

# Simulation as a Service: A Design Approach for large-scale Energy Network Simulations

Thomas Preisler, Tim Dethlefs and Wolfgang Renz

Faculty of Engineering and Computer Science,  
Hamburg University of Applied Sciences,  
Berliner Tor 7, 20099 Hamburg, Germany

Email: {thomas.preisler,tim.dethlefs,wolfgang.renz}@haw-hamburg.de

**Abstract**—In the ongoing GEWISS project it is planned to implement a geographical heat information and simulation system. It shall provide a planning and simulation tool for the interlinking of urban development and district heating network development to support the political decision making process in the City of Hamburg. The system shall combine macroscopic and microscopic simulations to a co-simulation system. The simulation as a service approach is presented as a loosely-coupled scalable solution to realize large-scale energy network simulations. It is based on cloud computing technologies for the optimal utilization of computing resources in heterogeneous simulation-infrastructures. This approach can be used to realize simulation systems integrating Multi-Agent System (MAS) based simulations and other simulation technologies. For practical evaluation, two implementation approaches based on a MAS platform as a service-oriented solution will be presented and compared to an approach involving standard web-service technologies.

## I. INTRODUCTION

**S**IMULATIONS take a significant stake in finding intelligent solutions and concepts for future Smart Energy Networks. They are vital to test the suitability of such concepts and solutions and therefore, needed before an actual realization can take place. This regards today's energy grid that undergoes a structural change towards the so-called *Smart Grid* as well as other energy networks as e.g., *heat supply systems*. Besides the realization of control and coordination strategies for the Smart Grid, the uniformed planning of both urban development and energy network development is essential. The holistic contemplation of such a *Smart City* requires the combination of different simulation concepts. Regardless of the domain, it is essential to test operational concepts with respect to their cost effectiveness and to optimize them if necessary [1], [2]. Contrasting to the test of operational concepts, which often requires fine-granular simulations with consideration of microscopic aspects, in terms of planning questions for urban and energy network development macroscopic simulations are of concerns.

The *simulation as a service* approach presented in this paper is related to the *GEWISS* project [3]. The goal of this project is to develop a geographical information and simulation system in order to support the political decision making in terms of an interlinking between urban development and district heating network development. It shall combine macroscopic modeling approaches from urban development regarding the evolution of building structures in a quarter, with microscopic

simulations of the heat energy demand of buildings respectively building blocks. A multi-agent based co-simulation system is envisioned, where the microscopic behavior of single buildings or building blocks will be modeled as agents and the selection of appropriate sub-simulations will be mapped to agent-based goal deliberation. The latter will not be part of this paper. This paper focuses on presenting a possible approach to support large-scale co-simulations of macroscopic and microscopic simulation components, that can be rolled out to a heterogeneous IT-infrastructure. The goal of the proposed approach is to utilize cloud computing technologies and service-oriented design concepts in order to build loosely-coupled simulation systems, that integrate different simulation types and models in a service-oriented fashion. Cloud computing technologies and approaches should be used to form a private cloud on heterogeneous university IT-infrastructure, consisting out of different types of servers, in order to utilize the available computation resource optimally and to integrate slow but cheap computers like, e.g., *Raspberry Pi* single-board computer meaningfully.

The remainder of this paper is structured as follows. Section II describes current cloud computing infrastructures and related simulation frameworks and approaches as well as a related approach in the domain of modeling and simulating the heat energy demand of buildings. In Section III the *GEWISS* project will be described in more detail before the concept of simulation as a service will be described in Section IV. Section V describes and discusses possible implementation alternatives and challenges, followed by Section VI where the implementation alternatives will be evaluated in terms of their scalability. Finally, Section VII concludes the paper.

## II. RELATED WORK

This Section introduces examples for state of the art cloud computing technologies, and describes related simulation approaches and frameworks, as well as related work in terms of the *GEWISS* Project.

### A. Cloud Computing Technologies

Cloud computing [4] is seen as a new approach to IT infrastructure management, facilitating a pay-as-to-go usage model. Computational resources are made available on a demand-driven basis instead of statically dedicated physical systems.

Therefore, the approach minimizes idle times and optimizes the utilization of resources, which leads to a minimization of resource dissipation. The authors of [5] bring forward the argument, that by adapting cloud computing ideas for optimal resource utilization, it is reasonable that existing computers in a company or university network could contribute their spare resources in a private cloud. This approach is picked up to propose the *simulation as a service* concept to realize large-scale simulations as required in the *GEWISS* project, that scale well and support optimal utilization of a heterogeneous IT-infrastructure. Cloud applications can be built on the *Infrastructure as a Service* (IaaS) or the *Platform as a Service* (PaaS) layer. On the IaaS layer, access to the cloud is granted by virtual machines that allow fine-grained control of the software stack and provide low-level aspects like operating systems. On the PaaS layer, a cloud operator establishes a new software layer with a dedicated middleware programming interface, and thus, lower level details are abstracted. PaaS facilitates the development of applications on top of the given platform, but restricts the types of applications to those which are supported by the platform. The *Software as a Service* (SaaS) layer are user-ready applications running in the cloud, which are typically built upon the IaaS or PaaS layer. There are many different approaches and technologies for the implementation and operation of cloud applications. This Section will focus on two different PaaS solutions and describe them exemplarily. PaaS solutions are more suitable for the realization of simulations as a service than IaaS approaches, as they abstract from the operating system level and offer an Application Programming Interface (API) for the realization of scalable applications on top of a cloud infrastructure. The *Mesos* platform [6] offers a thin resource sharing layer, that enables fine-grained sharing across diverse cluster computing frameworks by providing a common interface for accessing cluster resources. *Mesos* focuses on providing shared commodity clusters between diverse cluster programming frameworks like *Hadoop* [7] or *MPI* [8] to improve cluster utilization. Resources are shared in *Mesos* to allow frameworks to achieve data locality by taking turns in reading data stored on each machine. It introduces a distributed two-level scheduling mechanism called *resource offers*. A resource offer encapsulates a bundle of resources that a framework can allocate on a cluster node to run tasks. *Mesos* decides how many resources it offers each framework, while the frameworks decide which resources they accept and which computations are executed. *Mesos* delegates the control over the scheduling to the frameworks. This decentralized scheduling model may not always result in globally optimal scheduling, but according to [6] it performs well in practice and allows the frameworks to meet goals such as data locality near perfectly. While *Mesos* focuses on resource-sharing in data-centers, *JadexCloud* [5] is a PaaS infrastructure to develop, deploy and manage distributed applications with a strong focus on distribution transparency. It is based on a Multi-Agent Systems (MAS) platform and allows to build cloud-based agent applications with service-oriented communication. The key concept is a three layered model, that helps separating

responsibilities and managing complexity. A daemon layer provides a node infrastructure for managing cloud resources. It automatically detects and announces nodes joining and leaving the network. The platform layer on top supports application related management tasks including the automatic deployment of application artifacts on different nodes as well as starting and stopping of components. Finally, the application layer facilitates the application development by providing the API and tools for debugging. The current version of the provided PaaS infrastructure [9] supports the management of non-functional requirements. It allows to define non-functional properties for services that are monitored automatically, e.g., a wait-queue property that counts the request currently waiting to be answered. Based on this non-functional properties the service selection can be automated by defining non-functional requirements like, e.g., finding the service with the smallest wait-queue. Due to its focus on cloud-components offering services (agents), and the automatic deployment of application artifacts to dynamically joining and leaving network nodes, *JadexCloud* seems to be well suited to realize the simulation as a service approach.

#### B. Simulation and Modeling as a Service Approaches

In [10] it is described how simulation software can be accessed as service (SimSaaS) in a service-oriented architecture (SOA) approach. It is stated that carrying out an experiment can be achieved by connecting multiple simulation services to form a work-flow which represents how the experiment proceeds. The authors of [10] state that simulation services differ from regular web services as concepts of time and state are essential, thus, viewing them as stateful services, that treat each service request as a series of dependent transactions that are related to the previous requests as well as the model's current state. Therefore, different communication and synchronization paradigms are described and assessed in terms of their suitability to interconnect stateful simulation services. The approach presented in this paper adopts the ideas from [10] by combining them to a design approach for large-scale energy network simulation with the extension that stateless simulation services may also be suitable for the realization and integration of small sub-simulation models. A survey about current trends and technologies in the field of modeling and simulation as a cloud service is given in [11]. It defines the term Modeling and Simulation as a Service (MSaaS) as a model for provisioning modeling and simulation (M&S) services on demand from a cloud service provider (CSP), that keeps the underlying infrastructure, platform, and software requirements and details hidden from the user. In this case the CSP is responsible for licenses, software upgrades, scaling the infrastructure, and providing grade of service and quality of service as specified in service level agreements. The SaaS model provides access to hosted applications in a cloud environment, allowing users to access services at low cost and scale as needed. This model was extended in [12] to include high-performance hosted simulation and modeling services. The described *Polymer Portal* is a first-generation simulation

and modeling as a service (SMaaS) platform. Contrasting to the approach presented in this paper, where the simulation as a service approach is presented as a modeling approach to build distributed, scalable simulation systems, the approach presented in [12] focuses on ready to use simulation- and modeling-services that users can access for a fee.

### C. Simulation Frameworks and Approaches

There exist many different tools and frameworks that developers can use to build simulations upon. Some of these simulation frameworks focus on specific application domains and thus, support the developers by providing domain specific models and tools, while other frameworks focus on a broader application-domain independent scope and support simulations in general. While application-domain specific frameworks take some of the implementation work from the developer and often allow a faster implementation, they restrict developers in terms of implementational freedom and limit them, if their problem does not fit into the domain specific scope of the framework. Examples for domain specific simulation frameworks are, e.g., *Mosaik*, a flexible Smart Grid co-simulation framework [13] and *RinSim*, a simulator for logistics problems [14]. An example for a general simulation framework is the *Jadex* project [15]. When it comes to providing simulation functionality as a service there are two practicable ways. The first one is to implement the simulation using an application-domain dependent or independent simulation framework (or of course implementing it completely from scratch). In this case, the implemented simulation system will be considered as a black box with defined input and output parameters when it is encapsulated by a service component, that provides the simulation functionality as a service method. An example for this implementation scheme is given in [16] where 4GL Matlab simulation models are encapsulated by service components, to provide their functionality via REST (Representational State Transfer) web-services. The other implementational alternative is to implement the whole simulation with service-based technology. In this case it is possible to provide both, the macroscopic functionality of the simulation itself, as well as the microscopic functionality of single simulation components out of the box, without the need to encapsulate it first. An example for such a simulation system is given in [17], where a Multi-Agent based self-healing resource-flow system was built that used services to provide the functionality of the simulated robots. The standardization of simulation interoperability resulted in the *High Level Architecture* (HLA), an IEEE standard for modeling and simulation [18]. The HLA is a technical architecture developed to facilitate the reuse and interoperability of different simulation systems and assets. It provides a general framework, which can be used by developers to structure and describe their simulation systems and to interoperate them with other simulation systems. But due to its complexity it is hardly used outside the military domain [19].

### D. Modeling Urban Energy and Heat Demand

In terms of modeling the urban energy and heat demand the approach presented in [20] and [21] is related to the undertaking in the *GEWISS* project. Both projects aim at modeling the energy respectively heat demand of urban buildings and building structures. But while the *GEWISS* project will follow a Multi-Agent based simulation approach, modeling the building and building structures as microscopic agents the approach presented in [20] and [21] uses 3D city models to calculate the heat demand of whole district areas. It uses the OGC Standard *CityGML* [22], an open, multi-functional model that can be used for geospatial transactions, data storage, and database modeling, for the modeling of 3D buildings. Based on the surface of the 3D models of whole building blocks their energy and heat demand is calculated as a macroscopic approach.

## III. INTRODUCING THE GEWISS PROJECT

The requirements for the simulation as a service approach are derived from the *GEWISS* project. In this project a geographical heat information and simulation system will be developed to support the analysis of heat demand and urban development for the City of Hamburg (Germany). The goal is the development of a simulation tool (framework) that supports the interlinking of urban development and district heat network development by providing a planning tool for different scenarios.

### A. Project Description

In order to utilize the potential of climate protection and to allocate available resource with high cost-effectiveness, the strategic heat planning must be interlinked with urban development projections (and vice versa). In this case, an abstraction of the spatial location of existing buildings is for obvious reasons not possible. Aspects such as the conversion of urban areas, infill, redevelopment or demolition as well as renovation of buildings or building groups should be tailored to the local available heat sources. Conversely, this means that the grid-based heat supply should be planned with respect to existing and future building stock. For this indentation it is necessary that data will be collected and analyzed with respect to spatial location. The combination of data and analysis to the heat demand and urban development should take place in a geographic information system (GIS). The goal of the *GEWISS* project is to extend such a GIS with geographic heat information and simulation capabilities in order to provide planning assistance for future developments.

Therefore, a framework should be developed that allows users to simulate the medium and long-term development of a heat supply system. Here, the specification of both external conditions (e.g., the development of energy prices) as well as municipal political measures (e.g., regulations or financial incentives for certain structural measures) should be supported and also the analysis of their effectiveness. The goal hereby is to explore business models as well as different urban development plans in a systematic fashion. This should help

the City of Hamburg to find measures to achieve an optimal solution to match environmental protection objectives in the heating sector. From the overall project goal a set of scientific and technical working tasks are derived, which are relevant for the functional and non-functional requirements of the simulation and information system to be built:

- The strategic conception of heat planning and urban development should be integrated, so that energy and emission reductions can be implemented efficiently.
- All areas of a heating system (generation, transmission, storage and demand) should be considered through a participatory modeling and simulation approach that involves all relevant stakeholders.
- As a core, an agent-based simulation with visualization capabilities should be realized.

### B. Simulation Requirements

A series of functional and non-functional requirements unfolds from the mentioned project goals. For example, the design of the simulation entities must consider all areas of the heating system, taking into account production, transport, storage and demand. This means, that the simulation tool must be able to support large-scale simulations with more than 100,000 simulation entities (agents). The agents that are designed to model buildings or building blocks may be required to support a rule-based temporal evolution. The considerations of important stakeholders implies on the one hand influences on the design of simulation scenarios, but allows on the other hand the scenario-dependent modeling of rule-based descriptions for the automatic simulation through autonomous agents. The validation of these descriptions is to be ensured against separate micro-simulations. In order to fulfill this requirements a simulation tool that supports the realization of co-simulations based on a service-oriented middleware is envisioned. Co-simulation is a prominent method to solve multi-physic problems. Such simulations combine well-established and specialized simulation tools for different fields [23]. In the context of the *GEWISS* project a co-simulation is distinguished by a distributed simulation as well as a distributed modeling (defined by the usage of different modeling tools). A distributed simulation in this context is understood as the integration of multiple, self-contained simulations. Additional information about the strong impact of non-functional requirements on large-scale systems can be found in [24].

## IV. THE CONCEPT OF SIMULATION AS A SERVICE

The concept of the simulation as a service approach derives from the simulation requirements of the *GEWISS* project. The goal is to use cloud computing ideas and technologies to build loosely-coupled large-scale simulation systems. By adapting the SaaS concept to the simulation domain, simulations will be provided as services in a cloud infrastructure. The conceptual architecture and approach is depicted in Fig. 1. The idea is that simulations will be contained within a Simulation Service Component. This component encapsulates the functionality of

the simulation and provides a service interface to execute it in a service-oriented fashion. This approach unifies the invocation of different simulation types while also allowing the combination of different implementation models and programming languages. Fig. 1 exemplifies two different simulation types that can be combined and invoked in an unified way following this approach.

A common modeling technique in the engineering domain is the usage of fourth generation languages (4GL). These are programming languages focusing on rapid application development. The expression was coined by [25]. Recently, they have gained new attention through the introduction of model based software development [26]. Examples for 4GL are MATLAB, Simulink, Modelica, GAMS (General Algebraic Modeling System). These languages are used in many research projects for rapid prototyping as well as the realization of models or algorithms, e.g., for demand side management and the integration of regenerative energy resources into smart grids [27]. Although they reduce the overall development effort through the usage of comprehensible application-oriented paradigms, the integration of such 4GL models into large-scale simulations is a considerable challenge [16]. Encapsulating their functionality by a Service Simulation Component allows to execute them in an unified way, while also enabling their execution in a scalable cloud infrastructure.

The other depict simulation type is a MAS based simulation. They are well-known and widely spread [28]. Here, different types of agents are used to model the reality and to examine different problems. Such simulations exhibit different grades of complexity and inter-simulation dependencies. If a simple MAS based simulation that is executed on a single network node is considered, it can easily be encapsulated by a Simulation Service Component as a whole. This will provide an unified service-oriented way of invocation and control the execution and the life-cycle of the simulation as a whole. If a more complex, distributed MAS based simulation is considered, a possible integration approach is to encapsulate a dedicated starting component with a Simulation Service Component, which starts the distributed application and collects the results to return them.

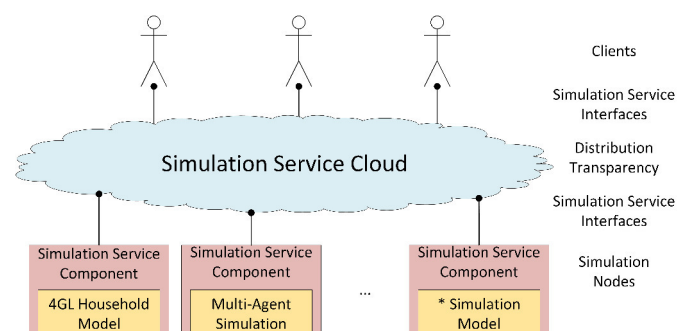


Fig. 1. Simulation as a Service Conceptual Architecture and Approach

Simulations that are encapsulated by a Simulation Service Component can either be called by clients directly if they

require a specific simulation functionality, or more complex simulations can be composed out of multiple simulation services. In this case a simulation component can use the services offered by other simulation components, if they require the results provided by these simulation components as a sub-simulation. An example for this use-case is the realization of a mesoscopic co-simulation where a macroscopic simulation requires results from microscopic simulation components. This will be the case in the *GEWISS* project, where questions regarding the development of district heat networks shall be examined with regards to macroscopic simulations regarding urban development and microscopic simulations regarding the heat demand of households. If the composition of different simulation services is of concern, approaches like the Web Service Business Process Execution Languages (WS-BPEL) respectively the Business Process Model and Notation (BPMN) for the orchestration of web-services might be adoptable to the simulation domain.

Basically there are two main paradigms for the composition of simulation services. Fig. 2 depicts the hierarchical composition and execution of simulations as an UML sequence diagram. It illustrates the execution of a simulation *A*. During its execution, the simulation requests services of the sub-simulations *B* and *C*, while sub-simulation *C* requires the results from sub-simulation *D* in order to answer the request from Simulation *A*. The hierarchical structure depict in Fig. 2 resembles the envisioned structure of the simulation system in the *GEWISS* project, where a main simulation (macroscopic) requires results from microscopic sub-simulations. Thereby, the sub-simulations are only active when their simulation services are called by the main simulation. The simulated time in this case can either be managed by the encapsulating root simulation and passed on to the sub-simulations or each sub-simulation can handle it's own time model, which allows to model interacting simulation with different time-scales (microscopic and macroscopic).

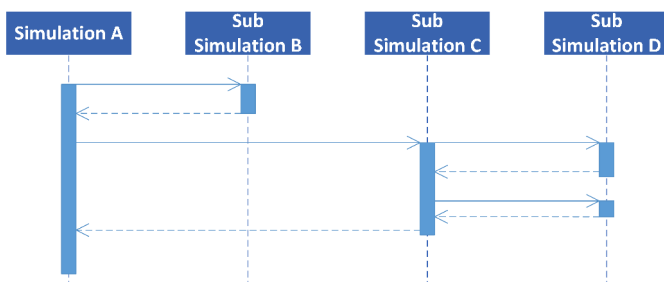


Fig. 2. UML sequence diagram: Hierarchical composition and execution of simulation services.

A different composition paradigm is depict in Fig. 3 where two simulations are executed parallel while exchanging data. This paradigm resemble the one described in [10] where it is stated that simulation services differ from web services as they have to be considered as stateful services, that treat each service request as a series of dependent transactions that are related to the previous requests as well as the model's current

state. Regarding this composition paradigm of interlinked simulations it has to be ensured that their time model is synchronized, so that events are processed in a correct order.

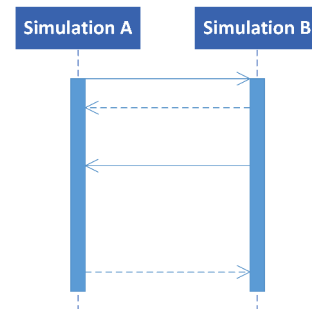


Fig. 3. UML sequence diagram: Parallel composition and execution of simulation services.

## V. IMPLEMENTATION ALTERNATIVES AND CHALLENGES

In general it is reasonable to facilitate established service-oriented technologies to provide simulations as a service. If distribution transparency, scalability and robustness properties should be adopted for the provided simulation services, it is reasonable to facilitate established cloud computing technologies as well. If the simulation is considered as a black box and not implemented with explicit cloud distribution in mind it might be more feasible to use the IaaS approach to abstract from the physical hardware in order to achieve the previous mentioned cloud computing properties. In this case the IaaS approach excels the PaaS approach, because the later one requires the application to be built on top of API of the according platform. Although it is also possible to encapsulate a black box simulation with a component regarding the API of the PaaS platform, but in this case the simulation does not profit from the distribution, scalability or robustness properties of the PaaS platform. So if an already existing simulation system should be provided as a cloud simulation service, a solution built upon a IaaS platform is advantageous. If the simulation is built from scratch or the simulation model is a simple one with a small footprint, it is beneficial to built it against the PaaS API respectively to encapsulate the simple model with a component that is built against the API.

The next Section will provide an evaluation of three different implementation alternatives and evaluate them in terms of their scalability. A case study was ventured, in order to show how a simulation as a service component could be built based on different technologies. A 4GL-model simulating the energy demand of a single household equipped with an electrical heater was used for this case study. In the following, the three implementation alternatives are described briefly before they are evaluated in the next Section.

**REST Webservices:** The first implementation follows the approach presented in [16] and encapsulated the 4GL-model with a REST web-service. In order to provide scalability, a load balancer was connected upstream to distribute the request based on a round-robin scheduling mechanism among multiple

simulation nodes.

**Jadex NFP Service Selection:** The second implementation facilitated the approach for elastic component-based cloud applications presented in [9]. This approach introduces the concept of service selection based on non-functional properties. Again the 4GL-model is encapsulated by a service component. In this case the service is equipped with a non-functional property observer that observes the length of the wait queue of this service. If the simulation service is going to be requested by a client, a non-functional property evaluator evaluates the mentioned waiting queue size property and selects the best service (lowest number of waiting clients). Thus, an optimal distribution of client calls is achieved.

**JadexCloud Service Pool:** The third implementation uses the *JadexCloud* PaaS middleware [5]. Like other PaaS solutions it simplifies the construction of a cloud application by providing common base abstractions, tool sets and an API. The *JadexCloud* infrastructure has two main advantages for the development and deployment of distributed application. Any service offered by a component can be made available in a platform-controlled service pool. The utilization of the service pool deliberates the developer from starting and handling the life-cycle of the service-components himself. The service pool will start new service-components automatically if new resources are needed to handle incoming requests. Therefore, the service pool acts as a proxy handling all incoming service-requests and forwards them to service-components with free resources. The usage of the service pool, does not only simplifies the development of scalable services but also supports the dynamical deployment of service-components to remote platforms. If the service pool detects new *JadexCloud* platforms in the network range, it automatically deploys the governed service-components on this platforms and starts new instances on the remote platform, in case such are required to handle incoming service requests. So following this implementation scheme, the service component encapsulating the 4GL-model was equipped with a service pool to handle the automatic deployment on a cluster of *JadexCloud* nodes and to handle the life-cycle and load-balancing in order to achieve scalability.

## VI. SIMULATION AND EVALUATION OF IMPLEMENTATION ALTERNATIVES

In order to evaluate the three proposed implementation alternatives in terms of their scalability three implementation prototypes were realized. Each of the three simulation setups will be described briefly before the results of the scalability analysis will be discussed. The simulation setup for the implementation based on REST web-services (1) is depicted in Fig. 4. Similar in all three setups is a client (a HTTP web-service client in this case) that places  $n$  parallel requests. For this example, the 4GL-model simulating the energy demand of a single household is encapsulated by a REST web-service. The service is deployed to five homogeneous workstations running *Apache Tomcat* 7<sup>1</sup> as an application container. On one of these

workstations the *Apache HTTP Server*<sup>2</sup> equipped with the *mod\_jk*<sup>3</sup> Tomcat connector is used as a load-balancer. It uses a round-robin scheduling mechanism to distribute the calls to the simulation services in an equal way. For the purpose of testing the scalability and distribution properties only one service-component was deployed on each of the network nodes.

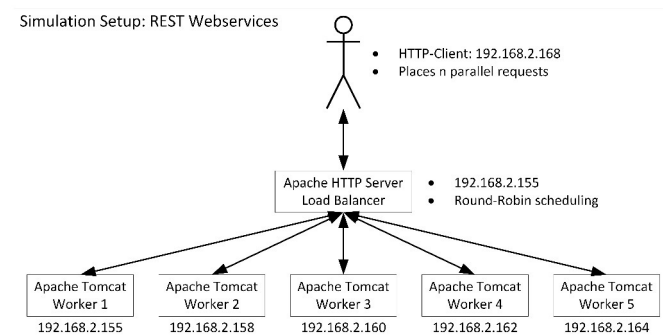


Fig. 4. Simulation Setup: REST Webservices.

The simulation setup for the service selection based on non-functional requirements (2) is depicted in Fig. 5. Again a client places  $n$  parallel service calls. In this setup the simulation service encapsulating the 4GL-model is realized as a *Jadex* service and is equipped with a non-functional property monitor, that monitors the size of the service's wait-queue. For each of the service calls, the client queries the service monitors and uses a non-functional property evaluator to evaluate them with regards to quality criteria. The best service (lowest number of waiting requests) is selected and then called directly. Thereby, an optimal degree of capacity utilization is achieved as it is ensured that always the service with the lowest capacity utilization is ensured.

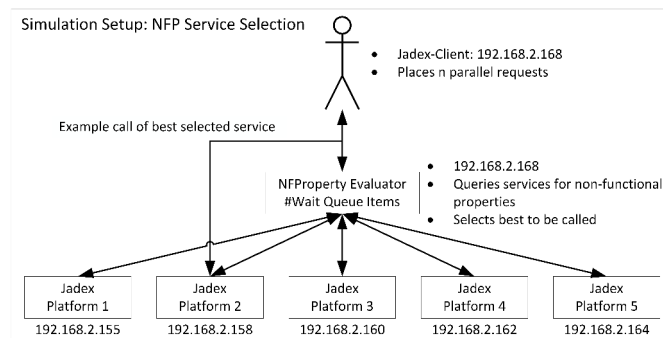


Fig. 5. Simulation Setup: NFP Service Selection.

The simulation setup for the utilization of a service pool and the *JadexCloud* infrastructure (3) is depicted in Fig. 6. It uses the same client and simulation service component as described before, but contrasting to the previous described service selection based on non-functional properties, a global

<sup>1</sup><http://tomcat.apache.org/> (accessed September 25, 2015)

<sup>2</sup><http://httpd.apache.org/> (accessed September 25, 2015)

<sup>3</sup><http://tomcat.apache.org/connectors-doc/> (accessed September 25, 2015)

service pool is used to act as a proxy between the actual services and the client. The service pool deploys the service code automatically to all five workstations and handles the execution of one service-component on each workstation. Due to comparability reasons with the two other setups, only one service-component was started on each of the *JadexCloud* platforms, although it is possible to configure the service pool accordingly to make use of more components. Comparable to the first simulation setup the service pool use a round-robin based scheduling mechanism to distribute the service requests equally to all pooled services. The service distribution is completely transparent for the clients, as they only interact with the proxy.

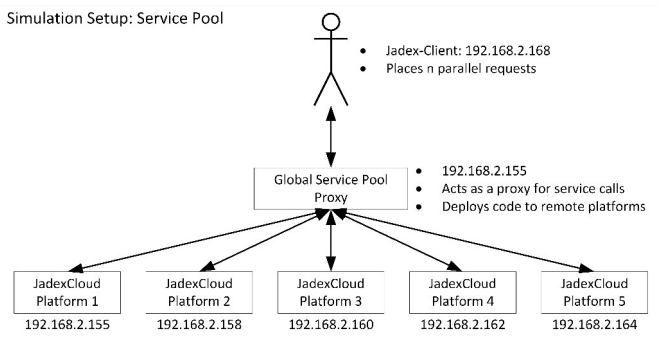


Fig. 6. Simulation Setup: Service Pool.

In order to analyze the scalability properties of the three presented implementation alternatives a number of experiments were conducted. The results of this analysis are shown in Fig. 7. The x-axis depicts the number of parallel calls on a logarithmic scale and the time in milliseconds it took to serve all the parallel requests is depicted on the y-axis (logarithmic scale as well). In order to dampen spikes for each number  $n$  of parallel calls 10 experiments were ventured and the mean values were used for the evaluation. For  $n = 10, 100, 1000$  parallel calls the three implementation alternatives perform quite similar and all three show a linear scalability. This behavior differs for  $n = 10000$  parallel service requests. The implementation utilizing the non-functional property based service selection still scales linear. But both, the implementation using REST web-services and the *JadexCloud* service pool scale considerably worst. Therefore further experiments for these two implementations were conducted. As shown in the graph for  $n = 2000, 3000, 4000$  parallel calls they still perform nearly linear. When the number of parallel calls reaches  $n = 5000$  they lose this scalable behavior. We assume that this deviation between the three approaches is because the according load balancer (Apache HTTP Server or *Jadex* Service Pool) became saturated. If a large amount of parallel calls has to be processed by a load balancer it may become overloaded. Therefore, the load balancer itself could be the bottleneck of the system. This behavior differs from the service selection based on non-functional properties, as in this case the client itself evaluates the degree of capacity utilization

of the offered services and selects the best. Therefore, the required distribution effort is shifted to the client. In order to further investigate this behavior, future work will analyze the scalability behavior of the three implementation alternatives on a System-on-a-Chip cluster. This cluster is characterized by a larger amount of machines with less computational power each in comparison to the workstation cluster used in this experiment. Thus, the scalability and load-balancing properties of the implementation alternatives will become more relevant. Also a combination of the non-functional property selection approach with the service pool approach is envisioned. Even if the scalability properties of the non-functional property selection approach excels the ones from the service pool approach, the dynamically deployment properties of the service pool approach are of great value, especially if an application is executed on dynamic infrastructure with joining and leaving network nodes. Therefore, a combination is envisioned where the default round-robin scheduling mechanism of the service pool is going to be replaced by one selecting the service based on non-functional properties like, e.g., the size of the wait-queue.

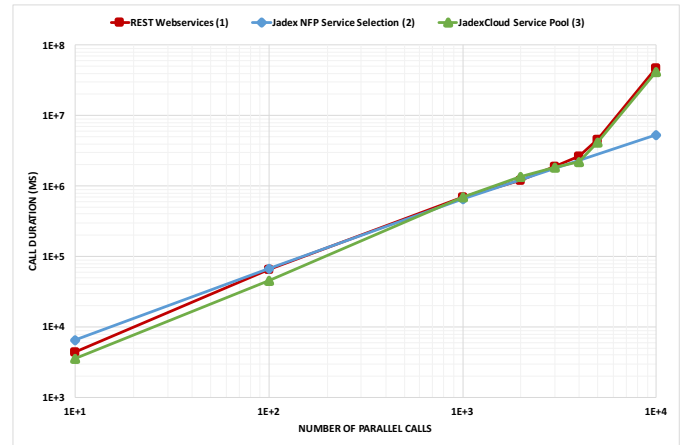


Fig. 7. Results of the Scalability Analysis.

VII. CONCLUSION

In this paper we presented the concept of simulation as a service as a promising approach for the realization of large-scale smart-energy network simulations. It was described how the approach will be used to implement a geographical heat information and simulation system in the *GEWISS* project. The goal is to provide a planning and simulation tool for the interlinking of urban development and district heat network development in order to support the political decision making process in the City of Hamburg. The envisioned system will combine macroscopic and microscopic simulations to a co-simulation system. The simulation as a service approach was presented as loosely-coupled and scalable solution to realize a large-scale simulation system based on cloud computing technology in order to optimal utilize the computing resources of a heterogeneous university IT-infrastructure. Three different

implementation alternatives were described and evaluated in terms of their scalability. The results encouraged the usage of the *Jadex* PaaS platform. A MAS platform that allows to built agent-based cloud application with service-oriented communication and supports the execution and deployment with service pools, service selection based on non-functional requirements and the automatic deployment of components (agents) and services to fluctuating network nodes.

#### ACKNOWLEDGMENT

We would like to thank Lars Braubach and Alexander Pokahr for their input and support while implementing the simulation prototypes based on the *Jadex* platform, as well as our partners in the *GEWISS* project.

#### REFERENCES

- [1] T. Logenthiran, D. Srinivasan, and T. Z. Shun, "Demand side management in smart grid using heuristic optimization," *Smart Grid, IEEE Transactions on*, vol. 3, no. 3, pp. 1244–1252, Sept 2012. doi: 10.1109/TSG.2012.2195686
- [2] D. Rivola, A. Giusti, M. Salani, A. Rizzoli, R. Rudel, and L. Gambardella, "A decentralized approach to demand side load management: The swiss2grid project," in *Industrial Electronics Society, IECON 2013 - 39th Annual Conference of the IEEE*, Nov 2013. doi: 10.1109/IECON.2013.6699895. ISSN 1553-572X pp. 4704–4709.
- [3] T. Preisler, G. Balthasar, T. Dethlefs, and W. Renz, "Servicekomponenten-basierte architektur für mikroskopische und makroskopische simulation der städtischen energieverorgung," in *Proceedings of the German VDE-Congress*, ser. Smart Cities, 2014.
- [4] B. Sosinsky, *Cloud computing bible*. John Wiley & Sons, 2010.
- [5] L. Braubach, A. Pokahr, and K. Jander, "Jadexcloud - an infrastructure for enterprise cloud applications," in *Multiagent System Technologies*, ser. Lecture Notes in Computer Science, F. Klügl and S. Ossowski, Eds. Springer Berlin Heidelberg, 2011, vol. 6973, pp. 3–15. ISBN 978-3-642-24602-9. [Online]. Available: [http://dx.doi.org/10.1007/978-3-642-24603-6\\_3](http://dx.doi.org/10.1007/978-3-642-24603-6_3)
- [6] B. Hindman, A. Konwinski, M. Zaharia, A. Ghodsi, A. D. Joseph, R. Katz, S. Shenker, and I. Stoica, "Mesos: A platform for fine-grained resource sharing in the data center," in *Proceedings of the 8th USENIX Conference on Networked Systems Design and Implementation*, ser. NSDI'11. Berkeley, CA, USA: USENIX Association, 2011, pp. 295–308. [Online]. Available: <http://dl.acm.org/citation.cfm?id=1972457.1972488>
- [7] A. Abouzeid, K. Bajda-Pawlikowski, D. Abadi, A. Silberschatz, and A. Rasin, "Hadoopdb: An architectural hybrid of mapreduce and dbms technologies for analytical workloads," *Proceedings of the VLDB Endowment*, vol. 2, no. 1, pp. 922–933, Aug. 2009. doi: 10.14778/1687627.1687731. [Online]. Available: <http://dx.doi.org/10.14778/1687627.1687731>
- [8] W. Gropp, E. Lusk, and A. Skjellum, *Using MPI: portable parallel programming with the message-passing interface*. MIT press, 1999, vol. 1.
- [9] L. Braubach, K. Jander, and A. Pokahr, "A middleware for managing non-functional requirements in cloud paas," in *Cloud and Autonomic Computing (ICCAC), 2014 International Conference on*, Sept 2014. doi: 10.1109/ICCAC.2014.32 pp. 83–92.
- [10] S. Guo, F. Bai, and X. Hu, "Simulation software as a service and service-oriented simulation experiment," in *Information Reuse and Integration (IRI), 2011 IEEE International Conference on*, Aug 2011. doi: 10.1109/IRI.2011.6009531 pp. 113–116.
- [11] E. Cayirci, "Modeling and simulation as a cloud service: A survey," in *Simulation Conference (WSC), 2013 Winter*, Dec 2013. doi: 10.1109/WSC.2013.6721436 pp. 389–400.
- [12] T. Bitterman, P. Callyam, A. Berryman, D. E. Hudak, L. Li, A. Chalker, S. Gordon, D. Zhang, D. Cai, C. Lee *et al.*, "Simulation as a service (smaas): a cloud-based framework to support the educational use of scientific software," *International Journal of Cloud Computing*, vol. 3, no. 2, pp. 177–190, 2014.
- [13] S. Schütte, S. Scherfke, and M. Sonnenschein, "Mosaik - smart grid simulation API - toward a semantic based standard for interchanging smart grid simulations," in *SMARTGREENS 2012 - Proceedings of the 1st International Conference on Smart Grids and Green IT Systems, Porto, Portugal, 19 - 20 April, 2012*, B. Donnellan, J. A. P. Lopes, J. F. Martins, and J. Filipe, Eds. SciTePress, 2012. ISBN 978-989-8565-09-9 pp. 14–24.
- [14] R. R. van Lon and T. Holvoet, "Rinsim: a simulator for collective adaptive systems in transportation and logistics," in *2012 IEEE Sixth International Conference on Self-Adaptive and Self-Organizing Systems*. IEEE, September 2012. doi: 10.1109/SASO.2012.41 pp. 231–232. [Online]. Available: <https://lirias.kuleuven.be/handle/123456789/361419>
- [15] L. Braubach and A. Pokahr, "The jadex project: Simulation," in *Multiagent Systems and Applications*, ser. Intelligent Systems Reference Library, M. Ganzha and L. C. Jain, Eds. Springer Berlin Heidelberg, 2013, vol. 45, pp. 107–128. ISBN 978-3-642-33322-4. [Online]. Available: [http://dx.doi.org/10.1007/978-3-642-33323-1\\_5](http://dx.doi.org/10.1007/978-3-642-33323-1_5)
- [16] T. Preisler, G. Balthasar, T. Dethlefs, and W. Renz, "Scalable integration of 4gl-models and algorithms for massive smart grid simulations and applications," in *28th International Conference on Informatics for Environmental Protection: ICT for Energy Efficiency, EnviroInfo 2014, Oldenburg, Germany, September 10-12, 2014*, J. M. Gómez, M. Sonnenschein, U. Vogel, A. Winter, B. Rapp, and N. Giesen, Eds. BIS-Verlag, 2014. ISBN 978-3-8142-2317-9 pp. 341–348. [Online]. Available: <http://www.enviroinfo2014.org/>
- [17] T. Preisler and W. Renz, "Scalability and robustness analysis of a multi-agent based self-healing resource-flow system," in *Federated Conference on Computer Science and Information Systems - FedCSIS 2012, Wroclaw, Poland, 9-12 September 2012, Proceedings*, M. Ganzha, L. A. Maciaszek, and M. Paprzycki, Eds., 2012. ISBN 978-83-60810-51-4 pp. 1261–1268. [Online]. Available: <https://fedcsis.org/proceedings/2012/pliks/194.pdf>
- [18] S. I. S. Committee *et al.*, "of the ieee computer society. ieee standard for modeling and simulation (m&s) high level architecture (hla)-ieee std 1516-2000, 1516.1-2000, 1516.2-2000. new york: Institute of electrical and electronics engineers," *Inc., New York*, 2000.
- [19] C. A. Boer, A. de Bruin, and A. Verbracke, "Distributed simulation in industry—a survey part 3—the hla standard in industry," in *Simulation Conference, 2008. WSC 2008. Winter*. IEEE, 2008, pp. 1094–1102.
- [20] R. Nouvel, C. Schulte, U. Eicker, D. Pietruschka, and V. Coors, "Citygml-based 3d city model for energy diagnostics and urban energy policy support," in *Proceedings of the 13th conference of international Building Performance Simulation Association*, 2013, pp. 218–25.
- [21] R. Nouvel, M. Zirak, H. Dastageeri, V. Coors, and U. Eicker, "Urban energy analysis based on 3d city model for national scale applications," in *Presented at the IBPSA Germany Conference*, vol. 8, 2014.
- [22] G. Gröger, T. H. Kolbe, A. Czerwinski, and C. Nagel, "Opengis® city geography markup language (citygml) encoding standard. version: 1.0. 0, ogc 08-007r1," 2012.
- [23] S. Sicklinger, V. Belsky, B. Engelmann, H. Elmqvist, H. Olsson, R. Wüchner, and K.-U. Bletzinger, "Interface jacobian-based co-simulation," *International Journal for Numerical Methods in Engineering*, vol. 98, no. 6, pp. 418–444, 2014. doi: 10.1002/nme.4637. [Online]. Available: <http://dx.doi.org/10.1002/nme.4637>
- [24] A. Garro and A. Tundis, "On the reliability analysis of systems and sos: The ramsas method and related extensions," *Systems Journal, IEEE*, vol. 9, no. 1, pp. 232–241, March 2015. doi: 10.1109/JSYST.2014.2321617
- [25] J. Martin, *Application Development Without Programmers*. Upper Saddle River, NJ, USA: Prentice Hall PTR, 1982. ISBN 0130389439
- [26] S. Beydeda, M. Bock, and V. Gruhn, *Model-Driven Software Development*. Springer Berlin Heidelberg, 2006. ISBN 978-3-540-25613-7
- [27] J. Braunagel, P. Vuthi, W. Renz, H. Schäfers, H. Zarif, and H. Wiechmann, "Determination of load schedules and load shifting potentials of a high number of electrical consumers using mass simulation," in *Electricity Distribution (CIRED 2013), 22nd International Conference and Exhibition on*, June 2013. doi: 10.1049/cp.2013.1240 pp. 1–4.
- [28] S. Moss and P. Davidsson, *Multi-Agent-Based Simulation: Second International Workshop, MABS 2000, Boston, MA, USA, July 2000; Revised and Additional Papers*, ser. Lecture Notes in Artificial Intelligence. Springer, 2001. ISBN 9783540415220