Multi-agent proactive charging infrastructure for electric vehicles

Adrian Wöltche
Institute of Information Systems (iisys)
Hof University of Applied Sciences
Alfons-Goppel-Platz 1
95028 Hof, Germany
Email: adrian.woeltche@iisys.de

Georg Jung
Institute of Information Systems (iisys)
Hof University of Applied Sciences
Alfons-Goppel-Platz 1
95028 Hof, Germany
Email: georgjung@acm.org

***Abstract*—In this paper, we propose a novel intelligent framework based on Multi-agent systems (MAS) for proactive operation of a charging infrastructure for electric vehicles.**

I. Introduction

Renewable energies differ fundamentally from conventional sources. Two aspects are in particular stark contrast. First, renewables are intrinsically *decentralized* as they lend themselves more to a high number of small, diverse, distributed, plants, whereas conventional, fossil or nuclear based, energies generally require few, large, centralized, facilities. Second, whereas conventional energies are essentially consumables, which can be used whenever needed until the ultimately finite stock is depleted, renewables are virtually unlimited, however in turn volatile and *fluctuating* and have to be harvested and used whenever available.

With the increasing fraction of renewables as primary sources, these differences require a tectonic shift in the layout and functionality of the energy provision and consumption infrastructure, moving from few separate, centralized and centrally controlled, demand-driven, systems regulated through human interaction or physical feedback loops (electricity, gas, fuels) towards complex, agile, multi-stakeholder, heterogeneous, closely interacting, networks of energy providers, transformers, storages, and consumers driven by supply and regulated through IT-based data exchange and processing. For such networks, which are coupled by IT rather than electro-mechanically and thus can synergically entail any form of energy provision or use (spanning all sectors including electricity, heat, cooling, pressure, motion, mobility, *etc.*) as well as integrate with any IT environment (ERPs, timetables, schedules, markets, forecasts), we employ the term *polyenergy* networks.

Finding a cost-optimal provision of energy to individual consumers—including but not limited to electric vehicles—inside polyenergy networks constitutes a core challenge for the general energy infrastructure. The information that can be leveraged for optimization varies substantially between the manifold constituents of a polyenergy network, in this case particularly the individual vehicles (*e. g.*, capacity, state of health, power consumption, projected future use), charging points (*e. g.*, location, socket type, power provision, operator), power backbones (*e. g.*, type of energy source, fraction of renewables), and market models (*e. g.*, time-cost-function, stakeholders) effectively prohibiting a centralized, all-knowing, approach.

Therefore we propose an intrinsically decentralized solution based on autonomous agents that strictly implement the subsidiary principle and individually leverage information wherever available. The system manifests the envisioned cellular approach [1] spanning all sectors of the general energy infrastructure and avoiding centralized control in favor of a distributed intelligence, thereby embracing the diversity of the individual contributors.

We contrast our approach against recent solutions with centralized control (*e. g.*, virtual power plants [2], swarm batteries [3]) that essentially emulate the properties of conventional, top-down, single sector, solutions in order to carry them over into the age of renewables. We argue that possible advantages of centralized solutions cannot effectively be preserved due to the differences in the very nature of renewables *vs.* conventional energies outlined above [4]. Moreover, the security of sensitive data exchanged in smart grid applications (see, *e. g.*, [5]) is much harder to guarantee in a centralized, single-point-of-failure architecture (see [6] for the respective regulation and legislation in Germany), whereas a distributed, thoroughly decentralized, intelligence is inherently robust in case of failures of any single constituent.

II. Multi-agent systems

Multi-agent systems (MAS) and *agent based modeling* (ABM) [7]–[9] manifest closely similar construction principles in software as the ones we identified for the cellular approach to the energy infrastructure and already have been proposed for energy networks (*e. g.*, [10]). In particular:

(a) MAS are built for strict modularity. Each agent is autonomous and can dynamically be added, altered, or removed from a platform. Agents are therefore ideal to model variable stakeholders inside an uninterrupted market.

(b) Agents encapsulate the low level hardware communication layers and instead communicate with other agents on a high level of abstraction via shared, semantically defined, ontologies. This eliminates the need for one single,

common, ubiquitous, hardware interface standard and facilitates the integration of components from multiple sectors.

(c) Agents are equipped with internal decision making capabilities which enable them to respond differently to the same outside stimulus. The advantage is twofold: First, each agent can take the particular properties and characteristics of the managed device into account, thereby optimizing its distinct contribution more accurately than a centralized intelligence could. Second, with each constituent reacting individually to the same outside stimulus, the overall system is intrinsically less likely to tip or oscillate.

(d) As MAS implement the subsidiary principle, information can be kept as local as the decisions based thereupon. This substantially reduces the volume of the information communicated, putting less stress on the communication links, increasing robustness in case of high latency or communication failures, and massively facilitating the protection of critical, private, information.

While these aspects make MAS an obvious choice for dynamic, multi-stakeholder, systems, we are aware that the lack of central supervision and control gives raise to various challenges. We will address these in the following sections.

III. Towards an integrated system

In our proposal, the agents represent the various power consumers and/or producers (for simplification dubbed "*prosumers*") such as individual charging stations, battery storages, or electric vehicles, that constitute the endemic energy system. Each agent is aware of the intrinsics (power, capacity, *etc.*) of its assigned entity (local prosumer) and apprehends the facts it needs for making decisions from its environment (*e. g.*, charging station observes state of charge of vehicle) and from other agents (*e. g.*, charging station observes power output of photovoltaics).

Moreover, agents are equipped with executable plans that are triggered by periodic timeouts, internal schedules, or changes in the state of the environment. Generally, such plans can be stateless and therefore purely reactive, or they can take internal states, internal and external history, or even projections about future conditions into account, and on these grounds act proactively.

The agents contribute by communicating their intent within their singular capabilities (*e. g.*, power or energy demand, residual capacity, charging power, backfeed, *etc.*). Barring any central supervisor that micromanages the individual components, each agent needs to be able to express a level of urgency of some kind for its own demands to receive the appropriate priority with respect to available resources. For this we introduced various mechanisms that we validate in various combinations such as energy/power provision within balanced ratios among the consumers and/or self set priority levels attached to the demands. Whenever priorities are used, provisions are in place to prevent exploitation and instead motivate cooperative behavior. High priorities will come with the penalty of higher cost, while low priority demands are encouraged trough lower prices. Therefore, agents are motivated to anticipate future needs, plan ahead, and ultimately contribute to a balanced load on the resources.

The communication takes place within an self-organizing, decentralized, market [11] (*p.* 40, 62 *ff*) that enables all participating agents to trade power and energy between each other, depending on own customer and trader relationships without central instance.

IV. Scenario

To validate our approach we built a simulation, consisting of a photovoltaics-array, a battery-storage system and two electric vehicles connected to two charging points, all connected to a common grid node providing power compensation over the superior transmission grid operator.

A. Simulation

For simulating the physical infrastructure we make use of the tool SimulationX that provides us with the necessary entities via its GreenBuilding and GreenCity libraries (see *fig.* 1). With these entities we were able to create an integrated micro-grid with the photovoltaics array producing power on a simulated summer day, the battery storage contributing to the grid within its power and energy limits, the main grid node compensating over- and underflows and both electric vehicles charging as economically as possible.

SimulationX models its objects as sets of differential equations that are numerically approximated in simulation runs. To connect the single threaded SimulationX solver with the concurrently running MAS we introduced three network communication nodes that serve as connection between the (simulated) physical world and the agents.

We use the first network node for observing the power produced by the photovoltaics array, as well as the residual load on the main grid node, the second for managing the battery storage, and the third for both electric vehicles.

For every battery storage and electric vehicle we perceive information about the maximum possible charging and discharging power, the maximum possible energy to store, the current state of charge and the maximum state of charge as energy limit.

Alternatively, we could have used one single network node for all five items or five network nodes one for each single item, however our choice of combining some network nodes together and separating others enables us to validate the flexibility of the MAS approach when it comes to interface communication between our software abstraction and the physical hardware level.

Each network node is seen as a function call by SimulationX. This implies the simulation being stopped during the transmission of data, through each network module, until an answer is received, one after another. It also means that we are able to manage the power evolution of the simulation by the output our MAS transmits on each network connection, which is used for setting the power for charging or discharging for the battery storage and both electric vehicles.

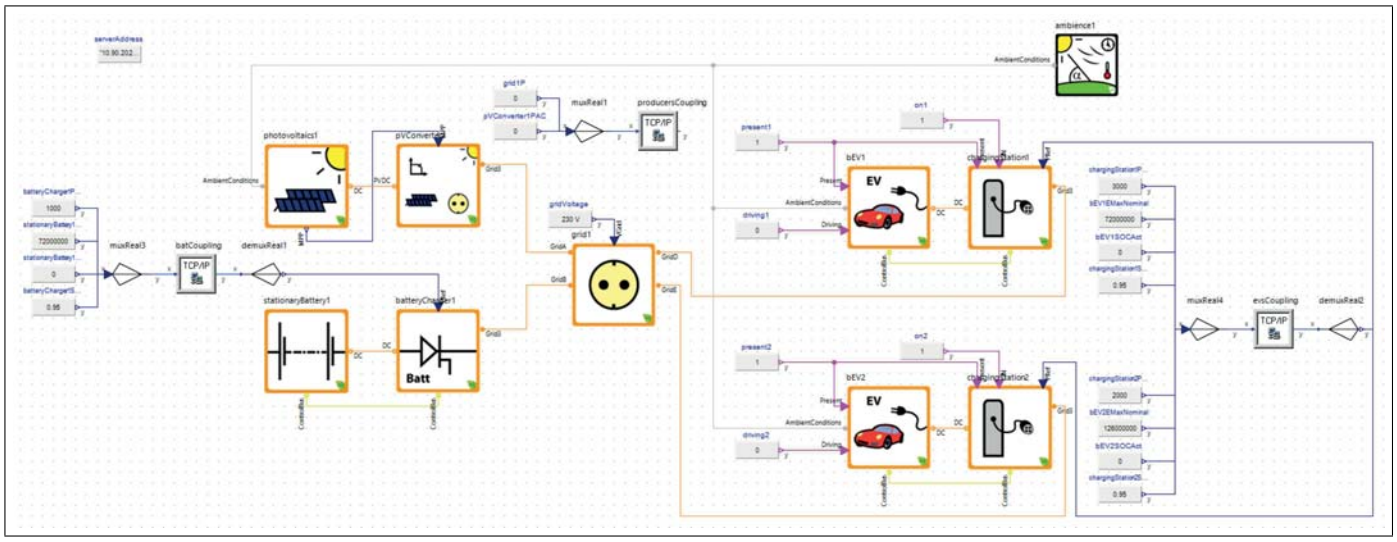

Fig. 1. Our micro-grid scenario in SimulationX. Grid node in the center, photovoltaics array in the top left, battery storage in the bottom left, two electric vehicles to the right. Three TCP/IP network communication nodes with various input and output connections as interfaces to our MAS.

With this setup we simulate a whole 24 hours within a few seconds by scaling the simulation time, which would not be possible in real world scenarios and enables us to launch a large number of simulations validating multiple alternatives. Because of the high simulation speed we have to set the time interval between network communication acts to 60 virtual seconds, as too narrow intervals would slow down the simulation speed dramatically.

B. Multi-agent system

Our multi-agent system is intrinsically abstract, only communicating via ontologies. However, as physical and simulated hardware or other software tools in fact are not abstract and instead have diverse communication and interaction protocols based on floating point and integer values, we need to break our abstraction layer when it comes to interfaces between our MAS and the real or simulated world.

For maintaining the abstraction layer within the homogeneous multi-agent system we let additional interface-agents translate between the specific device protocols, our ontologies, and agent communication protocols (see *fig.* 2). In this manner, every inner agent is able to exchange information and commands with the outer, real, world without knowing the specific protocols.

In our simulation scenario, each of the five devices is represented by a single agent, and each of the three network nodes is connected to a single, delegated interface agent. The five device-representing agents are then able to communicate and trade with each other by their likes, and may exchange information with the simulation over the three network interface agents transparently.

For the technical implementation of our multi-agent system we rely on the Foundation for Intelligent Physical Agents (FIPA [12]) IEEE standard describing a complete MAS platform including agent definitions, agent behaviors, ontology usage, and communication protocols. Of the frameworks implementing

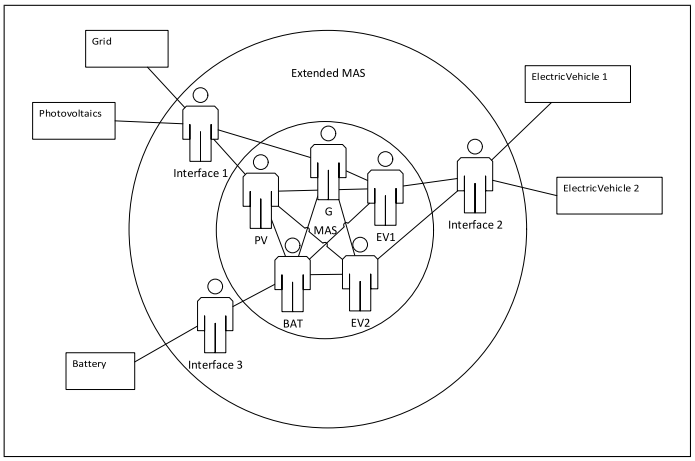

Fig. 2. Homogeneous multi-agent system in the center representing our simulated devices, here for engineering reasons extended by interface agents for the network interface nodes transparently translating between inner agent ontologies and simulated devices outside the MAS.

the FIPA standard, JADE [13], [14] is the most mature platform in our eyes.

C. Agents

The FIPA-agents come with a large number of capabilities (*e. g.*, message transaction with ontology support, subscription behavior for oberserving patterns, yellow pages service for finding each other, various interaction protocols, such as contract-net interaction protocol [15] for preventing resource conflicts while trading) that can be reviewed at the FIPA association [16].

With these features, our consumers are able to find the producers currently available for requesting and ordering power they need, and the producers are able to propose and sell power they have. With the battery storage representing a prosumer,

it is able to trade power in both ways for maximizing its own profit within the system.

Moreover, for optimizing costs, each agent sets the price limits it is currently willing to pay for buying power, normally depending on self-set priority levels. In our simulation, we decided to set fixed price limits, so that we can simulate different energy flows solely by adjusting the current price limit for each agent participating on the decentralized market.

We already stated that SimulationX sees each network node as a synchronous function call (see *sec.* IV-A), blocking the entire simulation until an answer is received. However, in a real-life situation—where the physical world keeps moving on—we cannot assume a global synchronization with these communication events and therefore we cannot allow the MAS to depend on the pulse of the physics simulator. Similarly, the agents must not influence the timing of each other. Every agent lives in its own timeline and manages its environment but is not controlled by it. This is why we use multiversion concurrency control [17], [18] as one of the two[1] common solutions for parallel computation synchronisation for our agents, as well as for the simulation communication.

This means that whenever the simulation inputs new information towards its delegated agent inside the MAS, that agent distributes the information towards its subscribing inner agents and in turn immediately responds to the simulation with the latest version of the current output value that the simulation needs to continue its calculation. By these means, as the trading inside the MAS is run asynchronously, the adaption of the agents to the simulation, and vice-versa, is automatically delayed by latencies that also happen in the real world. The information loss through short-time deprecation that occurs during communication latency is therefore faithfully simulated when using multiversion concurrency control.

D. Decentralized Market

The core algorithm permitting our agents to trade power with each other is ideally based on a decentralized market [11] system. Within our adoption of a decentralized market, there is no central database and no central mediator for the selling process. Instead, each pair of customer (equals consumer) and trader (equals producer) make their own and independent, direct negotiation. Of course, prosumers may pair up with either counterpart.

For the trading process each customer is equipped with a virtual basket for managing its orders, and each trader is equipped with a virtual marketplace for managing its offers and sells. All together, these represent the decentralized market state.

1) Ontology: The current state of the baskets and marketplaces as well as the communication between the trading agents all rely on the following market ontology, that itself depends on the ontology basics in JADE [19] (see `Concept`, `Predicate`, `AgentAction`):

(a) Every `Concept`, implemented as `Data` object, has a field for an `id` that will be important for referencing purposes between the decentralized states, when orders are sold, and a field for a `time` that is important for overriding old entries in states for multiversion concurrency control.

(b) `Need` is a `Data` representing customer needs, with the additional fields `customer` that is needed for traders to manage multiple customer needs, `power` that states the currently needed power, `energy` that indicates how much energy still is needed, which also could be unlimited, `ratio` giving a measure for the readyness to sharing in cooperation with other consumers on low capacities and `priority` that contains the self-set priority as a sorting criteria for traders to whom they preferably sell the capacities they have. The `priority` also can be set to `force` which represents consumers that have to take the power at any cost (*e. g.*, light switched on manually, electric vehicle in the urgent need of charge because of imminent calendric event suddenly assigned).

(c) `Offer` is a `Data` representing trader offers, with the additional fields `trader` that permits customers to manage orders from different traders, `power` representing the currently available power, `energy` indicating for how much energy-usage this power could be reserved, which may be unlimited, `price` for expressing the costs of this power offer and `reserved` as internal state for managing resource conflicts between totally available power, and the currently proposed, offered power.

(d) `Request` is an `AgentAction` a customer initializes with a given `Need`, which is then sent to all available or relevant traders. These each may respond with a duplicated `Request` with a set `Offer` as result to the requested `Need`.

(e) `Order` is an `AgentAction` a customer initializes after having received a relevant `Offer` back in a previous `Request`. The `Order` has an additional field `purchased` that only is set by the trader when the `Order` was successful. `Order` can also be used for priority override or ratio sharing purposes. Though it could be interpreted as an order in these cases, the customer still is free to accept or refuse it.

(f) `Info` is another simple `AgentAction` that enables traders to notifying its customers about new `Offer`s, such as more `power` or lower `price`. It could initiate another trading round, if relevant for the customer.

2) Direct Trading: The direct trading algorithm between customers and traders relies on, but does not exactly implement, the FIPA Contract-Net Interaction Protocol [15]. The original contract-net protocol performs between single customers and multiple traders, and requires synchronized locks for collecting all awaited responses between transaction steps. This is not applicable if traders may disappear (*e. g.*, network connectivity loss) and awaited responses never arrive. Though timeout mechanisms allow to reestablish working conditions, the agent and other dependent agents are inoperable until the timeouts trigger. Moreover, with slow and unreliable mobile network

[1]The other solution is enforcing mututal exclusion on resources, by, *e. g.*, blocking locks.

connections being used between agents in real-life situations, timeouts are nearly impossible to adjust for both fast timeout triggering for connectivity loss, and enough timeout for letting slow communication go through.

Multiversion concurrency control, which we use for our adaption of the contract-net protocol (see *fig.* 3), comes without any synchronization locks and timeouts. This permits the agent and the entire (partly-)dependent system to move on with the last known state, while still being able to adapt to new calculations when available. This way, we effectively averted dead locks during the entire trading algorithm.

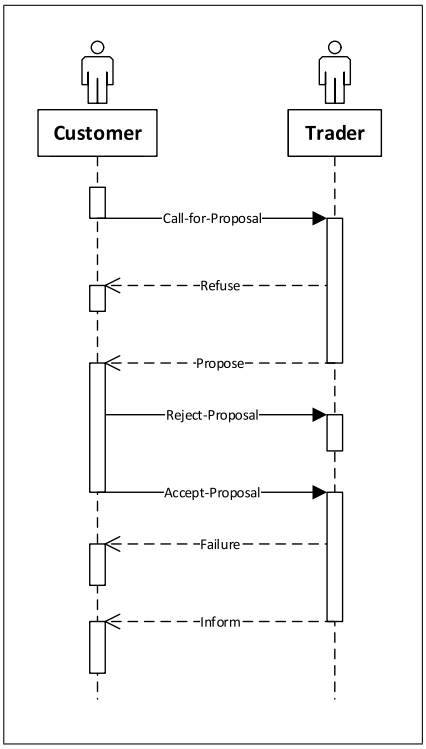

Fig. 3. Our implementation of the contract-net protocol without synchronization timeouts.

So when an agent, delegated for a consumer, receives the current need of its assigned (real) consumer (that is, in our scenario, transparently depicted by the interface agent connected to the simulation), he first updates its current state of need, and when the need changed (what it normally does, except when the agent receives the exactly same need multiple times, which unfortunately could happen in multiversion concurrency control environments), he sends a `Request` action (the Call-for-Proposal) with the current need to the producers currently available for him.

Similar to the consumers, the producers receive their current production (or to be precise, their possible sellable capacities) and also update their state. When the production actually changed from before, they notify the customers they already know (previous or current ones) by an `Info` action about the new `Offer`. If that `Offer` is in any case relevant for the customer that received the `Info` (*e. g.*, when it is cheaper than previously made orders, or helps to fulfill unsatisfied needs), the customer starts a `Request` action with its current need as described above.

Both possibilities may overlap or override each other through quick succession, however this poses no synchronization or consistency problem as of the multiversion concurrency control concept stepping in. In both ways, the simulation or real world is updated with the latest version as soon as available, as it could not proceed with anything else other than the last known state.

So in any case, more or less traders receive a `Request` with a current need from a customer. Each trader now checks the given need against its current power capacity available. If there is no `Offer` at all or if other customers with higher self-set priority currently occupy all available power, the trader responds to the `Request` with a refuse. In this case, the customer has to continue with its last valid state but could not satisfy new and changed needs.

If the trader is able to fulfill some or all of the `Need` the customer has, he creates a traceable `Offer` to propose, and reserves that offered power from its current available capacity. If the customer set a higher priority than existing customers currently have, the unavailable power is removed from the existing customer `Orders` and added to the current `Request` with the higher priority. Existing `Orders` having changed raise updates to the appropriate customers with the current `Order` overriding the previously received one. As soon as the `Need` of the customers changes again, or if the changed `Offer` is unsatisfying, they naturally may start another trading round.

It is important to state that we do not override the priority itself, or force previous customers to release their assigned power immediately, as this would harm the subsidiary principle that permits a customer to raise their priority again, instead of releasing their power for making space to the higher priority customer. Though a cooperating customer would immediately release the power not assigned anymore, he is not forced but probably would be penalized later with higher prices due to this behavior.

Independently, the power being reassigned to the higher priority customer can lead to tiny spikes in the residual load of the grid, when that power is still used by the customer with the lower priority. Normally, when the customer with the lower priority is willing, that resource conflict resolves itself, at the latest, within the subsequent trading round, but most of the time immediately, as soon as the reduction is applied by the customer. Else, other participants need to adapt again, for resolving the assigned power resource conflict, as in real grids, when participants are only behaving reactive to the frequency.

The same as with priorities can happen with ratios that only would apply in equal priority situations, when the ratios of all customers altogether represent the share of the power available for the equal priority customers. This way, force level customers override everything, priority levels override lower priorities and equal or no priorities (which also are equal) share within their ratios. If no priority or ratio is defined at all, first come first serve principle is applied for power trading. We can see these assignment levels in Figure 4.

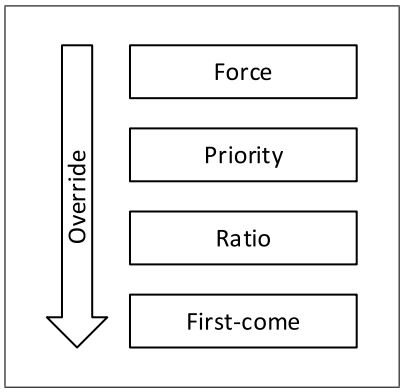

Fig. 4. Importance and override mechanics of the assignment algorithms for low power situations.

Either way, the originally requesting customer got proposed some amount of power in a traceable `Offer` that can be less or equal to his current `Need`. That `Offer` is then checked against self-set price limits and whether the offer is helping to satisfy the current `Need`. If anything is not to the customers satisfaction, he rejects the proposal and tries to request power towards this trader again when its `Need`s change or when new `Offer`s from this trader are available. For the participating trader, a reject permits to free the reserved resources again. The same goes for all other traders offering something to a customer `Request`.

In any case, when the `Offer` is in any way helpful and acceptable, the customer accepts it. By this, the trader again has the chance to send out a failure, if something unpredictable happened in the power supply, or if the `Offer` accepted does not fit to the `Offer` proposed before (*e. g.*, customer changed price to a lower value, which is not legit), but when everything is still valid, the trader sells the reserved power by marking it sold and informing the customer about it. That one now is able to use the purchased power (*e. g.*, for charging the electric vehicle) and request `Need`s later again, when they changed (or when the `Offer` changed), while still being aware that because of other customers participating in the system, the power he currently has assigned, might change in the future.

The most important special case is when a consumer does not need any more power or energy. In this case, it starts another trading round that is meant for notifying all traders currently having sold or reserved capacity to free these resources for others, as these resources would not be freed else way, because of the multiversion concurrency control maintaining the last state, until a new value or priority overrides it. After having received the acknowledge for this last information as `Propose` of zero capacity, the customer can be sure that his currently assigned resources were freed and he can surely stop its load then, too.

3) Proactiveness: By simply trading power depending on availability and needs, we would not need a state when we would only react on new signals, but we decided to have a basket and marketplace state, as this permits us to proactively shape the future by valuing previous decisions and by saving current decisions for future trading rounds.

We already mentioned the first come first serve principle for customers neither stating a priority or ratio that only works in succeeding calls, when we only sell power not already purchased by some other customer before. If we would not value the current state that was updated during previous rounds, traders would reassign their full power towards the current customer every time, though other customers might use that power already from before.

The same goes for priority overrides, as we only can override with a higher priority when we know of the lower priorities and of the capacity they might release. And with sharing ratio, this is also true.

Therefore, by maintaining a state and by setting priorities as well as by sharing depending on set and maybe changing ratios, we are able to reach proactive behavior throughout the whole system, because we can save decisions that can alter future calculations.

4) Priority calculation: For setting and overriding priorities, we valued the fact that no regular device likes being turned on and off in high frequency. That is why we developed the following decay formula as a centered bell-curve for decreasing a set priority by time after being turned on (or increasing after being turned off). This way, devices turned on override priorities of devices already running for a longer time, that are easier to turn off without damage at that time.

$$p(t) = p_0 \cdot e^{-(s(t-t_0))^d} \tag{1}$$

p_0 is the original priority, in our case set by the fraction of available to needed power, with the result that devices with higher power assignment also have a higher priority for not being turned off again too soon. When p_0 is negative, the curve rises instead of decaying for situations, when the device was turned off. s is the decay scale that permits us to stretch the curve on the horizontal time axis for reducing the decay speed over time, when a device is slower in turning on and off without taking damage. t_0 is the time the device was turned on or off, and it marks the starting point of curve. In combination with t, as the variable, we are able to get the current priority after the time t since t_0 passed. d is the decay rate, in our case set fixed to 4, that sets the strength of the decay in the curve.

This self-set priority, calculated by this formula, overrides lower priorities of other customers inside the trader reassignment routine. In case of the same priority (as after some time, when the curve reached near 0, and is forcefully set to 0, as the near-zero priority does not play any important role anymore), depending on whether a ratio is set or not, the power and energy reservations and assignments are shared or first come first served.

5) Ratio calculation: For the ratio, each consumer, having an energy capacity limit, calculates its ratio by the following simple formula:

$$r = 1 - \frac{l(t)}{l_{max}} \tag{2}$$

The ratio share r is calculated by dividing the current load $l(t)$ by the maximum possible load l_{max} and subtracting that

fraction from 1. This way, storages (like our battery storage and our two electric vehicles) with a high load set a low ratio share and vice versa. By the time, the ratio drops, when the storage fills. Setting a fixed ratio (*e. g.*, 0.5) is possible as well.

As no customer knows the ratios of the other customers, they surely altogether add to a sum being less or more than one, thus not being normalized. This is why the internal calculation in the trader, that knows its customers with same priority and different ratios, normalizes the different ratios towards a range between zero and one for further calculation.

Moreover, when a customer set a ratio that would lead to a power assignment higher than its needs, the difference between the ratio that would exactly meet its needs and the current ratio set too high is distributed on the other customers, so that the free energy is redistributed and completely shared as well. When there still is capacity left, that one can be used by lower priorities in further trading rounds then, so that there only is capacity left after all, when all customers are satisfied as good as possible.

We want to remark that the recalculated ratio of the traders is never told to the consumers that still set their own ratio after the above formula, but depending on the recalculation having taken place inside each trader, the offered power is adapted, and the already sold power is rebalanced by updating the consumers, with still letting them the choice to adapt or not. This way, we always value the subsidiary principle by only escalating what cannot be decided locally.

6) Offer creation: In summary, when a trader gets a new `Request` with a need, depending on whether the own capacities of power and energy are limited or not, either the full need is satisfied or depending on the set priority or ratio, overridings and reassignments of lower priorities as well as redistributions of shared capacities are calculated and sent to the appropriate customers (remember *fig.* 4).

This way, it is possible to fulfill more needs (concurrently and one after another) in a managed way than when every customer would just exhaust its full possibilities. So in our approach, we are able to proactively charge multiple electric vehicles and storages by various self-set and subsidiary importance levels with only few left capacities.

V. System operation

For validating our approach, we launched several simulation runs with different parameter settings concerning priority overrides and ratio shares. This way, we were able to observe the decentralized behavior of our agents in combination with SimulationX *sec.* IV-C as substitute for a micro-grid.

A. Priority and ratio charging

For this paper, we exemplarily launched a combined run with priority override being enabled, when batteries (storage and both electric vehicles) started or stopped charging and with ratio share enabled when priority levels were equal, and we launched a run with priorities disabled, so that the ratio share would be never overwritten by priority concerns.

The simulation was parameterized with fixed costs for provided power loads, and with fixed cost limits for consumers. This way, we could simulate cost-optimizing behavior, as the agents tried to minimize costs for charging their power by not exceeding their cost limits. In our runs, we set the grid costs to 0.25 € per kWh (for the superior transmission grid operator costs) and the photovoltaics costs to 0.07 € per kWh. The battery storage and electric vehicles were set a cost limit of 0.08 € per kWh, which restricted them to photovoltaics consumption (so to local power consumption) only. Therefore, these parameters also optimized the residual load on the main grid node, as the batteries were not permitted to charge any power from outside of their micro-grid over the superior transmission grid operator because of their cost limits.

We initialized the battery storage in all scenarios with 50 % charging state, the first electric vehicle with 60 % and the second one with 20 %. Moreover, the battery storage was limited at 1 kW peak loading power, the first electric vehicle at 3 kW, and the second electric vehicle with 2 kW. These settings were chosen purely randomly but they characterized the simulation runs we launched.

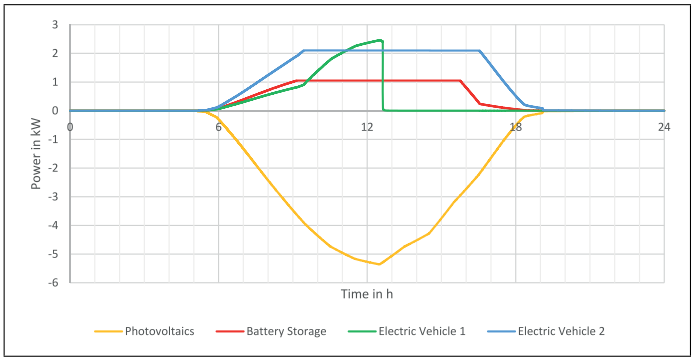

Fig. 5. Charging by ratio sharing on limited available power.

In Figure 5, we can see how the multiple agents charge by sharing their ratio calculated from their specific charging state (see *sec.* IV-D5), as soon as the photovoltaics provides power. We can see how the second electric vehicle, having the lowest state of charge, is taking the greatest share of the available power, in comparison to the other two batteries. Moreover, we can also see the power limits of the battery storage and the second electric vehicle, as well as the automatic rebalancing taking place when these limits are reached, that adds the difference between the requested and the limited power of these consumers to the consumers still having capacities left (the first electric vehicle here). Finally, at around 13:00 hours in virtual time, the first electric vehicle is fully charged, and the photovoltaics starts to feed the remaining power into the main grid. This can be seen in Figure 7 in the Ratio curve. In the end, except the second electric vehicle, all batteries could be fully charged while never using power from the main grid.

In Figure 6, the exactly same simulation scenario with unchanged parameters is managed by priority override (see *sec.* IV-D4) with ratio share as fallback when priorities would

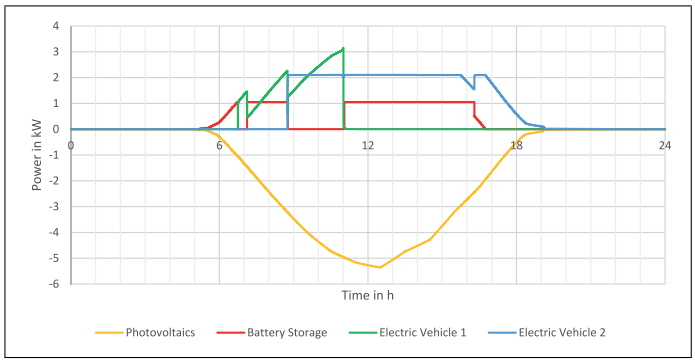

Fig. 6. Charging by priority override on limited available power.

be equal. We can see how a battery that was assigned power—which happens due to "first come, first turn on"—self-sets a priority that decays according to formula (1), permitting other agents to turn on and self-set a higher priority then later. This is why the devices override each other one after another and sometimes, when they both turn on at the same time, also overlap within their assigned capacities for a short period of time. In the end, except electric vehicle two, all batteries could be charged again.

Note that, as already stated in Section IV-D2, during the times of turning on and off, there is a chance of resource conflicts (also called overlapping). This can be seen in Figure 7 in the short orange spikes matching the power curve turnovers of Figure 6. This did not happen during ratio rebalancings, as the difference between priority and ratio here is that the self-set priority changes abruptly directly *after* having received enough power for either turning on or off, and that the ratio is calculated independently of the current power on status. These abrupt changes in the residual load can be seen as spikes, whereas gradual rebalancing cannot be seen.[2]

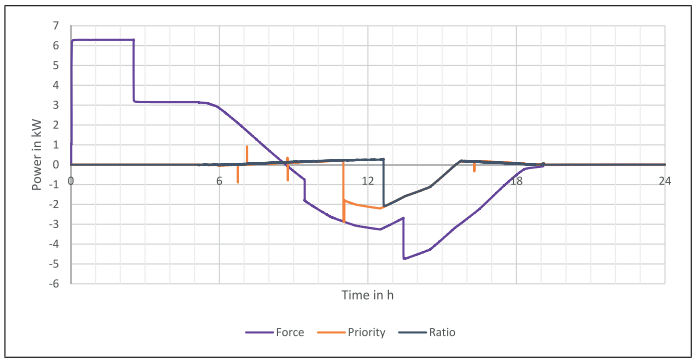

Fig. 7. Residual Load on the main power grid node, additionally compared with forced loading behavior.

In Figure 7, beside the residual loads corresponding to Figure 5 and Figure 6, we also added a forced load residual power curve. There, all three batterie agents were set to forcefully load, no matter the price, no matter the source of the power. The same would have happened when we set the price limit over 0.25 € per kWh which would have permitted the agents to load off the grid, which itself has virtually unlimited power.

The Force curve shows the first electric vehicle being fully charged at around three o'clock in the morning, the battery storage at around half past eight with the photovoltaics already feeding, and the second electric vehicle finishing at around half past two in the afternoon.

As we can see, a forced load would not change much with respect to the balance sheet of supplied *vs.* consumed energy, but the residual loads were much higher, which is more expensive for the grid operators and therefore for the company using the power. With our cost-optimizing agents working according to ratio share and priority override principles, we could reduce the power load on the main grid by several orders of magnitude in this simple simulation already.

Of course, with forced load, the electric vehicles and the battery storage were charged earlier, however without any calendric events being set in our simulation, this early charging was not necessary while only more expensive (for the company itself and unnecessarily for the grid operator that, in order to provide excess power on demand, finds himself in need of transmission line extensions, investments in virtual power plants, *etc.*).

In the figure, we can also see that the ratio distribution was (however by coincidence) overall better than the priority distribution in our two runs as it had no resource conflict spikes. In its algorithmic implementation according to the subsidiary principle, the only one to be able to share the power is the distributing agent (as opposed to the consuming agents). In comparison to that, the priority overriding algorithm, where each agent self-sets its priority, was worse, because as seen in Figure 6, the agent of the first electric vehicle with higher power limit in comparison to the other two agents charged during a time where there was not enough power for all batteries available. If the battery with high limit would have charged later and let the agents with lower limits come first, it could have let the slower agents, that need more time to charge, grant this time and charge later, when the photovoltaics would have had more power than needed by the batteries.

Nevertheless, this decision was not possible at the time when the first electric vehicle decided to set a high priority, because no agent had a projection about the future power availability development, and maybe the power at seven o'clock would have been the maximum of that day (as it could have been, *e. g.*, on a very cloudy day). So, barring knowledge about the future, our proactive approach optimized within its theoretical limits.

VI. Conclusion

To conclude, we have introduced a system which contributes to the endemic energy infrastructure by individually optimizing cost and energy flows through a decentralized, proactive, agent-based, framework.

[2]The spikes become shorter and ultimately negligible when the simulation time approaches real time, reducing the relative communication delay.

We also could show that an entirely decentralized, virtual, market based on multi-agent systems in JADE according to the FIPA standard is able to perform a cost-optimization with stateful, proactive, plans, and various power distribution algorithms, by simulating a simple scenario with SimulationX.

It was also possible to prove the decentralized system approach to be fully able of locally producing and consuming power in a cellular approach according to the subsidiary principle, which effectively disproves the necessity for frequent and pervasive grid operator interventions, transmission network expansions, and for centralized virtual power plants, too.

VII. Future work

Nevertheless, for even better cost-optimization, we need more knowledge about the future, such as forecasts of prices and/or renewable energy power source outputs, so that our agents become able to deciding whether further waiting improves the overall costs or not.

Moreover, further testing and expansion of the decentralized algorithms are needed for evaluation the approach in other, more comprehensive, simulations and real grid scenarios. We are planning to roll-out our solution to a local micro-grid already, with one main grid power node, various flexible and inflexible consumers, and at least one charging station having two charging points for electric vehicles.

Also, the legal rammifications about decentralized decisions are not fully clear, especially because actions are not deterministic in a parallel and independent calculation. The data accumulation in a decentralized system is what probably may not be compliant to management understandings of a centralized documentation and ticket approach, but as we already mentioned the dangers of central surveillance (see *sec.* I), we see no benefit in emulating a cetralized approach within the new reality of the endemic energy infrastructure and instead recommend embracing its decentralized nature.

For most optimal cost simulation, real price models must be implemented inside the agents for being able to announce correct real-world costs needed for billing and taxes. For the moment, we only proved that the trading depending on prices itself works, no matter the source of the prices.

Additionally, user interfaces, mobile applications (*e. g.*, for setting calendric events that would change priority and ratio needs), integration into enterprise resource planning (ERP) environments in industrial settings, demand-side management (DSM) integration in production enviroments, and monitoring services for administration of the MAS and its interfaces (*e. g.*, for exchanging agents and devices during system operation) need to be provided.

We believe that our MAS based approach provides an ideal structural core for a future, integrated, energy system.

References

[1] R. Speh *et al.*, "Der Zellulare Ansatz," VDE/ETG, Studie, 2015.

[2] BMWi. (2015) Was ist eigentlich ein "Virtuelles Kraftwerk". [Online]. Available: https://www.bmwi-energiewende.de/EWD/Redaktion/Newsletter/2015/13/Meldung/direkt-erklaert.html

[3] D. Steber, P. Bazan, and R. German, "Swarm - increasing households' internal pv consumption and offering primary control power with distributed batteries," in *Proceedings of the 4th D-A-CH Conference on Energy Informatics - Volume 9424*, ser. EI 2015. New York, NY, USA: Springer-Verlag New York, Inc., 2015, pp. 3–11. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-25876-8_1

[4] Bundesnetzagentur, "Netzausbau - Netzentwicklungspläne," April 2016.

[5] U. Greveler, B. Justus, and D. Löhr, "Hintergrund und experimentelle Ergebnisse zum Thema 'Smart Meter und Datenschutz'," Fachhochschule Münster, Technical Report, 2011.

[6] BMWi, "Gesetz zur Digitalisierung der Energiewende," 2016.

[7] Y. Shoham, "Agent-oriented programming," *Artifical Intelligence*, vol. 60, no. 1, pp. 51 – 92, 1993.

[8] Y. Shoham and K. Leyton-Brown, *Multiagent Systems*. Cambridge University Press, 2009.

[9] M. Wooldridge, *An Introduction to MultiAgent Systems*. John Wiley & Sons Ltd, 2002.

[10] S. Lehnhoff, *Dezentrales vernetztes Energiemanagement*. Vieweg+Teubner, 2010.

[11] T. Eymann, "AVALANCHE - Ein agentenbasierter dezentraler Koordinationsmechanismus für elektronische Märkte," Dissertation, Universität Freiburg, 2000.

[12] T. F. for Intelligent Physical Agents. (2005) FIPA. [Online]. Available: http://fipa.org/

[13] T. I. SpA. (2001) Jade - java agent development framework. [Online]. Available: http://jade.tilab.com/

[14] F. L. Bellifemine, G. Caire, and D. Greenwood, *Developing Multi-Agent Systems with JADE*. Wiley, 2007.

[15] T. F. for Intelligent Physical Agents. (2002) FIPA Contract Net Interaction Protocol Specification. [Online]. Available: http://www.fipa.org/specs/fipa00029/SC00029H.html

[16] ——. (2002) FIPA Standard Specifications. [Online]. Available: http://fipa.org/repository/standardspecs.html

[17] D. P. Reed, "Naming and synchronization in a decentralized computer system," Dissertation, Massachusetts Institute of Technology, 1978.

[18] P. A. Bernstein and N. Goodman, "Concurrency control in distributed database systems," *ACM Comput. Surv.*, vol. 13, no. 2, pp. 185–221, Jun. 1981. [Online]. Available: http://doi.acm.org/10.1145/356842.356846

[19] T. I. SpA. (2010) Jade tutorial - application-defined content languages and ontologies. [Online]. Available: http://jade.tilab.com/doc/tutorials/CLOntoSupport.pdf