Nat Comput (2015) 14:129–144
DOI 10.1007/s11047-014-9469-y

Advantages of Model Driven Engineering for studying complex systems

Jose Evora · Jose Juan Hernandez · Mario Hernandez

Published online: 7 November 2014

Abstract The evaluation of the emergent behaviour in complex systems requires an analytical framework which allows the observation of different phenomena that take place at different levels. In order to observe the dynamics of complex systems, it is necessary to perform simulations so that both local and the emergent behaviour can be observed. To this end, the way in which complex system simulators are built must be examined so that it will be feasible to model large scale scenarios. In this paper, the use of Model Driven Engineering methodology is proposed to deal with this issue. Among other benefits, it is shown that this methodology allows the representation and simulation of a complex system providing support for the analysis. This analysis is supported by a metamodel which describes the system components that are under study. The application of this methodology to the development of large scale simulators is explored through a case study. This case study analyses a complex socio-technical system: a power grid.

Keywords Agent based model · Electrical system · Domestic load curve simulation · Residential demand · Demand side management · Appliances · Model driven engineering · Simulation framework · Demand size management · Complex system

J. Evora (✉) · J. J. Hernandez · M. Hernandez
SIANI, University of Las Palmas de Gran Canaria, Las Palmas, Spain
e-mail: jose.evora@siani.es

1 Introduction

The study of physical systems requires the elaboration of models that show their dynamic behaviour. All physical systems have the following properties: they have many components located in space and time; they are represented with a state that changes throughout time; and they evolve under the influence of energy. These components possess energy due to their movement, their chemical composition, their position, their temperature, their mass or other properties.

To begin with, an exhaustive identification of all relevant components that are part of a system is required. Using this, a model that allows the study of emergent behaviour can be built by describing the individual behaviour of each component and interconnecting them.

These models reproduce the behaviour of physical systems: mechanical (translational or rotational), hydraulic, thermal, electromechanical or electrical. These models are usually created for a better understanding of the system. That is, models are made to develop the formulation of scientific theories as well as to enhance the system's design and operation.

However, models are built through simplification processes with the aim of making it easier to understand a real system. In this sense, part of the complexity of the real system is neglected in the process of creating the model.

Physical systems are complex by nature. However, traditionally, simplified models of these systems have been made, using ideal entities and simple equations. Nevertheless, this notion of simplification is not the only method of modelling. Systems can also be studied in a higher level of detail, with non-ideal entities, non-linear behaviours and with emergent phenomena. This is a paradigm that is focused on the creation of models that are more detailed

and, therefore, multi-purpose. That is, models can be used in different use cases instead of building simplified models with just one purpose.

Models of complex systems are made up of many components. These components are individually described within the model using known laws. An essential feature of complex system models is that emergent behaviours arise from the integration of simple components with their mutual interactions. That is, behaviour is not directly deducible based on the individual behaviour of their components.

Furthermore, many physical systems are subject to human intervention. From this point of view, they can be considered as Complex Adaptive Systems (CAS). A CAS is a special case of a complex system which has a large numbers of components, often called agents, that interact and adapt or learn (Holland 2006). These agents manage to develop a single, unique identity which keeps a stable and coherent pattern throughout time (Holland 1995).

Examples of CAS are embryonic development, insect societies and power grids. In all of these cases, emergent and self-organizing principles are present. In a network of components without a central control, but simple operational rules, creating a collective behaviour. The result is the product of a huge number of decisions that are made by many individual components.

Based on these models, software simulations can be developed to study and understand complex systems. However, it is necessary to approach simulator engineering with a methodology that supports the analysis, modelling and design of software simulations with a large number of components.

Simulation tools are not specifically designed for developing large and complex system simulations and show limitations when building systems with more than 1,000 components. In the case study presented in this paper there are 20,000 components, developed by five engineers.

This paper proposes the development of simulators under the Model Driven Engineering (MDE) methodology. MDE improves productivity, quality and flexibility in developing software (Schmidt 2006). An MDE platform, called Tafat, has been developed to conduct this research work in overcome the drawbacks of other simulation tools.

The major benefits of this platform are the support for: the analysis and design of software simulators; the description of simulation scenarios with a domain specific language; the integration of simulations through a shared semantic; and the management of complexity in large system development. At the same time, its modular architecture allows the specialisation of the software team. There are different types of engineers focused on the development of the simulation engine, behaviour components and simulation models.

This research is exploratory and contributes to the validation of MDE as a useful approach to building simulators that support large scale models. This paper presents an experimental case which shows that MDE offers practical advantages in the context of power grid simulations.

2 Case study on electrical systems

Classically, electrical system management has been done based on the aggregated consumption data at a global level. The nature of this management was centralized since the system was considered as an indivisible unit.

However, in recent years, an increasing interest in knowing more about electrical demand at low levels of the power grid can be observed (Palensky and Dietrich 2011). This is mainly due to the process of change the electrical systems are undergoing, causes by factors such as the introduction of renewable energy sources (RES) as well as distributed generation (DG). These changes require a better knowledge of the load curves at a distributed level, which involves different factors such as electrical equipment, user behaviour, environmental conditions, etc. that have a potential impact on consumption.

From this point of view, the management of future electrical systems must overcome the restrictions of aggregated models and start analysing the electrical system in a disaggregated manner. In the bibliography, several methods of analysis of the electrical system have been developed in a disaggregated manner in Capasso et al. (1994), Palensky et al. (2008), Evora et al. (2011), Ramchurn et al. (2011).

2.1 Electrical systems

The electric power system is composed of different electrical components that allow for the production, transmission and consumption of electric power. Production or generation of electric power is the process of converting energy in other forms (chemical, mechanical, nuclear, etc.) into electrical energy. For the transmission, different kinds of electrical networks are used. Generally, they are classified by their nominal voltage at each level. Long range transmissions over hundreds of kilometres are performed at high voltages of several hundreds of kilovolts (kV). These networks are called transmission systems. Transformer stations (substations), can reduce the voltage at a given point in order to feed electricity to the so called distribution system. The distribution system carries the electricity to the final consumers. These networks operate at medium voltage levels, usually between 1–50 kV. In a final stage, distribution transformers can convert from medium voltage to low voltage (less than 1 kV) which is the typical voltage

level found at residential or tertiary customers. Some specific customers (such as industries) may have direct connections at medium voltage level, too.

In a *classical* energy system, generation is injected at high or medium voltage level and consumed in the distribution system. Following the trend of introduction of renewable energy sources and distributed generation, injections of energy at almost all levels of the system are possible.

The power grid can be seen as a complex system, being composed of a large number of interacting entities. The reproduction of the behaviour of the system is therefore not possible through modelling the system as a whole.

2.2 Electrical systems complexity

An electrical system shows analogies with complex living systems, such as ant or bee colonies, in which there are several agents making decisions and acting locally. Actions that each agent are performing locally are aggregated, and these actions can lead to emergent phenomena.

In this sense, in electrical systems there are people living in households that can be considered as agents (Wooldridge 2002). Moreover, these agents are intelligent (Monti et al. 2010) since they are self interested units with goals and exhibit adaptive behaviour to their environment. These agents are able to make decisions and coordinate their actions with other agents. The main difference with complex living systems is that in the electrical system, we are not interested in the study of the emergent behaviour starting from the local actions, but the modification of local behaviour to obtain the desired emergent behaviour (Stepney et al. 2006). However, from a second point of view, a complex system simulation model can serve to observe unwanted emergent phenomena and study these effects on the system. An approach to analyse complex systems from this point of view can be seen in Polack et al. (2008).

2.3 Modelling Electrical Systems

According to the previous view, Electrical Systems can be modelled as Complex Systems. This kind of modelling requires the representation of all existing electrical entities. Nevertheless, in addition to an electrical behaviour (Mitchell 1992), the entities may also have mechanical, thermal or chemical behaviour.

For instance, a wind turbine is a machine that transforms mechanical movement into electricity. Therefore, this entity presents two different behaviours: a mechanical behaviour that transforms wind speed into the rotational speed of the blades, and an electrical behaviour that transforms rotational speed into electricity.

This means, the resulting model is not only including electrical behaviour, but also other types of behaviour. These types of behaviour are included in the model since they affect the electrical components.

Power grids cannot be studied in a mechanistic way since they are used by agents that are autonomous, independent and self-interested. Each of them interacts with the grid in order to consume or produce energy in different ways.

There are agents that are making simple decisions locally related to producing or consuming energy. For instance, in the case of customers, the decision of consuming energy is a consequence of specific activities such as cooking, heating the household or watching TV.

Therefore, the power grid can be observed as a system in which there is a huge number of relatively simple agents, that adapt themselves to new conditions of the environment or to new objectives. There are agents of different nature: producers and consumers. The interaction among these agents is produced through the power grid to which they are connected (Wildberger 1997).

In Massoud Amin and Wollenberg (2005), an agent-based modelling approach of a power grid is proposed. In this model, there are agents that directly influence the production or consumption of energy. Generally, it can be said that the dynamics of the system is not only dependent on the electrical behaviour of the devices but also social behaviour. That is:

1. The interaction of the inhabitants of the households with the electrical appliances that are inside involves a consumption. Since this is the main interest of the case study that will be described, this reality must be modelled. Based on this study, comparisons of the residential consumption of these households can be made according to profiles of standard consumption.
2. According to the external temperature, a household has a thermal behaviour that produces a power consumption in order to heat or cool the household. The comfortable temperature of the household is determined by the preferences of each individual, as well as the time when such individual is at home. Obviously, if the household is empty, the consumption will be lower.
3. The temperature inside the household is not only a consequence of the external temperature, but also of the number of people that are inside the household and the type of activity that they are doing. For instance, if the individuals are cooking or there are guests, the temperature of the household will increase.
4. Some of the appliances (e.g. refrigerators) have an electrical behaviour that is dependent on the temperature of the household as well as the number of times that the appliance is used during the day.

2.4 Related work

Aggregated data at substation level is no longer accurate enough to describe the consumption processes at the level of the distribution grid (Palensky et al. 2008).

Modelling consumption locally can help us to better understand the composition of aggregated load curves (also represented by load profiles) and analyse how local measures can have an effect on the global curve. Current aggregated models do not offer the possibility of modelling local measures on the grid, for example punctual actions taken by individual consumers, such as switching on a cooking stove or an oven. They can only be included by modelling their aggregated effect and integrating it at aggregated level. When considering the proposed changes in the electrical system, for planning and control activities at distribution level, these curves are not suitable, as they only apply to a large number of consumers, as they describe the average use of energy over time (Willis and Scott 2000; Paatero 2009).

Some approaches which consider these facts already exist and implement a demand modelling at lower scale. In Stokes et al. (2004) a simplified demand model for domestic lighting is presented providing an estimation of the aggregated demand or even the distributed loading for groups of households. In Paatero (2009) a multi-use bottom-up domestic consumption modelling tool is developed and verified. Wright and Firth (2007) gathered and analysed high resolution domestic electrical data and found that logging at intervals of few minutes is necessary to capture the fine detail of load patterns for evaluating on-site generation. In Widén and Wäckelgard (2009), a high resolution stochastic approach, using heterogeneous Markov chains is chosen to model domestic electricity demand.

3 Existing simulation platforms

Several simulation platforms have been developed for making experiments using an ABM approach. These platforms have succeeded since they provide standardised software designs and tools allowing the simulation of different models. However, these platforms have well-known limitations. In Railsback et al. (2006), five different platforms are reviewed in next paragraphs: NetLogo, Mason, Repast, Swarm for Objective-C, Swarm for Java. Furthermore, Anylogic and Flame are also reviewed as they are also popular solutions for agent-based simulation.

(a) *NetLogo* (Tisue and Wilensky 2004). This is recommended for its ease of use and excellent documentation, and suitability to develop models that are compatible with its paradigm of short-term, local interaction of agents and a grid environment which is nor extremely complex. Highly recommended as tool for prototyping models that can later on be implemented in lower-level platforms. However, its simplified programming environment may be uncomfortable for experienced programmers, since all the code must be placed in just one file (as opposed to good practices in object-orientation programming) and the lack of a stepwise debugger.

(b) *Mason* (Luke et al. 2004). Good alternative for experienced programmers who work on models that are computationally intensive since it performs well giving the best execution time of the five platforms tested. Things that can be improved are its non-standardised terminology, its incompatible classes with the scheduler, its lack of a terminal window for debugging purposes.

(c) *Objective-C Swarm* (Minar et al. 1996). This version of Swarm is stable providing a fairly complete set of tools, a clear conceptual basis and clever design, allowing for the separation of the model from the interfaces. This tool is oriented to help model organisation by enabling the design and implementation of modules in separated swarms, each one owning objects and schedule of actions. This helps to manage the complexity of the models which is useful when dealing with highly complex systems. On the downside, there is a lack of a friendly development tools, lack of garbage collection, weak error handling and low availability of documentation.

(d) *Java Swarm* (Minar et al. 1996). Actually, this provide the Swarm implementation for Java users but it contains significant drawbacks: clumsy workarounds to implement Swarm features in Java, difficulty in debugging errors that happen on Objective-C libraries and slow execution speed are some of these drawbacks.

(e) *Repast* (Collier 2003). It is certainly the most complete Java platform. Apart from implementing most of the Swarm's functions, they have added capabilities like reset and restart models from the graphical interface and the multi-run experiment manager. Execution speed is good when compared to other platforms. It also provides geographical and network support. However, downsides include the difficulty to getting started on it, especially for novice developers and poor documentation.

(f) *Anylogic* (Borshchev and Filippov 2004). AnyLogic is a multi-paradigm simulator supporting ABM as well as Discrete Event modelling. It gives support for developing flowcharts, System Dynamics and stock-and-flow descriptions. AnyLogic is capable of capturing arbitrary complex logic, intelligent behaviour,

spatial awareness and dynamically changing structures. AnyLogic is object oriented and based on the Java programming language. To a certain degree this ensures compatibility and reusability of the resulting models (Zauner et al. 2007).

(g) *Flame* (Holcombe et al. 2006). FLAME (Flexible Large-scale Agent-based Modelling Environment) is a agent-based modelling framework which allows modellers from various disciplines like economics, biology and social sciences to easily write agent-based models and simulate them on parallel hardware architecture. The environment allows modellers to create agent-based models that can be run in high performance computers and graphical processing units. The simulation code is generated by processing a model definition (Kiran et al. 2010).

After studying these simulation platforms, we have found some limitations. Firstly, there is the lack of an explicit formalisation of the semantic. We have analysed the code of many simulations where the same concept has been represented in different ways. For example, freezers can be represented considering either isolation, efficiency, or thermodynamic processes. Any of these allows the calculation of consumption. Therefore, developers are producing a code that cannot be combined further into a single model, since there is not a formal definition of a "freezer". That is to say, every model has its own semantic.

The semantic of any concept must be shared among all developers. This means that it should be agreed upon and formalised. The development platforms should facilitate the explicit formalisation of the problem domain's semantic.

Another limitation is the lack of support for developing large scale models. It is possible to build large scale models in these platforms, but the development process could become as complex as the models themselves. Since large scale models own a high complexity, the developed software can be also very complex. These tools are not providing a methodology that supports the development process on large scale models. That is, we find there is a lack of architectural support for developing large-scale models.

Furthermore, since these platforms are oriented to multi-purpose agent-based simulations they provide a language that is far from the problem domain. This involves a semantic gap between platform and modellers concepts. Ideally, modellers would be more comfortable expressing their models in a language closer to the problem domain.

4 Building complex electrical system simulations

It is necessary to approach the software engineering of the simulator with a methodology that supports the modelling of a large number of entities. The goal is to make simulator development and maintenance easier. In this paper, we propose the development of simulators using a Model Driven Engineering approach (MDE).

MDE (Schmidt 2006; Butler et al. 2002; Poole 2001) allows the development of software based on models. This is, the functionality of the software is defined in models and, by using a processor, the software is generated according to these models. The main advantage of this approach is that the software development at the model level is conceptually described using a DSL (Domain Specific Language) that is close to the domain problem.

Not only developers, but also domain experts, may have an important role in software development. In this way, communication problems between domain experts and software developers are mitigated. Models are the artefacts that drive the development process. From an methodological point of view, development methods have been trying to include more abstractions in order to reduce the semantic gap between software design and domain concepts.

In addition, this approach improves productivity and flexibility in developing simulators, since they are developed and modified just by changing the model (Schmidt 2006). Thus, MDE is useful to speed up the creation of simulators and enhance their evolution.

The application of MDE for developing complex system simulations is the most relevant contribution of this work. Several abstraction levels have been defined between the software design and the implementation. The lowest abstraction level is related to the description of instances that will exist in the simulation scenario. For example, at this level, different instances of buildings, power lines and customers can be specified in a power grid model description (Fig. 1).

Since these instances can be abstracted, the second level, known as Metamodel, represents the different classes of instances that can be found in a domain. For example, building, power line and customer concepts are included into the Metamodel to allow future specifications of concrete instances (Fig. 1).

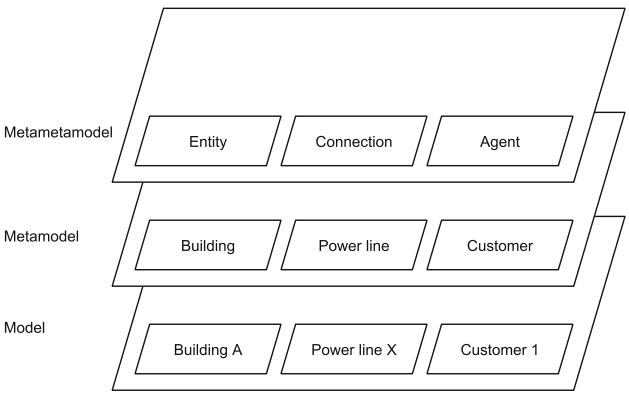

Fig. 1 Abstraction levels considered in the case of the power grid

Finally, the third level of abstraction, the metametamodel, allows the representation of any complex system using three different metaclasses: entities, connections and agents. For example, a building is an entity, a power line is a connection and a customer is an agent of the power grid system (Fig. 1).

5 Tafat framework

A framework called Tafat has been designed and developed at the Research Institute SIANI of the University of Las Palmas de Gran Canaria to support the development of complex system simulations based on MDE. Tafat uses an object oriented and agent-based approach.

Tafat allow building models of complex systems where the overall scene is decomposed into different components. Each component of the model is statically represented by means of attributes and variables. In this way, a structural view of the system is modelled. In order to model the behavioural view of the system, each component may include several behaviours that describe how this component changes over time. That is, the dynamics of the system.

This separation between structural and behavioural view is a major design concern in Tafat since it allows modelling a complex system in a modular approach. At the same time, this design technique is oriented to face the challenge of building large scale simulations. Basically, when a simulation is running, all the behaviours are concurrently executing and modifying the variables of their components.

A major contribution of Tafat is that it allows the definition of a Metamodel. This Metamodel supports the analysis, identification and description of system components. In this way, the Metamodel is shared with all simulation models, since the identified components come from the Metamodel. In this sense, the Metamodel can be understood as a Domain Specific Language. As well, this lays the foundation for a specific analytical framework according to the domain of the complex system.

Thus, models only include component types that have been already defined in the Metamodel. This means that all models share the same representation classes as well as the same semantic. This is a necessary condition to allow the integration of simulation models. Therefore, it does not matter the tool or language (e.g. Repast or Java) in which a behaviour has been developed as long as behaviours are associated to Metamodel component classes.

5.1 Architecture

As mentioned before, the metamodel is a core component of Tafat architecture (Fig. 2). The Metamodel provides an object oriented representation of the system. Systems are

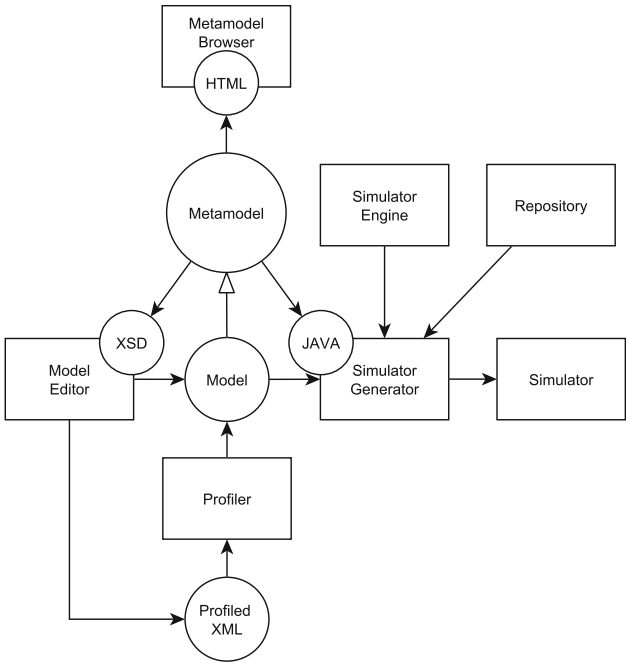

Fig. 2 Tafat architecture

represented in terms of entities, connections, agents, attributes, variables and context, amongst others. The Metamodel is oriented to a specific domain and defines the types of components that can be instantiated in the model. It determines how the semantic is shared amongst different models.

Besides, Tafat includes different tools that support the simulator development process. Checker is a tool to validate the model syntactically and semantically; Profiler is a tool to assist the automatic construction of large scale simulation models; and Simulator Generator is tool that can automatically create a new simulator for this model.

The Simulator Generator parses the model in order to automatically generate the required Java classes and objects. The simulator is completed by including the Simulator Engine and all the behaviours that the simulation requires.

Behaviours are stored in the behaviour Repository and can be used in different simulations. A behaviour describes the dynamic of a model component that can be used in different simulations.

Therefore, model components are implemented by composing two different views: structural, contained in the Metamodel, and behavioural, contained in the repository. The structural view concerns the description of the components in terms of attributes, variables and context. The behavioural view describes how states evolve and their interaction with other components.

This separation of concerns enhances the possibility of customising already developed behaviours or, even, creating new ones and plugging them into a model component.

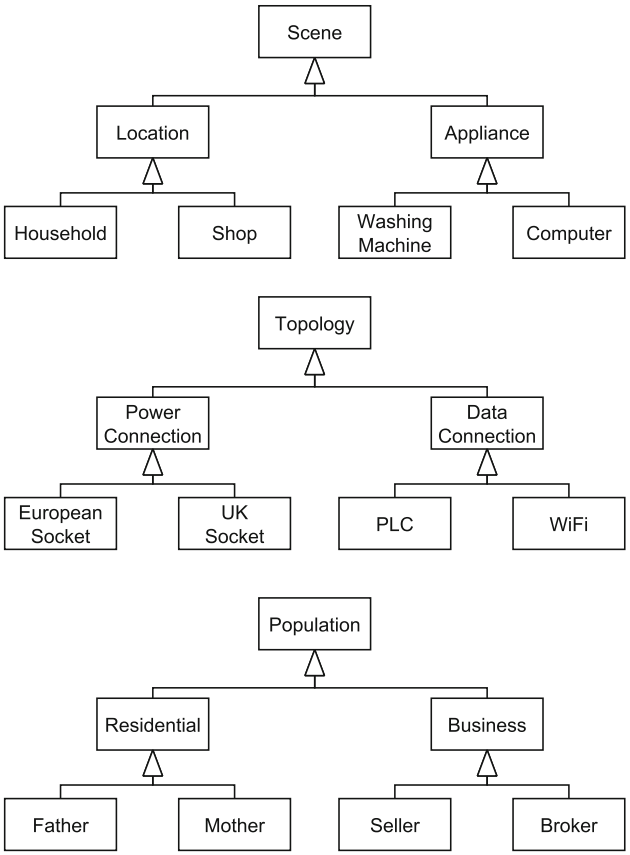

Fig. 3 A Metamodel example for a power grid. In the scene, entities that are related to power grids can be found. The topology contains connections. The population contains agents

5.2 Metamodel

The Metamodel constitutes the core of the framework architecture since it defines the metaclasses of components that can be simulated. A metaclass defines the structural composition of a component, which may include other components within. In this way, a model is described by decomposing high level components into low level ones.

In other words, the Metamodel is a representation that describes how the model can be structured. However, there are different layers in the Metamodel that have been defined by means of an abstraction process over several complex system models. These layers constitute the basis for defining the metaclasses included in the Metamodel:

- Entities. Objects contained in the model scenario (e.g. understanding the power grid as a complex system, a washing machine would be a component) (Wooldridge 2009).
- Agents. Active objects (intelligent or not) that are able to interact with entities or other agents (e.g. people living in the household where the washing machine is located).
- Connections. Components that link entities (e.g. the washing machine's power connection to the supply).

Metaclasses are classified into the Metamodel according to these categories (Fig. 3). This classification is useful since entities are used to model the scenario, agents to model the population and connections to model the topology. In addition, there are other categories within these major ones that have been defined according to the nature of the system.

Regarding the components description, there are several aspects that can be used. The list below presents the different perspectives from which an object can be described:

- Features. Attributes of a component which do not change their values during the simulation (E.G. capacity of a washing machine).
- Variables. Attributes of a component which change their values during the simulation (E.G. power of a washing machine).
- Contains. Set of components that can be placed within a component (E.G. a household can contain several appliances).
- Context. Components that influence the component that is being defined (E.G. as a radiator influences the internal temperature of a household, the radiator is considered to be in the context of the household).

The example below presents the static description of a household. This household description has one feature, one variable, two contains and one context. As a feature, the height of the household is described. The personCount variable indicates the number of people who are inside the household at a given time. This is variable since people leaving and entering the house would affect this value. The household can contain a powerMeters and appliances. A list of these appliances is part of its context since this list will be used by the powerMeters to calculate the overall consumption.

```
<class name="Household" parent ="Location">
  <feature name="height" type="double"
   unit="meter" default-value="3"/>
  <variable name="personCount" type="int"/>
  <contain name="powerMeter" type="PowerMeter"/>
  <contain name="appliance" type="Appliance"/>
  <context name="applianceList"
   type="Appliance" replicated="true"/>
</class>
```

Listing 1.1 Metamodel entity example of a Household

5.3 Model

The simulation scenario is expressed through the simulation model. This model contains a set of instances of the

components that are in the scenario. The next example presents a simple scenario where a household containing an appliance is represented. Furthermore, there is agent which is linked to this household.

```
<simulation>
  <scene>
    <outdoorEnvironment>
      <building>
        <household id="h0" height="2.5">
          <washingMachine>
            <behaviour name="WashingMachinebehaviour"
             release="Operational"/>
          </washingMachine>
        </household>
      </building>
    </outdoorEnvironment>
  </scene>
  <population>
    <family linkedTo="h0">
      <behaviour name="FamilyBehaviour"
       release="CoupleWithChildren"/>
    </family>
  </population>
</simulation>
```

Listing 1.2 Model example

5.4 Tools

Within the architecture of Tafat, there are different tools which assist different processes that are part of the development of the experiments. The list below briefly explains what these tools are for:

- Model editor. This tool is part of the architecture, but it is not an exclusive development of the framework. This functionality may be transferred to any model editor which supports XSD (XML schema) files for writing simulation models.
- Profiler. A tool for creating models in an automated manner especially oriented to develop huge scenarios based on statistical or incomplete data. The construction of large-scale models must be automatized. For instance, the model of a huge power grid in which the entire demand is represented in a disaggregated manner cannot be manually built. Furthermore, all required data to represent this scenario is usually not available at such a level of disaggregation. The Profiler allows the integration of several sources of data which are used to build large-scale simulation scenarios.
- Repository. Storage containing behavioural aspects of the components described in the metamodel.
- Simulator Generator. Based on a simulation model, this tool compiles all needed Java classes from the Metamodel translation and the instantiated behaviours from the repository. All this compilation is integrated with the engine generating a simulator which simulates the input model.
- Simulator Engine. This tool parses the model, instantiates Java Classes and runs the simulation. This tool can export results into different formats so that a posterior analysis can be performed.
- Metamodel browser. As well as the model editor, this tool is not a framework development. This functionality is provided by any web browser since the Metamodel is translated into html format.

5.5 Programming behaviours

To simulate the electrical system, all the metaclasses have to include programmed behaviours. In this section, an example of each type of behaviour will be described: environmental, device and social.

5.5.1 Environmental behaviour

Environmental behaviours represent the change over time of some environmental variables. Environmental variables are normally common to a group of entities or devices and describe the environment.

These can be, for example: solar radiation models, which can be used for calculation of the thermal gains of a building or, additionally, for energy production (photovoltaics, solar-thermal use, etc.). These behaviours do not directly change the attributes of an appliance or agents (such as human behaviour), but rather allow some interactions in an indirect way (e.g. through heat exchanges, etc.).

5.5.2 Device behaviour

Most of the electrical appliances used in a household are major appliances such as washing machines, refrigerators, etc. There are also some other, smaller appliances, such as CD players, televisions, HiFi Audio equipment, etc. Usually, the major appliances are responsible for the larger part of the electrical consumption. In order to recreate the individual load curves, EIFER (European Institute for Energy Research, a common research institute by KIT and EDF) has developed individual models for the behaviours of electrical appliances, which were integrated into Tafat. Simplified technical models are used, which take into consideration different technical parameters of a specific appliance. So, for example, the load curve of a TV will be characterised by its size and technology (CRT, LCD, Plasma, etc.). Major appliances also are modelled using the EU Energy Label as an input parameter, which is an indicator for the energy consumption of a device and is compulsory for appliances sold in the EU.

Different releases for the behaviours of the electrical devices were created. Using this modular approach, a

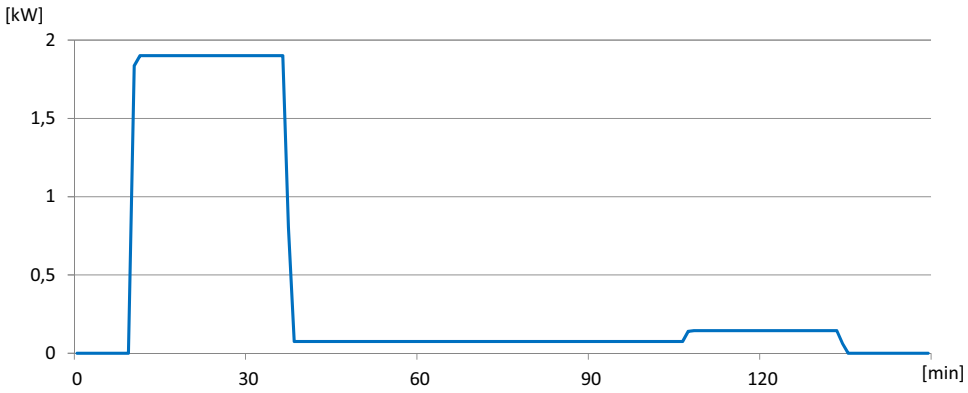

Fig. 4 Simulated load curve of a washing machine

behaviour of a single appliance can be *exchanged* in a simple manner. The different releases include simplified technical models with varying degrees of accuracy, thus allowing for an optimisation of execution time vs. the accuracy of the model. In Fig. 4 an example of the load curve generated by the behaviour of a washing machine can be seen. This load curve is created by a simplified technical model of this appliance.

5.5.3 Social behaviour

A flexible architecture is proposed to carry out a social behaviour (Fig. 5). Intentional stances are the most complex behaviours (Dennett 1987). For this reason, the architecture must be flexible to allow a range of behaviours from simple behaviour based on a list of tasks to a complex behaviour implemented as a neural network.

The mission maker is the intention launcher, the decision maker is in charge of choosing a recipe to accomplish the mission launched and the action maker is the executor of the recipe. The recipe is a list of actions that accomplishes a mission. With this architecture, a simple behaviour can be developed by creating a big recipe in which all the tasks are described and having a very simple mission and decision maker. Otherwise, in a complex social behaviour, the task can be launched by the mission maker according to several parameters of its own agent or the environment, which entails a more difficult process in choosing the recipe, but simpler recipes which only describe how to arrange a task as, for example eat.

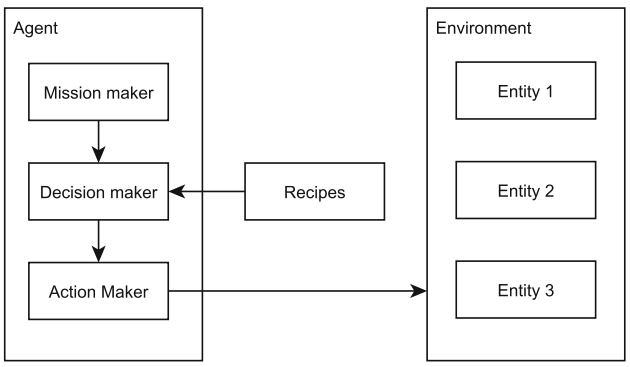

Fig. 5 Agent architecture composed of a mission maker (intentions), decision maker (recipe selector) and action maker (recipe executor)

5.6 Simulation

The main problem in developing good models that accurately represents a place (town) is the lack of data. Often, it is quite difficult to gather the needed data.

Ideally, models should be built using real data, since the simulation will help to understand what happens in the electrical system. However, when no data is available, a model approximation is done. The previously mentioned tool Profiler helps to carry out this task. Using a high-level description of a place (for example, the number of buildings and the population population), Profiler automatically generates an electrical system model that can be simulated directly. The profiler is part of Tafat, a MDE platform that supports the development of simulations.

Electrical models can be created to represent the load accurately but are light-weight enough for use in large scale simulations, and handle demand side management mechanisms through the use of an agent-based approach.

6 Experimental evaluation

This experiment is validating the MDE approach used in Tafat. This case study shows the feasibility of Tafat and its mechanisms for managing the complexity of huge systems.

The scenario used for validating the approach is conceptually divided into three different layers which represent different realities (Fig. 6). In the base layer, known as the territorial layer, facilities are placed.

The second layer, known as the power grid, includes, over this territorial layer, a set of components that may be

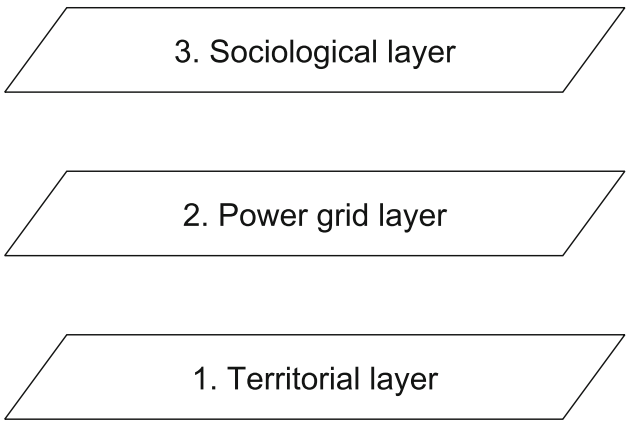

Fig. 6 Case study layers. The power grid devices are located over the territorial layer. These power grid devices are controlled by the people who are in the sociological layer. Note that the numbering of the layers corresponds to the order of development

placed inside facilities. These components are electrical devices that may be either power consumers or power producers. In this case study, the focus is made on the residential sector so that these electrical devices are mainly common appliances (e.g. refrigerators, radiators, televisions, etc.). Furthermore, distributed photovoltaic cells may be placed in the home so that the residential sector not only consumes, but also produces energy.

The third layer is the sociological one and involves the interaction of the people living in the households (agents) with the electrical appliances. Based on this interaction, the consumption of the appliances inside the household is calculated when people switch them on.

6.1 Territorial layer

This layer is modelled by locating facilities in a territory. These facilities may be buildings, roads, pipes or networks (e.g. electrical, communication), among others. In this case study, concepts as buildings and households are placed in this layer.

Buildings are the facilities that support the photovoltaic installations (which are defined in the power grid layer) and serve as households, which are part of the facility layer.

Households contain power grid appliances that used by the simulation agents. These agents, sociological agents that represent the behaviour of the people living in the household, are defined in the sociological layer and, then, related to households. This relation will define the action area in which they can switch devices on and off.

Therefore, it is necessary to create two new components in the Metamodel: "building" and "household". In this concrete case, "building" does not contain any static information. However, households are described by their area. This information is used to calculate the needs for heating in the power grid layer in terms of installed power for heating. An example of a simple model of this layer is described below:

- Building B1
  - Household H1

    area = 50 m^2
  - Household H2

    area = 45 m^2
- Building B2
  - ...

6.2 Power grid layer

At this point, the Metamodel must be altered to enable the introduction of electrical devices in the territory. To this end, a new concept must be modelled: the electrical device. Now, the household description of the Metamodel must be modified in order to include power devices inside. Furthermore, the description of the building must be changed to allow for the introduction of photovoltaic cells.

Apart from these modifications, it is necessary to develop models for all the different devices. This development involves not only the static model design but also the dynamics. In Fig. 7, the hierarchy of the electrical devices is shown.

Concerning the static description of the electrical devices, the parent Metamodel class *Electrical device* is described by the operational mode of the device (ON, STAND BY, OFF) and the active power that the device is consuming. These properties are then inherited so that all the electrical devices have these variables. Since each kind of electrical device has a specific way of working, a behaviour must be developed for each one. These behaviours are not developed using the meta language describes the Metamodel. In this case, they are developed using Java. An example of a model including this layer within the facilities is presented below:

- Building B1
  - Household H1

    Refrigerator R1

    mode = ON
    behaviour = RefrigeratorBehaviour

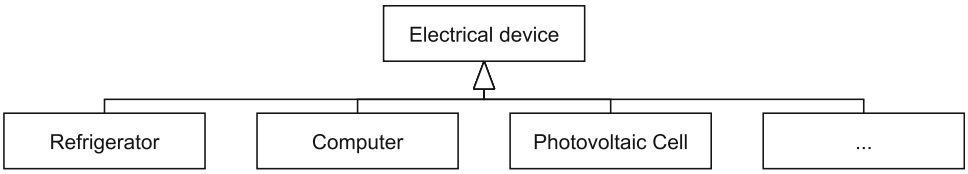

Fig. 7 Hierarchy of the electrical devices within the Metamodel

6.3 Sociological layer

The sociological layer is designed to allow the analysis of the electrical load of households according to social group characteristics. The following five different social groups have been taken into account to develop the case study:

1. younger single people,
2. older single,
3. younger single,
4. younger couple and
5. family with children.

These groups represent about 70 % of the population of Germany, being the most numerous groups identified by the German Socio-Economic Panel Study (SOEP). The constitution of this sample is shown in Fig. 8.

The survey[1] here provides the information about the timetables and appliance usage of a household according to social group. Based on this information the agent behaviour related to the household is implemented. The electrical usage behaviour data employed for this study was obtained through a local survey. Thus the data gathered from a small sampling was used as input parameter in the Tafat model. Hence, 20 different types of social behaviours are used to simulate the five socio-demographic groups. For example, cooking the dinner is at 20 h in a specific social behaviour. Obviously, not all the households start cooking at 20. Thus, a normal statistical distribution is defined, where the starting time mean is at 20 with a deviation of 15 min.

In arranging this study, a Tafat tool that automatically builds a scene has been used. This tool uses statistical data from the SOEP and the survey to create a model scene in which the social groups are distributed in the households. These households contain the electrical appliances, and their number, depending on the social group.

The introduction of these agents into the Metamodel involves a distribution of the components within the Metamodel among components that are placed in a scenario and components that are part of a population. For this specific case study, an agent is defined by its household. An example of a model is presented below:

- scene
  - Building B1

    Household H1

    ...
- population
  - SociologicalAgent A1

    householdId = H1

6.4 Simulation and results

In this simulation, the device load curves within a household are generated. The curves were aggregated using individual behaviours for each household, taking into account some deviation from the mean (variation of the duration and usage time for different electrical appliances).

The simulation results show the load curve for a day of 1,000 households with around 12 appliances in each, thus composing approximately 12,000 simulation components. This type of simulation can provide the relevance of a specific power consumer in the household as well as the influence of a specific type of consumer on the global load. The number of 1,000 entities was chosen, because it has been found that larger numbers of units do not change the results significantly, but only increase the simulation execution time for the simulation. This consistency of results is probably due to the use of a limited number of recipes (taken from the number of people surveys). The execution time for a 24-h simulation period was around 20 minutes on a standard desktop PC.

The 1,000 household sample contains a distribution of the different social groups according to real statistical data, in order to obtain a sample of households as close as possible to reality. In Fig. 9, the simulated load curve for one day can be seen. The simulation is run in a high time resolution, with a time step of one minute. This allows observing effects which are neglected in simulations at lower resolutions, in 15 minutes or one-hour models. Some sharp peaks can be observed, which are caused by the use of high power consumption devices in the household. A general trend to use more power during the daytime is

[1] The survey consisted in local interviews with around 20 people in Karlsruhe, Germany, in order to obtain data such as usage time and duration amongst specific socio-demographic groups. It should be noted that the survey is not representative, but rather a sample of the user behaviours of those groups.

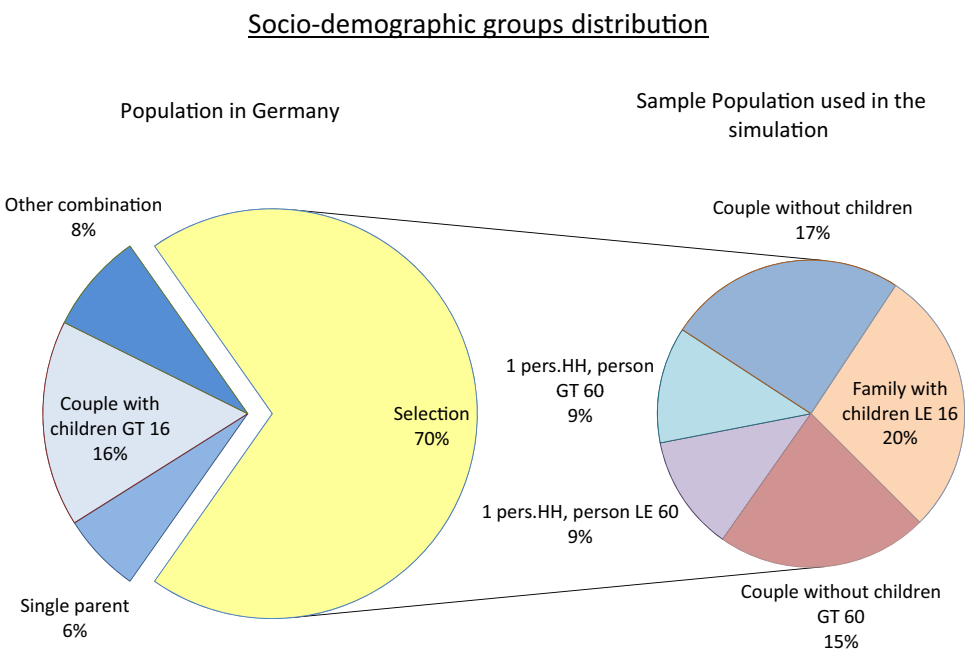

Fig. 8 Socio-demographic groups used in the case study. *Source* SOEP

Fig. 9 Simulated load curve of 1,000 households for one day

clearly visible. At night, the base load (devices that are constantly running, such as refrigerators and other permanent loads) cause a consumption that is only around one third of the daily peak load. The parameterisation of the simulation can be seen in Fig. 10.

Basically, the Profiler works based on a household template. This template has different parameters as it is shown in Fig. 10. The scenario model is automatically built based on this template and a household database which contains all the information available: household reference, location and area. Notwithstanding, the database does not contain all the parameters defined in the template. These parameters (e.g. number of computers) are calculated according to statistical distributions for each social group. In addition, the social group is adjusted based on different statistical distributions considering the household area and location.

It is necessary to validate simulation results. However, it is not possible to compare the simulation data to real data at a disaggregated level. Real disaggregated data is not

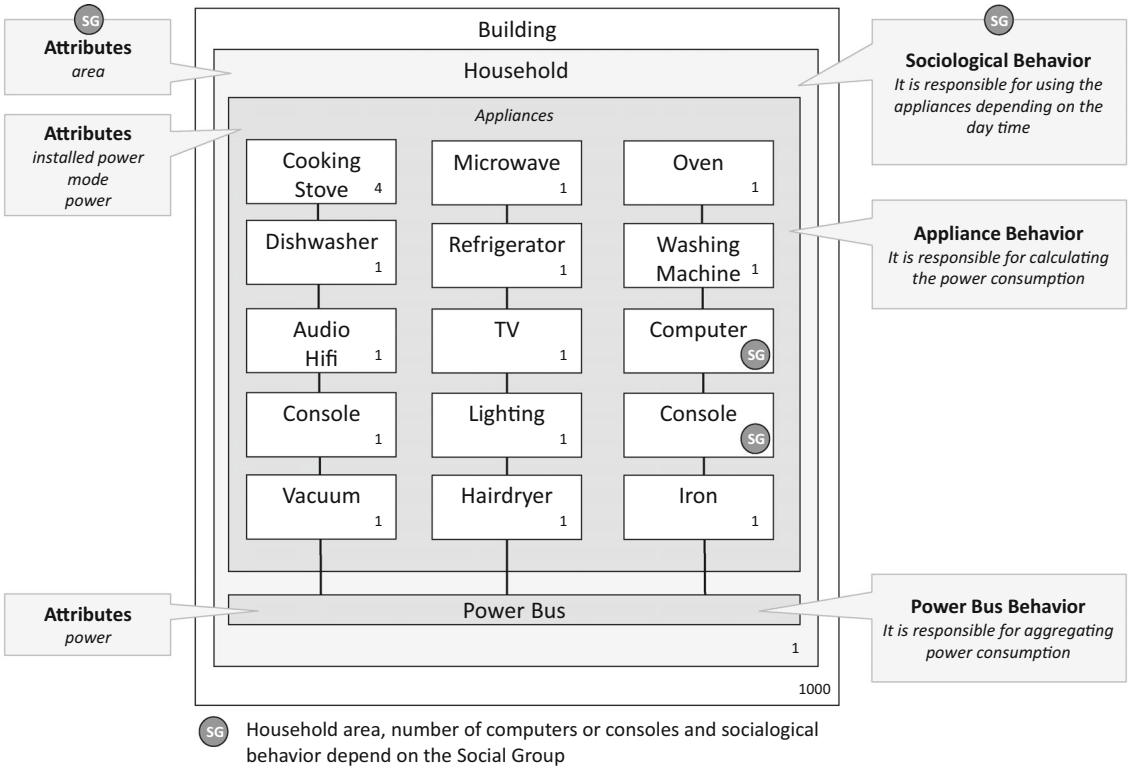

Fig. 10 Profiler pattern for creating 1,000 households. The Social Group (SG) is randomly selected for each household using a frequency distribution

available to the public, the only available is are at an aggregated level RWE (Rhein-Ruhr Verteilnetz 2011). Therefore, the simulation is validated comparing real data to simulated results at an aggregated level.

In Fig. 11 the simulation load curve is compared to a real load curve in Germany for a winter weekday [according to Bundesverband der Energie- und Wasserwirtschaft (2011) and reported by RWE (Rhein-Ruhr Verteilnetz 2011)] for household demand. This curve is a profile in a normalized form. This profile can be scaled to a given amount of energy. In this case, and for comparison purposes, the profile was weighted with the same amount of energy as the simulation load curve, i.e. that the daily energy consumed [kWh] is the same in both cases. The real load curve is smooth, since it represents an aggregated load behaviour at high levels of the electricity system for large number of consumers. They are the result of a statistical analysis based on representative samples from different consumer groups (Palensky et al. 2008).

The simulated curve represents the total power consumed by a sample of 1,000 households, modelled individually and with an autonomous behaviour for each of them. The curve has been built by averaging periods of 15 minutes in order to match the same time granularity as given in the standard load profile. The curve is more peak shaped, which is consistent with the relatively small amount of households (in comparison to the statistical samples taken to obtain a profile, which are representative) and the reduced number of types of behaviour (in total, only 20 different types of behaviour have been used). Furthermore, only 70 % of the household population is modelled, neglecting other social groups which may change the curve.

Even though the selected samples in the survey are not representative for all Germany nor German society as a whole (only five social groups were used), the general trends of both curves are similar. Three peaks can be observed, which are closely synchronized in time and correspond to morning, noon and evening peaks. These peaks are correlated with a large and concurrent usage of high power devices, such as cooking plates, ovens, microwaves, etc. due to eating habits, as well as lighting use in the evening. The morning and noon peaks are lower in the simulation than in the profile, whereas the evening peak is higher. The time synchronization of the ramps of the peaks matches quite well; this indicates that the activities (having breakfast, lunch, returning home, etc.) were modelled according to the average German user behaviour. Even though, some differences can be observed in the evening drop (21–23 h), as well as a small second peak that is not seen in the profile.

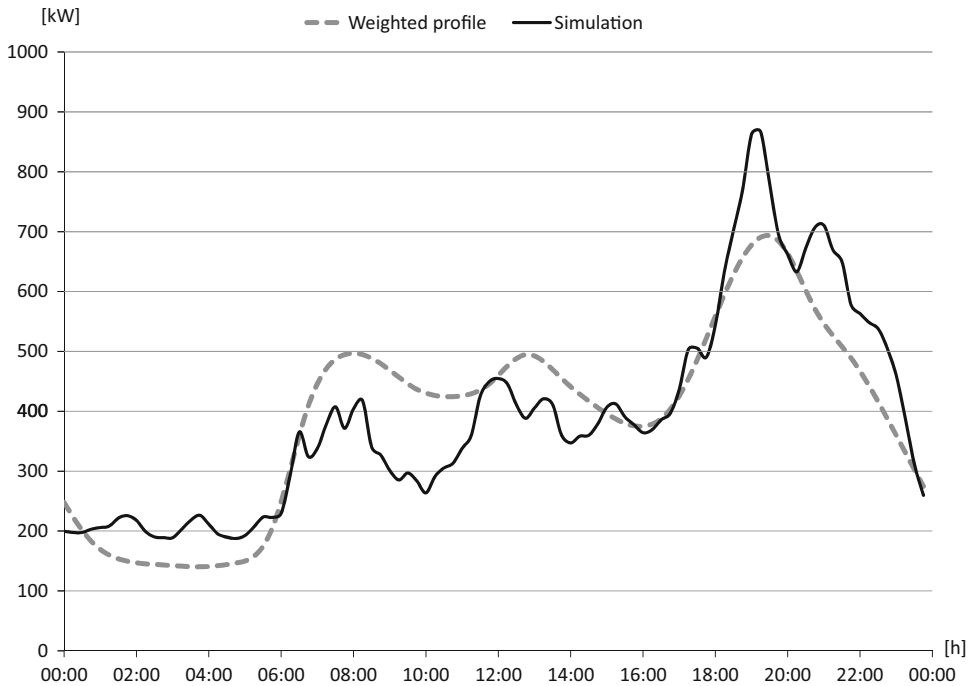

Fig. 11 Simulated load curve vs. standardized residential load profile

6.5 Discussion

Even using a relatively small sample of households and reduced number of types of behaviour, a curve that represents the major characteristics (peaks and troughs, as well as their timely synchronization) is generated. Concerning the differences in the height of the peaks, the model should be reviewed in order to check the individual power curves of each electrical device. This seems to be quite as almost no data is available for such a validation (at a representative sample). However, the differences could also be related to the use of only 70 % of the socio-demographic population share. Further, behaviour itself is another factor to consider, as a largely simplified and almost static model was used.

Some specific characteristics of the model create peaks, which could be explained by a rather homogeneous behaviour of the groups. For example the second evening peak (around 21:30 h) might be caused by some activities (watching TV, other evening activities) which start and end at similar times, because of the relatively small sample. Using data from a more representative survey or a more stochastic-based social behaviour, this could be avoided.

7 Conclusions and outlook

In this paper, a complex model of a Power Grid has been introduced. In engineering, model complexity is increasing, since it is necessary to analyse new proposed designs from different perspectives. In the case of Power Grids, modelling processes should be done using a large amount of detail in order to experiment and study new algorithms and strategies for management. New scenarios, new problems and new challenges will arise in the near future with the introduction of renewable energy sources and a distributed production in the electrical system. Simulations are necessary to design new management approaches.

The case study demonstrates that a bottom-up simulation of residential consumption using an agent-based approach has been successful, as the result curves show similarities to aggregated load curves. A comparison with a national aggregated profile shows similarities in the main characteristics of the curve. Moreover, due to the high resolution of the model, a large number of parameters (individual appliances types and models, socio-technical behaviours, etc.) are available for variation.

The simulation will allow us to further the study of the integration of demand side management strategies. Strategies, such as adaptive or reactive technologies, incentives or campaigns, can be addressed for studying their impact at load curve level. However, social behaviour components need to be validated and improved. This validation is necessary for studying the emergent behaviour and for finding the local actions of components which result in the desired emergent behaviour. Moreover, new social behaviour components should be improved in order to achieve a higher degree of heterogeneity to the models. Due to the high resolution of the household model, individual actions, such as changing specific parameters on an appliance can be performed.

Furthermore, the model developed could be expanded in order to simulate not only the demand side, but also distributed generation or other injections to the grid (like storage). These could interact with the existing components. This is already contemplated within Tafat, and would allow a disaggregated analysis of offer-demand balance as well as estimating the impact of those measures at a grid level.

From the software engineering point of view, Model Driven Engineering has been demonstrated to be a very useful methodology for dealing with the complexity of simulation models. Amongst advantages:

- Fast development of complex system simulations from scratch.
- Formalisation of a Metamodel to describe the component classes to have a shared representation and semantic. Therefore, modellers may integrate their models easily.
- The Metamodel enables the construction of a family of simulators which uses the same component classes and behaviours.
- Separation of concerns: component descriptions are fixed at the Metamodel level while their way of acting (behavioural aspects) are variable allowing to have different releases in which the behaviour of a component may be represented.
- Modular development which can improve the engine without affecting previously developed models.
- A high performance engines whose optimization is constantly being improved.
- A complete set of tools which assist the different processes in creating a new simulator.
- An easy access to the modification of experiments by merely changing the definition of the simulation model.

Acknowledgments This work has been partially supported by European Regional Development Fund (ERDF/FEDER) and Agencia Canaria de Investigacin, Innovacin y Sociedad de la Informacin (ACIISI) of Canary Islands Autonomous Government through the project whose reference is SolSub200801000137, and also through the ACIISI PhD Grant funding to José Évora with reference TESIS20100095.

References

Borshchev A, Filippov A (2004) Anylogicmulti-paradigm simulation for business, engineering and research. In: The 6th IIE annual simulation solutions conference, vol 150

Bundesverband der Energie- und Wasserwirtschaft (2011) BDEW - Standardlastprofile (SLP)

Butler M, Petre L, Sere K, Kent S (2002) Model driven engineering. In: Lecture notes in vomputer science, vol 2335. Springer, Berlin, Heidelberg, pp 286–298. doi:10.1007/3-540-47884-1_16

Capasso A, Grattieri W, Lamedica R, Prudenzi A (1994) A bottom-up approach to residential load modeling. Power Syst IEEE Trans 9(2):957–964

Collier N (2003) Repast: an extensible framework for agent simulation. Univ Chic Soc Sci Res 36:2003

Dennett D (1987) The intentional stance. M.I.T. Press, Cambridge

Evora J, Kremers E, Morales S, Hernandez M, Hernandez JJ, Viejo P (2011) Agent-based modelling of electrical load at household level. In: ECAL 2011: CoSMoS—Proceedings of the 2011 workshop on complex systems modelling and simulation, p 12

Holcombe M, Coakley S, Smallwood R (2006) A general framework for agent-based modelling of complex systems. In: Proceedings of the 2006 European conference on complex systems. European complex systems society Paris, France

Holland JH (1995) Hidden order: how adaptation builds complexity. Addison Wesley Longman Publishing Co.,Inc, Redwood City, CA

Holland JH (2006) Studying complex adaptive systems. J Syst Sci Complex 19(1):1–8

Kiran M, Richmond P, Holcombe M, Chin LS, Worth D, Greenough C (2010) Flame: simulating large populations of agents on parallel hardware architectures. In: Proceedings of the 9th international conference on autonomous agents and multiagent systems: volume 1–Volume 1. International foundation for autonomous agents and multiagent systems, pp 1633–1636

Luke S, Cioffi-Revilla C, Panait L, Sullivan K (2004) Mason: a new multi-agent simulation toolkit. In: Proceedings of the 2004 SwarmFest Workshop, vol 8

Massoud Amin S, Wollenberg BF (2005) Toward a smart grid: power delivery for the 21st century. Power Energy Mag IEEE 3(5):34–41

Minar N, Burkhart R, Langton C, Askenazi M (1996) The swarm simulation system: a toolkit for building multi-agent simulations. Santa Fe Institute Santa Fe

Mitchell WM (1992) Complexity: the emerging science at the edge of order and chaos. Touchstone, New York

Monti A, Ponci F, Benigni A, Liu J (2010) Distributed intelligence for smart grid control. In: Nonsinusoidal currents and compensation (ISNCC), 2010 international school on, pp 46–58

Paatero JV (2009) Computational studies on variable distributed energy systems. Phd thesis, Helsinki University of Technology

Palensky P, Dietrich D (2011) Demand side management: demand response, intelligent energy systems, and smart loads. Ind Inform IEEE Trans 7(3):381–388

Palensky P, Kupzog F, Zaidi AA, Kai Z (2008) Modeling domestic housing loads for demand response. In: Industrial electronics, 2008. IECON 2008. 34th Annual conference of IEEE, pp 2742–2747

Polack FAC, Hoverd T, Sampson AT, Stepney S, Timmis J (2008) Complex systems models: engineering simulations. In: ALife XI, Winchester, UK, August 2008. MIT Press, pp 482–489.

Poole JD (2001) Model-driven architecture: vision, standards and emerging technologies

Railsback SF, Lytinen SL, Jackson SK (2006) Agent-based simulation platforms: review and development recommendations. Simulation 82(9):609–623. http://sim.sagepub.com/content/82/9/609.short

Ramchurn S, Vytelingum P, Rogers A, Jennings N (2011) Agent-based control for decentralised demand side management in the smart grid. Taipei, Taiwan, pp 5–12. http://eprints.ecs.soton.ac.uk/21985/

RWE Rhein-Ruhr Verteilnetz: Lastprofile (2011) http://www.rwe-rhein-ruhr-verteilnetz.com/web/cms/de/201616/rwe-rhein-ruhr-verteilnetz/netzzugang-strom/netzzugang-netznutzung/lastprofile/

Schmidt DC (2006) Guest editor's introduction: model-driven engineering. Computer 39:25–31. doi:10.1109/MC.2006.58

Stepney S, Polack FAC, Turner HR (2006) Engineering emergence. In: Engineering of complex computer systems, 2006. ICECCS 2006. 11th IEEE international conference on, p 9

Stokes M, Rylatt M, Lomas K (2004) A simple model of domestic lighting demand. Energy Build 36(2):103–116

Tisue S, Wilensky U (2004) Netlogo: a simple environment for modeling complexity. In: International conference on complex systems, pp 16–21

Widén J, Wäckelgard E (2009) A high-resolution stochastic model of domestic activity patterns and electricity demand. Appl Energy 87(6):1880–1892

Wildberger AM (1997) Complex adaptive systems: concepts and power industry applications. Control Syst IEEE 17(6):77–88

Willis HL, Scott WG (2000) Distributed power generation: planning and evaluation. Marcel Dekker, New York

Wooldridge MJ (2002) An introduction to multiagent systems. Wiley, Chichester, West Sussex

Wooldridge M (2009) An introduction to multiagent systems—second edition. Wiley and Sons, New York

Wright A, Firth S (2007) The nature of domestic electricity-loads and effects of time averaging on statistics and on-site generation calculations. Appl Energy 84(4):389–403

Zauner G, Leitner D, Breitenecker F (2007) Modeling structural-dynamics systems in modelica/dymola, modelica/mosilab and anylogic. In: EOOLT, pp 99–110