Electrical Power and Energy Systems 75 (2016) 1–7

Contents lists available at ScienceDirect

Electrical Power and Energy Systems

journal homepage: www.elsevier.com/locate/ijepes

Utilizing capabilities of plug in electric vehicles with a new demand response optimization software framework: Okeanos

Wolfgang Lausenhammer [a], Dominik Engel [a], Robert Green [b,*]

[a] Josef Ressel Center for User-Centric Smart Grid Privacy, Security and Control, Salzburg University of Applied Sciences, Salzburg, Austria
[b] Dept. of Computer Science, Bowling Green State University, Bowling Green, OH 43403, United States

ARTICLE INFO

Article history:
Received 29 September 2014
Received in revised form 14 August 2015
Accepted 21 August 2015

Keywords:
Demand response management
Multi-agent systems
Game theory
Plug in electric vehicles

ABSTRACT

Particularly with respect to coordinating power consumption and generation, demand response (DR) is a vital part of the future smart grid. Even though, there are some DR simulation platforms available, none makes use of game theory. This paper proposes Okeanos, a fundamental, game theoretic, Java-based, multi-agent software framework for DR simulation that allows an evaluation of real-world use cases. While initial use cases are based on game theoretic algorithms and focus on consumption devices only, further use cases evaluate the effects of plug in electric vehicles (PEVs). Results with consumers show that the number of involved households does not affect the costs per household. Further evaluation involving PEVs demonstrates that with an increasing penetration of PEVs and feed-in tariffs the costs per household per month decrease.

Introduction

Energy demand in the USA is expected to increase by at least 19%, the supply, in contrast, is only expected to rise by 6% [1]. Furthermore, this energy mismatch is not a US-specific problem [2,3]. While renewable energy could help relieve the load on the grid, it also poses a significant challenge to the grid in terms of keeping supply and demand in balance. With respect to coordination, demand response management (DRM) could pose an ideal solution to this problem [4,5]. DRM refers to "changes in electric usage by end-use customers from their normal consumption patterns in response to changes in the price of electricity over time, or to incentive payments designed to induce lower electricity use at times of high wholesale market prices or when system reliability is jeopardized" [6, 21].

Game theory, in its essence, aims to help understand situations in which several decision-makers interact. Being a mathematical framework and analytical tool, game theory helps study the relationships and actions among rational players. This characteristic renders it an ideal tool to model and understand the inherent complexity of demand response (DR) resulting from this interaction. Publications in this area range from load shifting approaches [7,8] to using storage devices such as PEVs in micro-grid storage games [9] to games that focus on utility companies [10,11]. One thing that these works have in common is a mathematical proof that by optimizing a utility function, a stable point called a Nash equilibrium will be reached [12,13].

This study proposes Okeanos, a novel, game theoretic, Java-based, multi-agent software framework for DR simulation that is capable of investigating the effect of optimizing multiple electric appliances using a game theoretic approach. It is fundamentally different from other DRM software approaches as it plans consumption and production ahead of time. By utilizing game theory, Okeanos benefits from mathematically sound solutions for finding the optimal schedule for household appliances. It supports the simulation of different types of loads and can be configured to work with different game theoretic DRM approaches. The current source has been released as open source[1] and can be used and extended to fit various needs.

While initial results show that savings of up to 6% can be achieved by changing the switch-on time of three household appliances, higher savings can be achieved either by adding more manageable devices to the simulation or by incorporating elective vehicles (EVs) of some sort. In this study, the focus will remain on plugin EVs (PEVs).

The remainder of this paper is structured as follows: The fundamental DR simulation platform, Okeanos, is introduced and key concepts are highlighted in Section "Okeanos"; Results of load

* Corresponding author.
E-mail addresses: wolfgang.lausenhammer@en-trust.at (W. Lausenhammer), dominik.engel@en-trust.at (D. Engel), greenr@bgsu.edu (R. Green).

[1] https://github.com/wolfgang-lausenhammer/Okeanos.

http://dx.doi.org/10.1016/j.ijepes.2015.08.014
0142-0615/

shifting are presented and described in Section "Simulation of multiple households with load-shifting devices"; This is followed by simulations that incorporate PEVs in Section "Evaluation of Okeanos with plug in electric vehicles"; and, finally, Section "Conclusion" concludes this work.

Okeanos

Okeanos is a novel DR simulation platform with a special focus on the inclusion of game theory. Unlike the software presented in [4,14,15], any coordination mechanism that complies with the defined interface is compatible with Okeanos.

Okeanos aims to be a holistic platform for DRM with support for a wide variety of appliances. Through the means of extensibility, new devices can be added by writing a driver for the specific appliance. With OSGi as the foundation, new features can be easily developed, deployed or replaced.

Independent smart household appliances

Similar to other approaches, Okeanos utilizes the multi-agent paradigm to represent household appliances. Thus, with a one-to-one matching between agents and household devices, every device can work towards and set goals or targets on its own. Appliances are proactive and make independent decisions according to the information available to them.

In order not to implement all multi-agent features from scratch, Okeanos builds on JIAC, a feature-rich, modularized and easy to use framework [16]. JIACs modern approach that uses the Spring framework as the basis for the whole system is unique throughout a comparison of multi-agent frameworks including JADE [17], Janus [18] and Jason [19]. Additional evaluation criteria included functionality, active development, ease of use and adoption throughout the software developer community.

In JIAC, the functionality of agents is defined by agent beans. Each bean is a small module with a well-defined responsibility, leading to improved reusability [16]. The energy consumption game described in Section "Coordination mechanism in Okeanos" is an ideal example for this. Its responsibility is to ensure the correct sequential execution of the algorithm. All agents taking part in the schedule optimization process use this bean. Due to the autonomy of agents, it is possible that agents use different games. The meaningfulness of such a mixture, however, is questionable, as no guarantee of the existence of a Nash equilibrium can be given under such circumstances.

The callback functions (cf. Fig. 2) allow for separation of concerns, as the agent itself is still responsible to forward requests to the corresponding components. Similarly, drivers and other services are agent beans as well, ready to be used by agents to support its goals.

Plug in support

OSGi and the Spring framework are two well-known Java frameworks that provide a solid foundation for Okeanos. While both are very powerful tools and offer many features for their respective fields, they share some key concepts, most notably loose coupling and separation of concerns. Naturally, it is beneficial to combine the two and have a module-based, service oriented system as the platform Okeanos runs on, using Spring for the wiring of the components. Eclipse Gemini Blueprint provides a clean and easy to use interface for integrating the two frameworks.

However, to be able to fully utilize benefits of loose coupling, thorough planning is required. Device drivers are the perfect example for the need for extensibility. A flexible and powerful interface eases the interaction with new implementations and the integration of new modules into the system. This is crucial to be able to keep the threshold for developing new modules as low as possible.

With Okeanos built on OSGi, it comprises a conglomerate of various bundles (see Fig. 1) rather than a monolithic core. To allow for optional bundles, the OSGi R5 specification [21] recommends separating interfaces from the implementation in a separate bundle. Consider, for example, a logging service: The application does not necessarily need an implementation for a correct execution, however, at least the interface needs to be present to allow for proper resolution.

As indicated in Fig. 1, every service in Okeanos could be represented in its own module. While, this is possible, it also implies an explosion of projects and, therefore, an increase in complexity. Therefore, layers serve as the boundaries for modules in Okeanos. As recommended, the interfaces of each layer are separated from the implementation and consolidated in different bundles.

Likewise, as it is possible to have no implementation in an OSGi container, it is possible to have multiple implementations present. This is especially true for device drivers, as they all implement the same interface. To be able to distinguish between drivers, additional properties, such as year and brand of a household device, can be specified.

Fig. 1 shows the logic separation between the supporting libraries in the infrastructure bundles area and the application bundles that provide the actual functionality. The Spring extender bundle that is part of the Eclipse Gemini Blueprint project is responsible for activating all Spring powered application bundles and starting up their Spring contexts. This is similar to a J2EE environment, where the Spring application context is started by the application server, whereas here, the extender bundle is responsible for starting all application contexts.

Every such bundle has its own independent context that can import and export services by using special tags[2] in the `context.xml` file. The exported services are regular Spring beans that are registered in the OSGi service registry and, thus, made available to other contexts. For imported services, respectively, Gemini Blueprint searches for a suitable match in the OSGi service registry, fetches it and makes it available to the context.

Coordination mechanism in Okeanos

The requirement of game theory that players have to act rationally is ensured by representing every player by its own agent. Players in this context are household appliances as described earlier.

While there are a number of published game theoretic approaches to DR management [7–11], the game proposed by Mohsenian-Rad and co-authors [7] was modeled with Okeanos as a first proof of concept. Reasons for this include that the algorithm was formulated in pseudo code, which allows for accurate adaptation. Further, potentially more devices can be integrated in the first place by utilizing load shifting as if it were possible with storage devices due to the lack of available data.

The decentralized objective function in [7] with $x_{n,a}^h$ as the energy consumption of a scheduled appliance a of user n at hour h is given by

$$\begin{aligned} &\underset{x_n \in \mathscr{X}_n}{\text{minimize}} && \sum_{h=1}^{H} C_h\left(\sum_{a\in\mathscr{A}_n} x_{n,a}^h + \sum_{m\in\mathscr{N}\setminus\{n\}} l_m^h\right) \\ &\text{subject to} && l_m^h = \sum_{a\in\mathscr{A}_n} x_{m,a}^h \qquad h \in \mathscr{H} \end{aligned} \tag{1}$$

[2] For detailed instructions on the exact syntax see [22].

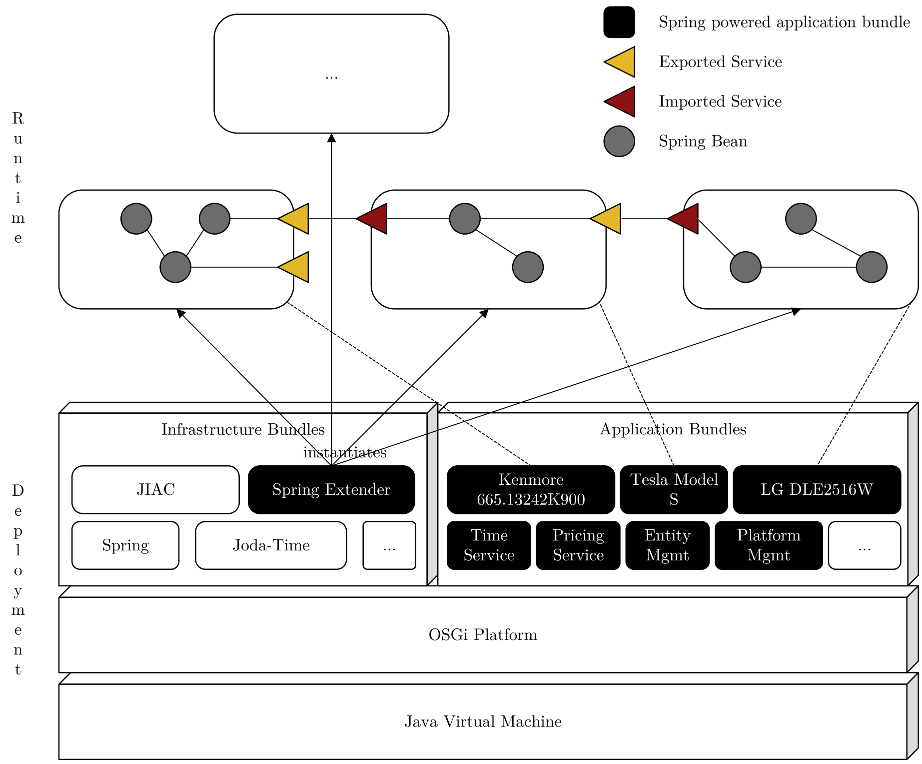

Fig. 1. Okeanos bundle structure with sample household devices and services. The indicated dependencies are only for illustrative purposes. Drivers do not depend on each other in Okeanos. Adapted from [20].

with a set of cost functions C_h that are increasing and strictly convex [7].

Every appliance only needs to optimize its own schedule $x^h_{n,a}$, because the consumption of all other players l^h_m, $m \in \mathcal{N} \setminus \{n\}$ is static. For debugging reasons and the sake of comprehensibility, Okeanos uses particle swarm optimization (PSO).

As denoted in Fig. 2, the algorithm proposed in [7] is started by the agent every time a new schedule is needed. Okeanos adopts the suggested 24 h planning horizon, which requires the agent to initiate it once a day.

The next step is to minimize the costs, i.e., solve the objective function (1). To be able to do that, the necessary information needs to be obtained first. The game has no knowledge, which device is used, therefore, it asks the agent. It knows about the configuration, obtains the information from the driver and returns it. Because the agent is the broker, it could also decide to alter this information. That is, stricter time frames could be set or it could remove itself completely from the schedule.

With that information in its memory, a configuration object of the local device and the most current information of all other devices is assembled. Subsequently, the agent is asked to optimize the configuration. Again, due to re-usability only the agent has knowledge about which optimization algorithm, game and drivers are used. The agent forwards the request to the optimization algorithm, e.g., PSO, which then returns the optimized schedule to the agent and, finally, to the energy consumption game.

The agent is then asked to approve the schedule before continuing. At that point, the algorithm proceeds by checking whether the optimized schedule has changed since the last announcement. If so, it broadcasts the new schedule to other agents. If not, Okeanos sets a timeout after which the agent assumes that no new schedules will be announced anymore.

If a new schedule is received within this time, the timeout is reset and the process starts again, as denoted by the loop-box in Fig. 2. This is repeated until no new schedules are received anymore and the timeout finally expires.

Once the timeout has expired, the equilibrium is reached and the agent informed about it by calling a callback function with the final schedule for the local device and the sum of the final schedules of all other devices.

Optimization algorithm

The optimization algorithm tries to find solutions to (1). Currently, only two different implementations of PSO are available. PSO belongs to the category of swarm algorithms and is loosely inspired by bird flocks or fish schools as first presented by Eberhart and Kennedy in 1995 [23].

The two implementations only differ in the solution space: The first implementation *PSORegulableLoadOptimizer* covers load shifting, while the second, *PSORegenerativeLoadOptimizer*, also handles charging and discharging of PEVs. Therefore, for load shifting, the velocity, as it represents a change relative to the current position, is represented as a vector of time differences in 15 min steps. This resolution tries to strike a balance between an optimal solution, which requires a higher resolution and a good solution, which can be calculated considerably faster. The position comprises the start times a device runs. That is, the position of a washing machine that has to run twice a day would be represented by a vector that comprises two values: the start time of the first run and the start time of the second run.

For charging and discharging of PEVs, Okeanos takes a different approach using *PSORegenerativeLoadOptimizer*. Not the start time is relevant here, but the amount of power charged or discharged at every 15 min interval is important. This is also exactly what a particle's position comprises. The velocity is a vector containing the change of charge for every interval. Optimizing regenerative loads like PEVs is more challenging, as a maximum capacity, minimum

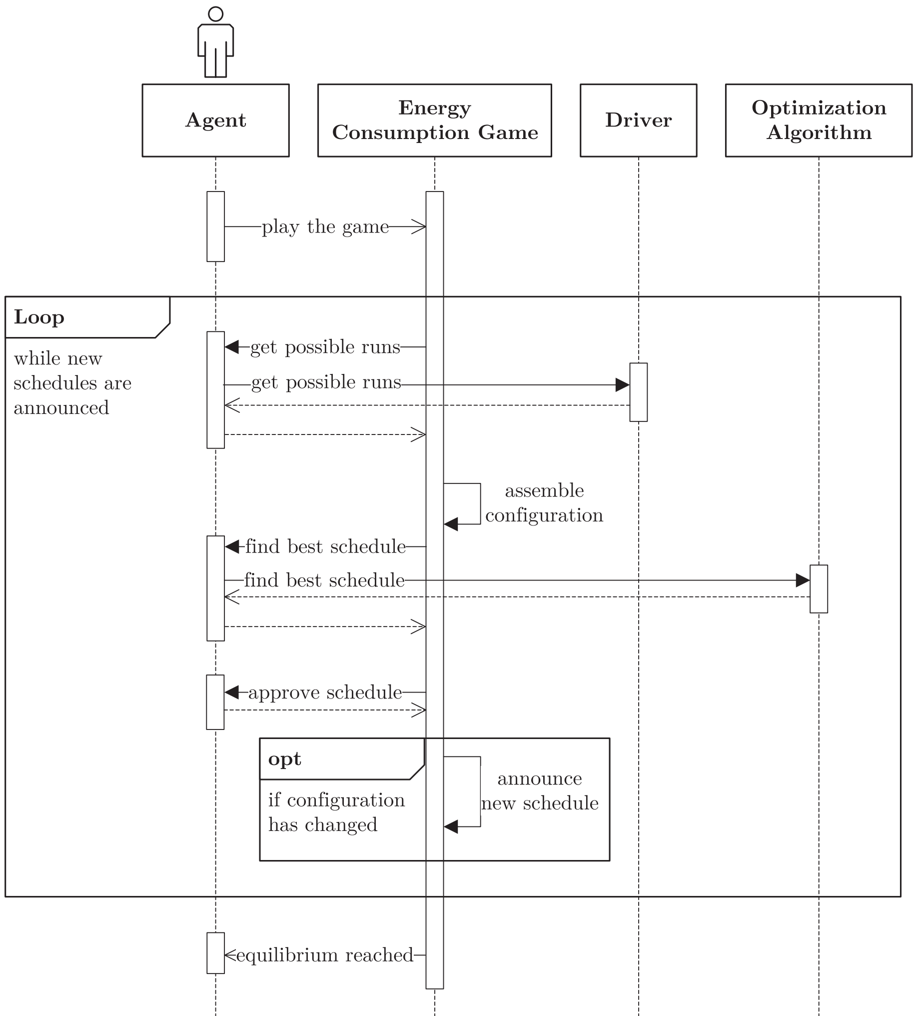

Fig. 2. Process flow of energy consumption game.

capacity and maximum charge per slot need to be taken into account too.

Simulation of multiple households with load-shifting devices

In a recent publication [24] Pipattanasomporn et al. collected the load profiles of selected major household appliances like dishwashers, AC units, refrigerators, washing machines and dryers. The data is available either in one second intervals or in one minute intervals that average the consumption over 60 s periods. Hence, due to the quality of the data, devices from this survey are modeled in Okeanos.

Here, the initial results presented in [25] are extended. These results show that by optimizing three household appliances of one household, Okeanos can save up to 5.9% of energy costs per month. The next logical step is to increase the number of households involved. That is, this section studies the impact of a rising number of households on the costs per household per month.

Not every household is alike, therefore, the load profile for every household is randomly scaled to either 25, 28, 30, 33 or 35 kWh per day. Additionally, it is randomly shifted between 1 h of its regular time. Finally, dishwashers, washing machines and clothes dryers run with a 33% chance. This configuration is chosen to account for different habits and usage patterns of customers.

As illustrated in Table 1 and Fig. 3, altering the number of households does not change the outcome. It, however, can be seen that the peaks are getting more extreme the more households are involved.

Table 1
Comparison of costs per household per month with an increasing number of households.

	10 Households	20 Households	50 Households
Costs per month per household	$88.57	$86.84	$90.23

At least two explanations should be considered when interpreting this results. On the one hand, there are too few devices that can be shifted. Also, because the load profiles of households have a minimum at the point in time when energy is cheapest, devices hardly have any other choice but to be switched on at that time. Further, because the average consumption of households is mostly the same, the energy consumption keeps stacking up and, as aforementioned, load shifting devices cannot smoothen the peaks.

On the other hand, the convex cost function at every point in time could need its parameters readjusted. This, however, is not very likely, as the devices that respond to costs, already run at the cheapest points in time. Households, however, do not react to different costs, which explains the peaks and the stacking of load profiles.

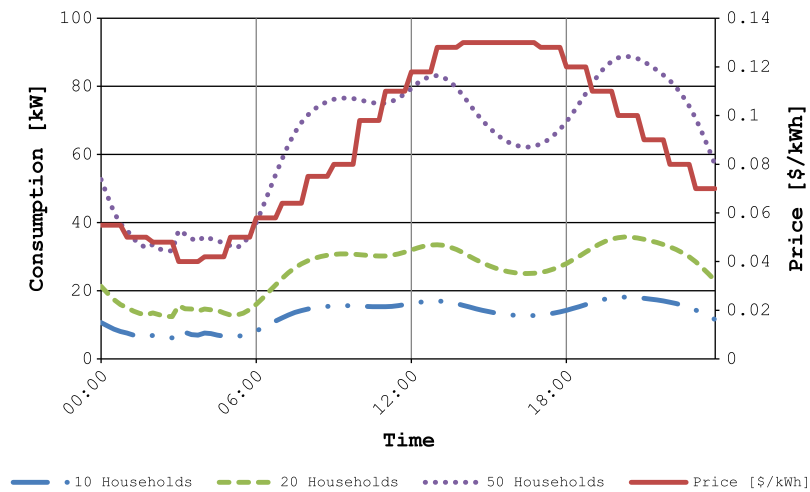

Fig. 3. Impact of the number of participating households on load profile.

According to Table 1, the costs per household per month do not show a significant difference when the number of participating households is increased. The reason for this is the same as described before: The load profiles are stacked.

Evaluation of Okeanos with plug in electric vehicles

This use case investigates the impact of integrating PEVs in the previous use case. As electric vehicles are all about storing energy, this is an extension to the implemented game theoretic algorithm [7], which proposes an energy consumption scheduling game. The original game was never designed for storage. The micro-storage management game proposed by [9] is contrary to that, it only proposes storage devices and does not do any load shifting. The proposed combination of both games is based on simulation only and there is no mathematical proof given unlike the individual games. Further, due to the use of PSO and the fact that it is a meta-heuristic, an optimal solution cannot be guaranteed. It should also be noted that all PEVs begin with an initial state-of-charge of zero.

Impact of penetration of plug in electric vehicles on costs per household

The first use case in the category of PEVs evaluates the impact of different penetrations of PEV on the total consumption. This simulation is based on 20 households, with either 0%, 25%, 50%, 75% or 100% of them owning one PEV. Owning really means having it stand around and not actively use it for transportation as for what it is made. In this configuration it acts like a rechargeable battery.

Furthermore, it uses a feed-in tariff of 50%. This means that if any device sells back energy to the grid, it will get 50% of the money it would cost the device to buy the same amount of energy. Additionally, as in the previous section, load shifting devices are switched on with a 33% chance.

As Fig. 4 shows, if only five of the 20 households, i.e., 25%, have a PEV, they completely change the load profile of households, overriding it with their own consumption pattern. This pattern, ultimately, is derived from the price function. As can be seen, PEVs charge themselves at the beginning of the day where the price for energy is cheap and use this energy later in the day to prevent the household from having to pay the peak price.

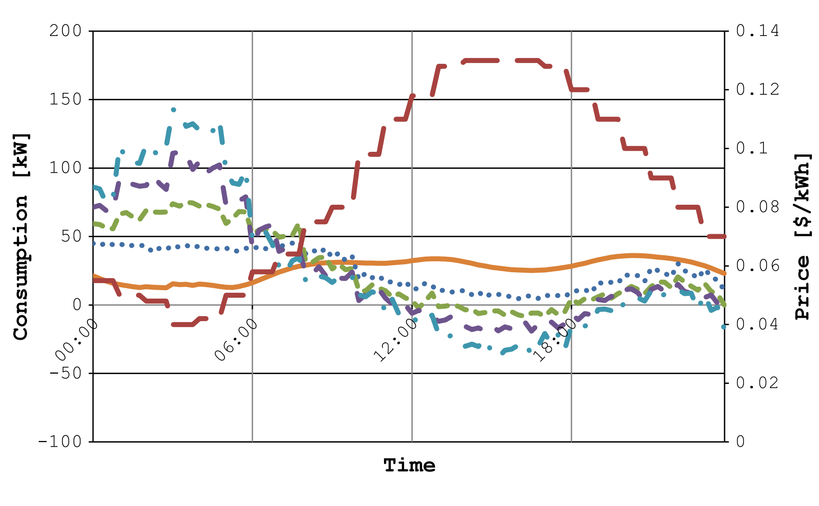

Fig. 4. Impact of penetration of PEVs on load profile. % PEV: variable, feed-in tariff: 50%.

An interesting phenomenon can be noticed at the end of the day at around 11 p.m. At this time devices start to discharge their remaining energy. This is due to the limited planing horizon, which is currently 24 h. Because devices cannot plan more than that, they want to sell the remaining energy to get the most out of the day.

The change of the load profile can be either wanted or unwanted. Even with a 25% penetration of PEVs, the peak consumption is nearly at 40 kW, compared to roughly 30 kW if there are no PEVs present. For higher penetrations, there is an even higher peak at the low-cost periods. This could be another unwanted peak as the grid needs to be prepared for that. If the grid is capable of transporting that amount of energy, this could be valuable to the utility company, because it sells cheap energy to customers and gets expensive energy for a cheap price, e.g., with a 50% feed-in tariff, which can be sold to other utility companies. Customers, despite the low feed-in tariff, still profit from selling energy back.

If the grid is not capable of handling that amount of energy, a possible countermeasure would be to adjust the cost function. The base price could either be changed or the factor, the costs per kWh at a point in time rise, could be adjusted as well. The latter countermeasure potentially has higher prospects of success, as it particularly penalizes high uses of energy, which, eventually, leads to a flatter load profile.

Table 2 compares the average costs per month for a household for a different penetration of PEVs with a 50% feed-in tariff. Most notably, the more households use PEVs the cheaper the average price for all households. Finally, when all households own a PEV and do not use it for anything else beside from participating in load scheduling, households can cut down electricity costs to approximately one fourth compared to not using PEVs at all.

This, however, is very unlikely to happen outside of simulation, as the simulation does not take a wide range of factors into account. Especially, (i) households own PEVs to use them and not let them stand in the garage at the charging station and (ii) the wear of batteries, etc. is not taken into account.

Table 2
Comparison of costs per household per month with an increasing number of households owning PEVs with a 50% feed-in tariff.

	Penetration of PEVs				
	0%	25%	50%	75%	100%
Costs per month per household	$88.20	$65.98	$52.27	$40.36	$27.50

The simulation, though, respects the maximum capacity, the minimum capacity, the maximum charge at a time and is also capable of "unplugging" a PEV, which means that the vehicle is currently in use and cannot be used for load scheduling. Furthermore, if a PEV is used, it also loses some charge, which can be expressed by the software as well.

Cross comparison of impact of feed-in tariff and penetration of plug in electric vehicles on costs per household

This use case is based on the previous use case, however, greatly expands the changed parameters. A parameter study of the feed-in tariff and the penetration with PEVs is done, unlike the previous use case that assumed a fixed feed-in tariff of 50%.

Fig. 5 illustrates the load profile when changing the feed-in tariffs. It clearly shows that the higher the incentive, i.e., the higher the feed-in tariff, the higher the likelihood that PEVs will charge during low-cost periods and discharge at high cost periods. Again, this is very similar to previous findings and is the result of trying to minimize the occurring costs for each device.

More interesting, however, is Table 3 and Fig. 6, which illustrates, respectively gives the exact numbers of the costs per household per month depending on the feed-in tariffs and the penetration with PEVs.

As previously pointed out, the costs per household per month decrease the more incentive is given (a higher feed-in tariff) or the more PEVs are available in the simulation. This effect results in households earning money at the end of the month when there are both, a high incentive and a high number of PEVs available.

The reason that the costs decrease with an increasing number of PEVs even with a 0% feed-in tariff is that the PEVs in that case are not actually selling the energy back to the grid, but provide it to other devices. Obviously, in total, this leads to a lower price, as PEVs provide energy during the high-cost periods.

Table 3
Comparison of costs per household per month with different feed-in tariffs and a different penetration of PEVs.

	Feed-in tariff				
	0%	25%	50%	75%	100%
Households with PEVs					
0%	$87.01	$86.41	$88.20	$86.78	$87.24
25%	$69.25	$68.70	$65.98	$63.77	$64.37
50%	$64.21	$61.55	$52.27	$38.81	$35.36
75%	$66.65	$58.73	$40.36	$17.47	$9.05
100%	$67.52	$56.01	$27.50	−$9.50	−$14.98

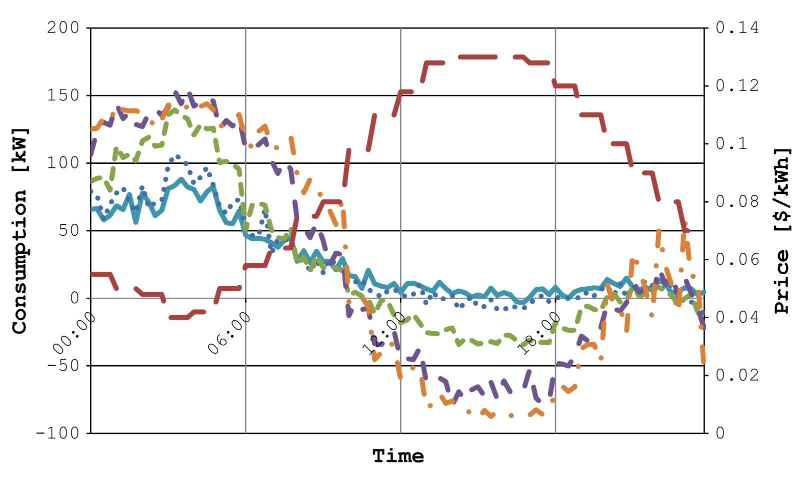

Fig. 5. Impact of feed-in tariffs on load profile. % PEV: 100%, feed-in tariff: variable.

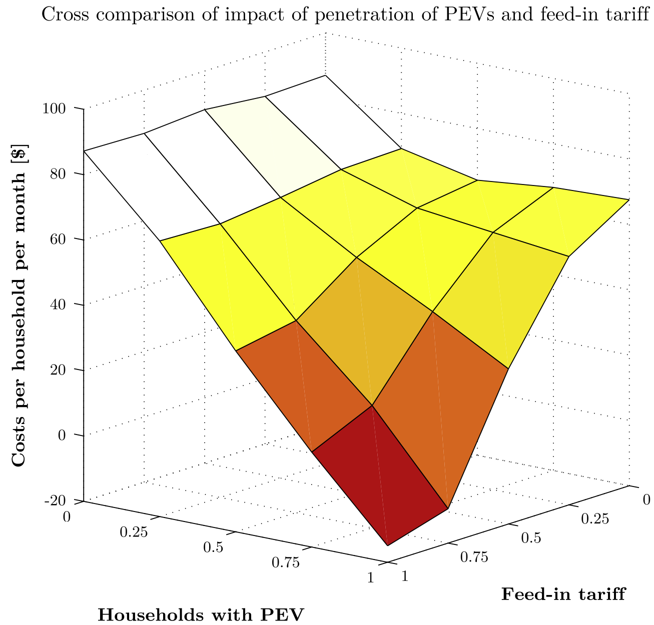

Fig. 6. Cross comparison on the impact of different feed-in tariffs and penetration of PEVs on the costs per household per month.

However, earning money through the use of PEVs seems unlikely as [9] simulated the impact of storage devices as well, with the result that in the UK 38% is ideal number of households owning a 4 kWh storage device, when the savings of up to 13% is at its maximum. These savings do not result in the households earning money at the end of the month. What can be done to make it more realistic is to adjust the aforementioned factor by which the costs per kWh rises.

Further, it can be noted that increasing the feed-in tariff from 75% to 100% has a significantly smaller impact than increasing it from 50% to 75%. One reason could be that the PEVs already use their whole available capacity when the 75% feed-in tariff is offered. Similarly, increasing the percentage of PEVs from 75% to 100% does only have a big impact with high feed-in tariffs.

There does not seem to be a particular parameter combination that is ideal for every case. The decision on the feed-in tariff has to be made by the utility company for every specific situation. Obviously, the number of PEVs in a grid need to be taken into account for that decision.

Conclusion

In this paper, Okeanos, a novel multi-agent demand response simulation platform focusing on the evaluation of game theoretic approaches was described. A major characteristic is its extensibility, which allows to support numerous household devices and enables the simulation of various games.

Simulation with three household devices shows that the costs per household are unaffected by number of involved households. Results involving PEVs demonstrate a decrease in the monthly utility bills per household with an increasing penetration of PEVs and feed-in tariffs. Future work will study the impact of using PEVs for commuting on the costs per household over longer periods (e.g. months, years, etc.).

Acknowledgments

The financial support by the Austrian Federal Ministry of Economy, Family and Youth and the Austrian National Foundation for Research, Technology and Development is gratefully acknowledged. Further, thanks belong to the Austrian Marshall Plan Foundation and The Chamber of Labour which have partly funded the research on this paper.

References

[1] Gudi N, Wang L, Devabhaktuni V, Depuru SSSR. Demand response simulation implementing heuristic optimization for home energy management. In: 2010 North American power symposium (NAPS 2010). Arlington (TX): IEEE; 2010. p. 1–6. http://dx.doi.org/10.1109/NAPS.2010.5619967.

[2] Han J-q, Piette MA. Solutions for summer electric power shortages: demand response and its application in air conditioning and refrigerating systems. Refrig Air Cond Electric Power Mach 2008;29(1):1–4.

[3] Palensky P, Dietrich D. Demand side management: demand response, intelligent energy systems, and smart loads. IEEE Trans Ind Inform 2011;7 (3):381–8. http://dx.doi.org/10.1109/TII.2011.2158841.

[4] Kok K. The PowerMatcher: smart coordination for the smart electricity grid. PhD thesis. Free University Amsterdam, Amsterdam, Netherlands; 7 2013.

[5] Fan Z, Kulkarni P, Gormus S, Efthymiou C, Kalogridis G, Sooriyabandara M, et al. Smart grid communications: overview of research challenges, solutions, and standardization activities. IEEE Commun Surv Tutor 2012;15(1):21–38. http://dx.doi.org/10.1109/SURV.2011.122211.00021.

[6] Federal Energy Regulatory Commission. Assessment of demand response and advanced metering; 12 2012.

[7] Mohsenian-Rad A, Wong VW, Jatskevich J, Schober R, Leon-Garcia A. Autonomous demand-side management based on game-theoretic energy consumption scheduling for the future smart grid. IEEE Trans Smart Grid 2010;1(3):320–31. http://dx.doi.org/10.1109/TSG.2010.2089069.

[8] Ibars C, Navarro M, Giupponi L. Distributed demand management in smart grid with a congestion game. In: 2010 1st IEEE international conference on smart grid communication (SmartGridComm 2010). Gaithersburg (MD): IEEE; 2010. p. 495–500. http://dx.doi.org/10.1109/SMARTGRID.2010.5622091.

[9] Vytelingum P, Voice TD, Ramchurn SD, Rogers A, Jennings NR. Agent-based micro-storage management for the smart grid. In: Proceedings of the 9th international conference on autonomous agents and multiagent systems (AAMAS 2010). Toronto (Canada): ACM; 2010. p. 39–46.

[10] Maharjan S, Zhu Q, Zhang Y, Gjessing S, Başar T. Dependable demand response management in the smart grid: a stackelberg game approach. IEEE Trans Smart Grid 2013;4(1):120–32. http://dx.doi.org/10.1109/TSG.2012.2223766.

[11] Bu S, Yu FR, Liu PX. A game-theoretical decision-making scheme for electricity retailers in the smart grid with demand-side management. In: 2011 IEEE international conference on smart grid communication (SmartGridComm 2011). Brussels (Belgium): IEEE; 2011. p. 387–91. http://dx.doi.org/10.1109/SmartGridComm.2011.6102353.

[12] Rieck C. Spieltheorie: Eine Einführung. 11th ed. Christian Rieck Verlag; 2012 [in German].

[13] Saad W, Han Z, Poor HV, Başar T. Game-theoretic methods for the smart grid: an overview of microgrid systems, demand-side management, and smart grid communications. IEEE Signal Process Mag 2012;29(5):86–105. http://dx.doi.org/10.1109/MSP.2012.2186410.

[14] Struß O. Open-Source-Modellierung und auktionsorientierte Regulierung dezentraler Energienetze. Diploma thesis. University of Bremen, Department of Computer Science and Mathematics, Bremen, Germany, in German; 1 2012.

[15] Lehnhoff S, Krause O, Rehtanz C, Wedde H. Dezentrales autonomes Energiemanagement – Für einen zulässigen betrieb innerhalb verfügbarer kapazitätsgrenzen (distributed autonomous power management – for a reliable operation under feasibility constraints). Automatisierungstechnik 2011;59(3):167–79. http://dx.doi.org/10.1524/auto.2011.0906.

[16] JIAC Development Team. JIAC – Java Intelligent Agent Componentware, DAI-Labor, manual. Version 5.1.3; 10 2012.

[17] Bellifemine FL, Caire G, Greenwood D. Developing multi-agent systems with JADE. John Wiley & Sons; 2007. http://dx.doi.org/10.1002/9780470058411.

[18] Galland S, Gaud N, Rodriguez S, Hilaire V. Janus: another yet general-purpose multiagent platform. In: Proceedings of the 7th agent-oriented software engineering technical forum (TFGAOSE 2010), Agent Technical Fora, Paris, France; 2010.

[19] Bordini RH, Hübner JF, Wooldridge M. Programming multi-agent systems in AgentSpeak using Jason. John Wiley & Sons; 2007. http://dx.doi.org/10.1002/9780470061848.

[20] Spring Developers. Spring framework reference documentation, 4.0.6. RELEASE; 7 2014.

[21] The OSGi Alliance. OSGi core release 5; 3 2012.

[22] The OSGi Alliance. OSGi compendium; 5 2013.

[23] Kennedy J, Eberhart R. Particle swarm optimization. 1995 IEEE international conference on neural networks (ICNN 1995), vol. 4. Perth (Australia): IEEE; 1995. p. 1942–8. http://dx.doi.org/10.1109/ICNN.1995.488968.

[24] Pipattanasomporn M, Kuzlu M, Rahman S, Teklu Y. Load profiles of selected major household appliances and their demand response opportunities. IEEE Trans Smart Grid 2014;5(2):742–50. http://dx.doi.org/10.1109/TSG.2013.2268664.

[25] Lausenhammer W, Engel D, Green R. A game theoretic software framework for optimizing demand response. In: 6th European innovative smart grid technologies (ISGT 2015). Washington (DC): IEEE; 2015. submitted to IEEE ISGT 2015.