UNIVERSITÄT ZU LÜBECK
INSTITUT FÜR INFORMATIONSSYSTEME

Template for the LaTeX Class “uzl-thesis” for Bachelor's and Master's Theses Written at the University of Lübeck

Vorlage für die LaTeX-Klasse »uzl-thesis« zur Nutzung bei Bachelor- und Masterarbeiten an der Universität zu Lübeck

Masterarbeit

verfasst am
Institut für Informationssysteme

im Rahmen des Studiengangs
Entrepreneurship in digitalen Technologien
der Universität zu Lübeck

vorgelegt von
Johann Mantler

ausgegeben und betreut von
Prof. Dr. rer. nat. habil. Sven Groppe

mit Unterstützung von
Benjamin Warnke

Lübeck, den 1. January 2020

IM FOCUS DAS LEBEN

Eidesstattliche Erklärung

Ich erkläre hiermit an Eides statt, dass ich diese Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Johann Mantler

Zusammenfassung

TODO

Abstract

TODO

Acknowledgements

TODO

Contents

1

Introduction

With the progressive downsizing of computers and the rapid evolution of the Internet, the *Internet of Things* (IoT) was established. While in the classic Internet humans generate information that can be read and processed by other humans, in the IoT it is the things that generate and possibly process information. Things are physical objects enhanced with appropriate electronics such as sensors or actuators. With sensors, things can perceive their environment and with actuators make changes in it. Equipped with means of communication, things can connect to other things or even larger computers via a network such as the Internet and act together. On this basis, innovative applications can be developed for various problem domains that have a lasting impact on our society. The spectrum of IoT encompasses both simpler applications in the home with a few things and more complex applications in the city with many thousands of things. Examples of applications in the home are networked things such as coffee machines or robotic vacuum cleaners, which control themselves based on information from the Internet or are remotely controlled via the Internet. Examples of applications in the city are Bluetooth beacons distributed along the roadside that communicate with a smartphone app of a visually impaired user in order to safely navigate them to their destination.

The need for such applications is evident in the enormous number of things in the environment. Gartner predicted that there will be roughly 50 to 100 billion of them by 2020 [8]. While the amount of data produced by sensors in the home remains manageable, very large amounts of data can accumulate in complex IoT applications. IoT is considered to be an essential driver for Big Data [19]. The cloud has therefore proven its worth as a data sink and location for data processing. More powerful things in the IoT network, such as smartphones, can preprocess the data and thus deliver initial interim results as well as relieve the communication path to the cloud. In addition to managing the amount of data, the semantic use of this data is also a challenge. With the help of ontologies, applications can infer the semantics of the data and act accordingly. The realization of such IoT systems with numerous networked things requires further research activity with regard to infrastructure, data storage and query as well as semantic enrichment of the data.

At the University of Lübeck, the DFG-funded project *Big Data Management for the Semantic Internet of Things* (BigSIoT) was launched, which is being worked on under the direction of Prof. Dr. Sven Groppe from the Institute of Information Systems together with the Institute of Telematics. One approach to addressing the above challenges is a database system that can serve as the backbone for IoT applications. Instances of the database management system (DBMS) run distributed in the IoT network on the more powerful things and, if necessary, on the nearby computers. The instances cooperate with one another as the network topology allows, in order to enable data queries across nodes. Since things are often heterogeneous in terms of their system structure, the database system must be implemented in a platform-independent manner. It must also support data models that help semantic enrichment of the data. In order to support the development of the database system, this master's thesis deals with a simulation of the IoT environment, in which the DBMS instances can be integrated for the purpose of initial investigations.

1.1 Contributions of this Thesis

The first contribution is a systematic comparison of previously published IoT simulators. A literature search is carried out to find existing simulators. The focus here is on those simulators that can simulate data processing in the vicinity of things or directly on things in order to be able to investigate a distributed execution of DBMS instances. Using a list of characteristics, the simulators are analyzed and their performance compared as a test tool for IoT applications.

The second contribution describes a sufficiently precise model for the integration of third-party program code such as DBMS instances or other independent systems in the simulator that can run on selected things in the simulated IoT environment. The term *sufficiently accurate model* means that a model should always be as accurate as necessary and not as accurate as possible. Various implementation concepts are discussed and evaluated for the integration.

The third contribution is the development of a new simulator that implements the model for the integration of code as well as other modeled characteristics. After describing the architecture of the overall system, the implementation of the models is explained. It is important that the simulator can also be used outside of the BigSIoT research project. This is achieved on the one hand by a careful description of the open-source program code and by extension points that allow new behavior to be added without changing existing program code. The effectiveness of the simulator is evaluated in a benchmark.

The fourth contribution is the development of a distributed algorithm that maps the operator graph of database queries to the network topology. TODO Maybe just approaches for developing the algorithm?

1.2 Structure of this Thesis

2

Basics

2.1 Internet of Things

blub [6] ewfdsfdsf [14] definition, evolution and growth, [8] applications like smart farming, smart city etc. ,

Environment

smart home as a simple example, smart and simple devices, protocols

Architectures

The growing application possibilities through IoT has also led to a further development of the architectures of the IoT applications so that they can meet the growing requirements. Cloud Computing as a system architecture was further developed to Fog Computing and then to Edge Computing [11]. The three architectures have in common that they determine the location of the data processing and storage. Depending on the complexity of the application, one of these architectures or a combination of these architectures can be used.

When developing IoT applications, the question of the right back end solution always arises. Sensors are distributed in an area and perceive their surroundings within a certain radius, generate corresponding data and send them to a server via the Internet. An application that runs on, for example, the smartphone may consult the server to the data in order based on the perceived state of each sensor to calculate a total state of the area. The amount of data can, however, greatly increase with the speed of the sensory acquisition, a finer granularity or simply by the number of sensors. Since applications are often only interested in data from a sub-area or certain aspects, the server can consider filtering or abstraction of the data. However, this approach does not scale sufficiently and a server in the back end has quickly reached its performance limit. The traditional way of manually adapting the servers according to the required performance is no longer necessary thanks to the concept of the cloud. With a network of servers, cloud computing offers, among other things, an automatic scaling of the performance and is a more suitable alternative for the back end. Cloud computing solves the problem of scaling in terms of data volume and computing power, but the data must first be sent to the cloud via the Internet. The communication path is then the central component in the distributed system. In general, centralized components in a distributed system lead to a performance bottleneck, to a single point of failure and to a restriction of the autonomy of the components. These disadvantages are particularly severe for IoT systems, which can be highly distributed systems with many thousands of things. Especially for IoT, connectivity to the central cloud is a critical point that has led to a move away from pure cloud-based architectures.

The data of things first have to find their way through the Internet to the remote cloud and the latency increases with the length of the path. Accordingly, a delay must be accepted in the data processing. In addition, a possible mobility of things also promotes the variance of the delay. In general, you want the waiting time between request and response as short as possible. This is even essential for some latency-sensitive applications [18]. Control systems such as smart traffic light systems in the city require certain upper limits in terms of latency for their functionality.

Another obstacle can be bandwidth. Things could generate more data than the network connection allows. This can occur, for example, if the sensors work continuously and in great detail for a precise analysis of their environment, or if the sensors are in an area with poor internet connections. The latter problem is characteristic of intelligent agriculture [15]. In addition to latency and bandwidth, the reliability of the Internet connection also plays a role. Despite the interruption of the connection, IoT applications should continue to function adequately [11].

The solution to these problems is a decentralization of computing and data storage within the IoT environment, so that a centralized communication path to the cloud is no longer required. The first level of decentralization is fog computing. Computing and data storage are carried out on nodes that are located between the cloud and the edge of the network. Typical intermediate nodes are routers in the LAN [11]. But servers that are explicitly interposed for this architecture can also represent fog nodes. These fog nodes have better connectivity to the end nodes in their local network in terms of latency, bandwidth and reliability than the end nodes to the cloud. Even the exchange between neighboring fog nodes can be better. The fog nodes can preprocess, aggregate or filter the received data and thus facilitate the forwarding to the cloud. For latency-sensitive applications, the locally available data may be sufficient so that forwarding to the cloud is not necessary. An exclusive processing in the LAN can ensure the possibly necessary data privacy. Better privacy when forwarding to the cloud can also be achieved by abstraction [9] or other preprocessing by the fog nodes.

The next level of decentralization is edge computing. Now some end nodes are also used for computing and data storage. This is possible with more powerful nodes such as single-board computers [6]. Smartphones and smartwatches are also edge nodes in the IoT [6], because they offer sensors and actuators as well as computing power and storage space. By preprocessing directly at the data source, higher scalability is achieved compared to fog computing. In addition, battery life can be increased because sending data uses more energy than processing it [6]. In addition, portable edge nodes can represent their wearer in the IoT environment [5]. It is characteristic of edge nodes that their knowledge of the network topology does not go beyond the LAN. Edge nodes communicate with their assigned fog node (or routers that are not fog nodes) to reach the cloud or other remote edge nodes.

2.2 Data Management for Internet of Things

definition, Big Data[10]

Semantic Internet of Things

definition, continue the smart home example as use case [9], LUPOSDATE

Databases

p2p, iot [4], hashtables?, multi-platform, LUPOSDATE3000

2.3 Modeling and Simulation

Modeling has a long tradition in research and development. Models help to draw conclusions about reality by showing a simplified representation of reality. Modeling can be used when the real object is too complex to analyze directly. A model is essentially characterized by its simplification. This means that a model is an abstract image of the real object in which only a few aspects that are relevant for the modeler are considered. According to Weisberg [17], a total of three categories of models can be distinguished: concrete, mathematical and computational. The latter models are increasingly used in science [17]. They are described algorithmically to be carried out on computers. The execution is called a simulation and the program that creates the simulation can be called a simulator. The model is related to its original system. This relationship is necessary in order to be able to draw conclusions about the original system. The relationship is shown by properties that the model and original share, which are still shared even after operations [17]. Figure 2.1 illustrates the relationship between a model and its original. For example, a model of an IoT device may be similar to the real device in that it behaves the same way under modeled operations as the real device behaves or would behave under real operations.

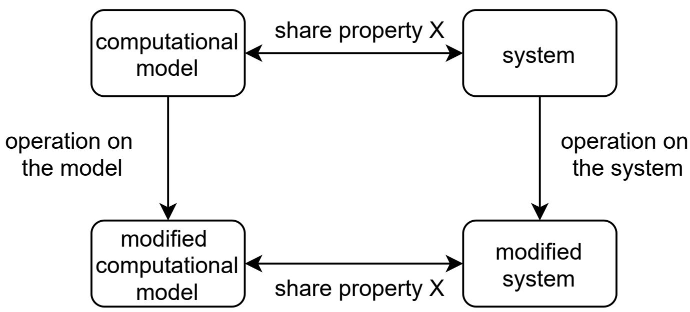

Figure 2.1: Relationship between a computational model and its system.

Benefits of Modeling and Simulation

Computer simulations have proven to be a suitable technique for the study of various aspects of a complex system [2]. They can be used at many points in the software development process. In the integration test, the interaction of the components of the system is tested [13]. Because not all components are available for integration at the beginning of development, a strategy is required that defines the sequence in which the components are implemented [13]. Dependencies on components that do not yet exist can initially be resolved using stubs [13]. For a deeper analysis of the correct integration, a model can be used as a stub. During the system test, the system is tested as a whole [13]. The runtime environment should come as close as possible to the later productive environment [13]. If it is too complex or not possible to use the production environment, modeling this environment can help [13]. The simulator can be designed as a framework for seamless integration of the system into the modeled environment. The program start point is then with the simulator and no longer with the system. In the past few decades, many simulation frameworks have been built to study the behavior of large-scale distributed systems [10]. Cloud, Grid, IoT or Peer-to-Peer systems fall under this category. From a certain degree of scaling, their behavior can be tested more easily in a simulated scenario.

In addition to testing external influences from the environment on the system, the ideal distribution of the components within the environment must also be found. IoT environments pose a particular challenge here with their large number of heterogeneous devices of different performance levels and distribution across cloud, fog and edge. Figure 2.2 illustrates what Ashouri et al. [2] calls a *"software component deployment problem in IoT system design"*. With a model you can try out different hardware and network resources in a controlled and repeatable manner [10]. In such complex environments, modeling and simulation may provide the capability of regression testing. In addition, due to the repeatability, the configuration data of the simulation can be shared with other developers in order to obtain better validation of the results [10].

2.4 Comparative study of simulators

Compare the existing simulators [14]. Take a look at [6] Analyze of Qualities [2]

CloudSim [3]
Agri-IoT [7]
IOTSim [19]
RIoT Bench [12]
City Bench [1]

2.5 Related Work

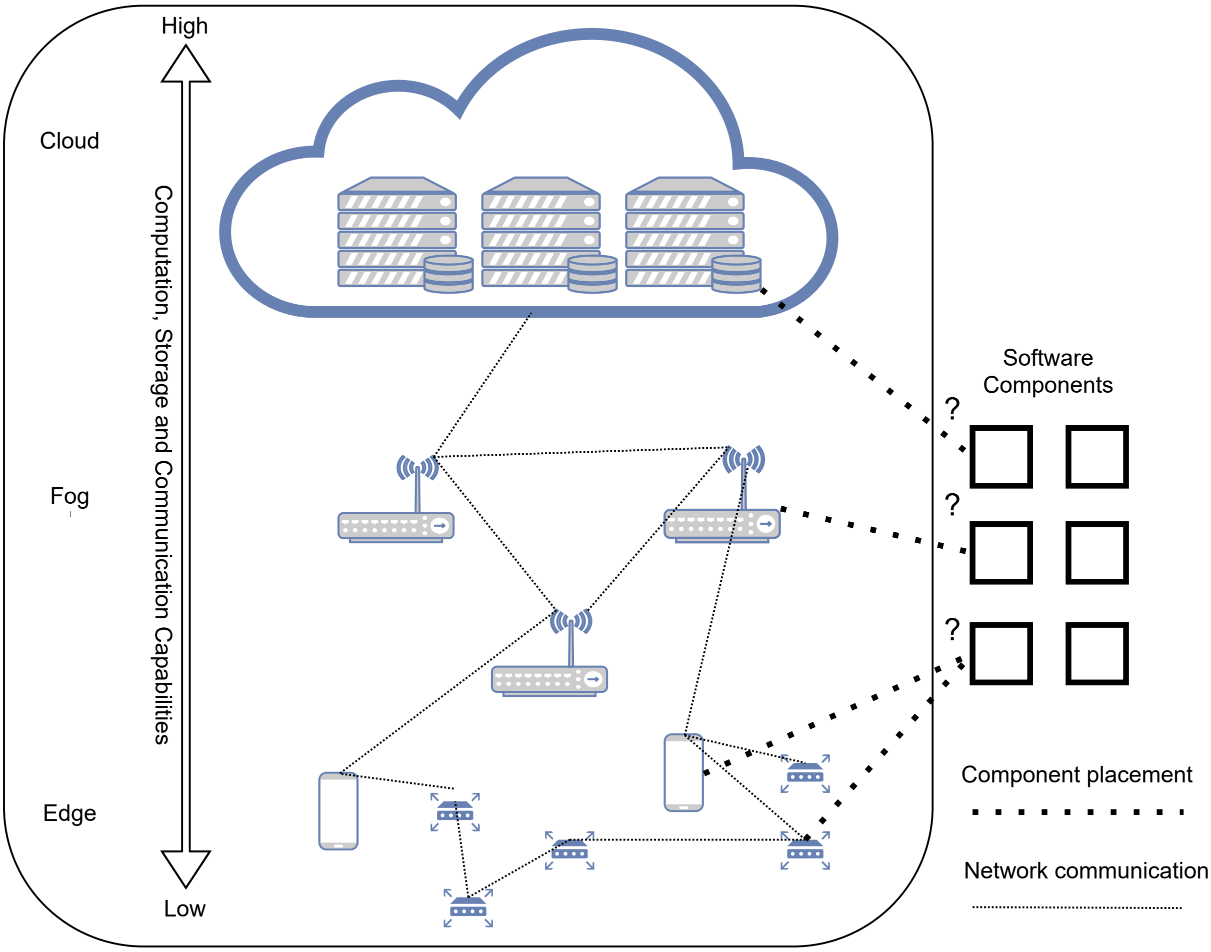

Figure 2.2: Each component could run on different nodes in the IoT environment (based on Ashouri et al. [2]).

3

Conception

3.1 Architecture

3.2 Modelling of IoT Chararacteristics

Devices

How to model restricted devices?
How to model the variety of devices?
How to model parallelism?

Sensors and Actuators

How to model data generation by sensors?
How to model actions by actuators?

Collocation

How to model the placement of the devices?

Communication

How to model network protocols?
How to model bandwith and latency?

Energy

How to model battery?

Mobility

How to model velocity and handover?

Application

How to model application layer?
How to model data aggregation?
How to model application composition?[16]
How to integrate own applications in the Framework?

3.3 Benchmark

4

Realization

Why should i program a completely new simulator and not expand an existing one?
patterns

4.1 Implementation

classes descriptions, sequence diagram

4.2 User Interface

if time left

4.3 Test

Unit Test, Test coverage

5

Evaluation

5.1 Experiment Scenario

5.2 Experiment Results

6

Conclusion

TODO

Bibliography

[1] Ali, M. I., Gao, F., and Mileo, A. CityBench: A Configurable Benchmark to Evaluate RSP Engines Using Smart City Datasets. In: *The Semantic Web - ISWC 2015, Bethlehem, PA, USA*. Springer, Oct. 2015, pp. 374–389. DOI: 10.1007/978-3-319-25010-6_25.

[2] Ashouri, M., Lorig, F., Davidsson, P., and Spalazzese, R. Edge Computing Simulators for IoT System Design: An Analysis of Qualities and Metrics. In: *Future Internet* 11(11), Nov. 2019. DOI: 10.3390/fi11110235.

[3] Calheiros, R. N., Ranjan, R., Rose, C. A. F. D., and Buyya, R. *CloudSim: A Novel Framework for Modeling and Simulation of Cloud Computing Infrastructures and Services*. Tech. rep. Grid Computing and Distributed Systems Laboratory, The University of Melbourne, Mar. 2009. URL: https://arxiv.org/abs/0903.2525 (visited on 11/30/2020).

[4] Groppe, S. and Groppe, J. Hybrid Multi-Model Multi-Platform (HM3P) Databases. In: *9th International Conference on Data Science, Technology and Applications (DATA), Online Streaming*. SciTePress, July 2020, pp. 177–184. DOI: 10.5220/0009802401770184.

[5] Iorga, M., Feldman, L., Barton, R., Martin, M. J., Goren, N., and Mahmoudi, C. *Fog Computing Conceptual Model*. Tech. rep. 500-325. Gaithersburg, MD, USA: National Institute of Standards and Technology, The University of Melbourne, Mar. 2018. URL: https://doi.org/10.6028/NIST.SP.500-325 (visited on 12/21/2020).

[6] Jha, D. N., Alwasel, K., Alshoshan, A., Huang, X., Naha, R. K., Battula, S. K., Garg, S., Puthal, D., James, P., Zomaya, A., et al. *IoTSim-Edge: A Simulation Framework for Modeling the Behavior of IoT and Edge Computing Environments*. Tech. rep. Newcastle University, United Kingdom et al., Oct. 2019. URL: https://arxiv.org/abs/1910.03026 (visited on 11/30/2020).

[7] Kamilaris, A., Gao, F., Prenafeta-Bolduú, F. X., and Ali, M. I. Agri-IoT: A Semantic Framework for Internet of Things-enabled Smart Farming Applications. In: *2016 IEEE 3rd World Forum on Internet of Things (WF-IoT), Reston, VA, USA*. IEEE, Dec. 2016, pp. 442–447. DOI: 10.1109/WF-IoT.2016.7845467.

[8] Kecskemeti, G., Casale, G., Jha, D. N., Lyon, J., and Ranjan, R. Modelling and Simulation Challenges in Internet of Things. In: *IEEE Cloud Computing* 4(1):62–69, 2017. DOI: 10.1109/MCC.2017.18.

[9] Mietz, R., Groppe, S., Römer, K., and Pfisterer, D. Semantic Models for Scalable Search in the Internet of Things. In: *Journal of Sensor and Actuator Networks* 2(2):172–195, 2013. DOI: 10.3390/jsan2020172.

[10] Ranjan, R. Modeling and Simulation in Performance Optimization of Big Data Processing Frameworks. In: *IEEE Cloud Computing* 1(4):14–19, 2014. DOI: 10.1109/MCC.2014.84.

[11] Ray, P. P. An Introduction to Dew Computing: Definition, Concept and Implications. In: *IEEE Access* 6:723–737, 2018. DOI: 10.1109/ACCESS.2017.2775042.

[12] Shukla, A., Chaturvedi, S., and Simmhan, Y. *RIoTBench: A Real-time IoT Benchmark for Distributed Stream Processing Platforms*. Tech. rep. Indian Institute of Science, India, Jan. 2017. URL: https://arxiv.org/abs/1701.08530v1 (visited on 11/30/2020).

[13] Spillner, A., Linz, T., and Schaefer, H. *Software Testing Foundations: A Study Guide for the Certified Tester Exam*. Rocky Nook, Mar. 2014. ISBN: 978-1937538422.

[14] Svorobej, S., Endo, P. T., Bendechache, M., Filelis-Papadopoulos, C., Giannoutakis, K. M., Gravvanis, G. A., Tzovaras, D., Byrne, J., and Lynn, T. Simulating Fog and Edge Computing Scenarios: An Overview and Research Challenges. In: *Future Internet* 11(3), Feb. 2019. DOI: 10.3390/fi11030055.

[15] Taylor, K., Griffith, C., Lefort, L., Gaire, R., Compton, M., Wark, T., Lamb, D., Falzon, G., and Trotter, M. Farming the Web of Things. In: *IEEE Intelligent Systems* 28(6):12–19, 2013. DOI: 10.1109/MIS.2013.102.

[16] Villari, M., Fazio, M., Dustdar, S., Rana, O., Jha, D. N., and Ranjan, R. Osmosis: The Osmotic Computing Platform for Microelements in the Cloud, Edge, and Internet of Things. In: *Computer* 52(8):14–26, 2019. DOI: 10.1109/MC.2018.2888767.

[17] Weisberg, M. *Simulation and Similarity: Using Models to Understand the World*. Oxford University Press, Dec. 2012. ISBN: 978-0190265120.

[18] Yannuzzi, M., Milito, R., Serral-Gracià, R., Montero, D., and Nemirovsky, M. Key ingredients in an IoT recipe: Fog Computing, Cloud computing, and more Fog Computing. In: *19th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), Athens, Greece*. IEEE, Dec. 2014, pp. 325–329. DOI: 10.1109/CAMAD.2014.7033259.

[19] Zeng, X., Garg, S. K., Strazdins, P., Jayaraman, P. P., Georgakopoulos, D., and Ranjan, R. IOTSim: A simulator for analysing IoT applications. In: *Journal of Systems Architecture* 72:93–107, 2017. DOI: 10.1016/j.sysarc.2016.06.008.