<table>
<thead>
<tr>
<th>Deliverable No.</th>
<th>HIFI-ELEMENTS D3.5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deliverable Title</td>
<td>Add-ons and plug-ins to connect to the project developed test framework and simulation framework with data management tool</td>
</tr>
<tr>
<td>Deliverable Date</td>
<td>2019-06-30</td>
</tr>
<tr>
<td>Deliverable Type</td>
<td>REPORT</td>
</tr>
<tr>
<td>Dissemination level</td>
<td>Public (PU)</td>
</tr>
<tr>
<td>Written By</td>
<td>Markus Deppe, dSPACE</td>
</tr>
<tr>
<td>Writeen By Date</td>
<td>2019-05-09</td>
</tr>
<tr>
<td>Checked by</td>
<td>Jens Ewald (coordinator)</td>
</tr>
<tr>
<td>Checked By Date</td>
<td>2019-09-16</td>
</tr>
<tr>
<td>Approved by</td>
<td>Steering Committee</td>
</tr>
<tr>
<td>Approved By Date</td>
<td>2019-09-06</td>
</tr>
<tr>
<td>Status</td>
<td>Final</td>
</tr>
<tr>
<td>Status Date</td>
<td>2019-09-17</td>
</tr>
</tbody>
</table>
H2020-GV7-2017 – 769935 – High Fidelity Electric Modeling and Testing

Acknowledgement

The author(s) would like to thank the partners involved with the project for their valuable comments on previous drafts and for performing the review.

Project partners
1 – FEV – FEV Europe GmbH – DE
2 – RWTH – Rheinisch-Westfälische Technische Hochschule Aachen – DE
3 – RIC – Ricardo UK Limited – UK
4 – TEC – Fundacion TECNALIA Research & Innovation – ES
5 – CID – Fundacion CIDETEC – ES
6 – IVE – Iveco S.p.A. – IT
7 – DEN – Denso Automotive Deutschland GmbH – DE
8 – TOF – TOFAS Turk Otomobil Fabrikasi A. S. – TR
9 – IDI – IDIADA Automotive Technology SA – ES
10 – TNO – Nederlandse Organisatie voor Toegepast-natuurwetenschappelijk Onderzoek – NL
11 – MGEF – Mondragon Goi Eskola Politiknikoia J.M.A. S.Coop – ES
13 – VUB – Vrije Universiteit Brussel – BE
14 – MAG – MAGNA Engineering Center Steyr GmbH & Co. KG – AT
15 – YASA – YASA Motors Limited – UK
16 – FORD – Ford Werke GmbH – DE

Disclaimer:
This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 769935.
D3.5 – Add-ons and plug-ins to connect the data management tool to the project developed test framework and simulation framework

### Document information

**Additional author(s) and contributing partners**

<table>
<thead>
<tr>
<th>Name</th>
<th>Organisation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pajenkamp, Thomas</td>
<td>dSPACE</td>
</tr>
</tbody>
</table>

### Document Change Log

<table>
<thead>
<tr>
<th>Name</th>
<th>Date</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>V0.1</td>
<td>2019-05-09</td>
<td>Initial draft</td>
</tr>
<tr>
<td>V1.0</td>
<td>2019-06-12</td>
<td>Version for Review</td>
</tr>
<tr>
<td>Final</td>
<td>2019-09-06</td>
<td>Approved version by Steering Committee</td>
</tr>
</tbody>
</table>

### Document Distribution Log

<table>
<thead>
<tr>
<th>Name</th>
<th>Organisation</th>
<th>Distributed to</th>
</tr>
</thead>
</table>

### Verification and approval

<table>
<thead>
<tr>
<th>Name</th>
<th>Organisation</th>
<th>Date</th>
</tr>
</thead>
</table>
Executive summary
This deliverable describes the SYNECT add-ons and plug-ins to connect the data management tool to the project developed test framework (Tool for Electrical Vehicle Testing) and simulation tool (xMOD). The test framework uses the SYNECT Requirements Management module to store the textual formal requirements and the SYNECT Test Management module to store and execute test cases. The integration between TEVET and SYNECT is implemented by means of the HIFI Testing Add-On which will be described in this deliverable. The integration of vehicle level simulation models is implemented in SYNECT by means of the Variant Management and Model Management modules. This deliverable presents the according HIFI Model Integration Add-On which supports the model integration process as well as the export of an xMOD specific model integration file format called ZXMIPS.
Contents

1 Purpose of the Document ................................................................. 7
  1.1 Document Structure ................................................................... 7
  1.2 Deviations from original Description in the Grant Agreement Annex 1 Part A ................................................................. 7
    1.2.1 Description of work related to deliverable in GA Annex 1 – Part A ................................................................. 7
    1.2.2 Time deviations from original planning in GA Annex 1 – Part A ................................................................. 7
    1.2.3 Content deviations from original plan in GA Annex 1 – Part A ........................................................................ 7

2 Introduction .................................................................................... 8
  2.1 Overall Target of the HIFI-ELEMENTS project ........................................ 8
  2.2 Target of Work Package 3 (WP3) ..................................................... 8

3 Architecture .................................................................................. 9
  3.1 Extending SYNECT ...................................................................... 10
  3.2 SYNECT Add-Ons ....................................................................... 10

4 SYNECT Testing Add-On ................................................................ 12
  4.1 Internal Architecture of the Testing Add-On ...................................... 12
  4.2 Manage Formal Requirements ...................................................... 13
  4.3 Create Test Cases ........................................................................ 14
  4.4 Execute Test Cases ....................................................................... 15
    4.4.1 Finished Test Executions .......................................................... 15
  4.5 Limitations .................................................................................. 16

5 SYNECT Model Integration Add-On .................................................. 17
  5.1 Internal Architecture of the Model Integration Add-On ................. 17
  5.2 Model Integration ....................................................................... 18
    5.2.1 Create a Variant Configuration ............................................... 19
    5.2.2 Create Model Implementation for a Variant Configuration .......... 21
  5.3 Create a ZXMIPS File .................................................................. 23
  5.4 Import ZXMIPS in xMOD ............................................................. 23
  5.5 Limitations ................................................................................ 24

6 Summary ....................................................................................... 25

7 Risk Register ................................................................................ 26
  7.1 References ................................................................................ 26

8 Quality Assurance .......................................................................... 27

9 Acknowledgment ......................................................................... 28

Figure 1-The AMDF architecture and the focus of this deliverable ....................... 9
Figure 2-Usage of the XIL API Model Access Port ..................................... 10
Figure 3-The Testing Add-On ZIP archive (*.addonz) ................................. 11
Figure 4-View and manage the Testing Add-On and the Model Integration Add-On in SYNECT ................................................................. 11
Figure 5-HIFI Testing Add-On ................................................................ 12
D3.5 – Add-ons and plug-ins to connect the data management tool to the project developed test framework and simulation framework

Figure 6-The Testing Add-On Integration with SYNECT ................................................................. 13
Figure 7-Example: Edit the formal requirements for an E-Motor FMU model ................................... 14
Figure 8-Example: Generated Test Cases ...................................................................................... 15
Figure 9-Example: Create Execution and execute Test Cases ......................................................... 15
Figure 10-Example: Test results ...................................................................................................... 16
Figure 11-HIFI Model Integration Add-On ....................................................................................... 17
Figure 12-The Model Integration Add-On Integration with SYNECT .................................................... 18
Figure 13-Model Integration defined by a Variant Configuration ..................................................... 19
Figure 14-Predefined Variant Configurations ..................................................................................... 19
Figure 15>Create a new Variant Configuration, Step 1 ................................................................. 20
Figure 16(Create a new Variant Configuration, Step 2 ...................................................................... 21
Figure 17-Create a Model Implementation for a Variant Configuration ............................................ 22
Figure 18-The created Model Implementation with the selected component Implementations ............. 22
Figure 19-Create the ZXMIPS file .................................................................................................... 23
Figure 20-Result of the ZXMIPS import in the xMOD editor ............................................................ 24

Tables
Table 1: D3.5 contribution to HIFI objectives. .................................................................................. 8
1 Purpose of the Document

This document is the fifth deliverable of WP3. It presents the SYNECT add-ons and plug-ins to connect the data management tool to the project developed test framework (Tool for Electrical Vehicle Testing) and simulation tool (xMOD). The test framework as well as the simulation framework are described in greater detail in respective deliverables. The focus of this document is to describe their integration with SYNECT data management by means of SYNECT add-ons and plug-ins.

1.1 Document Structure

The chapters of the document have the following focus.

Chapter 2: Introduces the goals of the HIFI project and WP3. Clarifies the scope of this deliverable.
Chapter 3: Presents the concepts of the architecture used to integrate the test and simulation framework.
Chapter 4: Explains the integration of the test framework in SYNECT.
Chapter 5: Describes the model integration in SYNECT targeting the simulation tool xMOD.
Chapter 6: Summary

1.2 Deviations from original Description in the Grant Agreement Annex 1 Part A

1.2.1 Description of work related to deliverable in GA Annex 1 – Part A

Within WP3, the component models (including parameter handling), the model integration process, as well as the test processes will be specified and implemented for offline as well as real-time simulation based on a central management system. Furthermore, the software framework to enable real-time co-simulation (including automated generation of system models) will be refined and extended by a test framework for automatic execution of functional and nonfunctional test cases on component and system level. Finally, the model-based calibration will be performed.

1.2.2 Time deviations from original planning in GA Annex 1 – Part A

Due to the review process of the Deliverable over the summer holidays, there is a slight delay in submission of the documents which does not have a further impact on the overall project progress.

1.2.3 Content deviations from original plan in GA Annex 1 – Part A

There are no deviations from the Annex 1 – Part A with respect to the content.
2 Introduction

2.1 Overall Target of the HIFI-ELEMENTS project

The HiFi-Elements project aims to reduce the design and validation effort for e-drivetrains by defining interface standards and workflows to combat insufficient model reuse and to ensure model interoperability and scalability. With the development of a structured toolchain it is possible to realize different software (virtual validation software, model data management software) to let them work in symbiosis by following a standardized automated path. Within the developed workflow, automatic parameter identification functions and automatic test case generation will also be implemented. The proposed workflow and interface standards will provide the possibility of earlier system validation and, as a result, it aims to:

- Increase the efficiency of the development process
- Decrease the development, testing and system integration effort to reduce time-to-market
- Increase safety and reliability of EVs

To test the improvement of the developed workflow and toolchain, four different use cases will be used to record the efforts expended for the different tasks in each of them. The recordings will be compared with the assessment of documented efforts from the past for similar tasks utilizing the current practice.

2.2 Target of Work Package 3 (WP3)

Within WP3, the component models (including parameter handling), the model integration process, as well as the test processes will be specified and implemented for offline as well as real-time simulation based on a central management system. Furthermore, the software framework to enable real-time co-simulation (including automated generation of system models) will be refined and extended by a test framework for automatic execution of functional and nonfunctional test cases on component and system level.

This deliverable summarizes the outcome of Task 3.3.3 “SYNECT Add-Ons for Test and Simulation” which is a subtask of Task 3.3 dealing with the development of the AMDF (Agile Model Development Framework) toolchain.

Contribution by partners:
The presented work has been performed by FEV (model integration), Mondragon University (TEVET test framework integration) and dSPACE (coordination, SYNECT integration, SW architecture and development support).

Table 1 lists the HIFI-Elements objectives which are supported by Task 3.3.3:

<table>
<thead>
<tr>
<th>HIFI Objective</th>
<th>Description</th>
<th>Deliverable D3.5 Contribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>O1b</td>
<td>Standard model meta data</td>
<td>The model integration workflow makes use of model meta data addressing aspects such as modeling-depth, real-time behavior and component combination constraints.</td>
</tr>
<tr>
<td>O2d</td>
<td>Create system co-simulation models that can be used in real-time HiL-test setups, targeting a 75% reduction in effort.</td>
<td>An efficient automated model integration workflow based on a formal variant model supports the fast and reliable creation of co-simulations.</td>
</tr>
<tr>
<td>O2e</td>
<td>Plan and execute a test program that meets a given set of test objectives, targeting a 50% reduction in effort.</td>
<td>Model Management, Variant Management and Test Management together with the TEVET framework allow efficient creation, planning and execution of tests.</td>
</tr>
<tr>
<td>O4</td>
<td>Increase in validation test coverage</td>
<td>Contribution by automatic test generation and Adaptive Random Testing (ART)</td>
</tr>
</tbody>
</table>

Table 1: D3.5 contribution to HIFI objectives.
3 Architecture

The tool chain which is developed in HIFI-Elements is composed of several software products as shown in Figure 1. The Data Management software SYNECT [10] is the central data backbone to connect the tools for vehicle model architecture specification, testing, modeling, calibration and simulation. One focus of this deliverable is to describe the SYNECT integration of the test framework TEVET (Tool for Electrical Vehicle Testing) implemented by Mondragon University. The second topic is the SYNECT integration of a model integration process developed by FEV and dSPACE allowing to build vehicle level co-simulations for the xMOD platform.

![Figure 1-The AMDF architecture and the focus of this deliverable](image)

For details on the Test Framework and the co-simulation platform there are separate deliverables [5, 6]. These deliverables also describe the coupling of the Test Framework and xMOD with the XIL API Model Access Port [11] as depicted in Figure 2. In this deliverable, the XIL API MA Port is not further discussed.
3.1 Extending SYNECT

SYNECT is designed to be an open platform allowing to access all data via scripts, to add custom plugins and to extent the graphical user interface of the client by adding user context menus or ribbon buttons. For this, SYNECT provides a powerful Client and Server API allowing to integrate in highly automated and highly customer specific environments without the need to change the core product.

The SYNECT client can be accessed via the client API COM interface by all programming languages supporting COM (Microsoft Common Object Model) such as Python, C# (.NET) and MATLAB M-scripts.

Main features of the client API are:
- Determine the elements selected by the user in the GUI to have a context for actions (e.g. a Test Case)
- GUI extensions like additional context menu entries or ribbon buttons
- Reaction to events of the client, e.g., button pressed, view activated
- Running plugins
- Starting scripts on the SYNECT server

SYNECT uses plugins to exchange data between SYNECT and files or other applications. SYNECT lets you select a plugin when you import or export items or you execute test cases (Test Execution Plugins). A plugin consists of a configuration file (ECXML) and a program or script. SYNECT comes with a predefined set of plugins and is prepared to be easily extended by custom plugins.

3.2 SYNECT Add-Ons

To manage and distribute a whole set of client/server scripts, context menu extensions, ribbon button extensions and SYNECT plugins as one consistent software package, SYNECT supports a so-called add-on concept. One add-on basically consists of a ZIP archive containing metadata and a predefined folder structure holding all implementations and configurations. Figure 3 shows the internal structure of the Testing Add-On *.addonz file archive.

Within the HIFI-Elements project the add-on archives are distributed among the partners using the central HIFI SVN server hosted by dSPACE.
D3.5 – Add-ons and plug-ins to connect the data management tool to the project developed test framework and simulation framework

Figure 3-The Testing Add-On ZIP archive (*.addonz)

The SYNECT client provides an “Extensions” page that allows to view and manage the add-ons. Basic operations are installation, update or removal of add-ons. Figure 4 shows the Testing and Model Integration Add-Ons on the extensions page. For each add-on basic information, such as name, author, version and a description, is shown. In addition, a link to further documentation (e.g. a PDF file) is available. This documentation is part of the add-on archive.

Figure 4-View and manage the Testing Add-On and the Model Integration Add-On in SYNECT
SYNECT Testing Add-On

Mondragon University has developed a tool named TEVET (Tool for Electrical Vehicle Testing), which includes an editor for formal requirements specification, a test specification generator and a test execution tool for signal-based testing. The test cases can be generated automatically by considering functional requirements. To specify requirements, a Domain Specific Language (DSL) has been developed. The test generator parses each requirement and automatically generates from 1 to N test cases complying with the ASAM XIL standard. The test cases are exported as *.STI files. The test execution involves logic that reads the automatically generated *.STI files and converts these test cases into stimulation signals to test the system. For details on the TEVET there is an extra deliverable D3.3 [5]. Here only the SYNECT integration of TEVET is explained and not the functionality of the test framework itself.

The SYNECT Testing Add-On connects TEVET to SYNECT Model Management, Requirements Management and Test Management. In addition, the SYNECT Testing Add-On provides the graphical user interface (GUI) to operate TEVET. TEVET itself has additional interfaces to Eclipse (DSL Plugin to edit formal requirements) and to xMOD (XIL API Model Access Port for stimulation and simulation). The user works in the SYNECT and Eclipse environments. The stimulation and simulation run as an automated sequence without the need for user interaction. The figure below illustrates the interactions between SYNECT, Eclipse, TEVET and xMOD.

![Figure 5-HIFI Testing Add-On](image)

Internal Architecture of the Testing Add-On

To clearly separate the programming work between dSPACE and Mondragon University a simple internal architecture for the Testing Add-On has been set up. The basic part is the definition of an “Internal API” between the dSPACE and the Mondragon part (see Figure 6). Many parts of the implementation consist of Python scripts, but the TEVET core parts consist of Java scripts and Eclipse plugins. For test execution (simulation and stimulation) TEVET makes use of xMOD controlled by a XIL API Model Access Port Client/Server interface. All shown Python and Java modules are part of the add-on archive. Nevertheless, additional software installations such as Eclipse and xMOD are required to use the add-on.
4.2 Manage Formal Requirements

The formal requirements are formulated by means of a Domain Specific Language (DSL). These textual requirements are managed with in the SYNECT Requirements Management (RM) module. By this, any DSL fragment becomes a SYNECT artifact (referred to as RM Content object) within a RM document. This RM document is specific for one component model implementation (e.g. one FMU or one XMODEL). This dependency is modeled in SYNECT by introducing a link between the RM document and the according model implementation artifact in SYNECT Model Management. Since SYNECT RM cannot support syntax checking and syntax highlighting for the DSL, the actual editing happens in Eclipse. There, Mondragon University provides a dedicated DSL Eclipse plugin to support comfortable and efficient editing of the DSL. The data exchange between SYNECT RM and Eclipse is automated via temporary DSL files. The user can easily open the Eclipse editor from within SYNECT RM. This feature is part of the Testing Add-on. Figure 7 shows the according context menu, the link to the model implementation and the DSL in the eclipse editor.
4.3 Create Test Cases

Once the formal requirements are ready, it is possible to generate test cases automatically. For each test, a SYNECT Test Case item and a test implementation file (*.STI) are created. Each SYNECT Test Case item is linked automatically to the formal requirement item in the SYNECT Requirements Management, to the test implementation file (*.STI) and to the model implementation (system under test). Thereby, the subsequent test execution has all the information required to setup and run simulation-based tests with xMOD. The figure below shows the context menu to be used for requirements management documents to trigger the generation of test cases.
D3.5 – Add-ons and plug-ins to connect the data management tool to the project developed test framework and simulation framework

4.4 Execute Test Cases

The creation of test executions from a set of selected Test Cases (see left part of Figure 9) and the triggering of the execution are basic tool features of SYNECT. The only HIFI-Elements-specific extension in this area is the implementation and usage of the TEVET Test Execution Plugin called “HIFI Testing”. The user is prompted to select a Test Execution Plugin after he clicks the “Execute” context menu. During the test execution the progress is monitored. In addition, it is possible to abort the test execution manually (see right part of Figure 9).

4.4.1 Finished Test Executions

Each Test Execution run is logged in the “Finished Executions” area of SYNECT Test Management (see Figure 10). For each Execution the individual Test Case Results are stored. Every Test Case Result has a verdict, a simulation capture file (*.mf4) and a html or PDF report.
D3.5 – Add-ons and plug-ins to connect the data management tool to the project developed test framework and simulation framework

Figure 10-Example: Test results

4.5 Limitations

The HIFI Testing Add-On has been implemented and tested for component models (e.g. one e-motor) using scalar input and output signals. The basic concept and the add-on software architecture in principle is also extendable to support system models (vehicle level models) as well as non-scalar signals without significant effort.
5  SYNECT Model Integration Add-On

For test and simulation purposes in the HIFI-Elements “Use Case” working packages, there is the requirement to integrate the components of the electric drive into an overall system (vehicle model). The components are available in SYNECT in different versions for the different vehicle topologies (see D1.2 [1]). The topologies can be defined using the SYNECT Variant Management module. To integrate a topology, a Variant Configuration must be selected in SYNECT. Subsequently, the corresponding models as *.FMU or *.XMODEL are collected and connected to run on the simulation platform xMOD. For this, SYNECT generates a *.ZXMIPS file that contains the models and the metadata of the topology-related models for integration in xMOD. The models and configuration data are then imported into xMOD and the simulation can be loaded and executed by the user. For details on the simulation framework xMOD see deliverable D3.4 [6]. Here, only the SYNECT integration of the model integration is explained and not the functionality of the simulation framework itself.

The SYNECT Model Integration Add-On provides a user interface for the model integration process as well as an ZXMIPS export feature. The figure below shows the basic steps for the user and the coupling between SYNECT and xMOD. The focus of the chapter is to explain the SYNECT Model Integration Add-On. Details about the ZXMIPS file format or about the operation of xMOD will not be explained. Please refer to D3.4 [6] for more details on these topics.

![Figure 11-HIFI Model Integration Add-On](image)

### 5.1  Internal Architecture of the Model Integration Add-On

To clearly separate the programming work between dSPACE and FEV a simple internal architecture for the Model Integration Add-On has been set up. The basic part is the definition of an “Internal API” between the so-called Model Integration Adapter and the ZXMIPS Creation Adapter as shown in Figure 12. The add-on is mainly implemented in Python but the part dealing with the XML handling of XMIPS meta data such as XML schema validation or serialization and deserialization has been implemented in C# (.NET) which can easily be integrated in Python. All shown Python and .NET modules are part of the add-on archive. Nevertheless, the additional software installation of xMOD is required to deploy the resulting ZXMIPS files.
5.2 Model Integration

In the scope of this project the typical topologies of pure battery electric light commercial vehicles are considered. These typical topologies are very similar and are variants of each other. All in all, 5 vehicle architecture variants are considered and described in detail in D1.2 [1] (section 3: “System Architecture in Enterprise Architect”).

As an example, the vehicle architectures 1 and 2 are both having a rear wheel drive (RWD) with one central E-Motor and a central transmission with differential. The only difference is that in architecture 2 no extra high voltage DC/DC converter between battery and inverter is required since the battery and inverter DC voltage level is identical.

Based on the system architecture model in Enterprise Architect (SysML) a 150% system model has been created in SYNECT as a superposition of all 5 architectures. To be able to derive the individual 100% architectures from the 150% system model in an automated way a formal variant model is established.

A variant model comprises all variable aspects of a domain and their options. By variant management, the model integration process of the HiFi-Elements project is simplified. A variant configuration that completely characterizes a simulation model at vehicle level can be defined by referring to the variants of the variant model. For details on the formal Variant Model please refer to deliverable D3.2 [4].

The model integration process with the goal to derive a simulation model for a vehicle has two layers (see Figure 13). First, the desired 100% vehicle architecture needs to be chosen. The next major step for the model integration is the selection of the desired component model implementations (including model formats like FMU, XMODEL, …) for all entities of the architecture. In this step, normally many alternative interface compliant implementations are available. Depending on the purpose of the simulation, each component implementation is selected regarding his capabilities such as modeling fidelity, real-time capabilities or compatibility with other components of the system.
The selection of suitable component models to integrate into a system model requires a comprehensive understanding of the individual component model's capabilities. The pool of models that are going to be developed in the HiFi-Elements project support a wide range of simulation scenarios. SYNECT Variant Configurations guide the model integrator and help him or her to avoid invalid model combinations.

5.2.1 Create a Variant Configuration

Model Integrations require a Variant Configuration. Either an existing Variant Configuration is chosen, or a new Variant Configuration needs to be created to meet the requirements of the intended co-simulation. In this section, the creation of a new Variant Configuration in SYNECT is explained.

To support the HiFi-Elements project the central SYNECT server provides a predefined Variant Models as described in D3.2 [4]. Thus, the starting point for editing the variant model is to open the predefined Workspace “Integration Variant Model” in SYNECT. It contains the single Variant Project called “Variant Model” as shown in Figure 14. This Variant Project contains the list of predefined Variant Configurations.
By clicking the ribbon button “New” an additional Variant Configuration can be created by using the “New Variant Configuration” dialog as shown in Figure 15.

![New Variant Configuration dialog](image)

**Figure 15**: Create a new Variant Configuration, Step 1

The desired 100% architecture is defined by binding the first 5 variation points called “Drivetrain Config”, “Transmission Type”, “Usage of DC/DC Converter” and “Architecture”. Now, each Component Variation Point needs to be bound to specify the exact component model implementation (FMU, XMODEL, …) to be used for the co-simulation. If a component is omitted for the co-simulation, it is marked by the special Variant “_None”.

Finally, a reasonable name needs to be provided to identify the Variant Configuration. Figure 16 shows an example for Architecture 4 where all component models are bound to XMODEL mockup models provided for demo and testing purposes.
5.2.2 Create Model Implementation for a Variant Configuration

Once a Variant Configuration is set-up or selected, a workflow can be initiated to automatically derive a system model based on the information in the Variant Configuration. The system model is set up to include all the component implementations defined in the Variant Configuration. A ZXMIPS file can be generated from the system model for simulation in xMOD. Beforehand, the model integrator can fine-tune the communication step-sizes for model exchange simulation of the individual components in the system model.

To create a model implementation for the co-simulation of a vehicle, first a corresponding target workspace in SYNECT must be chosen or created. Some predefined workspaces already exist. There are workspaces for the four HIFI-Elements use case work packages (WP4,5,6,7) and one workspace for testing and development called “WPx Mockup Integration Workspace”. The latter workspace contains a Model Management Project called “_WPx Mockup System Integration” (see Figure 17). It contains frame models as FMUs and XMODELS which were generated based on the system architecture. For details about the frame models please refer to D3.1 [3].

The project node or the project summary node in the Navigator Tab offer the context menu “HIFI: Create Implementation for Model Integration...”. Once this is clicked a dialog opens allowing to specify the name of the co-simulation model implementation and the name of the Variant Configuration to be used.
D3.5 – Add-ons and plug-ins to connect the data management tool to the project developed test framework and simulation framework

Figure 17 - Create a Model Implementation for a Variant Configuration

If the dialog is confirmed with “OK” a new Model Implementation will be created automatically in SYNECT and assigned to the matching 100% system model. In the given example, the system model is “BE Vehicle A4”. Thus, the new model implementation “ArchitectureA4Example” can be found under the “Implementations” node of “BE Vehicle A4” as shown in Figure 18. The screenshot in Figure 18 also shows the content of the “Submodel Implementations” of “ArchitectureA4Example”. Here the assignment of XMODELS and the configuration of the “CommunicationStepSize” is located.

For Co-Simulation the simulation time interval is split into a grid of communication points, where the data exchange between the subsystems is restricted to discrete points. The time interval between these communication steps is defined by the communication step size. The time unit of the step sizes is "seconds". Communication step sizes of “Null”, 0 or negative values are interpreted as being the same as the simulation step size.

Figure 18 - The created Model Implementation with the selected component implementations
5.3 Create a ZXMIPS File

To transfer the co-simulation setup to xMOD, the file format *.ZXMIPS has been designed. SYNECT allows to create ZXMIPS files of a model integration. To do this, the model implementation needs to be selected and the context menu “HIFI: Create ZXMIPS” needs to be clicked (see Figure 19). As a result, the ZXMIPS file will be exported to the working folder of the Model Implementation. In addition, the ZXMIPS file is classified as “xMOD Integration” to allow processing by automation scripts.

![Figure 19 - Create the ZXMIPS file](image)

5.4 Import ZXMIPS in xMOD

“xMOD is a co-simulation and virtual experimentation laboratory platform. The over-arching concept for xMOD is that it is intended to connect heterogeneous models (e.g. Simulink®, GT-SUITE®, AMESim® ...etc.) through a versatile and efficient coupling process. This is achieved by optimizing model execution through multi-cored and multi-solver processing with multi-rate, and by providing a standalone model execution capability. However, xMOD also provides some additional benefits, including compatibility with emerging standards such as Functional Mock-up Interface (FMI), as well as continuity in the transition from Software-in-Loop (SiL) to Hardware-in-Loop (HiL). Finally, xMOD allows customization of graphical user interfaces which can be essential when facilitating ‘non-expert’ end user access.” (taken from xMOD Product Information [9])

For HIFI-Elements, FEV xMOD has been extended by a ZXMIPS import feature. Figure 20 shows the result of the ZXMIPS import for the “Architecture A4” example. All component models as well as all connections between the inputs and outputs of the components have been imported. The graphical properties such as position, size and color of the blocks are also contained in the ZXMIPS and originate from settings in SYNECT. This guarantees that every model integration produces identical or similar variants of block diagrams with a high recognition value for the user.
5.5 Limitations

The ZXMIPS metadata contains a section called “Quantities” where the model variables which are subject to recording/measurement are defined. On export, SYNECT adds all known output and input variables from the HIFI-Elements architecture definition as Quantities. For the interactive use cases, the user then is relieved from doing this definition manually in the GUI of the tool. For the automation use cases with XIL API MA [11], this technique is the only way to specify the available model variables being available for stimulation and capturing.

There is one notable limitation with xMOD 2019: The ZIMPS import is only capable of reading scalar quantities from the file. Thus, the SYNECT ZIMPS export omits all non-scalar variables and issues according warning messages in the SYNECT log.
6 Summary

In this document, the SYNECT integration as well as the basic user interface for the automatic test case generation tool (TEVET) and for the model integration targeting xMOD have been presented.

The automatic test case generation tool (TEVET) highly interacts with SYNECT Requirements Management and SYNECT Test Management. Details about the TEVET approach and concepts are documented in D 3.3 [5]. The documentation of the integration of TEVET in the HIFI-Elements workflow is done in D3.6 [7]. Details about the ASAM XIL API MA Port [11] are given in D3.4 [6].

The model integration highly interacts with SYNECT Variant Management and SYNECT Model Management. Details about the simulation framework xMOD, XIL API MA Port [11] and the ZXMIPS format are documented in D3.4 [6].

All in all, the automatic test case generation tool (TEVET) and the model integration targeting xMOD are successfully integrated with SYNECT as planned. There are minor limitations regarding non-scalar variables which are documented in this deliverable. For the documentation of the usage of these tools within the HIFI-Elements workflow (D1.3 [2]) please refer to D3.6 [7].
## 7 Risk Register

<table>
<thead>
<tr>
<th>Risk number</th>
<th>Description of Risk</th>
<th>Proposed Risk Mitigation Measure</th>
<th>Probability / effect(^1)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\(^1\) Level of probability / effect: 1 = high, 2 = medium, 3 = low

### 7.1 References

[1] Deppe, M. et al, *HIFI-ELEMENTS Deliverable D1.2; Document describing the SYNECT generated component model reports, the component interfaces and the system architecture*, 2018


8 Quality Assurance

The Steering Committee is the body for quality assurance. The procedure for review and approval of deliverables is described in the deliverable report D8.1 – “Project Handbook”. The quality will be ensured by checks and approvals of WP Leaders as part of the steering committee (see front pages of all deliverables).
D3.5 – Add-ons and plug-ins to connect the data management tool to the project developed test framework and simulation framework

9 Acknowledgment

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under Grant Agreement no. 769935.

Disclaimer
This document reflects the views of the author(s) and does not necessarily reflect the views or policy of the European Commission. Whilst efforts have been made to ensure the accuracy and completeness of this document, the HIFI-ELEMENTS consortium shall not be liable for any errors or omissions, however caused.