TradeRES
New Markets Design & Models for
100% Renewable Power Systems

D4.3.2 – Principles and usage of a multi-simulation electricity market tool (D4.7)

Deliverable number: *D4.7 (D4.3.2)*
Work Package: *WP4*
Lead Beneficiary: *TUD*

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 864276

Author(s) information (alphabetical)

Name	Organisation	Email
Milos Cvetkovic	TUD	m.cvetkovic@tudelft.nl
Juha Kiviluoma	VTT	juha.kiviluoma@vtt.fi
Erkka Rinne	VTT	Erkka.Rinne@vtt.fi
Ingrid Sanchez Jimenez	TUD	I.J.SanchezJimenez@tudelft.nl
Christoph Schimeczek	DLR	Christoph.Schimeczek@dlr.de

Acknowledgements/Contributions

Name	Organisation	Email

Document information

Version	Date	Dissemination Level	Description
1.0	3 .12.2020	Public	The design principles and features of the newly developed multi-model simulation tool will be covered in this report. The report will be updated biannually to cover recent developments.

	Prepared by	Reviewed by	Approved by
	Miloš Cvetković Juha Kiviluoma Erkka Rinne Ingrid Sánchez Jiménez Christoph Schimeczek	André Lust Germán Morales-España	Ana Estanqueiro

Disclaimer

The views expressed in this document are the sole responsibility of the authors and do not necessarily reflect the views or position of the European Commission or the Innovation and Network Executive Agency. Neither the authors nor the TradeRES consortium are responsible for the use which might be made of the information contained in here.

Executive Summary

This deliverable, as a part of Task 4.3, reports on the principles and usage of a multi-simulation electricity market tool. This deliverable is updated biannually to cover the most recent developments.

The first version of this report focuses on the functional requirements of the tool. The functional requirements include model requirements, software and software engineering requirements, and requirement on result assessment. The model requirements refer to requirements for model linking, which include consolidation of temporal, spatial, vertical (system) scales of various power system models. The software requirements include requirements for workflow development, computational performance, model transparency, etc. The software engineering requirements focus on versioning and reliability of the tool. Finally, result assessment requirements focus on interpretation of results.

In the last part of this deliverable, we review the existing software solutions that could potentially serve as the basis for the model linkage toolbox. The deliverable ends with the proposed architecture of the electricity market tool.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

RES	Renewable Energy Systems
OM	Optimization Models
ABM	Agent Based Models
LP	Liner Programming
MILP	Mixed Integer Linear Programming
OEO	Open Energy Ontology
RCE	Remote Component Environment
HLA	High Level Architecture
FOM	Federate Object Model

1. Introduction – Use cases for the tool

A market design for an electricity system with a (nearly) 100% renewable energy sources (RES) needs to provide efficient incentives to its market participants in the operational and investment stages. In addition, this market design needs to provide security of supply to the entire society by guaranteeing sufficient affordable controllable capacity. Finally, market risks must be allocated in a socially acceptable way among all stakeholders.

In this deliverable, we translate the aforementioned objectives of the market design into functional requirements for an electricity market tool that would be used to study such electricity markets. Since a number of electricity market models already exists (among the project participants and outside of the consortium), we particularly look at the means of their linkage into (a) master-model(s) using a model-linkage toolbox. The main premise of this approach is to reuse what is already available to us (in terms of the market models and model linkage tools), instead of developing new models and tools from scratch. This approach consumes less resources and relies on already validated models, ensuring their longevity even further. The approach also allows for integration of new models developed within the TradeRES project. The existing models used in the TradeRES project, to be linked within the model linkage toolbox, are reported in D4.6 (D4.3.1), while this deliverable focuses on the requirements with respect to the model-linkage toolbox.

For the sake of clarity, it must be said that the multi-simulation electricity market tool from the title of this deliverable refers to the existing and new models developed in this project and linked together with the model-linkage toolbox. From now on, the electricity market tool (or only tool) will be used to denote this assembly. The term model-linkage toolbox will refer, more narrowly, to the software solution for linking the models together.

We start this report by listing possible use cases of the tool. These use cases are then employed to conceptualize the functional requirements of the tool. The functional requirements are given in terms of modelling requirements and software requirements. Next, nonfunctional requirements and interpretation of results are discussed. Finally, the possible architectural choices for the model linkage-toolbox are discussed and the architecture of the tool is proposed.

1.1 Market design questions (under 100% RES)

One set of the use cases for the electricity market tool comes from the main research questions of the TradeRES project about the market design, such as, for example, questions related to investment recovery, security of supply, market risk allocation, robustness of market design, etc. To answer these questions, several different models are needed, starting with the investment recovery model.

The investment recovery model answers the question of how market players would invest, if they obliged to certain rules of the market, incentives and forecasts of future scenarios. The purpose of the model is to capture the risks and opportunities that market players may experience.

The factors that fundamentally influence the capacity investments, such as short-term and long-term electricity prices, capacity remuneration mechanisms, RES support and the CO_2 market, have to be included in this model in order to accurately capture the impact of these factors on the investment decisions. Network congestion, operational schedules, sector coupling interdependencies, and operation of storage technologies could also have an impact on the investment decisions. Hence, an accurate operational dispatch model is needed to supplement the investment recovery model.

The operational dispatch model provides information on operating schedules and variable operating costs of market participants. It further gives information on the flexibility needs and costs of the electricity system, including factors such as congestion, re-dispatch, reserve activation, etc. The purpose of the model is two-fold. First, to supplement the investment model with a highly accurate representation of electricity system operations. Second, to validate short-term market design choices.

By combining the two models, a number of market design questions can be addressed. Let's take, for example, the investment recovery question. Will the market parties be able to recover their investment in new generation units? To answer this question, we must consider that the variable operating costs of controllable power plants to a large extent impact their payback period. These costs rise with the increase in RES penetration as the controllable power plants are asked to follow RES variability. A number of investment recovery models simplifies operational intricacies in order to cope with the computational complexity of managing both, investment and dispatch decisions within one model. The most common are assumptions on perfect competition and economic rationality of market participants. Such simplifications could result in inaccurately estimated payback periods. Hence, it is essential to rely on less simplified operational dispatch models in order to increase the fidelity of the investment recovery model. One possible way for achieving this is by model coupling using model-linkage toolbox.

1.2 Comparative analysis of models based on the same case study

In this use case, the optimization-based and simulation-based modelling methodologies are compared to answer questions of market design, as each of these methodologies has its advantages and disadvantages.

In particular, there are two basic approaches for modelling of investments, through optimization and through simulation. While the simulation provides an exploratory solution for a given set of parameters as well as investment and operational policies, the optimization provides an optimal system design by definition. Although there are optimization models which feature aspects such as risk-aversion, the optimal system design is typically obtained under the assumptions of full rationality of market players under perfect competition, their perfect knowledge and foresight, and without taking into account their risk aversion. These assumptions are the main limitations of the optimization models and can be overcome with the simulation models, particularly agent-based. The main limitation of the simulation models is that they do not provide any insight into the optimality of the solution, and hence, cannot be used on their own to judge about the optimality of the market design. Therefore, a comparative analysis of the two types of modelling approaches is necessary in order to quantify how close market efficiency is to its theoretical maximum and how close the social cost is to its theoretical minimum.

In order to be able to compare the results from a simulation model with the results from an optimization model, it is necessary that they share the same input parameters. Hence, a case study will be chosen so that both methodologies can be applied to it and corresponding results compared numerically.

In addition, different modelling frameworks or tools could use different terminology or different kinds of input parameters. For example, ‘power plants’ and ‘generating units’ or ‘grid nodes’ and ‘buses’ might mean the same thing in two energy system models or one model uses thermal efficiency (usually a number between zero and one) and another uses heat rate (usually greater than one) for fuel consumption calculations. For this, data from one model (a representation of the real system) to another needs to be translated or otherwise converted. In some cases, it might be beneficial to store the data in a chosen general format and convert from that to all compared models and then back to the general format for comparison. The translation and conversion processes might not be 100% accurate due to different methodologies used in different models.

Another layer of complexity is brought by the different execution and data input methods of the compared modelling approaches. In addition to the data content conversion, the data might need to be written and read in different formats. Different tools might use different programming languages and the execution, passing of data and command line arguments as well as reading output and results of each need to be handled. The ability to execute the compared models in parallel would be beneficial.

2. Requirements

The following subsections specify detailed requirements for the setup of a model-linkage toolbox. These requirements are made considering the existing models of the consortium partners as well as the coupling software itself. These identified requirements are to be considered in the design of the toolbox architecture.

2.1 Model requirements

In the following paragraphs the requirements for the model-linkage toolbox are presented. These requirements are derived looking at the capabilities, assumptions and limitations of the existing TradeRES models, as well as the intended case studies from WP5.

2.1.1. Model categories

There are two different model types to be coupled within the TradeRES project: Optimization models (OM) and agent-based models (ABM). Those two model types often assume different perspectives and thus allow complementary analysis of energy systems. Furthermore, models in energy systems analysis can be differentiated with respect to their decision-aspects, covering investment and / or dispatch decisions.

These aspects are explained below, highlighting the associated requirements to bridge the gaps between these model perspectives within a model-linkage toolbox.

2.1.1.1. Optimization and agent-based models

Energy system optimization models utilising linear programming (LP) and mixed-integer linear programming (MILP) are the work horse of energy systems analysis globally. These kinds of models have been used for a long time and every time they can handle more and more complex problems, thanks to the constant increase in available computational power and improvements in solving algorithms. They often cover energy generation, demand, and transportation for multiple energy carriers and sectors and across multiple countries. Typically, these models try to minimise the total system cost for the setup and operation of an energy system utilising restrictions formulated as a set of equations (e.g., linear (in)equalities). The solving algorithms then find the best (i.e., cost-minimal) solution. With such an optimal solution all elements controlled by the optimisation model are assumed to behave optimally as well. Thus, the perspective of optimization models resembles a social planner with detailed insights assuming perfect competition on their represented markets. The models Backbone and COMPETES are representatives of this model type in TradeRES.

Agent-based models have been used in the field of energy systems analysis for well over a decade. The models AMIRIS, EMLab-Generation, MASCEM and RESTrade, are long-term projects, actively developed and enhanced, representing the ABM category of models in TradeRES. Also, these models cover a lot of aspects of the energy system, albeit not as comprehensive as their OM cousins. In contrast to OM, ABM can reflect decisions of market actors and to deviate from the assumptions of perfect competition, especially with respect to the bounded rationality of market actors and imperfect market information. Thus, ABM can more realistically assess the impact of imperfect markets, actor strategies and their interplay with market designs and regulations.

Due to the wider technological and spatial scales of the OMs in TradeRES, it is expected that the ABMs will need to integrate information from the OMs within their calculations (e.g. information neighbouring countries not directly modelled in the ABMs). It must be considered that this information (e.g. dispatch / investment of power plants) has been obtained under the assumptions of the social planner – and that discrepancies between the results of the two modelling perspectives are inevitable to arise. The model coupling process in TradeRES will be required to identify these model discrepancies and, if necessary and possible, to make up for them by applying model-driven adjustments to the exchanged data.

2.1.1.2. Investment and dispatch models

Energy system models typically cover two different types of decisions: investment (e.g., power plants or grids) and dispatch (e.g., power plants, flexible demand, or storage systems). These types of decisions are connected to different temporal scales: investments are long-term decisions, whereas dispatch-decisions require a much shorter foresight. Thus, these two decision aspects are often regarded separately within the models or may even be addressed individually by specific models. Another approach is to model investment and dispatch decisions jointly, however, this approach typically uses strong simplifications and assumptions regarding the future dispatch decisions (e.g., extrapolating the utility of power plant dispatch decisions into the future). The TradeRES model federation covers both investment and dispatch decisions. Some models are capable of addressing both aspects (Backbone, COMPETES and EMLab), whereas the other models focus on dispatch decisions only (AMIRIS, MASCEM, RESTrade).

Regardless of the modelling capabilities, investment decisions are coupled with the expected future dispatch decisions. Thus, these decisions and related market designs need to be assessed in an integral way as well. Therefore, the model toolbox will need to allow detailed data exchange between the models, where each model may focus on dispatch or investment decisions. Furthermore, the toolbox will require to support loops (i.e., repeated data exchange) within its workflows in order to reflect the bidirectional dependency of these two decision types.

2.1.2. Temporal scale

When it comes to temporal representation of the TradeRES studies, it is important to differentiate the time resolution of the models and the time horizon of the studies. The intention to study electricity markets of the systems with (near) 100% RES penetration, necessitates the time horizon of 20 to 40 years into the future in order to understand the investment behaviour of market participants and the regulatory framework to reach such objective. Since the TradeRES objective is to evaluate the efficiency of the market design in such a system, and not the timeline of its development, the exact pinpointing of the year when the system achieves 100% RES penetration is irrelevant to studies, as long as the correct market conditions are modelled.

To study the questions of system adequacy and investment recovery, it is necessary to accurately model the operational time scale and accurately represent running costs of power plants. Hence, the time resolution needed for the studies has to reflect the products traded in the European energy markets and capture hourly dispatch decisions of power plants, ideally including their participation in providing flexibility services and their participation in reserve and balancing markets. The 15-minute time resolution might be relevant for studies including stronger reliability considerations such as congestion alleviation and reserve provision. The usefulness of such studies would, however, highly depend on the availability of data, while the computational complexity would be increased, and hence, this resolution is considered as desirable but not as necessary. In the case of short-term market design questions, the 15-minute time resolution will be given higher importance.

Since some studies require coupling of investment decisions (happening yearly) with dispatch decisions (happening hourly or with a higher frequency), it is necessary to consolidate information exchanged between these models using representative time intervals. The model linking toolbox should support such data exchange, while the semantic interpretation of the data and its preparation for exchange are assumed as the responsibility of the modeller.

2.1.3. Spatial (horizontal) scale

The scope of different case studies in WP5 covers local energy systems, national energy systems and a Pan-EU case study. Each of the studies focuses on different aspects of market design. For example, the Pan-EU case study finds as important to look at implications of the finite cross-border capacity on market efficiency and to investigate implications of different technology shares in different regions of EU. The national case studies look at the questions of market design for system adequacy, etc.

Most of the TradeRES models are already focused on the local or national level, while some support EU28 representation (COMPETES and MASCEM). Some of the TradeRES models (like AMIRIS and EMLab) already account for cross-border effects and market zones. Hence, the role of model linkage toolbox with respect to representation of spatial scales, if any, remains to be further specified in the following iterations of this deliverable. Once WP5 starts, these requirements will become clearer.

2.1.4. System (vertical) scale

Most of the market design questions asked in this project belong to the wholesale and European scale, and hence, the representation of national markets or coupled market zones is central to the project. One general requirement for model linkage toolbox is to provide support to couple wider geographical scales with downstream system models. Such generic support functionality can be provided by anticipating links for exchange of power/price time series between wholesale and local energy market models. This support should also include the possibility to link the market model with the physical grid model, allowing the possibility to investigate congestion questions. Similarly to the time scale representation, the semantic interpretation of information is left on the modeller, while the model linking toolbox provides the vehicle for information exchange.

2.1.5. Sector coupling scale

The questions asked in the TradeRES project are electricity system-centred and hence, the electricity sector is seen as central to the project. Other sectors, such as heat, gas, and mobility are all expected to have higher interdependencies with electricity sector in the future and should be modelled within the project. The electricity market tool should include the electricity demand from electric heating and cooling, electric transport and demand created by power-to-gas technologies. The data has to be market-zone and time-specific to be relevant for the model. Some other technologies which might also be of interest to model are hydrogen-fired power plants, electrolyzers and hydrogen storage, electric boilers for industry, near zero energy buildings, flexible buildings and heat pumps, desalination, etc.

Although it is required to model various sector dependencies and a number of upcoming technologies, at the moment, these requirements are seen as the modelling requirements and not as requirements imposed on the model-linkage toolbox. In other words, they are either available within the existing suite of models, or they are to be implemented as new models or as enhancements of the existing models. At this point, no specific requirements for model-linkage toolbox are derived based on sector coupling scale. This decision will be re-evaluated in the future.

2.1.6. Control and coordination scale

Investment decisions and energy-market trading decisions form the basis for modelling. These decisions are already encapsulated in the TradeRES investment and dispatch models. In addition to this, the dispatch models capture other operational decisions, such as reserve scheduling, etc. If possible, local energy trading and self-consumption can be considered at a local level. The activation of different flexibility options, such as demand response or storage technologies, will have an impact on the system security and overall requirement for controllable generation, and hence has to be taken into account with the energy market tool.

Since the dispatch and investment decisions will form the essential part of the model, their connection is anticipated with the model linking toolbox. The other aspects of control and coordination will be either embedded into various models, or will be synthesized through vertical integration of models (see Section 2.1.4).

2.1.7. Uncertainty (and rare-case) inclusion

Two types of uncertainties are perceived thus far: technical uncertainties and strategic uncertainties. Technical uncertainty can be divided into three categories:

i. Short-term uncertainty in the range of hours to days. Main source of uncertainty are unpredicted weather phenomena, which can be mitigated using weather forecasts.
ii. Medium-term uncertainty in the time of weeks to months. Main source of uncertainty are unpredicted weather phenomena or commodity prices, which can be mitigated using climatological and other statistics.
iii. Long-term uncertainty in the range of years. Main source of uncertainty (in addition to medium-term phenomena) are the underlying economic situation, technological development and interactions between other sectors. The number of potential combinations in this range is large, and in practice the uncertainty can be covered using a subset of coherent scenarios including all interdependent parameters.

Strategic choices in the modelling process itself create another dimension of complexity. This includes, for example, the parameterization of the actors in ABMs. As some parameters cannot be known exactly, there is also inherent uncertainty in this dimension. For some parameters, multiple values could be tested with multiple runs of the same model and results compared on statistical level.

Modelling of the rare events is important to stress test the market design. Some events which come to mind are dunkelflaute spells of multiple days with no energy coming from RES, simultaneous outages of several large power plants which can have a massive effect on short term prices especially in high demand situations, etc. These events will be considered on case by case basis, per requirements of the case studies. The tool should allow easy configuration of such scenarios.

The modelling tool thus needs to be able to handle stochastic time series in the short- and medium-term to enable stochastic modelling of the phenomena in these ranges. For the long-term, support for alternative scenarios in the data is required. The tool should support multiple runs of the same model using different strategic parameters to capture their impact. These choices should also be easily combined with techno-economic data for which there is data available.

2.2 Software requirements

The following sections describe the requirements of the model-linkage toolbox with respect to software capabilities.

2.2.1. Workflow

In addition to coupling different models together, the model linkage toolbox should support building comprehensive workflows form original data sources to modelling results. This would benefit from expressing the whole chain of tasks needed to achieve meaningful results as well as the sharing of such workflows.

A complete workflow from data to results usually contains tasks from the following list:

- get input data from a database, a set of files or through an API
- transform data to meet the format and practises of the current modelling tools
- execute modelling tools or other programs
- fetch results of models
- transform model results to match common data format
- write results to a (shared) database
- combine data and plot results

If the tasks are formulated as independent operations with well-defined inputs and outputs, they can be combined to create a workflow. Parallel tasks (e.g. comparing different models or scenarios) create branches of the workflow. The toolbox should allow parallel execution of independent tasks for efficient operation.

2.2.1.1. Exploiting complementarity of the models

Exploiting the complementarity of modelling tools forms a sequential workflow where results of the predecessor models are fed to their successors (unidirectional information flow). An example is given by splitting investment modelling and operational simulation into two different models (see Section 2.1.1.2) to form a chain of models. In this mode, all the information required by a modelling tool is given by its predecessors in the workflow.

In some cases, bi-directional information exchange and/or iteration between two or more models is required for finding a solution which satisfies some criteria. An example here is co-simulation of investments and dispatch operations using two separate modelling tools which communicate to each other during execution. A set of termination criteria has to be set to end the loop of iterations.

2.2.1.2. Comparative analysis of different models in the same case study

Figure 1 shows an example of a workflow where two models are compared. In this process, some data is first imported from the original source to a ‘master dataset’. From there the data needs to be transformed and given to both of the two models (formulated in Julia and GAMS in this example). In the lower branch there are more tasks related to writing and reading files, whereas in the upper branch direct reading from and writing to a database is utilized. Finally, the results are compared. In the TradeRES project, optimisation and agent-based models will be compared using this approach.

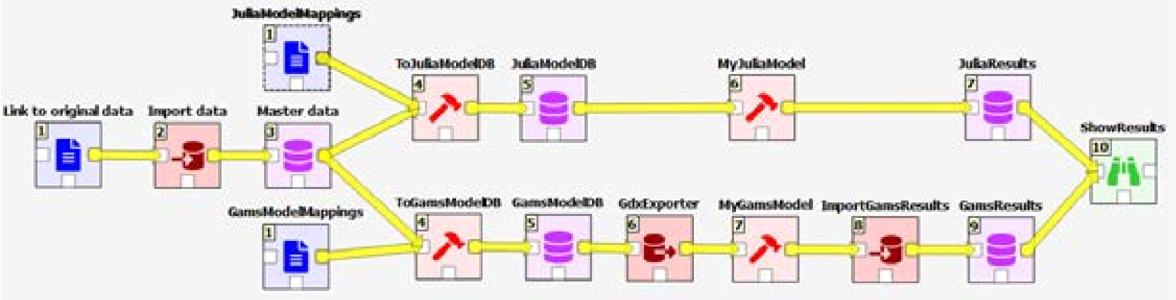


Figure 1. A workflow with a master dataset serving two different modelling tools. Items with blue document icons are inked to source data files on disk or elsewhere, red arrows to/from cylinders are data import/export functions, items with purple cylinders are data stores for managing and storing data, red hammers execute external processes and the green binocular item is a view combining one or more data stores. Yellow arrows connecting the items indicate data flow direction in the process.

2.2.2. Computational performance

The computational effort to run the tool should be negligible in comparison to the computational effort necessary to run the individual models and scripts coordinated by the tool. In addition, there should be only a minimal time delay in between two active processes of a workflow. In this way it can be ensured that the available computation time is utilised best. If possible, parallelized computations should be utilized to increase the computational performance of the toolchain, i.e. models, data transformation and workflow management.

It is expected that the performance conditions can be met with ease by the workflow management tool. First, the tool itself will only be responsible for coordinating the workflow (see Section 2.2.1), i.e. the corresponding calls to data transformation scripts and models. Second, some of the TradeRES models require a significant computational effort (e.g. optimisation models with runtimes typically well over an hour per model execution). To facilitate the use of the workflow tool on server machines, a headless mode without graphical interface is required.

Also, data transformation scripts are expected to require significant computational resources compared to workflow management. To minimize the computational effort for data transformation an API should be provided to read and write data from and to a central data storage system. The transfer tools can then utilise this API to directly connect the models to the central data storage without having to copy data to files in an intermediary step.

The scalability of the tools and whether or how processes within the workflow or its models can be executed in parallel should be analysed only after a model workflow has been established and possible bottlenecks with regard to computational performance have been identified.

2.2.3. Numerical stability and accuracy

The successful coupling of models requires a solid understanding of the numeric accuracy of these models. For optimisation models, numeric accuracy can be controlled via solver parametrisation. Solving an optimisation with higher numeric accuracy increases the required computational effort. Due to the complexity of the optimisation models used, a trade-off between numerical accuracy and computational effort is inevitable. The TradeRES optimisation models should thus use a common accuracy setting and also make these settings publicly available to facilitate the understanding of the model outcomes.

Numeric accuracy typically cannot be controlled directly in simulation models. Instead, it depends on the utilised data types and operations. While with a standard 64-bit program architecture about 15 significant digits are theoretically possible, floating-point arithmetic can cause a loss of significance. Unless a deep analysis of the employed program code has been made numerical accuracy is unknown for these models. Due to the high effort of such an analysis, they are typically not conducted. Instead, sufficient numeric stability of the simulation models is assumed.

Besides the numeric accuracy within the models, the precision of the input and output data needs to be considered and harmonized. For this, it has to be ensured that data transformation scripts do not cause any loss of accuracy. This cannot be enforced by the workflow tool itself, but relies on testing of the data transformation scripts. Rounding operations (i.e. deliberate loss of numeric accuracy) should not be used on input or output data and should be used cautiously in the logic of both scripts and models. It should be ensured that the workflow tool does not cause loss of numeric precision on top of that caused by models and scripts.

2.2.4. Time synchronization and data exchange

Several functionalities are needed in order to ensure model linkage. First, it is necessary to formally define the interfaces for model linkage. The interfaces establish a common language between models of different categories (for example, a dispatch model and an investment model) and enable interoperability between the models of the same category (for example, two different kinds of dispatch models). The interface definition should include the characteristics such as names of the variables, the units in which their values are given, simulation time stamp or time frame for which they are valid, etc.

Besides the specification of the interface variables, it is also necessary to specify metadata of each model, including features such as time step of the model, time horizon of the model run, technologies and sectors covered, etc.

It is commonly understood as the responsibility of the modeler to specify the interfaces and metadata in accordance with the tool guidelines. The electricity market tool developers will provide guidelines for specification of both and the technical means to do so, but will not enforce the form of either. This way, a sufficient degree of freedom is given to the modeler to adjust the linkage according to their needs.

The TradeRES team will look to propose a new or align with existing ontologies for sharing model information (such as, for example, Open Energy Ontology[1]).

---

[1] https://openenergy-platform.org/ontology/

The individual models can technically be connected in many ways. They could be sharing as much as shared memory space on the same workstation, or as little as the internet connection across the globe. However, since all TradeRES models come as open access, minimal requirements on the architecture of the computational cluster are imposed. These requirements relate to the inclusion of open access-black box models packaged as executables or dynamically linked libraries.

In terms of data exchange, it is important to consider data volume and data throughput. On one side, coupled simulation models often have the requirement of high throughput and low volume of exchanged data (think of co-simulation of automotive and aerospace models). On the other side, coupled optimization models might rather have the requirement of low throughput and high volume of exchanged data. Since the energy models and investment models are typically data-heavy, we are considering the low throughput and high volume as the dominant requirement on data exchange.

Finally, time synchronization is another aspect which is often considered when it comes to model coupling. In the simulation community, global time keeping and time synchronization functionality is often externalized from the models and assigned to the master algorithm. Since the co-simulation master handles the synchronization between models, such architecture has high extendibility as every additional model can easily be added to the existing assembly of coupled models. Although in the case of electricity market tool, such functionality could be useful to couple simulation models together, we do not perceive the immediate need for coupling a large number of simulation models and will, hence, keep this option in mind if the need becomes apparent.

2.2.5. Transparency requirements

TradeRES aims to maximise transparency and traceability of its findings. Therefore, the modelling workflows are required to be openly publishable. This will enable reviewers to understand how data flows between the models are organised and which data was exchanged for a specific assessment of market design.

Ideally, the tool to be developed, all its comprising models and the framework toolbox are also openly available. However, this cannot be guaranteed as of now, since not all models within the project or all of its components are published open source (e.g. AMIRIS, RESTrade, COMPETES). Thus, the minimum requirement for the project to succeed is open access to the (compiled) models for all project partners and expert stakeholders associated with the project. The performance assessment of market designs in WP5 will require all aforementioned parties to be able to execute all published workflows and associated models. Each party must be enabled to create its own input data and to perform its own assessments of market designs.

2.2.6. Data management (logging, debugging)

Due to the complexity of model linking, data logging and tool debugging are highly relevant functionalities of the electricity market tool. The models used in the TradeRES project generate a significant amount of data, which must be stored and linked to appropriate case studies and appropriate case scenarios. It is also necessary to trace the versions of models which produce saved results. Finally, it is necessary to keep track of data propagated among models and also to keep this data aligned with the internal state and internal parameters of each of the models.

In summary, the electricity market tool must support deposition of large volumes of data, which is aligned (in terms of the case study, case scenario, input and output data, model parameters, exchanged data among models, and tool versions).

In addition to data relevant for case studies, the electricity market tool must also allow accessible logging of the workflow, model metadata, and error information for easier debugging of the tool.

2.3 Result assessment requirements

The toolbox is required to facilitate the assessment of the model outcomes and of the market designs under investigation.

2.3.1. Model result assessment

In order to assess the model outcomes, their understanding is a fundamental requirement. The model results must thus have an appropriate metadata attached to enable their interpretation. The metadata must be based on a common ontology. In addition, the metadata of the model outputs must ensure that model inputs and outputs are connected, supporting traceability and transparency of the model outcomes. This is especially important for managing multiple different workflows and models at the same time. The data transformation scripts for each model must ensure that the transformed results follow their metadata descriptions at all times.

The above-described rigorous documentation of the results will be necessary in order to compare the results from different models, from different versions of the same model, or from the same model with different input data.

2.3.2. Market design assessment

The assessment of possible future market design options is a central part of TradeRES. Requirements additional to those in the previous paragraph apply for this task: The toolbox workflows need to be easily adjustable with respect to all model input parameters, especially those inputs that vary in the scenarios. There must be a central database to store results for all models of a workflow on demand (e.g. if a specific workflow was completed successfully). These data build the foundation for the web-based decision tool to be developed in Subtask 7.3.1. However, it can be assumed that not all model outcomes are of value: There might be incomplete result sets as a consequence of modelling errors, computational problems incomplete workflows, erroneous scripts etc. Therefore, result files should be stored locally first, and uploaded to the central results database only on demand.

2.4 Software engineering requirements

The models within the TradeRES toolbox are subject to constant improvement. To keep the toolbox operable despite the constant change of its models, principles of sustainable software engineering must be employed. With these principles long-term use of these tools can be achieved. Thus, the following requirements do not only apply to the toolbox itself, but also to its encompassing models.

There are many aspects of sustainable software engineering not touched by the following paragraphs (e.g., software security). These aspects are not deemed unimportant. Instead, the following paragraphs highlight the most pressing aspects to be considered for the setup and development of the TradeRES model toolbox.

2.4.1. Interface reliability

The term “interface” here relates to the structure of the input and output data of the TradeRES models to be coupled in the toolbox. This interface needs to be compatible with the central data storage system of the toolbox at all times. Therefore, updates of the toolbox’s models also need to encompass appropriate updates of their connected data transfer tools in order to not break the established workflows. Additionally, to be able to compare results from different versions of the employed models, model results should include meta-data (e.g., units and descriptions) based upon a common modelling ontology.

2.4.2. Tool extension and versioning

The models to be coupled in the toolbox are subject to extensions and enhancements within this project. The models shall be improved in order to represent temporal, sectoral and spatial flexibility options as well as to represent new actors, markets and policies of the European energy system. Therefore, the toolbox must include a versioning system or rule set for the packaged models. Here, a packaged model refers to a model executable including appropriate input and output scripts. In addition, the packaged models must include meta-data to describe the tool capabilities and changes. The toolbox must provide a central repository of the packaged models and an easy way to update the models within a workflow. The workflows should specify the version of each employed packaged model in order to be able to reproduce workflow results.

3. Architecture of the multi-simulation electricity market tool

3.1 Existing model coupling solutions

TradeRES has explored several different existing solutions for model coupling. The following subsections detail the three most promising solutions, namely RCE, Spine Toolbox and Mosaik.

3.1.1. RCE

Following, the Remote Component Environment (RCE)[2] is an Open Source distributed, workflow-driven integration environment. It is used by engineers and scientists to design and simulate complex systems (e.g., aircraft, ships, or satellites) by using and integrating their own design and simulation tools. An overview of RCE can be found here[3].

The main advantage of RCE is its capability to compile and execute workflows of distributed "black-box" tools. This means that only the interfaces to the tools need to be publicly known, whereas source code and executables of tools integrated into workflows can remain at their remote locations. Once a tool from the workflow is called to work, it is executed at its remote origin. This allows models to interact within a workflow without publishing the models –allowing the integration of closed-source and proprietary tools into work flows. Yet, everyone can use the output of these tools within their workflows.

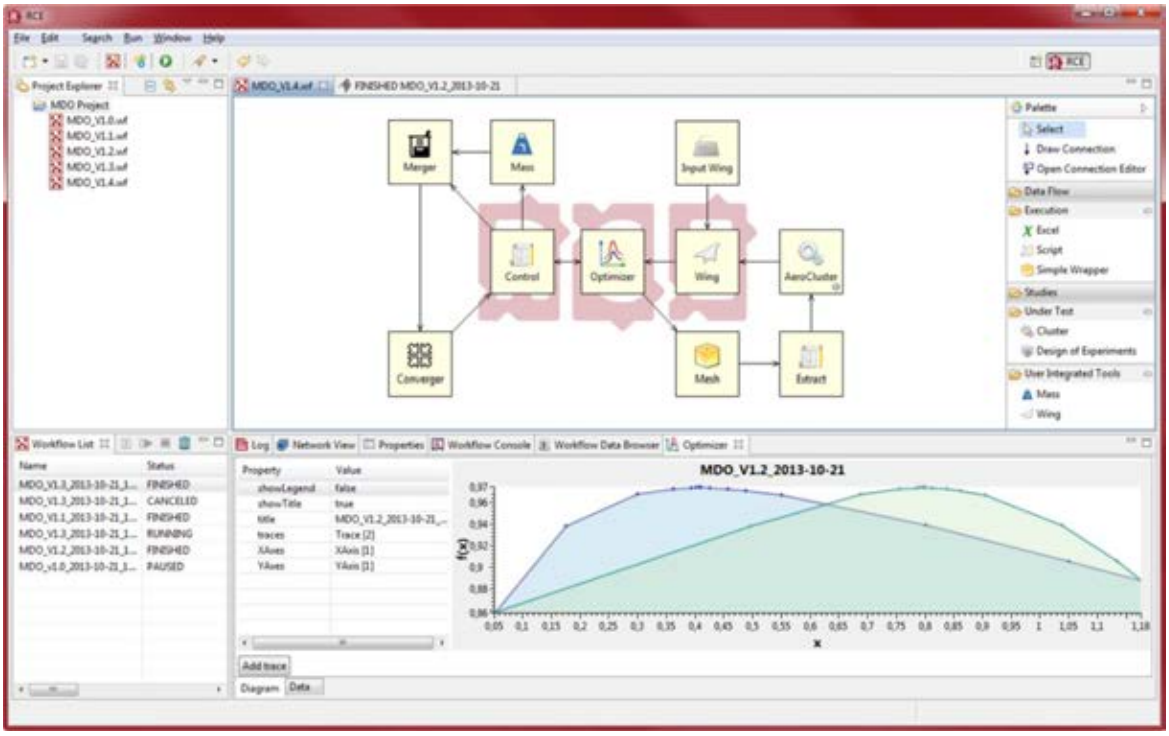
Figure 2 gives an example of RCE's powerful graphical user interface which is capable to manage multiple projects and workflows. RCE can be run in local and remote mode, allows integrating any tool that can be called from command-line, provides a large set of pre-defined workflow control tools (e.g., loop controllers & optimizers) and handles data input from various sources (e.g., from xml-files, excel sheets, or databases).

RCE has been co-developed by DLR's institute for software technology. Thus, DLR has extensive experience in using RCE for several national projects. Although RCE has been used in several international projects as well, supporting this tool is not trivial. When used in cross-party workflows, RCE requires a central managing instance which has access to all tools that are to be executed within the workflow. Unfortunately, restrictive firewall policies of tool-providing partners have proven to complicate the setup of distributed RCE workflows, often causing project delays in the past. Given DLR's own restrictive firewall rules, a successful demonstration workflow could not be created up until now.

---

[2] https://rcenvironment.de

[3] http://elib.dlr.de/77442/



Name	Status
MDO_V1.3_2013-10-21_1...	FINISHED
MDO_V1.3_2013-10-21_1...	CANCELED
MDO_V1.1_2013-10-21_1...	FINISHED
MDO_V1.3_2013-10-21_1...	RUNNING
MDO_V1.2_2013-10-21_1...	FINISHED
MDO_v1.0_2013-10-21_1...	PAUSED

Property	Value
showLegend	false
showTitle	true
title	MDO_V1.2_2013-10-21_...
traces	Trace [2]
XAxes	XAxis [1]
YAxes	YAxis [1]

Figure 2. RCE’s graphical user interface: Workbench with different views and opened workflow editor, taken from (RCE-10 User Guide)

3.1.2. Spine Toolbox

Spine Toolbox is a graphical workflow management application for computational tasks (P. Savolainen, 2020). It is problem and software agnostic, but currently has direct support for Julia, GAMS and Python tools/models. Other software languages can also interact, but more work is likely required to exchange information. Figure 3 shows an overview of the Toolbox user interface.

Working with the Toolbox is based on *projects*, each consisting of one or more workflows with various different processing items*:*

- *Data Connections* for selecting files for further processing.
- *Importers* for reading and parsing various files (csv, GAMS GDX, Excel workbooks are currently supported).
- *Tools* support executing a Python or Julia script, a GAMS program or any other executable file or script (Not to be confused with the multi-simulation electricity market tool described in this document).
- *Gimlets* are light-weight Tools for running, e.g., shell commands.
- *Exporter* items can be used to write data to various file formats (GAMS GDX currently supported while Julia and Python can use the Spine DB API directly and do not therefore need an exporter).
- *Data Stores* are used to view and manipulate data stored either locally or remotely.

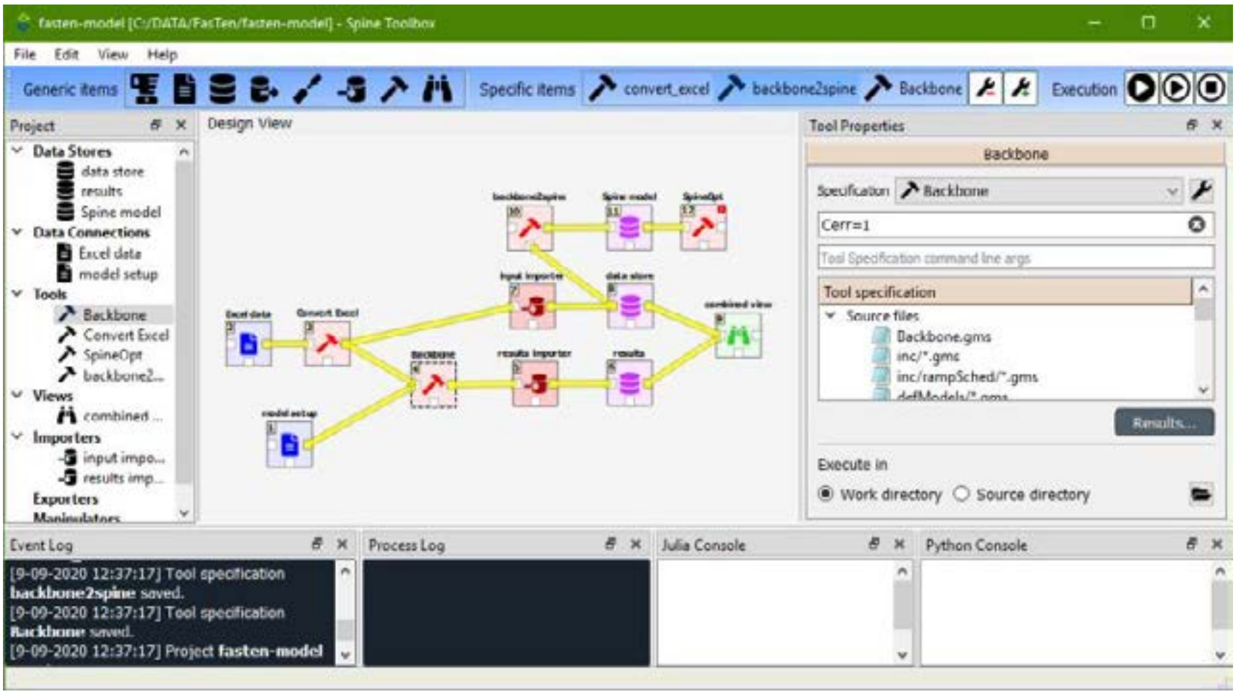
Each Tool requires a Tool specification which defines the input and output files of the process as well as the command to execute. Once a specification is created, it can be used in different projects.

A key component of the application is the generic data structure (based on entity–attribute–value with classes and relationships) which can be used to store data for many different purposes. A class defines possible attributes for the entities that are members of the class, for example class of power plants would contain individual power plant entities. Objects are one-dimensional entities while relationships are multi-dimensional entities. Relationships establish connections between object entities and consequently enable multi-dimensional sets that can be used to represent, e.g., a connection between two nodes in a power grid and hold the parameter data for the connection.

For the purposes of TradeRES, we would establish a common data nomenclature to serve all the different models from a single database and thus allow execution of modelling chains and comparison of results from multiple different models. The nomenclature could be based on existing efforts for common energy systems nomenclature.

Similar to RCE, it is possible to execute closed source executables within the workflow. However, the approach is different, since each non-sharable part would have their own local Spine Toolbox workflow for executing their part of the modelling chain and the data exchange would take place over a shared database.

Spine Toolbox is developed by the Spine project consortium (EU Horizon 2020, grant no. 774629). The Python source code is published under GNU Lesser General License (open source) here: https://github.com/Spine-project/Spine-Toolbox.



Specification: Backbone

Cerr=1

Tool Specification command line args

Backbone.gms
inc/*.gms
inc/rampSched/*.gms

Execute in
● Work directory ○ Source directory

```
[9-09-2020 12:37:17] Tool specification
backbone2spine saved.
[9-09-2020 12:37:17] Tool specification
Backbone saved.
[9-09-2020 12:37:17] Project fasten-model
```

Figure 3. Spine Toolbox user interface with a workflow

3.1.3. Mosaik

Mosaik is a simulation compositor for smart grid simulations (OFFIS e. V., n.d.). It integrates existing simulators by coupling them to simulate large-scale smart grid scenarios. Its origins trace back to the simulation of digitally enhanced and renewable rich distribution grids. Mosaik implements handles for different kinds of simulator processes on this scale (from distributed generation components to smart grid controls). It schedules the step-wise execution of the different simulators and manages the exchange of data between them.

Mosaik uses the discrete event simulation library SimPy for coordination. It contains a module, sim- manager, which enables data exchange with the simulators through TCP connections. The sim-manager is responsible for processing the simulators and their interconnections, while the scheduler tracks the dependencies between the simulators and performs simulation steps. It also allows in-process execution. Mosaik is able to start a new process, connect to running processes and import a simulator module. Its simpleAPI allows us to create simulation scenarios, to start simulators and to instantiate models from them. It uses network sockets and JSON messages to communicate with the simulators.

In contrast to RCE and Spine Toolbox, which are intended to couple stand-alone models, Mosaik is a co-simulation orchestrator which is intended for coupling of partial models. In other words, RCE and Spine Toolbox are built on the premise that each model they couple can produce a meaningful result when running independently. Mosaik, on the other hand, is built on the premise that the models it couples require each other in order to produce a meaningful result. This assumption has had several distinctive characteristics for the architecture of each tool. In this regard, the model execution in RCE and Spine Toolbox is controlled through a workflow which recognizes only the notions of inputs and outputs of models and how these are connected together, while the model execution in Mosaik is controlled by a co-simulation master, which, besides the connections of inputs and outputs of the models, keeps track of global time and synchronizes all models against it, ensuring with it that data is exchanged between models at the correct time instance. In other words, RCE and Spine Toolbox provide looser coupling among models, while Mosaik provides tighter coupling. Hence, less effort is required to adopt RCE and Spine Toolbox for model coupling when compared to Mosaik. For simple workflows, such as sequential execution of models, their capabilities are quite sufficient. On the other hand, Mosaik provides additional benefits when it comes to coupling of models into more advanced workflows (which, for example, include loops or multiple data exchanges among same models). It must be said, however, that such coupling is only possible if the models themselves are capable of supporting iterative operation and data exchange.

From the perspective of the TradeRES case studies and the questions of the new market design, RCE and Spine Toolbox are the first candidates for underlying control of model execution due to the looser coupling they provide. In addition, some of the perceived workflows in TradeRES are quite simple in nature (sequential) for which these tools would suffice.

3.1.4. Others

There are a couple of other workflow management tools that are worth mentioning, like HLA and HELICS.

High Level Architecture (HLA)

HLA is a US Department of Defense specification for co-simulation targeted to distributed simulations. Comparing to previously described tools, it is more similar to Mosaik. It consists of a runtime infrastructure which provides time synchronization and communication among the federated simulators. Each simulator is referred to as a federate, while the entire co-simulation is referred to as a federation. HLA is capable of synchronizing time stepped and discrete event simulators.

Users define federations that may involve complex interactions but are time consuming to set up. It is required that a Federation Object Model (FOM) specifies all valid interaction types and exchanged data. HLA specifies a set of ten design rules for the creation of Federations and federates. The HLA also demands the specification of Simulation Object models, which are designed for a specific domain and can be reused for new simulations in the same domain. They are subordinated to the FOM. In contrast, Mosaik requires users to define scenarios that only allow for basic data exchange.

In comparison to Mosaik, HLA federates are more autonomous and retain more control over the simulation process. They can dynamically connect or disconnect from running simulations. HLA is also more versatile in terms of the federates, especially for time synchronizations. HLA allows for negotiation of time advancement and for iterative coupling, unlike Mosaik. The work in (Steinbrink, 2018) compared Mosaik with HLA (see Figure 4 for comparative view of their architectures) and they concluded that Mosaik is more useful for entry level prototyping co-simulation and HLA is more suited for complex and extensive use cases.

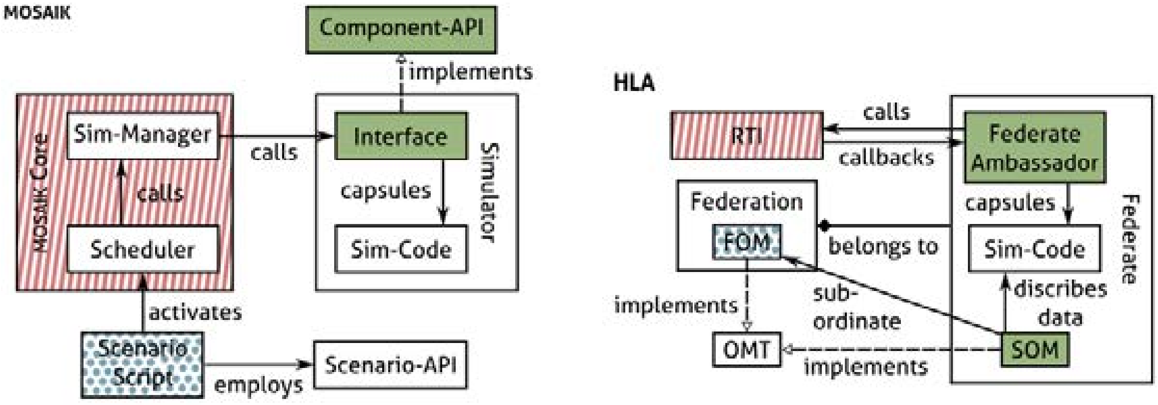


Figure 4. Architecture overviews of Mosaik and HLA (Steinbrink, 2018)

Hierarchical Engine for Large-scale Infrastructure Co-simulation (HELICS)

HELICS (Hierarchical Engine for Large-scale Infrastructure Co-simulation) is an open-source co-simulation framework designed to integrate simulators, and also designed for separate domains to simulate regional and interconnection scale power system behaviours (Palmintier, 2017). This co-simulation framework is highly scalable, supporting from 2 to more than 100,000 federates. It is open source, modular and can integrate diverse models with no need for intense interface development. It can also integrate diverse simulation types, as quasi steady state time series, time series, phasor dynamics and discrete simulation. Finally, it also supports co-iteration enabling inter federate convergence before advancing time.

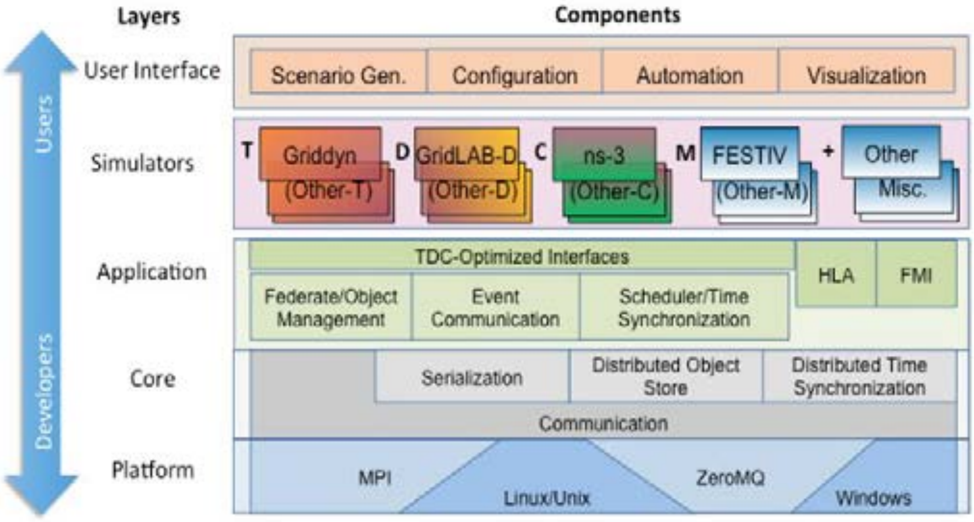


Figure 5. HELICS architecture shows key subcomponents in each layer (Palmintier, 2017)

HELICS adopted design features from HLA such as the callback based API. The federates are running instances that are assigned to specific models. A collection of federates interacting with each other is defined as a federation. The core layer, shown in Figure 5, contains constructs to parameter-based and message-based interactions from end points. The minimum features needed for co-simulation are represented in the core layer. These are the time management and data flow for discrete events and time series simulations. After initializing the core layer, federates are registered in the federation. Time management allows federates to operate with different timescales and to iterate at any time step to achieve convergence. The application API is designed to be simple and intends to make interaction of generic applications easier. The API defines three types of federates with diverse types of interaction supported by the core.

- Value federates: They interact through a publish and subscribe mechanism. Provide functions to query if a value is updated, obtain the value and note the time of the update.
- Message federates: They interact with federates simulating an ICT exchange. A message has a specific source, destination and time sent.
- Message filter federates. The message filter federate builds on a message federate to add additional ability to modify or manipulate a message itself or its timing. Message filters can be defined for sources and/or destinations.
- Programming interfaces: allow user flexibility to fit with a variety of simulators
- Interaction with existing co-simulation standards: the application layer will expose interfaces for co-simulation standards such as HLA and FMI
- Interface flexibility and local routing: application can connect through an API layer or can include an independent execution application wrapped around the core API.

The Simulation layer has two classes of simulator interfaces:

- General purpose: interface support for arbitrary user provided federates
- Optimized interfaces for common TDC+M application types

Finally, the user interface layer attempts to streamline with standardized approaches the processes of assembling the input data, organizing and running the co-simulation and parsing the results.

3.2 Proposed architecture

Based on the tool requirements from Section 2 and available tools for model linking from Section 3.1, we propose the architecture of the electricity market tool developed on the shoulders of the model-linkage toolbox.

The first design choice we make is to pick Spine Toolbox as the core model-linkage toolbox. This choice is deliberately made considering the strong workflow support and database-cantered architecture of the Spine Toolbox which allow for relatively simple integration of a number of TradeRES models. Since Mosaik is capable of handling more complex workflows (such as iterative loops), it will be considered as an alternative option to support model coupling when Spine Toolbox capabilities do not suffice.

The second design choice concerns the storing and versioning of models. First, the models will be stored in a model repository. This model repository will be deployed on a remote server. A versioning system will be put in place to make sure that the working versions of the models in the workflows are available, even if individual models and scripts are being updated. The repository will be made available to the members of the TradeRES consortium and expert stakeholders associated with the project.

The third design choice is made regarding data and workflow storing and versioning. Common input data for all TradeRES studies and the common workflows will be stored on a remote server accessible by all aforementioned parties. The output data of case studies is intentionally left out of this central repository as its volume might quickly enlarge beyond tractable.

Next, the execution of the tool is illustrated in Figure 6. Each partner will be able to make a working copy of the toolchain (including the model-linkage toolbox and collection of models for a particular workflow or a part of a workflow) at their local server or computing cluster. This local version of the toolchain is executed and the case study results are stored locally for analysis and processing. Since Spine Toolbox runs a database in the background, a local database server is necessary.

Finally, to ease model interfacing, a common TradeRES ontology will be developed. Since several large-scale efforts for developing energy model ontologies exist (openENTRANCE[4], OEO[5], etc.), TradeRES will look to align with these advancements whenever appropriate and possible.

Further architectural choices, or any amendments to this architecture, will be published in the updated version of this deliverable.

---

[4] https://openentrance.eu/

[5] https://openenergy-platform.org/ontology/

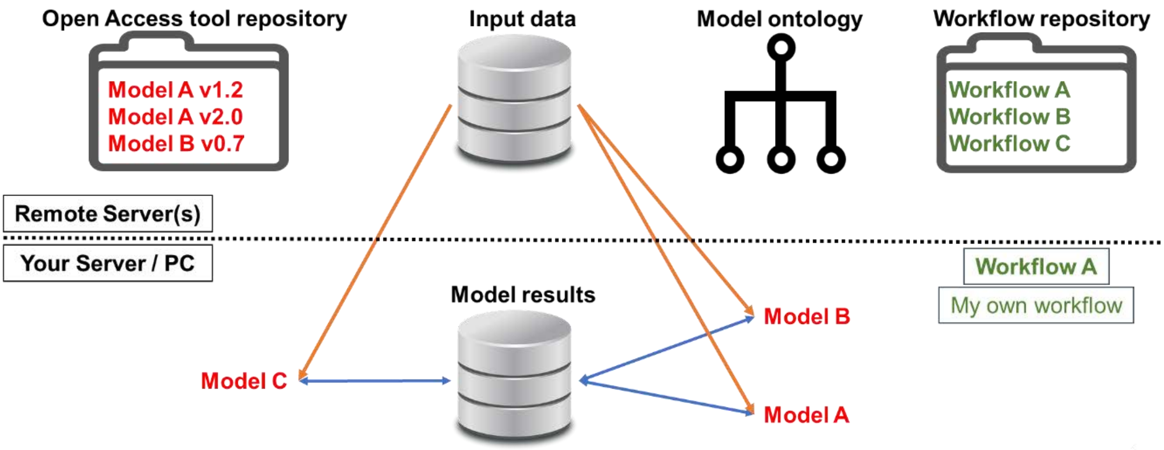


Figure 6. Deployment architecture of the Electricity market tool

Workflow examples

The TradeRES consortium is interested in several types of studies on the market design for a (near) 100% RES power system. These studies can be accomplished by implementing different workflows. Figure 7 shows some workflows of interest which are further explained in Table 1. It is important to emphasize that these workflows serve only as an example. The exact workflows which will be implemented within TradeRES project depend on the market design choices made in WP3 and on case study needs of WP5, and hence, will be defined in the upcoming period.

In these examples, optimization-based models are denoted with (A) and (C), while the simulation (agent-based) models are denoted with (B) and (D). As mentioned earlier, some workflows exploit complementarity of models to create larger coupled models, while other workflows compare results of different model categories.

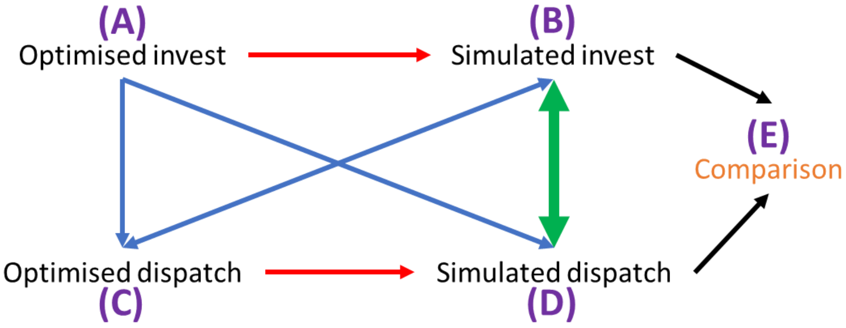


Figure 7. Graphical representation of various workflow segments

Table 1. Examples of particular workflows

Path	Explanation	Example / Market Design
A ➔ B	Optimal investment results used in investment simulation	Optimal investments from other EU countries are fed into an investment simulation model of the country under study.
C ➔ D	Optimal dispatch results used in dispatch simulation	Optimal dispatch in other EU countries is fed into a dispatch simulation model of the country under study.
B ↔ D	Data exchange between agent-based models	Simulated dispatch computes accurate variable costs which are included in the investment model at each investment stage.
A ➔ B ➔ E	Comparison of optimal & simulated investment	Test investment-related market designs (e.g. tenders)
A ➔ C ➔ D ➔ E	Comparison of optimal and simulated dispatch	Test dispatch-related market designs (e.g. premia) based on an optimal investment model
B ➔ C ➔ D ➔ E	Comparison of optimal and simulated dispatch	Test dispatch-related market designs (e.g. premia) based on a simulated investment model
A ➔ C ➔ E B ➔ D ➔ E	Comparison of optimal and simulated dispatch and investment	Test interactions of dispatch- & investment-related market designs based on simulated investment compared to fully optimized result

4. Final remarks

In summary, the first version of this deliverable focuses on the modeling and software requirements for the TradeRES electricity market tool. These are outlined in detail in order to provide sound guidelines for the development of the tool. The initial version of the software architecture is also proposed.

This deliverable will be updated biannually (in months M17, M23, M29, M35, M41). The updates will focus on further refinements of the proposed architecture. The tool requirements will be updated if the need arises.

References

OFFIS. (n.d.). *Mosaik.* Retrieved from https://mosaik.readthedocs.io/en/latest/overview.html

P. Savolainen, M. M. (2020). *Spine Toolbox's User Guide.* Retrieved from https://spine-toolbox.readthedocs.io/

Palmintier, B. K. (2017). Design of the HELICS High- Performance Transmission- Market Co-Simulation Framework Preprint . *Work-shop on Modeling and Simulation of Cyber-Physical Energy Systems.*

RCE-10 User Guide. (n.d.). Retrieved from https://rcenvironment.de/pages/documentation/documentation.html

Steinbrink, C. v.-l. (2018). Smart grid co-simulation with MOSAIK and HLA: a comparison study. *Computer Science - Research and Development 33(1-2)*, 135–143.