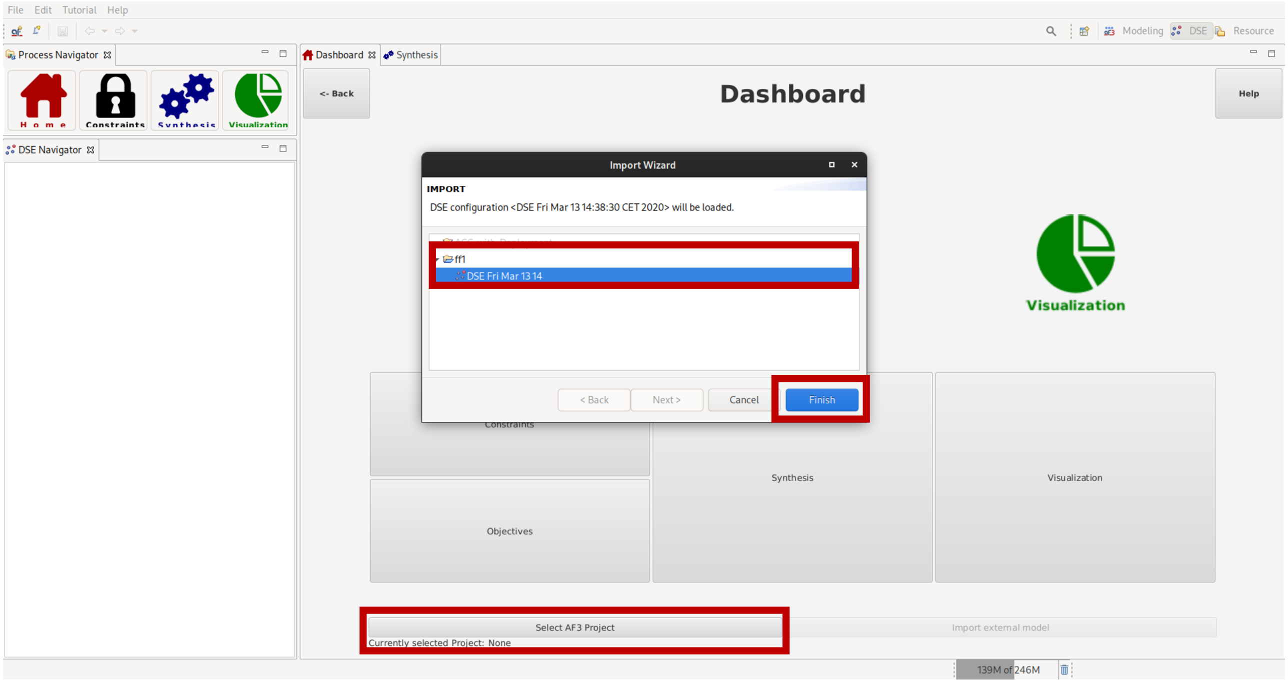


Figure 6: Importing a DSE project contained in an AF3 project.

Once the DSE project is imported, its content can be seen in the navigator on the left hand side (see Figure 7).

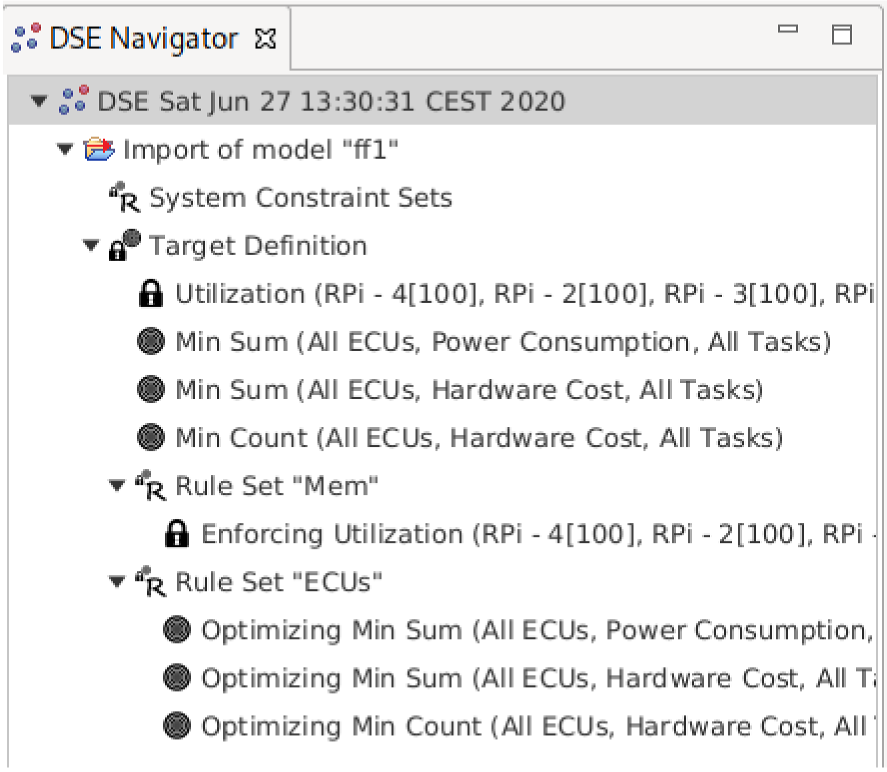
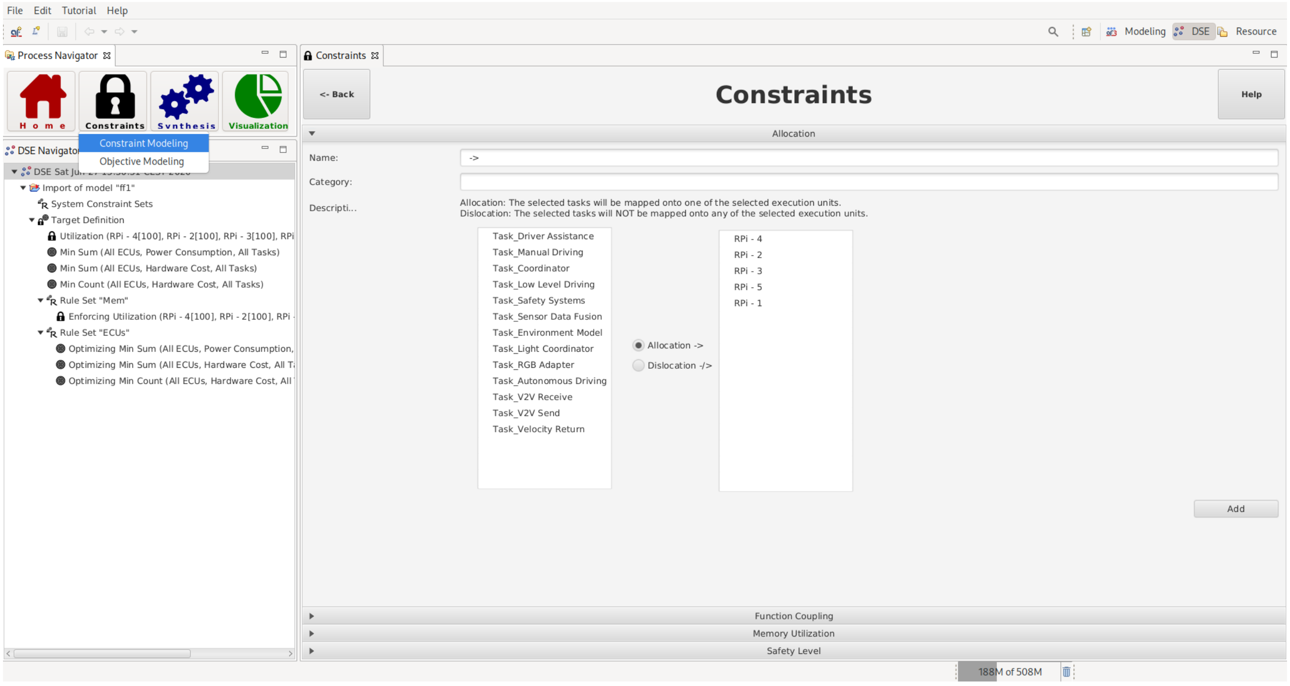


Figure 7: The content of a DSE project is shown in the DSE navigator.

In order to define the design space, constraints and objectives can be created by the user – however, the example projects already contain a set of objectives and constraints such that no editing is needed to execute a DSE. Therefore, the buttons “Constraints” or “Objectives” can be clicked in the dashboard or selected from the DSE process navigator (top left). The constraint and objective editing bases on a first-order language where the elements of an objective or constraint must be selected. For instance, an allocation constraint requires selecting the tasks and execution units.



- Defining RuleSets
- Launching a DSE run

Next, a synthesis (or exploration) can be executed. First, a synthesis type must be selected from the top of the view (see Figure 8).

Figure 8: Selecting a synthesis / an exploration.

Then, *RuleSets* can be defined by a user of the DSE. In the example models these are already pre-configured. *RuleSets* are containers that group constraints or objectives to group similar constraints and ease user interaction. In order to start a synthesis, the desired *RuleSets* must be selected from the tables on the right hand side (see Figure 9). The solver can be selected from the drop down box and configured by the “Configure” button in the green button shown in Figure 9.

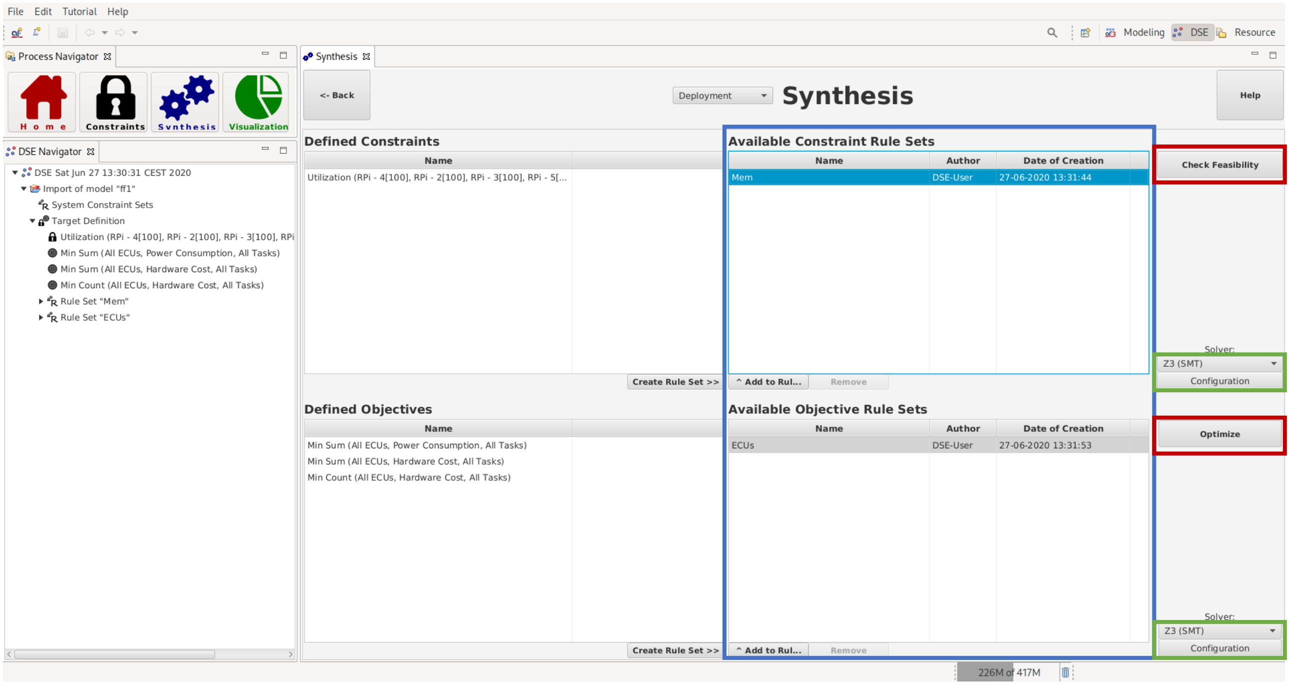


Figure 9: DSE synthesis view. Blue: Rul eSets to apply; Red: Start synthesis / exploration; Green: Configure solver.

An exploration is then started by clicking the buttons “Check Feasibility” or “Optimize”. The first button starts a feasibility check where only the selected *Rule Sets* of constraints are considered (see the blue boxes in Figure 9). By clicking “Optimize” an optimizing exploration is launched that additionally considers selected *Rule Sets* of objectives that represent design goals. A progress bar informs the user about the status of the exploration.

When the exploration is finished, the results can be investigated by clicking the “Visualization” button in the Process Navigator in the top left corner. By default, a spiderchart representation of the objective evaluation results is shown (see Figure 10) that is empty if only a constraint check was launched. Moreover, a table representation of the results is available that also allows users to transform the results found by an exploration into model artifacts that can be re-exported to the container AF3 project (see Figure 11). Alternatively, a solution can also be *selected* such that it is used as a basis for subsequent exploration step such as scheduling. Finally, a GANNT viewer is available that allows to view schedules from a corresponding synthesis.

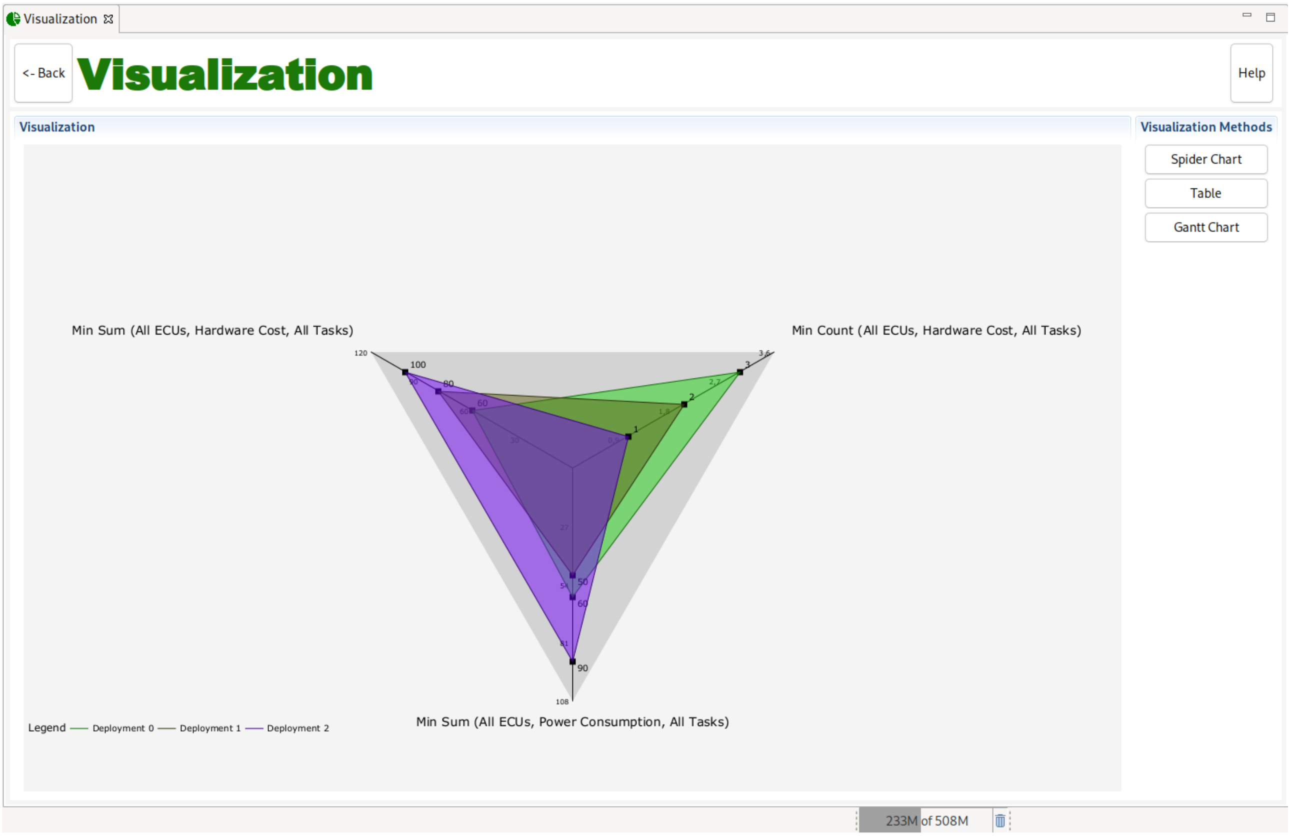


Figure 10: Spiderchart of the solution metrics (objectives).

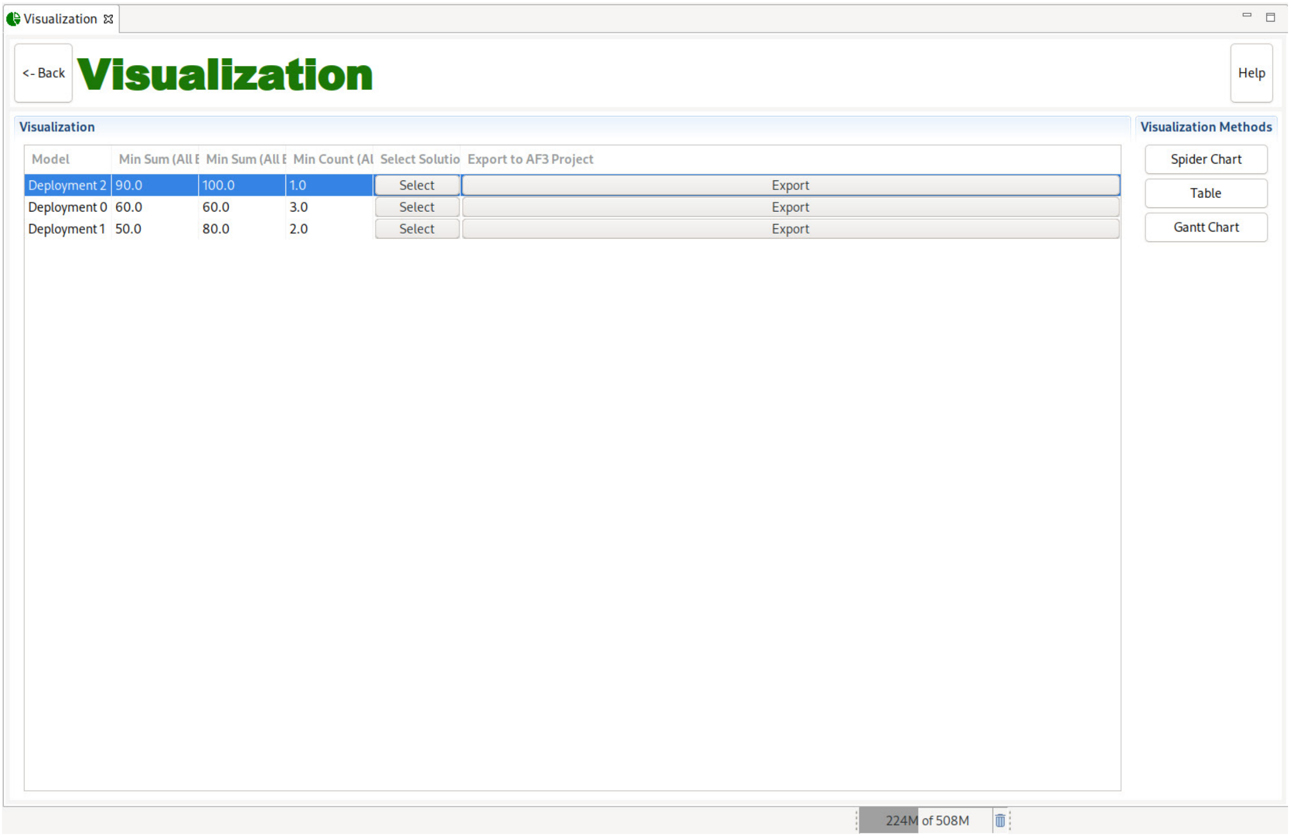


Figure 11: Table of the produced solutions. The solutions can be exported to the container AF3 project using the "Export" button.

2.5.2. HUBCAP Sandbox Integration

A Command Line Interface extension was added to AF3 in order to provide import capabilities for AF3 and DSE projects, the GUI of the DSE service, and to be able to launch a DSE from the command line for eased tool integration. Some of these extensions are also used in task T3.5.

The CLI extension is realized as an application kernel service to which each plugin may contribute a command line switch and a handler that processes the argument of a switch. Thus, the AF3 CLI is modular and CLI handlers can be implemented according to a plugin's dependencies. In Table 1, the CLI switches that are implemented at the time of writing the report are listed. Each of the switches are processed in the order they are provided to the application.

Switch	Argument Format	Description
--import	Path	Imports a AF3 project, wildcards and batch importing is supported.
--show-dse	N/A	Starts AF3 with the DSE perspective.

--load-dse	*AF3 Prj name/DSE name*	Starts AF3 with the DSE perspective and preloads the given DSE project.
--exec-dse	*AF3 Prj name/DSE name/Synth Type*	Executes the given DSE considering all present *RuleSets*. As synthesis types “Deployment” and “Scheduling” are supported.
--dump-smtlib	*AF3 Prj name/DSE name/Synth Type*	Dumps the given DSE specification as a file in the SMT-Lib v2.5 format. Only constraint satisfaction problems are supported, no optimization.
--validate-dse	*AF3 Prj name/DSE name*	Validates the results of a DSE run.
--shutdown	N/A	Terminates the AF3 RCP when evaluating the switch.
--gen-dse-testcases	N/A	Generates a set of DSE test cases in the workspace.

Table 1: List of implemented CLI switches.

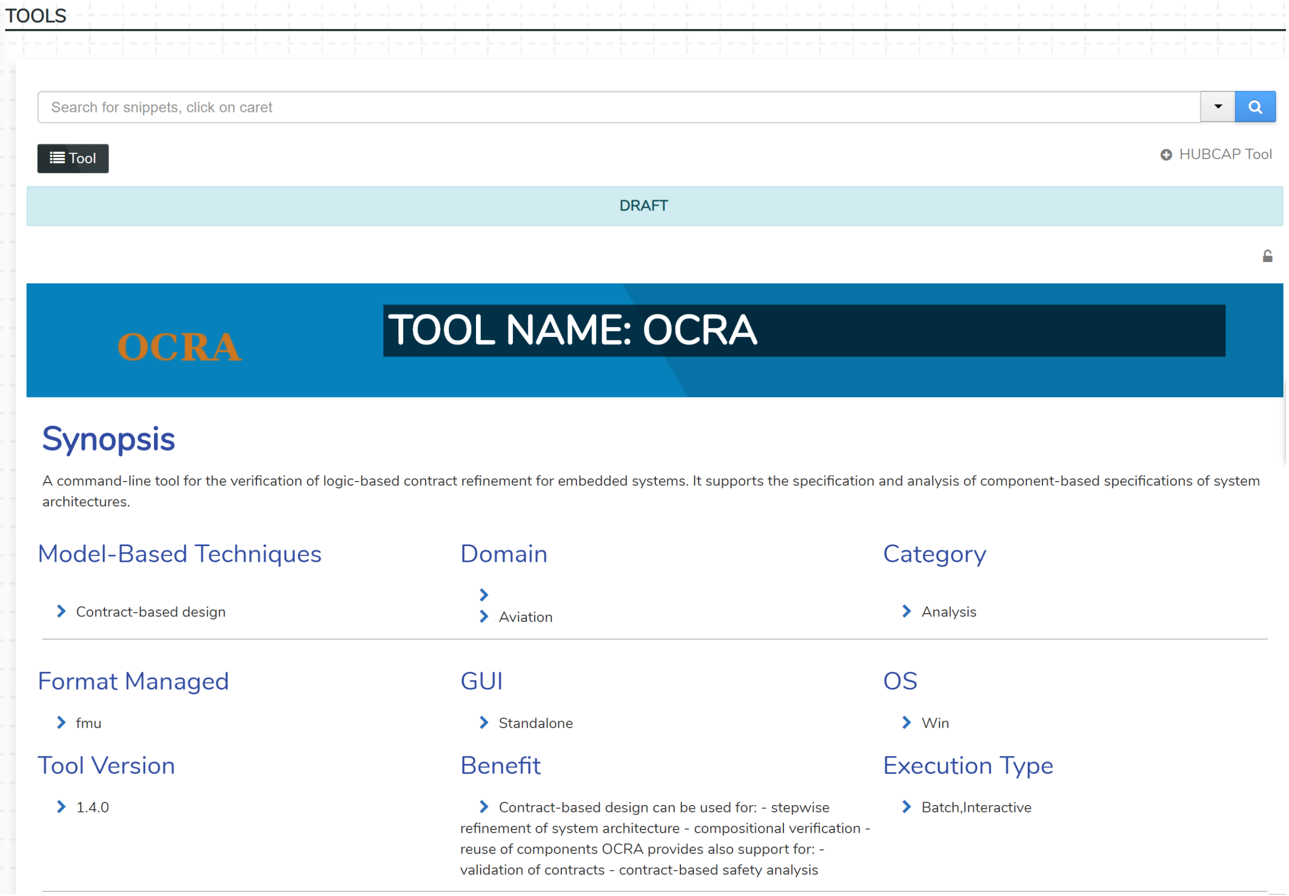
Using the command line switches, a convenient solution could be provided for users of the DSE services. The users are presented with the DSE perspective upfront but can still switch to the modelling perspective if needed, for instance to inspect results of the exploration.

The sandbox itself has been found to provide sufficient performance for executing the application and its integrated DSE solver. There were no noticeable latencies when using the HUBCAP sandbox such that there was no noticeable difference compared to using the application on a local machine.

2.6. OCRA

Provided by: **FBK**
Sandbox integration status: **Integrated**

OCRA is a command-line tool for the verification of logic-based contract refinement for embedded systems. It supports the specification and analysis of component-based specifications of system architectures. Components are enriched with contracts specified in discrete or hybrid linear-time temporal logics. The analysis supports the verification of the contract refinement along the architectural decomposition, the compositional verification of component behaviours, the validation of contracts via satisfiability and parameter synthesis, the generation of fault trees based on the contract specification, the specification and analysis of parametrized architectures.



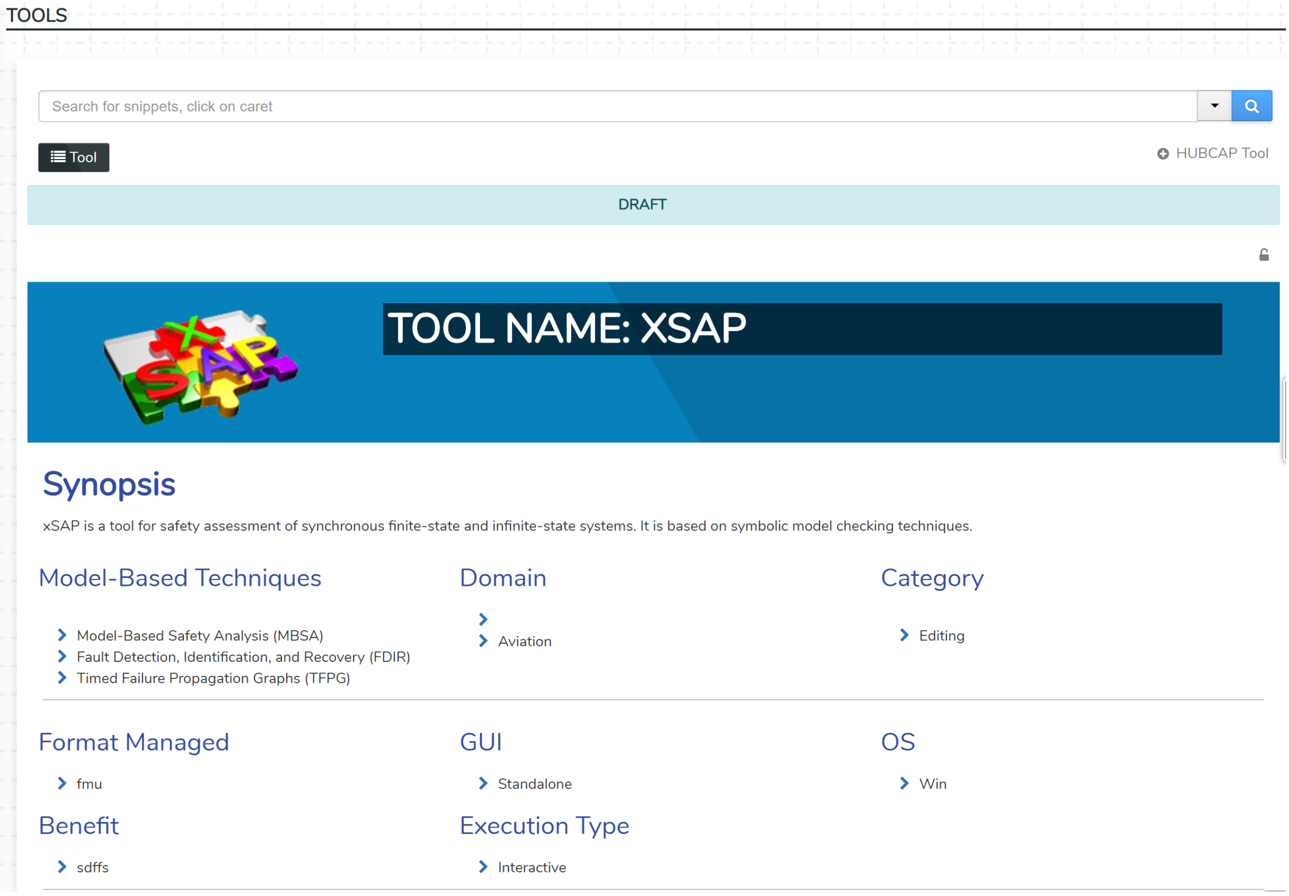
2.7. xSAP

Provided by: **FBK**

Sandbox integration status: **Integrated**

xSAP is a tool for safety assessment of synchronous finite-state and infinite-state systems. It is based on symbolic model checking techniques. xSAP provides the following main capabilities:

- Library-based specification of faults, fault effects, and fault dynamics
- Automatic model-extension with fault specifications
- Fault Tree Analysis (FTA) and generation of Minimal Cut Sets (MCS) for dynamic systems, for both the monotonic and non-monotonic case
- Failure Modes and Effects Analysis (FMEA)
- Fault propagation analysis based on Timed Failure Propagation Graphs (TFPG)
- Common Cause Analysis (CCA)

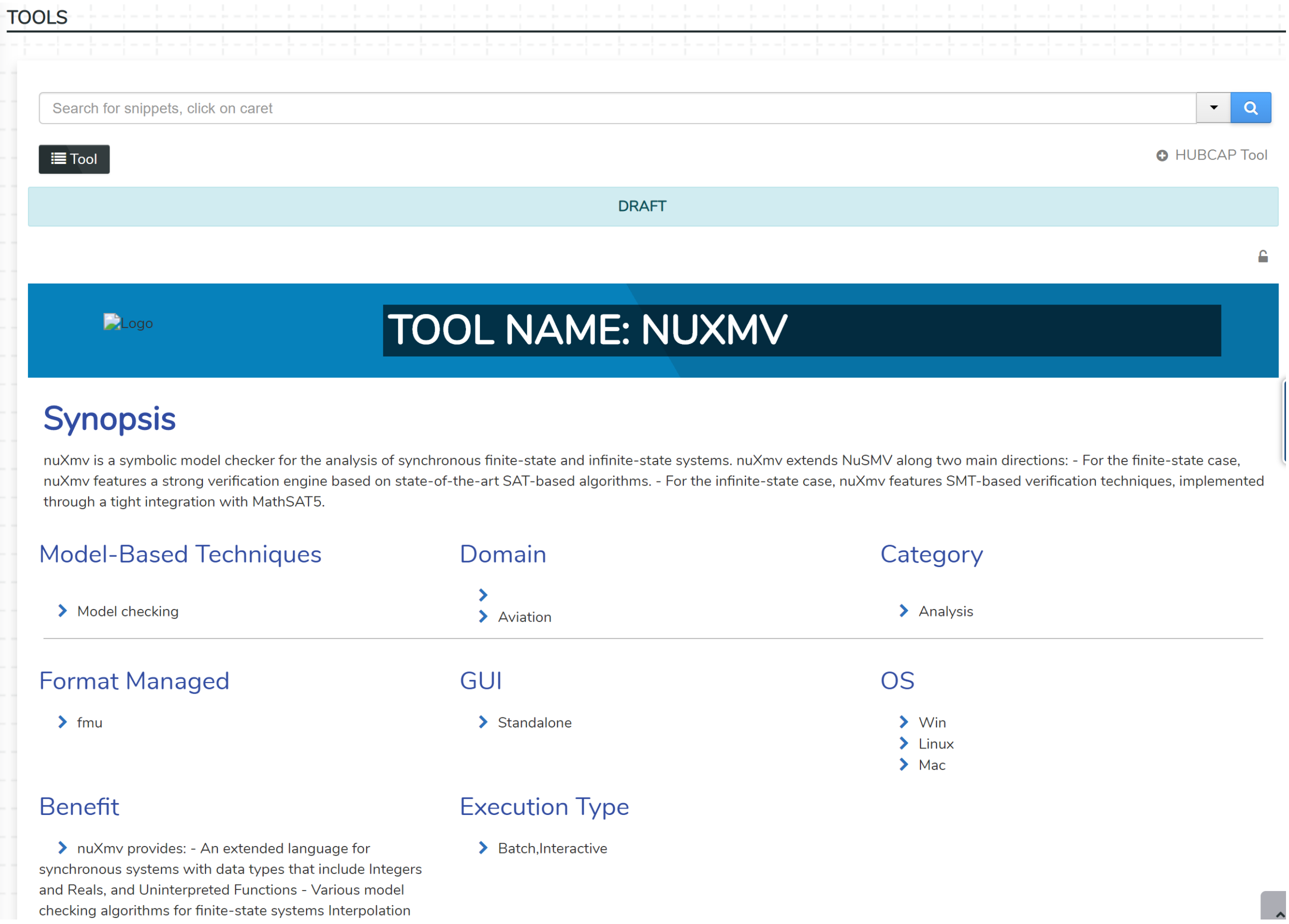


2.8. nuXmv

Provided by: **FBK**

Sandbox integration status: **Integrated**

nuXmv is a symbolic model checker for the analysis of synchronous finite-state and infinite-state systems. nuXmv extends NuSMV along two main directions: - For the finite-state case, nuXmv features a strong verification engine based on state-of-the-art SAT-based algorithms. - For the infinite-state case, nuXmv features SMT-based verification techniques, implemented through a tight integration with the MathSAT5 SMT solver.



3. The Tools from SMEs partners

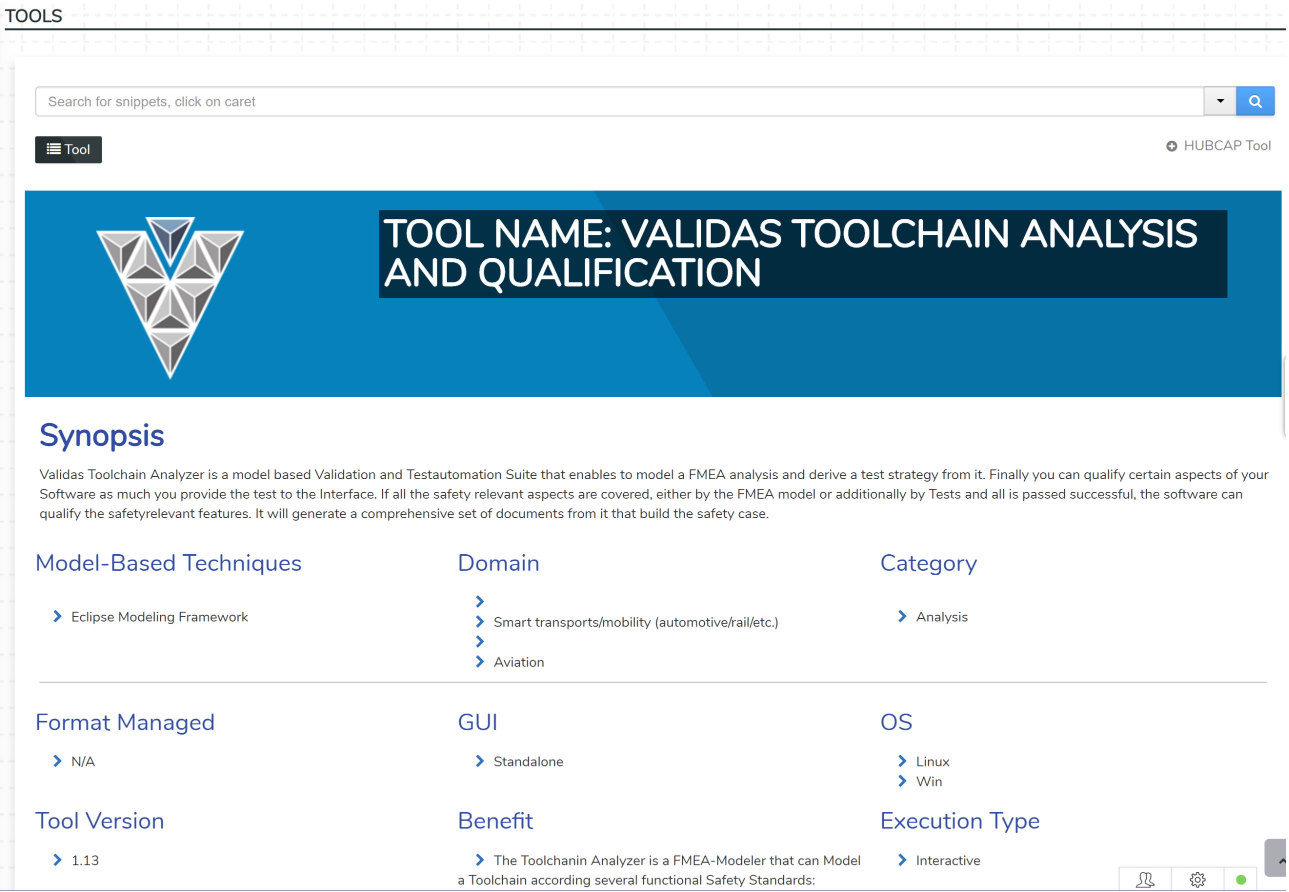
These tools are provided by the seed SMEs and they are more detailed in the Wp3 deliverables.

3.1. Validas Toolchain Analysis and Qualification

Provided by: **VAL**

Sandbox integration status: **Integrated**

Validas Toolchain Analyzer is a model based Validation and Test automation Suite that enables to model an FMEA analysis and derive a test strategy from it. Finally you can qualify certain aspects of your Software as much you provide the test to the Interface. If all the safety relevant aspects are covered, either by the FMEA model or additionally by Tests and all are passed successfully, the software can qualify the safety relevant features. It will generate a comprehensive set of documents from it that build the safety case.

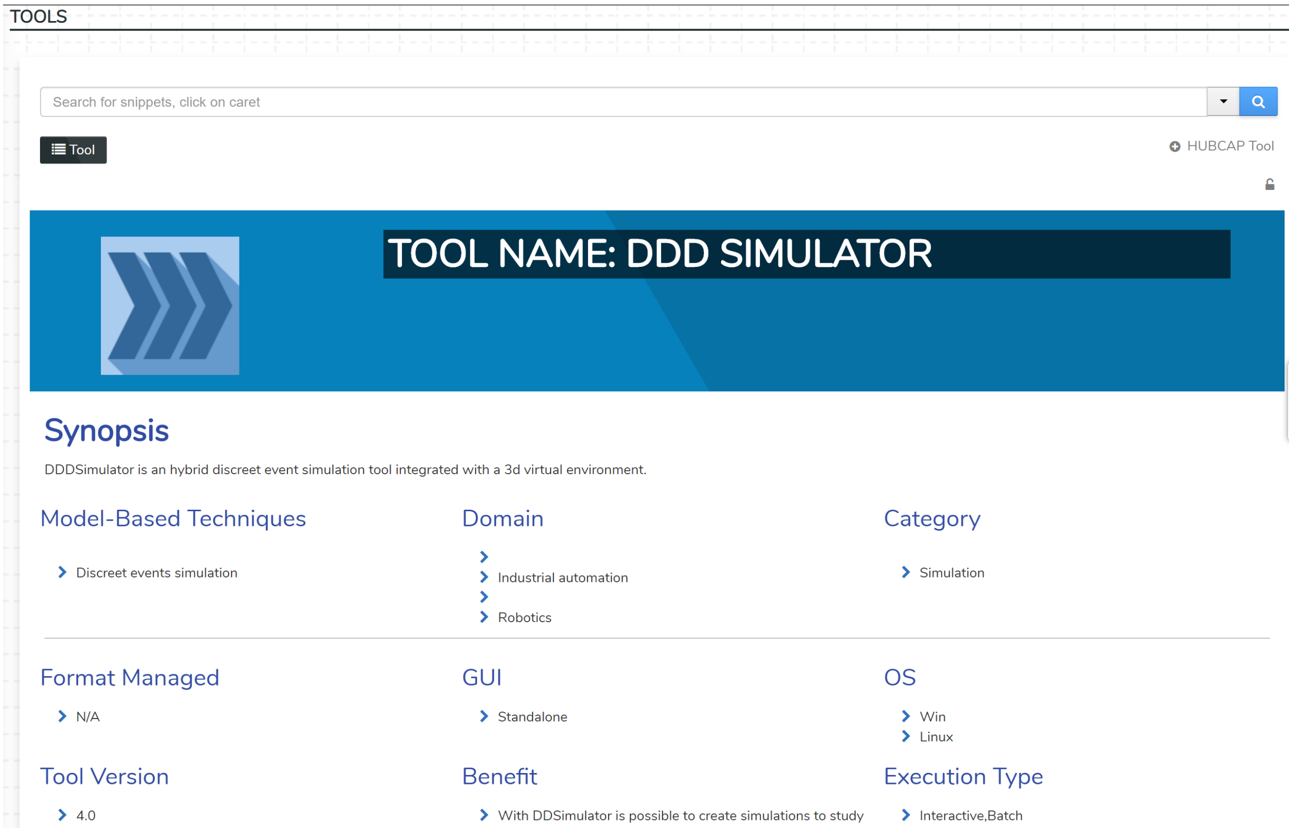


3.2. DDD Simulator

Provided by: **TTS**

Sandbox integration status: **Integrated**

DDDSimulator is a hybrid discrete event simulation tool integrated with a 3d virtual environment.

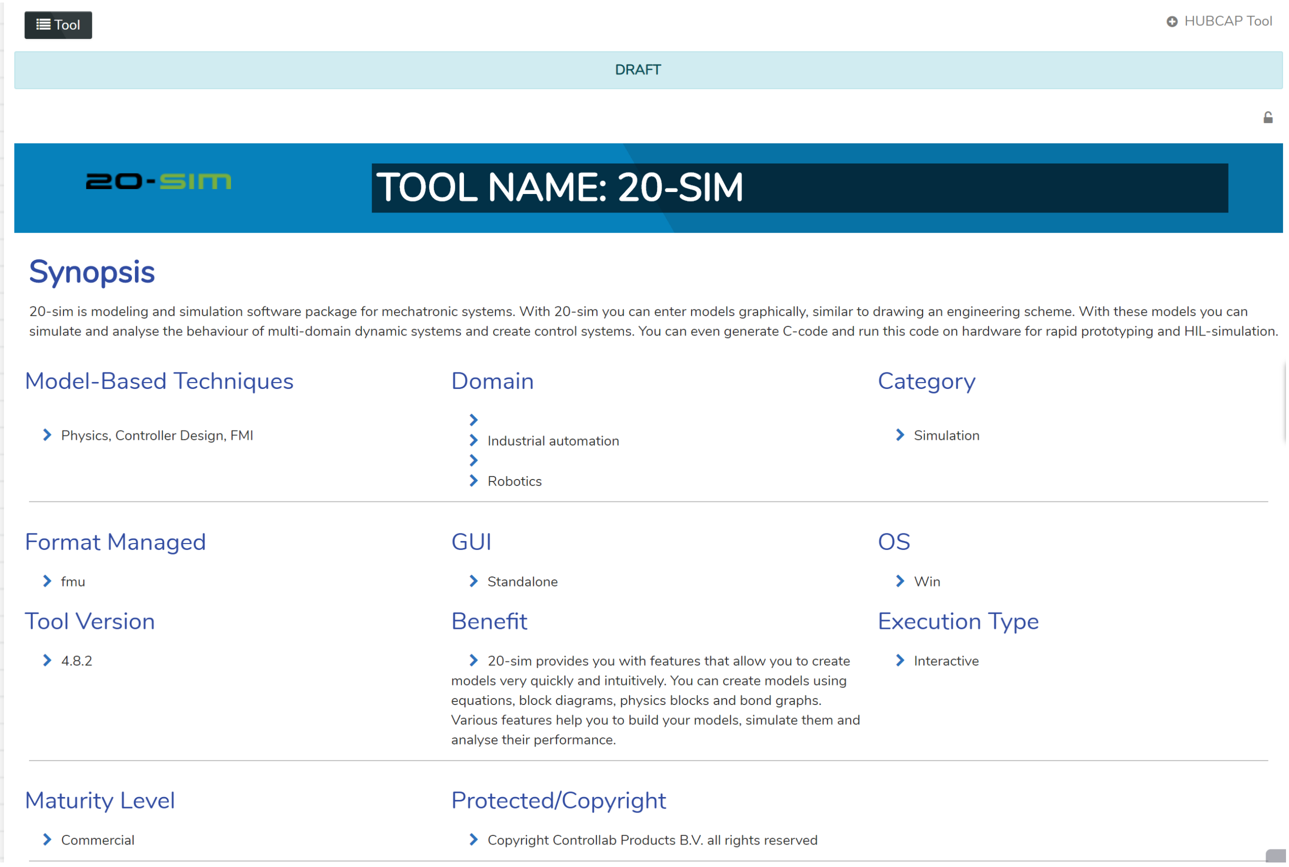


3.3. 20-sim

Provided by: **CLP**

Sandbox integration status: **Integrated**, Windows sandbox has performance issues, investigation in progress.

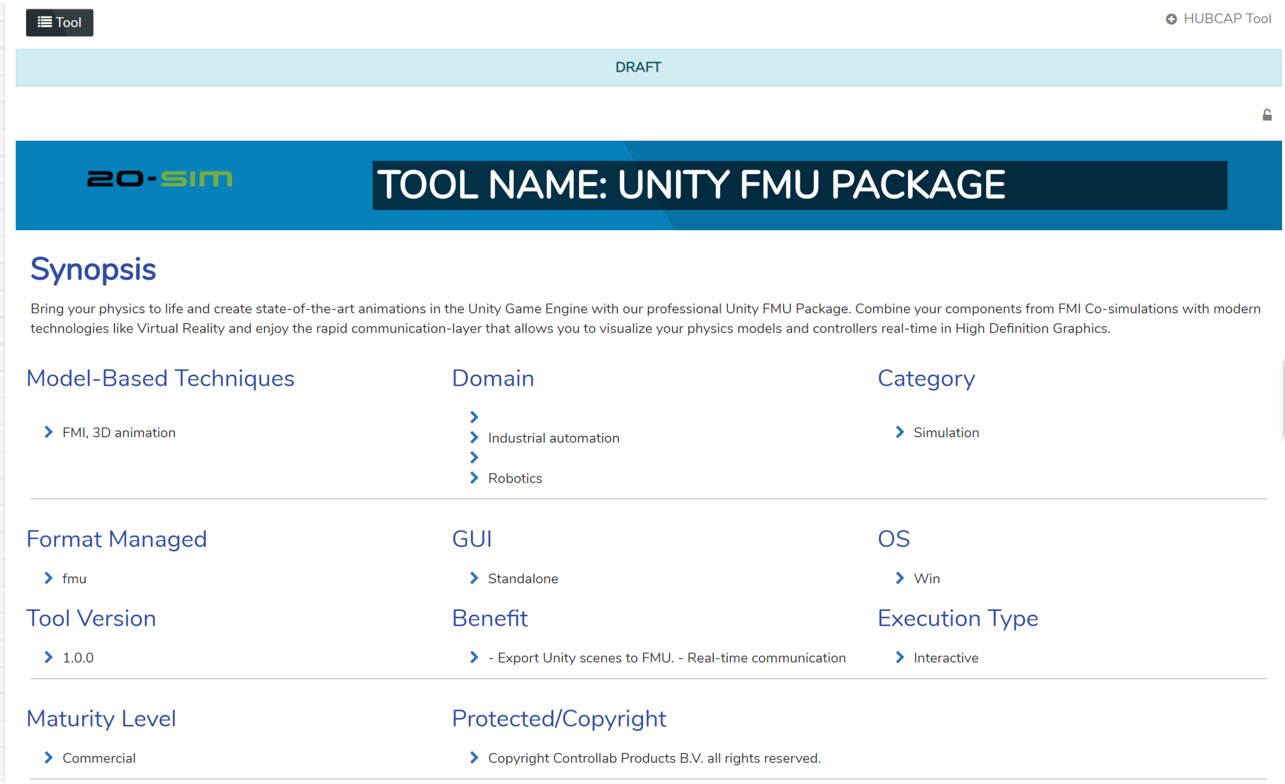
20-sim is a modelling and simulation software package for mechatronic systems. With 20-sim the user can enter models graphically, similar to drawing an engineering scheme. With these models one can simulate and analyse the behaviour of multi-domain dynamic systems and create control systems. The user can even generate C-code and run this code on hardware for rapid prototyping and HIL-simulation.



3.4. Unity FMU Package

Provided by: **CLP**

Sandbox integration status: **Integrated**, Windows sandbox has performance issues, investigation in progress.

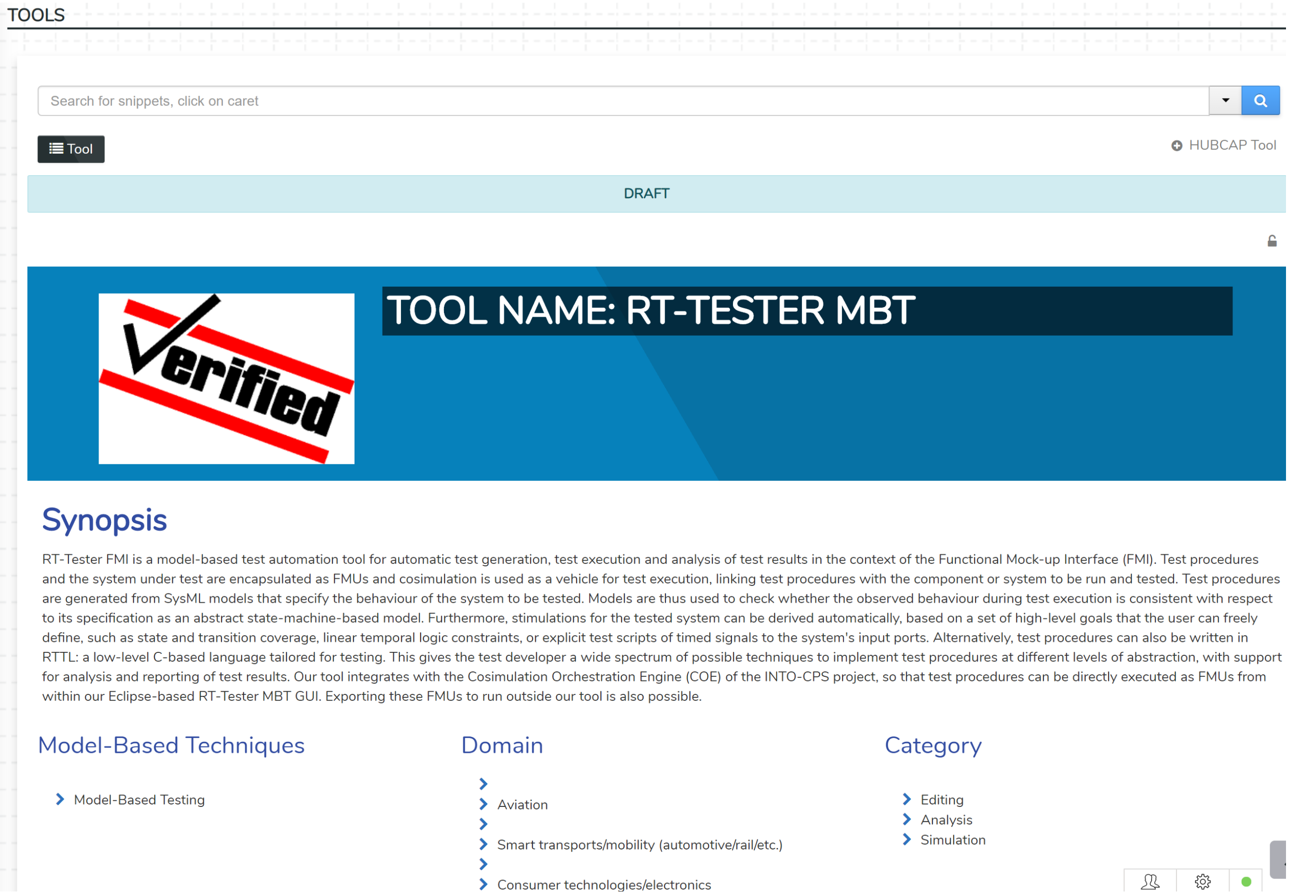


3.5. RT-Tester

Provided by: **VSI**

Sandbox integration status: **Integrated**

RT-Tester FMI is a model-based test automation tool for automatic test generation, test execution and analysis of test results in the context of the Functional Mock-up Interface (FMI). Test procedures and the system under test are encapsulated as FMUs and cosimulation is used as a vehicle for test execution, linking test procedures with the component or system to be run and tested. Test procedures are generated from SysML models that specify the behaviour of the system to be tested. Models are thus used to check whether the observed behaviour during test execution is consistent with respect to its specification as an abstract state-machine-based model. Furthermore, stimulations for the tested system can be derived automatically, based on a set of high-level goals that the user can freely define, such as state and transition coverage, linear temporal logic constraints, or explicit test scripts of timed signals to the system's input ports. Alternatively, test procedures can also be written in RTTL: a low-level C-based language tailored for testing. This gives the test developer a wide spectrum of possible techniques to implement test procedures at different levels of abstraction, with support for analysis and reporting of test results. Our tool integrates with the Co-simulation Orchestration Engine (COE) of the INTO-CPS project, so that test procedures can be directly executed as FMUs from within our Eclipse-based RT-Tester MBT GUI. Exporting these FMUs to run outside our tool is also possible.



4. New Case Study Models

In this section we briefly introduce the new models added to the platform after D6.1 was delivered.

4.1. AD/ADAS on MPSoC Platform

The model represents a set of software-defined AD/ADAS functions that are deployed to MPSoC architecture integrating both general-purpose microprocessor cores and a GPU. It originates from the Industrial Challenge that accompanies the yearly WATERS workshop at the ECRTS conference. The model at hand results from importing the AMALTHEA-model of the 2019 challenge with minimal manual adaptations. The modelled hardware platform is a Tegra X2 platform that consists of a quad-core and a dual-core ARM Cortex CPU supported by a GPU-accelerator. The software architecture features image processing, lane detection, and route planning components. The component and the task architecture also model the read and write tasks that shall be deployed onto a DMA unit. By that, a read-execute-write execution schema is implemented in the model. In contrast to the original challenge, we use AutoFOCUS3's DSE to provide a valid deployment that satisfies the provided design constraints, rather than synthesizing an execution schedule for the platform.

MODELS CATALOGUE

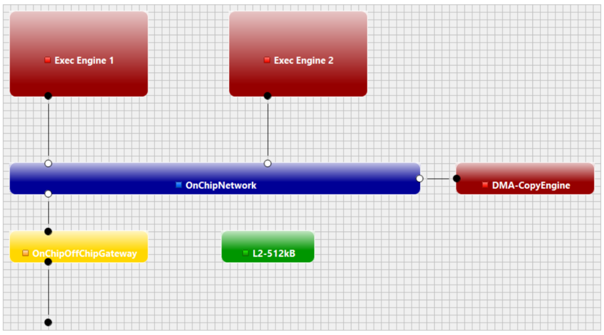
Search for snippets, click on caret

Model

HUBCAP Model

AD/ADAS on MPSoC Platform

The model represents a set of software-defined AD/ADAS functions that are deployed to MPSoC architecture integrating both general-purpose microprocessor cores and a GPU. It originates from the Industrial Challenge that accompanies the yearly WATERS workshop at the ECRTS conference. The model at hand results from importing the AMALTHEA-model of the 2019 challenge with minimal manual adaptations. The modelled hardware platform is a Tegra X2 platform that consists of a quad-core and a dual-core ARM Cortex CPU supported by a GPU-accellerator. The software architecture features image processing, lane detection, and route planning components. The component and the task architecture also model the read and write tasks that shall be deployed onto a DMA unit. By that, a read- execute-write execution schema is implemted in the model. In contrast to the original challenge, we use AutoFOCUS3's DSE to provide a valid deployment that satisfies the provided design contraints, rather than synthesizing an execution schedule for the platform.



Model-Based Techniques

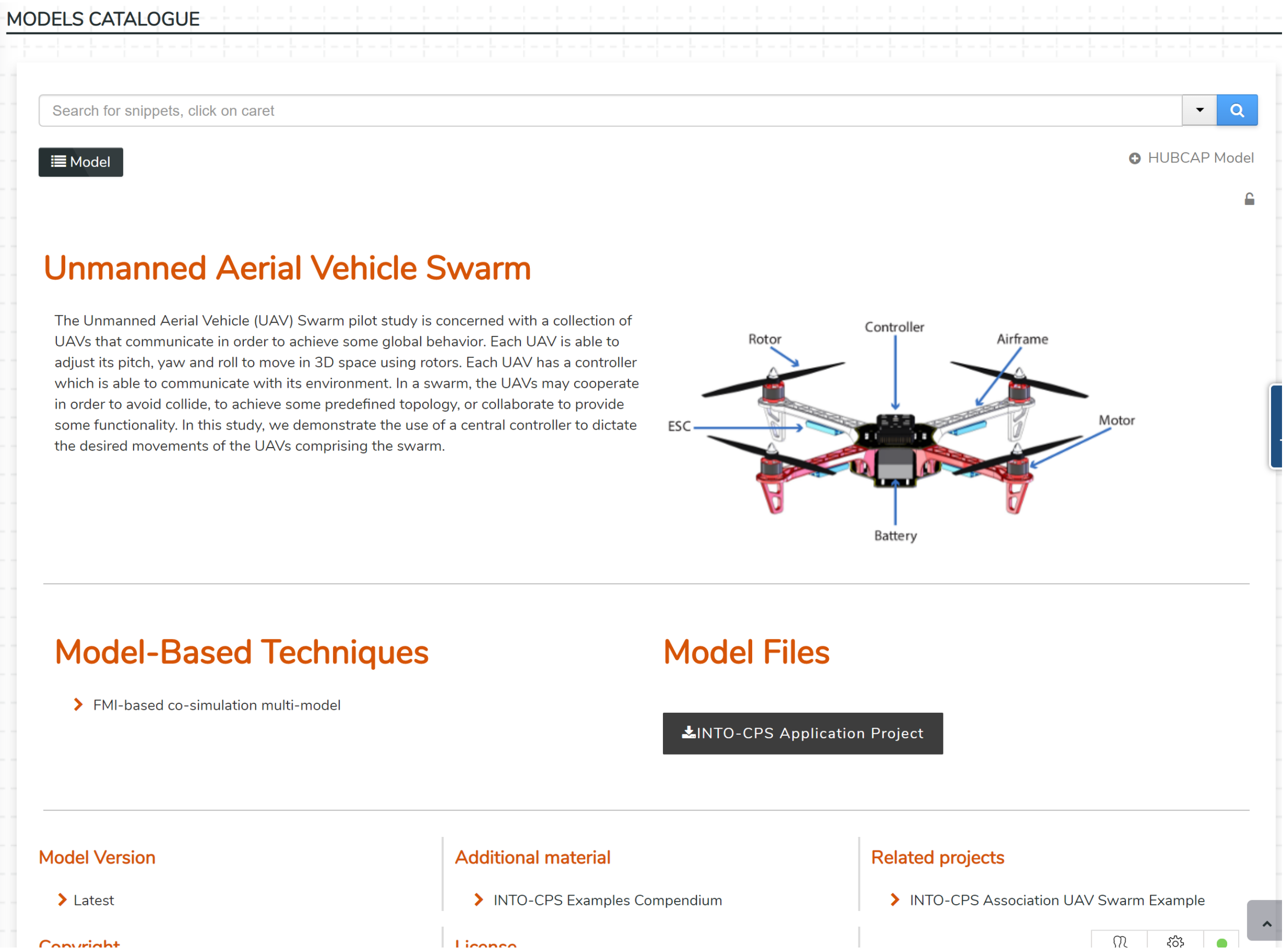
- Component-Based Design

Model Files

AD_ADAS_MPSoC

4.2 Unmanned Aerial Vehicle Swarm

The Unmanned Aerial Vehicle (UAV) Swarm pilot study is concerned with a collection of UAVs that communicate in order to achieve some global behaviour. Each UAV is able to adjust its pitch, yaw and roll to move in 3D space using rotors. Each UAV has a controller which is able to communicate with its environment. In a swarm, the UAVs may cooperate in order to avoid collide, to achieve some predefined topology, or collaborate to provide some functionality. In this study, we demonstrate the use of a central controller to dictate the desired movements of the UAVs comprising the swarm.



4.3 Train Gate Controller

The model represents the train crossing problem. The model is composed by a system component, a Train component, a Gate component and a Controller component. The model's behaviour is the following: When a train reaches the gate the controller raises the gate. When the gate is open the train passes through the gate. When the train passed the gate the controller brings down the gate. This model is encoded in hybrid time giving the possibility to consider continuous time events.

MODELS CATALOGUE

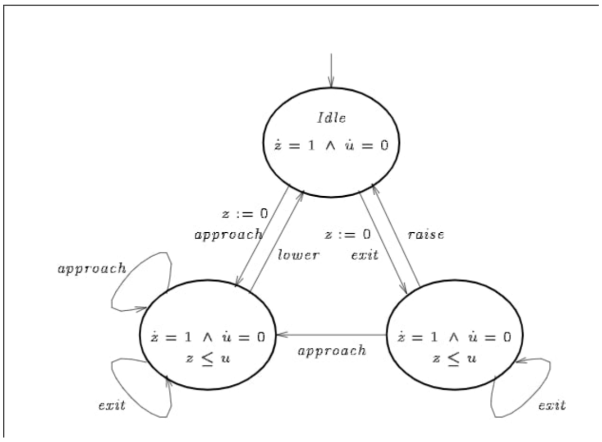
Search for snippets, click on caret

Model

HUBCAP Model

Train Gate Controller

The model represents the train crossing problem. The model is composed by a system component, a Train component, a Gate component and a Controller component. The model's behavior is the following: When a train reaches the gate the controller raises the gate. When the gate is open the train passes through the gate. When the train passed the gate the controller brings down the gate. This model is encoded in hybrid time giving the possibility to consider continuous time events.



Model-Based Techniques

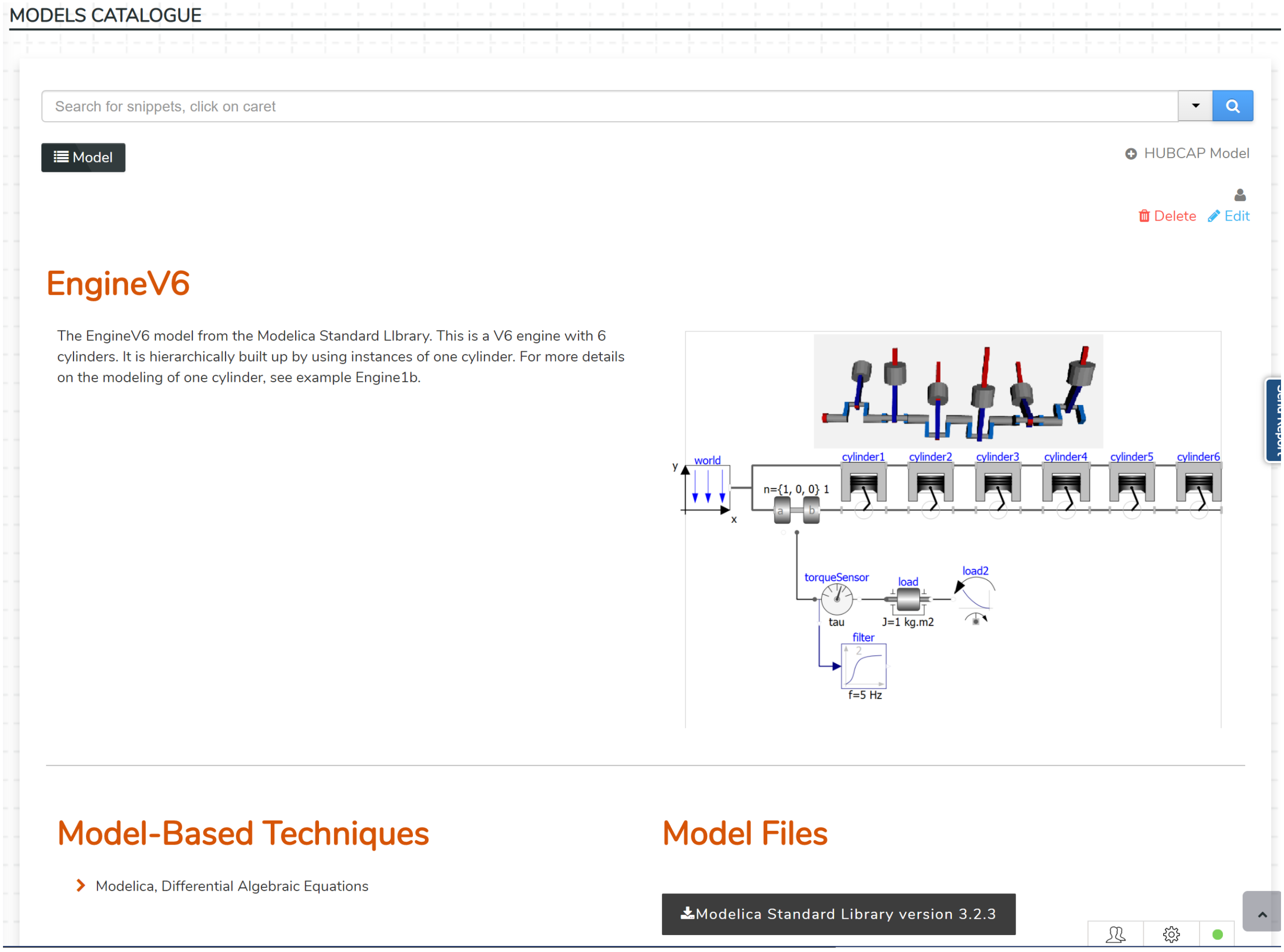
- Contract-based design

Model Files

Models and OCRA commands

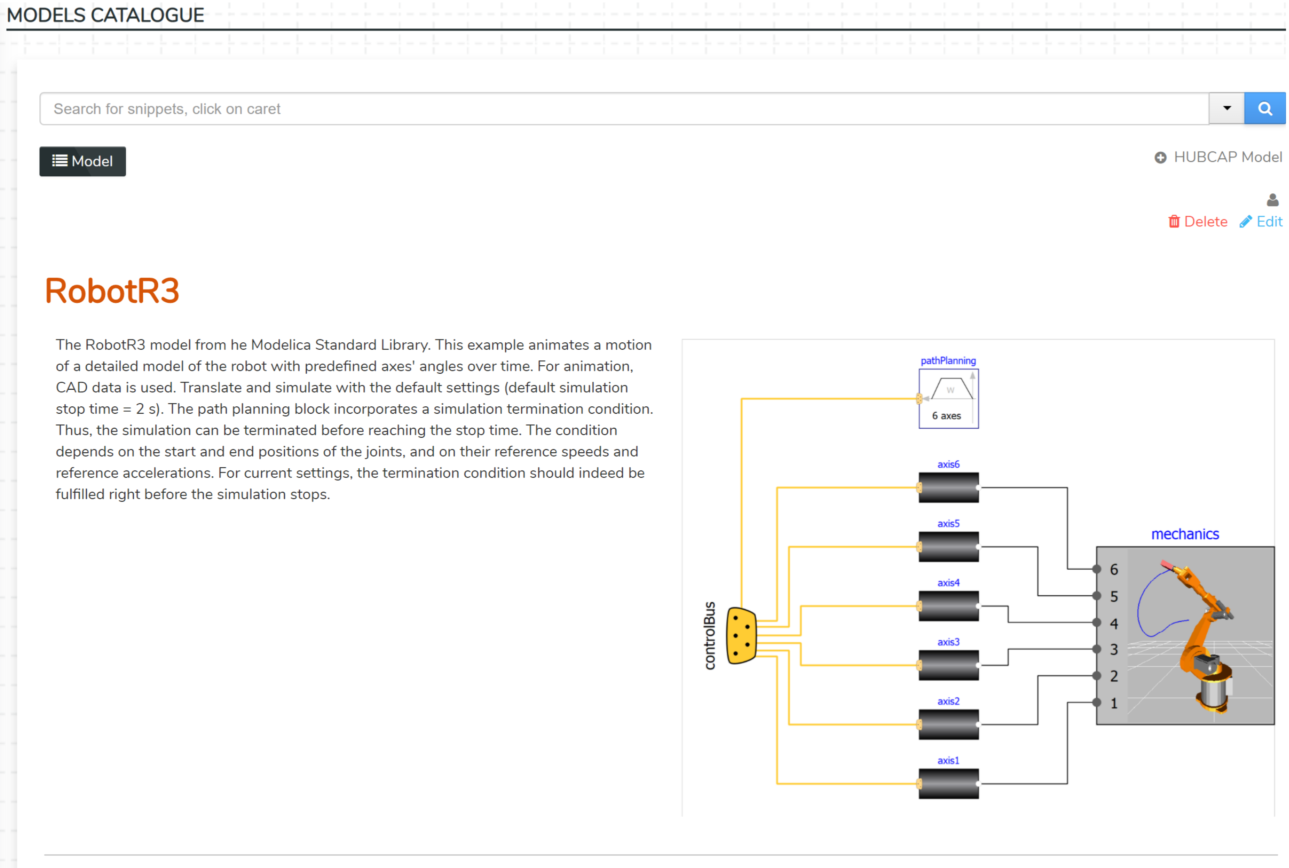
4.4 EngineV6

The EngineV6 model from the Modelica Standard Library. This is a V6 engine with 6 cylinders. It is hierarchically built up by using instances of one cylinder. For more details on the modelling of one cylinder, see example Engine1b.



4.5 RobotR3

The RobotR3 model from he Modelica Standard Library. This example animates a motion of a detailed model of the robot with predefined axes' angles over time. For animation, CAD data is used. Translate and simulate with the default settings (default simulation stop time = 2 s). The path planning block incorporates a simulation termination condition. Thus, the simulation can be terminated before reaching the stop time. The condition depends on the start and end positions of the joints, and on their reference speeds and reference accelerations. For current settings, the termination condition should indeed be fulfilled right before the simulation stops.



5 Summary

This document reports on the deliverable D6.2, which consists of an initial set of tools that populate the tools catalogue on the HUBCAP platform. These tools are provided by the HUBCAP partners, and they are installed and tested in the platform sandbox. The models described in D6.1 and in Section 4 can be used together with the tools.