



**HAL**  
open science

# A generic agent-based framework for modeling business ecosystems: a case study of electric vehicle home charging

Magnus Værbak, Zheng Ma, Yves Demazeau, Bo N Jørgensen

## ► To cite this version:

Magnus Værbak, Zheng Ma, Yves Demazeau, Bo N Jørgensen. A generic agent-based framework for modeling business ecosystems: a case study of electric vehicle home charging. 1st Energy Informatics Academy Conference, EIA\_Asia'21, May 2021, Beijing, China. pp.1-26, 10.1186/s42162-021-00158-4. hal-03435666

**HAL Id: hal-03435666**

**<https://hal.science/hal-03435666>**

Submitted on 18 Nov 2021

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution 4.0 International License

METHODOLOGY

Open Access



# A generic agent-based framework for modeling business ecosystems: a case study of electric vehicle home charging

Magnus Værbak<sup>1\*</sup>, Zheng Ma<sup>2</sup>, Yves Demazeau<sup>3</sup> and Bo N. Jørgensen<sup>1</sup>

From 1st Energy Informatics.Academy Conference Asia  
Beijing, China . 29-30 May 2021

\* Correspondence: [mavar@mmmi.sdu.dk](mailto:mavar@mmmi.sdu.dk)

<sup>1</sup>Center for Energy Informatics,  
Maersk Mc-Kinney Moeller Institute,  
University of Southern Denmark,  
5000 Odense, Denmark  
Full list of author information is  
available at the end of the article

## Abstract

Modeling and simulation have been popularly used for system investigation and evaluation. With proper evaluation, distribution system operators can decide on a reasonable course of action for encouraging energy flexibility and make predictions on the recommended timing and magnitude of system updates under different scenarios. However, there is no efficient tool for system operators to quickly set up and perform simulations of alternative scenarios for system updates before planning their course of action, without much experience with programming or system modeling. This paper proposes an agent-based modeling framework for developing agent-based simulation models of business ecosystems that can be applied to multiple evaluation scenarios by simple configuration of agents and roles. There are two steps in this proposed framework: Step 1 – Interface and role interactions design and Step 2 – Agent architecture and connections design. In addition, the framework depends on a pre-step that covers mapping and architecture development of the business ecosystem to be modeled. The framework is demonstrated with a case study of an energy business ecosystem consisting of an electricity distribution grid with 137 connected domestic consumers. The case study shows that the proposed agent-based modeling framework supports the development of agent-based models for simulating energy business ecosystems. To verify the behavior of the developed agent-based simulation models, a verification procedure of the agent models is briefly discussed, which includes unit, integration, and system testing approaches similar to the ones used in software testing.

**Keywords:** Agent-based modeling, Model framework, Smart grid

## Introduction

The 2015 United Nations Paris Agreement on climate change has caused a need for an energy sector that relies more on renewable energy resources and less on conventional fossil fuels, such as oil, coal, and natural gas. Due to the increasing prevalence of distributed energy resources (DER), distribution grids will face the challenge of



© The Author(s). 2021 **Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

insufficient grid capacity (Billanes et al., 2017). For instance, although 97% of the consumed energy for transportation was based on fossil fuels in Denmark 2017 (Basisfremskrivning, 2020), it is expected that the personal vehicle pool on the Danish roads will be comprised of 380,000 Electric Vehicles (EV) in 2030 (Basisfremskrivning, 2020). This is expected to impact the electricity distribution system that supports the charging (Fatras et al., 2020), as personal transportation constituted the considerable share of 75.2% of the energy consumption within the Danish transport sector in 2019 (Data, tabeller, statistikker og kort Energistatistik, 2019).

To avoid risks and unnecessary costs, system operators must evaluate the performance of potential solutions before they are carried out in practice by modeling the system (Ma et al., 2016). This system model can be subjected to different parameter configurations and stimuli to test how it behaves and responds under these circumstances and thereby get a better understanding of the real system (Christensen et al., 2019). Meanwhile, the adoption of a certain technological or behavioral solution to a problem often affects and is affected by other actors in the system (Ma et al., 2018). These actors in a value chain must gain non-negative value from the solution to ensure the adoption (Adner, 2012). Therefore, all actors relevant to the value chain must be considered.

With proper evaluation, system operators can decide on a reasonable course of action for encouraging energy flexibility and make predictions on the recommended timing and magnitude of the low-voltage grid extension under different scenarios. However, these scenarios are often many, and there is no efficient tool for decision-makers to quickly set up and perform numerous scenario experiments before planning their course of action, without much experience with programming or system modeling. Prioritizing intelligibility as part of a model design can improve the efficiency of decision-making as well.

Agent-based modeling (ABM) (Ringler et al., 2016; Macal & North, 2010) has been used for modeling and analyzing individual actors as well as emergent patterns subjected to different scenarios, e.g. energy flexibility in greenhouses (Christensen et al., 2020), water supply (Værbak et al., 2019), and breweries (Howard et al., n.d.). Therefore, this paper aims to develop a generic ABM framework for creating simulation models of targeted distribution grids that can be employed to facilitate distribution system operators' (DSOs) investigation and evaluation of various grid scenarios. A case study of a distribution grid in Denmark is used in this paper for presenting an application of the developed ABM framework.

The agent-based simulation model framework proposed in this paper covers Part III of the business ecosystem modeling approach proposed by (Ma, 2019) (shown in Table 1). There are two steps in this proposed framework. Besides the two steps, the Part I Business ecosystem architecture development in the business ecosystem modeling (Ma et al., 2021) constitutes a pre-step that collects information on the ecosystem to be modeled before the model creation itself is conducted.

By using the proposed methodology of business ecosystem architecture development from (Ma et al., 2021), and an energy ecosystem, e.g. the Danish electricity market, the elements of relevant actors and objects along with their roles and interactions can be defined (Roles & responsibilities, 2021). In the ABM framework, these actors and objects are converted into agents that interact with each other based on the roles they

**Table 1** Business ecosystem modeling process (Ma, 2019)

Part	Stage	Business ecosystem modeling
Part I Business ecosystem architecture development	1	Identify the boundary of a selected ecosystem.
	2	Identify actors and their roles in the ecosystem.
	3	Identify actors' value propositions and business models.
	4	Identify interaction between actors (different types of interactions)
Part II Factor analysis	1	Investigate influential factors and their impact on the elements in the ecosystem (actors, roles, and interaction)
	2	Investigate potential changes in the ecosystem.
<b>Part III</b> <b>Ecosystem simulation and reconfiguration</b>	1	Multi-agent based ecosystem modeling to identify ecosystem reaction towards the potential changes.
	2	Ecosystem reconfiguration (including reconfiguration of actors, roles, and interaction) due to changes, system dynamics modeling might be applied at this stage.
	3	Business model reconfiguration.

serve (Zhu & Zhou, 2008). The model framework intends to be extendable and allows agents to represent new technologies and actors without requiring significant changes to the set-up of the existing agents in the model.

The rest of the paper is structured as follows: The Background section introduces the Danish electricity system and ABM and simulation of smart grids. The Methodology section, Generic Agent-Based Modeling Framework Development, introduces the methodology proposed in this study, and the subsequent Case Study section presents the structure of an agent-based simulation model that has been developed using this framework. The Discussion section follows up on the case study and briefly discusses a few limitations of the proposed framework and the need for verification testing of created models. Lastly, the Conclusion section summarizes the paper.

## Background

### The Danish electricity system

The Danish electricity infrastructure consists of a transmission grid (132–400 kV) owned, operated, and maintained by the state-owned transmission system operator (TSO), Energinet.dk, and numerous local distribution grids which connect electricity consumers and smaller producers at medium-voltages (60–10 kV) and low-voltages (400–230 V). These distribution grids are operated and maintained by DSOs, each of them being responsible for a specific and delimited region. Energinet.dk owns the national energy system data platform, DataHub, and, along with the TSOs of the other Nordic and Baltic countries, co-owns the electricity trading market platform, Nord Pool (Roles & responsibilities, 2021).

Following the gradual liberalization of the Danish electricity system, system operators are obligated to provide grid access and services on equal and indiscriminate terms to all market players in the energy system. Furthermore, DSOs are in charge of performing electricity consumption measurements on the consumers connected to their respective grids and send the data to the DataHub (Zheng et al., 2016). The national roll-out of smart meters by 2020 enables hourly settlement of electricity consumptions

(The Danish electricity retail market, 2021). Demand response is not fully developed in Denmark compared to Finland, France, and UK (Market models for aggregators, 2021).

Electricity suppliers act as the link between the electricity system and electricity customers. The suppliers buy electricity and sell it to customers through agreements. Based on measurements from the DataHub, the suppliers are furthermore tasked with collecting payments for all electricity services provided to their respective customers in one single bill that includes the raw electricity price, net service tariffs, duties, etc. (The electricity grid, 2021).

### **ABM for smart grids**

Many simulations and modeling tools have been used to investigate the electricity system, especially electricity markets with different configurations of mechanisms and players (Ma et al., 2019). However, according to (Santos et al., 2016), the majority of the tools lack interoperability between heterogeneous energy systems and hence the possibility for investigating cases that feature participants from different energy systems participate in the same markets.

The studies of (Oliveira et al., 2012; Praca et al., 2003) attempt to close that gap by proposing a set of electricity market ontologies and provide a common semantic platform to allow communication between agents of different systems. To achieve that objective, a hierarchy of classes adhering to agent and entity roles and communication types is established. Comparatively, (Santos et al., 2016) focuses on market processes and participation.

Using a more overall agent hierarchy classification approach, (Shafie-Khah & Catalão, 2015) proposes a framework that divides agents into two layers: the First-layer that contains agents directly participating in the energy electricity market, and the Second-layer that contains agents which participate through connections to First-layer agents. Using the Layer approach, the agents are categorized into a hierarchy, where agents from one layer aggregate exchange information with agents in the layer below, make decisions based on this information, and sends a signal to agents in the layer above.

Another study that arranges agents into groups depending on their characteristics and roles, (Duan & Deconinck, 2010), converts a smart grid into a set of delimited domains representing functionalities, including generation, distribution, operations, markets, customers, and services. Each domain is then constituted by several agents that fulfill sub-parts of the domain. Furthermore, the study sets up a simple model that features an electricity market with smart grid load and generation agents performing bids and negotiations.

### **Generic agent-based modeling framework development**

ABM is a versatile tool that can be employed to analyze complex systems consisting of multiple autonomous agents. However, ABM is, by itself, just a tool and requires knowledge on agents' mutual relations and structure before organizations of real actors and objects can be validly modeled. The business ecosystem architecture development methodology proposed in (Ma et al., 2021) offers a solution to this task by mapping the architecture of an energy business ecosystem with the actors and objects, and their roles in the given business ecosystem. Relations and hence interactions between actors

and objects in the business ecosystem are considered to depend on the roles they serve rather than the actors and objects themselves. Furthermore, (Ma et al., 2021) categorizes identified relations between roles into five types of value flows (shown in Table 2).

The main output from (Ma et al., 2021) is a structured depiction of the investigated ecosystem information, Fig. 1, that is translated to the model framework proposed in this paper (shown in Table 3).

In the proposed model framework, an agent consists of parameters, state variables, behavioral methods, and functions that define the goals and decision-making mechanisms of the agent. Meanwhile, these elements are subjected to encapsulation and hence not visible to any entities in the environment outside the agent that contains them. Instead, as previously mentioned, an agent is represented externally by a set of roles it serves and the interactions that it can perform given by these roles. Based on these conditions, the model framework is divided into two steps:

- Step 1 – Interface and role interactions design
- Step 2 – Agent architecture and connections design

After performing these two steps, the agent logic components, i.e. the content that defines the behavior of the respective agent types, can be added. This task is not considered within the scope of this paper.

The framework is based on Java and employs some of the mechanisms of this programming language, notably abstraction by using interfaces. Therefore, the framework is applicable to ABM creation using ABM software implemented in Java.

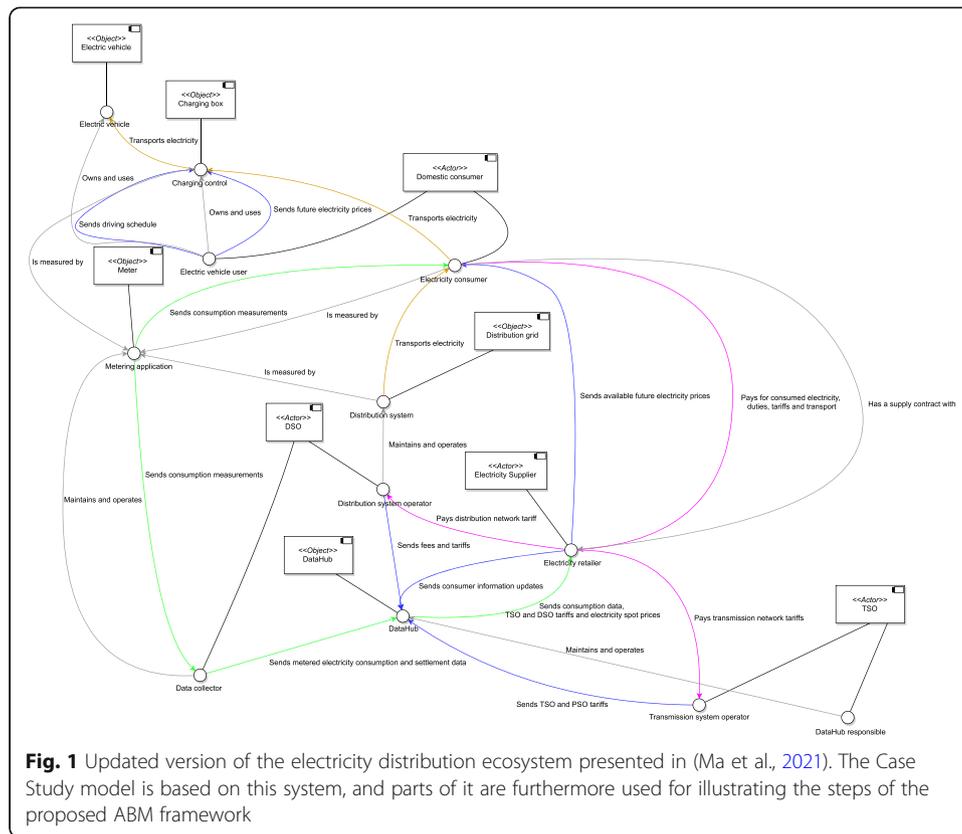
The steps and sub-steps of the model creation process are explained in the following sub-sections and are accompanied by illustrations in Fig. 2, 3, 4, 5, 6, 7 and 8. Note that these figures are based on a delimited set of elements from the system modeled in the Case Study section. The delimitation has been chosen to simplify the illustrations for explanatory purposes, and the figures therefore do not depict a complete system.

**Step 1 – Interface and role interactions design**

This step defines the structure of the roles and relations as interface and method counterparts in the agent-based simulation model. There are three sub-steps:

**Table 2** Five types of flows in the business ecosystem (Ma et al., 2021)

Type of flow	Description
Goods (Product & Service)	The most basic products of an economic system that consist of tangible consumable items (products) and tasks performed by individuals (service).
Monetary value	The amount of value an item or a service has in relation to if I were sold for cash to a willing buyer.
Information	Information is data that has been processed, organized, structured, or presented to make it useful in a given context.
Data	Data is raw, unorganized measurements and facts that need to be processed to become useful.
Intangible value	Intangible value is value created or owned by an individual, company, or organization that has no physical form. For example, the goodwill of an established business, the value of a trademark, or the ranking of a company or university.



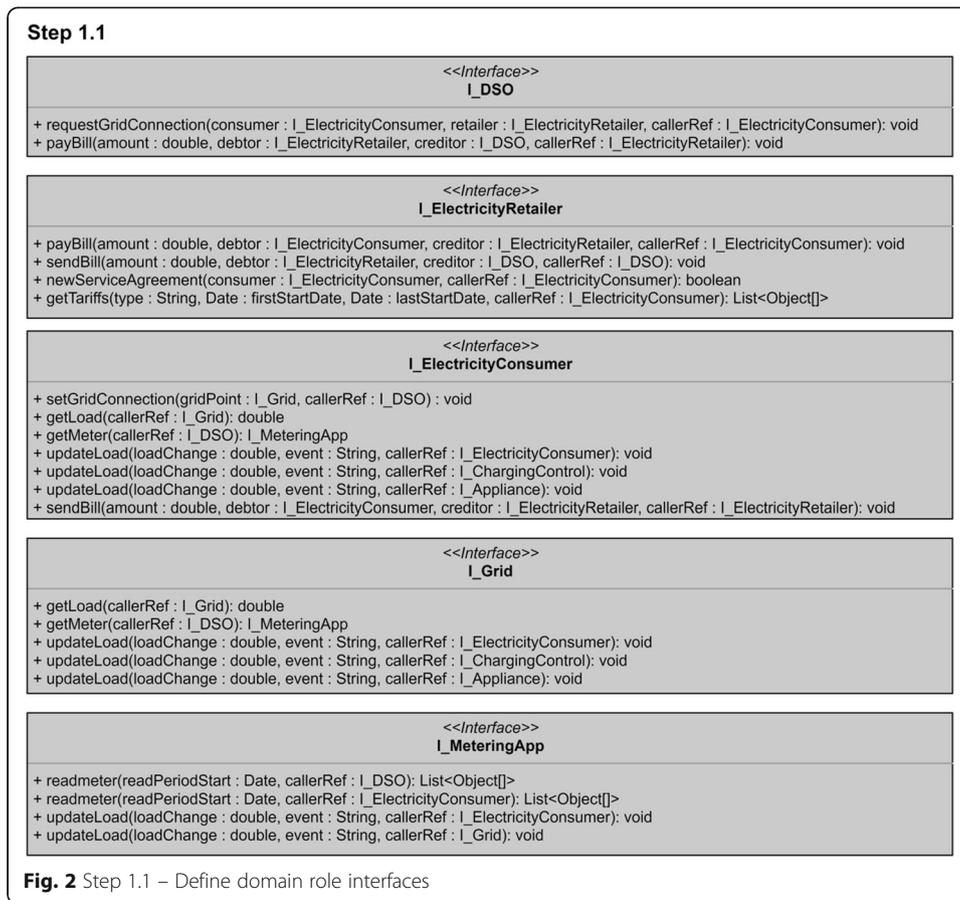
**Sub-step 1.1 - Define domain role interfaces**

**Purposes** The interface concept from object-oriented programming is considered a convenient way to represent roles in simulation models. Interfaces directly represent roles and come with sets of defined methods. Each method is given a name, an ordered set of expected inputs with types, and an output/return type. Interface methods represent the requests that agent types implementing the interface are expected to accept and handle as part of maintaining the corresponding role. These methods come without bodies, i.e. without any specified actions that must be performed when a request is handled. The method bodies are instead defined within the individual agent types, and thereby different agent types that implement the same interface might act differently to a certain request. The format of input information that must be provided and the format of the information that is returned, if any, are therefore the only properties visible to other agents that make requests through interface methods.

This sub-step considers the translation of roles from the selected energy ecosystem as interfaces in the ABM. As the actors in the ecosystem interact with each other based

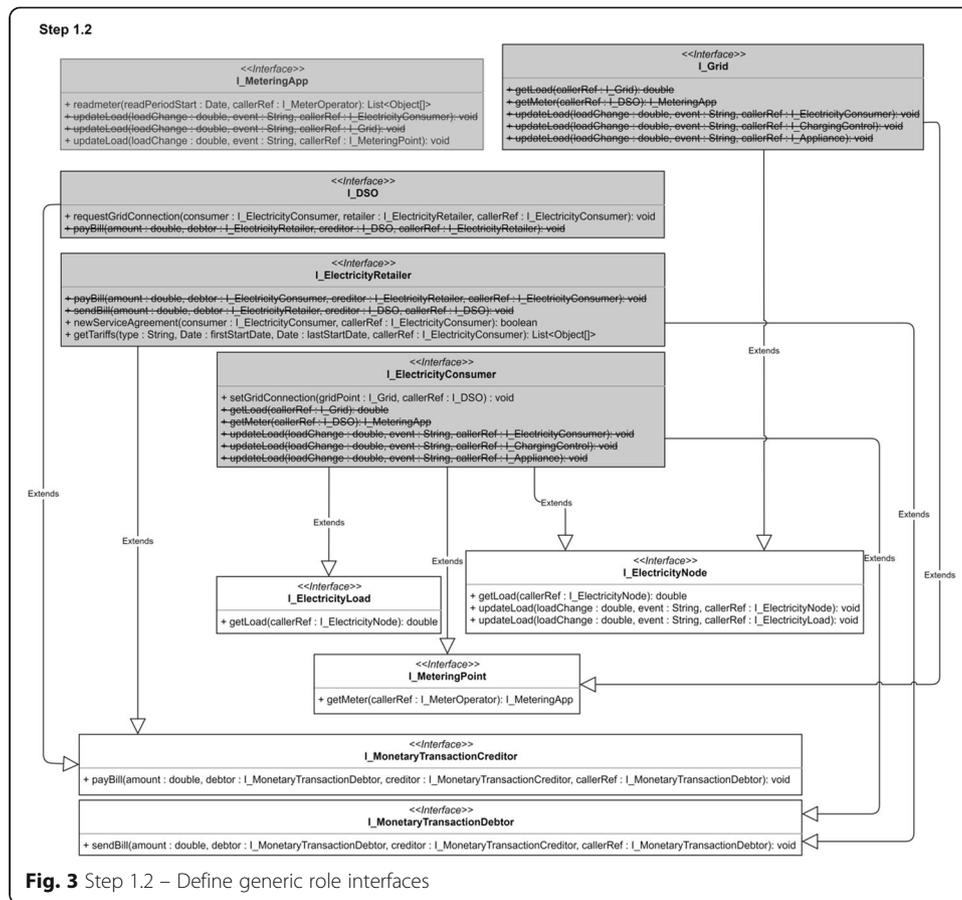
**Table 3** Translation of ecosystem elements into ABM framework

Ecosystem element	ABM element
Role	Interface
Actor/object	Agent type
Interaction	Interface method



on their roles, the agents’ interactions defined in the model must be based on the roles assigned to the agents. Hence, agents are completely unable to access and affect other agents unless these interactions are performed through the roles of the agents. This approach reflects the fact that actors in the real world typically possess little knowledge of other actors’ roles, except the ones who are relevant to them.

**Activities** Interfaces represent roles from the energy ecosystem as interface types. After the empty interfaces have been defined, they need to be assigned sets of methods that represent the requests they can receive from other interface roles. The methods of an interface are based on the flows in the ecosystem architecture that go into or out of the role represented by the interface. It should be noted, however, that the ecosystem architecture depicts flows and not interactions. Every flow comprises one to several different interactions that reflect procedures that commence between the real system roles. For instance, the exchange of electricity might encompass power requests, establishments, modifications and terminations of supply contracts, physical conditions such as failures, voltage fluctuations and maintenance, and many other interactions. The information on the energy ecosystem determines the scope and which interactions are relevant, and this should be reflected in the set of methods that are defined to represent the flows. Based on this condition, the methods must be defined by considering each flow rather than each role, identify necessary methods for this flow based on the energy ecosystem



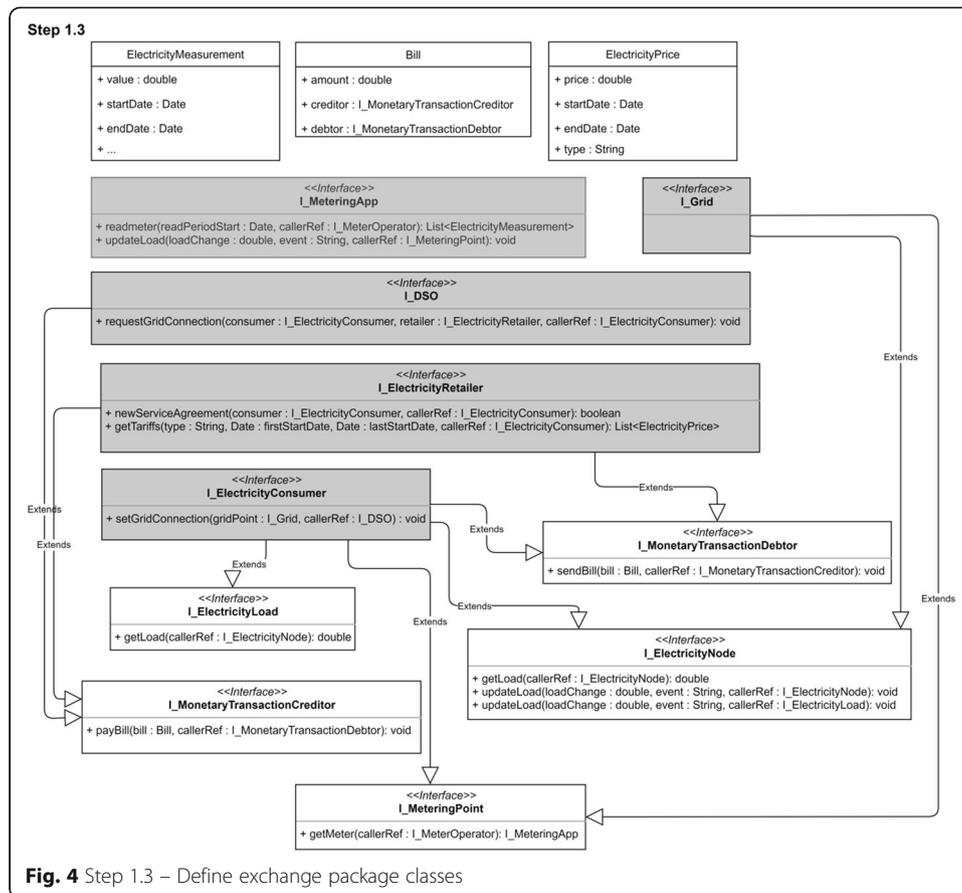
and assign these methods between the two roles that exchange the flow depending on who is the receiver, again according to the information on the energy ecosystem.

Furthermore, due to the requirement that not only the agent handling a request but also the agent making the request must assume a certain role, every interface method defined in the model must include an input parameter, *callerRef*, of the interface type corresponding to the role attained by agents expected to call that method. This ensures that agents that should not be able to call the method cannot access it without throwing an exception. The Fig. 2 *I\_ElectricityRetailer* interface method, *getServiceAgreement*, can for instance only be called by agents implementing the *I\_ElectricityConsumer* interface, i.e. agents fulfilling the electricity consumer role that wish to initiate an electricity supply contract with an electricity retailer role agent.

**Outputs** A list of domain role interfaces, one for each ecosystem role, with methods is the main output as shown in Fig. 2.

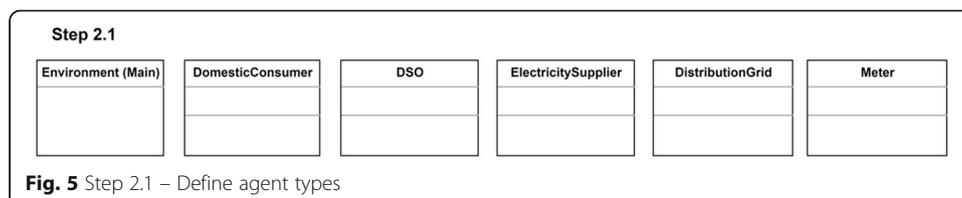
**Sub-step 1.2 - Define generic role interfaces**

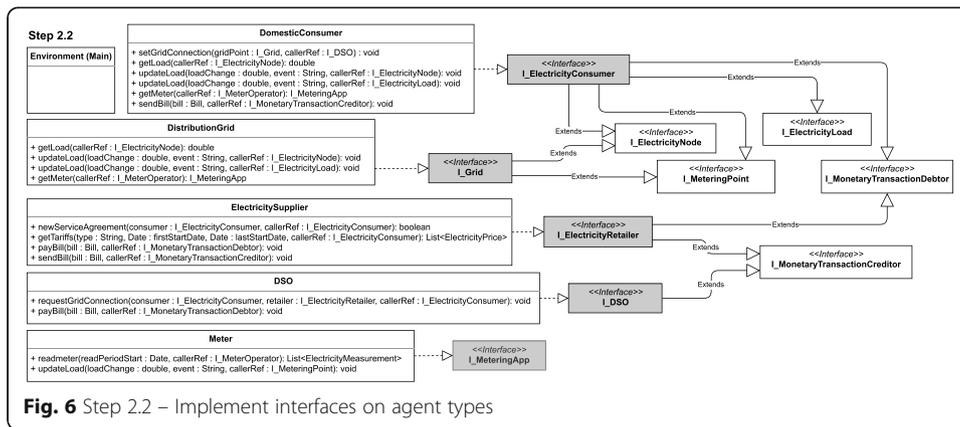
**Purposes** Usually, interfaces for different roles feature methods with identical signatures and purposes and hence share some functionalities that can be covered by a common super-interface, the generic role interface. Furthermore, a single



interface might feature multiple versions of the same method with different caller-Ref argument types, e.g. the *I\_ElectricityConsumer updateLoad* methods in Fig. 2. The number of versions can be reduced by convening the calling interface types under a common super-interface that reflects their similarities. Creating generic role interfaces has the following purposes and potential contributions they can provide to ABM of energy systems:

- Simplification: Duplicates of the same method across multiple roles (see the six *updateLoad* methods in *I\_ElectricityConsumer* and *I\_Grid* in Fig. 2) and within the same role, the latter to allow access from multiple different interfaces (see the two versions of the same method in *I\_ElectricityNode* in the same figure), can be reduced to one item.

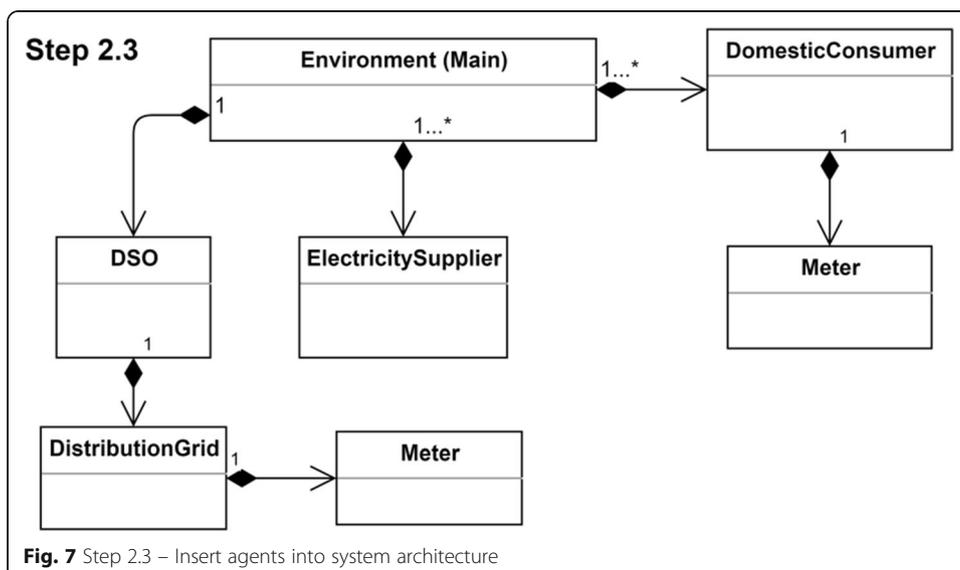




- Reusability: Generic role interfaces represent generalized roles in the domain of which the energy ecosystem is a part. These interfaces can be applied to roles with the same properties across different ecosystems and simulation models.
- Standardization of concepts: Using the same generic role interface for all roles that perform a certain task or service ensures that this service remains consistent across all domain interfaces that provide it and all ecosystems where it appears.

**Activities** This sub-step revises the interfaces from sub-step 1.1 to capture similarities between them and to separate these similarities into generic role interfaces that are valid within the context of the domain considered in the energy ecosystem. To give an example, a heat pump role and an EV charging control role are both characterized by acting as electricity loads, and the methods pertaining to this aspect of the roles can be separated into an independent electricity load super-role interface that is extended by the original two roles.

As Fig. 3 illustrates, the activity performed in this sub-step moves certain methods from the domain role interfaces into separate generic role interfaces that are extended



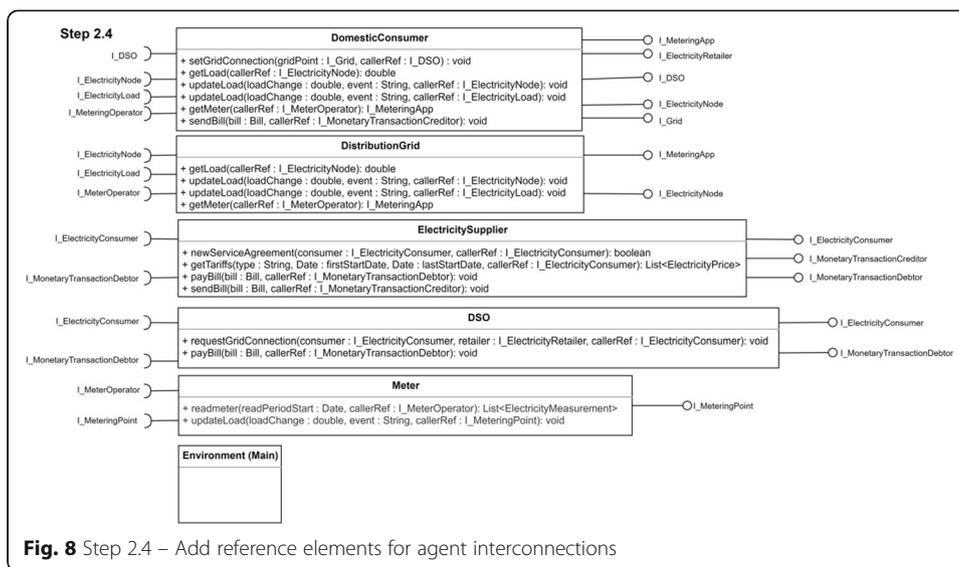


Fig. 8 Step 2.4 – Add reference elements for agent interconnections

by the former. For instance, the *I\_ElectricityConsumer* and *I\_Grid* interfaces both represent roles that serve as physical nodes in the electricity system, hence extending the generic *I\_ElectricityNode* interface. In some cases, generic role interfaces themselves might also feature similarities that can produce new generic role super-interfaces, thus forming a multi-layered hierarchy of interfaces. It is difficult to establish a definite guideline that describes when a generic role interface is warranted, however, when creating generic role interfaces based on domain role interfaces and their methods, the following requirements should be fulfilled:

- Methods from domain role interfaces that are moved into the same generic interface must feature similarity (not necessarily equality) in name, arguments, and purpose. There might be small differences between the methods, e.g. the exact name or the input arguments ordering, and these differences are eliminated when the methods become one in the generic role interface, thus standardizing the method.
- A generic role interface must assume a physical or virtual identity that is rational in the context of the modeled domain and which can be named accordingly. For instance, TSO and DSO domain role interfaces might both feature similar methods for grid maintenance and tariff updates, however, these consider unrelated tasks that cannot be combined into one generic role interface with a reasonable identity and name.

**Outputs** The output is a list of interfaces that cover abstract functionalities and corresponding generic roles as shown in Fig. 3.

**Sub-step 1.3 - Define data and information exchange classes**

**Purposes** While some interactions only include exchanges of few and primitive-typed values, e.g. a few double values or strings, other interactions feature extensive and

complex sets of exchanged objects as part of the input and return parameters. A special type of classes, data and information exchange classes, are used in this regard to contain several pieces of information in the same object. These classes are almost entirely constituted by fields and rarely contain a few methods to support certain functionalities. The use of these classes is convenient due to the same parameter type configurations often being used for more than one interaction in a system. For instance, the DataHub platform uses some standard protocols pertaining to electricity consumption measurement data and electricity consumer information, protocols which are also employed by other actors handling said information in the system. A consumption measurement exchange class that closely resembles the format and content dictated by the protocol can therefore conveniently represent the type of data in the model. Furthermore, as depicted in Table 2, the ecosystem interactions are categorized into exchanges of goods or finances, unprocessed or processed data or information, and requests and regulations through agreements and ownership relations between the actors. This separation of interaction types can also be represented by data and information exchange classes for the individual interaction types, for instance, ones that are used to represent the exchange of certain goods and finances. Finally, assembling multiple fields in the same objects makes it convenient to define methods with multiple return values instead of listing these values in an object array which is less manageable and clear.

Apart from assembling multiple output values and long lists of input arguments into single objects, the purposes of defining and using data and information exchange classes are the same as for the generic role interfaces.

**Activities** The activity of this sub-step considers the methods of the interfaces from steps 1.1 and 1.2 and identifies cases with multiple output values, long input argument lists, identical argument lists between methods, and packets of information that possess an identity within the modeled domain and its procedures, e.g. an electricity consumption measurement, a bill or an electricity price. With the creation, each data and information exchange class gets a constructor and fields with corresponding *get* and *set* methods.

Figure 4 illustrates this sub-step by introducing three data and information exchange classes, one representing electricity consumption measurements performed between two dates (ElectricityMeasurement), one representing a generic bill sent by a creditor to a debtor (Bill), and one representing an electricity price that is valid between two dates (ElectricityPrice).

It should be noted that data and information exchange objects are not only used in interactions but might also be subsequently stored and processed by involved agents.

It should furthermore be noted that this sub-step does not necessarily need to be carried out after sub-steps 1.1 and 1.2. Instead, the data and information exchange classes can be created gradually with the creation of the interface methods in sub-step 1.1, which might be more convenient in some cases.

**Outputs** A list of classes that assemble multiple fields into single objects, that are used for the interactions between interfaces and, in some cases, subsequently stored within the agents.

## **Step 2 – agent architecture and connections design**

With the interfaces and corresponding interaction methods and data and information exchange classes in place, the content of the agent types that relate to their position in the system can be created. This step considers the creation of the empty agent types and the elements that directly pertain to how they are positioned in the system architecture and how they are connected for communication.

### ***Sub-step 2.1 - Define agent types***

**Purpose** Before any activities can be carried out that involve agents, they must be defined as entities in the simulation model.

**Activities** This sub-step hence defines the empty agent types (or classes) which will act as templates that prescribe the structure and content elements that constitute agents of their respective type and are used to create agent instances of those types. Like the energy ecosystem role to domain interface conversion, actors/objects from the energy ecosystem are defined as agent types in the simulation model in a one-to-one manner. Therefore, for each actor or object in the energy ecosystem, one agent type must be created, using the same name as the actor/object (without spaces). Furthermore, an environment agent type must be created, which acts as the Main class that is instantiated at the start of a simulation and from which other agents are instantiated. The agents introduced for illustrative purposes are depicted in Fig. 5.

**Outputs** A set of empty agent types that obtain properties and contents in the following sub-steps.

### ***Sub-step 2.2 - Implement interfaces on agent types***

**Purposes** The actors and objects in the ecosystem assume roles, and correspondingly, the actor/object-based agent types in the simulation model must implement the role-based interfaces, hence providing the agents with the properties needed for acting in the simulated system.

**Activities** Given the list of agent types defined in sub-step 2.1 and the list of domain role interfaces from sub-step 1.1, the two entities can now be joined according to the role-actor/object relations given in the energy ecosystem architecture. This is done by letting the respective agent types implement the relevant domain role interfaces (shown in Fig. 6). This activity implicates that the methods in the implemented interfaces must be defined as part of the contents in the agent types themselves, however, the function bodies are not added in this step as they belong to the logic of the agent types.

**Outputs** Interfaces (and the extended generic interface types) are assigned to the agent types, and the corresponding empty-bodied methods are added to the agent types.

### ***Sub-step 2.3 - Insert agent into system architecture***

**Purposes** Until now, the sub-steps have considered the creation of concepts, i.e. agent types, object classes, and interfaces, but nothing has yet been set to instantiate agents during simulation runtime. This sub-step performs this task by embedding agent types within the environment agent or as part of other agents.

**Activities** Instances of agent types are stored in designated fields (or containers) which can either be variables that can only contain a single agent or collections that can contain populations of agents of the same type. As illustrated by Fig. 7, these fields are parts of agent type definitions of other agent types, hence agents are embedded in container elements in other agents in a structure with the environment agent (sometimes called Main) as the highest-leveled agent. This means that an agent type might be embedded in multiple other agent types. The agent type hierarchy must be based on the ecosystem architecture, and intangible interactions such as “owns” or “uses” in the architecture typically denote situations where an agent type, due to its maintained role, embeds instances of another agent type in an ownership relation. Hence, even though the agent hierarchy is per se based on agent types, it still must follow the role-relation requirement, i.e. an agent type should only be embedded in another agent type if the roles of the two types warrant it.

In Fig. 7 there is exactly one DSO agent instance embedded in the environment, together with populations of DomesticConsumer and ElectricityRetailer instances that each can contain at least one instantiated agent. Furthermore, adhering to the “owns” relation in the Fig. 1 ecosystem, the DomesticConsumer instance will feature a single embedded Meter agent.

Agents in these container fields, whether embedded in the environment or another agent type, can be instantiated immediately by their owner at the simulation start and/or be instantiated or destroyed dynamically during simulation runtime. The number of agents instantiated in the respective containers at simulation start is set as part of this task as well. Also note, as the arrows in Fig. 7 signify, composition is used as the association relation between owner and children, hence if an agent is destroyed, its embedded agents are destroyed too.

**Outputs** A definite ownership hierarchy of agent types and the instantiation of (empty) agents during simulation runtime.

### ***Sub-step 2.4 - Add reference elements for agent interconnections***

**Purposes** Before agents can interact with other agents based on the interfaces they implement, they must be able to access the other agents through references. This step introduces fields to store references to other agents of the interface types that a given agent can initiate interactions with.

**Activities** An agent type that implements an interface must be able to call methods in interfaces that feature *callerRef* arguments of the implemented interface type.

Therefore, it is necessary to identify every interface that an agent type can interact with through the interfaces it, itself, implements. For each identified interface, a field must be added in the agent type definition to store values of the identified interface type. These fields serve as references to other agents that implement said interfaces. The fields (called links henceforth) might either be variables or collections, depending on how many agents of a given interface type the agent must maintain connections to at a time. All outgoing requests performed by the agent go through these links. A link might contain references to agents of different types that implement the same interface, thereby enabling polymorphism. This approach fits with the ecosystem architecture development methodology where relations and interactions are formed between roles, not actors/objects.

In Fig. 8, sockets and circles are added to the individual agent types to denote which interfaces they can receive calls from, and which interfaces they can call, respectively. For each circle, a link of that interface type must be added to the agent type.

Link values, i.e. references to agent instances assuming a certain role, are sometimes given as a default value at simulation start or created dynamically, where an agent locates and requests connection to another agent, either by perceiving the latter in the environment or receiving a reference through a third party. However, this sub-step only adds the links while the actual references stored are instead defined as part of the individual agent type logic. Therefore, it is not certain that a link will be used at all during the simulation. For instance, a DomesticConsumer agent that, for some reason, remains unconnected to any grid role agent will not store any references in the link to contain an *I\_Grid* link that is depicted by a circle attached to the DomesticConsumer agent in Fig. 8.

**Outputs** The agent types contain interface-typed links to store references to other agents of the given interface type. The links are used as outgoing ports for interactions with those agents as part of the agent type logic.

### Case study

A local low-voltage distribution grid in Denmark is chosen as a case study to demonstrate the application of the proposed model framework. The low-voltage grid connects 137 domestic electricity consumers (shown in Fig. 9). The consumers are connected to a branch of the low-voltage grid that is connected upwards in the grid by a transformer station.

Currently, there are very few photovoltaics (PV) and EVs in this distribution grid. However, in the future, the loads might include various loads and productions by DERs, such as EVs, PVs, and heat pumps. The potential overload in the distribution grid by DERs is a big concern for the DSO. Therefore, an agent-based simulation model is developed with the proposed framework to investigate this scenario.

### Agent-based simulation model development

The model is developed by following the steps presented in the Methodology section. The energy ecosystem information for the case study is based directly on the preliminary work of (Ma et al., 2021) and the ecosystem presented in Fig. 1. The aim of this



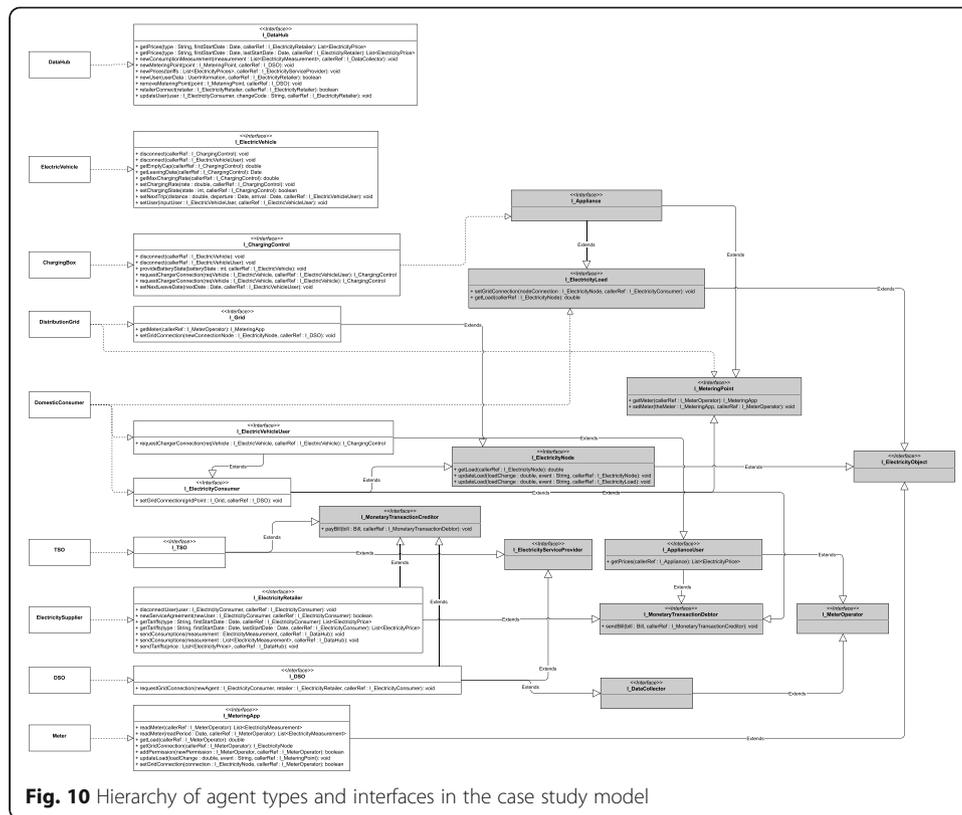


Fig. 10 Hierarchy of agent types and interfaces in the case study model

starting number and growth rate of ChargingBox and ElectricVehicle initialization reflects a function that depends on the scenario used for the simulation run.

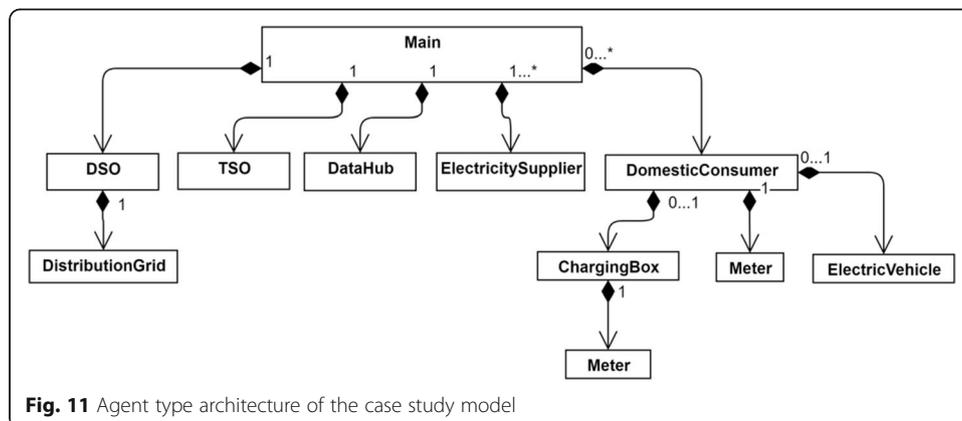
### Agent logic and behavior

As stated in the Methodology section, the logic of the agents is not within the scope of the framework presented in this paper. However, to understand the purpose of the case study model, a summary of the most important behavioral aspects of the agents found in the model is provided:

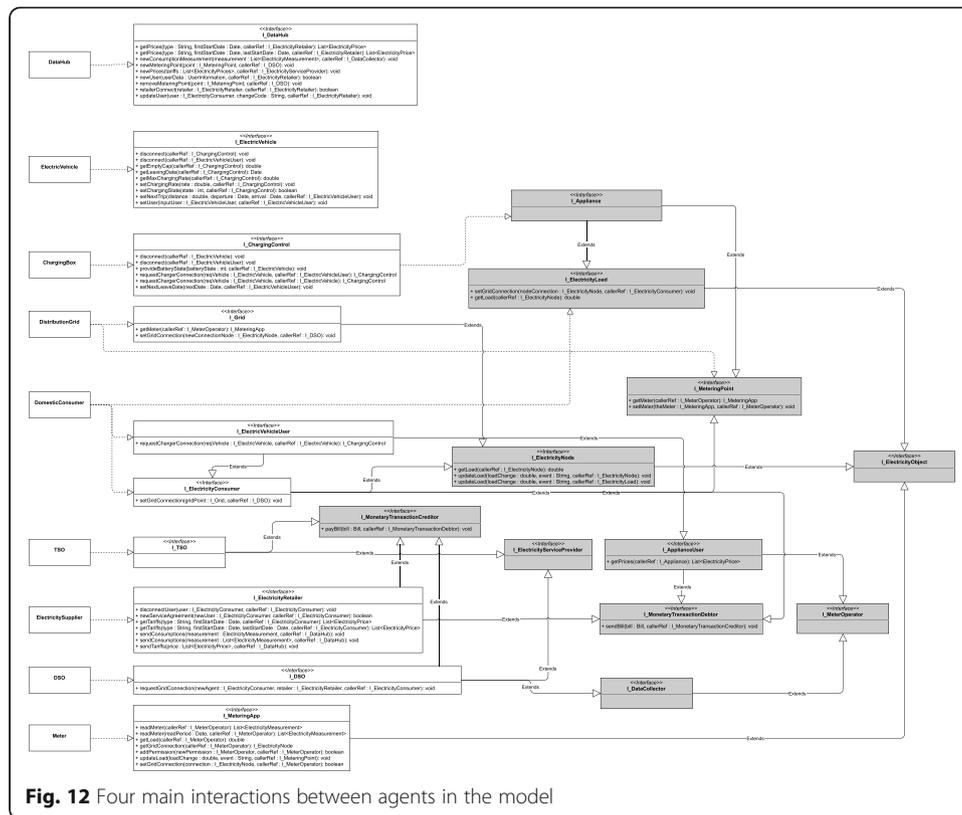
The main interactions between the agents in the model are divided into four different sequences, and each sequence covers a certain flow type from the ecosystem architecture (shown in Table 2). Furthermore, a set of data and information exchange classes is used in each sequence, as presented in Table 4. Figure 12 illustrates the four main interactions with sequence diagrams. Note that interfaces and agent types are both depicted, where the interactions are based on the interfaces while the performed actions are based on the agent types and their corresponding logic. Also, note that the electricity loads sequence contains a lifeline where only the interface is known (*I\_Appliance*). This denotes that the DomesticConsumer agent type does not care from which of its connected Appliance agents it received a load change signal. What is important is that the load changes. Thereby, any agent types that implement the *I\_Appliance* interface, e.g. PV systems, heat pumps, or batteries, can be embedded in and connected to the DomesticConsumer agent type without requiring changes to the agent type's logic that relate to the *I\_ElectricityConsumer* interface. Depending on the usage and control

**Table 4** A list of data and information exchange classes, the information they contain, and the sequences where they are prominently used

Sequence	Data and information exchange class	Contained information and [type]
Electricity load	PlannedLoad	Load value [double] Start date [Date] End date [Date]
Electricity consumption measurement	ElectricityMeasurement	Value [double] Start date [Date] End date [Date] Measurement event [String] Measurement type (power or energy) [String] Meter reference [ <i>IMeteringApp</i> ] Consumer reference [ <i>IElectricityConsumer</i> ]
Electricity price	ElectricityPrice	Price [double] Start date [Date] End date [Date] Type/component [String]
Electricity billing/payment	Bill	Amount [double] Creditor [ <i>IMonetaryTransactionCreditor</i> ] Debtor [ <i>IMonetaryTransactionDebtor</i> ] Note [String]
	BillBundle	Creditor [ <i>IMonetaryTransactionCreditor</i> ] Debtor [ <i>IMonetaryTransactionDebtor</i> ] Sub-bills [List<Bill>] Total amount [double] Note [String]



**Fig. 11** Agent type architecture of the case study model



interactions between the added Appliance role agent types and the DomesticConsumer agent, the latter might need to implement an interface, *I\_XApplianceOwner*, and corresponding logic for specifically handling these additional interaction requirements (like the *I\_ElectricityVehicleOwner* interface).

Regarding the ChargingBox and ElectricVehicle agents, the logic is as follows:

The DomesticConsumer agent, given that it has initialized and hence owns an instance of each of the ElectricVehicle and ChargingBox agents, exhibits a scenario-dependent driving pattern. The first trip is scheduled when the ElectricVehicle agent is instantiated, and the expected distance and departure and arrival times of the trip are sent to the vehicle.

The ElectricVehicle agent can attain four different states, which are depicted in the Fig. 13 flowchart:

- Driving – the vehicle is away from the household and cannot be charged, and the charge level of its battery is reduced depending on the distance of the trip and the mileage of the vehicle.
- Connected to Charger – the vehicle is home and connected to a charging control role agent, in this case the ChargingBox agent type, but is currently not charging.
- Charging – the vehicle is home and connected to a charging control role agent and is currently charging, hence increasing the battery charge level and causing a load at the electricity node to which it is connected.
- Idle – The vehicle is home but not connected to a charger (this only occurs if the battery is fully charged).



curves, as is the load capacity supported by the grid. Figure 15 shows the aggregated number of ElectricVehicle agents in each state in the system, hence providing an overview of the charging and driving behavior exhibited by the vehicles during simulation runtime.

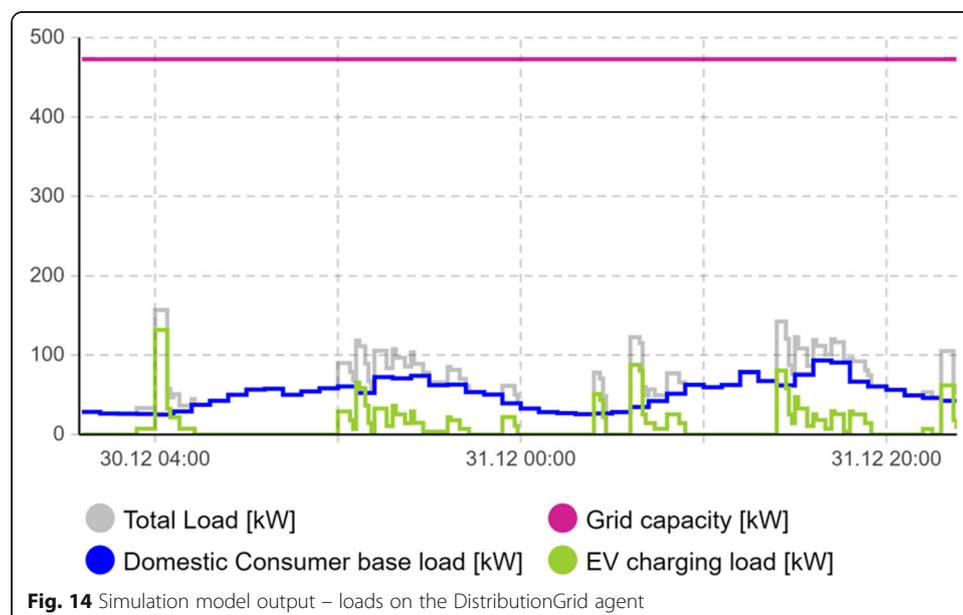
At the end of the simulation, all measurements of the DistributionGrid loads and ElectricVehicle states obtained over the entire simulation period are exported with time-stamps to a spreadsheet file for further analysis, i.e. investigation of grid overload timing and causes and how these relate to the observed driving and charging patterns of the ElectricVehicle agents. The exported data also includes the input parameter configuration of the simulated scenario.

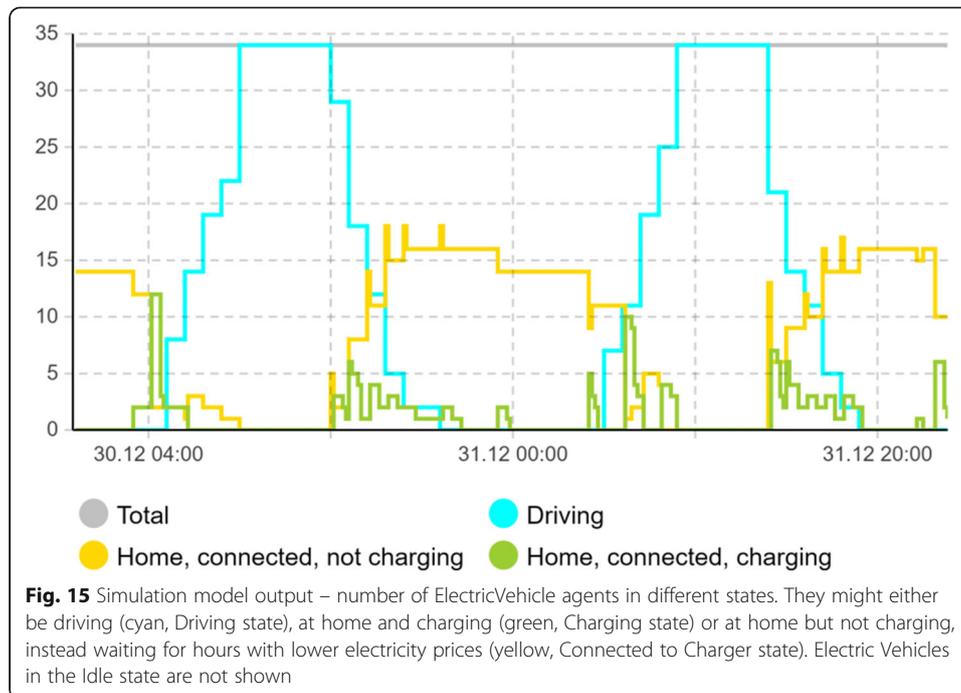
It should be noted that the set of variables that are measured during simulations can be extended to include new variables, for instance accumulated electricity service payment/revenue of the DSO, TSO, ElectricitySupplier and/or DomesticConsumer agents. The Main (environment) agent contains a list of variables that are measured. Adding new variables to the list will automatically print them in a new column in the output spreadsheet. Furthermore, the measurement frequency can be increased or decreased from the 10 min if a higher or less level of detail is needed for the output dataset, respectively, and the 48 h period of the two charts can be modified if measurements over a longer or shorter period must be monitored during simulation runtime.

## Discussion

The case study demonstrates how an ecosystem with several actors and objects can be translated into an agent-based simulation model. Even though a simulation model was successfully created, the case study also showed that the proposed ABM framework has some limitations:

- It is difficult to exactly define the methods needed for interactions as part of creating the domain interfaces in sub-step 1.1. Even though the interactions





depicted in the ecosystem and their flow types can be used as guidelines for the needed interface methods, they only provide an overall notion of which requests a certain interface must be able to handle. The need for some methods might not be apparent before later steps of the framework and some of the agent type logic contents creation have been conducted, and hence new methods might be added at later stages in the modeling process or existing methods might be removed/modified.

- While sub-step 1.2 provides some overall guidelines regarding situations where generic interfaces should be introduced, the framework does not offer any definite way to determine when a generic interface should be added and which methods that should be moved from other interfaces that extend it. A similar limitation exists regarding the definition of data and information exchange classes in sub-step 1.3.
- Defining the logic of the individual agent types is not part of the framework as it belongs to the private content of the respective agents and hence does not directly affect the role-based organization between the agents. However, the roles attained by an agent type might affect how the logic should be structured. For instance, it might be appropriate to strictly separate content belonging to different roles of an agent into distinct modules and set up interactions between these modules as if they belonged to different agents, thereby limiting changes to other modules if a certain role and corresponding module is removed from the agent type. Hence, the framework could be extended with step 3 which provides some recommendations on conveniently structuring agent type content pertaining to its logic.

During the case study model creation process, these limitations caused a need to cycle through previously performed sub-steps multiple times, notably sub-steps 1.1–1.3

and in some cases 2.4. Regardless, the framework still provides some level of standard structure, both in the creation process, but also in the final product.

A future task that needs to be solved concerns testing and evaluation of the performance, usability, and reliability of the proposed framework through a verification process. The verification process should test whether the agent-based simulation models behave according to the specifications, both in terms of individual agent behavior, interactions between agents and the environment, and the overall system behavior. Software testing methods can be used for this verification and can identify if agents and components in a simulation model behave properly to various conditions. Special conditions that are unlikely to occur or have been overlooked by developers should also be identified and evaluated. For instance, a *DomesticConsumer* agent might not have any reference to an upper grid node agent stored in its designated reference variable, and in this situation, it is important to investigate whether this agent still receives electricity and behaves as if it were grid-connected.

Circumstances that comprise changes to parameters and properties might feature various values between scenarios. Therefore, the evaluation of model components' behaviors is required. For instance, what would happen to the consumption measurement values exchange flow from Meter agent to Electricity Supplier agent if the DSO agent, instead of every 60 min., performs meter readings every 15 min or 1 day? Other situations that need to be tested could be: What would happen if the information on any components of the final electricity price, e.g. availability of the DSO tariff become delayed or missing altogether for some hours, or the same happens with the entire final electricity price? Will these changes to data and information flow availability and timing cause critical impacts on other agents' behaviors and processes such as the billing procedure?

For the verification process, a bottom-up approach is recommended, beginning with the testing of individual model components, and gradually combining increasing numbers of components until the entire system is tested. This approach is similar to software testing, including:

- Unit testing: tests individual agent types' responses to certain sets of input stimulations and verifies their reactions and behaviors according to expectations.
- Integration testing: tests the performance of specific interactions between two or few agents in a delimited version of the simulation model and verifies that chains of interactions behave according to expectations.
- System testing: tests the behavior of large parts of the model that combine numerous components, and the full model, and verifies that the entire system, incl. Eventual emergent patterns, behave according to expectations or in an explainable way.

Therefore, a standardized verification procedure should be conducted to ensure the performance of the developed ABM framework. Furthermore, the verification procedure should be designed together with the model design and conducted when new features are added to the system.

## Conclusion

This paper proposes an agent-based modeling framework for developing agent-based simulation models of energy business ecosystems. The framework constitutes Part III

of the business ecosystem modeling proposed by (Ma, 2019), uses information from the business ecosystem modeling for organizing the agents included in the simulation, and is intended to be applicable across multiple simulation scenarios by configurations of agents and roles. This system-oriented approach differs from but also potentially complements the general agent logic-oriented approach.

The proposed framework consists of two steps: Step 1 considers interface and role interactions design while step 2 considers agent architecture and connections design. In addition, a pre-step to the framework is constituted by Part I of the business ecosystem architecture development methodology in (Ma et al., 2021), which provides the information required to organize the agents in the simulation model created using the framework.

The framework is illustrated with a case study of an energy business ecosystem consisting of an electricity distribution grid with 137 connected domestic consumers and a gradually increasing electric vehicle prevalence. The case study presents the design and implementation of actors, roles, and relation flows from the ecosystem architecture of the energy business ecosystem into the corresponding agents and their respective relationships in the agent-based simulation model.

While multiple cycles of some framework sub-steps have been required during the creation process, the agent-based simulation model in the case study shows that the proposed framework can be used as a generic agent-based modeling framework for modeling energy business ecosystems. To ensure the performance of developed agent-based simulations, a verification procedure should be included in the model framework as the next step. The verification procedure should include unit, integration, and system testing similar to software testing. Meanwhile, multiple case studies with different distributed energy resources and different energy business ecosystems (e.g. heating and cooling) can be conducted with the proposed model framework in the future for verifying the framework and extending it as necessary.

#### **Abbreviations**

ABM: Agent-based modeling; DER: Distributed energy resource; DSO: Distribution system operator; EV: Electric vehicle; PV: Photovoltaics; TSO: Transmission system operator

#### **Acknowledgments**

Not applicable.

#### **About this supplement**

This article has been published as part of Energy Informatics Volume 4, Supplement 2 2021: Proceedings of the Energy Informatics Academy Conference Asia 2021. The full contents of the supplement are available at <https://energyinformatics.springeropen.com/articles/supplements/volume-4-supplement-2>.

#### **Authors' contributions**

MV, ZM and BNJ have contributed to the methodology design development. MV has conducted the case study model creation and written the first draft of the paper, while ZM, YD and BNJ have contributed to the final manuscript. All co-authors have read and approved the final manuscript.

#### **Funding**

This work is conducted as part of the national electricity digitalization transition project, Flexible Energy Denmark (FED) which is funded by Innovation Fund Denmark.

#### **Availability of data and materials**

Not applicable.

## Declarations

### Ethics approval and consent to participate

Not applicable.

### Consent for publication

Not applicable.

### Competing interests

The authors declare that they have no competing interests.

### Author details

<sup>1</sup>Center for Energy Informatics, Maersk Mc-Kinney Moeller Institute, University of Southern Denmark, 5000 Odense, Denmark. <sup>2</sup>Center for Health Informatics and Technology, Maersk Mc-Kinney Moeller Institute, University of Southern Denmark, 5000 Odense, Denmark. <sup>3</sup>Laboratoire d'Informatique de Grenoble, Centre National de la Recherche Scientifique, Grenoble, France.

Published: 24 September 2021

## References

- Adner R (2012) *The wide lens*: Penguin Putnam Inc; 2012 29. Mar
- AnyLogic. 8 University 8.7.1 ed. anylogic.com: The AnyLogic Company; 2021
- Basisfremskrivning 2020 – Danmarks Klima- og Energifremskrivning: The Danish Energy Agency; 2020. Available from: [https://ens.dk/sites/ens.dk/files/Basisfremskrivning/basisfremskrivning\\_2020-webtilg.pdf](https://ens.dk/sites/ens.dk/files/Basisfremskrivning/basisfremskrivning_2020-webtilg.pdf). Access date: 19 May 2021
- Billanes JD, Ma Z, Jørgensen BN (2017) Consumer central energy flexibility in office buildings. *J Energy Power Eng* 2017(11): 621–630
- Christensen K, Ma Z, Demazeau Y, Jørgensen BN (2020) Agent-based Modeling of Climate and Electricity Market Impact on Commercial Greenhouse Growers' Demand Response Adoption. 2020 RIVF international conference on computing and communication technologies (RIVF); 2020. IEEE, Ho Chi Minh City, pp 1–7
- Christensen K, Ma Z, Værbak M, Demazeau Y, Jørgensen BN (2019) Agent-based Decision Making for Adoption of Smart Energy Solutions. 2019 IEEE Sciences and Humanities International Research Conference (SHIRCON); 2019 13–15 Nov, pp 1–4
- Data, tabeller, statistikker og kort Energistatistik 2019: The Danish energy agency; 2019 Available from: [https://ensdk/sites/ensdk/files/Statistik/energistatistik2019\\_dk-webtilg.pdf](https://ensdk/sites/ensdk/files/Statistik/energistatistik2019_dk-webtilg.pdf) Access date: 19 May 2021
- Duan R, Deconinck G (2010) Future electricity market interoperability of a multi-agent model of the Smart Grid. 2010 International Conference on Networking, Sensing and Control (ICNSC); 2010 10–12 Apr, pp 625–630
- Fatras N, Ma Z, Jørgensen BN (2020) System Architecture Modelling Framework Applied to the Integration of Electric Vehicles in the Grid. 17th International Symposium on Distributed Computing and Artificial Intelligence, DCAI 2020. Springer International Publishing, pp 205–209
- Howard D, Ma Z, Engvang J, Hagenau M, Jørgensen K, Olesen J et al Optimization of energy flexibility in cooling process for brewery fermentation with multi-agent simulation. 6th IEEJ international workshop on sensing, actuation, motion control, and optimization; 16/03/2020. Shibaura Institute of Technology, Tokyo <http://id.nii.ac.jp/1031/00127065/2020>
- Ma Z (2019) Business ecosystem modeling - the hybrid of system modeling and ecological modeling: an application of the smart grid. *Energy Informatics* 2(1). <https://doi.org/10.1186/s42162-019-0100-4>
- Ma Z, Asmussen A, Jørgensen B (2018) Industrial consumers' smart grid adoption: influential factors and participation phases. *Energies*. 11(1):182. <https://doi.org/10.3390/en11010182>
- Ma Z, Christensen K, Jørgensen BN (2021) Business ecosystem architecture development: a case study of electric vehicle home charging energy informatics
- Ma Z, Schultz MJ, Christensen K, Værbak M, Demazeau Y, Jørgensen BN (2019) The application of ontologies in multi-agent Systems in the Energy Sector: a scoping review. *Energies*. 12(16):3200. <https://doi.org/10.3390/en12163200>
- Ma Z, Sommer S, Jørgensen BN (2016; Ottawa, Canada) The smart grid impact on the Danish DSOs' business model. 2016 IEEE Electrical Power and Energy Conference (EPEC); 2016 12–14 Oct, pp 1–5
- Macal CM, North MJ (2010) Tutorial on agent-based modelling and simulation. *J Simulation* 4(3):151–162. <https://doi.org/10.1057/jos.2010.3>
- Market models for aggregators - activation of flexibility: Energinet.dk; 2017. Available from: <https://en.energinet.dk/-/media/Energinet/Publikationer-TLU/Markedsmodel/Market-models-for-aggregators.pdf?la=en>. Access date: 19 May 2021
- Oliveira P, Pinto T, Morais H, Vale Z (2012) MASGrIP - A Multi-Agent Smart Grid Simulation Platform. 2012 IEEE Power and Energy Society General Meeting; 2012 22–26 July, pp 1–8
- Praca I, Ramos C, Vale Z, Cordeiro M (2003) MASCEM: a multiagent system that simulates competitive electricity markets. *IEEE Intell Syst* 18(6):54–60. <https://doi.org/10.1109/MIS.2003.1249170>
- Ringler P, Keles D, Fichtner W (2016) Agent-based modelling and simulation of smart electricity grids and markets – a literature review. *Renew Sust Energ Rev* 57:205–215. <https://doi.org/10.1016/j.rser.2015.12.169>
- Roles & responsibilities: Energinet.dk. Available from: <https://en.energinet.dk/Electricity/New-player/Roles-and-responsibilities>. Access date: 19. May 2021
- Santos G, Pinto T, Praça I, Vale Z (2016) An interoperable approach for energy systems simulation: electricity market participation ontologies. *Energies*. 9(11):878. <https://doi.org/10.3390/en9110878>

- Shafie-Khah M, Catalão JPS (2015) Multi-layer Agent-Based Decision Making Model with Incomplete Information Game Theory to Study the Behavior of Market Participants for Sustainability. 2015 48th Hawaii International Conference on System Sciences; 2015 5–8 Jan, pp 856–865
- The Danish electricity retail market: Energinet.dk. Available from: <https://en.energinet.dk/-/media/230C57ABF72741C2A45072D8BD992E14.pdf?la=en&hash=8E704F8E14F2A6092731E1B3B4517AB9E80D31E1>. Access date: 19. May 2021
- The electricity grid - Analysis of Danish power system function: The Danish Energy Agency. Available from: [https://ens.dk/sites/ens.dk/files/El/elanalyse\\_uk.pdf](https://ens.dk/sites/ens.dk/files/El/elanalyse_uk.pdf). Access date: 19 May 2021
- Værbak M, Ma Z, Christensen K, Demazeau Y, Jørgensen BN (2019) Agent-Based Modelling of Demand-Side Flexibility Adoption in Reservoir Pumping. 2019 IEEE Sciences and Humanities International Research Conference (SHIRCON); 2019 13–15 Nov, pp 1–4
- Zheng M, Prijaca Z, Jørgensen BN (2016) The international electricity market infrastructure-insight from the nordic electricity market. 2016 13th International Conference on the European Energy Market (EEM); 2016 6–9 June, pp 1–5
- Zhu H, Zhou M. Role-based multi-agent systems. *Personalized Information Retrieval and Access: Concepts, Methods and Practices*. 2008, Role-Based Multi-Agent Systems

### Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

**Submit your manuscript to a SpringerOpen<sup>®</sup> journal and benefit from:**

- ▶ Convenient online submission
- ▶ Rigorous peer review
- ▶ Open access: articles freely available online
- ▶ High visibility within the field
- ▶ Retaining the copyright to your article

---

Submit your next manuscript at ▶ [springeropen.com](https://www.springeropen.com)

---