GA No. 101095856

HORIZON EUROPE PROGRAMME

TOPIC HORIZON-CL5-2022-D5-01-08

Clean and competitive solutions for all transport modes

GA No. 101084046

Zero Emission flexible vehicle platform with modular powertrains serving the long-haul Freight Eco System

ZEFES - Deliverable report

D2.1 - Vehicle Simulation Platforms

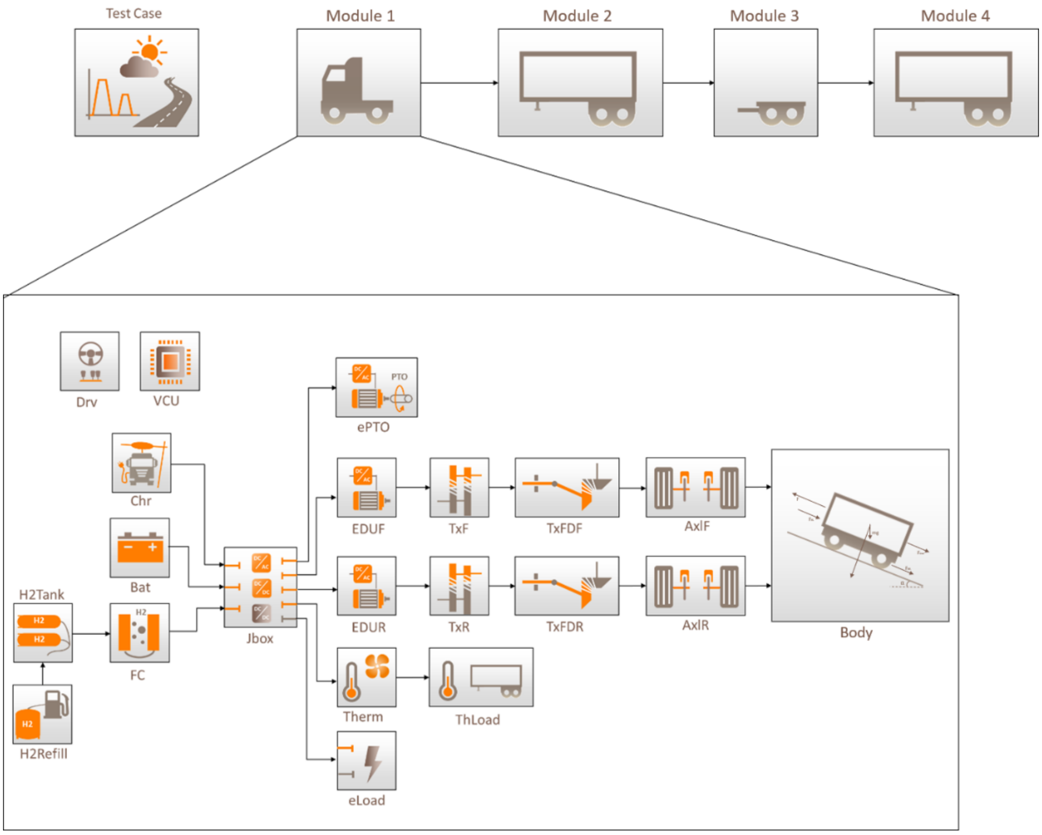
Funded by the European Union

Deliverable No.	ZEFES D2.1	
Related WP	WP2	
Deliverable Title	Vehicle Simulation Platforms	
Deliverable Date	2024-05-24	
Deliverable Type	REPORT	
Dissemination level	Public (PU)	
Author(s)	Marina Roche (IDI) Roberto Esteban (IDI) Daniela Andreina de Lima (IDI) Roberto Barrio (IDI) Jordi Plana (IDI)	
Checked by	Jordi Plana (IDI)	2024-04-18
Reviewed by (if applicable)	Christer Thoren (SCA) Pascal Revereault (RIC-UK)	2024-04-25
Approved by	Omar Hegazy (VUB) – Project coordinator	2024-05-28
Status	Final	2024-05-28

Publishable summary

Europe's commitment to be the first CO2-neutral continent by 2050 is going to impact the road transport industry, in part by requiring massive investments. To achieve EU CO2 reduction goals, research, policy, technology, and industry need to cooperate and ensure a smooth transition to ZE-HDVs. This objective requires that manufacturers of heavy-duty vehicles (HDV) deliver more efficient vehicles: a reduction of CO2 emissions for the newly produced fleet of 15% in 2025 and 30% in 2030.The use of zero tailpipe emissions vehicles (ZEV) for long distance heavy transport is an important part towards achieving the above targets. Such ZEV are Battery Electric Vehicles (BEVs) and Fuel Cell Electric Vehicles (FCEVs).

As part of the development of new HD ZEV, simulation models of the different components have been developed, with a focus on modularity and compatibility between technologies, makers and developers. The structure of these models can be seen in the following figure.



As a part of the simulation models, a Total Cost of Ownership (TCO) tool has also been developed. This tool calculates the price of a new vehicle considering technology used, country of purchase and use of the vehicle.

Contents

List of Figures

List of Tables

Abbreviations & Definitions

Abbreviation	Definition
BEV	Battery Electric Vehicle
BMS	Battery Management System
BoP	Balance of plant
b-trailer	Trailer that includes a battery for energy storage
CAPEX	Capital Expenditure
DI	Deionized
DoD	Depth of discharge
DoE	Design of Experiments
EAFO	European Alternative Fuels Observatory
EDU	E-Drive Unit
EEM	Electrical equivalent model
EMS	European Modular System
EOL	End Of Life
ePTO	electric Power Take-off
e-trailer	Trailer that includes a battery and a traction system
EV	Electric Vehicle
FC	Fuel Cell
FCEV	Fuel Cell Electric Vehicle
FHG	Fraunhofer-Gesellschaft
FWD	Front Wheel Drive
H2	Hydrogen
HD	Heavy Duty
HFCS	Hydrogen Fuel Cell System
ICCT	International Council on Clean Transportation
LFP	Lithium Iron Phosphate
Li-ion	Lithium-ion
LuT	Lookup table
NMC	Nickel Manganese Cobalt
NTU	Number of Transfer Units
OCV	Open circuit voltage
OPEX	Operational Expenditures
PI	Proportional Integral
ROM	Reduced Order Model
RWD	Rear Wheel drive
SoC	State of charge
SoH	State of health
SPC	Split Power Control
TCO	Total Cost of Ownership
VCU	Vehicle Control Unit
VECTO	Vehicle Energy Consumption calculation Tool

1 Introduction

The present report is part of the second ZEFES work package which aims at identifying the requirements and targets to develop the BEV and FCEV modular simulation tools (T2.1), the development of said models ensuring its flexibility (T2.2), and the co-design of an optimization framework to reduce TCO (T2.3); the right sizing of powertrain components for 90% payload (T2.4) and the feasibility of further improvements to reach 100% payload (T2.5); the thermal and energy management optimization of the modular powertrain concept (T2.6), the durability of a novel FC power unit (T2.7), reduced order models for selected use cases (T2.8) and the development of standardized connection and control interfaces between vehicle combinations (T2.9).

The current report (D2.1) is focused on the **Simulation Platform** that has been developed as part of WP2. The design philosophy, based on the requirements and targets, has focused on the **modularity and inter-compatibility** of the different components of the vehicle, to allow for the simulation of different types of BEVs and FCEVs, with different configurations of traction, trailer, and load. As part of T2.2, a **Total Cost of Ownership** tool has also been developed to evaluate and compare the costs related to the use of different technologies, considering available incentives, and the variable costs of electricity and hydrogen.

These tools will be linked to the optimization activities of T2.3 and T2.4, and to the development of an integrated tool for a digital twin of ZEV in WP4.

2 Multi-architecture modelling platform

This chapter describes the characteristics and the development of the multi-architecture modelling platform. This chapter and is structured in the following representative subsections:

- 2.2 Platform concept: overall overview of the platform characteristics

- 2.3 Component models for modular ZEV-powertrain concepts: details and development of the individual component models of the multi-architecture platform.

- 2.3.4.5 EDU

The electric drive units (EDUs) are the components that transform electric power into mechanic power, which is later transmitted via the transmission components, axles and wheels. The dynamics of the motor have some considerations:

- The available torque depends on the instant speed of the EDU shaft, which depends on the truck and drive line inertia and the applied torques on previous instants.
- The torque is limited with maximum and minimum torque in each instant. The torque boundaries are calculated each timestep using the exponential average torque and comparing it with peak and continuous torques.
- The rotor inertia is considered, and part of the indicated torque is used for accelerating its own rotational inertia.
- Internal losses are considered as efficiency, which are defined with LuTs, which can be of different dimensions:
  - 0D: The efficiency is a constant applied to the mechanical power.
  - 2D: The efficiency matrix is a 2D vector that depends on speed and torque.
  - 3D: The efficiency matric is a 3D vector that depends on speed, torque and voltage.
  - 4D: The efficiency considers speed, torque, voltage and temperature.
- Thermal considerations are included in a simple thermal model used to calculate EDU internal temperature.

2.1.1.1 Transmission system

The transmission system is the model for transmitting the power from the EDUs to the traction axles. In this general vehicle model, it is divided into two components: transmission main reduction (Tx) and final drive (TxFD). Both components are modelled the same, and they are connected in series, making it useful for transmission ratio sizing or BM with multiple ratios, permitting a constant final drive ratio with the same efficiency for all combinations in a simulation batch.

The torque going through the drive line from the EDU is reduced in each of the transmission components increasing it in exchange of angular speed. The power that goes through the transmission components has efficiency considerations and rotational inertia is added to the drive system, which affects the accelerations and decelerations.

2.1.1.2 Axles

The axles model takes as input the power coming from the transmission and distributes it among the traction wheels of the truck. This component also aggregates the wheels that don't have traction functionalities. It adds the axles inertia and losses modelled in 0D, i.e. a constant efficiency, and adds inertia, increasing the rotational inertia of the vehicle.

The hydraulic braking system is included in the axles, so this model also takes the brake request from the VCU and applies it.

2.1.1.3 Body

The body component aggregates all the masses of the module and all the external and internal forces coming from every element on the model. In this component, the rolling resistance, aerodynamic resistance and gradient force are calculated for the current module. The functionality of this model is thus calculating every interaction with factors external to the module, aggregate them and make them usable for all the calculations in VCU and other components. The force transmission among modules is also calculated here.

Multi-architecture modelling platform: concept of the platform, implementation actions and a user guideline to define the parameters of a specific simulation batch.

2.2 Platform concept

The simulation platform represents an energy and longitudinal dynamics vehicle model that is governed by physical equations and parametrization of the components in the model. The simulation platform aims to represent the same physical phenomena that can be simulated in specific licenced market software such as GT-Suite, Simcenter Amesim or AVL Cruise M but with the difference that all the possible vehicle and component architectures are already pre-implemented and the user or machine that communicates with the platform only needs to choose the parametrization to execute. This difference is a breakthrough in comparison to market software that require to manual actuation for most of the architecture modifications and aims to make a difference in terms of automatic vehicle optimization.

2.2.1.1 Special care was made to maximize the parametrization options when configuring the powertrain at different levels as specified in section 2.3.4.5 EDU

The electric drive units (EDUs) are the components that transform electric power into mechanic power, which is later transmitted via the transmission components, axles and wheels. The dynamics of the motor have some considerations:

- The available torque depends on the instant speed of the EDU shaft, which depends on the truck and drive line inertia and the applied torques on previous instants.
- The torque is limited with maximum and minimum torque in each instant. The torque boundaries are calculated each timestep using the exponential average torque and comparing it with peak and continuous torques.
- The rotor inertia is considered, and part of the indicated torque is used for accelerating its own rotational inertia.
- Internal losses are considered as efficiency, which are defined with LuTs, which can be of different dimensions:
    - 0D: The efficiency is a constant applied to the mechanical power.
    - 2D: The efficiency matrix is a 2D vector that depends on speed and torque.
    - 3D: The efficiency matric is a 3D vector that depends on speed, torque and voltage.
    - 4D: The efficiency considers speed, torque, voltage and temperature.
- Thermal considerations are included in a simple thermal model used to calculate EDU internal temperature.

2.2.1.2 Transmission system

The transmission system is the model for transmitting the power from the EDUs to the traction axles. In this general vehicle model, it is divided into two components: transmission main reduction (Tx) and final drive (TxFD). Both components are modelled the same, and they are connected in series, making it useful for transmission ratio sizing or BM with multiple ratios, permitting a constant final drive ratio with the same efficiency for all combinations in a simulation batch.

The torque going through the drive line from the EDU is reduced in each of the transmission components increasing it in exchange of angular speed. The power that goes through the transmission components has efficiency considerations and rotational inertia is added to the drive system, which affects the accelerations and decelerations.

2.2.1.3 Axles

The axles model takes as input the power coming from the transmission and distributes it among the traction wheels of the truck. This component also aggregates the wheels that don't have traction functionalities. It adds the axles inertia and losses modelled in 0D, i.e. a constant efficiency, and adds inertia, increasing the rotational inertia of the vehicle.

The hydraulic braking system is included in the axles, so this model also takes the brake request from the VCU and applies it.

2.2.1.4 Body

The body component aggregates all the masses of the module and all the external and internal forces coming from every element on the model. In this component, the rolling resistance, aerodynamic resistance and gradient force are calculated for the current module. The functionality of this model is thus calculating every interaction with factors external to the module, aggregate them and make them usable for all the calculations in VCU and other components. The force transmission among modules is also calculated here.

Multi-architecture modelling platform. As this is a platform targeted to very technical profiles with hundreds of parameters required for each simulation, the interface with the platform is mainly through text files that specify the parameters of each the components. The priority of the platform is not the interface, it is to permit more architecture and parametrization choices and to integrate detailed models from 3rd parties in the project. However, it is possible that in a second phase we develop a simplified interface targeted to logistic operators to modify the main vehicle parameters without the need of editing text files.

The tool is implemented in MATLAB/Simulink with inclusions of 3rd party detailed models that are integrated as S-Functions. The whole tool is compiled to be executable in the open server to the rest of the partners as depicted in Figure 1.

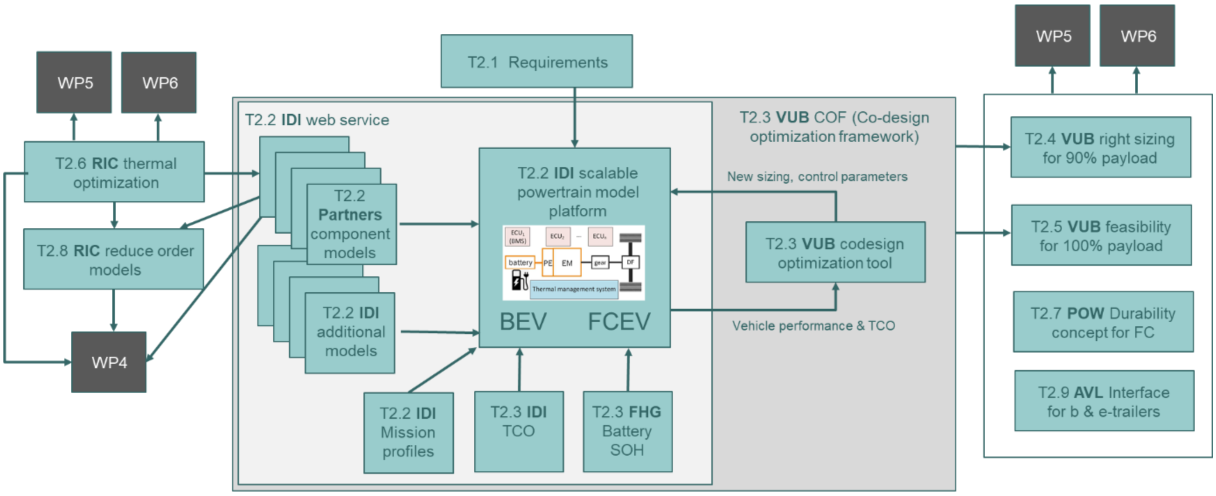


Figure 1: Platform concept and interaction with other tasks and WPs

2.2.2 Platform flexibility

The flexibility that the vehicle model provides to the platform users is the main highlight of the ZEFES model in comparison to commonly used compiled models with fixed parameters. This flexibility is required to perform vehicle topology optimization and permit correlation with different the demonstrators of the project.

The flexibility of the model can be split in different levels of parametrization options:

- Road train architecture: flexibility to concatenate modules (tractor, dolly, trailer, semitrailer) with or without EMS (European Modular System) traction in each of them.
- Module architecture: flexibility to model a module as passive or active (e-trailer, b-trailer, e-dolly, b-dolly) by including a powertrain traction architecture or a high voltage battery.
- Powertrain architecture: flexibility to choose between different powertrain architectures, with and without FC (Fuel Cell) supply, FWD (Front Wheel Drive), RWD (Rear Wheel Drive), 4WD, 1 or 2 reduction steps.
- Component model: flexibility to choose between model implementations of different fidelity by incorporating simplified mathematical models and detailed 3rd party models for some of the components (fuel cell, battery, tyre).
- Component sizing: flexibility to easily modify the size of a component (e.g.: power, torque, capacity…) to permit sizing optimizations. This update can be performed by modifying all the related parameters or making use of the scaling factors that modify the parameters with pre-defined formulas.
- Component modelling detail: most of the mathematical models permit to choose the level of detail of the input parameters ranging from simple averaged values to incorporating higher dimensional maps with up to 4D.
- Thermal system: possibility to represent thermal system estimated consumption required to condition the powertrain components, refrigerated trailers and cabin comfort at different external conditions with an architecture-agnostic approach. The maps required by this approach can be calibrated with the results of a detailed 1D model.
- Control system: possibility to simulate different control strategies of the VCUs (Vehicle Control Units) to control the regeneration strategy, split power control between fuel cell and battery, traction split between front and rear, power limitations and different communication schemes between the VCUs of the EMS (all master and master-slave configurations).

2.2.2.1 All the parametrizations have been implemented in an efficient structure so that the computational cost of the simulation is proportional to the complexity of the component models that are operative in the simulated assembly and the level of detail of each model, so that the most complex parametrization options do not affect the computational cost when not in use. The details of the implementation of the different parametrization levels are further explained in section 2.3.4.5 EDU

The electric drive units (EDUs) are the components that transform electric power into mechanic power, which is later transmitted via the transmission components, axles and wheels. The dynamics of the motor have some considerations:

- The available torque depends on the instant speed of the EDU shaft, which depends on the truck and drive line inertia and the applied torques on previous instants.
- The torque is limited with maximum and minimum torque in each instant. The torque boundaries are calculated each timestep using the exponential average torque and comparing it with peak and continuous torques.
- The rotor inertia is considered, and part of the indicated torque is used for accelerating its own rotational inertia.
- Internal losses are considered as efficiency, which are defined with LuTs, which can be of different dimensions:
  - 0D: The efficiency is a constant applied to the mechanical power.
  - 2D: The efficiency matrix is a 2D vector that depends on speed and torque.
  - 3D: The efficiency matric is a 3D vector that depends on speed, torque and voltage.
  - 4D: The efficiency considers speed, torque, voltage and temperature.
- Thermal considerations are included in a simple thermal model used to calculate EDU internal temperature.

2.2.2.2 Transmission system

The transmission system is the model for transmitting the power from the EDUs to the traction axles. In this general vehicle model, it is divided into two components: transmission main reduction (Tx) and final drive (TxFD). Both components are modelled the same, and they are connected in series, making it useful for transmission ratio sizing or BM with multiple ratios, permitting a constant final drive ratio with the same efficiency for all combinations in a simulation batch.

The torque going through the drive line from the EDU is reduced in each of the transmission components increasing it in exchange of angular speed. The power that goes through the transmission components has efficiency considerations and rotational inertia is added to the drive system, which affects the accelerations and decelerations.

2.2.2.3 Axles

The axles model takes as input the power coming from the transmission and distributes it among the traction wheels of the truck. This component also aggregates the wheels that don't have traction functionalities. It adds the axles inertia and losses modelled in 0D, i.e. a constant efficiency, and adds inertia, increasing the rotational inertia of the vehicle.

The hydraulic braking system is included in the axles, so this model also takes the brake request from the VCU and applies it.

2.2.2.4 Body

The body component aggregates all the masses of the module and all the external and internal forces coming from every element on the model. In this component, the rolling resistance, aerodynamic resistance and gradient force are calculated for the current module. The functionality of this model is thus calculating every interaction with factors external to the module, aggregate them and make them usable for all the calculations in VCU and other components. The force transmission among modules is also calculated here.

Multi-architecture modelling platform.

2.2.3 Ambition beyond state of the art

The tool has the ambition to go beyond current state of the art simulation platforms. Most commercial software is based on libraries of components that need to be connected in a specific manner to create a viable vehicle assembly. Correct connection of components is not trivial and usually requires a training.

In the ZEFES platform, all the different combinations that can be modelled (more than a hundred) are pre-implemented in the platform. Thanks to this the end user does not need to model to modify the topology of an architecture. This task can be simply performed by updating parameters to activate or deactivate different submodules of the modular system. This is a great advantage because it makes complex simulations accessible to users without the expertise to use a complex software or with the access to a commercial license.

Besides, even though most or all of the configurations of the ZEFES platform could be modelled individually in commercial software, it would not be possible to have all of them in a single modular model that changes the architecture topology depending on user parametric choices. It would require several different individual vehicle models which significantly increases the creation and maintenance effort and hinders the possibility of comparing architectures or performing architecture optimizations.

The flexibility of the models that permit to modify all of the internal parameters also differentiates from the common practice in industry when sharing IP protected models in which none or just few parameters are tuneable. This flexibility was made available by permitting the usage of the model through a web-service cloud platform instead desktop files.

The web service is also an added value of this project, as it democratises the access to simulation tools and 3rd party models to users without the need of a specific simulation license.

We aim the platform to be a common space available for the participants of the ZEFES project and to evolve the functionalities by incorporating the outcomes of the different project contributions being an example of good practise.

2.2.4 Platform interface

The platform interface consists of a home page and two main blocks: the TCO calculation and the simulation platform. The home page is depicted in Figure 2 and contains the log in and the links to the two the TCO and the simulation platform. Login is compulsory to prevent automated cyber-attacks to the platform. Login credentials will be handled to the project partners and stakeholders interested in working with the platform.

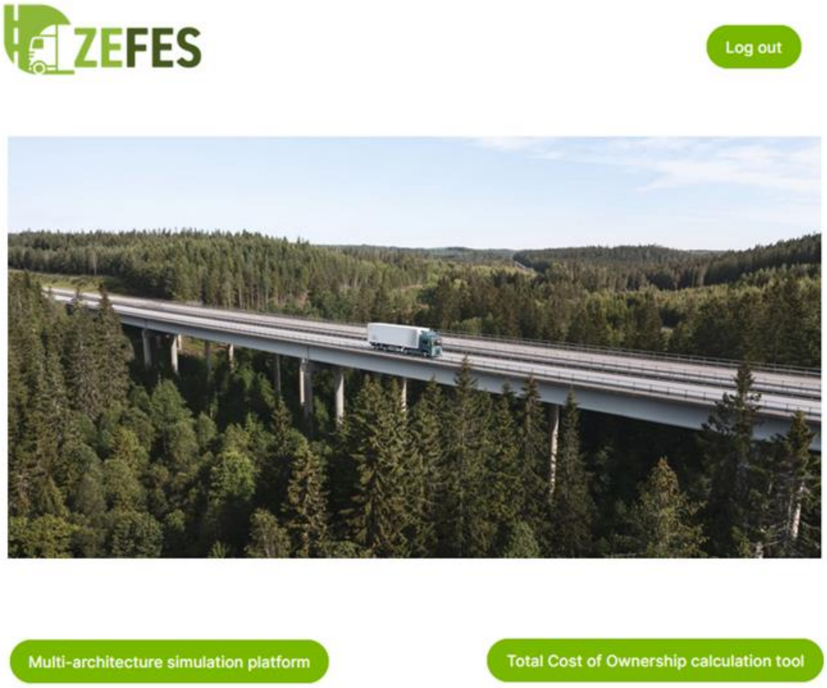


Figure 2: Home page of the simulation platform

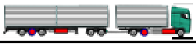
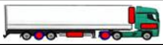
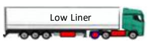
2.2.5 Vehicle architectures

The different vehicle architectures options that can be simulated in the multi-architecture simulation platform developed in WP2 were chosen based on the ZEFES project defined vehicle demonstrators: 9 vehicles concepts in total, from which 6 are battery electric heavy duty vehicles BE-HDV and 3 are fuel cell electric vehicles FCE-HDV, adding to a total of 16 different configurations, using standard semi-trailers, container semi-trailers, reefer semi-trailers, and low-lines semi-trailers.

Several powertrain concepts can be modelled such as battery packs in the semi-trailer as range extender, a full e-propulsion powertrain in both truck and semi-trailer and an e-propulsion for an emission free reefer operation.

As the simulation platform is developed with a modular approach, it also permits to simulate hypothetical combinations modules and architectures that are not included among ZEFES demonstrators such as 4WD architectures or including a fuel cell system in the active trailers.

Table 1: Summary table of vehicle configurations and ZEFES use case demonstrations

	Use Case	Vehicle configurations		Payload	Goods to be transported	Shipper	Battery Electric HDV	Fuel Cell Electric HDV	Rigid Unit (R)	Tractor Unit (T)	Semi-trailer (ST)	Dolly (D)	e-Dolly (e-D)	Semi-trailer (ST)	e-Trailer (e-T)	Cooled e-Trailer (e-reefer)
7.2 VOLVO	7.2.1	FC-1		64t GCW	Steel Scrap	Ovako		X	X		X					
	7.2.2	BEV-1		44t GCW	Automotive parts	Volvo Logistics	X			X	X					
				64t GCW	Automotive parts		X			X	X	X		X		
	7.2.3	BEV-2		64t GCW	Partly Hazardous goods	PG	X			X	X		X	X		
				44t GCW	Temperature-controlled goods	Primafrio	X			X						X
	7.2.4	BEV-3		48t GCW	Parcel distribution	DPD	X		X				X		X	
7.3 SCANIA	7.3.1	BEV-4		44t GCW	Automotive parts	Scania Logistics	X			X					X	
	7.3.2	FC-2		44t GCW	Temperature-controlled goods	GRU		X		X						X
	7.3.3	FC-2		44t GCW	Temperature-controlled goods	Primafrio		X		X						X
		BEV-4					X			X						X
	7.3.4	BEV-5		44t GCW	Automotive goods	GSS	X			X	X					
				64t GCW			X			X	X	X		X		
7.4 Renault	7.4.1	BEV-6		44t GCW	Automotive components	Renault + Michelin	X			X	X					
	7.4.2	BEV-6		44t GCW	Parcel distribution	DPD	X			X	X					
				64t GCW			X			X	X			X		
7.6 FORD	7.6.1	FC-3		44t GCW	Automotive components	Ekol		X		X	X					
	7.6.2	FC-3		44t GCW	Parcel distribution	GBW		X		X	X					
	7.6.3	FC-3		44t GCW	Partly Hazardous goods	PG		X		X	X					

2.3 Component models for modular ZEV-powertrain concepts

2.3.1 Naming standard

A naming standard was agreed between the partners of ZEFES project, to ensure a standardized process when a new model is shared by a third party and needs to be connected and integrated into the simulation platform in a fast and efficient manner. Developing a unified naming convention also facilitates a quick understanding of the meaning and type of main signals and model parameters, by providing information about:

- The **Component** name to which the variable or parameter belongs to, as ***compulsory field***, to provide information about the signal source (and component instance if we have the case where several instances of the same component are present in the model)
- The **Location** of the variable or parameter, as ***optional field***, including if it belongs to a sub-component, identifier, operator, and direction/condition
- The physical quantity or **Variable** represented, as **compulsory field**, including its attribute and magnitude, and when needed, additional description of the variable
- The **Parameter** that is represented, as ***optional field*** if applies, and should also include the magnitude and type of the parameter
- The **Units** in which the physical quantity or parameter is expressed, as ***compulsory field***

Some real examples of variables and parameters can be found in Table 11, Table 12 and Table 13. For example the name of the output signal "Bat_LimDchgContCurr_A" represents the limit continuous discharge current of the battery where "Bat_" is the component name, "LimDchgCont" specifies the location where the current is considered that is a combination of an operator prefix "Lim" and two direction prefixes "Dchg" and "Cont", "Curr_" specifies the type of magnitude for the variable that is the current and "A" represents the units. The name of the parameter "Bat_LimDchgContCRate_TempBrk2_K" has an additional optional naming section "TempBrk2_" which represents that it is a parameter containing the temperature breakpoints for dimension 2 of the C-Rate table calibration for the battery limits.

The main naming rule or convention used in the standard can be represented as following:

$$\{Component\}\_\{Location\}\{Variable\}\_\{Parameter\}\_\{Units\}$$

Where a **Component** can be:

$$\{Component_{Instance}\} = Component + Identifier$$

$$\{Component\} = Component$$

Table 2: Naming standard for Component field

Component	Keyword	Identifiers	Mandatory (to have at least one component in the platform)
Driver	Drv		X
VCU	VCU		X
Trailer/Semitrailer/Dolly Main Body	Body		X
Axle	Axl	AxlF, AxlR	X
Battery + BMS	Bat		X
Electric Drive Unit (Control+Inv+EM+(Trans))	EDU	EDUF, EDUR, ePTO	X*
Transmission (Reducer, AT, FD…)	Tx	TxF, TxR, TxFDF, TxFDR	
Thermal model	Therm		
Fuel Cell	FC		
H2 Tank	H2Tnk		
Junction Box (junctions, CDCs…)	Jbox		X
Electric Loads (HV and LV consumers)	eLoad		X
Thermal Loads (temperature-controlled volume)	ThLoad		
H2 Refuelling station	H2Refill		
Battery Charging station	Chr		
Test Case	TC		X

**At least one EDU should be included in the model*

In case different instances of a component are available in the model, an additional identifier should be added to the component name to distinguish between the different instances. For example, if two EDUs are used in the tractor, one in the front and another in the rear axle, the correct naming for each instance would be EDUF (Component Name + F, for signalling front axle EDU) and EDUR (Component Name + R, for signalling rear axle EDU).

The **Location** field, which is an *optional field* so is marked in *green* in the main naming rule, can be:

$$\{Location\} = Subcomponent + Identifier + Operator + Direction/Condition$$

Table 3: Naming standard for Location field

Component Category	Subcomponent	Keyword
eLoad	LV DCDC	LVDC
	HV DCDC	HVDC
	LV Battery	LVBat
	LV/HV Consumer	Cons
Body	Gradient slope	Grad
	Road load	RL
	Cargo load	Load
EDU	Control (Powertrain Control Unit)	PCU
	Inverter	Inv
	Electric Motor	EM

Component Category	Subcomponent	Keyword
Axl	Brake	Brk
	Wheel	Whl
	Shaft	Shf
Bat	BMS	BMS
	Dashboard	Dash
FC	Electric Compressor	eComp
	Humidifier	Hum
	Stack	Stack
	Valve	Vlv
	Pump	Pmp
TC	Air	Air
	Road	Road
Drv	Lever (D/R/N/P)	Lvr
	Key	Key
	Accel pedal	Accel
	Brake Pedal	Brk
Therm	Compressor	eComp
	PTC	PTC
	Radiator	Rad
	Pump	Pmp
	Blower	Blw
	Cabin	Cabin
	Refrigerant, Coolant	Refri
	Air	Air
	Refrigerated volume	Frigo
Tx	TCU	TCU
	Shaft	Shf
	Gear	Gear
VCU	Torque Control	TC
	Energy Management	EM
	ABS	ABS
	Adaptative Cruise Control	ACC
	EDU	EDU
	…	…

If there is more than one sub-component in the model available, an additional identifier field should be used to identify the different sub-components that will be used. Therefore, a list of possible identifiers that could be added to the Location field is defined below:

Table 4: Naming standard for Location field – Identifier Examples

Identifier examples	Keyword
Location n Axles	1L, 2L, 3L… (L for left) or 1R, 2R, 3R… (R for right)
Location Twin tyres	I (inner) or O (outer)
Numeric	1,2,3…
Alphabetic	a, b, c…

Table 5: Naming standard for Location field – Operator Examples

Operator examples	Keyword
Absolute	Abs
Average	Avg
COG	COG
Cumulated	Cum
Difference	Dlta
Equivalent	Eq
Exponential average	eAvg
Limit	Lim
Maximum	Max
Minimum	Min
Negative	Neg
Positive	Pos
Root Mean Square	RMS

Table 6: Naming standard for Location field – Direction/Condition Examples

	Direction/Condition	Keyword
Space Direction	Longitudinal	Lon
	Lateral	Lat
	Normal	Norm
	Roll	Roll
	Pitch	Pch
	Jaw	Jaw
Energy Direction	Charge	Chg
	Discharge	Dchg
Space Position	x	X
	y	Y
	z	Z
Limit type	Peak	Peak
	Peak 1	Peak1
	Peak 2	Peak2
	Continuous	Cont
Power type	Electric	Elec
	Mechanic	Mec
	Heat	Heat
	Loss	Loss
	In	In
	Out	Out

The **Variable** field in the naming standard, which is a **compulsory field** (with Attribute and Magnitude as compulsory fields and additional description as optional), can be:

$$\{Variable\} = Attribute + Magnitude + Additional\ description$$

Table 7: Naming standard for Variable field – Attribute List

Attribute	Keyword
Actual	Act
Available	Avail
Estimated	Est
Feedback	Fbk
Flag	Flg
Nominal	Nom
Physical magnitude*	
Residual	Res
Remaining	Rem
Request	Rqt
Target	Tgt

** For physical magnitudes keyword is not needed.*

Table 8: Naming standard for Variable field – Magnitude List

Magnitude	Keyword
Acceleration	Acc
Angle	Ang
Coefficient	Coef
Convection coefficient	hconv
C-rate	Crate
Current	Curr
Capacity	Cap
Cycle	Cyc
Damping coefficient	Dmp
Density	Dns
Distance	Dist
Efficiency	Eff
Energy	Ener
Energy Capacity	ECap
Force	Frc
Friction coefficient	Mu
Gradient	Grd
Humidity	Hum
Inertia	Iner
Irradiance	Irr
length	Len
Linear Velocity	Vel
Mass	Mass
Mass flow	Mflw
Number	Num
Position	Pos

Magnitude	Keyword
Power	Pow
Pressure	Pres
Price	Price
Radius	Rad
Ratio	Rtio
Resistance	R
Rolling resistance	CRR
Rotation inertia	Irot
Rotational acceleration	Aplh
Rotational speed	Spd
Specific heat	Cp
Specific heat capacity	Cp
State	St
State of Charge	SOC
State of Health	SOH
Stiffness	Stfn
Surface	Surf
Temperature	Temp
Thermal resistance	Rth
time	Time
Torque	Trq
Variant	Variant
Voltage	Vlt
Volume	Vol
Volume flow	Vflw

The **Parameter** field in the naming standard, is required only in the case that the signal name represents a parameter. This field of the name completes the information specifying the type of parameter (e.g.: “Cal” for calibrated constant parameters of 1x1 dimension, “Fun” for coefficients of a function, “Val” for look-up table data with N dimension, etc).

$$\{Parameter\} = \{Magnitude\} + Type$$

The different possible Parameter types are shown in Table 9:

Table 9: Naming standard for Parameter field – Type List

Parameter Type	Keyword
Calibration value (independent 1x1)	Cal
Number of table dimensions	Dim
Function calibration value	Fun
Value of LuT	Val
Value of LuT for 4D tables (2D slices)	Val11, Val12...
Breakpoints for dimension 1	Magnitude + Brk1

Parameter Type	Keyword
Breakpoints for dimension 2	Magnitude + Brk2
Breakpoints for dimension 3	Magnitude + Brk3
Breakpoints for dimension 4	Magnitude + Brk4

In case of table data with breakpoints all the parameters that are used to represent the same table share the radical before the parameter field ({Component}_{Location}{Variable}) and have a different ending that specifies their function and units (_{Parameter}_{Units}). In the case of breakpoints of a table the Magnitude in the {Variable} field specifies the type of magnitude of the table. Therefore, it is mandatory to include the Magnitude of the breakpoint as radical in the {Parameter}. The {Unit} field specifies the units of the breakpoint).

To illustrate the use of Parameters, see the following example below to describe the name of parameters for a 1D lookup table of maximum peak traction torque of the EDU as function of EDU speed:

- EDU_LimPosPeakTrq_Dim_int: this parameter defines the number of dimensions that will be active in the model
- EDU_LimPosPeakTrq_Val_Nm: refers to a single value (0D), a vector (1D) or a matrix (2D) of values from a lookup table of EDU maximum torque
- EDU_LimPosPeakTrq_SpdBrk1_rpm: refers to a vector with the breakpoints of the lookup table in the first dimension “Brk1”, in this case the magnitude would be Speed in rpm, for the EDU maximum torque table
- EDU_LimPosPeakTrq_VltBrk2_V: refers to a vector with the breakpoints of the lookup table in the second dimension “Brk2”, in this case the magnitude would be Voltage in V. In this example, if the table data is defined as 1D this field will be set to “null”, otherwise it should contain the breakpoints for the second dimension

The **Units** field in the naming standard, is a **compulsory field**, and typically refers to the unit of the variable or parameter using the international system convention whenever possible, and can be defined as follows:

$$\{Units\} = Units$$

Table 10: Naming standard for Units field

Category	Attribute	Keyword
ACCELERATION	meters per square seconds	mps2
ANGLE	radian	rad
AREA	Square meter	m2
CURRENT	Ampere	A
CONVECTION COEFF	Kilojoule per square meter per Kelvin	kJpm2pK
	Joule per square meter per Kelvin	Jpm2pK
DAMPING	Newton second per meter	Nspm
	Newton second meter per radian	Nsmprad
DENSITY	kilogram per cubic meter	kgpm3
DIMENSIONLESS	Categorical	cat
	Dimensionless factor, coefficient	dl
	Percent value	perc

Category	Attribute	Keyword
	Integer state, position, flag	int
	Boolean	bol
DISTANCE/POSITION	Meter	m
	kilometre	km
ELECTRICAL CHARGE	Ampere hours	Ah
ENERGY	Joule	J
	Watt per hour	Wh
	Kilowatt per hour	kWh
FORCE	Newton	N
FREQUENCY	Hz	Hz
IRRADIANCE	Watt per square meters	Wpm2
MASS	Kilogram	kg
POWER	Watt	W
	Kilowatt	kW
PRESSURE	Pascal	Pa
	Bar	Bar
RESISTANCE	Ohm	Ohm
ROTATIONAL INERTIA	Kilogram square meter	kgm2
SPECIFIC ENERGY	Kilowatt-hour per kg	kWhpkg
	Kilowatt-hour per Liter	kWhpl
SPECIFIC HEAT CAPACITY	Kilojoule per Kilogram per Kelvin	kJpkgpK
	Joule per Kilogram Kelvin	JpkgpK
SPEED (ROTATIONAL)	radian per seconds	radps
	revolutions per minute	rpm
STIFFNESS	newton per meter	Npm
TEMPERATURE	Kelvin	K
	Centigrade	C
THERMAL CAPACITY	Joule per kelvin	JpK
THERMAL RESISTANCE	Kelvin per Watt	KpW
TIME	Seconds	s
	1 / seconds	1ps
	Minute	min
	Hour	h
TORQUE	Newton per meter	Nm
VELOCITY (LONGITUDINAL)	meters per seconds	mps
	kilometres per hour	kmph
VOLTAGE	Volt	V
VOLUME	Cubic meter	m3
	Liter	l
VOLUME FLOW	cubic meter per second	m3ps
	Liter per second	lps

2.3.2 Components interface specifications

Based on the functional requirements of each component, and the relationship among the components connected into the multi-architecture platform, a detailed interface specification is also defined and agreed within WP2 partners for each of the components. Specific tracking interface documents tabulated in Excel Files are available for each component in the project SharePoint an in the Multi-architecture modelling platform.

The purpose is to use this document to track inputs, outputs, and parameters of each component and to keep them updated along the project execution, to ensure model exchangeability between partners and facilitate integration of new models into the simulation platform. Each of the fields of the components also contains a description to ease the understanding in the cases that the signal name is not fully self-explainable.

In the next tables, from Table 11 to Table 13, a detailed example of the interface specifications tracking file is presented in the case of the battery + BMS model (named “Bat” in the naming standard). The tables show the specifications for the inputs, the outputs and the parameters. Similarly, this interface specification tracking file is developed for all the remaining components included in the multi-architecture platform.

Table 11: Example of interface specification file – Input list for “Bat” component (Battery + BMS)

Entity	Port Type		Description	FHG	Generic
Battery + BMS	Input	TC_TimeStep_s	Time Step of the simulation	●	●
Battery + BMS	Input	TC_EnvAirTemp_K	Ambient air temperature	●	●
Battery + BMS	Input	Chr_Pow_kW	Input charging power	●	●
Battery + BMS	Input	Jbox_TotPow_kW	Total consumption from battery	●	●
Battery + BMS	Input	Therm_RefriBatInTemp_K	Inlet coolant temperature	●	●
Battery + BMS	Input	Therm_RefriBatMflw_kgps	Inlet coolant Mass flow	●	●

For the outputs, the file specifies which component models will make use of the output. If no model is specified, the purpose of the output is just reporting or the results.

Table 12: Example of interface specification file – Output list for "Bat" component (Battery + BMS)

Entity	Port Type		Description	FHG	Generic	Destination
Battery + BMS	Output	Bat_LimChgPeakCurr_A	Maximum battery peak current (charge). Interpolated from datasheet at current SOC and T	●	●	
Battery + BMS	Output	Bat_LimChgContCurr_A	Maximum battery continuous current (charge). Interpolated from datasheet at current SOC and T	●	●	
Battery + BMS	Output	Bat_LimChgAvailCurr_A	Maximum battery available current right now (charge). The only one that is effectively applied as a limit in the VCU	●	●	VCU
Battery + BMS	Output	Bat_LimDchgPeakCurr_A	Maximum battery peak current (discharge). Interpolated from datasheet at current SOC and T	●	●	
Battery + BMS	Output	Bat_LimDchgContCurr_A	Maximum battery continuous current Interpolated from datasheet at current SOC and T	●	●	
Battery + BMS	Output	Bat_LimDchgAvailCurr_A	Maximum battery available current right now (discharge). The only one that is effectively applied as a limit in the VCU	●	●	VCU
Battery + BMS	Output	Bat_Curr_A	Total Battery current	●	●	
Battery + BMS	Output	Bat_Vlt_V	DC voltage	●	●	Jbox
Battery + BMS	Output	Bat_ElecPow_W	Output electric power		●	Jbox
Battery + BMS	Output	Bat_PowLoss_W	Battery losses (internal resistance)	●	●	
Battery + BMS	Output	Bat_estSOH_perc	Battery state of health	●	●	
Battery + BMS	Output	Bat_RemCap_Ah	Battery residual capacity considering SOH. ~=Bat_NomCap_Cal_Ah*Bat_SOH_perc/100			
Battery + BMS	Output	Bat_ActCap_Ah	Capacity depending on Temperature, not properly implemented			
Battery + BMS	Output	Bat_DashSOC_perc	Battery state of charge. This is the reported SOC to the dashboard (goes from 100% to 0%), not the internal SOC	●	●	
Battery + BMS	Output	Bat_BmsSOC_perc	Battery state of charge. Internal SOC from the BMS	○	●	VCU
Battery + BMS	Output	Bat_DchgCap_Ah	Dischargable current capacity. ~=(Bat_ResCap_Ah-Not_usable_Ah-Temp_Los_Ah) *Bat_DashSOC_perc/100	●	●	
Battery + BMS	Output	Bat_ChgCap_Ah	Chargable current capacity. ~=(Bat_ResCap_Ah-Not_usable_Ah-Temp_Los_Ah)-Bat_DchgCap_Ah	●	●	
Battery + BMS	Output	Bat_EnvHeatPow_W	Battery heat loss to the environment	●	●	
Battery + BMS	Output	Bat_ThermHeatPow_W	Battery heat cooling to the thermal module	○	●	Therm
Battery + BMS	Output	Bat_AvgMaxTgtTemp_K	High limit for battery target temperature (information for cooling system). Datasheet parameter	●	●	
Battery + BMS	Output	Bat_AvgMinTgtTemp_K	Lower limit for battery target temperature (information for cooling system). Datasheet parameter	●	●	
Battery + BMS	Output	Bat_AvgTemp_K	Battery average temperature	●	●	
Battery + BMS	Output	Bat_RefriOutTemp_K	Output temperature of the coolant	●	●	
Battery + BMS	Output	Bat_RefriLimMflw_kgps	Limit for the coolant mass flow. Datasheet paramenter	●	●	
Battery + BMS	Output	Bat_RefriCp_JpkgpK	Refrigerant specific heat capacity in J/(kg·k). Datasheet parameter	●	●	
Battery + BMS	Output	Bat_R_Ohm	Battery internal resistance	●	●	
Battery + BMS	Output	Bat_Mass_kg	Battert mass. Datasheet parameter	●	●	Body
Battery + BMS	Output	Bat_NomECap_kWh	Energy capacity of the battery in nominal conditions	●	●	

For the parameters, the file specifies which of them are parametrizable in the detailed 3rd party model and in the generic mathematical model. In the case of look-up-tables it also specifies the number of dimensions of input data that can be handled by the model. In some cases, it is possible to choose between 0D, 1D and 2D data (for example, for the battery resistance). In other cases, it is possible to choose between 2D, 3D and 4D data but 0D and 1D options are not valid (for example, for the EDU efficiency map).

The parameter table also contains the information of the scaling formulas. The component models permit the users to configure all the parameters, but this level of parametrization is a drawback when performing parametric studies and optimizations because several parameters need to be modified when changing the size of the component. This drawback is overcome by the scaling factors. These factors can be easily tuned to change the size of a component in comparison to a reference calibrated model, and the rest of parameters will be recalculated with the scaling formula specified in the component file.

Table 13: Example of interface specification file – Parameter list for "Bat" component (Battery + BMS)

Entity	Port Type		Description	FHG	Generic	Resize formula (only generic model)	Valid dimensions
Battery + BMS	Parameter	Bat_Variant_Cal_cat	Battery Variant	○	○		
Battery + BMS	Parameter	Bat_IniDashSOC_Cal_perc	Initial SOC in the dashboard	●	●		
Battery + BMS	Parameter	Bat_AvgIniTemp_Cal_K	Initial average temperature	●	●		
Battery + BMS	Parameter	Bat_IniBmsSOH_Cal_perc	Initial SOH	●	●		
Battery + BMS	Parameter	Bat_NomVlt_Cal_V	Nominal voltage.		●	*Bat_SeriesScaleFactor_Cal_dl	
Battery + BMS	Parameter	Bat_NomCap_Cal_Ah	Nominal capacity.	●	●	*Bat_ParalellScaleFactor_Cal_dl	
Battery + BMS	Parameter	Bat_Cap_Dim_int	Battery Capacity number of dimensions	○	●		[0,1]
Battery + BMS	Parameter	Bat_Cap_Val_Ah	Capacity table	○	●	*Bat_ParalellScaleFactor_Cal_dl	
Battery + BMS	Parameter	Bat_Cap_TempBrk1_K	Capacity table temperature brk	○	●		
Battery + BMS	Parameter	Bat_VltOCV_Dim_int	Battery OCV voltage table number of dimensions	○	●		[0,1,2]
Battery + BMS	Parameter	Bat_VltOCV_Val_V	Battery OCV voltage table	○	●	*Bat_SeriesScaleFactor_Cal_dl	
Battery + BMS	Parameter	Bat_VltOCV_BmsSOCBrk1_perc	Battery OCV SOC BRK (Internal SOC from BMS)	○	●		
Battery + BMS	Parameter	Bat_VltOCV_TempBrk2_K	Battery OCV Temp BRK	○	●		
Battery + BMS	Parameter	Bat_R_Dim_int	Battery resistance table number of dimensions	○	●		[0,1,2]
Battery + BMS	Parameter	Bat_R_Val_Ohm	Battery resistance table	○	●	/Bat_ParalellScaleFactor_Cal_dl	
Battery + BMS	Parameter	Bat_R_BmsSOCBrk1_perc	Battery resistance SOC BRK (internal SOC from the BMS)	○	●		
Battery + BMS	Parameter	Bat_R_TempBrk2_K	Battery resistance Temp BRK	○	●		
Battery + BMS	Parameter	Bat_ChgPeakTime_Cal_s	Battery peak time charge	○	●		
Battery + BMS	Parameter	Bat_DchgPeakTime_Cal_s	Battery peak time discharge	○	●		
Battery + BMS	Parameter	Bat_LimCRateDt_Cal_1ps	Battery limit to the derivative of the C-rate	○	●		
Battery + BMS	Parameter	Bat_LimChgPeakCRate_Dim_int	Battery limit charge peak C-rate number of dimensions	○	●		[0,1,2]
Battery + BMS	Parameter	Bat_LimChgPeakCRate_Val_dl	Battery limit charge peak C-rate table	○	●	*Bat_ParalellScaleFactor_Cal_dl	
Battery + BMS	Parameter	Bat_LimChgPeakCRate_BmsSOCBrk1_perc	Battery limit charge peak C-rate SOC Brk (internal SOC from the BMS)	○	●		
Battery + BMS	Parameter	Bat_LimChgPeakCRate_TempBrk2_K	Battery limit charge peak C-rate Temp Brk	○	●		
Battery + BMS	Parameter	Bat_LimChgContCRate_Dim_int	Battery limit charge cont C-rate number of dimensions	○	●		[0,1,2]
Battery + BMS	Parameter	Bat_LimChgContCRate_Val_dl	Battery limit charge cont C-rate table	○	●	*Bat_ParalellScaleFactor_Cal_dl	
Battery + BMS	Parameter	Bat_LimChgContCRate_BmsSOCBrk1_perc	Battery limit charge cont C-rate SOC Brk (internal SOC from the BMS)	○	●		
Battery + BMS	Parameter	Bat_LimChgContCRate_TempBrk2_K	Battery limit charge cont C-rate Temp Brk	○	●		
Battery + BMS	Parameter	Bat_LimDchgPeakCRate_Dim_int	Battery limit discharge peak C-rate number of dimensions	○	●		[0,1,2]
Battery + BMS	Parameter	Bat_LimDchgPeakCRate_Val_dl	Battery limit discharge peak C-rate table	○	●	*Bat_ParalellScaleFactor_Cal_dl	
Battery + BMS	Parameter	Bat_LimDchgPeakCRate_BmsSOCBrk1_perc	Battery limit discharge peak C-rate SOC Brk (internal SOC from the BMS)	○	●		
Battery + BMS	Parameter	Bat_LimDchgPeakCRate_TempBrk2_K	Battery limit discharge peak C-rate Temp Brk	○	●		
Battery + BMS	Parameter	Bat_LimDchgContCRate_Dim_int	Battery limit discharge cont C-rate number of dimensions	○	●		[0,1,2]
Battery + BMS	Parameter	Bat_LimDchgContCRate_Val_dl	Battery limit discharge cont C-rate table	○	●	*Bat_ParalellScaleFactor_Cal_dl	
Battery + BMS	Parameter	Bat_LimDchgContCRate_BmsSOCBrk1_perc	Battery limit discharge cont C-rate SOC Brk (internal SOC from the BMS)	○	●		
Battery + BMS	Parameter	Bat_LimDchgContCRate_TempBrk2_K	Battery limit discharge cont C-rate Temp Brk	○	●		
Battery + BMS	Parameter	Bat_LimMinBmsSOC_Cal_perc	Battery minimum SOC limit for internal SOC	○	●		
Battery + BMS	Parameter	Bat_LimMinBmsSOC_Cal_perc	Battery maximum SOC limit for internal SOC	○	●		
Battery + BMS	Parameter	Bat_NomCycLife_Cal_int	Battery nominal life cycles	○	●		
Battery + BMS	Parameter	Bat_ThermFlg_Cal_bol	Defines if thermal model is active to calculate battery temperature	○	●		
Battery + BMS	Parameter	Bat_Mass_Cal_kg	Battery Mass	○	●	Bat_SeriesScaleFactor_Cal_dl	
Battery + BMS	Parameter	Bat_AvgMaxTgtTemp_Cal_K	High limit for battery target temperature (information for cooling system). Datasheet parameter	○	●		
Battery + BMS	Parameter	Bat_AvgMinTgtTemp_Cal_K	Lower limit for battery target temperature (information for cooling system). Datasheet parameter	○	●		
Battery + BMS	Parameter	Bat_EqCp_Cal_JpkgpK	Battery equivalent specific heat	○	●		
Battery + BMS	Parameter	Bat_Eqhconv_Cal_Wpm2pK	Battery equivalent convection coefficient	○	●		
Battery + BMS	Parameter	Bat_EqSurf_Cal_m2	Battery equivalent surface	○	●	*sqrt(Bat_ParalellScaleFactor_Cal_dl* Bat_SeriesScaleFactor_Cal_dl)	
Battery + BMS	Parameter	Bat_RefriLimMflw_Cal_kgps	Limit for the coolant mass flow. Datasheet paramenter	○	●		
Battery + BMS	Parameter	Bat_RefriCp_Cal_JpkgpK	Refrigerant specific heat capacity in J/(kg·k). Datasheet parameter	○	●		
Battery + BMS	Parameter	Bat_SeriesScaleFactor_Cal_dl	Factor to resize the battery with Cells in Paralell	○	●		
Battery + BMS	Parameter	Bat_ParalellScaleFactor_Cal_dl	Factor to resize the battery with Cells in Series	○	●		

2.3.3 Model requirements

Each of the component models needs to comply with the following requirements to be integrated in the platform:

- Compliance with the interface standard file
- Compliance with the functional requirements list for the components that contains the agreements of all the functions that the model will performed and the assumptions of the model
- Developed MATLAB/Simulink 2020a
- The individual model must be validated by the model supplier after the compilation to guarantee that the model performance is maintained after compilation
- The models need to be delivered in a Simulink model that includes a model test bench (inputs and outputs) that permits to execute a validation simulation before integration in the platform

2.3.4 Individual component models

This section specifies the details of the main component models conforming the multi-architecture platform making special emphasis in the detailed 3rd party models that provide additional value to the platform by including aging functions or model calibration with data from real experiments or 1D models.

2.3.4.1 Battery model

In this section of the deliverable, we describe the battery model developed by FHG for the powertrain simulation and optimisation toolchain of IDI and VUB. The battery model is developed in MATLAB/Simulink and converted into an S-function to become suitable for the easy integration with the work of other consortium partners. In the generated S-function block, the user can choose the battery cell chemistry (Lithium Iron Phosphate (LFP) or Nickel Manganese Cobalt (NMC)), initial state of charge (SoC), initial state of health (SoH), initial temperature, the nominal capacity, and nominal voltage of the battery as the block parameters. According to the given capacity and voltage, the number of cells in series and parallel is calculated. Due to this setup parameters can easily be adjusted by the user of the entire simulation tool.

2.3.4.1.1 Methodology

The overall battery model can be divided into the following sub parts:

1. Electrical equivalent model (EEM) of the battery
2. Thermal model
3. SoH estimator
4. SoC estimator
5. Current limiter

Electrical Equivalent Model of Lithium-ion Batteries

An equivalent model of Li-ion battery helps in predicting its performance under different working conditions. Additionally, the equivalent model is a vital part in estimating SoX (state of X where X=charge, health) and vice versa. A good Li-ion battery model is attributed by high accuracy along with low computational complexity. Among the different modelling techniques, electrical modelling is very popular for the anticipation of the electrical behaviour of the Li-ion batteries and is shown in Figure 3.

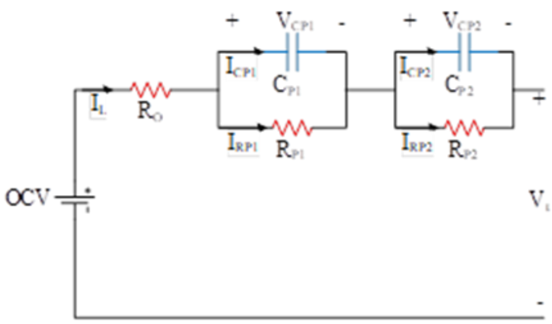


Figure 3: Electrical equivalent model of lithium-ion battery

The model consists of one ideal voltage source, one ohmic resistance, and parallel RC (resistor-capacitor) branches connected in series. The capacitance and the resistance in the RC branch represent the polarization of the electrode and the electrode-electrolyte contact resistance, respectively. The ohmic series resistance depicts the charge transfer phenomenon inside the electrolyte. In Figure 3, R_0 is the internal resistance, R_{p1} and R_{p2} are the polarization resistance, C_{p1} and C_{p2} are the polarization capacitance, I_L is the loading current (positive for discharging and negative for charging), and V_{cp1}, V_{cp2}, and V_t denote the polarization voltage and terminal voltage, respectively. OCV is the open circuit voltage of the cell which is a function of SoC. The governing equation for terminal voltage in this electrical model is given as follows:

$$V_t = OCV(SOC,T) - I_L R_0(SOC,T) - V_{cp1} - V_{cp2}$$

The dynamic equations that describe the voltage across the parallel RC branch shown in Figure 3 are presented as follows:

$$\frac{dV_{cp1}}{dt} = -\frac{1}{R_{p1}C_{p1}}V_{CP1} + \frac{1}{C_{p1}}I_L$$

$$\frac{dV_{cp2}}{dt} = -\frac{1}{R_{p2}C_{p2}}V_{CP2} + \frac{1}{C_{p2}}I_L$$

In general, higher number of RC branches provide better accuracy, but at the cost of higher computational complexity. Since OCV (open circuit voltage) vs SoC characteristics for LFP cell is flat in mid SoC range compared to NMC, LFP needs a more accurate model compared to NMC for better depiction of its behaviour. Hence, to develop EEM of LFP cell in this work, two parallel RC are branches are used, whereas only one parallel RC branch is used for NMC cell. It is considered that the values of EEM parameters (resistances and capacitances) are varying with temperature and SoC. The values of resistances and capacitance for each SoC and temperature breakpoints are stored in a lookup table and used to get its value at each instants considering the temperature and SoC at that instant.

Thermal Model of Lithium-ion Batteries

Since all EEM parameters are temperature-dependent, the evolution of temperature of the battery needs to be modelled. Moreover, the safety, performance, and durability of Li-ion batteries are very sensitive to temperature which necessitates efficient thermal model. In our work, we use a 0-D thermal model of Li-ion battery. The governing equations of the thermal model are:

$$T_b\,(k) = T_b(k-1) + \frac{1}{m}(P_{net}(k))\,\Delta T$$

$$P_{net}(k) = P_g(k) - \dot{Q}_a(k) - \dot{Q}_c(k)$$

$$\dot{Q}_a(k) = \dot{Q_{cn}}(k) + \dot{Q_{rd}}(k)$$

$$\dot{Q_{rd}} = c_{rd} * \sigma_{rd} * \left(T_b^4 - T_a^4\right)$$

$$\dot{Q_{cn}} = c_{cn} * (T_b - T_a)$$

$$\dot{Q}_c = \dot{M}_f * C_p * (T_{out} - T_{in})$$

where, T_b= battery temperature, T_a= ambient temperature, m = thermal mass, P_{net}= total power, ΔT= sampling time, P_g= battery heat losses due to internal resistances, $\dot{Q}_a$= battery heat loss to the environment, $\dot{Q}_c$= battery heat loss to coolant, $\dot{Q_{cn}}$= heat loss due to convection, $\dot{Q_{rd}}$= heat loss due to radiation, c_{rd}= coefficient of radiation, σ_{rd}= Stefan-Boltzmann constant, c_{cn}= convection coefficient, $\dot{M}_f$= coolant mass flow, C_p= specific heat, T_{out}= coolant output temperature, T_{in}= input coolant temperature.

For different battery temperature, the mass flow is controlled using different weights along with the input mass flow rate at each temperature. The cooling circuit is turned on when the battery temperature is higher or lower than a threshold temperature. The coolant circuit is turned off when the battery temperature again reaches to another threshold temperature.

Estimation of State of Health

The state of health can be related to irreversible degradation of the battery. It provides remaining life and allows the users to compare the current condition of the battery with the new one (beginning of life). The degradation is either represented by capacity loss or increase in internal resistance of the battery. In this work, the current nominal capacity is considered as indicator of SoH and defined as:

$$SoH = \frac{C_{nom,current}}{C_{nom,new}} * 100$$

Where, $C_{nom,new}$ is the nominal capacity of the new cell while $C_{nom,current}$ is the present maximum capacity calculated under nominal conditions (residual capacity). The cell is called to reach its end of life (EOL) when the current nominal capacity reduces to a certain level (usually 80%) of that of the new cell. The degradation of a battery depends on various factors such as depth of discharge (DOD), operating temperature, C-rate, and SoC.

In this work, two empirical cycling degradation models designed for NMC and LFP Li-ion battery chemistries in [1] are used. These degradation models in [1] are built upon the data from 232 degradation tests for NMC and 85 degradation tests for LFP. To estimate the SoH, a Loss is calculated considering the relevant factors and the relevant stress factors into account and then subtracted from the starting value of 100.

$$SOH\ (\%) = 100 - Loss$$

Unlike [1], where is SoH only starts from 100%, our model is valid for any initial SoH. We have converted the initial SoH to the equivalent consumed number of cycles considering the nominal conditions (temperature, DOD, SoC, charging and discharging C-rates) as given in datasheet for which the nominal cycle life is specified.

Based on the SoH, the battery residual current capacity and Energy capacity of the battery in nominal conditions are calculated [2]. Battery residual current capacity considers the degradation of the battery and is defined as the current maximum capacity (Ah) of the battery which can be extracted (100% actual SoC to 0 % actual SoC) under nominal conditions (1 C rate, 298.5 K). It can be calculated by simply multiplying Nominal Capacity to SoH. The Energy capacity of the battery in nominal conditions is calculated by simply multiplying Nominal Energy capacity to SoH.

Estimation of State of Charge

The information on state of charge (SoC) is crucial to a battery model. SoC is defined as the ratio of remaining capacity to the current maximum capacity. Current maximum capacity is the maximum charge which the cell can store in the present condition. Mathematically, SoC is defined as:

$$SoC = \frac{C_{rem}}{C_{max}} * 100$$

where C_{rem} and C_{max} are the remaining and current maximum capacity, respectively.

For estimation of SoC, the coulomb counting method is used. The governing equation for coulomb counting method is given as:

$$SOC(k) = SOC(k-1) + \frac{\eta_c \Delta T I_L(k)}{3600 C_{max}}$$

where ΔT is the sampling time, I_L is the terminal current, and η_c is the Coulombic efficiency. The Coulombic efficiency depends on the temperature as well as the mode of operation (charging or discharging). The capacity of Li-ion battery varies with temperature and ageing conditions. In general, lower temperatures lead to a decrease in capacity. In this work, the above effects are taken care of while calculating the SoC.

In this work, for safe operation of the battery, we have restricted the usage of the cell between 98% to 5% of actual SoC. So, the 98-100% and 0-5% SoC range is considered as not usable. Based on this restriction of usable capacity, the dashboard SoC is calculated which varies from 0-100%. The user can see only the dashboard SoC. However, for all internal calculation, we have used internal SoC.

Based on the dashboard SoC, a few outputs of the s-function are defined. The dischargeable and chargeable current capacity are most important of them. The dischargeable current capacity is defined as the remaining usable capacity (current dashboard SoC to 0% dashboard SoC). It can be calculated by simply multiplying Usable Capacity to Dashboard SoC. The chargeable current capacity is determined by subtracting dischargeable current capacity from the Usable capacity. All the considerations considered to calculate these outputs are depicted in the following figure.

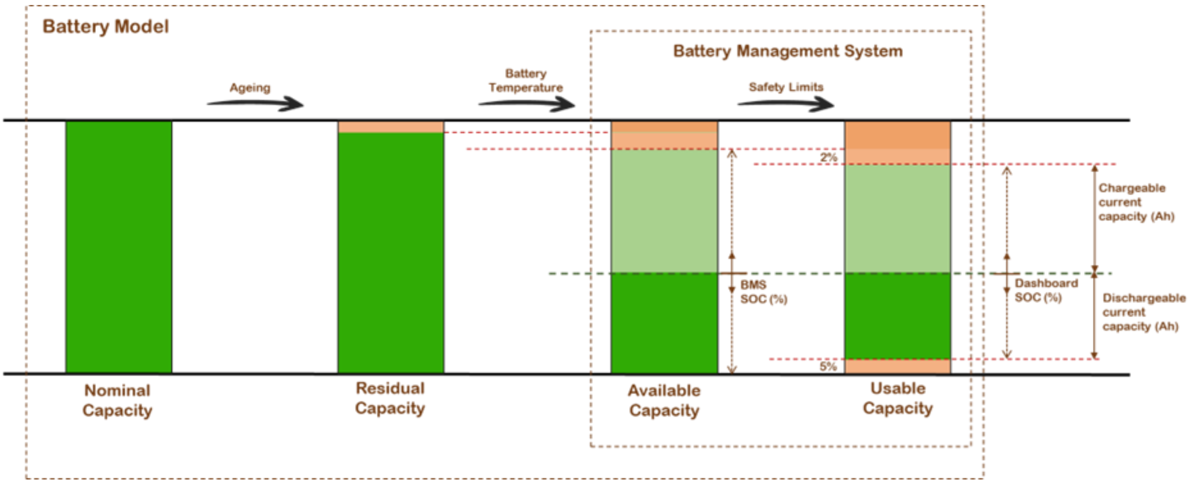


Figure 4: Schematics of considerations to calculate the battery dashboard SOC

Current Limiter

The charge/discharge peak/continuous current limits from the datasheet for each SoC and temperature breakpoints are stored in a lookup table and used to get the limits at each instants considering the temperature and SoC at that instant. Due to increase in the resistance with ageing of the battery, the actual continuous current limits are calculated using the following:

$$Present\ allowable\ max\ dischrge\ current = max\ (max\ discharge\ current\ , \frac{Voltage - V_{cut\_off}}{R_0})$$

$$Present\ allowable\ max\ charge\ current = min\ (max\ charge\ current, \frac{V_{peak} - Voltage}{R_0})$$

The max discharge/charge currents are continuous current limits from datasheet considering present SoC and temperature.

2.3.4.1.2 Battery cells specifications

The detailed battery model developed by FHG is calibrated to two specific NMC and LFP cell chemistries. The specifications of the NMC and LFP cells used in this work are given in Table 14 and Table 15 respectively:

Table 14: Battery NMC cell specification

Characteristic	Value
Nominal capacity	31 Ah
Nominal voltage	3.6 V
Nominal charging C-rate	1 C
Nominal discharging C-rate	1 C
Cycle life (1 C-rate, 298.15 K, 50% SoC, 80% DoD)	2600
Battery mass	445 g
Nominal energy	111.6 Wh
Cut-off voltage	3 V
Peak voltage	4.2 V
Maximum continuous discharge current (50% SoC, 298.15 K)	5 C
Maximum continuous charge current (50% SoC, 298.15 K)	2 C
Maximum peak discharge current (50% SoC, 298.15 K, 10 s)	10 C
Maximum peak charge current (50% SoC, 298.15 K, 10 s)	4 C
Internal resistance (50% SoC, 298.15 K)	8.4 mΩ

Table 15: Battery LFP cell specification

Characteristic	Value
Nominal capacity	4.2 Ah
Nominal voltage	3.2 V
Nominal charging C-rate	1 C
Nominal discharging C-rate	1 C
Cycle life (1 C-rate, 298.15 K, 50% SoC, 80% DoD)	7500
Battery mass	72 g
Nominal energy	13.44 Wh
Cut-off voltage	2.75 V
Peak voltage	3.65 V
Maximum continuous discharge current (50% SoC, 298.15 K)	10 C
Maximum continuous charge current (50% SoC, 298.15 K)	4 C
Maximum peak discharge current (50% SoC, 298.15 K, 10 s)	20 C
Maximum peak charge current (50% SoC, 298.15 K, 10 s)	8 C
Internal resistance (50% SoC, 298.15 K)	5.6 mΩ

2.3.4.1.3 Model structure

In this section, first the input, output, and parameters of the detailed battery model are discussed. After that, the overall interface of the simulation is shown.

Input/Output/Parameter of the detailed battery model

Table 16: List of inputs, outputs and parameters of the detailed battery model

Port Type	Nomenclature	Description
Input	TC_EnvAir_Temp_K	Ambient air temperature
Input	Chr_Pow_kW	Input charging power
Input	Jbox_TotPow_kW	Total consumption from battery
Input	Therm_RefriBatInTemp_K	Inlet coolant temperature
Input	Therm_RefriBatMflw_kgps	Inlet coolant Mass flow
Output	Bat_LimChgPeakCurr_A	Maximum battery peak charging current
Output	Bat_LimChgContCurr_A	Maximum battery continuous charging current
Output	Bat_LimChgAvailCurr_A	Maximum battery available charging current at present
Output	Bat_LimDchgPeakCurr_A	Maximum battery peak discharge current
Output	Bat_LimDchgContCurr_A	Maximum battery continuous discharging current
Output	Bat_LimDchgAvailCurr_A	Maximum battery available discharge current at present
Output	Bat_Curr_A	Total Battery current
Output	Bat_DashSOC_perc	Battery dashboard state of charge
Output	Bat_SOH_perc	Battery state of health
Output	Bat_Vlt_V	DC voltage
Output	Bat_PowLoss_W	Battery losses due to internal resistances
Output	Bat_EnvHeat_W	Battery heat loss to the environment
Output	Bat_ResCap_Ah	Battery residual current capacity
Output	Bat_DchgCap_Ah	Dischargeable current capacity
Output	Bat_ChgCap_Ah	Chargeable current capacity
Output	Bat_AvgMaxTgtTemp_K	High limit for battery target temperature
Output	Bat_AvgMinTgtTemp_K	Lower limit for battery target temperature
Output	Bat_AvgTemp_K	Battery average temperature
Output	Bat_RefriOutTemp_K	Output temperature of the coolant
Output	Bat_RefriLimMflw_kgps	Limit for the coolant mass flow
Output	Bat_RefriCp_JpkgpK	Refrigerant specific heat capacity in J/(kg·K)
Output	Bat_R_Ohm	Battery actual resistance
Output	Bat_Mass_kg	Battery mass
Output	Bat_NomECap_kWh	Energy capacity of the battery in nominal conditions
Parameter	Battery_cell_type	Selection of the battery cell type
Parameter	Bat_AvgIniTemp_Cal_K	Initial average temperature
Parameter	Bat_IniDashSOC_Cal_perc	Initial SoC in the dashboard
Parameter	Bat_IniSOH_Cal_perc	Initial SoH
Parameter	Bat_NomCap_Cal_Ah	Nominal capacity
Parameter	Bat_Nom Volt_V	Nominal voltage

Table 16 specifies the input, outputs and parameters of the detailed battery model. As the model is already calibrated and validated for two specific cell chemistries, all the model parameters are automatically derived from a reduced amount of user inputs.

The overall interface of the s-function is depicted in the following figure.

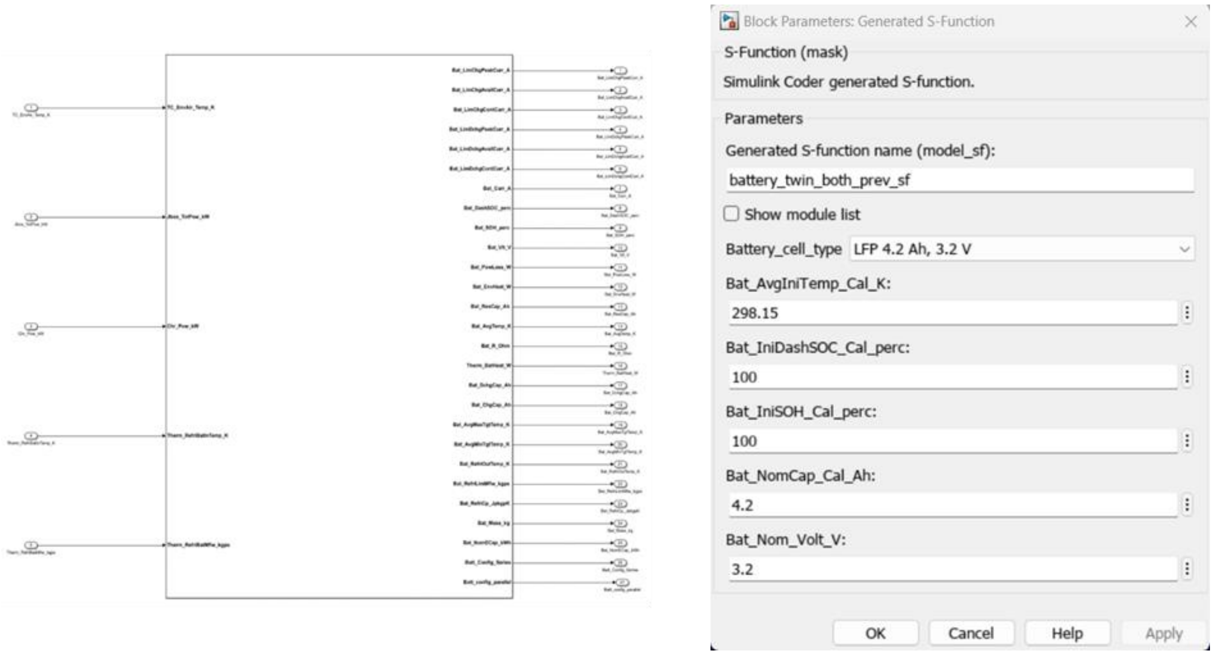


Figure 5: Overall interface of the detailed battery S-function

In the generated S-function block, the user can choose the battery cell chemistry (Lithium Iron Phosphate (LFP) or Nickel Manganese Cobalt (NMC)), initial state of charge (SoC), initial state of health (SoH), initial temperature, the nominal capacity, and nominal voltage of the battery as the block parameters.

2.3.4.2 Fuel cell BoP model

In this section of the deliverable, we describe the fuel cell BoP (balance of plant) model developed by RIC. For this purpose, Ricardo developed a standalone generic 1D fuel cell stack efficiency and BoP model in GT-SUITE. The 1D model was used to run Design of Experiments (DoE) covering an agreed-upon range of inputs and to collect the steady state results for a list of outputs. These results were then used to generate nD-maps for each output variable to generate a ROM (Reduced Order Model). The reduced-order Simulink model utilizes the generated nD-maps for predicting output variables. This section report describes the list of inputs and outputs, as well as some model functionality and development methodology.

2.3.4.2.1 Methodology

GT-SUITE 1D model description

The Hydrogen Fuel Cell System (HFCS) 1D model performed in GT-SUITE consists of Ricardo-developed generic Fuel Cell Stack physical model and Balance of Plant components. These components are supplying air and hydrogen and ensure that FC Stack works in conditions which are within recommended ranges.

The Balance of Plant includes:

- Cathode circuit

- Anode circuit
- H2 supply circuit
- Simplified Deionized (DI) Cooling circuit
- Control units

Highlighted below are the model's key features, summarizing its strengths and limitations, categorized accordingly. All assumptions are based on Ricardo's experience in modelling hydrogen FC systems.

- 1D model comprises FC Stack, Anode, Cathode, and Coolant circuits into a single model, enhancing modelling accuracy compared to separate circuit models.
- FC Stack features include:
  - Polarization curve fitted to test data.
  - Consideration of electro-osmotic drag, back-diffusion, and nitrogen crossover effects.
- Cathode eCompressor, anode blower and Coolant Pump performance maps are based on physically valid data.
- Pipes, valves, manifolds, and other gas flow components (Intake air filter, charge air cooler, humidifier, FC stack anode and cathode internal flow paths, water separator, etc.) geometry and resulting pressure drops were assumed.
- Thermal effects:
  - Pipes and manifolds heat transfer through the walls to ambient was neglected.
  - HFCS has no external heat transfer via radiation or convection considered.
  - Water vapor condensation and liquid water evaporation are enabled at the cathode and anode stack outlet manifolds and the latent heat is released entirely to the fluid.
- The water separator sub-model removes only the liquid phase of water.
- PI-controllers are tuned to regulate the operation of eCompressor motor, exhaust control valve, anode blower motor, and H2 supply valves to achieve air and hydrogen FC Stack stoichiometry and pressure targets under varied operating conditions.
- Power losses of eCompressor motor and inverter are modelled based on real-world data, blower power losses are estimated based on typical blower performance.
- Anode purge subsystem is modelled to control the H2 concentration in the anode circuit to target.
- The coolant circuit ensures both FC stack coolant inlet temperature and temperature delta across FC stack meet their targets. HFCS cooling circuit interaction with vehicle cooling circuit is simulated in Simulink model, reducing the number of inputs required for DoE study.

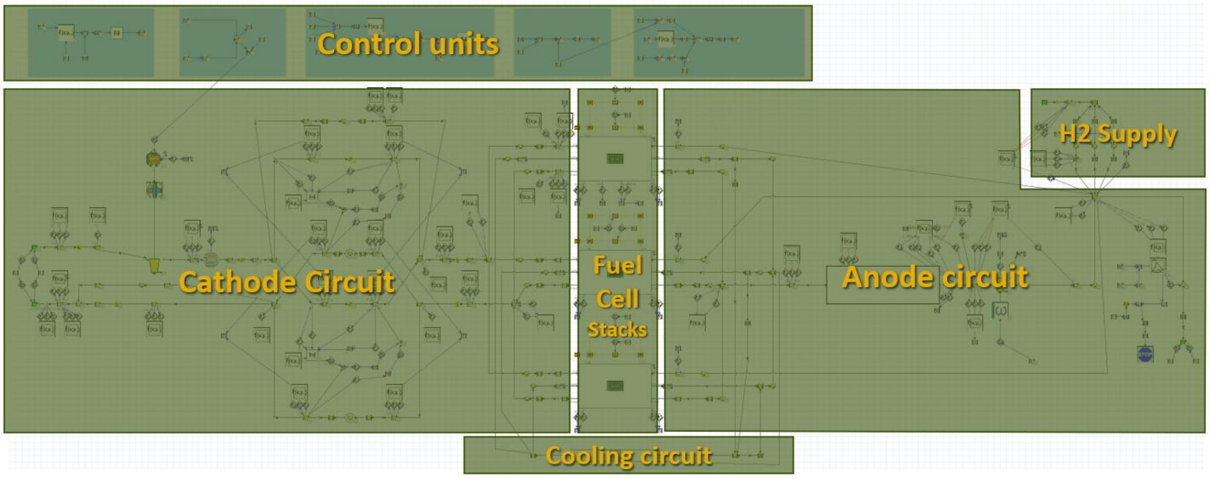


Figure 6: HFCS GT-SUITE 1D model overview

Simulink reduced-order model (ROM) description

The steady state output data from the 1D model's Design of Experiments (DoE) was utilized to construct nD maps, establishing outputs as functions of four key inputs: net current demand, ambient humidity, ambient temperature, and ambient pressure. These nD maps were then integrated into the Simulink model.

The Simulink ROM offers remarkably quicker run time compared to its 1D counterpart, enabling the partners to model numerous scenarios and configurations of fuel cell trucks, whilst maintaining accurate predictions of fuel cell performance, H2 consumption, efficiency, and other key metrics.

The Simulink model supports scalability, enabled by the adjustment of the rated net power demand. This feature empowers users to model various powertrain configurations of fuel cell trucks.

To ensure the effective heat transfer between the DI-coolant and vehicle coolant, a power de-rate mechanism was incorporated into the Simulink model. This mechanism guarantees that the actual heat rejection consistently exceeds the required heat rejection, defined by the corresponding nD-map output, thereby ensuring the FC Stack coolant inlet temperature target is consistently met.

The actual heat transfer is computed within the Simulink model using the NTU-method, which leverages vehicle coolant inputs, DI-coolant outputs, and the effectiveness of the FC/Vehicle heat exchanger.

The de-rated power demand serves as a feedback signal to the vehicle, indicating that the requested load is unattainable due to the thermal system limitations. Meanwhile, Simulink model provides FC system results that are based on the achievable level.

2.3.4.2.2 Model structure

Inputs from the vehicle model to the Simulink FC model

Table 17 shows the list of inputs and parameters for the ROM FC model.

Table 17: List of inputs and parameters of the ROM FC BoP model

Inputs	Limit Range	Units	Notes
Rated net power demand	-	kW	240 kW is default, can be scaled up or down
Net power demand	1 to 240 kW	kW	The range is default, can be scaled together with Rated net power demand
Ambient pressure	0.70134 to 1.01325	bar. a	Corresponds to the altitude range 3000 to 0 m at ambient T = 15 ° C
Ambient RH	40 to 100	%	
Ambient temperature	-40 to +50	° C	
Vehicle coolant mass flow rate	-	kg/s	
Vehicle coolant inlet T	-	° C	
Vehicle coolant heat capacity	-	J/(kg*K)	
FC/Vehicle heat exchanger effectiveness	-	%	

Two constraints were implemented to define the ambient conditions ranges for the generated Design of Experiments (DoE) cases used to create the ROM as depicted in Figure 7. Deviation from the actual results of the 1D model may increase when using inputs outside of these defined ranges.

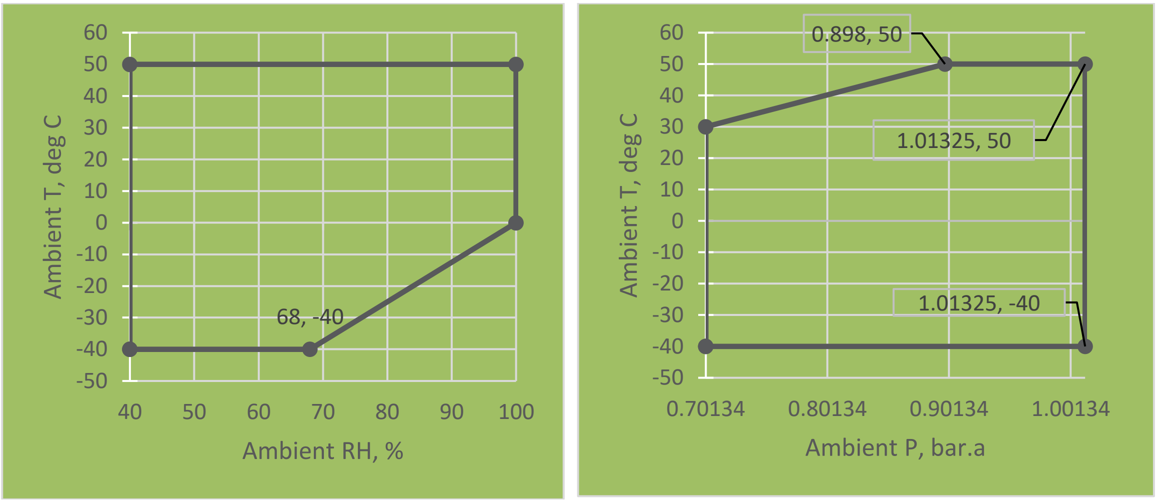


Figure 7: Ambient conditions limits for the FC ROM model

Outputs

Table 18: List of outputs of the ROM FC BoP model

Group	Outputs	Units
FC System	BoP components Power	kW
	Actual system Gross Power	kW
	Actual system Net Power	kW
	Stacks' internal power losses = Fuel Power – electrical power	kW
	System Current	A
	System Voltage	V
	H2 consumption	g/s

Group	Outputs	Units
	Stack Electrical (Fuel) Efficiency	%
	System Net efficiency (LHV)	%
	Stack Cathode inlet RH w.r.t. Coolant inlet temperature	%
	Stack Anode inlet RH w.r.t. Coolant outlet temperature	%
	Stack operating temperature	° C
Cooling system	DI-coolant mass flow	kg/s
	DI-coolant T upstream to Vehicle/FC Heat exchanger	° C
	Heat capacity of the DI-coolant	J/(kg*K)
	Vehicle coolant outlet temperature (downstream FC/vehicle heat exchanger)	° C
	WCAC heat flux to vehicle coolant	kW
	eCompressor heat flux to vehicle coolant	kW
	Anode Blower heat flux to vehicle coolant	kW
	De-rated net power (due to insufficient cooling)	kW
Heat transfer	Exhaust heat at the Cathode outlet	kW
	Actual heat transferred from Deionized to Vehicle coolant	kW

2.3.4.3 Tyre model

The goal is to provide a real time model able to predict wear and rolling resistance as a function of usage information. We intend to keep it as simple as possible so that the model performs fast enough to achieve real time computation.

2.3.4.3.1 Hypothesis

Since we need a real time model, it has been necessary to make several hypotheses to ease the computation and have a fast model.

- Uniform wear
- Homogeneous temperature on the tire
- Linearity between sliding and wear

2.3.4.3.2 Tire performances in the model: wear and rolling resistance

Wear is produced due to the sliding between the tire and the ground. We can use Archard's approach to model wear based on sliding.

Rolling resistance is meanly due to 2 phenomena: energy dissipation as heat due to the deformation of the viscous materials (rubber) and energy dissipation when sliding in the contact between the tire and the ground.

Both the sliding length and the volumetric energy dissipation (heat) can be computed through finite element simulations.

However, the tire evolves as it rolls:

- Due to the volumetric energy dissipation, the tire heats up and temperature increases.

- Due to the temperature difference, the internal air temperature increases, and so does the inflation pressure
- As the tire wears out, the tread depth decreases, and since there is less rubber, the volumetric energy dissipation decreases.

Therefore, wear depends on rolling resistance and rolling resistance depends on wear, and they should be computed together. An additional equation is required to take this into account:

$$RR = f(F_x, F_y, F_z, P_{inf}, V, T_{amb}, TD_{t-1})$$

$$T_{tire}, P_{inf} = f(RR, T, V)$$

$$TD_t = f(Fx, Fy, Fz, P_{inf}, V, T_{tire}, TD_{t-1})$$

Table 19: MICH tyre model variables

Parameter	Description
RR [kg/t]	Rolling resistance
P_{inf} [bar]	Tire inflation pressure
V [km/h]	Vehicle velocity
T_{amb} [°C]	Ambient temperature
T_{tire} [°C]	Tire mean temperature
TD [mm]	Thread depth
F_Z [kg]	Tire load
F_x [kg]	Longitudinal effort
F_y [kg]	Lateral effort

We have decided to model wear and rolling resistance based on finite element simulations. Finite elements seem to be the most adequate reference since it is possible to vary the usage variables and get reproductible results.

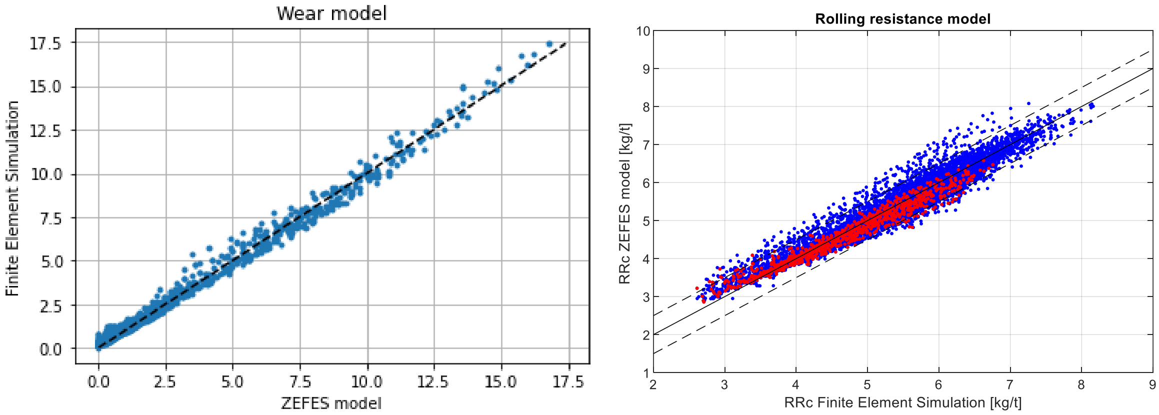


Figure 8: Tyre model correlation between finite elements and Simulink model.

Over 2000 simulations per tire model have been carrried out to make sure all of the functionning points are covered. The quality of the fit is very good with a R2 greater than 0.9.

A catalogue of 5 tires with different rolling resistance and wear performances is provided:

Table 20: MICH tyre catalogue included on the compiled model.

Parameter	Tire 1	Tire 2	Tire 3	Tire 4	Tire 5
Initial Tread depth [mm]	17.5	15	18.5	18.5	17
RR	Medium	Top	Top	Medium	Low
Lifespan	Top	Medium	Medium	Medium	Low
Price	Ref	-4%	-5%	-8%	-11%

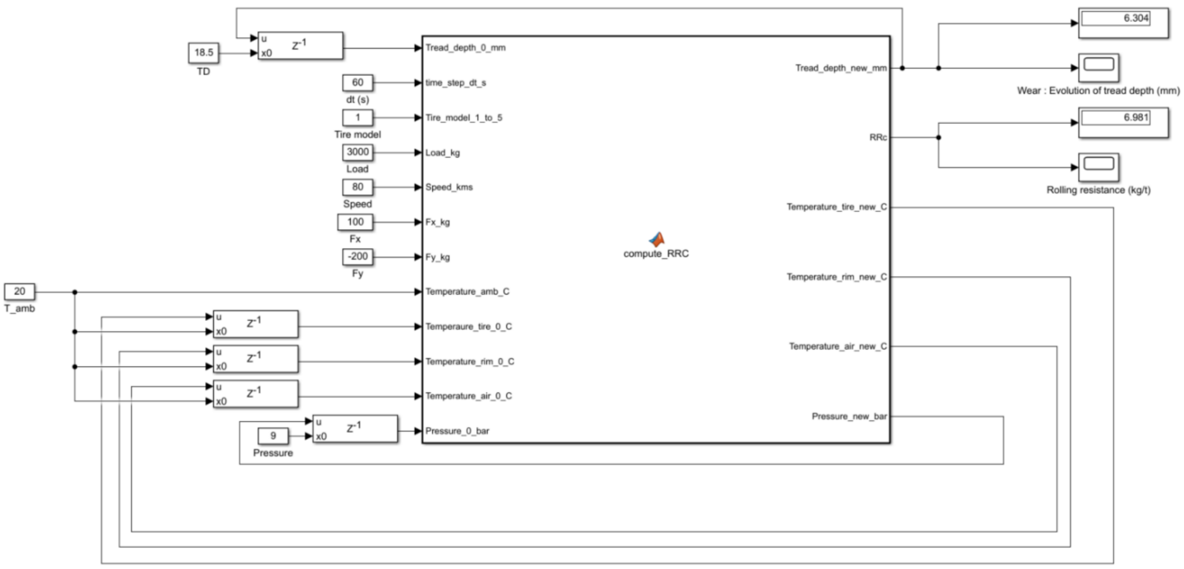


Figure 9: Simulink preview of the MICH tyre model.

2.3.4.3.3 Performance of the model

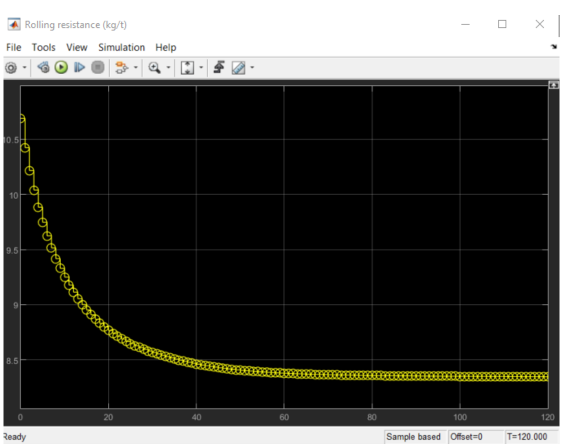
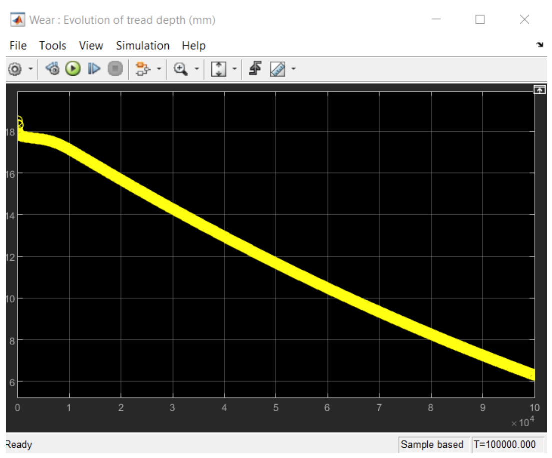
Rolling resistance	Wear
	

Figure 10: Rolling resistance and tyre wear time plots.

The typical computation time on standard computers is as follows:

- Uncompiled: Around 0.08s uncompiled
- Compiled: Around 0.05s

The measured computation times satisfies the requirements of the project to run along all the other models.

2.3.4.4 VCU model

In this section of the deliverable, the main characteristics of the VCU are described. Regarding this component, its function on the simulation model is to integrate the vehicle high level control of the powertrain and energy management. The VCU has two main functionalities: torque distribution among front and rear axles and split power control (SPC).

2.3.4.4.1 Force distribution among axles

The VCU defines the torque distribution among the front and rear drives and transmissions, taking onto account the components' efficiency and inertias and considering the EDUs torque maps and the mass of the vehicle. Together with this, the brake activation and the amount of braking torque requested to each section are considered, dividing the negative torque between regeneration torque and mechanical brakes. To do this, the VCU considers whether exist or not EDUs on the frontal and rear sides, the efficiencies of each of the powertrain systems, the mass distribution among the vehicle and the force request to calculate an optimal force distribution for each timestep.

Regarding the regenerative brake calculations, the minimum torque appliable by the EDUs is considered, and corresponding restrictions to these torques are applied, having a balance of required force to apply to the vehicle, which is then completed with the application of the mechanical torques. To do this, the maximum regeneration capacity of the battery together with the consumers on each timestep are also considered.

2.3.4.4.2 Split Power Control for Fuel Cell Vehicles

Regarding the energy management, the VCU controls the high-level management and balance of powers of the electric and thermal signals given by the components. To do this, it adjusts the thermal and electric power demand of all the components to guarantee the power delivery constraints derived from the current limitations of the battery and the boundaries of the FC, applying traction, PTO, thermal and ancillary electrical limitations to the components' requests if needed, thus achieving a feasible simulation that do not overcome the physical limitations of the battery and FC systems.

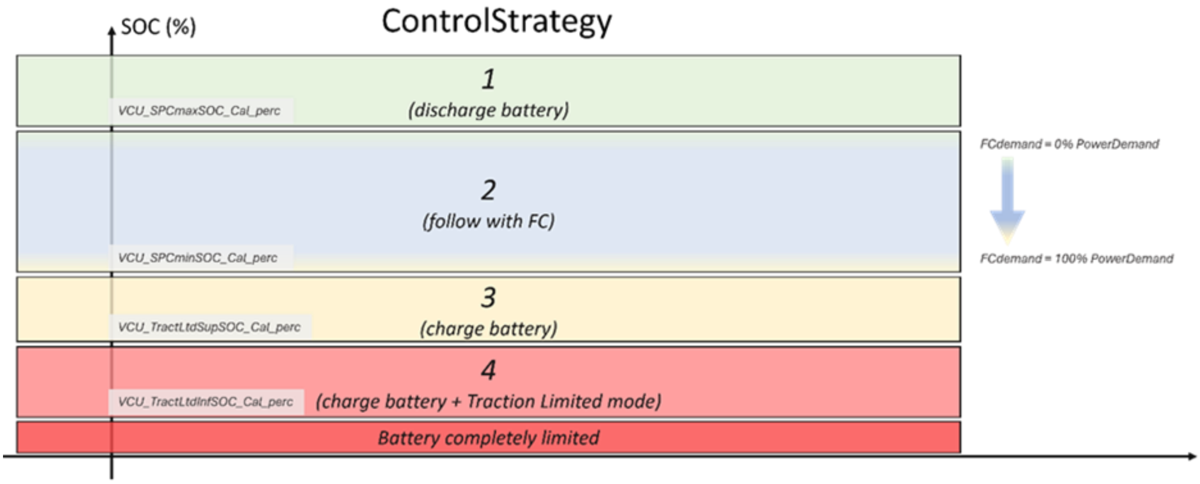


Figure 11: FC role given by SPC on different points of operation.

The SPC unit applies a strategy to divide the required power generation between FC and battery, considering the battery current availability limitations and the FC maximum and minimum requestable powers in each timestep of the simulation, together with the availability of H2 in the H2 tanks and the SoC of the battery.

The strategies are tuneable via parameters that are open to the user from parametrization of the VCU definition input, using thresholds that change the role of the FC, aiming to discharge the battery, charge the battery, or keep the battery SoC stable and use the FC to follow the power demand fully or partially.

These three main power division strategies change with the SoC evolution and have as objective to have a battery SoC high enough to cover more range availability and flexibility, while having enough available capacity to charge the regenerated energy on the ranges of negative forces applied to the vehicle dynamics.

2.3.4.5 EDU

The electric drive units (EDUs) are the components that transform electric power into mechanic power, which is later transmitted via the transmission components, axles and wheels. The dynamics of the motor have some considerations:

- The available torque depends on the instant speed of the EDU shaft, which depends on the truck and drive line inertia and the applied torques on previous instants.
- The torque is limited with maximum and minimum torque in each instant. The torque boundaries are calculated each timestep using the exponential average torque and comparing it with peak and continuous torques.
- The rotor inertia is considered, and part of the indicated torque is used for accelerating its own rotational inertia.
- Internal losses are considered as efficiency, which are defined with LuTs, which can be of different dimensions:
  - 0D: The efficiency is a constant applied to the mechanical power.
  - 2D: The efficiency matrix is a 2D vector that depends on speed and torque.
  - 3D: The efficiency matric is a 3D vector that depends on speed, torque and voltage.
  - 4D: The efficiency considers speed, torque, voltage and temperature.

- Thermal considerations are included in a simple thermal model used to calculate EDU internal temperature.

2.3.4.6 Transmission system

The transmission system is the model for transmitting the power from the EDUs to the traction axles. In this general vehicle model, it is divided into two components: transmission main reduction (Tx) and final drive (TxFD). Both components are modelled the same, and they are connected in series, making it useful for transmission ratio sizing or BM with multiple ratios, permitting a constant final drive ratio with the same efficiency for all combinations in a simulation batch.

The torque going through the drive line from the EDU is reduced in each of the transmission components increasing it in exchange of angular speed. The power that goes through the transmission components has efficiency considerations and rotational inertia is added to the drive system, which affects the accelerations and decelerations.

2.3.4.7 Axles

The axles model takes as input the power coming from the transmission and distributes it among the traction wheels of the truck. This component also aggregates the wheels that don't have traction functionalities. It adds the axles inertia and losses modelled in 0D, i.e. a constant efficiency, and adds inertia, increasing the rotational inertia of the vehicle.

The hydraulic braking system is included in the axles, so this model also takes the brake request from the VCU and applies it.

2.3.4.8 Body

The body component aggregates all the masses of the module and all the external and internal forces coming from every element on the model. In this component, the rolling resistance, aerodynamic resistance and gradient force are calculated for the current module. The functionality of this model is thus calculating every interaction with factors external to the module, aggregate them and make them usable for all the calculations in VCU and other components. The force transmission among modules is also calculated here.

2.4 Multi-architecture modelling platform

2.4.1 Simulation model architecture

The architecture of the truck for the simulation platform consists of a generic model which contains all the possible architecture and module combinations of the requestable components (150% architecture model concept). This modular truck can have up to 4 different modules: tractor, first semitrailer, dolly and second semitrailer and each of them includes the implementation of the most complex architecture which consist of 4WD traction supported by a fuel cell system, ePTO and refeer.

This approach is named 150% vehicle architecture after the HIFI_ELEMENTS project because there will never be a physical vehicle with 4 modules equipped with the most complex FCEV powertrain architecture on each of them, but it permits to simulate almost any choice of modules and architecture combinations by switching off components in the simulation platform. For example, if we disenable the fuel cell system of a trailer we have a e-trailer, if we disenable the EDUs we have a b-trailer and if we disenable the battery and the rest of electric components we have a passive trailer.

In each assembly that is configured by the user for simulation it is possible to define the architecture that will be simulated in that case and the sub-models and parametrizations for each of the selected components.

Thus, the simulation platform is prepared to enable or disable the subcomponents requested by the user, and non-used components of the full model will be either bypassed or disabled depending on the nature of each component. In Figure 12 shows a model schematic where the different modules of the truck are defined with all the selectable components that the user can specify for each module.

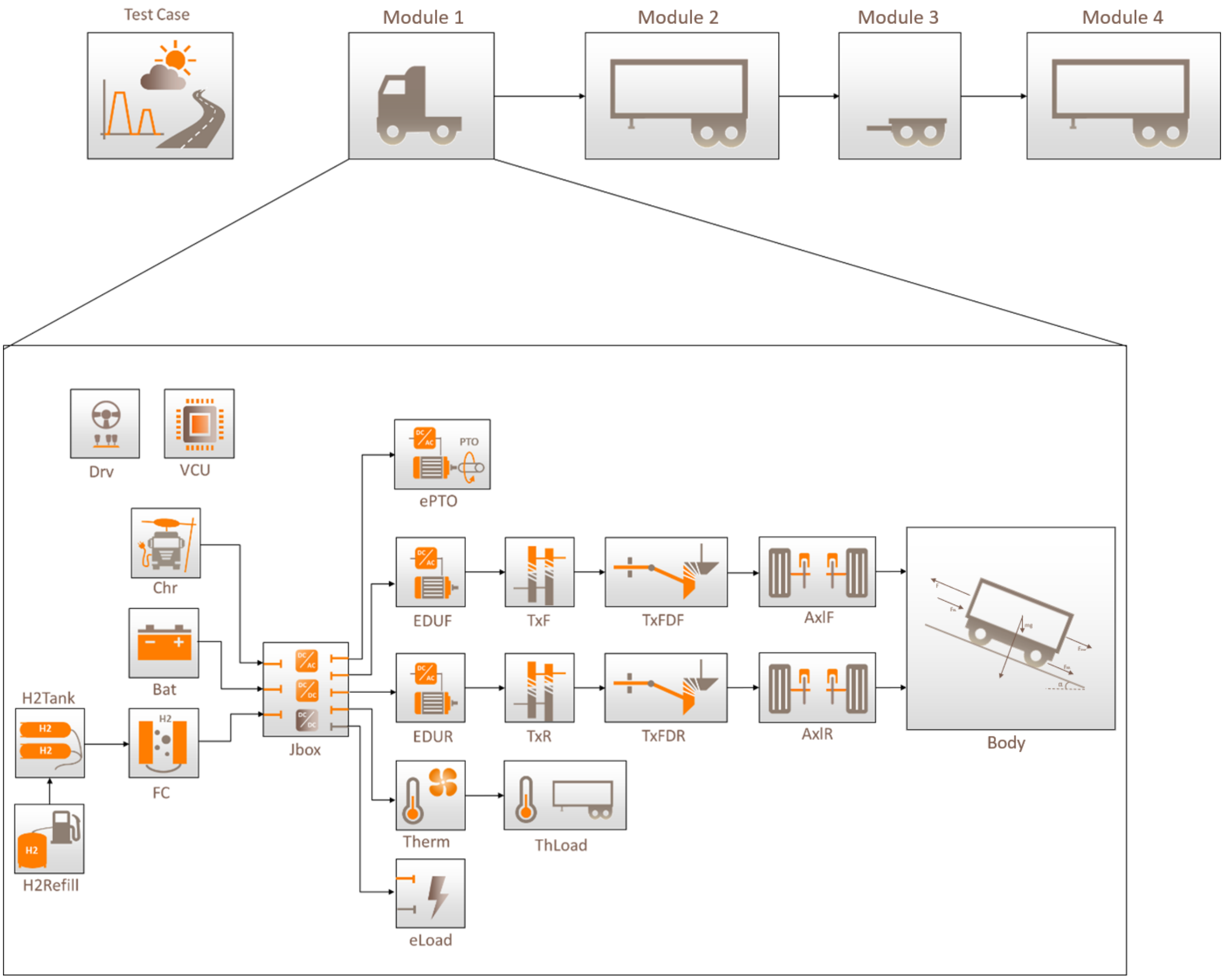


Figure 12: Schematics of the truck model with its subcomponents.

As can be seen, the subcomponents of the vehicle follow the ones defined on the naming convention, and consist of powertrain units from energy storage in batteries (Bat) and H2 tanks (H2Tank), to fuel cell (FC), a power take-off (PTO), thermal control system (Therm), electric ancillary consumers (eCons), e-drive units (EDU) in front axles and rear axles, transmission (Tx, TxFD) and the vehicle body (Body) which includes inertias, road load and gradient slope calculation together with the body dynamics. Also, there are the driver (Drv) unit which defines the torque to request needed to cover the target of the test case, and a vehicle control unit (VCU) that controls and integrates all the components making them compatible with each other. There is also a junction box (Jbox) that unifies all the electricity connections with the corresponding connection efficiencies in DC/DC or DC/AC conversions.

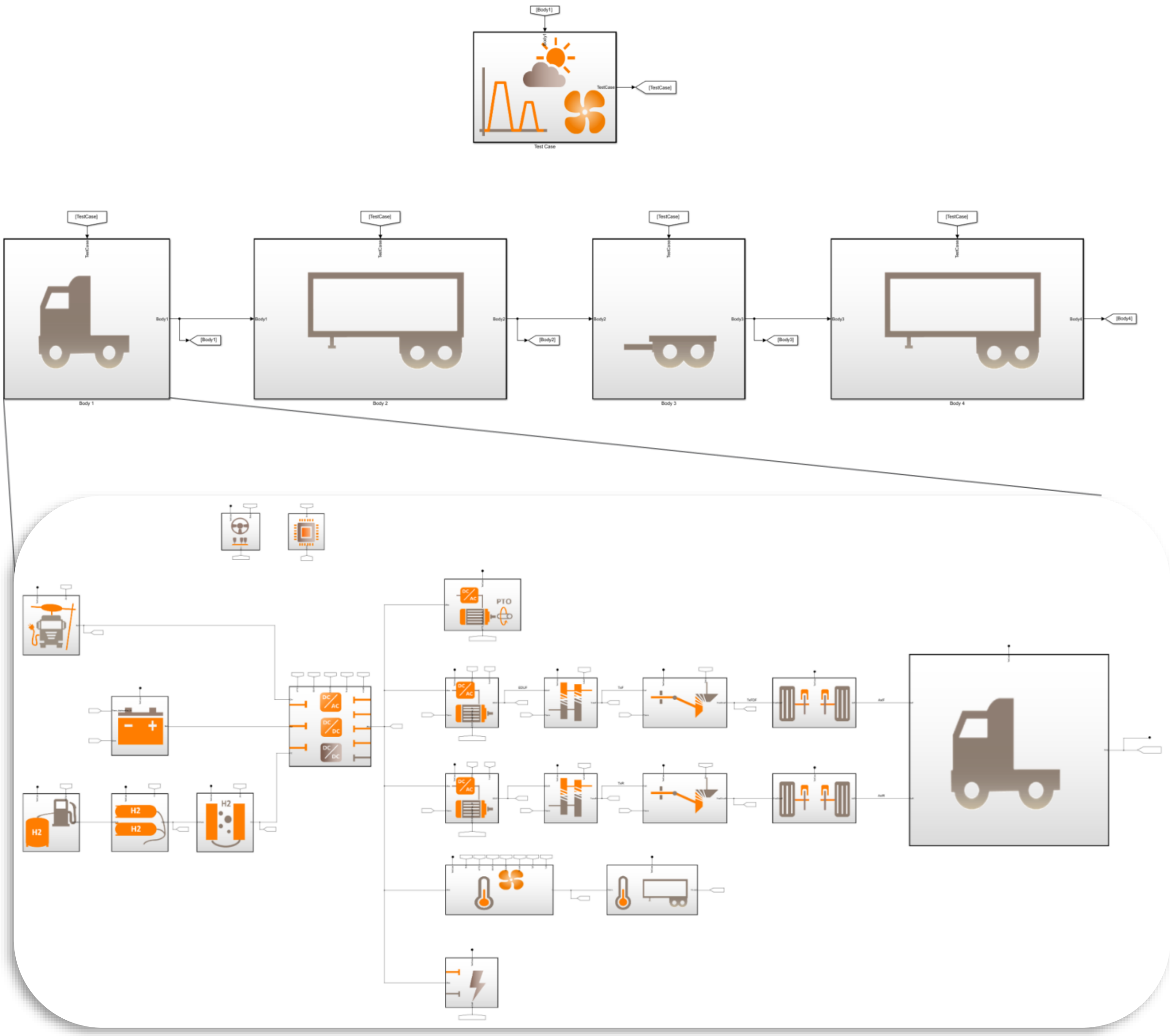


Figure 13: Simulink model of the truck with its subcomponents for each module

2.4.2 Multi-architecture implementation

2.4.2.1 Component variants

Regarding the component's selection, an example can be seen in Figure 14. For each of the components, there is a selector that allows the user to choose either a *null* component (non-existent, will be bypassed or removed), or a component which is available on the module. In the case of the battery, the user has the option to choose the IDI generic model of the battery, or the more precise model given by FHG, and this block distribution will adapt the component to the distribution chosen by the platform user.

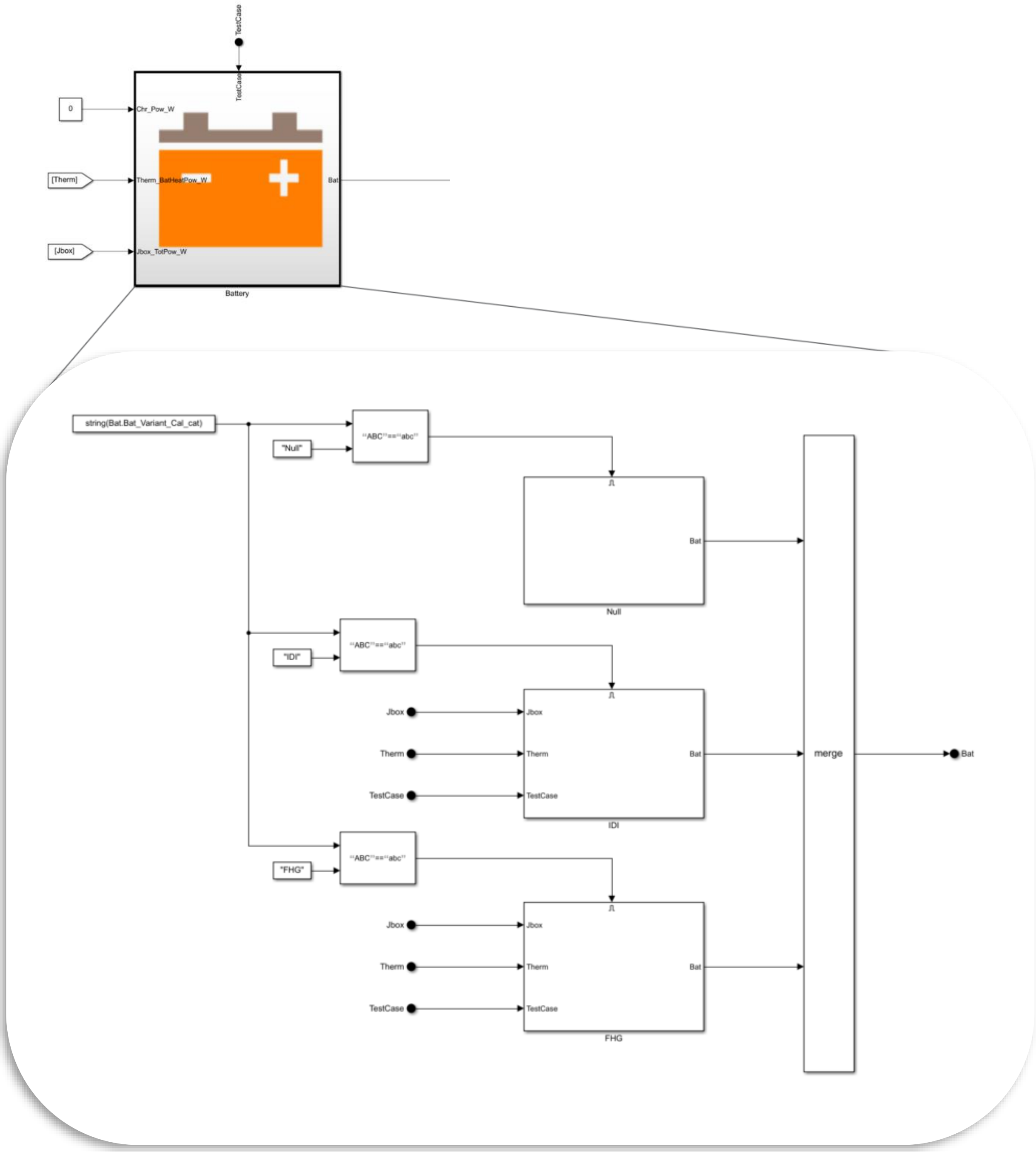


Figure 14: Simulink model of the model selector for the battery

2.4.2.2 Configuration of specific architectures

The configuration of specific architectures is performed by activation and deactivation of components of the complete multi-architecture model. When the user selects the *null* variant for a component or module, the *null* variant is activated in Simulink. The *null* variant sets the signal values and routes the signals of the *null* model to emulate that the model is not present in the architecture.

This permits to simulate numerous combinations of architectures though activations and deactivations of modules and components.

In a similar manner, inside each module it is possible to activate and deactivate components to simulate different topologies:

- A full BEV or a FCEV by activation and deactivation of the FC, H2 tank and H2 refill.
- A plug-in or non-plug-in battery by activation and deactivation of the charger.
- Every combination of front and rear wheel drive can be considered via activation or deactivation of EDUs, thus allowing also non-conventional other than rear wheel drive distribution.
- The availability of ePTO by activation or deactivation of the ePTO component and its power request from the test case definition.
- Different amount of reduction steps by activation and deactivation of front and rear transmissions and front and rear final drives. Also, both transmission and final drive can be deactivated in the case of an eAxle in which the supplier provides the final drive efficiency together with the EDU efficiency map.
- Climate control volume (such as goods or cabin) by activation and deactivation of the thermal load.
- Semitrailers which have a king pin instead of an equivalent frontal axle, by deactivation of the frontal axle.

Besides, different topologies of the front and rear equivalent axles are also possible but will be specified by parameters and not by activating and deactivating components. Each equivalent axle will permit to parametrize:

- Single, tandem or tridem axles
- Single or twin tyres
- Tractive and not tractive axles
- Liftable and not liftable axles

The following figures illustrate with an example the model architecture that would be configured to simulate the Use Case 7.2.1 which consists of a FCEV rigid truck with an eDolly and an eSemiTrailer.

Truck: FCEV	R+TR (Swedish EMS1)
Distance: 4,800km/week (2 x 340km x 5days/week)	March 2025 – February 2026
Shipper: OVAKO	Steel scrap
Cross border: No	Gothenburg to Hofors

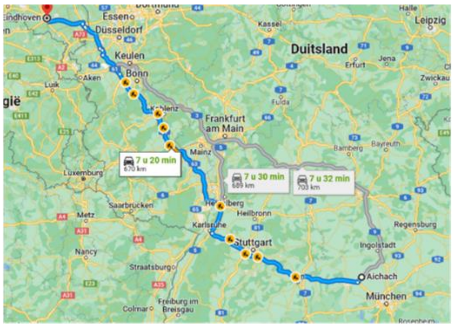


Figure 15: Example of a Use Case definition

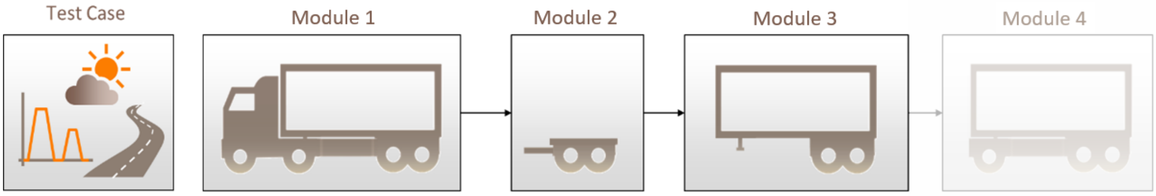


Figure 16: Main module simulation structure for Use Case 7.2.1

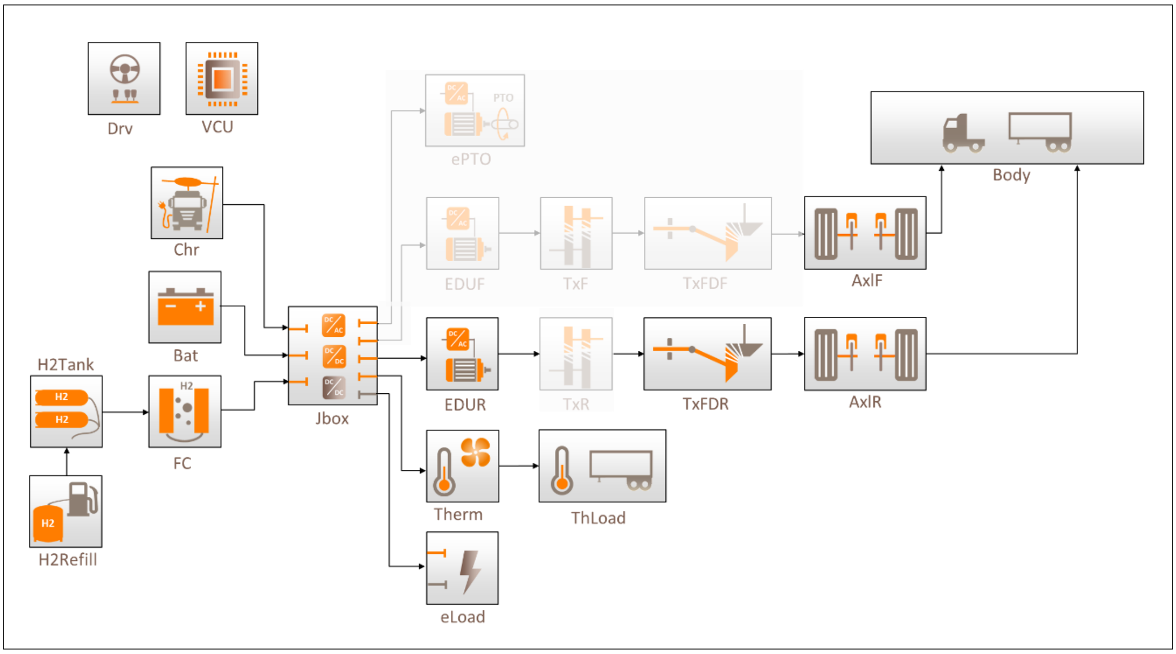


Figure 17: Module 1 (FCEV rigid truck) simulation structure for Use Case 7.2.1

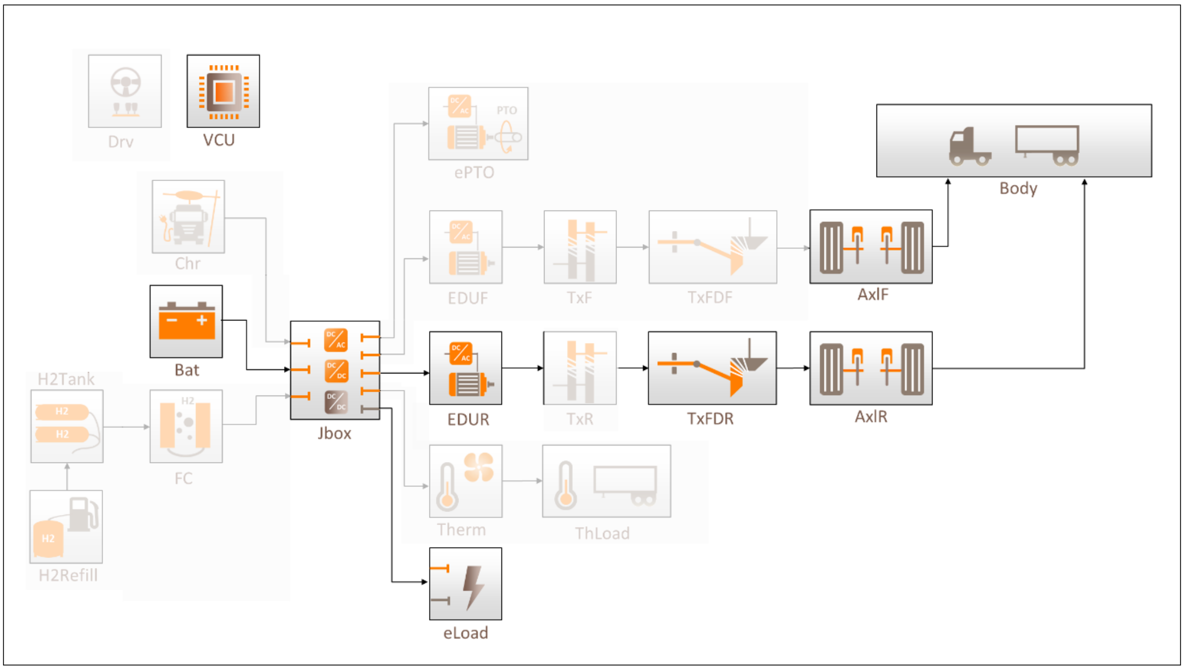


Figure 18: Module 2 (e-dolly) simulation structure for Use Case 7.2.1

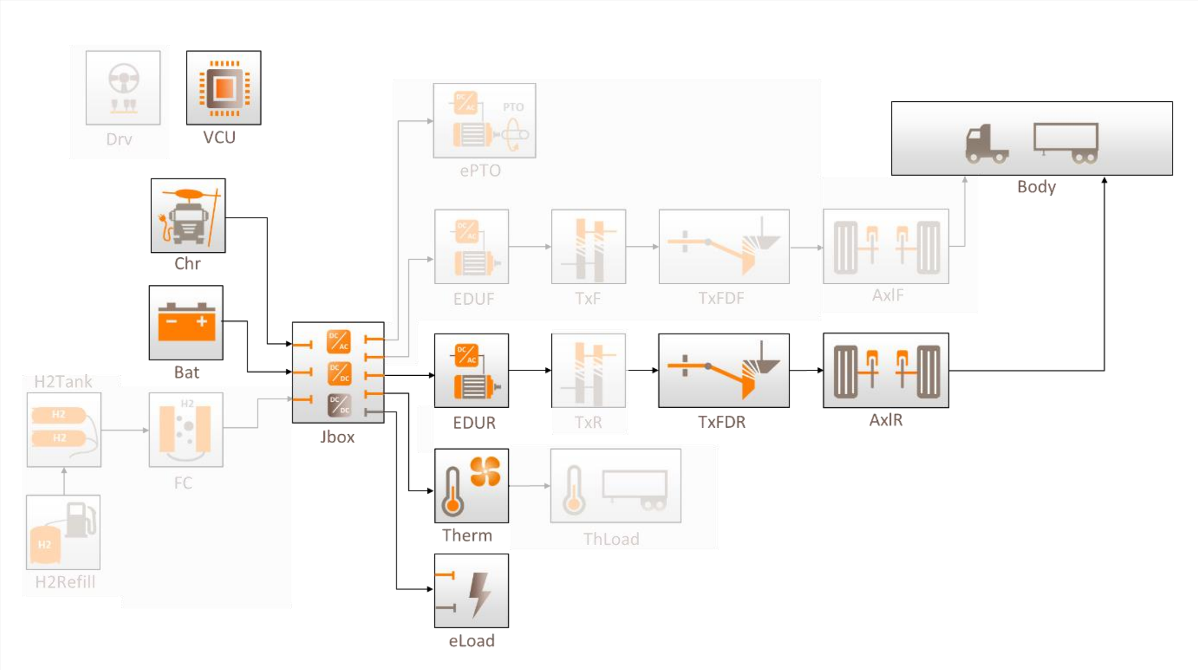


Figure 19: Module 3 (e-semitrailer) simulation structure for Use Case 7.2.1

As an example, given the component definition for the simulation model, some of the demonstrators have been defined in

				Drv	Body	VCU master	VCU slave	AxlF King pin	AxlF Single	AxlF Tandem	AxlF Tridem	AxlR Single	AxlR Tandem	AxlR Tridem	Bat	EDUF	EDUR	ePTO	TxF	TxR	TxFDF	TxFDR	Therm	FC	H2Tnk	H2Refill	Jbox	eLoad	ThLoad	Chr
	7.2.2	BEV-1	Body 1	X	X	X			X				X		X		X			X		X	X				X	X	X	X
			Body 2		X									X																
			Body 3																											
			Body 4																											
	7.2.1	FC-1	Body 1	X	X	X			X				X		X		X			X		X	X	X	X	X	X	X	X	
			Body 2		X								X																	
			Body 3		X								X																	
			Body 4																											
	7.3.2	FC-2	Body 1	X	X	X			X				X		X		X			X		X	X	X	X	X	X	X	X	
			Body 2		X		X							X	X		X			X		X	X				X		X	X
			Body 3																											
			Body 4																											
	7.3.4	BEV-5	Body 1	X	X	X			X				X		X		X			X		X	X				X	X	X	X
			Body 2		X										X												X			X
			Body 3																											
			Body 4																											
	7.4.2	BEV-6	Body 1	X	X	X			X				X		X		X			X		X	X				X		X	X
			Body 2		X							X																		
			Body 3		X								X																	
			Body 4		X								X																	
	7.6.1	FC-3	Body 1	X	X	X			X				X		X		X			X		X	X	X	X	X	X	X	X	
			Body 2		X									X																
			Body 3																											
			Body 4																											

Figure 20 regarding the presence or not of each of the available components in each of the modules. It can be noted that each of modules can have all of the combinations for their subassembly components, thus including the possibility to have a wide variety of distributions thanks to the modular definition.

				Drv	Body	VCU master	VCU slave	AxlF King pin	AxlF Single	AxlF Tandem	AxlF Tridem	AxlR Single	AxlR Tandem	AxlR Tridem	Bat	EDUF	EDUR	ePTO	TxF	TxR	TxFDF	TxFDR	Therm	FC	H2Tnk	H2Refill	Jbox	eLoad	ThLoad	Chr
	7.2.2	BEV-1	Body 1	X	X	X			X				X		X		X			X		X	X				X	X	X	X
			Body 2		X									X																
			Body 3																											
			Body 4																											
	7.2.1	FC-1	Body 1	X	X	X			X				X		X		X			X		X	X	X	X	X	X	X	X	
			Body 2		X								X																	
			Body 3		X								X																	
			Body 4																											
	7.3.2	FC-2	Body 1	X	X	X			X				X		X		X			X		X	X	X	X	X	X	X	X	
			Body 2		X		X							X	X		X			X		X	X				X		X	X
			Body 3																											
			Body 4																											
	7.3.4	BEV-5	Body 1	X	X	X			X				X		X		X			X		X	X				X	X	X	X
			Body 2		X										X												X			X
			Body 3																											
			Body 4																											
	7.4.2	BEV-6	Body 1	X	X	X			X				X		X		X			X		X	X				X		X	X
			Body 2		X							X																		
			Body 3		X								X																	
			Body 4		X								X																	
	7.6.1	FC-3	Body 1	X	X	X			X				X		X		X			X		X	X	X	X	X	X	X	X	
			Body 2		X									X																
			Body 3																											
			Body 4																											

Figure 20: Architecture definition examples of some of the demonstrators.

It can be noted that there is also the possibility to add modules without traction, and in case that some of the modules have traction they should have a VCU definition as to follow the master VCU which is located on the traction unit.

The modularity of the architecture will permit to calculate the impact of using other topologies for each of the demonstrators, such as more tractive modules, optimized management of electrified modules torque, liftable axis, etc.

2.4.3 Multi-architecture input files

The user can configure multiple simulation runs to be executed in batch simulation. Each simulation run consists of a tandem of a vehicle assembly definition and a test case definition as depicted in Table 21. This section specifies the instructions to define both parametrizations.

As can be seen, there is a TC section in which the environmental conditions are defined together in the ‘TestCase’ model to the speed that the truck must follow through the entire simulation. On the other hand, in Modules there must be more sections for the components of the truck in which the user wants to set an initial value. In this example, for the ‘Tractor’ unit, the user has introduced an initial average temperature and SoC for the battery, and initial temperature for the FC and eDrive units.

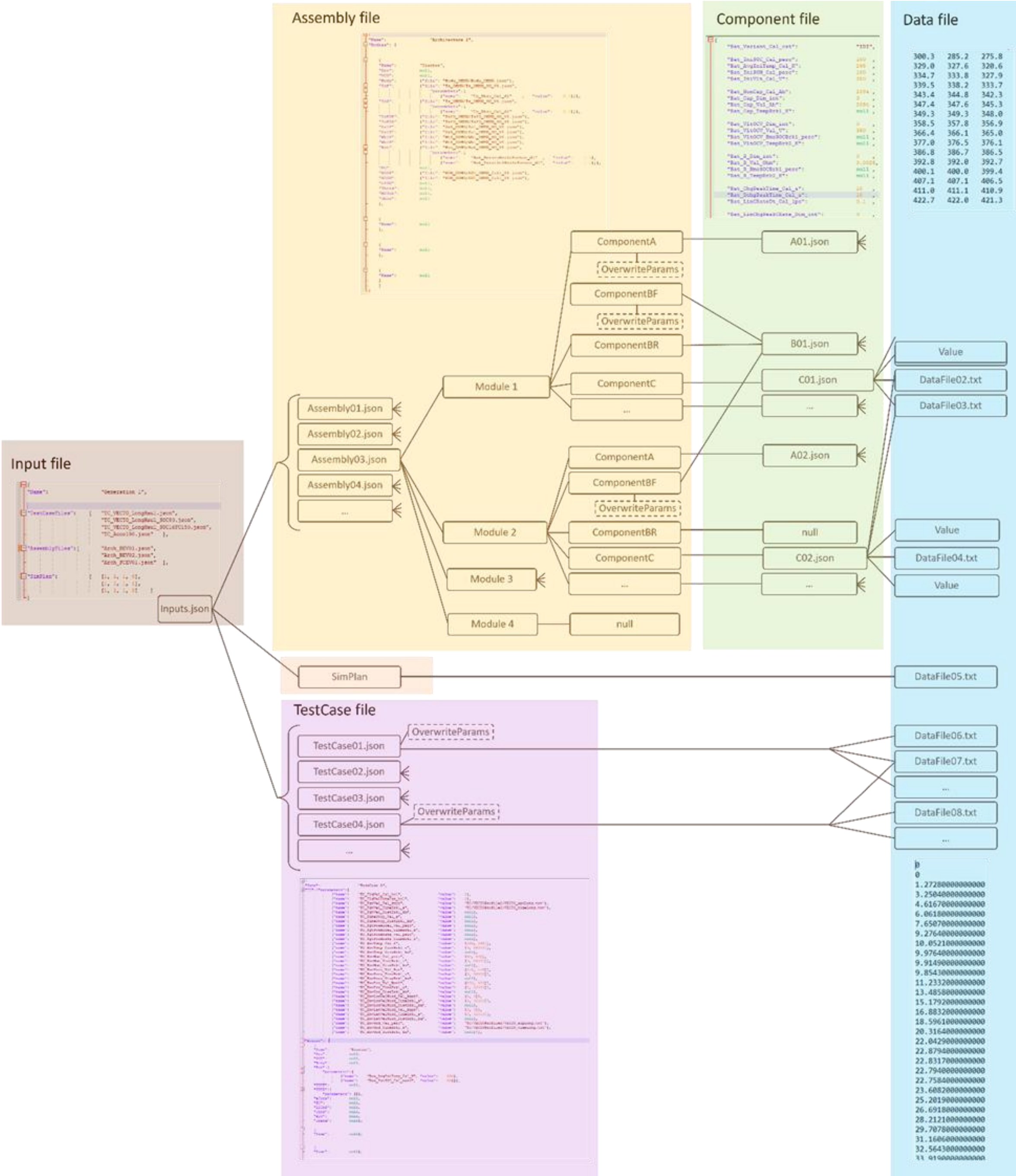


Figure 21: Structure scheme oh the input files of the tool.

Each vehicle configuration has an assembly file on *json* format, which defines the vehicle architecture and contains the path to the subfiles that define each of the components, also on *json* format. Inside of the *json* that defines a component, parameter values are written, and in some cases, they can also contain a path to a *txt* file that can contain either a vector or a matrix to define nD maps or vectors. All the parameters that need to have a value other than zero must be defined on these *json* files, otherwise the simulation will raise errors. These *json* files can be named as the user requires for convenience.

On the other hand, the structured input must also include a test case file in *json* format in which all the parameters regarding each simulation cycle are defined, once again allowing the user to give a path to a *txt* file in case the parameter consists of a nD vector, such as speed or slope profiles versus time. These *json* files can also have whatever name the user requires to use for convenience.

Finally, another mandatory file must be defined, it is the *Inputs.json* file depicted in Figure 22. It must have always the same name, and it aggregates the key information about the simulation batch:

- Name: it is the name of the simulation batch, and it can have any name the user desires.
- TestCaseFiles: it is a vector of names. Each name is the path of the *json* where each of the testcase or simulation cycles are defined.
- AssemblyFiles: it is a vector of names. Each name is the path of the *json* where each of the vehicle configurations are defined. The assembly files can contain different architectures or the same vehicle architecture with different component parameters.
- SimPlan: it’s a matrix in which the desired combinations of TestCase and Assembly are selected to simulate. Being columns the assemblies and rows the testcases, the user must put to 1 the elements corresponding to the desired combinations. If the user wants to simulate all the combinations, they can either put all elements to 1 or set SimPlan parameter to *null*.

```
{
"Name":                 "Generation 1",


"TestCaseFiles":    [   "TC_VECTO_LongHaul_SOC100.json",
                        "TC_VECTO_LongHaul_SOC3.json",
                        "TC_Accel90.json"    ],

"AssemblyFiles":    [   "Assy_FCEV1.json",
                        "Assy_BEV1.json",
                        "Assy_BEV2.json"     ],

"SimPlan":          [   [1,1,1],
                        [1,0,0],
                        [1,1,1]    ]
}
```

Figure 22: Demo json file for the Inputs

2.4.3.1 Vehicle assembly definition

Naming convention and functional requirements definition are used as a standard for the parameter input of the individual component models. The global architecture, components and subcomponents selection are defined using *json* files format.

For the definition, a global assembly *json* file defines the modules that the vehicle to simulate have, and for each module the components to consider. Each of these components has an additional *json* file that defines all its parameters, and in the case one of these is not present in the module, it must be marked as *null* (e.g., a module only has frontal eDrive, so rear eDrive is marked as *null*). If this is done, the simulation model will adapt the functionalities of this component, either bypassing it or disabling it depending on the component's nature.

```json
{
"Name":                 "Architecture DEMO",
"Modules": [


    {
    "Name":         "Tractor",
    "Drv":          null,
    "VCU":          {"file": "VCU_DEMO/VCU_DEMO.json"},
    "Body":         {"file": "Body_DEMO/Body_DEMO.json"},
    "TxF":          {"file": "Tx_DEMO/Tx_DEMO_0D_V4.json"},
    "TxR":          {"file": "Tx_DEMO/Tx_DEMO_0D_V4.json"},
    "TxFDF":        {"file": "TxFD_DEMO/TxFD_DEMO_0D_V4.json"},
    "TxFDR":        {"file": "TxFD_DEMO/TxFD_DEMO_0D_V4.json"},
    "AxlF":         {"file": "Axl_DEMO/Axl_DEMO_0D_V4.json"},
    "AxlR":         {"file": "Axl_DEMO/Axl_DEMO_0D_V4.json"},
    "WhlF":         {"file": "Whl_DEMO/Whl_DEMO_0D_V4.json"},
    "WhlR":         {"file": "Whl_DEMO/Whl_DEMO_0D_V4.json"},
    "Bat":          {"file": "Bat_DEMO/Bat_DEMO_0D_V4.json",
                        "parameters":[
                            {"name":    "Bat_SeriesScaleFactor_dl"  ,   "value":    1.0},
                            {"name":    "Bat_ParallelScaleFactor_dl",   "value":    1.0}]},
    "FC":           {"file": "FC_DEMO/FC_DEMO_1D_V4.json",
    "EDUF":         {"file": "EDU_DEMO/EDU_DEMO_full_V4.json",
                        "parameters":[
                            {"name":    "EDU_TrqScaleFactor_dl" ,   "value":    1.0},
                            {"name":    "EDU_SpdScaleFactor_dl",    "value":    1.0}]},
    "EDUR":         {"file": "EDU_DEMO/EDU_DEMO_full_V4.json",
                        "parameters":[
                            {"name":    "EDU_TrqScaleFactor_dl" ,   "value":    1.0},
                            {"name":    "EDU_SpdScaleFactor_dl",    "value":    1.0}]},
    "H2Tnk":        {"file": "H2Tnk_DEMO/H2Tnk_DEMO_V4.json"}
    },


    {
    "Name":         "Trailer",
    "Drv":          null,
    "VCU":          {"file": "VCU_DEMO/VCU_DEMO.json"},
    "Body":         {"file": "Body_DEMO/Body_Trailer_DEMO.json"},
    "AxlF":         {"file": "Axl_DEMO/Axl_DEMO_0D_V4.json"},
    "AxlR":         {"file": "Axl_DEMO/Axl_DEMO_0D_V4.json"},
    "WhlF":         {"file": "Whl_DEMO/Whl_DEMO_0D_V4.json"},
    "WhlR":         {"file": "Whl_DEMO/Whl_DEMO_0D_V4.json"}
    },


    {
    "Name":         null
    },


    {
    "Name":         null
    }
    ]
}
```

Figure 23: Demo json file for the vehicle assembly

In Figure 23, an example of a tractor with semitrailer architecture is defined in a demo *json*. As can be observed, each of the components of the system has a *json* file that specifies the values of the required parameters needed to properly simulate the vehicle.

As an example, the definition of a component json is seen in the next figure, in this case the battery component. As defined in the functional requirements, all the parameters required for the model must be inserted in the json file for the model to work properly. In case the user of the platform wants to change their battery parameters for more accurate values that define their component better, they should change those parameters in this json file.


```
{
    "Bat_Variant_Cal_cat":                    "IDI",
    "Bat_IniSOC_Cal_perc":                    100  ,
    "Bat_AvgIniTemp_Cal_K":                   298  ,
    "Bat_IniSOH_Cal_perc":                    100  ,
    "Bat_IniVlt_Cal_V":                       350  ,
    "Bat_NomCap_Cal_Ah":                      2094 ,
    "Bat_Cap_Dim_int":                        0    ,
    "Bat_Cap_Val_Ah":                         2094 ,
    "Bat_Cap_TempBrk1_K":                     null ,
    "Bat_VltOCV_Dim_int":                     0    ,
    "Bat_VltOCV_Val_V":                       350  ,
    "Bat_VltOCV_BmsSOCBrk1_perc":             null ,
    "Bat_VltOCV_TempBrk2_K":                  null ,
    "Bat_R_Dim_int":                          0    ,
    "Bat_R_Val_Ohm":                          0.0025,
    "Bat_R_BmsSOCBrk1_perc":                  null ,
    "Bat_R_TempBrk2_K":                       null ,
    "Bat_ChgPeakTime_Cal_s":                  10   ,
    "Bat_DchgPeakTime_Cal_s":                 10   ,
    "Bat_LimChgPeakCRate_Dim_int":            0    ,
    "Bat_LimChgPeakCRate_Val_dl":             0.025,
    "Bat_LimChgPeakCRate_BmsSOCBrk1_perc":    null ,
    "Bat_LimChgPeakCRate_TempBrk2_K":         null ,
    "Bat_LimChgContCRate_Dim_int":            0    ,
    "Bat_LimChgContCRate_Val_dl":             0.02 ,
    "Bat_LimChgContCRate_BmsSOCBrk1_perc":    null ,
    "Bat_LimChgContCRate_TempBrk2_K":         null ,
    "Bat_LimDchgPeakCRate_Dim_int":           0    ,
    "Bat_LimDchgPeakCRate_Val_dl":            0.05 ,
    "Bat_LimDchgPeakCRate_BmsSOCBrk1_perc":   null ,
    "Bat_LimDchgPeakCRate_TempBrk2_K":        null ,
    "Bat_LimDchgContCRate_Dim_int":           0    ,
    "Bat_LimDchgContCRate_Val_dl":            0.03 ,
    "Bat_LimDchgContCRate_BmsSOCBrk1_perc":   null ,
    "Bat_LimDchgContCRate_TempBrk2_K":        null ,
    "Bat_LimCRateDt_Cal_1ps":                 0.1  ,
    "Bat_LimMinBmsSOC_Cal_perc":              5    ,
    "Bat_LimMaxBmsSOC_Cal_perc":              100  ,
    "Bat_NomCycLife_Cal_int":                 3000 ,
    "Bat_ThermFlg_Cal_bool":                  0    ,
    "Bat_Mass_Cal_kg":                        7000 ,
    "Bat_AvgMinTgtTemp_Cal_K":                293  ,
    "Bat_AvgMaxTgtTemp_Cal_K":                308  ,
    "Bat_EqCp_Cal_JpkgpK":                    1000 ,
    "Bat_Eqhconv_Cal_Wpm2pK":                 12   ,
    "Bat_EqSurf_Cal_m2":                      2    ,
    "Bat_RefriLimMflw_Cal_kgps":              0.4  ,
    "Bat_RefriCp_Cal_JpkgpK":                 3559 ,
    "Bat_SeriesScaleFactor_dl":               1    ,
    "Bat_ParallelScaleFactor_dl":             1
}
```


Figure 24: Demo json file for the IDIADA 0D battery definition

It is also possible to select if the user wants to use the IDIADA generic model 'IDI' that has almost every parameter customizable, or the more precise model given by the supplier. In this example of the battery, the used could set *"Bat_Variant_Cal_cat"* parameter to "FHG" and change all the parameters to the ones required by the precise model.

In this case, the component used for the battery in the assembly definition is the IDIADA 0D generic model, and since it is a 0D model the required parameters are single values.

In Figure 25 there is a demo *json* of the IDIADA 2D generic model, for which tabularized parameters are required, with their corresponding breakpoints. The tabularized data is obtained from specified *.txt* files and the number of dimensions and breakpoints for each table is given.

```
1  {
2      "Bat_Variant_Cal_cat":                       "IDI",
3      "Bat_IniSOC_Cal_perc":                       100  ,
4      "Bat_AvgIniTemp_Cal_K":                      298  ,
5      "Bat_IniSOH_Cal_perc":                       100  ,
6      "Bat_IniVlt_Cal_V":                          350  ,
7      "Bat_NomCap_Cal_Ah":                         2094 ,
8      "Bat_Cap_Dim_int":                           1    ,
9      "Bat_Cap_Val_Ah":                            [2069,2108,2094,2066],
10     "Bat_Cap_TempBrk1_K":                        [253,283,293,313],
11     "Bat_VltOCV_Dim_int":                        2    ,
12     "Bat_VltOCV_Val_V":                          "Bat_DEMO/Bat_VltOCV_Val_V.txt"  ,
13     "Bat_VltOCV_BmsSOCBrk1_perc":                [0,2.5,3.5,4.5,6,8,10,20,30,40,50,60,70,80,90,100],
14     "Bat_VltOCV_TempBrk2_K":                     [273,298,318],
15     "Bat_R_Dim_int":                             2    ,
16     "Bat_R_Val_Ohm":                             "Bat_DEMO/Bat_Res_Val_Ohm.txt"  ,
17     "Bat_R_BmsSOCBrk1_perc":                     [0,2,4.5,7.5,10,20,30,40,50,60,70,80,90,95,100] ,
18     "Bat_R_TempBrk2_K":                          [273,293,318] ,
19     "Bat_ChgPeakTime_Cal_s":                     10   ,
20     "Bat_DchgPeakTime_Cal_s":                    10   ,
21     "Bat_LimChgPeakCRate_Dim_int":               2    ,
22     "Bat_LimChgPeakCRate_Val_dl":                "Bat_DEMO/Bat_LimChgPeakCRate_Val_dl.txt" ,
23     "Bat_LimChgPeakCRate_BmsSOCBrk1_perc":       [0,90,90.1,100] ,
24     "Bat_LimChgPeakCRate_TempBrk2_K":            [273,283,298,318] ,
25     "Bat_LimChgContCRate_Dim_int":               2    ,
26     "Bat_LimChgContCRate_Val_dl":                "Bat_DEMO/Bat_LimChgContCRate_Val_dl.txt",
27     "Bat_LimChgContCRate_BmsSOCBrk1_perc":       [0,90,90.1,100] ,
28     "Bat_LimChgContCRate_TempBrk2_K":            [273,283,298,318] ,
29     "Bat_LimDchgPeakCRate_Dim_int":              2    ,
30     "Bat_LimDchgPeakCRate_Val_dl":               "Bat_DEMO/Bat_LimDchgPeakCRate_Val_dl.txt",
31     "Bat_LimDchgPeakCRate_BmsSOCBrk1_perc":      [0,10,10.1,20,20.1,100] ,
32     "Bat_LimDchgPeakCRate_TempBrk2_K":           [273,283,298,318] ,
33     "Bat_LimDchgContCRate_Dim_int":              2    ,
34     "Bat_LimDchgContCRate_Val_dl":               "Bat_DEMO/Bat_LimDchgContCRate_Val_dl.txt",
35     "Bat_LimDchgContCRate_BmsSOCBrk1_perc":      [0,10,10.1,20,20.1,100],
36     "Bat_LimDchgContCRate_TempBrk2_K":           [273,283,298,318] ,
37     "Bat_LimCRateDt_Cal_1ps":                    0.1     ,
38     "Bat_LimMinBmsSOC_Cal_perc":                 5    ,
39     "Bat_LimMaxBmsSOC_Cal_perc":                 100  ,
40     "Bat_NomCycLife_Cal_int":                    3000 ,
41     "Bat_ThermFlg_Cal_bool":                     0    ,
42     "Bat_Mass_Cal_kg":                           7000 ,
43     "Bat_AvgMinTgtTemp_Cal_K":                   293  ,
44     "Bat_AvgMaxTgtTemp_Cal_K":                   308  ,
45     "Bat_EqCp_Cal_JpkgpK":                       1000 ,
46     "Bat_Eqhconv_Cal_Wpm2pK":                    12   ,
47     "Bat_EqSurf_Cal_m2":                         2    ,
48     "Bat_RefriLimMflw_Cal_kgps":                 0.4  ,
49     "Bat_RefriCp_Cal_JpkgpK":                    3559 ,
50     "Bat_SeriesScaleFactor_dl":                  1    ,
51     "Bat_ParallelScaleFactor_dl":                1
52 }
```

Figure 25: Demo json file for the IDIADA 2D battery definition

Additionally, an example of the tabulated data is given in Figure 26. The format to define this file is a matrix with a row on each line and space or indent separation for columns.

In this example, the *Bat_VltOCV_Val_V* in Figure 26 represents the matrix of the open circuit voltage of the battery in relation to the SoC and the battery temperature. And as seen in Figure 25, the dimensions of the table are given in line 11, the name of the *txt* file to check is in line 12 and the given breakpoints for temperature and SoC to use this table in the simulation are given in lines 13-14.

300.3	285.2	275.8
329.0	327.6	320.6
334.7	333.8	327.9
339.5	338.2	333.7
343.4	344.8	342.3
347.4	347.6	345.3
349.3	349.3	348.0
358.5	357.8	356.9
366.4	366.1	365.0
377.0	376.5	376.1
386.8	386.7	386.5
392.8	392.0	392.7
400.1	400.0	399.4
407.1	407.1	406.5
411.0	411.1	410.9
422.7	422.0	421.3

Figure 26: Demo txt file of required tabulated data.

2.4.3.2 Test Case definition

With the vehicle assembly fully defined, an additional input file is required to define the cycle conditions, i.e. all the input data required for the simulation which does not depend on the vehicle but on the environmental conditions, the road characteristics and some initial conditions of the components like initial SoC, initial level on the H2 tank, initial temperature of the components, etc. In the following figure there is a DEMO *json* file as an example definition of the environmental conditions on the simulation cycle.

As can be seen, there is a TC section in which the environmental conditions are defined together in the 'TestCase' model to the speed that the truck must follow through the entire simulation. On the other hand, in Modules there must be more sections for the components of the truck in which the user wants to set an initial value. In this example, for the 'Tractor' unit, the user has introduced an initial average temperature and SoC for the battery, and initial temperature for the FC and eDrive units.


```
{
"Name":                "TC_VECTO_LongHaul",
"TC":{  "parameters":[
            {"name":    "TC_Variant_Cal_cat",            "value":    "Consumption"},
            {"name":    "TC_MaxSimTime_Cal_s",           "value":    32580},
            {"name":    "TC_TimeStep_Cal_s",             "value":    1},
            {"name":    "TC_FlgVel_Cal_bol",             "value":    1},
            {"name":    "TC_FlgVel_Cal_bol",             "value":    1},
            {"name":    "TC_FlgVel_Cal_bol",             "value":    1},
            {"name":    "TC_FlgVelTimeTgt_bol",          "value":    1},
            {"name":    "TC_TgtVel_Val_kmph",            "value":    "TC_DEMO/VECTO_LongHaul/VECTO_spdLong.txt"},
            {"name":    "TC_TgtVel_TimeBrkl_s",          "value":    "TC_DEMO/VECTO_LongHaul/VECTO_timeLong.txt"},
            {"name":    "TC_TgtVel_DistBrkl_km",         "value":    null},
            {"name":    "TC_TimeStop_Val_s",             "value":    null},
            {"name":    "TC_TimeStop_DistBrkl_km",       "value":    null},
            {"name":    "TC_TgtPosAccel_Val_perc",       "value":    null},
            {"name":    "TC_TgtPosAccel_TimeBrkl_s",     "value":    null},
            {"name":    "TC_TgtPosBrake_Val_perc",       "value":    null},
            {"name":    "TC_TgtPosBrake_TimeBrkl_s",     "value":    null},
            {"name":    "TC_EnvTemp_Val_K",              "value":    [296, 296]},
            {"name":    "TC_EnvTemp_TimeBrkl_s",         "value":    [0, 36000]},
            {"name":    "TC_EnvTemp_DistBrkl_km",        "value":    null},
            {"name":    "TC_EnvHum_Val_perc",            "value":    [50, 50]},
            {"name":    "TC_EnvHum_TimeBrkl_s",          "value":    [0, 36000]},
            {"name":    "TC_EnvHum_DistBrkl_km",         "value":    null},
            {"name":    "TC_EnvPres_Val_Bar",            "value":    [1e5, 1e5]},
            {"name":    "TC_EnvPres_TimeBrkl_s",         "value":    [0, 36000]},
            {"name":    "TC_EnvPres_DistBrkl_km",        "value":    null},
            {"name":    "TC_EnvIrr_Val_Wpm2",            "value":    [500, 500]},
            {"name":    "TC_EnvIrr_TimeBrkl_s",          "value":    [0, 36000]},
            {"name":    "TC_EnvIrr_DistBrkl_km",         "value":    null},
            {"name":    "TC_EnvLonVelWind_Val_kmph",     "value":    [0, 0]},
            {"name":    "TC_EnvLonVelWind_TimeBrkl_s",   "value":    [0, 36000]},
            {"name":    "TC_EnvLonVelWind_DistBrkl_km",  "value":    null},
            {"name":    "TC_EnvLatVelWind_Val_kmph",     "value":    [0, 0]},
            {"name":    "TC_EnvLatVelWind_TimeBrkl_s",   "value":    [0, 36000]},
            {"name":    "TC_EnvLatVelWind_DistBrkl_km",  "value":    null},
            {"name":    "TC_EnvGrd_Val_perc",            "value":    "TC_DEMO/VECTO_LongHaul/VECTO_slpLong.txt"},
            {"name":    "TC_EnvGrd_TimeBrkl_s",          "value":    "TC_DEMO/VECTO_LongHaul/VECTO_timeLong.txt"},
            {"name":    "TC_EnvGrd_DistBrkl_km",         "value":    null}]},

"Modules": [
    {
    "Name":         "Tractor",
    "Drv":          null,
    "VCU":          null,
    "Body":         null,
    "Bat":{
        "parameters":[
                {"name":    "Bat_AvgIniTemp_Cal_K",     "value":    300},
                {"name":    "Bat_IniDashSOC_Cal_perc",  "value":    100}]},
    "EDUF":         null,
    "EDUR":{
        "parameters": []},
    "eCons":        null,
    "FC":           null,
    "H2Tnk":        null,
    "Jbox":         null,
    "PTO":          null,
    "Therm":        null},

    {
    "Name":         null},


    {
    "Name":         null},


    {
    "Name":         null}]
}
```


Figure 27: Demo json file for the test case definition

With this parameter settings, the user can overwrite parameters of the truck that despite being defined on the assembly file, could have different values on different cycle simulations.

2.4.4 Input parametrizations

2.4.4.1 Parametrization of generic models from data handover or test activities

Input parameters for the platform simulations will eventually be collected by WP1, WP4, WP5 and WP6. The input data can be obtained from vehicle and component specifications of the demonstrators or by calibrating the model with data received from the experimental activities.

At the time being, during the development of the platform the input data for the specific components is not available to develop and validate the models.

For this reason, the platform was designed to have open parameters tuneable for the end-users when the input is available and not condition the availability of the platform to the availability of the inputs. Therefore, the main focus of the activity is the creation of the flexible platform, not the calibration.

2.4.4.2 Default DEMO parametrizations for generic models

However, the platform requires a significant amount of input values to run a successful simulation and to validate the correct operation and communication of all the architectures and feasibility of the results. Also, the usage of the platform is not trivial, and the end-users need example cases in order to have a starting point from where to calibrate their own simulations.

For this purpose, IDIADA generated from internal know-how sets of DEMO data for different model fidelity levels (maps from 0D to 4D) that will be provided with the platform as starting point.

This DEMO parametrizations represent component calibrations that are physically reasonable considering state-of-the-art and market maturity of the component, and compatible in sizing with the rest of components of the DEMO platform. However, it has to be remarked that the data is not tuned to any specific component of ZEFES or other project and IDIADA is not responsible for the validity of these DEMO inputs for the ZEFES project. These DEMO definitions are available in the web app as a template example, and some of them will be also defined in Section 4 Validation.

2.4.4.3 Expert supplier calibrated models

Apart from the generic models, there will be three models that will be supplied from partners representing expert suppliers in the toolchain. These models are battery model from FHG, tyre model by MICH and fuel cell model from RIC. These models are documented in Section 2.3.4 Individual component models

These models will be calibrated to specific components in the market. Calibration parameters will not be open, but there will be open parameters to configure starting conditions and for resizing the modelled components for optimization activities.

The main advantages of these models will be that are calibrated to specific existing components or detailed simulation models, and that will include a more detailed modelling in some of the functions.

2.4.5 User inputs pre-processing

All user input files and parameters pass through a process to load, validate, re-size, re-scale and overwrite scaling process in order to generate a valid input to the simulation platform.

One of the main challenges is that the platform is implemented in Simulink and compiled to generate an executable file that can be simulated in the web server. In the compilation process the dimensions and sizes of the maps in the Simulink model become fixed hindering the flexibility to change sizes and dimensions (from 0D to 4D) of the input data. The input pre-processing module permits the user to define different dimensions and sizes of input data and re-scales them to the sizes that were fixed in the compilation process. The main functions performed during the input pre-processing are explained in the following subsections.

2.4.5.1 Generate the list of cases to simulate

In a first step, the platform generates the list of cases to simulate by combining the list of assemblies and test cases provided by the user as depicted in Table 21.

Table 21: Example of list of cases to simulate

Run	Assembly	Test Case
1	Architecture1.json	TestCaseA.json
2	Architecture1.json	TestCaseB.json
3	Architecture1.json	TestCaseC.json
4	Architecture1.json	TestCaseD.json
5	Architecture2.json	TestCaseA.json
6	Architecture2.json	TestCaseB.json
7	Architecture2.json	TestCaseC.json
8	Architecture2.json	TestCaseD.json
9	Architecture2_scaled.json	TestCaseA.json
10	Architecture2_scaled.json	TestCaseB.json
…	…	…

2.4.5.2 Prepare input data structure for each simulation run

Each simulation run is initialized with a template data structure containing all the fields and maps fixed sizes required by the simulation. The fields and maps need to be initialized with empty values even for the components that will not be active in the current simulation run. For those that are active in the current assembly the empty values will be overwritten with the values defined by the user.

Internally the data structure is defined as:

$$Run\{i\}.Body\{j\}.Component\{k\}.Parameter\{l\}$$

2.4.5.3 Read the files of the simulation run

The file reading process of a specific simulation run can be split in several steps:

The simulation platform reads the configuration .json files for the assembly. An example of configuration file for an assembly presented in Figure 23.

1. Afterwards, the platform loops for all the modules and for all the components inside each module. If the component has a linked .json file, it reads the file with the parameters. An example of a configuration file for component is depicted in Figure 25.
2. Then the platform loops for all the parameters in the current component:

2.1. In case one of the loaded parameters of the component is not present in the template data structure, the configuration files are not valid. The simulation process stops, and the user receives an error message to correct the parameters that are not according to the standard.

2.2. In case the Assembly file specifies that some of the parameters of the component will be overwritten in the current assembly, it overwrites the parameters. This is used mostly for sizing studies.

2.3. In case the Test Case file specifies that some of the parameters of the component will be overwritten in the current run, it overwrites the parameters of the component and the assembly. This is used mostly for initializing component parameters linked to the test case such as initial temperatures.

2.4. In case one of the parameters of the components refers to a file in .mat, .txt or .csv format such as in the example of Figure 25, the pre-processing code re-assigns the values in the file to the parameter field.

2.4.5.4 Parameters sizes and dimensions validation

Many of the parameters that are linked to look-up-tables permit the user to choose between implementing a constant value (0D case) or a look-up-table with different number of dimensions (1D, 2D, 3D and even 4D) depending on the desired complexity of the model and the availability of the inputs.

However, it is important to validate that the number of dimensions specified by the user are valid for the given look-up-table (not all permit all dimensions from 0D to 4D), that the required breakpoints for the selected dimensions are present, and that the sizes of the table are coherent with the sizes of the breakpoints. The main validations that take place in this import step are:

1. Identify if the number of dimensions defined by the user for a table is valid for the component implementation. If we use as an example the EDU consumption map (`EDU_ElecPow_Val11_W`) , the valid dimensions are 2D, 3D and 4D (the map can depend on speed, torque, voltage and temperature). In case the user chooses to simulate with a 1D map, it will not pass the dimensions validation and produce an error.
2. Validate that each dimension has a definition for its break points.
3. Validate that all the breakpoints are monotonically increasing and do not contain invalid values.
4. Validate that the size (the length) of each breakpoint is coincident with the size of the table in the dimension it aims to represent.
5. In the case of 3D and 4D maps, the maps are defined as a collection of 2D slices of the complete map which name indicates the position in the third and fourth dimension. For example, for the case of the EDU consumption map that can have up to 4 dimensions, the slices are defined as `EDU_ElecPow_Val11_W`, `EDU_ElecPow_Val21_W`, `EDU_ElecPow_Val31_W` … In these cases, the input data has additional validation steps:

   5.1. Generate the list of slices that are needed to match the size of the breakpoints for dimensions 3 (Brk3) and 4 (Brk4). For example, if we have a size of 2 in Brk3 and a size of 3 in Brk4, the list of required slices would be: `11, 12, 13, 21, 22, 23`.

   5.2. Validate that all the slices in the list of required slices are present in the input file.

   5.3. Validate that the size of the matrix defined for each of the slices matches the 2D dimensions corresponding to the breakpoints for dimension 1 (Brk1) and for dimension 2 (Brk2).

2.4.5.5 Parameters resizing to standard dimensions

Matlab and Simulink are tools widely used in the industry, highly versatile, and, in general terms, requiring a moderate investment. A big blocking point is generated when deploying protected models through code generation or compilation, as the sizes of each dimension of the vectors and tables gets fixed.

This is a big limitation when generating tools to simulate different cycles and component characteristics, because the maps obtained from suppliers can be of any size and it is not convenient to force the end user to manually convert the inputs to a pre-defined fixed size. This drawback was also a pain point in previous EU funded projects with model exchange such as HIFI-Elements and Longrun.

The WP2 platform of the ZEFES aims to overcome this limitation. Each of the vectors and tables in the model is compiled to a specific size that is big enough to represent the physics of the component. However, the platform pre-processes user inputs of any dimensions and sized to the compiled size so that the size is transparent to the user.

The pre-processing algorithm is able to add dimensions, interpolate to increase the size or reduce size when needed both for table data and breakpoint data. In the following paragraphs there are some examples of why resizing is required and how it is performed.

Figure 28 shows the case of pre-processing a table to add dimensions. These situations may happen when a table can be defined in various dimensions (for example internal battery resistance can be defined as a function of SoC and temperature) but the user decides to model it as a constant resistance (0D case) due to lack of sufficient input data to define a 2D table. In the figure this would be represented by the case in the left. In this case the full size of the table needs to be filled with data to prevent size mismatch errors, but that part of the table will never be used in simulation time. In these cases, the new dimensions are filled with null values.

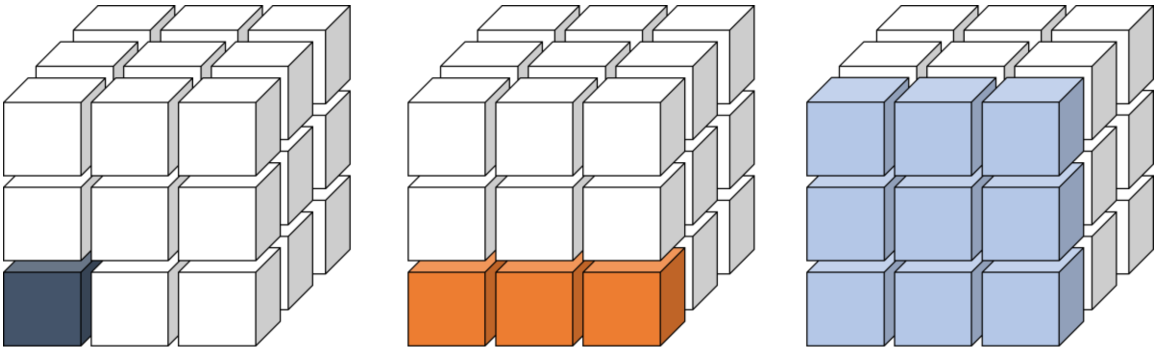


Figure 28: Pre-processing of tables to add dimensions

In order to improve the computation time of the simulation, the model that will be executed is different depending on the number of empty dimensions. In case the user chooses 0D it will be executed as a constant and in case the user chooses more dimensions as a look-up-table with the correct dimensions and the rest of implementations will not be executed. This type of implementation is depicted in Figure 29 and significantly improves the computation time when user selects 0D or 1D models, because the model does not need to look for the data in matrixes with numerous dimensions.

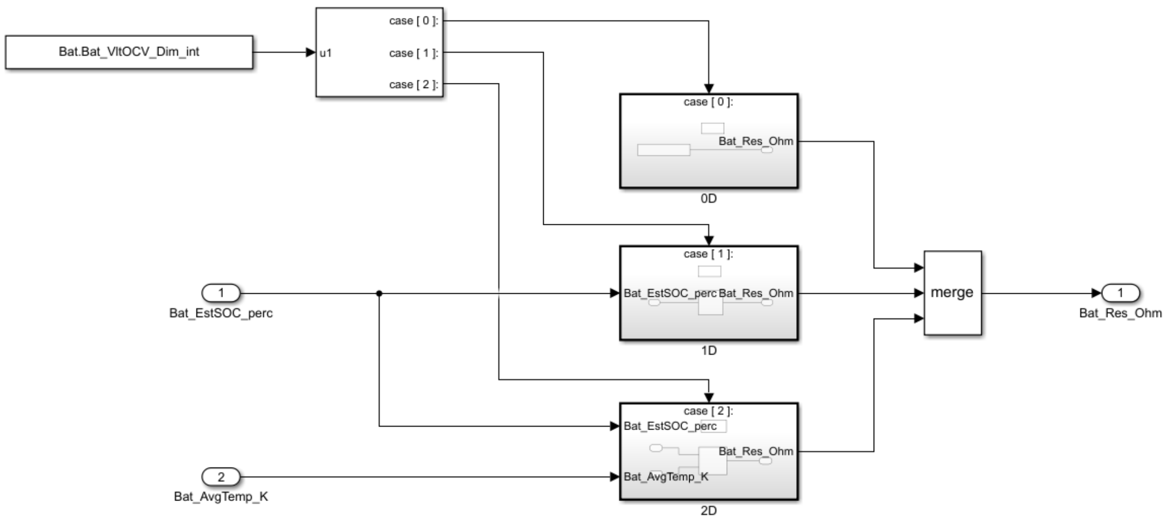


Figure 29: Implementation of a table model that can be parametrized with different number of dimensions

Other pre-processing that is required is the increase of size within the dimensions that are already defined. This occurs when the user models with the same number of dimensions, but the size of the input data is smaller. In these cases, there are some solutions that may lead to mistakes. For example, filling the table with null data would generate invalid results if, for some reason, the simulation needs to interpolate near to the null data.

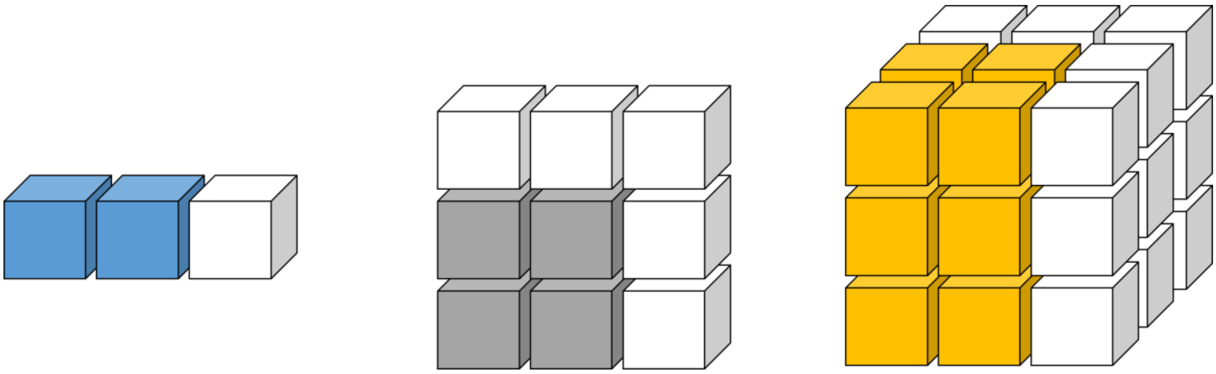


Figure 30: Pre-processing of tables to increase the size within the same dimensions

Other solution would be to make an interpolation to evenly distributed breakpoints, but this method could generate big errors in the cases that the breakpoints were selected carefully to represent non-continuous behaviours as those of the strategies as depicted in

Figure *31* (centre). The selected method for the platform adds the missing breakpoints before the last breakpoint so that the interpolated data does not modify the information content in the map. The same process is applied to the breakpoints.

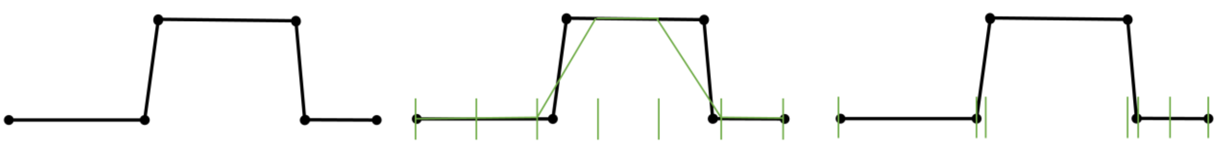


Figure 31: Different methods to interpolate to increase the size. Original data (left) evenly distributed breakpoints (centre) new breakpoints before the last breakpoint (right)

Other casuistic is that the size of the input data from the user is bigger than the size of the compiled tables as depicted in

Figure 32. The sizes of the matrix are defined sufficiently big to cover main physics that needs to be represented in the selected maps and the granularity that is achieved in component testing. However, there are cases in which suppliers provide huge maps, especially in the cases that the maps are produced by simulation software. In this case, the algorithm does not interpolate with evenly distributed breakpoints either. The method is to select specific breakpoint positions to select from the dataset. Anyway, there is risk of deleting important information or smoothing a non-linearity, so in cases the platform provides a warning to a user, so that in case non-linearities in the map are important, the user can reduce the size manually while preserving the non-linearity.

Warning message when downsizing:

```
Warning:
   Assembly: XX
     Module: XX
       Component: XX
          Parameter: XX
Table values for dimension XX were downsized from XX elements to XX
elements to match the look-up-table size.
```

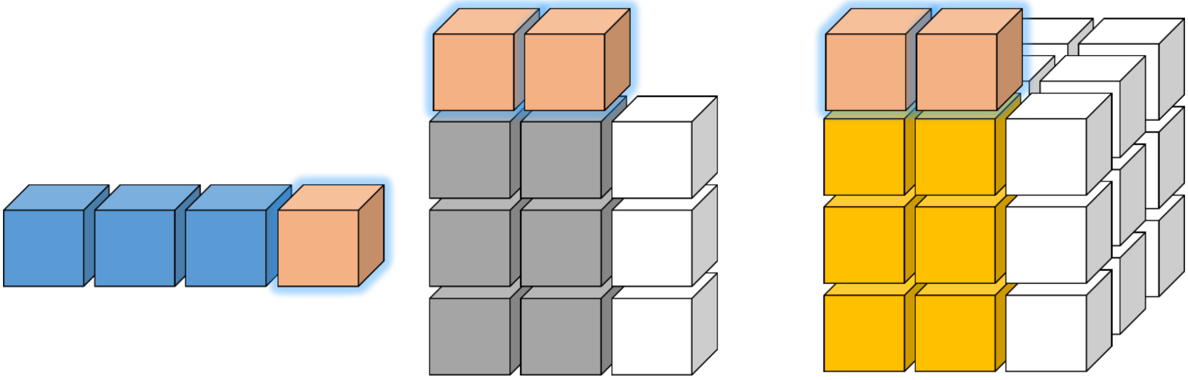


Figure 32: Pre-processing of tables to downsize within the same dimensions

As a final example, the most common case is that the pre-processing that is required is different for each dimension. Figure 33 shows the example of a 4D table that could for example represent the EDU electric consumption. The dimension 1 that is torque needs to be downsized. The dimension 2 that is speed needs to be interpolated because there is not enough data. The same happens for dimension 3 that is voltage. There are only maps for 2 voltage levels as an input (two 2D maps as an inputs) and these maps need to be interpolated to fill the full size of the voltage table. However, there is no input data to represent different behaviour in dimension 4 that is temperature. The maps in the 4D dimensions will be filled with null data, and the simulation model will execute only the implementation with a 3D map to improve computing time.

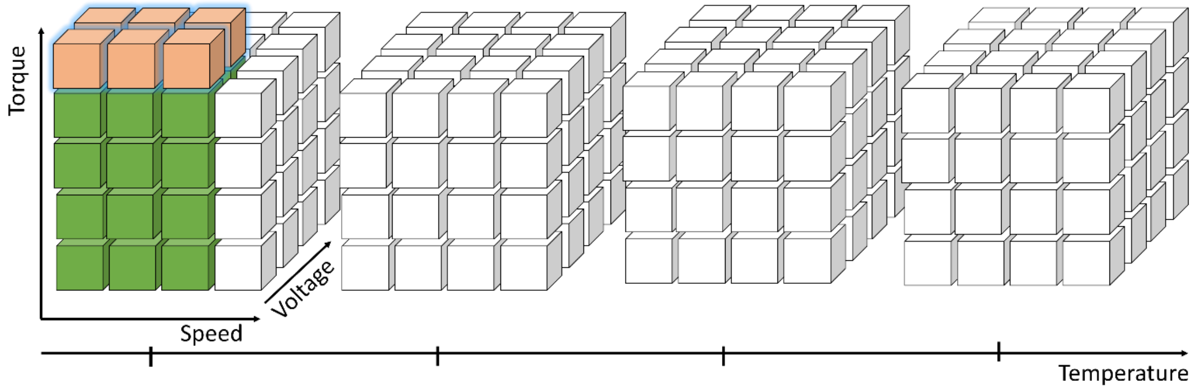


Figure 33: Pre-processing example for a 4D map

In the case of 3D and 4D maps, the input data is provided by the user as a collection of 2D maps indicating the position of each of them in the 3rd and 4th dimension. When pre-processing, the platform rearranges these sets of 2D maps to generate the 3D and 4D maps.

2.4.5.6 Parameters rescaling with scaling factors

The final step is parameters re-scaling. The main objective of permitting parametric re-scaling is to perform architecture and component sizing optimizations using complex multidimensional maps, but without the need to update all the maps on each run.

The possibility to manually modify the maps is very useful for implementing different suppliers' data, but it is not convenient when running automated parametric studies or optimizations. The algorithm would need to modify data in numerous files: the assembly file that calls the components, the component file that defines the parameters (and modify all the parameters that are affected by the re-scaling) and all the map files that are called by the component file.

To overcome this issue scaling factors were defined for each component. The component interface file specifies the formulas with which the rest of parameters are updated when applying a scaling factor. Table 13 shows the example for the battery, in that case, applying a scaling factor to the cells in parallel automatically updates the parameters referring to battery surface, weight, C-rate limits, capacity and internal resistance.

The scaling factors are defined in the component files. But can be retrieved as overwritten parameters in the assembly file as depicted in Figure 23. When they are retrieved from the assembly file, it is possible to modify the scaling of the component, including the mapped parameters, by only modifying the assembly file.

The pre-processing scripts identify if there are scaling factors, and, in such cases, after loading the component parameters those get updated with the scaling factors.

3 Total Cost of Ownership calculation tool

Total Cost of Ownership (TCO) is a measure that includes all the expenses during the lifetime of a truck: the capital expenses for the acquisition of the truck and the expenses needed for its operation. TCO also accounts for the amount recovered in the resale of the vehicle at the end of its life.

The purpose of the platform is the calculation of the TCO for ZEVs (Zero Emission Vehicles) and the simulation of the longitudinal performances and energy consumption of all the possible traction configurations for ZEVs considered in the project.

Besides, the platform needs to have a layer that is friendly to use for the logistic operators for the decision-making process and at the same time provide a flexible and accurate vehicle model to communicate and produce results for other tasks and WPs of the project.

Both tools are accessible through an interface hosted in a web server with granted access to project partners and stakeholders, offering grater accessibility to the simulation platforms.

The TCO calculation tool is implemented in Python with a user-friendly interface and the formulation is based on a deep literature review to define cost contributions as per state-of-the-art. The cost-contributions were adapted to the ZEFES logistic use-case and validated trough workshops with logistic and OEM stakeholders.

In this chapter, an interactive tool for calculating the TCO of zero-emission trucks is presented.

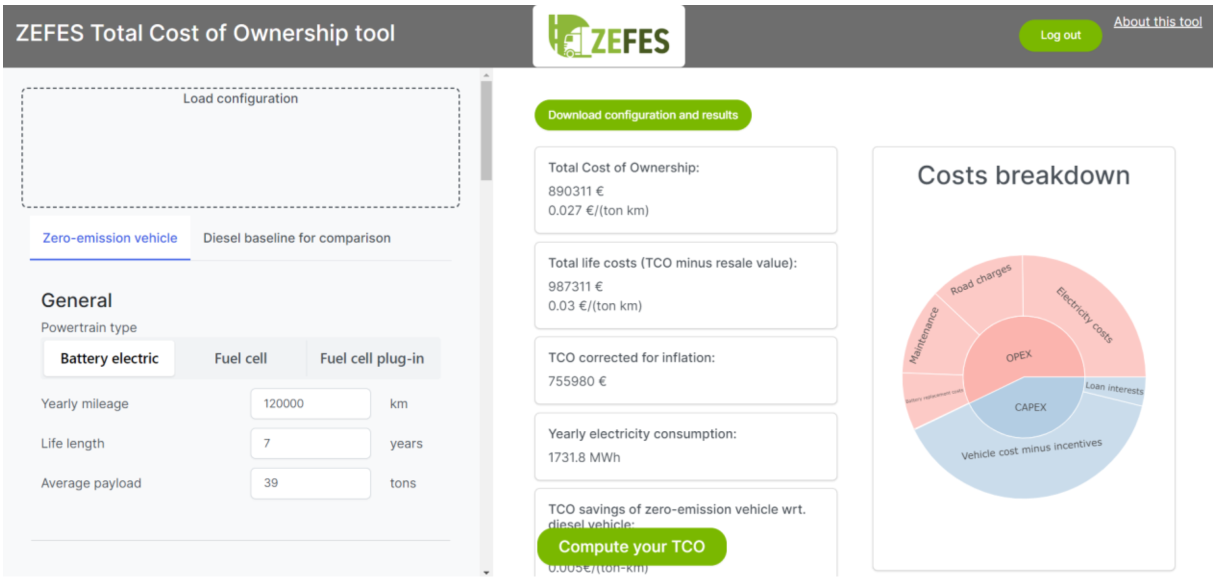


Figure 34: Total Cost of Ownership tool

3.1 Objective of the tool

The tool allows the user to calculate the Total Cost of Ownership of a zero-emission vehicle, with the choice between three different configurations: battery electric, fuel cell electric and fuel cell plug-in (hydrogen-electric hybrid). In addition, a model diesel vehicle is included as a baseline to compare TCO.

3.2 Interface design

The interface has two separate sections: a left section with all the input fields and an output section at the right with all the generated results.

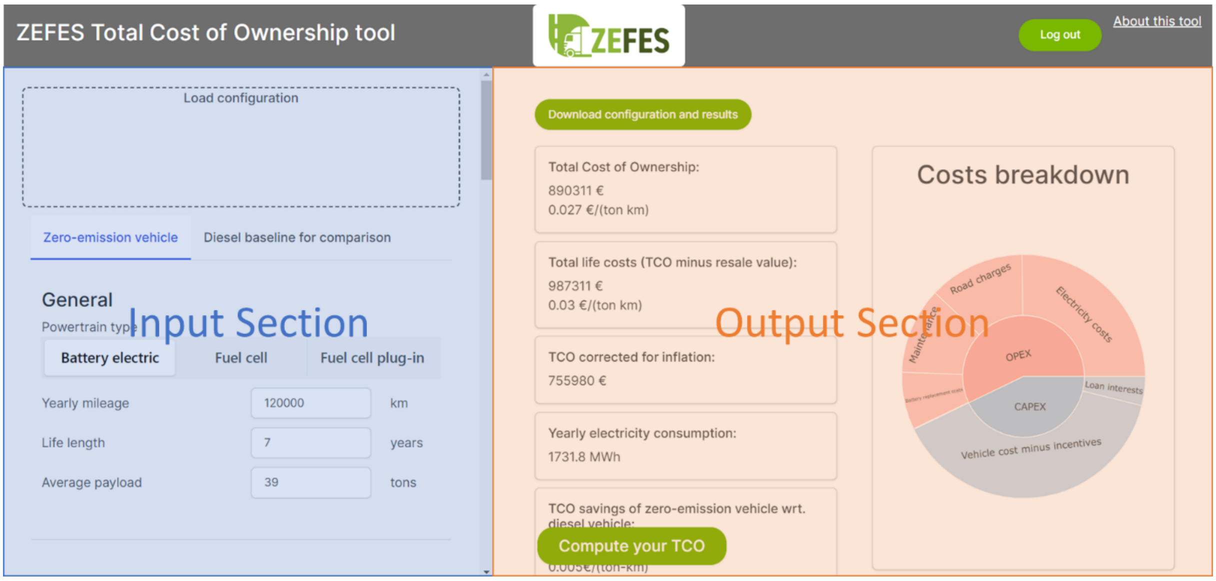


Figure 35: Total Cost of Ownership interface

3.3 Input section

The input section is divided in two main parts, one corresponding to the Zero-emission vehicle and the other corresponding to the Diesel baseline vehicle for comparison.

The Zero-emission vehicle part has the following subsections:

- General: selection of type of powertrain, yearly mileage, life of the vehicle and average payload.

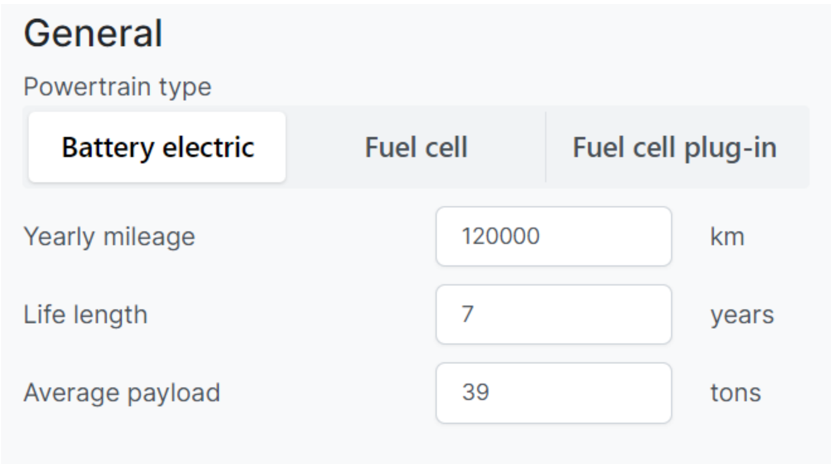


Figure 36: General Input Section

- Purchase: definition of vehicle cost, resale value, and loan.

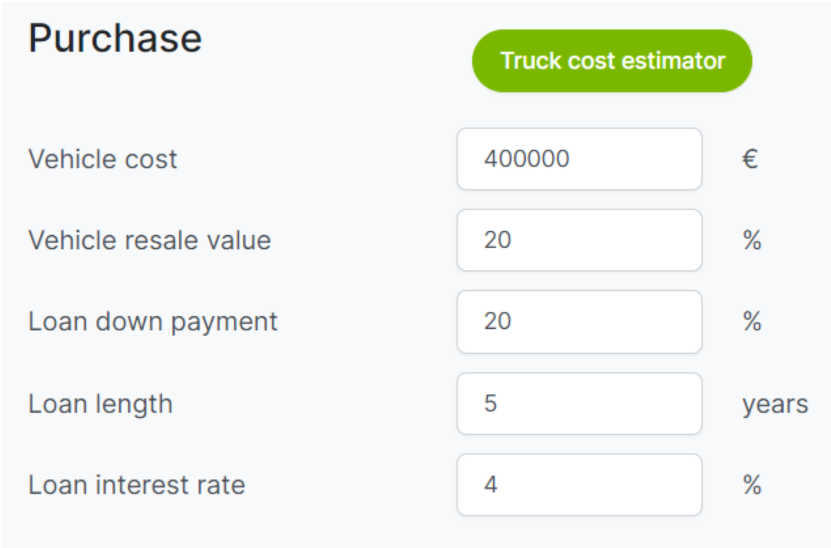


Figure 37: Purchase Input Section

  - Truck cost estimator: a module to estimate the truck price based on market maturity, battery or fuel cell requirements, e-drive required power and trailer and dolly body price.

    As the price of the e-drive units and the batteries is considered linear with the component size, in case of a vehicle with several motors and batteries (eg: e-trailer case) we should introduce the total power in kW or capacity in kWh of the aggregation of all motors and batteries for the price estimation.

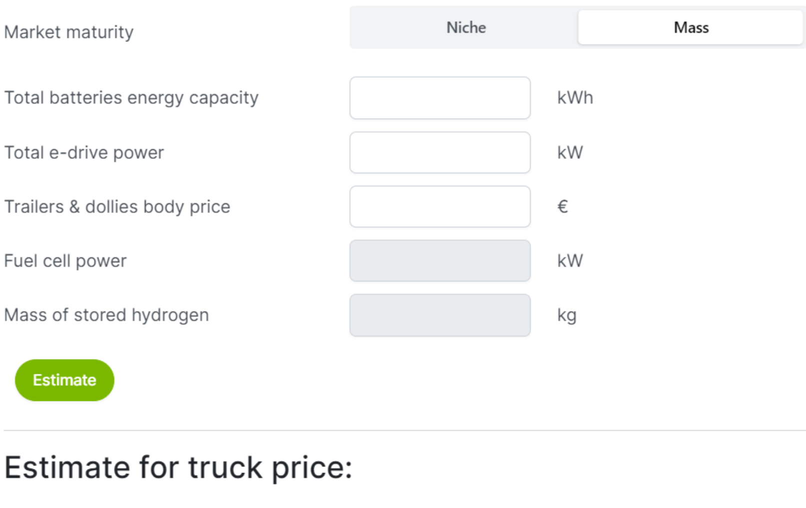
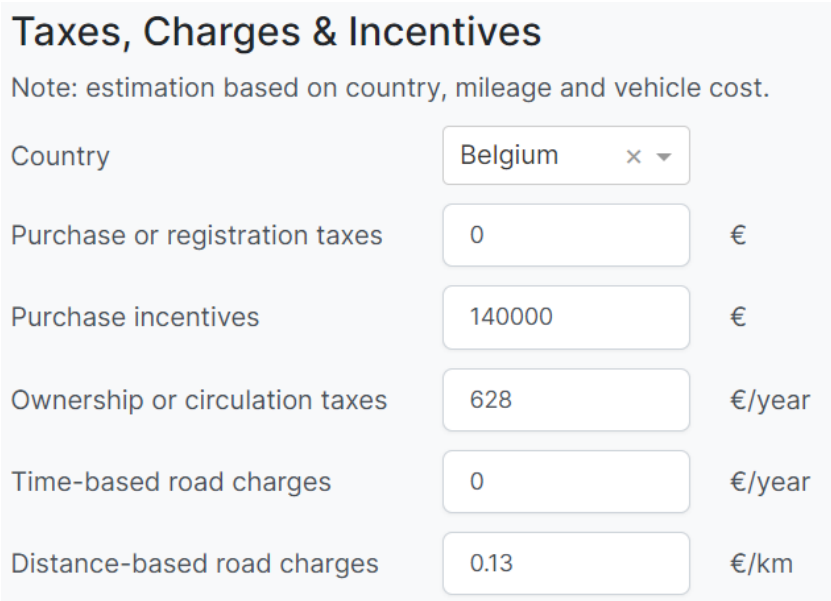


Figure 38: Truck cost estimator

- Taxes, Charges & Incentives: selection of country to automatically load relevant taxes and incentives.


Taxes, Charges & Incentives

Note: estimation based on country, mileage and vehicle cost.

Country	Belgium	
Purchase or registration taxes	0	€
Purchase incentives	140000	€
Ownership or circulation taxes	628	€/year
Time-based road charges	0	€/year
Distance-based road charges	0.13	€/km

Figure 39: Taxes, Charges & Incentives Input Section

- Electricity cost: for BEV, selection of consumption, charging efficiency, rate of public charging, and electricity prices.

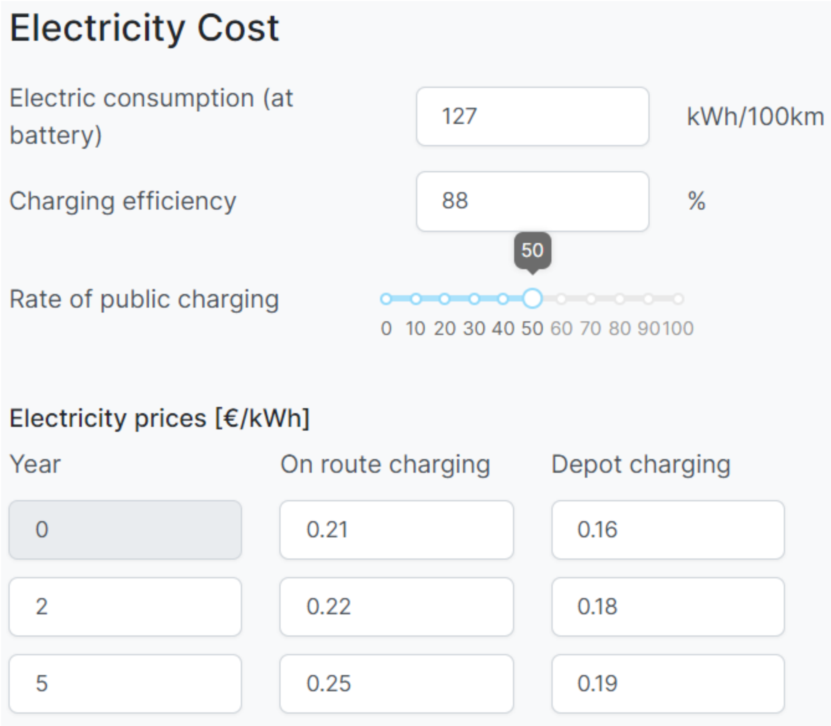


Figure 40: Electricity Cost Input Section

- Hydrogen cost: for FCEV, selection of consumption, rate of electric driving, and Hydrogen prices.

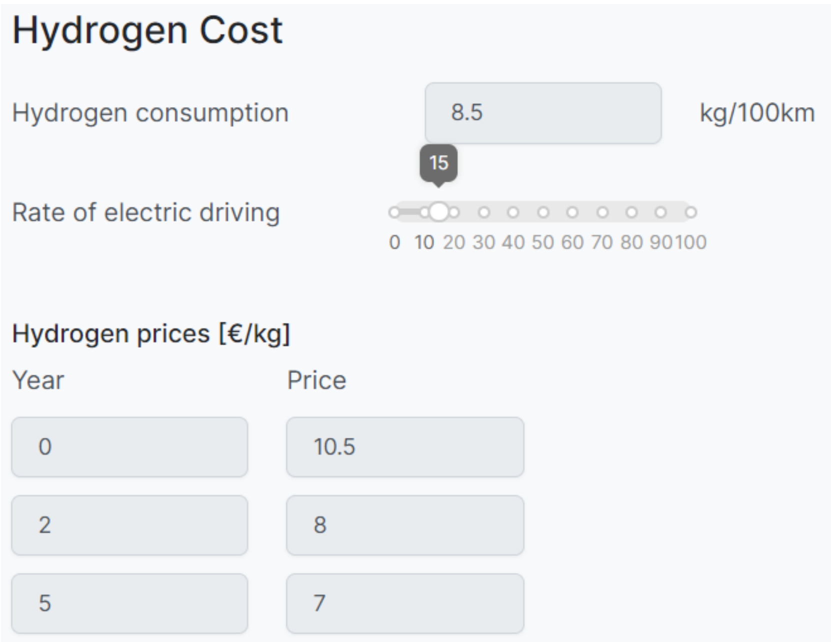


Figure 41: Hydrogen Cost Input Section

- Battery replacement: option to consider one (or more) battery replacement, including resale value and cost of the new battery.

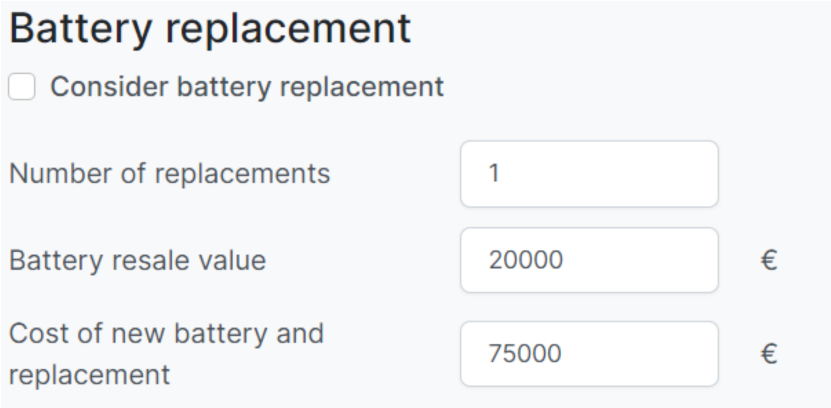


Figure 42: Battery Replacement Input Section

- Other: includes maintenance costs, insurance, driver wages and annual discount rate.

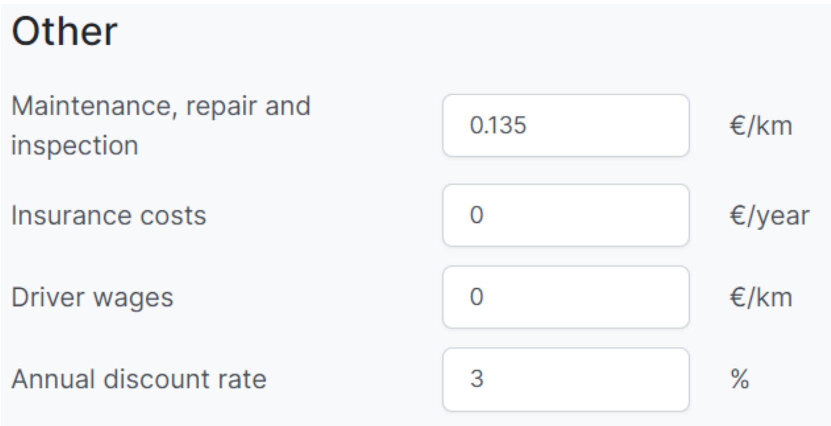


Figure 43: Other Input Section

The Diesel baseline vehicle part has the following subsections:

- General: selection of average payload.

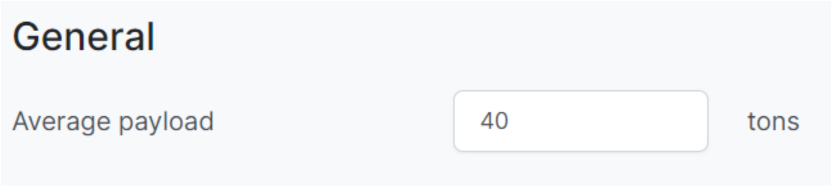


Figure 44: General Input Section

- Purchase: definition of vehicle cost, resale value and loan.

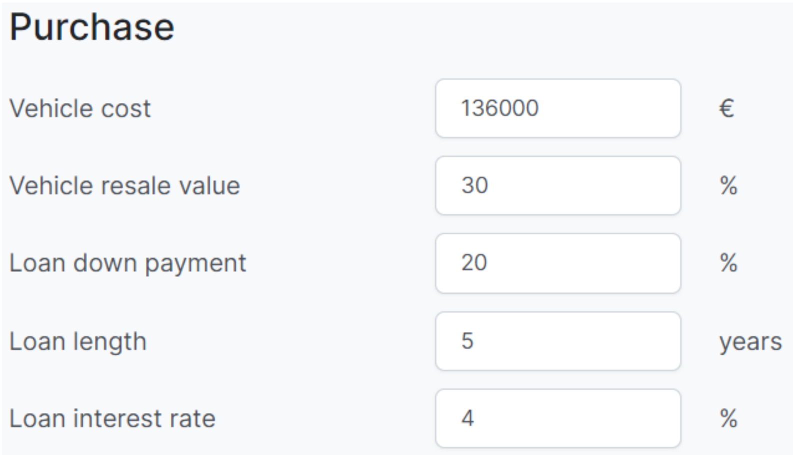


Figure 45: Purchase Input Section

- Taxes, Charges & Incentives: selection of country to automatically load relevant taxes and incentives.

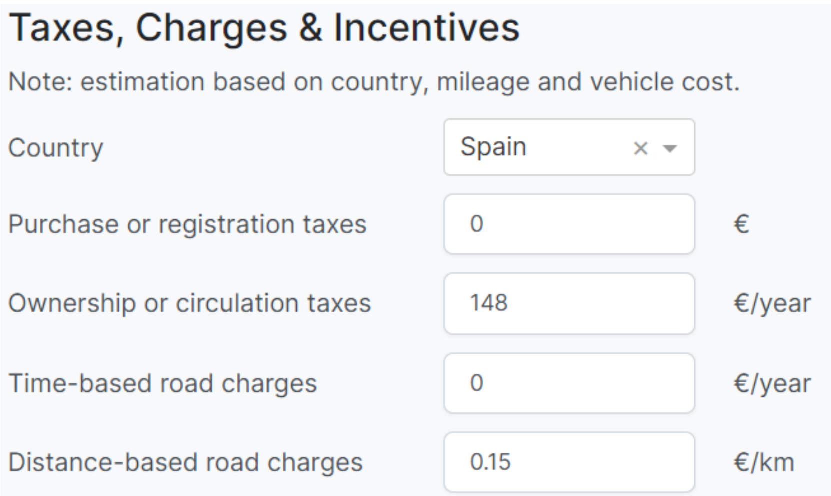


Figure 46: Taxes, Charges & Incentives Input Section

- Diesel cost: selection of consumption and fuel prices.

Figure 47: Diesel Cost Input Section

- Other: definition of maintenance costs and insurance costs.

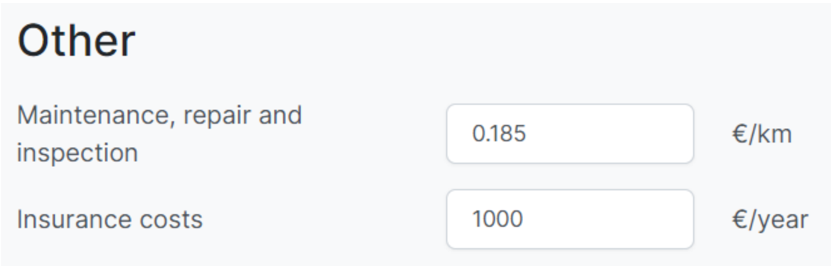


Figure 48: Other Input Section

3.4 Output section

The output section presents the costs in several ways:

- Total Cost of Ownership (TCO), in € and €/(ton km).
- Total life costs (TCO minus resale value), in € and €/(ton km).
- TCO corrected for inflation, in €.

- Electricity consumption in MWh per year in the case of BEVs and Hydrogen consumption in kg per year in the case of FCEVs.

- TCO savings of zero-emission vehicle with respect to baseline diesel vehicle, in € and €/(ton km). This option only appears if all the fields are complete in the diesel baseline configuration tab.

Total Cost of Ownership:

890311 €

0.027 €/(ton km)

Total life costs (TCO minus resale value):

987311 €

0.03 €/(ton km)

TCO corrected for inflation:

755980 €

Yearly electricity consumption:

1731.8 MWh

TCO savings of zero-emission vehicle wrt. diesel vehicle:

174922€

0.005€/(ton-km)

Figure 49: Numerical Output Section

Intractive charts are used to display results in more detail. The following charts are used to provide relevant information:

- Costs breakdown: a sunburst plot that represents life costs breakdown and distinguishes between capital expenses (CAPEX) and operational expenses (OPEX)

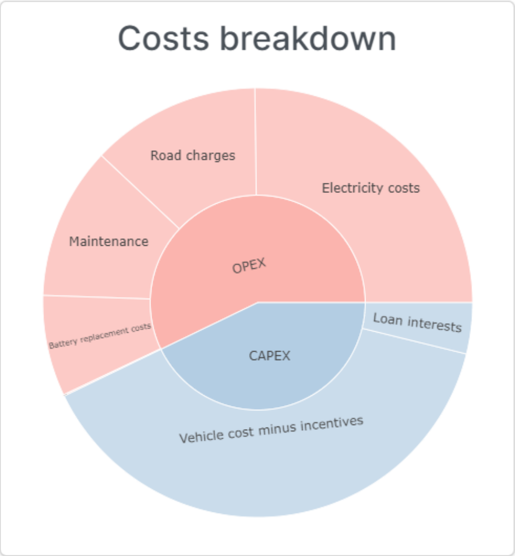


Figure 50: Costs Breakdown Output Section

- Cumulative costs: an area plot showing the cumulative life costs split by category.

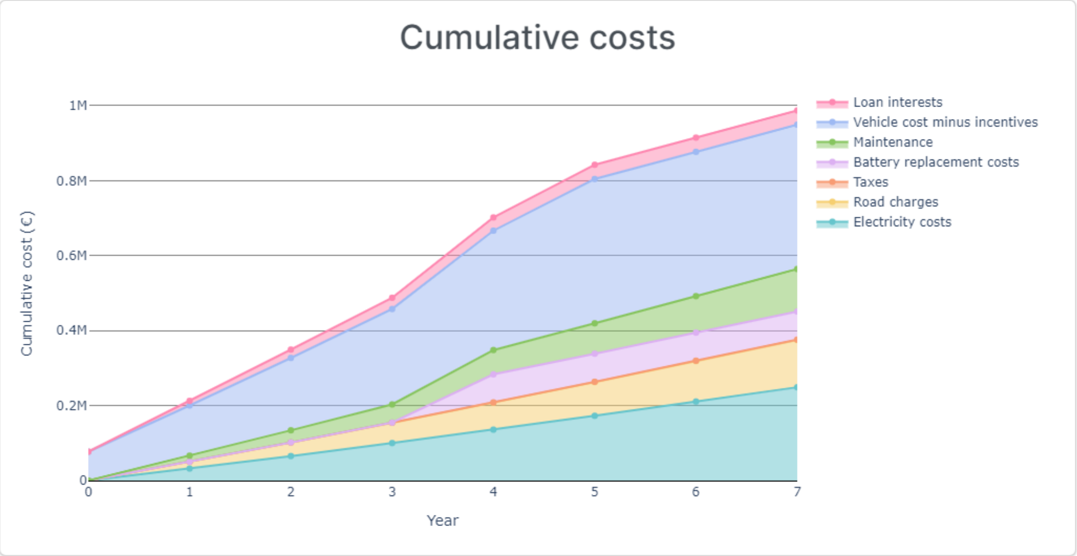


Figure 51: Cumulative Costs Output Section

- Year by year costs: a bar plot for visualising the different categories of costs year by year.

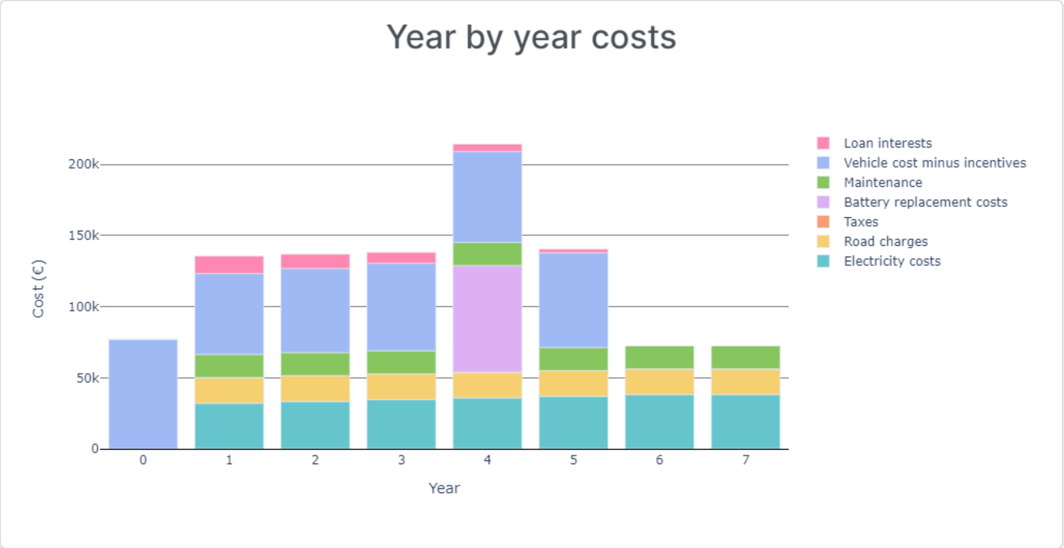


Figure 52: Year by Year Output Section

- Zero-emission versus diesel cumulative costs comparison: a line plot that compares the cumulative costs of a diesel vehicle with respect to the selected zero-emission case. This chart is active only when all diesel inputs are completed. Resale value is not represented in the plots.

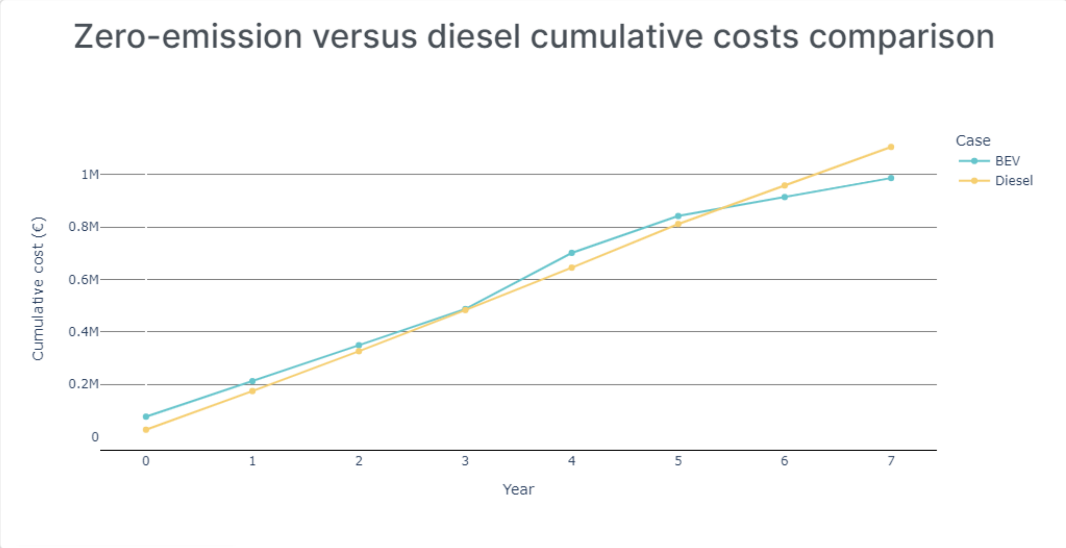


Figure 53: Cumulative Costs Comparison Output Section

3.5 Items included in calculation

The items that make up the TCO are shown and explained in more detail below.

- Vehicle cost
- Loan interests
- Purchase incentives
- Purchase or registration taxes
- Electricity or hydrogen costs
- Ownership or circulation taxes
- Road charges (road tolls or vignettes)
- Maintenance, repairs and inspection
- Battery replacement
- Insurance costs
- Driver wages

In the “Purchase” section, the option of purchasing the truck via a loan is included. The down payment is the fraction of the truck price minus the incentives that is paid at the moment of purchase. The rest is paid during the loan amortization. If no loan is wanted in the calculation, this field can be set to 100%. The loan amortization is of French type, which means that the annual paid amount is constant, and the interests consist of a fix amount with respect the pending amortization. The interest rate is at an annual basis. Resale value is presented as a percentage of the truck price minus incentives.

Purchase incentives are a way that governments and institutions use to help society transition faster towards an overall cleaner transportation, by subsiding a fraction of the purchase cost of a zero or near-zero vehicle. Purchase incentives often make up a substantial part of the vehicle purchasing price and can lean the buyer to opt for a zero-emission vehicle instead of a petrol vehicle when the TCO is foreseen to be lower.

Purchase or registration taxes refers to the one-off tax on the purchase or registration of a new vehicle. Ownership or circulation taxes refer to the annual tax on the ownership of a vehicle. Hydrogen fuel taxes are omitted as it is expected that during the coming years governments exempt it from taxes as a measure to promote hydrogen transportation. Electricity taxes on public charging are not included as a separate field in the input section but can be included in the electricity price itself.

Road charges consist of charges for the usage of the road network. Road charges are divided into those that depend on distance i.e., road tolls, and those that are a fix yearly amount i.e., vignettes.

Both electricity and hydrogen prices are variable in time and inherently uncertain. Because of this variability, the input is not presented as a single constant value but a list of prices that can be changed depending on the year. This way, the user can test various future non-linear price scenarios. Electricity prices can also vary significantly depending on the place of charging. Charging in public chargers may mean that the charging price is higher because there is an overhead for infrastructure maintenance included in the electricity price. To consider this, the price is shown in two columns, one that sets the public electricity price and another one to set the private charging price. Charging efficiency is enabled in the case of BEV vehicles to correct for the extra amount of energy that is consumed by the grid but not absorbed in the battery due to inefficiencies in the charging process. It is set to 88% taking [3] as reference. In the case of fuel cell plug-in vehicles, it is necessary to know the rate of time spent on pure electric driving because both electric and hydrogen-electric driving are present. There is a slider for setting this.

The item "Maintenance, repairs and inspection" groups these three expenses into a single field. Maintenance refers to the periodic efforts to keep the truck working properly, which includes things such as tyre substitution and washing. Repairs are the expenses related to reverting the damage of components, such as broken windows. An expense item which is part of the maintenance is the battery replacement. Battery might need replacement due to its degradation. Because this expense takes place in a specific point in time and it is significant in comparison to the TCO, it is a differentiated item which is more visible in the plots. Although it is rarely expected to have more than one battery replacement along all the vehicle life, the option of more than one replacement is provided. The moment of substitution is distributed evenly throughout the vehicle life.

The formulas used in the calculation are explained in Appendix A – TCO formulation.

3.6 Explanation of the default values

The input section has some default filled-in values that reflect an average scenario to help orient the user in case there are unknown or uncertain variables.

3.6.1 Truck price estimation option

To help the user decide purchase options, a truck estimation menu is left to help the user. The input fields are shown in Table 21.

Table 21. Inputs for truck price estimation

Input	Units
Powertrain type	None (options: BEV or FCEV)
Market maturity	None (options: niche or mass)
Total battery energy capacity	kWh
Total e-drive power	kW
Fuel cell power (FCEV only)	kW
Mass of stored hydrogen (FCEV only)	kg

Price is estimated by adding the cost of each component. Costs of components as a function of their sizing are obtained from literature. Market maturity is a variable that considers the scale of manufacturing. A product sold in a niche market is more expensive to the consumer.

The detailed price calculation can be seen in Appendix B – Method for estimating truck price. In addition, a series of workshops were performed with stakeholders from the ZEFES project. The feedback received from them was also used to determine some default values. The detailed feedback can be seen in Appendix C - Inputs from stakeholders.

3.6.2 Default values from country selection

All inputs in the section named "Taxes, charges and incentives" can be set automatically selecting a country from a dropdown once the yearly mileage and the truck cost fields are filled. These values are collected from a table obtained from the publication "Transport taxes and charges in Europe" [4] and corrected for exemptions on zero-emission vehicles obtained from the Alternative Fuels Observatory webpage [5]. All this data can be checked in Appendix D – Sources for default values.

3.7 Upload/Download functionality

In addition to the manual interface the tool allows, it is possible to interact with the tool using input /output files. To download the data, the button at the top of the results panel is clicked. This file contains relevant inputs and outputs about TCO. To upload the data again the button at the top of the input section is clicked. File type is *.tco* and file format is JSON. JSON format is easily readable and editable, so it is possible to create and modify input files.

4 Validation

For the validation of the simulation platform some simulations have been defined following the input interfaces and simulated. The results are shown in this section along with a description of the output given by the platform and the analysis of the results. For the validation of the platform three different component combinations have been considered, and three different test cases have been defined for the vehicles to carry out, all of them based on repeated VECTO long haul profiles [6] to simulate a standardised real truck long distance logistic operation.

Regarding the simulated vehicles for the validation batch, a FCEV and two BEV have been defined according to the synthesised characteristics that can be seen in the following table. Regarding the other parameters that define each of the components of the vehicle, coherent estimations have been carried out to have a working demonstration of the platform. It must be noted that all the real parameters of the actual vehicle must be known and introduced to obtain a set of results that can be correlated to the real vehicle.

Table 21. Input synthesis for validation batch.

		Tractor	Tractor	Tractor	Tractor	Tractor	semiTrailer1	semiTrailer1	semiTrailer1	Dolly	Dolly	Dolly	semiTrailer2	semiTrailer2	semiTrailer2
		Mass	Bat	FC	H2Tnk	EDUR	Mass	Bat	EDUR	Mass	Bat	EDUR	Mass	Bat	EDUR
Assy_FCEV1		20500 kg	322 kWh	200 kW	100 kg	400 kW	-	-	-	2500 kg	-	-	24000 kg	-	-
Assy_BEV1		8000 kg	498 kWh	-	-	348 kW	20000 kg	-	-	-	-	-	-	-	-
Assy_BEV2		8000 kg	601 kWh	-	-	451 kW	22000 kg	-	-	2500 kg	161 kWh	100 kW	22000 kg	-	-

Three test cases are defined, two test cases that request a VECTO long haul speed cycle replicated until the cycle reaches around 750km, and an acceleration from 0 to 90km/h for performance simulation. Regarding the long-haul test cases, the first one specifies the batteries to start at 100% SoC.

The second long haul cycle is used only for the FCEV and has an initial 3% SoC and will be used to validate that the SPC works properly on low SoC, enough FC power and available H2 situations together with, if it is the case, H2 exhaust considerations in which the VCU must limit the power requests (traction, thermal and ancillaries) and eventually stop the vehicle once there is not enough energy available from H2 nor battery.

4.1 Time series plots

4.1.1 VECTO long haul cycle for FCEV

The VECTO long haul cycle is set as target for the FCEV architecture, consisting of three modules: a 400kW rigid truck unit with a 322kWh battery, a dolly and a semitrailer. There are two cases of study defined for this architecture: initial 100% SoC and initial 3% SoC, both with fully loaded H2 tank. The low initial SoC simulation is done to validate the SPC performance on limited power and energy availability, and to check that the battery limits are considered for SoH of the battery means. Also, regarding the vehicle adaptability and its capability to charge the battery up to the desired boundaries of normal operation using the FC system. Both cases have the default DEMO H2 tank capacity of 100kg availability (considering only usable hydrogen, i.e. after depressurisation considerations).

Regarding the first simulation, it can be observed that the target can be followed with the defined battery and thus it fulfils the power requirements. As can be seen the SPC splits the requirement of power between battery system and FC system, achieving a stationary range of SoC around 60%. This stabilisation SoC can be tuned via the VCU parameters that define the SPC, setting the SoC values between which the SPC will be used.

Also, since the SoC is reduced during the simulation, the equivalent H2 consumption is over the actual H2 consumption due to the energy given by the battery, which is corrected to equivalent H2 with the corresponding efficiency considerations.

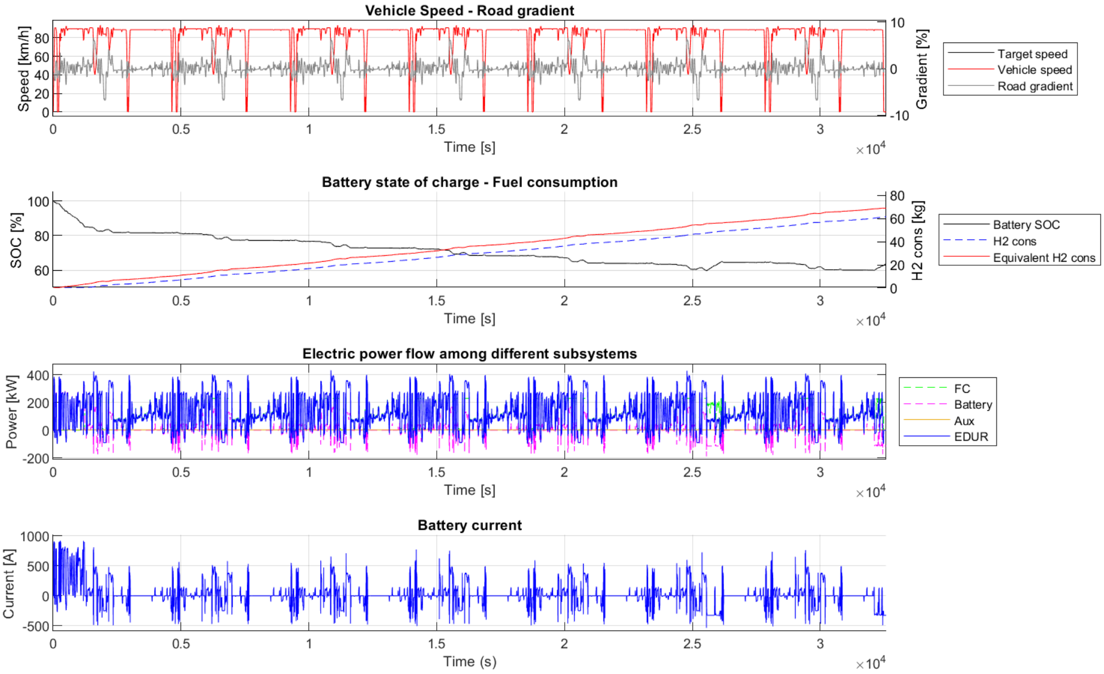


Figure 54: FCEV initial 100% SoC, VECTO long haul cycle 750km.

In the first stage of the FCEV on the long-haul cycle, it can be observed that the SPC is assigning all the power demand to the battery system, keeping the FC system off. This is done according to the set parameters of the VCU control, in which the system is expected to drain the battery until it reaches a SoC under 90%, point in which the SPC starts the calculation of the division of demands battery-FC.

In the second stage, the SPC is activated, when SoC reaches the 90% threshold. After this, the SPC begins to request power to the FC, but in a way that it keeps slowly discharging the battery until it reaches the steady SoC range of around 60%.

In the next simulation, the same profile has been simulated but with an initial 3% SoC, testing the capability for the vehicle to use the FC to supply the power requirements of the cycle while charging the battery until it reaches a stable SoC range, again defined by the same VCU parameters. During the simulation, several characteristics can be observed regarding the SPC performance and its interaction with FC and battery.

First, on the very first stage, the traction limited section is occurring, in which the SoC is under 10% and the SPC logic applies a limitation to the battery to avoid discharging it. In this section, which goes until around minute 20, the speed profile is slightly limited by the VCU, and the battery current is always negative (charging mode), with some punctual positive peaks that are limited inversely proportional to the SoC. In this case, the traction limited range is quite narrow, and the limitation can only be seen in the first speed profile, in which the vehicle speed is lower than the target during some seconds.

Secondly, the battery charging section, which goes until around 3 hours 45 minutes. In this section the SPC is aware of battery SoC being under the set boundary and thus it is in charging mode, but it does not apply any traction limitations, discharging the battery if it is needed to supply the power consumers. In this section the SoC keeps raising due to the average power demand being lower than the FC maximum power. This section keeps charging the battery until the stationary SoC range is reached, at around 60% SoC.

Finally, the normal operation range, in which the SPC divides the power supply to keep the battery SoC at around 60%. This operation mode can be kept stable until H2 tank runs out of hydrogen, when the battery will discharge again, reaching the traction limited operation and eventually stopping the vehicle. That last stage is not in this simulation because the H2 in the tank was not fully used.

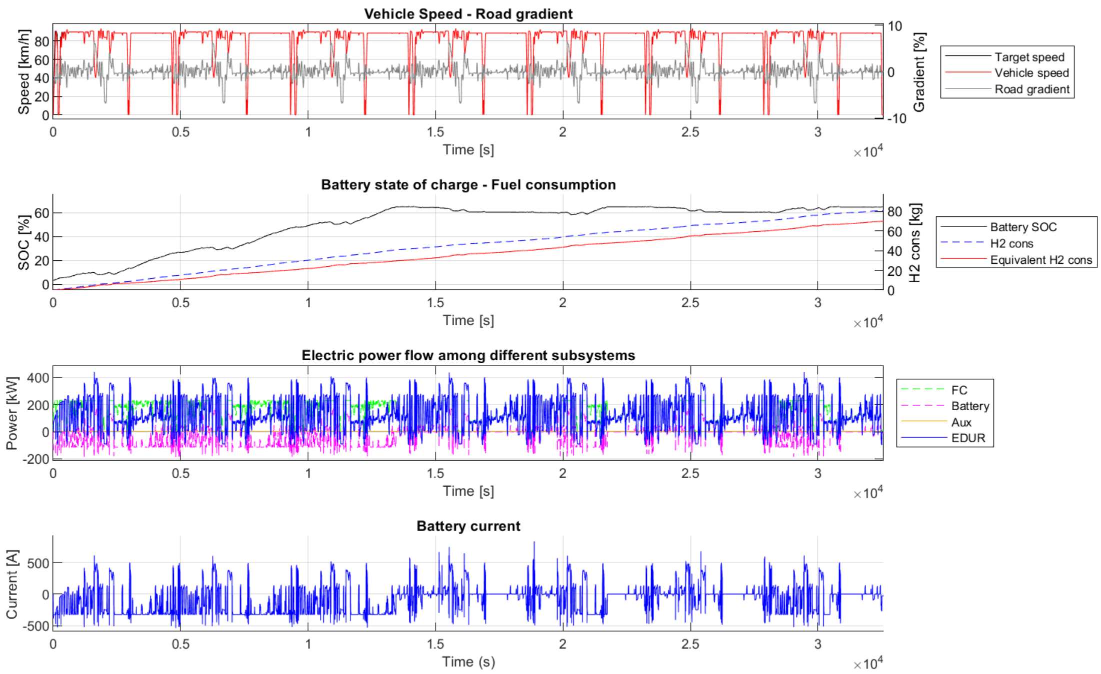


Figure 55: FCEV initial 3% SoC, VECTO long haul cycle 750km.

As can be seen in this long-haul cycle, since the initial SoC was lower than the final SoC, the equivalent H2 consumption is lower than the real H2 consumption because part of the H2 consumption was not used to operate the vehicle but to charge the battery up to a normal SoC level.

4.1.2 VECTO long haul cycle for BEV1 and BEV2

In this section the VECTO long haul cycle is set as target for the BEVs architecture. The first simulation has a 498kWh battery configuration, defined using the default DEMO parameter definition with a parallel scaling, and a bi-modular assembly consisting of a 348kW tractor and a semitrailer. The second configuration has a 451kW tractor with a 601kWh battery and it carries a first semitrailer, a dolly and a second semitrailer. In both cases, the fixed transmission ratio has been set to a value that allows to reach nominal speeds and maximise the usable torque.

As can be seen, the battery capabilities are not enough to reach the desired range of the long-haul profile. However, the vehicle range can be determined from these simulations, using the final vehicle distance KPI to obtain a range value for the specific simulated speed and slope profiles.

In the case of the BEV1 assembly, the vehicle stops after 4 hours 50 minutes of simulation, having reached the minimum SoC value of 2%, set as a parameter on the VCU configuration. As can be noted, the last part of the vehicle speed was in the traction limited range due to the SoC being under the VCU parameter that sets the boundaries for the application of the limitation.

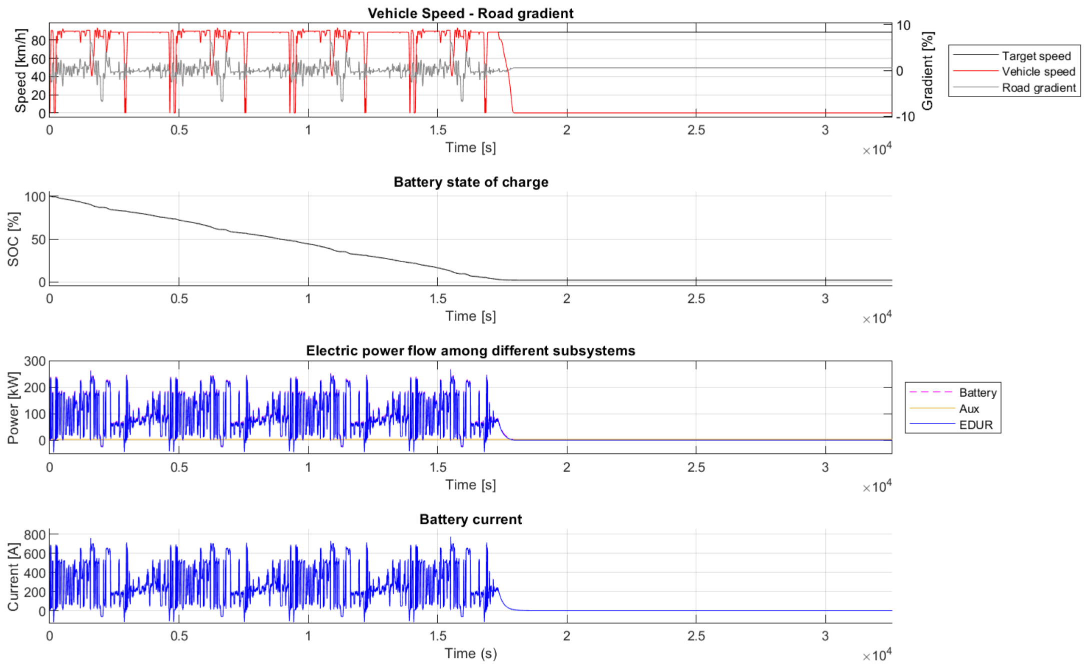


Figure 56: BEV1 initial 100% SoC, VECTO long haul cycle 750km.

The same cycle has been requested to simulate with the BEV2. In this case, having a 498kWh battery, i.e. more capacity than BEV1, is not enough to supply the required energy to move the extra payload along a long-haul profile for a longer distance, since this simulation stops the vehicle sooner than the previous one, after around 4 hours and 40 minutes of simulation.

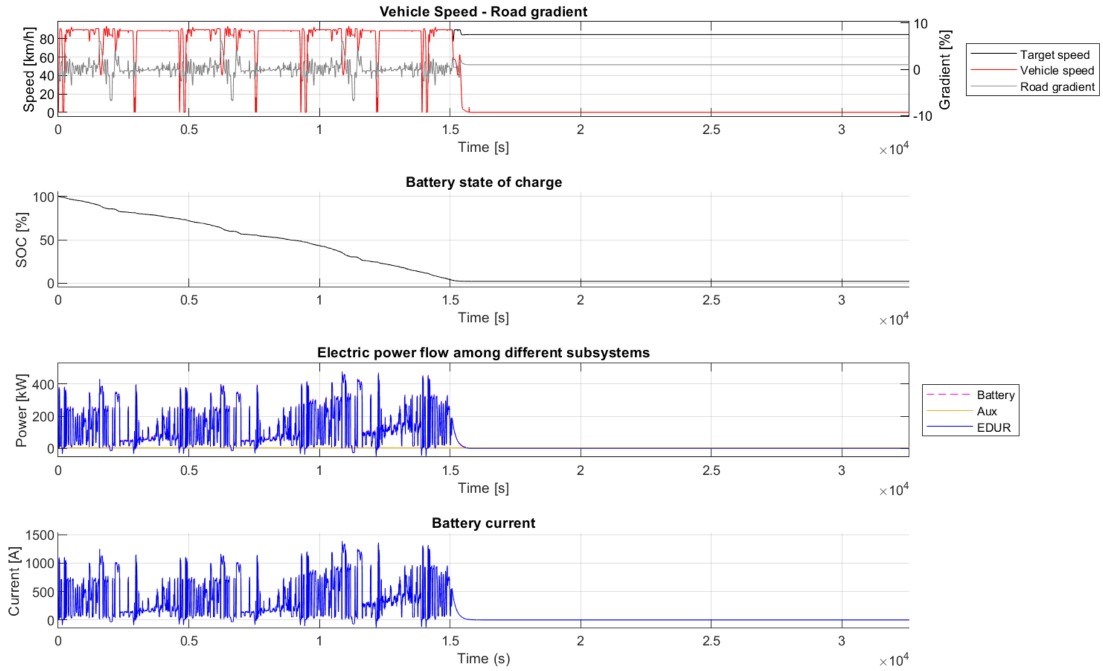


Figure 57: BEV2 initial 100% SoC, VECTO long haul cycle 750km.

4.1.3 Acceleration for BEVs and FCEV

An acceleration test has also been included in the simulation tool, allowing the user to test a specific vehicle configuration in an acceleration profile, thus getting the 0 to 90 km/h time, together with the same time plots as the other simulations. All the simulated truck configurations can reach 90km/h.

For the BEVs, the 0 to 90km/h acceleration time is 22 seconds for BEV1 and 30 seconds for BEV2.

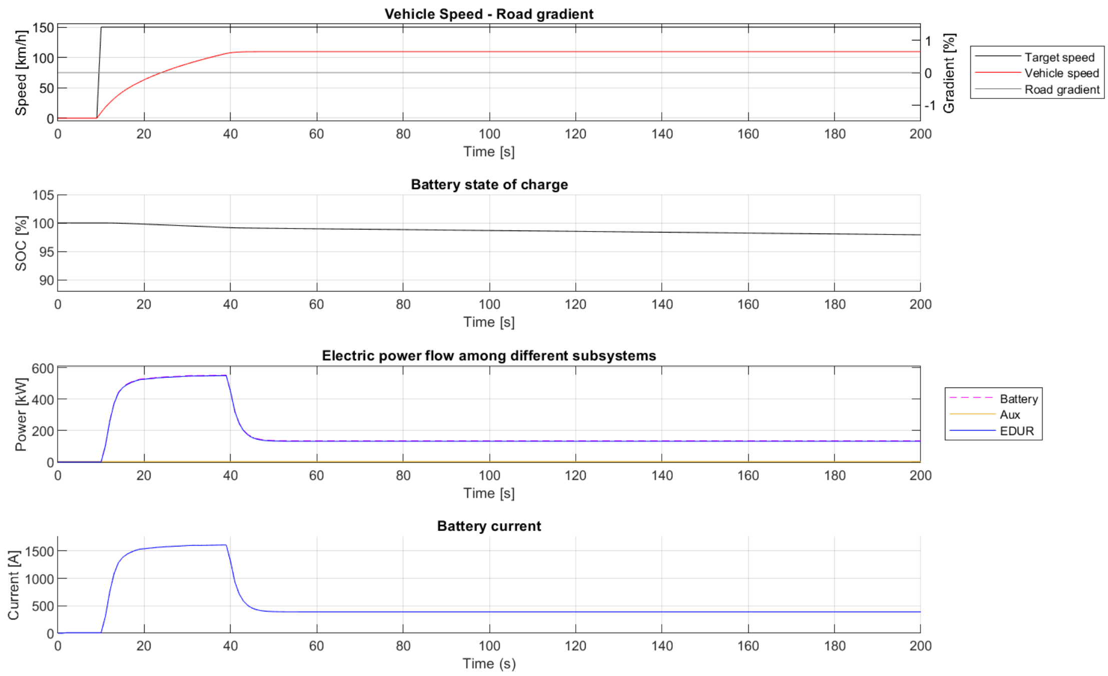


Figure 58: BEV1 0-90km/h acceleration test.

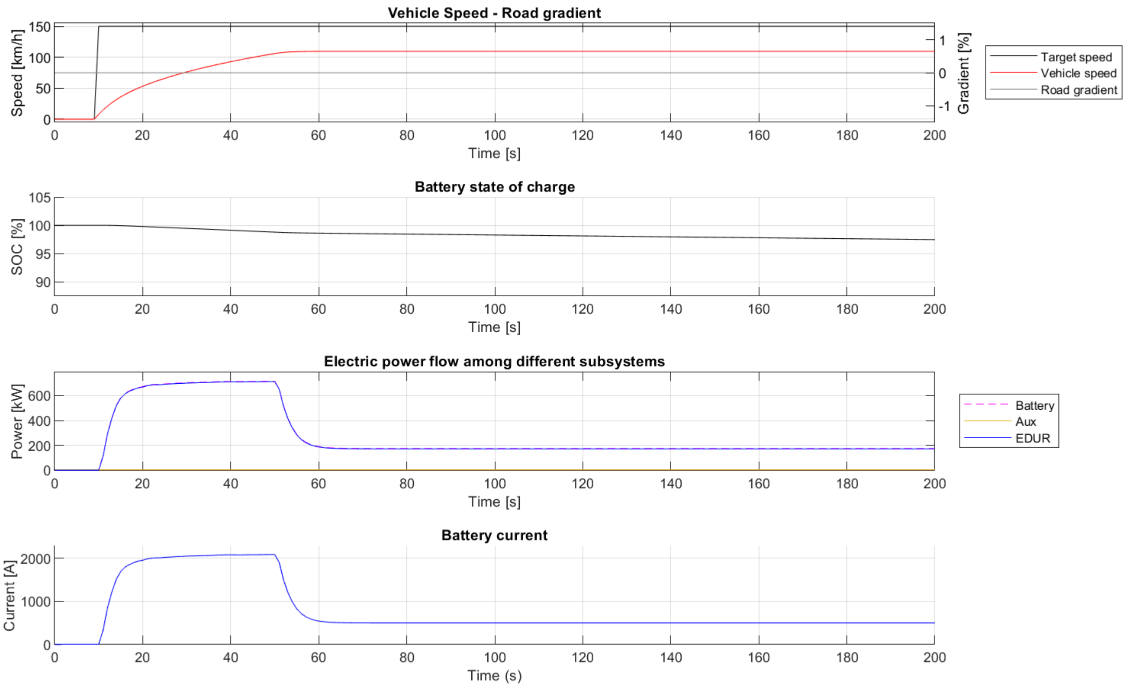


Figure 59: BEV2 0-90km/h acceleration test.

The simulated FCEV can also reach the target, and this configuration has a reaching time notably slower than the simulated BEVs, of 37 seconds.

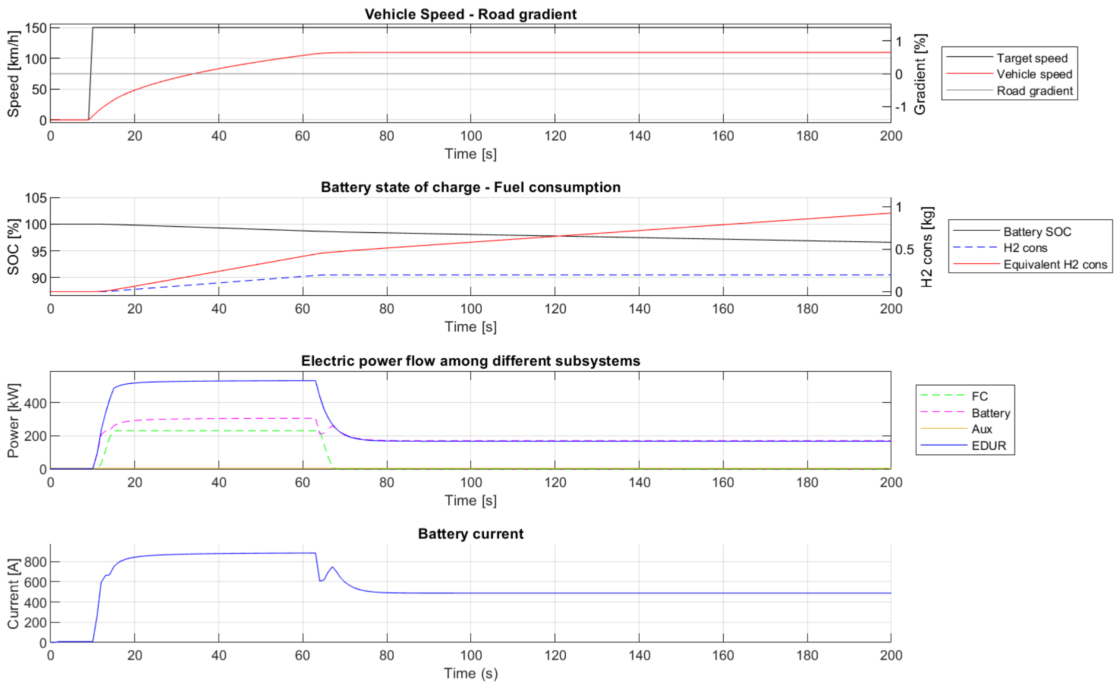


Figure 60: FCEV 0-90km/h acceleration test.

In the three acceleration simulations, a steady speed of around 110km/h is reached, and the truck will not accelerate more that. This is because, for this example, the transmission ratio defined for all the configurations has been chosen so that the truck can reach the motor maximum speed at around 110km/h, which is the steady state that is being reached in the three simulations.

4.2 KPIs table output

For each of the batch requests, the simulation tool gives as an output a table file with KPIs that give information of the performance and consumption of each configuration. Different test types give different KPIs, and they are given in both an *.xlsx* and *a .csv* format, for a better availability for all users.

To evaluate the defined truck configuration, the KPIs of each of the test cases are obtained from the tool and available in an understandable format. For the simulation batch carried out as a validation, the KPIs given by the platform are the ones on the following table.

Table 21. Simulation KPIs of the DEMO template used for the platform validation.

TestCase	*KPIs*	*Assy_FCEV1*	*Assy_BEV1*	*Assy_BEV2*
Vehicle	FC_NomPow_kW	200.00	0.00	0.00
Vehicle	EDU_NomContPow_kW	400.43	348.13	550.79
Vehicle	Bat_NomECap_kWh	322.48	498.37	762.22
Vehicle	H2Tnk_H2Mass_kg	100.00	0.00	0.00
TC_VECTO_LongHaul_SOC100	Bat_ElecConsDC_kWhpkm	0.15	1.19	1.66
TC_VECTO_LongHaul_SOC100	Bat_ElecEnerDC_kWh	114.96	486.21	585.43
TC_VECTO_LongHaul_SOC100	Bat_endSOC_perc	63.82	2.00	2.00
TC_VECTO_LongHaul_SOC100	FC_H2Cons_kgpkm	0.08	0.00	0.00
TC_VECTO_LongHaul_SOC100	FC_H2Mass_kg	61.57	0.00	0.00
TC_VECTO_LongHaul_SOC100	Veh_Dist_km	750.00	407.35	352.80
TC_VECTO_LongHaul_SOC100	Veh_TotElecConsDC_kWhpkm	1.62	1.19	1.66
TC_VECTO_LongHaul_SOC3	Bat_ElecConsDC_kWhpkm	-0.27		
TC_VECTO_LongHaul_SOC3	Bat_ElecEnerDC_kWh	-202.72		
TC_VECTO_LongHaul_SOC3	Bat_endSOC_perc	65.08		
TC_VECTO_LongHaul_SOC3	FC_H2Cons_kgpkm	0.11		
TC_VECTO_LongHaul_SOC3	FC_H2Mass_kg	80.81		
TC_VECTO_LongHaul_SOC3	Veh_Dist_km	750.00		
TC_VECTO_LongHaul_SOC3	Veh_TotElecConsDC_kWhpkm	1.62		
TC_Accel90	Veh_Accel90Time_s	37	22	30

This output table has all the combinations of configuration vs. test case that were requested on the simulation batch that was given as input to the tool.

Regarding the long-haul test with an initial 3% SoC for the FCEV, the electric consumption of the battery and the per kilometer consumption have negative values. This means that the battery was charged during the simulation, accumulating an energy amount of 202kWh, which came from the FC system. Thus, according to this example configuration with the estimated parameters, this FCEV configuration could set the battery SoC in the correct boundaries and at the same time follow a VECTO long-haul cycle, reaching over the 750km range that was requested for the simulation. The KPIs table also gives information of the consumed H2 for the specific simulation, which in the normal long-haul was 62kg, and in the initial 3% SoC case was of 81 kg of hydrogen, which was partially used to charge the battery to 65% SoC.

On the other hand, for the BEVs the long-haul could not be held for more than 407km and 352km respectively. The range of the BEV1 assembly would fulfil the 400km range requirements for the ZEFES project demonstrators, while the BEV2 assembly results would not be enough for covering the 400km range, so sizing optimisations should be carried out on the components to achieve the objectives.

Also, there is the *Vehicle* section in which basic characteristics of the vehicle configuration, regarding FC, Battery nominal capacity, the EDUs nominal power and the H2 tank capacity in case the configuration has a H2 tank on its architecture. These Vehicle KPIs give basic information of the assembly, and they aggregate all the modules of the truck, being that if for example there is an e-Trailer with battery and motor, that capacity and power are summed into these KPIs, giving thus the value of the whole vehicle capabilities.

5 Results and discussion

5.1 Results

The development of this task has resulted in the creation of two simulation tools: a multi-architecture modelling platform and a Total Cost of Ownership calculation tool. Also, as part of the task, online accessibility has been implemented, via the creation of a graphic interface. These tools are available to all project partners and will be updated during the life of the project.

5.2 Contribution to project objectives

This deliverable contributes to the achievement of the following objectives of the project:

- Objective 1: improve modular Heavy Duty (HD) Battery Electric Vehicles (BEVs) and Fuel Cell Electric Vehicles (FCEVs).
    - Sub-objective 1.3: add functionality to the co-design tool especially for HD, to help to choose the right sizes for components.
- Objective 3: provide digital and fleet management tools specifically for HD ZEVs, fleet integration with remote operational optimisation of vehicle performance.
    - Sub-objective 3.1: develop and validate truck Digital Twins (DTs) and fleet management tools.
- Objective 5: define pathways for a significant price reduction and volume increase.
-

5.3 Contribution to major project exploitable result

The main contribution of this deliverable to the project exploitable results is the development of more efficient HD ZEV (BEV/FCEV) flexible vehicle platforms.

6 Conclusions and Recommendations

This document reports the work carried out to develop the multi-architecture modelling platform and the TCO calculation tool to fulfil the requirements and needs of the technologies developed in the project. Both tools are already functional and available to the project partners.

Both tools are functionally complete, and additional development on the models will be performed to further improve the quality of the models and validate their coherence with the experimental data obtained in the demonstrators.

7 Risks and interconnections

7.1 Risks/problems encountered

Risk No.	What is the risk	Probability of risk occurrence[1]	Effect of risk[1]	Solutions to overcome the risk
1	Lack of information/ development of the third-party components (2.3.6)	2	2	A simplified model of the component will be developed and used as a placeholder
2	Input data missing or not compatible (2.3.2)	2	2	Input system has been developed to ensure lack of data will be completed automatically and will be made compatible with the system

[1] Probability risk will occur: 1 = high, 2 = medium, 3 = Low

7.2 Interconnections with other deliverables

The simulation platform will be used in tasks T2.3 and T2.4 related to the optimization of the components, which will be described in D2.2 and D2.3.

The results obtained from this deliverable, mainly the simulation platform, will be used as the basis for T4.2, and consequently D4.1.

8 Deviations from Annex 1

This task has been delayed due to the complexity of the development of the models, definition and implementation of the multi-architecture platform and development of the web service with a specific security following the rules of sharing data for both BEVs and FCEVs with their combination with e-trailers. For this reason, this task has been delayed to M16. To mitigate this delay, limited versions of the simulation platform were made available to share with partners in T2.3/T2.4 and T4.1/T4.2. An interim, confidential version of deliverable D2.1 has been kept at the original date and a final version of D2.1 scheduled for M16, which is publicly available. In addition, even if the task will be considered as completed by M16, additional refinement of the models will be done until at least M24, to ensure that the platform keeps its relevance during the life of the project and integrates the outcomes of the different project tasks.

9 References

[1] Olmos, Josu, et al. *Modelling the cycling degradation of Li-ion batteries: Chemistry influenced stress factors*. Journal of Energy Storage 40 (2021): 102765.

[2] Gundogdu, Burcu, and Daniel Thomas Gladwin. *A fast battery cycle counting method for grid-tied battery energy storage system subjected to microcycles*. 2018 International Electrical Engineering Congress (IEECON). IEEE, 2018.

[3] https://go-e.com/en/magazine/ev-charging-losses

[4] https://op.europa.eu/en/publication-detail/-/publication/4de76a04-a385-11e9-9d01-01aa75ed71a1 European Commission, Directorate-General for Mobility and Transport, El Beyrouty, K., Gatto, M., Essen, H., et al., *Transport taxes and charges in Europe : an overview study of economic internalisation measures applied in Europe*, Publications Office, 2019 https://data.europa.eu/doi/10.2832/416737

[5] https://alternative-fuels-observatory.ec.europa.eu/

[6] https://code.europa.eu/groups/vecto/-/wikis/home

[7] https://theicct.org/publication/purchase-cost-ze-trucks-feb22/

[8] https://theicct.org/wp-content/uploads/2022/01/Final-Report-eTruck-Virtual-Teardown-Public-Version.pdf

10 Acknowledgements

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

#	Partner short name	Partner Full Name
1	VUB	VRIJE UNIVERSITEIT BRUSSEL
2	FRD	FORD OTOMOTIV SANAYI ANONIM SIRKETI
4	KAE	KASSBOHRER FAHRZEUGWERKE GMBH
5	REN	RENAULT TRUCKS SAS
6	SCA	SCANIA CV AB
7	VET	VAN ECK TRAILERS BV
8	VOL	VOLVO TECHNOLOGY AB
9	ABB	ABB E-MOBILITY BV
9.1	ABP	ABB E-MOBILITY SPOLKA Z OGRANICZONAODPOWIEDZIALNOSCIA
10	AVL	AVL LIST GMBH
11	CM	SOCIEDAD ESPANOLA DE CARBUROS METALICOS SA
11.1	APG	AIR PRODUCTS GMBH
12	HEPL	HITACHI ENERGY POLAND SPOLKA Z OGRANICZONA ODPOWIEDZIALNOSCIA
13	MIC	MANUFACTURE FRANCAISE DES PNEUMATIQUES MICHELIN
14	POW	PLASTIC OMNIUM NEW ENERGIES WELS GMBH
15	RIC-CZ	RICARDO PRAGUE S.R.O.
15.1	RIC-DE	RICARDO GMBH
16	UNR	UNIRESEARCH BV
17	ZF	ZF CV SYSTEMS HANNOVER GMBH
18	ALI	ALLIANCE FOR LOGISTICS INNOVATION THROUGH COLLABORATION IN EUROPE
19	DPD	DPD (NEDERLAND) B.V.
20	COL	ETABLISSEMENTEN FRANZ COLRUYT NV
21	GRU	GRUBER LOGISTICS S.P.A.
22	GBW	GEBRUEDER WEISS GESELLSCHAFT M.B.H.
23	PG	PROCTER & GAMBLE SERVICES COMPANY NV
23.1	PGP	PROCTER AND GAMBLE POLSKA SPOLKA Z OGRANICZONA ODPOWIEDZIALNOSCIA
23.2	PGA	PROCTER & GAMBLE AMIENS
24	PRI	PRIMAFRIO CORPORACION, S.A.
25	PTV	PTV PLANUNG TRANSPORT VERKEHR GmbH
26	Fraunhofer	FRAUNHOFER GESELLSCHAFT ZUR FORDERUNG DER ANGEWANDTEN FORSCHUNG EV
27	HAN	STICHTING HOGESCHOOL VAN ARNHEM ENNIJMEGEN HAN
28	IDI	IDIADA AUTOMOTIVE TECHNOLOGY SA
29	TNO	NEDERLANDSE ORGANISATIE VOOR TOEGEPAST NATUURWETENSCHAPPELIJK ONDERZOEK TNO
30	UIC	UNION INTERNATIONALE DES CHEMINS DE FER
31	CFL	CFL MULTIMODAL S.A.
32	GSS	Grupo Logistico Sese

33	HIT	Hitachi ABB Power Grids Ltd.
34	IRU	UNION INTERNATIONALE DES TRANSPORTS ROUTIERS (IRU)
35	RIC-UK	RICARDO CONSULTING ENGINEERS LIMITED

Disclaimer/ Acknowledgment

Funded by
the European Union

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Commission. Neither the European Union nor the granting authority can be held responsible for them.

Appendix A – TCO formulation

The total cost of ownership is:

$$TCO = I + \sum_{i=1}^{N} OPEX_i - P$$

Where P is the resale value and I is the initial investment:

$$P = (V_p - V_i) * r_r/100$$
$$I = V_p - V_i + T_p$$

Where:
V_p: vehicle purchase [€]
V_i: vehicle purchase incentives [€]
T_p: purchase and registration taxes [€]
r_r: resale value in percent [%]
The yearly operation expenses $OPEX_i$ in each year $i = 1, 2, \ldots N$ of the N years of life is:

$$OPEX_i = X_i + T_o + R + M + I_c + W + L_i$$

Where:
T_o: ownership taxes [€]
R: road tolls [€]
M: maintenance, repairs, and inspections [€]
I_c: insurance costs [€]
W: driver wages [€/km]
L_i: loan interests paid at year i [€]
X_i are the expenditures corresponding to recharging in the case of BEVs and to hydrogen in the case of FCEVs:

$$X_i = \begin{cases} P_{e,i} * C_e * D, & for\ a\ BEV \\ P_{H2,i} * C_{H2} * D, & for\ a\ FCEV \end{cases}$$

Where:
D: yearly mileage [km]
$P_{e,i}$: electricity price in the year i [€/kWh]
C_e: electricity consumption [kWh/km]
$P_{H2,i}$: hydrogen at-the-pump price in the year i [€/kg]
C_{H2}: hydrogen consumption [kg/km]
The ownership taxes depend on time and distance:

$$T_o = T_t + T_d * D$$

Where:
T_t: time-based taxes [€/year]
T_d: distance-based taxes [€/km]
Driver wages are based on distance:

$$W = W_d * D$$

Where W_d are distance-based driver wages [€/km].

Loan interests are calculated according to the French loan method with constant interest rate. The needed parameters to define the loan are loan size S_l [€], loan length N_l [years] and yearly interest rate r_l. The repayment term is:

$$C_l = S_l * \frac{r_l * (1 + r_l)^{N_l}}{(1 + r_l)^{N_l} - 1}$$

The loan interests $i_{l,i}$ are calculated as a constant interest rate applied to the pending loan amount $P_{l,i}$, the amortized amount $A_{l,i}$ is the difference between the repayment term and the loan interests, and the pending amount decreases by the amortized amount with respect to the previous year.

$$i_{l,i} = r_l * P_{l,i}$$
$$A_{l,i} = C_l - i_{l,i}$$
$$P_{l,i} = P_{l,i-1} - A_{l,i}$$

In the time of opening the loan ($i = 0$), the pending interests are the loan size $P_{l,0} = S_l$.

In the next year of the loan length and afterwards interests are zero if the loan length is shorter or equal than the life length. Otherwise, in the model used here it is considered that the interests pending after the vehicle life are returned in the last year. Mathematically:

$$L_i = \begin{cases} 0, & N_l \leq N,\ i = N_l + 1, \dots, N \\ \sum_{i=N}^{N_l} L_i, & N_l > N,\ i = N \end{cases}$$

The net present value of the TCO is also calculated:

$$NPV = I + \sum_{i=1}^{N} \left(\frac{OPEX_i}{(1 + r)^i} \right) - \frac{P}{(1 + r)^N}$$

Where r is the yearly inflation rate.

Appendix B – Method for estimating truck price

The method for estimating the truck price is described in this appendix.

Each element has associated a price that depends on the sizing of the specific component or a fixed price. The truck purchase price consists of the direct manufacturing costs, which consist of the total material cost and labour cost plus indirect manufacturing costs, that are added as a constant rate to the manufacturing expenses and is already included in the component costs in the table. Indirect costs raise the purchase price to about 40% and are expected to decrease as the market grows in the coming years [7].

With the exception of battery costs, all data sources of all the figures in Table 22 are extracted from Ricardo Strategic Consulting report [8] and comparisons are made with a report on component costs from the International Council on Clean Transportation [7]. Battery costs have been extracted from said report of the ICCT [7].

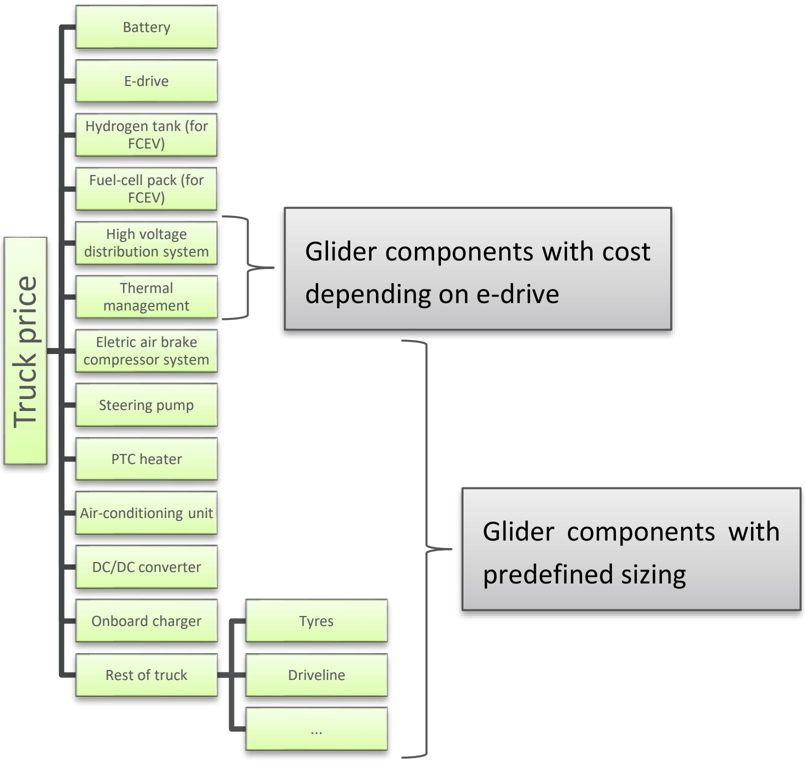


Figure 61: Vehicle component teardown

Table 22: Component costs used

Component	Powertrain type	Market maturity	Cost	Unit	Notes
Battery	BEV	niche	374	€/kWh	niche is 30% additional cost over mass
Battery	BEV	mass	288	€/kWh	
Battery	FCEV	niche	780	€/kWh	
Battery	FCEV	mass	600	€/kWh	
E-drive		niche	92	€/kW e-drive power	
E-drive		mass	60	€/kW e-drive power	
Hydrogen tank		niche	1708	€/kg tank capacity	
Hydrogen tank		mass	1180	€/kg tank capacity	
Fuel cell		niche	1145	€/kW fuel cell power	niche is average price in 2020, mass is average price in 2035
Fuel cell		mass	458	€/kW fuel cell power	
Air conditioning			641	€	
DC/DC converter		niche	396	€	
DC/DC converter		mass	366	€	
Brake compressor			8244	€	
HV distribution system	BEV		25	€/kW e-drive power	
HV distribution system	FCEV		23	€/kW e-drive power	
Onboard charger	BEV		2350	€	niche is applied a 19% price reduction; mass is applied a 28% price reduction
Onboard charger	BEV		2089	€	
Onboard charger	FCEV		328	€	
Onboard charger	FCEV		291	€	
PTC heater			687	€	
Steering pump			2473	€	
Thermal management	BEV		19	€/kW e-drive power	
Thermal management	FCEV		8	€/kW e-drive power	
Rest of truck			27007	€	average price of a day cab tractor is $117K and 18%of its price consists of driveline, cab and chassis; $1 = 0,916€

Appendix C - Inputs from stakeholders

Two online workshops were performed with stakeholders from the ZEFES project. In the first workshop, several OEMs attended, in the second, stakeholders from transport operators participated.

In the workshops, an overview of ZEFES WP2 was presented, following an explanation of the TCO tool, detailing the inputs and basic usage. Afterwards, a live demonstration of two use cases of TCO computation was made with a prototype of the tool. While the explanation took place, participants were asked to answer an online survey with questions regarding their experience on truck manufacturing and operation and on feedback related to the TCO tool usage. The answers to each question were constrained to a close set, although participants were allowed to further explain their answers if they wished to. A different set of questions was selected for each workshop separately considering the specific field of knowledge of the stakeholders. Feedback from stakeholders is intended to improve the prototype of the tool both by getting to know their needs and through inputs from their personal knowledge.

The specific questions presented in the workshop with OEMs and the answers were:

Question 1: How long is the first life of an average heavy-duty truck (in years)?

- Less than 7 years - 3
- 7-10 years - 2
- 10-13 years - 2
- More than 13 years - 0
- I don't know / I don't want to answer - 0

Question 2: What is the expected life of a new battery in a heavy-duty vehicle (in years)?

- Less than 6 years - 1
- Between 6 and 8 years - 2
- Between 8 and 10 years - 2
- More than 10 years - 0
- I don't know / I don't want to answer - 1

Question 3: Do you think the price output obtained from the truck estimation price is realistic?

- No, it’s very low (less than 50% of typical value) - 0
- No, it’s low (between 51 and 80% of typical value) - 0
- Yes, it’s accurate (between 81 and 120% of typical value) - 0
- No, it’s high (between 121 and 150% of typical value) - 0
- No, it’s very high (more than 151% of typical value) - 0
- I don’t know / I don’t want to answer - 3

Question 4: Overhead costs of public charging are not included as a separate input but can be included in the electricity price itself. Should a price input be added?

- Yes - 6
- No - 1
- I don't know / I don't want to answer - 0

The specific questions presented in the workshop involving transport companies and the possible answers were:

Question 1: How long is the first life of an average heavy-duty truck (in years)?

- Less than 7 years - 3
- 7-10 years - 1
- 10-13 years - 0
- More than 13 years - 0
- I don't know / I don't want to answer - 1

Question 2: How does the rate of use of a heavy-duty truck change as the truck gets older?

- The use of the vehicle diminishes - 0
- The use of the vehicle remains the same - 4
- The use of the vehicle increases - 0
- It depends on the vehicle - 0
- I don't know / I don't want to answer - 0

Question 3: What is a realistic resale value for the first life of a truck?

- Up to 10% of purchase value - 1
- 11 to 30% of purchase value - 2
- 31 to 50% of purchase value - 0
- 51 to 70% of purchase value - 0
- Over 71% of purchase value - 0
- I don't know / I don't want to answer - 2

Question 4: When replacing a battery in a BEV, do you consider the old battery as a sellable asset?

- Yes, it is sold for other uses - 2
- No, it is considered a waste and is recycled - 0
- I don't know / I don't want to answer - 3

From the workshop from OEMs some useful information for the design of the tool was extracted. The life length of a new battery is roughly equal to the life length of the vehicle, so we can expect in most of the cases there is one battery replacement maximum during the whole vehicle life. Most of the participants answered that overhead costs of public charging should be a necessary input, so the option for setting different prices for public and private charging was included along with a slider to set the rate of time spent on public charging. From the fourth question it is impossible to validate the result of the price estimation.

From the workshop with transport companies the following conclusions were taken. All the participants answered in question 2 that the rate of use of the vehicle stays the same, so it is reasonable to use constant value of yearly mileage over the years. From the answers to question 3, we chose a default resale value of 20% of vehicle purchase value minus incentives. The majority of answers to question 4 didn’t know if the battery is sellable after the end of its first life or didn’t respond, so we decided to omit battery resale value input in the tool. In question 4 the participants that answered affirmed that used batteries can be a sellable asset, so new inputs were added to the tool that consider the expenses of replacing the battery, the cost of the new battery and the gains of the resale of the old battery.

Question 1 is common to both workshops and is intended to set the default value of life length. It has been decided to set it to 7 years.

Some participants noted that they were interested in a feature to compare the TCO of the zero-emission vehicle with that of a diesel vehicle. This option was added afterwards.

Appendix D – Sources for default values

In this appendix the source of the values for estimating the purchase or registration taxes, purchase incentives, ownership or circulation taxes, time-based road charges and distance-based road charges.

These values are determined by powertrain type, yearly mileage, truck price and country. There are many other factors that have an influence on these items of the TCO, plus they change over time as legislation changes, however, they serve as an estimation.

Taxes and charges are shown in Figure 62. The columns with grey heading are obtained from [4] considering a typical case of heavy-duty truck, as the database does not contemplate heavy-duty BEVs or FCEVs. The changes for zero-emission vehicles are made afterwards. The assumptions for the truck for filtering taxes and charges in the database are the following. Vehicle type is set to “Truck trailer (>32t)”, as it is assumed that this is a very prevalent type in heavy duty transport. Fuel type is set to Fuel efficiency is set to “High” instead of “Low” and emission class is set to “Euro 6” instead of “Euro 3” (higher emission category) because by today’s standards, most diesel freight transport in the roads has become relatively efficient and will continue to improve. Fuel type is set to “Diesel” instead of “LNG” (Liquified Natural Gas) because it is the most used fuel in heavy duty transport. With the selected configuration, there is no available data for BEV of FCEV vehicles. In the case of zero-emission vehicles, taxes and charges can be the same, in the case there is no legislation that differentiates these from traditional vehicles, or they are lower in the case there is a government exemption. VAT are added for ownership or circulation tax and purchase or registration tax.

The exemptions on zero-emission vehicles of each country and the purchase incentives are obtained from the European Alternative Fuels Observatory (EAFO) website [5]. The computation is divided into a fix subtraction or addition plus a part that depends on vehicle price, as it is often calculated as a percentage of truck price. For this objective, separate columns represent subtraction or addition and a multiplier. For example, a purchase incentives multiplier of 0,35 means that there is a 65% exemption.

$$Purchase\ or\ registration\ taxes = Tax\ on\ diesel\ vehicle * (1 + \frac{VAT}{100}) - Subtractor - Multiplier * Truck\ price$$

$$Incentives = Subsidies\ adder + Truck\ price * Multiplier$$

$$Ownership\ or\ circulation\ taxes = Tax\ on\ diesel\ vehicle * (1 + \frac{VAT}{100}) - Subtractor - Multiplier * Truck\ price$$

Figure 62: Taxes, charges and incentives and the corrections used for the default values

Country	Ownership or circulation tax (€/year, including VAT)	Purchase or registration tax (€, including VAT)	Distance based road charges (€/km), motorways	Time based road charges (€/year), motorways	BEV - Registration tax multiplier	FCEV - Registration tax multiplier	BEV - Registration tax subtractor	FCEV - Registration tax subtractor	BEV - Ownership tax multiplier	FCEV - Ownership tax multiplier	BEV - Ownership tax subtractor	FCEV - Ownership tax subtractor	BEV - Purchase subsidies multiplier	FCEV - Purchase subsidies multiplier	BEV - Purchase subsidies adder	FCEV - Purchase subsidies adder
Austria	1094	0	0.381	0	1	1	0	0	0	0	0	0	0	0	22000	22000
Belgium	628	0	0.128	0	1	1	0	0	1	1	0	0	0.35	0.225	0	0
Bulgaria	942	0	0.000	685	1	1	0	0	0	1	0	0	0	0	0	0
Croatia	0	0	0.251	0	0	1	0	0	1	1	0	0	0	0	9333	9333
Cyprus	357	0	0.000	0	0	0	0	0	1	1	0	0	0	0	20000	0
Czech Republic	848	0	0.152	0	0	0	0	0	0	1	0	0	0	0	0	0
Denmark	364	0	0.000	1250	0.4	0.4	22481	22481	1	1	5067	5067	0	0	0	0
Estonia	630	0	0.000	0	1	1	0	0	1	1	0	0	0	0	0	0
Finland	147	0	0.000	0	1	1	0	0	1	1	0	0	0	0	0	0
France	619	1202	0.260	0	0.5	1	0	0	1	1	0	0	0	0	30000	30000
Germany	662	0	0.000	0	1	1	0	0	1	1	0	0	0	0	0	0
Greece	1320	6455	0.215	0	0	1	0	0	1	1	0	0	0	0	0	0
Hungary	1455	0	0.290	0	0	1	0	0	1	1	0	0	0	0	0	0
Ireland	1107	246	0.191	0	1	1	0	0	1	1	780	0	0	0	0	0
Italy	674	1654	0.180	0	1	1	0	0	0	1	0	0	0	0	0	0
Latvia	363	0	0.000	0	1	1	0	0	1	1	0	0	0	0	0	0
Lithuania	654	0	0.000	0	1	1	0	0	1	1	50000	5000	0	0	0	0
Luxembourg	140	0	0.000	1250	1	1	0	0	1	1	0	0	0	0	3000	3000
Malta	451	0	0.000	0	1	1	0	0	1	1	0	0	0	0	0	0
Netherlands	823	0	0.000	1250	1	1	0	0	0	0	0	0	0	0	0	0
Norway	889	0	0.000	0	0	1	0	0	1	1	840	0	0	0	0	0
Poland	711	24180	0.060	0	1	1	0	0	1	1	4799	4799	0	0	0	0
Portugal	657	0	0.166	0	1	1	0	0	0	1	0	0	0	0	0	0
Romania	227	0	0.000	0	1	1	0	0	0	1	0	0	0	0	4450	0
Slovakia	2172	33	0.181	0	1	1	0	0	0	1	0	0	0	0	0	0
Slovenia	1019	0	0.210	0	1	1	0	0	1	1	0	0	0	0	0	0
Spain	179	0	0.147	0	1	1	0	0	1	1	0	0	0	0	15000	15000
Sweden	32	0	0.000	1250	1	1	0	0	1	1	0	0	0	0	0	0
Switzerland	2707	0	0.837	0	1	1	0	0	1	1	0	0	0	0	0	0
United Kingdor	1977	0	0.000	616	1	1	45711	45711	1	1	0	0	0	0	0	0

Assumptions for simplifications:

- 1 DKK = 0,1342 €; 1 GPB = 1,143 €; 1 PLN = 0,2133 €; 1 SEK = 0,0888 €
- All blank cells from Schroten et al. (2019) (left block) set to zero.
- Green cells have values taken from EAFO.
- Yellow cells have price differences when benefit only applies to BEV.
- Red cells overwrite other data with figures from Noll et al. (2022) as this source is considered more reliable.
- Latvia and Lithuania first registration benefit omitted.
- Slovenia's small tax rate omitted.
- Omitted the "special tax" in Spain.
- Exemption of ownership tax in Germany until 2025 omitted.
- Ownership tax in Italy considered of 100% always.
- Ownership tax exemption in Lithuania (of 10.000 €) for N2 vehicles, considered of 5.000€.
- Ownership tax exemption of Poland considered of 10% of maximum.
- Purchase subsidies in Spain considered of 22.000 €.
- In Belgium, purchase subsidies considered in Flanders and with no limit.
- Purchase subsidies in Finland omitted, as they are valid only until 2025.
- Minimum purchase subsidies considered in Spain (15.000 €).
- VAT of United Kingdom for vehicle ownership or circulation tax and purchase or registration tax changed from 2000% to 20%.