International Journal of Smart Grid and Clean Energy

A conceptual solution for integration of EV charging with smart grids

Slobodan Lukovic*, Bojan Miladinovic

Faculty of Informatics – AlaRI, University of Lugano, Lugano 6900, Switzerland

Abstract

Coordinated charging of electric vehicles (EVs) represents one of important aspects of future smart grid implementation. The main focus of the present work is on designing a conceptual architecture as a solution for managing (mobile) electric storages utilization. The solution requires a holistic approach which takes into consideration all actors in power system management. Requirements and use-cases are specified, structural and behavioral models are developed and presented. The ICT (Information and Communication Technologies) model of the architecture that supports smart charging with dynamic pricing is developed using systems engineering methods. Results are verified by means of system level simulations.

Keywords: Smart charging, electric vehicles, systems engineering, dynamic pricing

1. Introduction

Exiting power production is built to respond to demand and in particular peak demands (so called consumption oriented approach). Still, these peak demands occur for a tiny fraction of system operation which makes this infrastructure not optimally utilized contributing to the costs of the energy and maintenance. Therefore, any improvements in utilization of present power system would reflect in significant savings. Modern methods for shaving demand peaks include demand side management based on price policies which encourage consumers to use energy in periods of low energy demand, while storing of electric power is done in off peak hours. This work proposes a solution to integrate and manage 'mobile' storages in order to improve utilization of power system.

On the other hand, renewable energy resources (RES) are getting widely deployed causing various challenges for the grid [1], [2]. In parallel that massive deployment of EVs is expected, this could cause significant disturbance in power system because they may become significant energy consumers. Even more their charging will have time constraints which could require high rates of charging with unpredictable demands. On the other hand, if EVs' charging would be efficiently managed, they could emerge as a mean of balancing power system (especially when observed as fleet of EVs – aggregated view [3]). In fact, EVs are highly mobile which contributed to energy flexibility in both time and space.

Most of existing works consider only technical challenges of EV integration [1], [4], [5]. Since this is not only a technical problem, but includes many other aspects, it requires interdisciplinary approach such as for instance systems engineering [6] method (coupled with model driven development (MDD)[7]). Based on these proven strategies we developed models in UML (using IBM Rhapsody tool). The methodology greatly facilitated communication in system design process between various partners.

The paper is organized as follows: Section 2 presents applied methodology; Section 3 concerns the context and system requirements; Section 4 studies the structural models of the proposed architecture; Section 5 shows use cases and sequence diagram; Section 6 presents validation results of the proposed conceptual solutions; while Section 7 brings conclusions and introduces future work.

* Manuscript received July 10, 2014; revised August 28, 2014.

Corresponding author: Slobodan Lukovic; Tel.: +041 586664217; *E-mail address*: slobodan.lukovic@usi.ch.

doi: 10.12720/sgce.4.1.1-7

2. Applied Methodology

The electric power system represents a large and complex system of systems that involves various stakeholders and includes different technologies. Commercial aspects such as demand side management are getting more important. Pricing solutions are mostly based on price manipulations in order to maintain energy balance not considering wider context of the system including some sociological issues such for instance if final users would agree to allow system to change prices and/or control consumption.

In order to cope with such problems we rely on systems engineering strategies. Systems engineering has been successfully applied for a long time for design of complex systems such as avionics or software integration [5]. In [8], [9] successful applications of systems engineering for creating referent model of Smart Grids and Virtual Power Systems have been presented. This work relies on the proposed systems engineering approach. System engineering (SE) heavily relies on modelling and simulations. The goal of this work is to develop ICT models of efficient storage management system and to validate results by simulation of the model behavior. Also we use the same languages UML [10] and SysML [11] within tool IBM Rational Rhapsody. SysML is chosen because we are modelling “system of systems” for which it was designed. SE is in general iterative process. Changes made in a model in any of the steps cause changes in other steps. But there is some linearity and order of steps which we describe in the following. It starts with context description and identification of stakeholders. Context description describes system that is developed and its interaction with surrounding environment. Stakeholders are parties that will use the system or that can affect it. Next step represents requirement engineering- gathering of requirements, their structuring tracing, managing and organizing. User requirements are mapped into system requirements. Basing on these requirements, rough idea of the system and envisioned scenarios use cases are developed. Satisfaction of requirements represents main criterion for validation of developed model.

There are two concurrent and correlated tracks of system modelling – structural and behavioral. It means that change in one causes changes in the other. Behavioral design is presented through sequence diagrams, while structural are presented by class (UML) or block (SysML) diagrams.

For the purpose of this work are not designing any specific algorithm. We assume usage of some of existing charging optimization algorithms, billing algorithms, geographical mapping (spatial) algorithms or services, cryptographic algorithms, etc. So we omit this phase from this work. Finally there is a phase of testing and validation. It is done by comparing designed and simulated behavior of components.

During the whole process, we rely on inputs from AlpStore project [http://www.alpstore.info] pilots so that we sometimes illustrate model application by real world examples.

3. System Context and Requirements

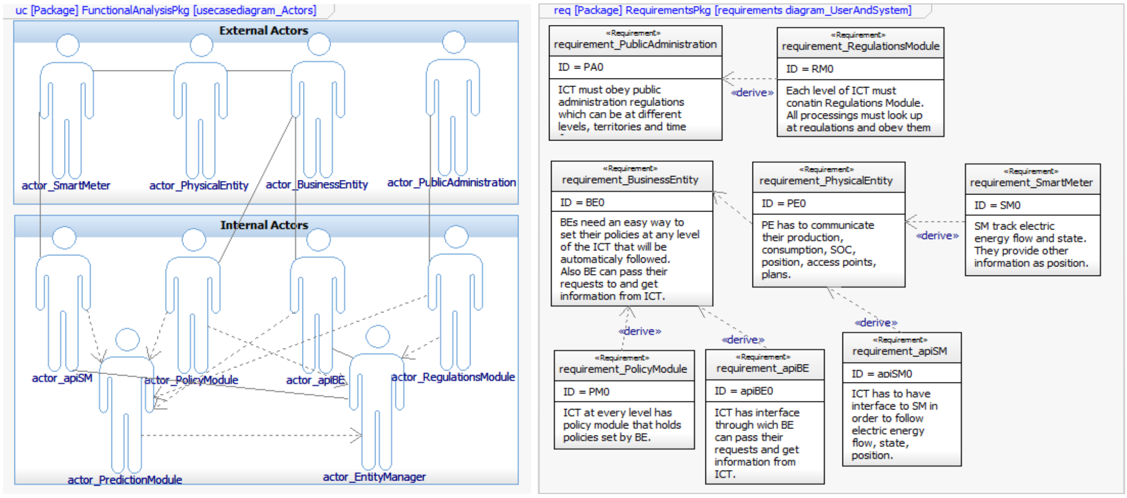
We refer to the components of the system as entities. Attributes and operations of an entity are still not stated until we discuss the context because the entity is designed to meet requirements and use cases implied by the context. We state four groups of external actors/stakeholders, their requirements and implications on the ICT architectures (Fig. 1(a)). In the sequel we explain basic entities.

Physical Entities (PE) – represent real physical components that produce, consume or store electric energy. All physical entities may have attached some storage capacities. For example traditional storages for big power plants are pumped hydro storages. And again EVs can act as storage. These entities can have different demands, State of Charge (SOCs) and production capacities that are changed in time.

Due to intermittent nature of RES their production as well as consumption of most of residential users should be known in advance that’s why predictions are getting important. So we introduce a *Prediction Module* (PM) which comprises all types of predictions needed.

Physical entities are distributed in space and some of them even mobile (e.g. EVs). Data about their position (their access points) has to be communicated through the system. PEs can be aggregated and associated differently. ICT needs to manage physical electric energy flow by communicating with PEs. Interface between the ICT and the power flow/PEs are *smart meters* (SM). Smart meters provide information on how much energy passed in/from PE, state of charge (SOC) of PE, position of the PE’s access points (i.e. charging ports) and actuating - i.e. prevent (dis)charging at certain SOC level. We discuss here only information important for our use cases. *Public Administration* (PA) (and/or authorities) set regulations on energy markets. But at the same time energy market is becoming more and more deregulated. The system must be able to apply all regulations without significant changes in the system and implied investments. We name that part *Regulations Module* (RM). Each level of the ICT system contains this module and all its decisions have to comply with the regulations. So PA is an external actor while RM is its counterpart - an internal actor. Deregulated power systems are opening space for many *business entities* (i.e. companies - BE) comparing to today's relatively small number of monopolistic companies. Examples of BEs can be consumers, producers, traders, EV owners etc. Their *physical entities* (EVs) are coming and going away all the time. The ICT model must support these scenarios. The ICT manages all these data (aggregations, associations, mappings etc.). ICT system has to contain Policy Module for automatic decisions according to BE requests, but it also needs to provide *access points* (APs) for setting demands/offers, time schedule, prices, etc. We have grouped all external actors in these four groups in order not to make diagrams crowded. Considering complexity of the ICT system, not all internal actors are given at system level view.

Actors are shown in Fig. 1 (a). We show four external actors that are not part of the ICT system and as internal actors we show parts of the system that will handle requests from the external actors/stakeholders and provide information to them. Please note that this is the highest level of abstraction for one entity from which the whole system is built. Requirements are gathered during investigation process based on survey of five pilot implementations (in scope of AlpStore project). Formal requirements derivation is extracted from actors (stakeholders). User requirements are then translated in system requirements (shown in Fig. 1 (b).



ICT must obey public administration regulations which can be at different levels, territories and time

Each level of ICT must conatin Regulations Module. All processings must look up at regulations and obey them

BEs need an easy way to set their policies at any level of the ICT that will be automaticaly followed. Also BE can pass their requests to and get information from ICT.

PE has to communicate their production, consumption, SOC, position, access points, plans.

SM track electric energy flow and state. They provide other information as position.

ICT at every level has policy module that holds policies set by BE.

ICT has interface through wich BE can pass their requests and get information from ICT.

ICT has to have interface to SM in order to follow electric energy flow, state, position.

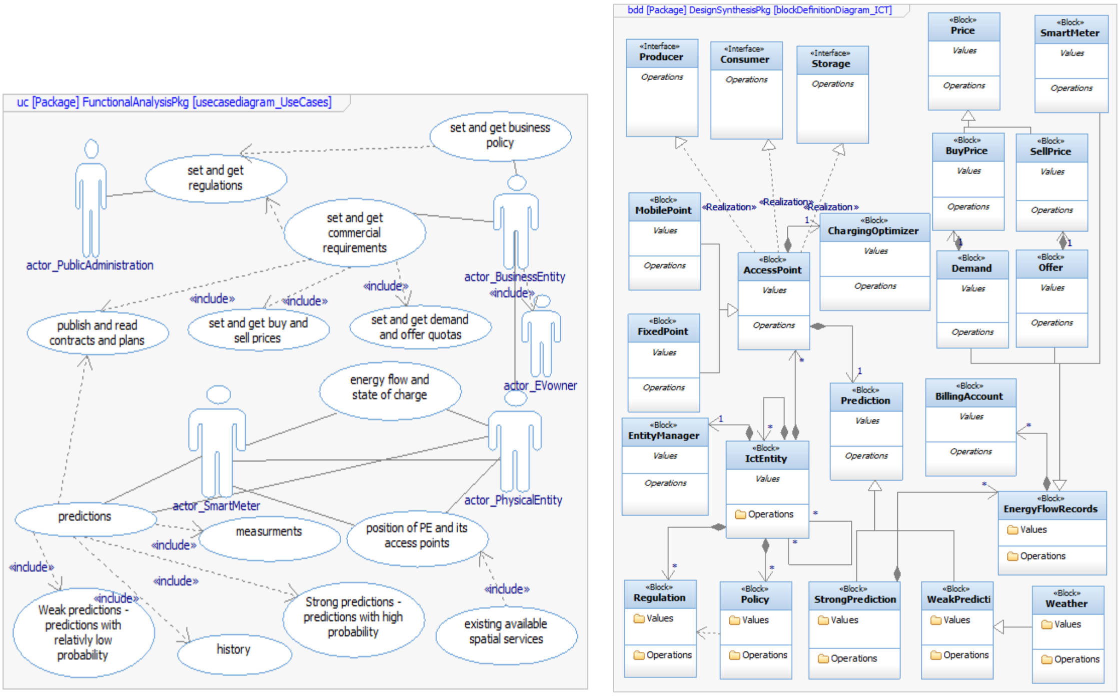
(a) (b)

Fig. 1. Starting systems engineering phase: (a) context description and (b) user requirements description.

Since in this subsection we are defining context of overall ICT system, on Fig. 2 (a) are shown high level use cases (functionalities) by which stakeholders (actors) satisfy their requirements. Fig. 2 (a) shows what functionalities ICT system provides to the stakeholders (external actors). When designing a system we have to satisfy the specific functional requirements as well non-functional ones such as security, privacy and scalability. In the following sections we show structural and behavioral model that satisfies stakeholders' requirements and here stated functionalities (use cases).

4. Structural Model

At first, we'll briefly describe all the modules in the system. *IctEntity* represents basic building block in the structural model (given in Fig. 2 (b)). *Entity Manager* manages one elementary cell. *Regulation* *block* holds regulation issued by public authorities related to that *IctEntity*. All functionalities of *IctEntity* have to comply with regulations. *Policy* block holds policy issued by *business entity* which (usually owners or managers of *physical entities*) define automatic behavior of the *IctEntity*. *Access points* are electric energy flow interfaces to outside world. *IctEntity* may have many access points. *Access point* can act as *producer*, *consumer* or *storage*. We suppose that EV has one slot for charge or discharge of its battery. *Access points* can be mobile (in case of EV) or fixed (access points for charging EVs at one parking). *Charging optimizer* and *Prediction* are presented in every *Access point*. Although its name implies future, *Prediction* also provides information about present state of *IctEntity* (real-time feed). Predications can be strong and weak. The weak ones are those of low reliability. Strongest "prediction" is actually a measurement of a SM. It provides information about current energy flow and SOC. This information can be published as real time feed. Also source of strong prediction are contracted flows, both demands and offers, for electric energy - especially contracted ones. Weak predictions are for example forecasts of weather, traffic, fuel prices etc. This module also keeps valuable history of predictions and their accuracy.



(a)

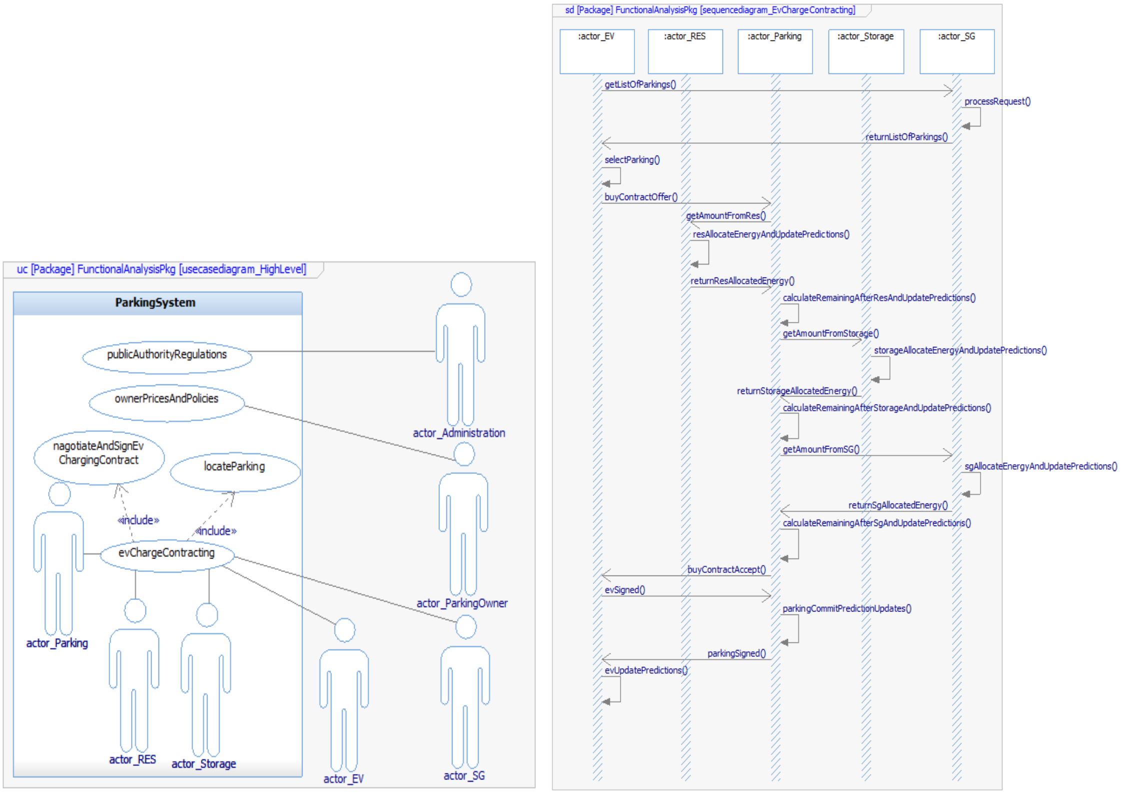
(b)

Fig. 2. Use cases and basic solutions: (a) use cases and actors (b) reference ICT architecture of the solution.

5. Use-Case and Sequence Diagram

As a demonstrator of the proposed solution we apply this architecture to a concrete example (scenario) - EV making contract with a parking for charging (provided with access points for charging of EVs as well as service to sell energy from their batteries). Parking is connected to RES generators and attached to distribution network. Charging point is equipped with SM and associated *IctEntity* supporting its activity.

Fig. 3 (b) shows sequence diagram of the scenario. Basically it shows communication between involved actors with common goal of signing contract between EV and parking for charging EV's battery. Once owner of EV decides to charge its batteries it requires list of available parking slots from a top *IctEntity* on that territory. Request contains parameters such as territory range, charging time frame, maximum acceptable price (owner is aware of prices due to live feed) and amount of energy. Based on these inputs top *IctEntity* returns the list of parkings that can satisfy the request. The EV owner selects most suitable parking. It is important to note here that some parking may offer a lower price but it can be located in remote area . This is the way how fact that energy resources are distributed (DER) is used to balance load not only in time but in space also. After that EV owner sends his offer to a selected parking. The offer holds the same parameters as initial request for the list of parking.


uc [Package] FunctionalAnalysisPkg [usecasediagram_HighLevel]
ParkingSystem
publicAuthorityRegulations
ownerPricesAndPolicies
nagotiateAndSignEv ChargingContract
locateParking
«include»
«include»
evChargeContracting
actor_Administration
actor_ParkingOwner
actor_Parking
actor_RES
actor_Storage
actor_EV
actor_SG
sd [Package] FunctionalAnalysisPkg [sequencediagram_EvChargeContracting]
:actor_EV
:actor_RES
:actor_Parking
:actor_Storage
:actor_SG
getListOfParkings()
processRequest()
returnListOfParkings()
selectParking()
buyContractOffer()
getAmountFromRes()
resAllocateEnergyAndUpdatePredictions()
returnResAllocatedEnergy()
calculateRemainingAfterResAndUpdatePredictions()
getAmountFromStorage()
storageAllocateEnergyAndUpdatePredictions()
returnStorageAllocatedEnergy()
calculateRemainingAfterStorageAndUpdatePredictions()
getAmountFromSG()
sgAllocateEnergyAndUpdatePredictions()
returnSgAllocatedEnergy()
calculateRemainingAfterSgAndUpdatePredictions()
buyContractAccept()
evSigned()
parkingCommitPredictionUpdates()
parkingSigned()
evUpdatePredictions()

(a) (b)

Fig. 3. Parking scenario: (a) Use case diagram and (b) Sequence diagram for EV charging contracting.

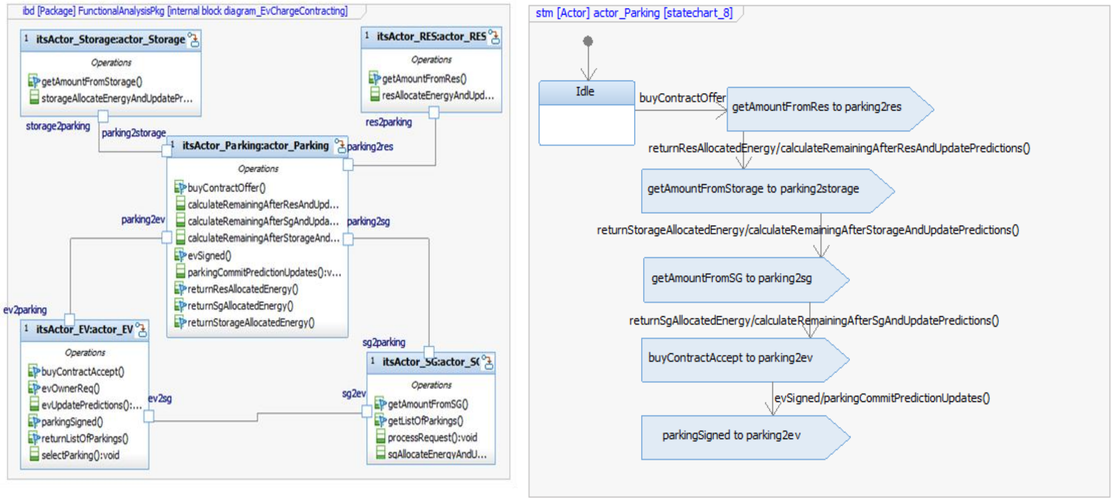
This time request is addressed to a specific selected parking that handles the request by allocating requested energy, calculating exact price, making optimized charging plan and updating its predictions. Although it is not shown on the above diagram, all operations of parking allocation must be aligned with regulations set by authorities and business policies set by owner of the parking. To illustrate what these could be – regulation could define minimum reserve of energy that must not be breached and business policy could be a fee that parking owner takes for each transaction. Parking allocates energy from resources and storages to which it is associated via access points. Its *IctEntity* holds records about offers from the associated resources and storages. It will try to make the best (optimal) offer to the EV owner. In this example we assumed that associated RES offered their production at lowest price, then EV owners that keep their vehicles parked and then a distribution company operates a feeder associated to the parking. In that order parking requires amount of energy for the EV owner. Just as EV owner requires energy from the parking, the same way parking requires energy from associated RESs. For simplicity reasons and without loss of generality we omit most of details that are not essentially important in the communication.

In the following subsection we show detailed implementation of components involved in these actions. In form of internal block diagram we list their attributes and functionalities.

5.1. Internal block diagram and state charts

In Fig. 4 (a) we show *internal block diagram* (IBD) implementing above described scenario. All five actors are mapped onto five high level blocks. Within blocks operations are defined for communication with other blocks or for internal processing. Also ports and connectors for communications between blocks are defined. Please note that these are not all blocks in designed system but only those needed for implementation of above described scenario. Still, these are only the most fundamental operations.

In Fig. 4 (b) we present as an example developed state charts (yet another instrument to present activities in the system) describing communications and operations of the Parking actor.


ibd [Package] FunctionalAnalysisPkg [internal block diagram_EvChargeContracting]
1 itsActor_Storage:actor_Storage
Operations
getAmountFromStorage()
storageAllocateEnergyAndUpdatePr...
storage2parking
parking2storage
1 itsActor_RES:actor_RES
Operations
getAmountFromRes()
resAllocateEnergyAndUpd...
res2parking
1 itsActor_Parking:actor_Parking
parking2res
Operations
buyContractOffer()
calculateRemainingAfterResAndUpd...
calculateRemainingAfterSgAndUpda...
calculateRemainingAfterStorageAnd...
evSigned()
parkingCommitPredictionUpdates():v...
returnResAllocatedEnergy()
returnSgAllocatedEnergy()
returnStorageAllocatedEnergy()
parking2ev
parking2sg
ev2parking
1 itsActor_EV:actor_EV
Operations
buyContractAccept()
evOwnerReq()
evUpdatePredictions():...
parkingSigned()
returnListOfParkings()
selectParking():void
ev2sg
sg2parking
sg2ev
1 itsActor_SG:actor_SG
Operations
getAmountFromSG()
getListOfParkings()
processRequest():void
sgAllocateEnergyAndU...
stm [Actor] actor_Parking [statechart_8]
Idle
buyContractOffer
getAmountFromRes to parking2res
returnResAllocatedEnergy/calculateRemainingAfterResAndUpdatePredictions()
getAmountFromStorage to parking2storage
returnStorageAllocatedEnergy/calculateRemainingAfterStorageAndUpdatePredictions()
getAmountFromSG to parking2sg
returnSgAllocatedEnergy/calculateRemainingAfterSgAndUpdatePredictions()
buyContractAccept to parking2ev
evSigned/parkingCommitPredictionUpdates()
parkingSigned to parking2ev

(a) (b)

Fig. 4. Models of the actors for the parking use case: (a) internal block diagram (b) parking state chart

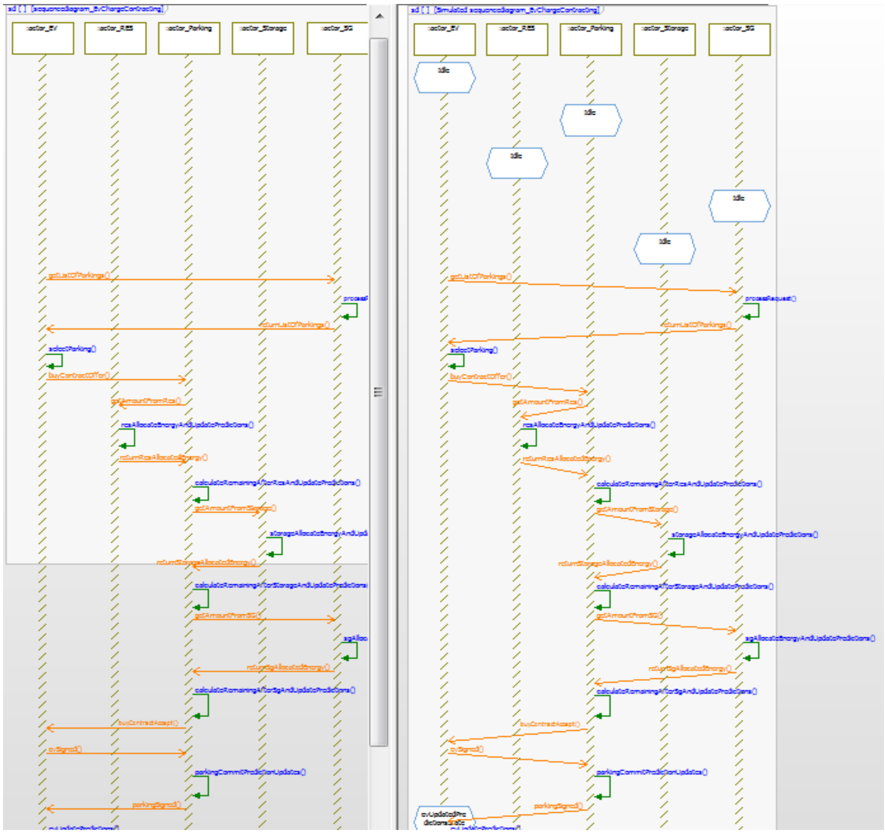

:actor_EV
:actor_RES
:actor_Parking
:actor_Storage
:actor_SG
Idle
getListOfParkings()
processRequest()
returnListOfParkings()
selectParking()
buyContractOffer()
getAmountFromRes()
resAllocateEnergyAndUpdatePredictions()
returnResAllocatedEnergy()
calculateRemainingAfterResAndUpdatePredictions()
getAmountFromStorage()
storageAllocateEnergyAndUpdatePredictions()
returnStorageAllocatedEnergy()
calculateRemainingAfterStorageAndUpdatePredictions()
getAmountFromSG()
sgAllocateEnergyAndUpdatePredictions()
returnSgAllocatedEnergy()
calculateRemainingAfterSgAndUpdatePredictions()
buyContractAccept()
evSigned()
parkingCommitPredictionUpdates()
parkingSigned()

Fig. 5. Comparison of designed and simulated sequence diagram

6. Validation – Simulation of Sequence Diagrams

Validation of a modelled system is done by building the designed model which results in C++ source files compiled into model's executable file. And then running such built execution model and comparing execution result with the designed sequence diagram.

Fig. 5 shows designed sequence diagram on left side and simulated sequence diagram on the right side. It may be seen that we got the same simulated diagram as one we designed (except some states that are added into state charts and shown on simulated diagram). Since diagrams are the same, the designed model can be considered as valid.

7. Conclusions and Future Work

Systems engineering strategies relying on direct inputs from pilot solutions combined with state of the art solutions represent a solid basis for developed of ICT reference model of supporting architecture for mobile storage management. The proposed approach enables system designer to tackle complexity of coordinated EV charging and dynamic pricing.

Starting with identification of stakeholders and their requirements we built use cases and accordingly behavioral and structural models of the possible solution. The proposed architecture has been simulated and validated for the specific parking use case.

Future work will consist of development of BPMN models involving financial issues and resulting in improved executable models. We would set comprehensive simulation environment which could be used for testing and simulation of various scenarios and use cases.

References

[1] E. Sortomme, M.A. El-Sharkawi, "Optimal Charging Strategies for Unidirectional Vehicle- to-Grid" , *IEEE Transactions on Smart Grid,* 2(1): 2011:131-138

[2] O'Gorman R, Redfern MA. The difficulties of connecting renewable generation into utility networks. In: *Proc. Power Engineering Society General Meeting,* IEEE, 2003.

[3] Clement-Nyns K, Haesen E, Driesen J. The impact of charging plug-in hybrid electric vehicles on a residential distribution grid. *IEEE Transaction on Power* Systems, 2010; 25(1): 371-380.

[4] Sundstrom O, Binding C. Optimization methods to plan the charging of electric vehicle fleets. IBM Research, ZH

[5] Ma Z, Callaway D, Hiskens I. A. Decentralized charging control for large populations of plug-in electric vehicles. In: *Proc. of the IEEE Conference on Decision and Control*, 2010:191-195

[7] Milicev D. *Model-Driven Development with Executable UML*. Wiley: 2009.

[6] Weilkiens T. *Systems Engineering with SysML/UML: Modeling, Analysis, Design*. Morgan Kaufmann OMG Press, 2007.

[8] Lukovic S, Kaitovic I, Bondi U. Adopting system engineering methodology to virtual power systems design flow. In: *Proc. First Workshop on Green and Smart Embedded System Technology: Infrastructures, Methods and Tools,* 2010:1-4.

[9] Kaitovic I, Lukovic S. Adoption of model-driven methodology to aggregations design in smart grid. In: *Proc. of the 9th IEEE International Conference on Industrial Informatics,* Indin, 2011:1-6.

[10] Unified Modeling Language. [Online]. Available: http://www.uml.org/

[11] Systems Modeling Language. [Online]. Available: http://www.omgsysml.org/