Mobile Simulation Environment for Investigating IoT-Cloud Systems

Summary of the Ph.D. Dissertation

Tamás Pflanzner

Supervisor:

Dr. Attila Kertész

Doctoral School of Computer Science
University of Szeged

Szeged
2020

Introduction

Cloud computing [11] has become a widespread and reliable solution over the past decade by providing scalable, virtualized data storage and computation. By overcoming the interoperability issues of public cloud providers and various middleware implementations, the process of creating and managing cloud federations was clarified and applied [10]. The Internet of Things (IoT) is one of the latest trends of the current ICT evolution, represented by a huge amount of powerful smart devices that have started to appear on the Internet [17]. According to reports in this field, Gartner [16] predicted that there will be 30 billion devices online, and more than 200 billion devices discontinuously online by 2020, while Huawei foresees 100 billion new IoT connections till 2025 [6]. By responding to this new trend, many cloud providers have started to offer services for IoT management. Recent advances have already shown that Cloud Computing can be used to serve IoT needs by performing data generation, processing and visualization tasks. Gubbi et al. [13] were one of the pioneers by describing IoT systems in 2013, and they also emphasized the importance of the joint utilization of IoT and clouds. Botta et al. presented the main properties, features, and open issues of such combination in [8], while Nastic et al. [15] listed the numerous challenges of realizing IoT-Cloud systems in practice. These related works also served as motivations for my research by raising the need for managing a large number of protocols and data formats by means of simulation.

To create such ecosystems, IoT system developers need to purchase, connect and configure these devices, and they also have to choose the right infrastructure provider offering the combination of protocols and data structures fitting their applications. To ease these tasks, I propose a semi-simulation environment in this dissertation that can be used to investigate the inner workings of complex, IoT-Cloud systems. This environment consists of three components that represent the main contributions of my work: (i) an Android-based, mobile IoT device simulator called MobIoTSim that can mimic real world sensors and devices; (ii) a customizable cloud gateway to manage IoT devices by receiving, processing and visualizing sensor data; and (iii) an IoT trace archiving service called SUMMON that can be used to gather real world sensor data.

One of the main motivating forces behind my research was that several cloud providers had started to offer IoT-specific services to ease the development of IoT-Cloud applications, but such cases, where many different things needed to be operated and managed at the same time, were hard to realize. Just to mention a few, smart city application scenarios using SIGFOX [4] or LoRa [2] technologies are very expensive and time-consuming to be deployed with real devices, since a base station costs up to a thousand euros needing enormous configuration work. In my thesis I respond to these challenges by designing and developing solutions for simulating IoT devices and managing the simulated applications with real world cloud services.

Concerning state-of-the-art solutions, some of the the existing simulators for examining distributed systems can be categorized as general network simulators, and there are also specialized, IoT simulators addressing issues raised by this work. Chernyshev et al. [12] presented a recent survey of IoT simulators and testbeds. They argued that research in this area is especially challenging, and existing simulators are specialized in the sense that they focus on a particular architectural layer. After examining the related simulators in the IoT field, I also created a survey for comparing the related works, and highlighting their differences. Its results are shown in Table 1. I found wanting to have a solution that puts an emphasis on mobility, and provides means for performing simulations at specific geolocations. This fact served as a major motivation for designing a simulator for mobile devices, which I named MobIoTSim. The main purpose of such a mobile IoT device simulator is to help cloud application developers to learn IoT device handling without buying real sensors and to test and demonstrate complex IoT applications.

Table 1: Comparison of related simulation solutions

Simulator	**Cloud nodes /services**	**IoT sensors /devices**	**Artificially gen. sensor data**	**Real sensor /device data**	**Learning curve**	**GUI support**
NetSim	Simulated	no	yes	no	long	yes
Qualnet	Simulated	no	yes	no	long	yes
OMNet++	Simulated	no	yes	no	long	limited
SimIoT	Simulated	yes	yes	no	long	limited
IoTSim	Simulated	yes	yes	no	long	no
iFogSim	Simulated	yes	yes	no	long	yes
DISSECT-CF	Simulated	yes	yes	yes	long	no
SimpleIoTSim.	Real	yes	yes	limited	short	yes
Atomiton	No inf.	yes	yes	No inf.	short	yes
MobIoTSim	Real	yes	yes	yes	short	yes

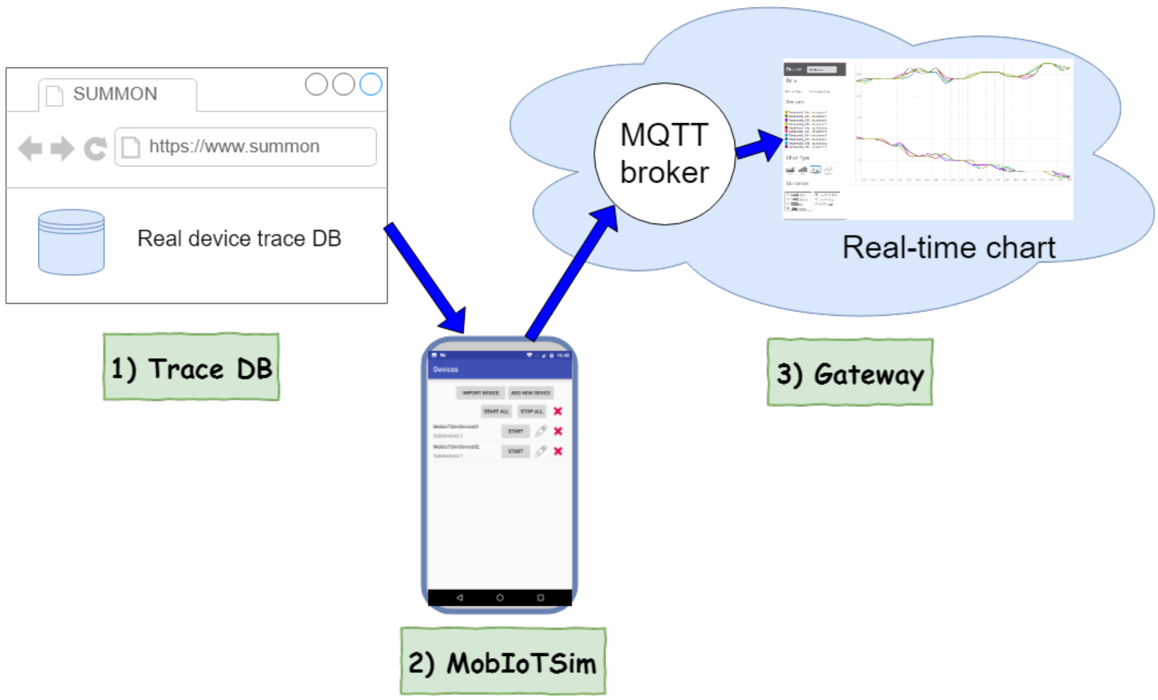

Figure 1: The proposed framework for simulating IoT-Cloud systems

New scientific results

The architectural view of my proposal for simulating IoT-Cloud systems is depicted in Figure 1. Compared to traditional simulators, I propose a semi-simulated environment. To be as close as possible to real world systems, I started to gather real sensor trace files of public IoT initiatives, and made them available through an archive (I.), which can be connected to MobIoTSim (II.) to simulate real device behavior. I also developed gateway services (III.) to manage IoT devices by processing and visualizing their sensor data. These gateways can be instantiated and operated at public or private cloud or fog providers.

Thesis I. I designed and developed a mobile IoT simulator called MobIoTSim, which can simulate multiple IoT devices in a user-friendly way, and send the generated data on the real network to the cloud. I validated its usability, and evaluated its scalability for managing multiple devices simultaneously.

My proposed mobile IoT device simulator is designed to simulate up to hundreds of IoT devices, and it is implemented as a mobile application for Android platforms. Sensor data generation of the simulated devices can be done by artificially generated, random values in a range given by the user – this is the default option. For more advanced simulations, data from IoT-specific trace files can also be loaded and replayed. The data sending frequency can be specified for each device for both cases. The application uses MQTT protocol (by default) to send messages with the use of the Eclipse Paho library, which is open source. The MQTT protocol is an established messaging protocol for IoT systems [14] and widely supported by IoT-Cloud services. Message data is represented in a structured JSON object in the simulator, compatible with the IBM IoT Foundation message format.

The basic usage of the simulator is to connect the application to a cloud, where the data is to be sent, to create and configure the devices to be simulated, and to start the (data generation of the) required devices. These main steps are represented by three main parts of the application: the *Cloud settings*, the *Devices* and the *Device settings* screens.

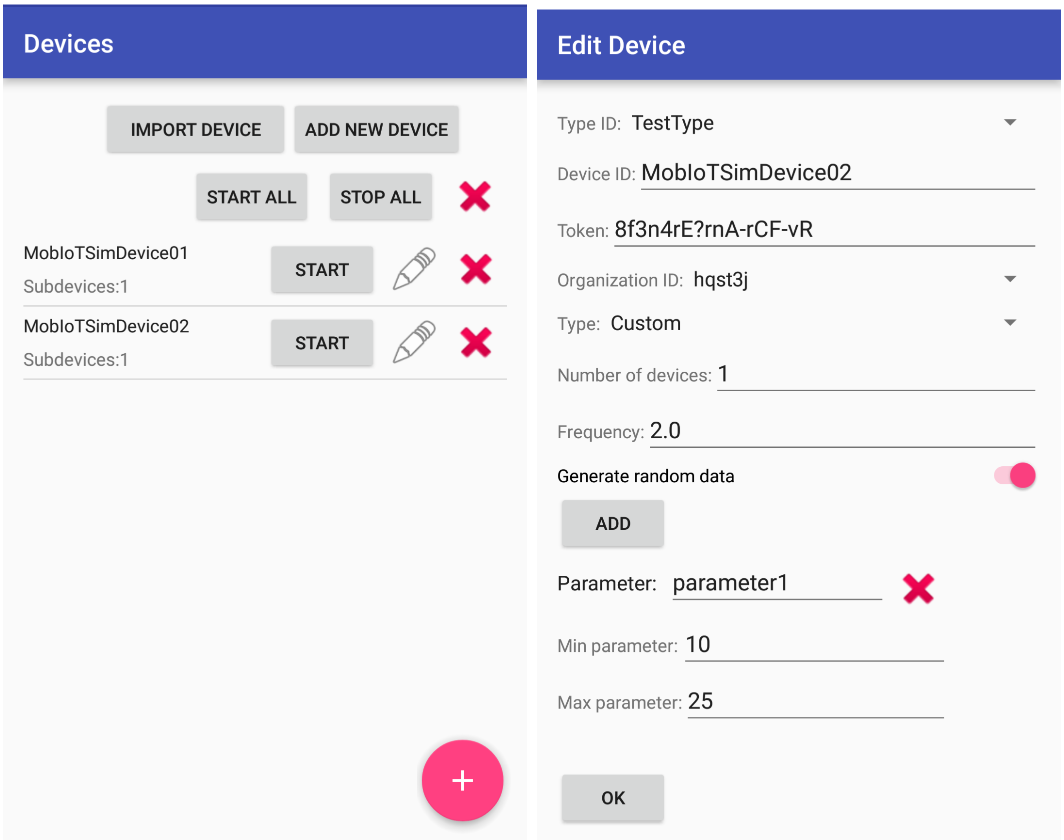

Figure 2: The user interface of the MobIoTSim Android application

In the *Devices* screen (left part of Figure 2), we can create or import devices, and also view the list of existing devices, where every row is a device or a device group. These entities can be started and stopped by the user at will, both together or separately for the selected ones. New device creation and the editing of an existing one can be done in the *Device settings* (or *Edit device*) screen (right part of Figure 2). If the user selects an existing template for the base of the device in the *Type* field, the message content and frequency will be set to some predefined values. For example, the *Thermostat* template has a temperature parameter, and the device turns on by reaching a predefined low-level value and turns off at reaching a high-level value.

Sensor data generation for the simulated devices can be done in different ways. The default way to generate device data is to create a custom device, where the user can define parameters with ranges, within the random value will be generated (by default the *Generate random data* option is activated). To simplify this process the user can use device templates, which has predefined parameters and ranges, specific to the device type, and also an artificial, random value generation will be performed. If the *Generate random data* option is deactivated, the user can select a file to load the sensor values. In this way, we can replay previous experiments. Beside using random generation and replays, we can even load trace files of real world applications gathered from public sources. An example file from the OpenWeatherMap public weather data provider [3] can be used, which monitors many cities and stores many parameters of them, including temperature, humidity, air pressure and wind speed. Using the trace loader feature of MobIoTSim, the simulation can mimic a real life scenario.

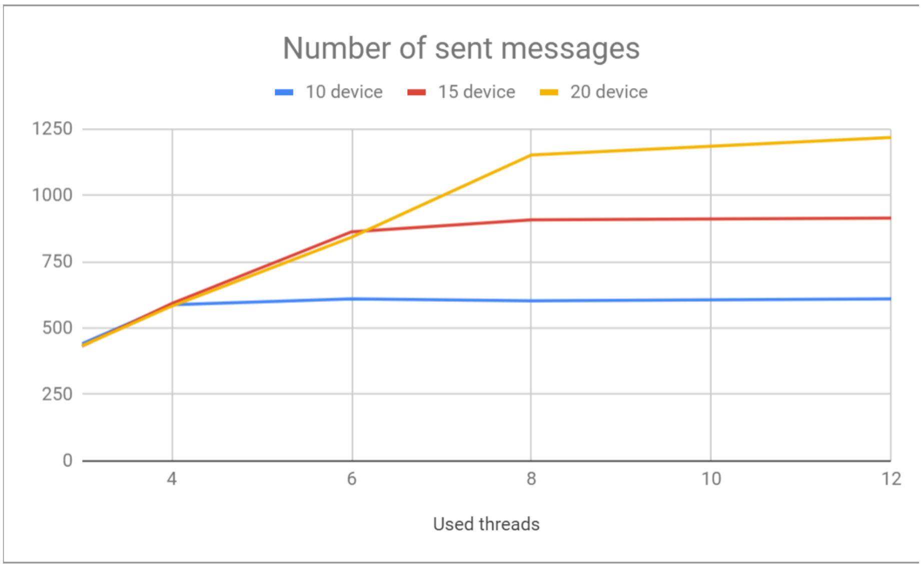

Figure 3: The tradeoff of sent messages and used cores

I analyzed the device simulation scalability performance of the MobIoTSim application. For this purpose I created a simplified version of MobIoTSim (called MQTTDemo), containing only its core parts for device management functionalities to enable automated, script-based message generation. It allows for accessing low-level configurations, like specifying the number of threads used for managing the simulated devices. It also collects detailed statistical information of the simulation by measuring elapsed times for executing certain functions. It can be connected to the private gateway deployed in the IBM Cloud, and can send messages to it using the MQTT protocol. By the time I performed the evaluations, the number of simulated devices were limited to 20 by the IBM Cloud platform. In the Settings we can define the number of devices to be simulated, the message frequency for sensor data generation, the number of used threads

(*ThreadPool*) and the message type and contents. This type can be a simple JSON object with a random parameter, or an OpenWeatherMap device data in JSON format describing sensor values of certain smart cities. Other settings are hard-coded to the application for this testing, as well as the address of the MQTT broker of the IBM Cloud service.

The evaluation results shown in Figure 3 proved that the random data generalization consumed an almost negligible time, so it did not interfere with the simulation. To manage 20 devices (the highest number for IBM Cloud, i.e. Bluemix till 2017), we can see that six threads were too few for an acceptable performance, therefore we had to use eight threads instead.

Thesis II. I designed and developed a generic cloud gateway service, which can be deployed on any cloud service, and can manage IoT devices by receiving, storing and processing or visualizing data. I also validated its usability together with MobIoTSim, and evaluated its scalability.

After examining the IoT services of public cloud providers, I performed a requirement analysis for a service that is easy to use and able to handle both simulated and real world IoT devices. Based on these results, I developed a customizable, private gateway that can be deployed to any cloud provider to handle up to hundreds of different devices at the same time.

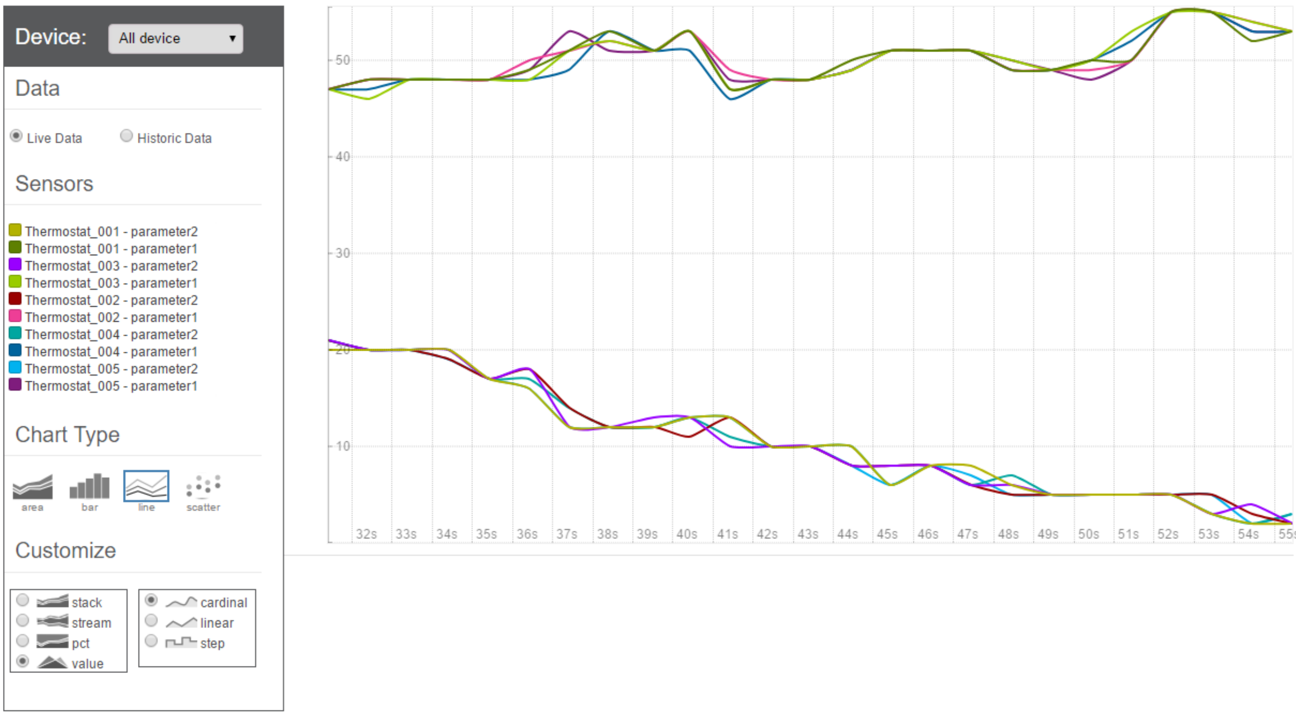

Figure 4: Detailed, parameter-wise data visualization in the private gateway in the IBM Cloud sent by a group of devices in MobIoTSim

This generalized gateway is a node.js application that is able to manage multiple device topics with the MQTT protocol, and it has a web-based graphical interface to visualize sensor data coming from MobIoTSim – as shown in Figure 4. Messages (defined in JSON format) received from the simulated devices are managed by an MQTT server. It can also be used to send responses (or notifications) back to the simulated IoT devices in MobIoTSim. To support multiple device handling, it uses paging to overcome device management limitations introduced in the free version of the IBM Cloud. In order to enhance and better visualize heterogeneous device data simultaneously, I introduced device grouping for the chart generation. This way we can see the data sent by all devices in a single real-time chart. Though the IBM Cloud provides some monitoring information for deployed applications, we can have access to more detailed monitoring data using custom Docker containers. As a result, I placed my extended gateway solution to a Docker container, which has also became more portable at the same time.

Table 2: Measurements with the Weather template.

No. of devices	**10**	**100**	**250**	**450**
CPU util. (%)	1.59	12.27	29.53	52.29
Memory (MB)	110.07	110.22	110.35	111.05
Network (B/s)	853.6	855.16	881.34	890.66
Msg. size (KB)	2468	24695	61666	111110
No. of msg.	6000	60046	149940	270165

I evaluated the scalability of this generic gateway service. I deployed the gateway in a Docker container as a micro application to the IBM Cloud. I performed multiple rounds of evaluation. In the initial phase evaluation, I used the Thermostat template of MobIoTSim to create 900 simulated devices in groups. These results indicated that even with this high number of devices only a little load could be observed. For the second round of experiments, I introduced CPU stressing, and created device groups for 10, 100, 250 and 450 devices in MobIoTSim, and re-executed the simulation. We can see that for doubled messages (i.e. 0.5 second frequencies) the CPU utilization is also doubled. In the last evaluation round I created 10, 100, 250 and 450 devices in four groups, using the Weather template of MobIoTSim. Since MobIoTSim is capable of loading previously saved IoT trace data, I used this feature for loading real data provided by OpenWeatherMap. I performed the experiments for all device groups with one second frequency. The detailed results are shown in Table 2. The results showed that the Weather template uses bigger messages, which also results in higher CPU load.

Smart farming is also a rapidly growing area within smart systems, that need to respond to great challenges of the near future. Due to recent ICT developments we can apply novel sensor and IoT technologies to provide a promising alternative, called affordable phenotyping. To address these aims, in the frame of the Internet of Living Things project [5], I participated in developing a low cost plant phenotyping platform for small sized plants, which enables the remote monitoring of plant growth in a standard greenhouse environment, called the IoLT Smart Pot. To manage this smart pot cluster, I designed and participated in the implementation of the IoLT Smart Pot Gateway for monitoring its environmental parameters, based on the results of the generic gateway solution. I also participated in the evaluation of the proposed solution with scalable simulations, and exemplified its real world utilization. Figure 5 shows a screenshot of the specialized gateway web interface that generated a chart of the calculated values based on the received real sensor values (of a selected time interval) to represent a time course of the projected leaf area of the selected plant in a pot of the cluster. The curve nicely reveals a cirkadian oscillation pattern due to periodic leaf movement (flattening in the dark and erection in the light period).

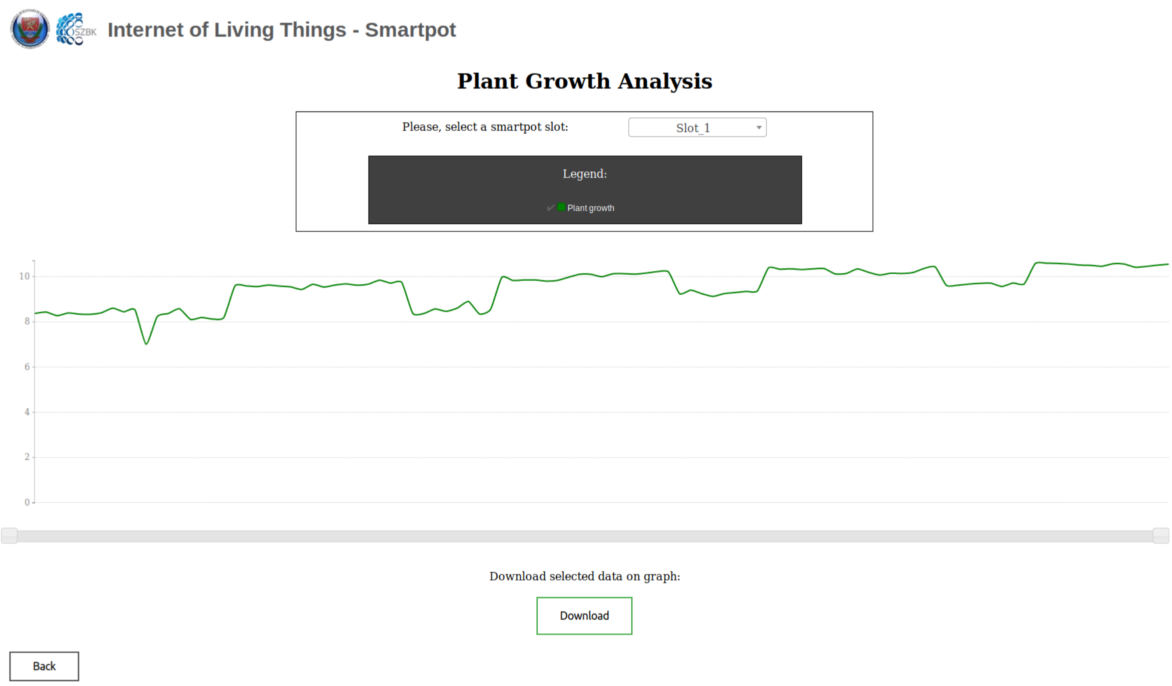

Figure 5: Visualizing leaf area changes over time in the IoLT Smart Pot Gateway

Thesis Point III. I designed and developed an IoT trace archiving service called SUMMON, which can periodically collect and store data from real IoT applications, even with using web crawling techniques. I demonstrated its operation with three use cases. The gathered datasets are shared in SUMMON to facilitate more realistic IoT simulations.

The European Commission has published a guideline document [1] for encouraging open access to reuse digital research data generated by Horizon 2020 projects. In this report they defined the so-called FAIR data principles to help Horizon 2020 beneficiaries to make their research data findable, accessible, interoperable and reusable. Following these guidelines, I aimed to create an open IoT trace archive to facilitate data search and accessibility, and to reuse sensor data for experimenting with IoT-Cloud systems.

In the first versions of MobIoTSim one could only use artificially generated values for sensor simulation. Later, I developed a trace loading feature to support easier configuration of experiments, and to replay real world sensor data in certain simulated scenarios. I introduced a Weather template for representing weather monitoring features of smart city devices, for which I used data from the OpenWeatherMap service. To further extend the supported application areas, I envisioned a public IoT trace archive to be used together with MobIoTSim. For this aim, I developed a solution called Smart system Usage data Management and MONitoring (in short SUMMON), which is a web service for gathering and filtering data from these projects. With this service we can list the available datasets, and query and filter a selected dataset to be loaded by MobIoTSim for large scale experiments. I gathered and filtered real sensor data of smart cities, such as the SmartME [9] and the CityPulse [7] projects, and archived it in SUMMON.

Besides filtering and archiving datasets of smart city projects, it would be beneficial to reuse

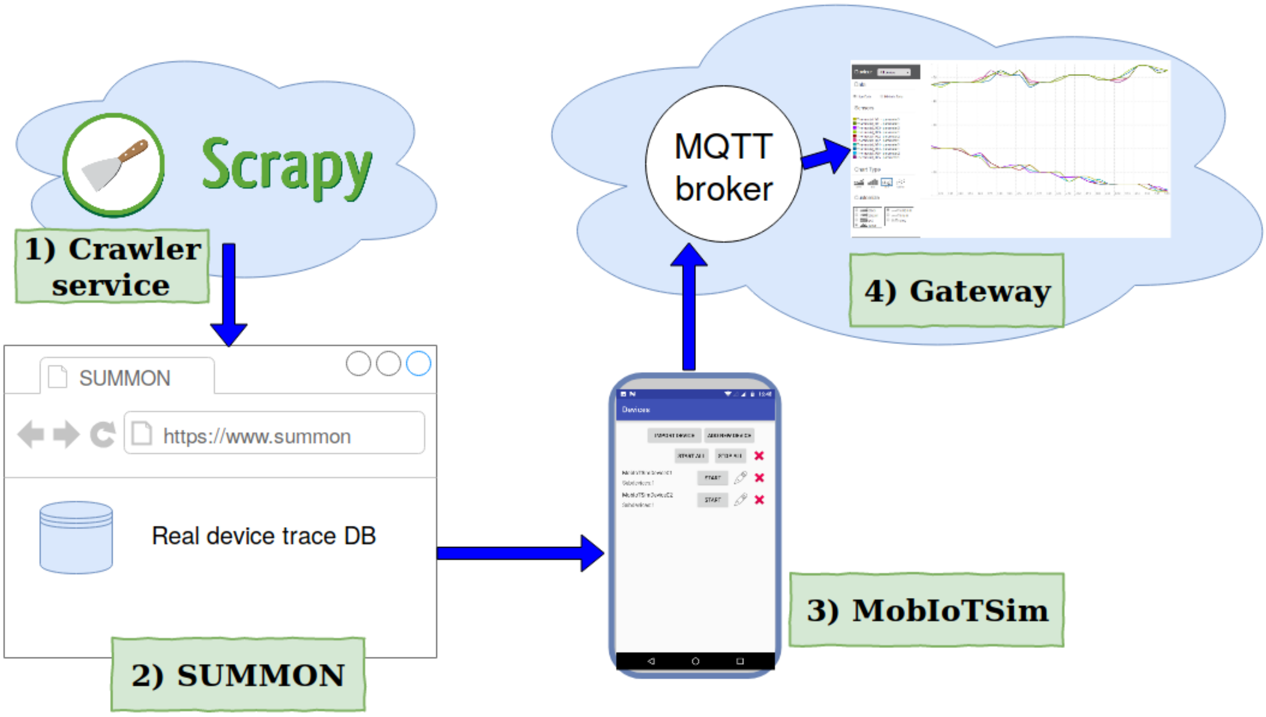

Figure 6: The extended environment for simulating IoT-Cloud systems

data from social networking sites and from such dynamically refreshed websites that provide visualized information of crowd-sourced smart systems that do not share their sensed data. To gather data from such services, I proposed a web crawling approach to facilitate open IoT data archiving and reuse. This proposal extends the previous solutions, and the designed crawling service can provide additional datasets to perform various experiments with the semi-simulation framework, as shown in Figure 6.

Table 3: Comparison of the three use cases

Project	Time (sec)	Data (KB)	No. of messages
idokep	4	20.24	50
livetraffic	30	17.7	47
parking	10	7.48	33

I participated in the development of the crawling service, for which we used Scrapy, which is an open-source python-based framework for data mining, and Scrapyd, which is an application for managing multiple processes of the spiders' requests in a parallel way. We realized the service as a container-based Docker application. A prototype instance is deployed and available in the Google Cloud, which is connected to a running SUMMON prototype through a REST API to evaluate their interaction. To get the values of the crawled fields, we had to perform various searching and filtering methods on the HTML source of the sites. We also pseudonymised the nicknames for privacy issues before saving the post messages to the MongoDB database of the crawling sercive. In this way the data gathered by our crawler service can be regularly retrieved and stored in SUMMON. I also participated in the evaluation of its utilization with three smart city use cases: a crowdsourced meteorological service called Idokep.hu, a live traffic news service of the Automobile Association Developments Ltd., and a a cloud-based parking management solution of the ParkMe project. Table 3 shows the average values for the execution times, saved data sizes and number of messages for the spider jobs run on a given day. From the results we could see that the web page layouts of the corresponding smart systems varied highly, therefore no generic spider could be used to crawl them.

Conclusions

In the phase of the latest Internet evolution, a high number of powerful devices appeared on the network, which led to the birth of the IoT paradigm. As a result, cloud technology providers responded to this trend, and broadened their services to attract users of IoT applications. In this new world, IoT developers are facing many challenges: they need to purchase smart devices, to design and build-up a network of these components, to develop IoT applications, to test and evaluate these applications on the created system and finally, to fine-tune the applications based on the evaluation results.

In this work I provided means to overcome some of these challenges. I developed a complex, semi-simulation environment composed the following tools: an Android-based, mobile device simulator called MobIoTSim capable of simulating up to hundreds of IoT devices, a customizable cloud gateway service that is able to manage the simulated devices, and an IoT trace archive service called SUMMON, which can be used to gather real-world sensor data to be fed to the simulation experiments. I exemplified the usage of this complex simulation environment, and evaluated its device management scalability and responsiveness. I do believe that these tools can contribute to a more cost efficient IoT system planning, development and evaluation, even for different academic research communities and industrial stakeholders.

The scientific results presented in this dissertation have been published in numerous journals, conference and workshop papers and have been presented in various scientific forums. These publications have inspired further research, generated collaborations, and are well represented by many independent citations – which are summarized in Table 4.

Most of the research I performed in the past five years has resulted from active involvement in the Internet of Living Things (GINOP-2.3.2-15-2016-00037), Smart Systems Research Institute (20391-3/2018/FEKUSTRAT and TUDFO/47138-1/2019-ITM), EFOP-3.6.1-16-2016-00008, and OTKA FK 131793 national, and EU COST Action IC1304 (ACROSS) and EU COST Action CA17136 (INDAIRPOLLNET) international projects.

The components of the proposed semi-simulation environment are under continuous development and improvement, their source codes are open and available on GitHub at the following links:

- https://github.com/sed-szeged/MobIoTSim/
- https://github.com/sed-szeged/MobIoTSimBluemixGateway
- https://github.com/sed-szeged/iotgateway
- https://github.com/sed-inf-u-szeged/IoLT-Smart-Pot-Gateway
- https://github.com/sed-szeged/SUMMON

Table 4: Publications, Theses and Independent citations

	Thesis I.	Thesis II.	Thesis III.	Independent citations Google Scholar	MTMT
[P14]		♦		9	4
[P12]		♦			
[P13]		♦		1	
[P6]	♦	♦		35	21
[P1]			♦	1	
[P10]	♦	♦		12	6
[P9]	♦	♦	♦	1	1
[P4]	♦	♦		2	
[P7]	♦		♦	5	
[P11]	♦	♦	♦	1	1
[P8]	♦		♦	1	1
[P2]	♦	♦	♦	12	8
[P3]			♦		
[P5]		♦			
Sum	8	10	7	80	42

Acknowledgements

Over the past five years I had the opportunity to be part of a research team that started to build up a competence in combining Cloud and IoT technologies and providing simulation tools for investigating such ecosystems. This means that I achieved the results reported in this thesis not as an individual researcher, but as a team member. I am grateful to my colleagues for providing these collaboration opportunities.

I would like to thank my supervisor, Dr. Attila Kertész, for the guidance, the helpful pieces of advice, the time and the energy he spent on me. I am grateful that he started me on my research career and that we always had great experiences on shared professional trips.

I would also like to thank Prof. Dr. Tibor Gyimóthy, the former head of the Software Engineering Department, and Dr. Rudolf Ferenc, the current head of the Software Engineering Department, for helping me advance in my PhD studies. Many thanks to my coauthors for giving me a lot of great ideas and helping me overcome the technical challenges, and to Edit Szűcs for proofreading the English version of my dissertation.

I am grateful to my colleagues, especially András Márkus, Dr. Zoltán Tóth, Dr. Dénes Bán and Dr. István Kádár for I could count on them.

Last but not least, I would like to thank my parents for their care and encouragement. I feel very fortunate that my family members are setting a good example for me: my mother with reliability, my father with tirelessness and wide range of interests, my grandmothers with aptitude and perseverance, my grandfathers with dedication and cheerfulness, my uncle with creative thinking, my godmother with kindness, my godfather with entrepreneurial spirit. Thank you Kata, from the bottom of my heart, for being by my side in the moments of struggle, for being my partner in the wonderful experiences, but most of all, for being with me in simple everyday life. I am grateful to Dóri and Dani for being like my real siblings. Thank you to my friends for the cheerful moments, love and understanding.

Research reported in this thesis was supported by the Hungarian Government and the European Regional Development Fund under the grant number GINOP-2.3.2-15-2016-00037 (Internet of Living Things).

Tamás Pflanzner, 2020

References

[1] European Commission, Guidelines on FAIR Data Management in Horizon 2020. Version 3.0, July 2016. Online: http://ec.europa.eu/research/participants/data/ref/h2020/grants_manual/hi/oa_pilot/-h2020-hi-oa-data-mgt_en.pdf.

[2] LoRa Technology. `https://www.lora-alliance.org/What-Is-LoRa/Technology`, 2017. Accessed: 2017-03-20.

[3] OpenWeatherMap. `http://www.openweathermap.org`, 2017. Accessed: 2017-03-19.

[4] SIGFOX. `http://www.sigfox.com/en/#!/connected-world`, 2017. Accessed: 2017-03-21.

[5] GINOP Internet of Living Things (IoLT) Project. `"https://www.sed.inf.u-szeged.hu/iolt"`, 2020. Accessed: 2020-10-03.

[6] Huawei GIV 2025. `"https://www.huawei.com/minisite/giv/Files/whitepaper_en_2018.pdf"`, 2020. Accessed: 2020-10-03.

[7] Stefan Bischof, Athanasios Karapantelakis, Cosmin-Septimiu Nechifor, Amit P Sheth, Alessandra Mileo, and Payam Barnaghi. Semantic modelling of smart city data. 2014.

[8] Alessio Botta, Walter De Donato, Valerio Persico, and Antonio Pescapé. Integration of cloud computing and internet of things: a survey. *Future generation computer systems*, 56:684–700, 2016.

[9] Dario Bruneo, Salvatore Distefano, Francesco Longo, Giovanni Merlino, and Antonio Puliafito. Turning messina into a smart city: The# smartme experience. In *Smart Cities in the Mediterranean*, pages 135–152. Springer, 2017.

[10] Wojciech Burakowski, Andrzej Beben, Hans van den Berg, Joost W Bosman, Gerhard Hasslinger, Attila Kertesz, Steven Latre, Rob van der Mei, Tamas Pflanzner, Patrick Gwydion Poullie, et al. Traffic management for cloud federation. In *Autonomous Control for a Reliable Internet of Services*, pages 269–312. Springer, Cham, 2018.

[11] Rajkumar Buyya, Chee Shin Yeo, Srikumar Venugopal, James Broberg, and Ivona Brandic. Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation computer systems*, 25(6):599–616, 2009.

[12] Maxim Chernyshev, Zubair Baig, Oladayo Bello, and Sherali Zeadally. Internet of things (iot): Research, simulators, and testbeds. *IEEE Internet of Things Journal*, 5(3):1637–1647, 2017.

[13] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. Internet of things (iot): A vision, architectural elements, and future directions. *Future generation computer systems*, 29(7):1645–1660, 2013.

[14] N. Naik. Choice of effective messaging protocols for iot systems: Mqtt, coap, amqp and http. In *2017 IEEE International Systems Engineering Symposium (ISSE)*, pages 1–7, 2017.

[15] Stefan Nastic, Sanjin Sehic, Duc-Hung Le, Hong-Linh Truong, and Schahram Dustdar. Provisioning software-defined iot cloud systems. In *2014 international conference on future internet of things and cloud*, pages 288–295. IEEE, 2014.

[16] B Rossi. Gartner's internet of things predictions. *Information Age, Vitesse Media, January*, 2015.

[17] Harald Sundmaeker, Patrick Guillemin, Peter Friess, and Sylvie Woelfflé. Vision and challenges for realising the internet of things. *Cluster of European Research Projects on the Internet of Things, European Commision*, 3(3):34–36, 2010.

Publications

[P1] Attila Kertesz and Tamas Pflanzner. Towards enabling scientific workflows for the future internet of things. In *Internet of Things. IoT Infrastructures. Second International Summit, IoT 360° 2015, Rome, Italy, October 27-29, 2015. Revised Selected Papers, Part I*, 1 2017.

[P2] Attila Kertesz, Tamas Pflanzner, and Tibor Gyimothy. A mobile iot device simulator for iot-fog-cloud systems. *Journal of Grid Computing*, 17(3):529–551, 2019.

[P3] Tamas Pflanzner, Zoltan Feher, and Attila Kertesz. A crawling approach to facilitate open iot data archiving and reuse. In *2019 Sixth International Conference on Internet of Things: Systems, Management and Security (IOTSMS)*, pages 235–242. IEEE, 2019.

[P4] Tamas Pflanzner, Marta Fidrich, and Attila Kertesz. Simulating sensor devices for experimenting with iot cloud systems. In *Connected Environments for the Internet of Things*, pages 105–126. Springer, 2017.

[P5] Tamas Pflanzner, Miklos Hovari, Imre Vass, and Attila Kertesz. Designing an iot-cloud gateway for the internet of living things. In *International Conference on Cloud Computing and Services Science*, pages 23–41. Springer, 2019.

[P6] Tamas Pflanzner and Attila Kertesz. A survey of iot cloud providers. In *2016 39th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 730–735. IEEE, 2016.

[P7] Tamas Pflanzner and Attila Kertész. A taxonomy and survey of iot cloud applications. *EAI Endorsed Transactions on Internet of Things*, 3(12), 10 2017.

[P8] Tamas Pflanzner and Attila Kertesz. Analyzing iot, fog and cloud environments using real sensor data. In *Fog Computing*, pages 83–105. Springer, Cham, 2018.

[P9] Tamas Pflanzner and Attila Kertész. A private gateway for investigating iot data management. In Víctor Méndez Muñoz, Donald Ferguson, Markus Helfert, and Claus Pahl, editors, *Proceedings of the 8th International Conference on Cloud Computing and Services Science, CLOSER 2018, Funchal, Madeira, Portugal, March 19-21, 2018*, pages 526–532. SciTePress, 2018.

[P10] Tamas Pflanzner, Attila Kertesz, Bart Spinnewyn, and Steven Latre. Mobiotsim: towards a mobile iot device simulator. In *2016 IEEE 4th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW)*, pages 21–27. IEEE, 2016.

[P11] Tamas Pflanzner, K Zs Leszko, and Attila Kertesz. Summon: Gathering smart city data to support iot-fog-cloud simulations. In *2018 Third International Conference on Fog and Mobile Edge Computing (FMEC)*, pages 71–78. IEEE, 2018.

[P12] Tamas Pflanzner, Roland Tornyai, Akos Zoltan Goracz, and Attila Kertesz. Characterizing paas solutions enabling cloud federations. In *Application Development and Design: Concepts, Methodologies, Tools, and Applications*, pages 1095–1120. IGI Global, 2018.

[P13] Tamas Pflanzner, Roland Tornyai, and Attila Kertesz. Towards enabling clouds for iot: Interoperable data management approaches by multi-clouds. In *Connectivity Frameworks for Smart Devices*, pages 187–207. Springer, 2016.

[P14] Roland Tornyai, Tamas Pflanzner, Attila Kertesz, Anita Schmidt, and Balazs Gibizer. Performance analysis of an openstack private cloud. In *Proceedings of the 6th International Conference on Cloud Computing and Services Science - Volume 1 and 2*, CLOSER 2016, page 282–289, Setubal, PRT, 2016. SCITEPRESS - Science and Technology Publications, Lda.