Microservice-based IoT for Smart Buildings

Dilshat Salikhov, Kevin Khanda, Kamill Gusmanov
Manuel Mazzara, Nikolaos Mavridis
Innopolis University, Russia
{d.salikhov, k.khanda, k.gusmanov, m.mazzara, n.mavridis}@innopolis.ru

***Abstract*—A large percentage of buildings in domestic or special-purpose is expected to become increasingly "smarter" in the future, due to the immense benefits in terms of energy saving, safety, flexibility, and comfort, that relevant new technologies offer. As concerns hardware, software, or platform level, however, no clearly dominant standards currently exist. Such standards, would ideally, fulfill a number of important desiderata, which are to be touched upon in this paper. Here, we will present a prototype platform for supporting multiple concurrent applications for smart buildings, which is utilizing an advanced sensor network as well as a distributed microservices architecture, centrally featuring the Jolie programming language. The architecture and benefits of our system are discussed, as well as a prototype containing a number of nodes and a user interface, deployed in a real-world academic building environment. Our results illustrate the promising nature of our approach, as well as open avenues for future work towards its wider and larger scale applicability.**

I. Introduction

The Internet of Things (IoT), as per the ITU Recommendation ITU-T Y.2060 [2], has been defined as: *"a global infrastructure for the information society, enabling advanced services by interconnecting (physical and virtual) things based on existing and evolving interoperable information and communication technologies."* Among physical objects, buildings are playing a major role in this technological transition. Building function either as human habitats, for example domestic buildings, or for some specialized goals, for example storehouses, shops, industrial buildings, or schools. In some cases they can have both functions.

There are multiple aspects about the operations of modern-day buildings that will lead to future automation and optimization. It has been shown that the benefits of an improved energy management through automation are indeed significant [32]; also, security of buildings, human-friendliness and adaptation to preferences has a vast spectrum of improvement. *Smart buildings*, as well as *smart cities* that are supposed to contain them, are the target of much research today, and promise to dramatically improve our lives increasing sustainability and improving the environment.

Traditionally, most building automation systems were made for specific applications and offered a few properties of openness and flexibility. However, with the fast maturation of a number of supporting technologies, the opportunity to change this *status quo* is rapidly growing. First, cheap sensing and perception technologies have become available for a wide range of examples: covering not only physical properties of the building and its spaces, such as temperature, light, and humidity, but also providing information about the presence, number, identities, activities, and even emotional states of the people inside a building or in its surrounding spaces [17]. Second, affordable and miniature microprocessor-based platforms have become widespread and are easily inter-connectable to sensors, which often have enough processing power to support perception and machine vision; with multiple network transports, even the necessary small battery power is readily available. Third, networking technology for such platforms has developed significantly, and nowadays it is easy to implement building-wide networks, often with dynamic ad hoc topologies, which also support on-the-fly introduction and replacement of new nodes, while providing secure communications. Fourth, special languages and middleware has been developed, in order to support straightforward development of distributed systems based around a number of paradigms, including microservices [9], [16]

Given all these development, one can not only envision, but can also start implementing and experimenting with the usage of the IoT for providing generic smart building infrastructure, which goes beyond the constraints of existing specialized systems, and provides fluid support for diverse concurrent applications, sharing the infrastructure, and operating through microservices provided by the nodes. Furthermore, infrastructure such as the *"Jolie Good Buildings"* that we are going to present here, promises strong scalability, reliability, and ease of evolution as more powerful hardware becomes available, and finally direct utilization of the immense power of external services available on the Internet, as part of the distributed applications running on the nodes.

In this paper, we will start by providing background on relevant existing work. Then, we will present the overall architecture of our system and will explain the requirements and design choices we have made. Also we are going to present results, discuss current and future steps, and finish with a forward-looking conclusion. We hope that this work will help create a future, in which not only resources are saved and the environment is protected, but also human life will become less stressful with enhanced efficiency and creativity.

II. Background

The IoT is an important component of current and future human life. At the moment, the number of connected devices is greater than the estimate of 22 billion [3], which is nearly three

978-1-5090-6231-7/17 $31.00 © 2017 IEEE
DOI 10.1109/WAINA.2017.77

times greater than the number of Earth's inhabitants. However, the problem is that most devices are incompatible with each other, and some of them are used for other purposes (not IoT). Have you ever thought about what kind of information can you get from a fitness tracker? Using the accelerometer and gyroscope you can get information about if person is now running or just walking, is he/she eating or swimming in the pool. However, the most important part is the ability to work with this information. On the basis of data from the heart rate sensor, you can get a small feature on the state of human health and, if required, suggest various treatments. Also, this information is very useful in medicine, the physician can better evaluate diseases based on previous data. For example, since the disease first spotted, how often problems happened and so on.

A. *Smart Buildings and Cloud Computing*

The IoT can be used in almost all spheres of human life. In this paper, we will focus on Smart Buildings and Human-Building interactions. There have been many different studies on the topic, which uses different kinds of sensors, frameworks and, sometimes, robots. The creation of a single and simple platform that can communicate with all types of sensors and robots is a very important step towards the development of Smart Buildings. It should be noted that it is impossible to use large computers for computing in each smart building. In this regard, there is need for Cloud Computing (CC) where computation is viewed as a "utility". In a similar sense, with modern power and water networks, cloud users do not need to own the means of production or distribution (i.e. power generators, water sources and distribution networks): they just need to connect with the cloud service and let the distant distributed computations, storage, and code resources in a transparent fashion (not knowing the whereabouts or the specifics of them) to do all the job done with high reliability.

Of course, cloud computing also has some limitations. For example, it is not effectively connected to the physical world in the way that a situated robotic agent would be [18]. In this regard, there is the problem of creating a service view of a Human-Robot Cloud, in our case a Human-Sensor Cloud. A further stage of development of the system for Smart Buildings is to create a system of interaction between sensors and robots in buildings and humans. Frameworks for human-machine systems [19] are created and used as the basis for interaction with robots, and provide easy and fast connection with new devices instead of old ones. In fact, the connection and further work with new devices strongly affects the level of performance. Creating a platform for connection and operation of devices of various types is necessary to build the right network.

B. *Microservices*

Microservices are very useful for the IoT. The Microservice software architecture is an approach for service-oriented development which popularity is increasing now because of growing interest to parallel computations. Designed and built for microservices architecture, Jolie programming language was developed to work directly with a service-oriented paradigm, which distinguishes it from other popular languages like C#, Java, or Python. This means that the language contains features that are unique to this approach, i.e. example representation of building blocks. In object-oriented languages, there are usually classes or functions; in Jolie building blocks are services themselves.

The Jolie programming language [31], [4] was created to maximize the use of the microservices architectural style. In Jolie, microservices are first-class citizens: every microservice can be reused, orchestrated, and aggregated with others [29]. This approach brings simplicity in components management, reducing development and maintenance costs, and supporting distributed deployments [10].

The development of Jolie followed a major formalization effort on workflow and service composition languages, and the EU Project SENSORIA [1] has successfully produced a plethora of models for reasoning about composition of services (e.g., [15], [22], [23]). On the mathematical side, the formal semantics of Jolie [12], [11], [28] have been inspired by different process calculi, such as CCS [26] and the π-calculus [27]. From a practical point of view, however, Jolie is a descendant of standards for Service-Oriented Computing such as, for example, WS-BPEL [6]. With both theoretical and practical influences, Jolie is a suitable candidate for the application of recent research techniques, e.g., runtime adaptation [33], process-aware web applications [30],or correctness-by-construction in concurrent software [8].

III. Architecture

At this stage, the main goal is to create a small and simple system, where major responsibility was taken by Jolie. The system architecture scheme is shown on a Figure II.1. In order to achieve the goal, the entire process was divided in three steps.

A. *Step 1: Connecting and configuring sensors*

The first step was to connect and configure sensors and Raspberri Pi through BLE. The main data that is temperature, humidity and luminosity was collected using CC2650 SensorTags, because they contain all necessary sensors, they are compact and require low power consumption (just a coin cell battery). Working with sensors are very similar to the creation of sketches for Arduino: C code is written with the necessary libraries for the SensorTag, sensors that are used and ways of exchanging information with these sensors. In our case, data was sent via BLE, as it is very comfortable to use and configure. Also we used a door sensor from Aeon Labs, which works on the Z-Wave protocol that was used to create a monitoring system entrance/exit to the premises. To work with this device, HomeOS [20] was used, which is written usinge C# and has all necessary functionality to work with the devices of this company in the protocol Z-Wave.

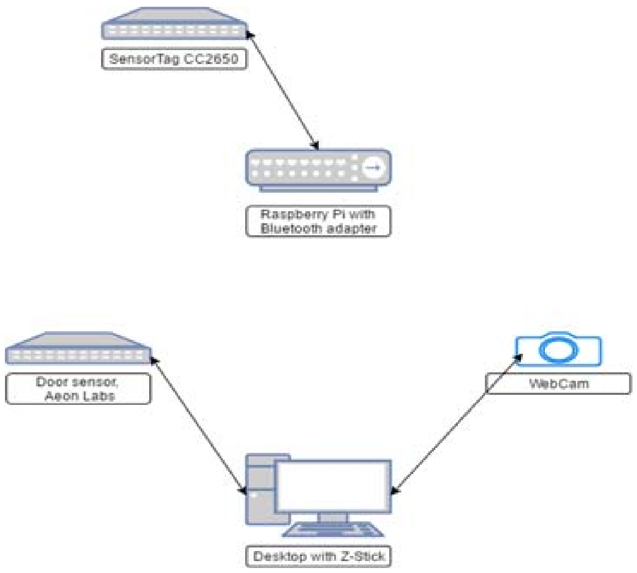

Figure II.1. A scheme of devices and their connections

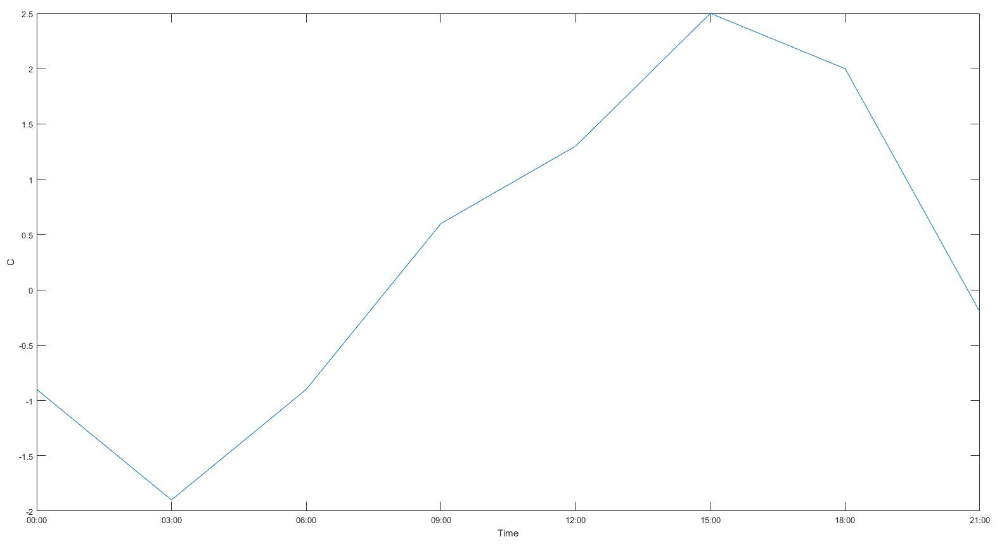

Figure III.1. A graph of the outdoor temperature. Axis of abscissae: Time of the day. Axis of ordinates: Celsius degrees

B. Step 2: Coding in Jolie to work with sensors

The next step was to write simple code to work with sensors using Jolie. A major advantage of this approach is code reusability and scalability. Our system will support different types of sensors, because main logic of their connectivity and data extraction is similar for most of them. Also, product support becomes easier because of code reusability. The same service can be used for several sensors. From this advantage we receive some kind of bug avoidance. With reusable code it is easier to determine a bug, if it will appear in several modules. A third advantage that may be interesting for future collaborators is simplicity. Jolie divides all system logic into small parts: we have several services that are responsible for each sensor or each action, the naming of each block is intuitively understandable, so these language features increase code readability. In addition, one more significant advantage is working with Java code in Jolie. Thus, the part of the work with getting data from BLE devices was written using Java programming language and was divided into simple functions, e.g. connecting to devices and data retrieval, which is called from Jolie. Thus, the code is easy to read, clean, and ready for further work as a client for other sensors.

C. Step 3: Data Collection

The last step is data collection. From SensorTags, we read data about room temperature, humidity, light and pressure. Also we parsed outdoor temperature from the meteorological data gathered through the Internet (Figure III.1). All of these datasets were recorded in .csv format for later processing and graphing in MATLAB (we used it for graphs inside this paper). Data from Aeon Labs sensor is the number of openings and closings of the door. Also we were monitoring incoming and outgoing people using the webcam for accuracy check by ourselves, which recorded in .csv and .jpg format respectively. Also we developed a way to determine who is inside the room by using the MiBand2 fitness-trackers. They were also sending a BLE signal with mac address of device which our Raspberry Pi was capturing as soon as the person came inside the room. This is also how we determined the number of people inside the room and also a person recognition by device mac address.

It is also worth to note that all codes were working on Raspberry Pi, which has all necessary libraries and connections. Despite its apparent simplicity, there were several issues at work with devices. The first serious problem was the lack of libraries for Java to work with BLE devices. But we found a great and simple library for Intel Edison devices and used it. The next issue was working with Z-Wave devices in HomeOS, which periodically generated exceptions and did not connect to the device. However, all these problems were solved, which allowed us to come to the result, which will be described in the next section.

IV. Results

We have developed a platform to work with the SensorTags based on Jolie and Java programming languages. Despite its small size, it obtained most of the data, with which we were working. The ability to embed Java code allows to add necessary segments and give more functionality and usability to Jolie language. By the way, Jolie language is still developing with a worldwide community. So, maybe ine day we won't need to embed a Java code in it.

At this stage, our goal was to build the prototype of device that is working with sensors, understand how they work and how to interact with them and get ideas for the further development of the project. In order to analyze performances, we have placed sensors in three rooms: the students' rooms in the dormitory and the laboratory of Innopolis University (Russian Federation), in which we spend most time of the day. It is worth noting that both rooms can be ventilated and fitted with lights and heaters.

In the students' rooms we set the Raspberry Pi with the connected Bluetooth adapter and the SensorTag to obtain data on the temperature. For tracking of location of the student in the room, we used data from a fitness-tracker MiBand2, using its mac address. Since the room is small we could accurately determine when the student enters and when he/she leaves. The collected dataset has been used to construct graps of temperature, humidity, light and pressure (Figures IV.1, IV.2, IV.3, IV.4) in the room, which will later be used to optimize environment conditions (future development and researches).

Regarding humidity trend, the ideal value should be 40-50% [5]. In our experiment humidity is always lower than 30%, which causes discomfort and can lead to illness, and more in general does not support students to perform at their best in daily intellectual activities.

For the light sensor it is worth noticing that in the second room values are never greater than 200 lux, while in first room there is an average of 600 lux. The reason is that the second room is located between two building and never receive sufficient sunlight, while the first has an open view from a luminous window.

For now, system is just collecting the data from sensors and devices. What we are planning is to connect some household appliances to our system and allow it to configure environment automatically for users needs.

In the laboratory, we placed more devices and used multiple platforms to work with them. This time SensorTag also sent data about room temperature, humidity, light and pressure. Also, we used a sensor of opening/closing doors and a web camera to record those who entered/exited the room just to learn how to transfer this data inside our system, without people recognition for now. But web camera was equipped with OpenCV and allowed to take photos of people that were in the lab, so we collected a small dataset with photos of lab visitors. Using a sensor on the door, we determined how many people were in the room in each moment of time.

As noted earlier, for obtaining data on temperature, humidity and light sensors, their metrics were saved to file which we were using form graphs in MATLAB and which will use for computations in future.

We used two programming language: Java and Jolie, which were running on the Raspberry Pi. Work with other sensors and a camera was assigned to HomeOS, which has all the necessary functionality for working with the devices.

This completes the main work of the first stage. Plans for further development of the project and data will be described in the next section.

V. Future Steps

Despite of having solved many of the issues related to obtaining data from the sensors and its further processing, in order to get the targeted result there are further aspects that need to be taken into consideration. First, we must configure all SensorTags to work on the ZigBee protocol. Also we are going to set up these sensors in a variety of classrooms at our university, and there is a need to build a network or a protocol for communication between devices and further processing using the Jolie language. Second, we need to expand the functionality of this to work with all the devices. In this case, we will have to write the modules for BLE, ZigBee and Z-Wave devices to continue to connect easily and share data between smart devices. In addition, we want to rewrite the module for working with the Aeon Labs sensors using Jolie language.

As we briefly mentioned before, we need to place all sensors and the Raspberry Pi in the classrooms of our University and in

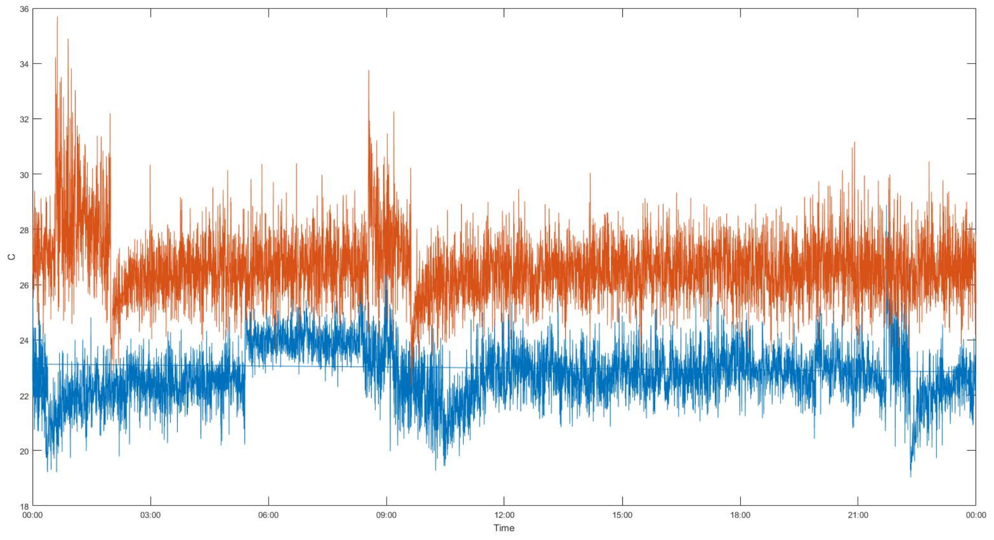

Figure IV.1. Temperature in the rooms. Blue figure in first room and red in second. Axis of abscissae: Time of the day. Axis of ordinates: Celsius degrees

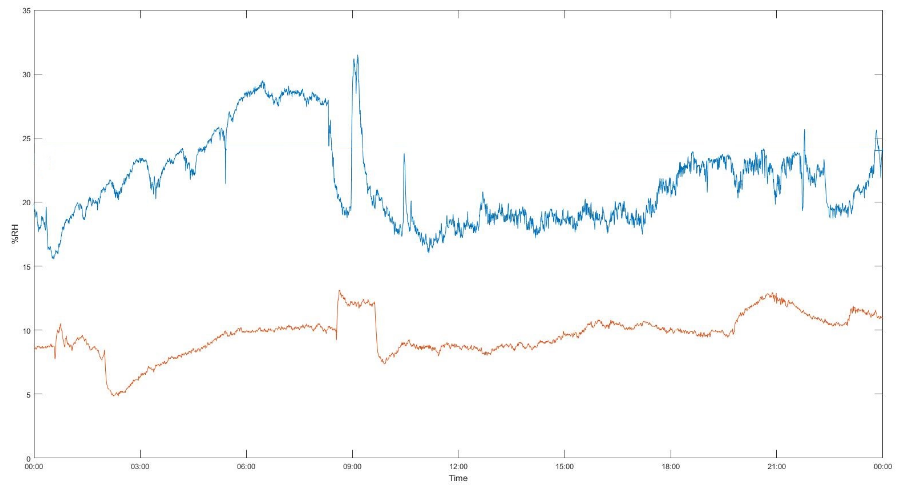

Figure IV.2. Humidity in the room. Axis of abscissae: Time of the day. Axis of ordinates: Percentage of Relative humidity (RH)

the students' rooms and start collecting data. In the rooms of the students, system will continue collecting data on temperature, including the data on humidity and lighting as well, and we are going to install a window-opening sensor that will allow us to get better dataset to determine the preferred mode of temperature, light and humidity in the room and drawing a small daily schedule of a student. In the classrooms, we will also install a small cameras to track the number of people in the room and launch a small bot to Telegram messenger to gather feedback from students in terms of comfort temperature and humidity in the auditorium. Determining number of people in a class is also a tricky task. Possible, we may use some of the existing approaches, like OpenCV, etc.

In the lab, we will add a motion sensor and several relays to control lights and power sockets in the room that will be activated either automatically or manually through a central control system. In the longer term, we also envision monitoring outside areas, such as the building parking spaces [14], outdoor and indoor vegetation [13], and also connecting with social-driven recommendation systems to create customized hospitality tours in large buildings [7].

Based on the obtained data, which will take about a month to collect, we will built a prediction system for students and

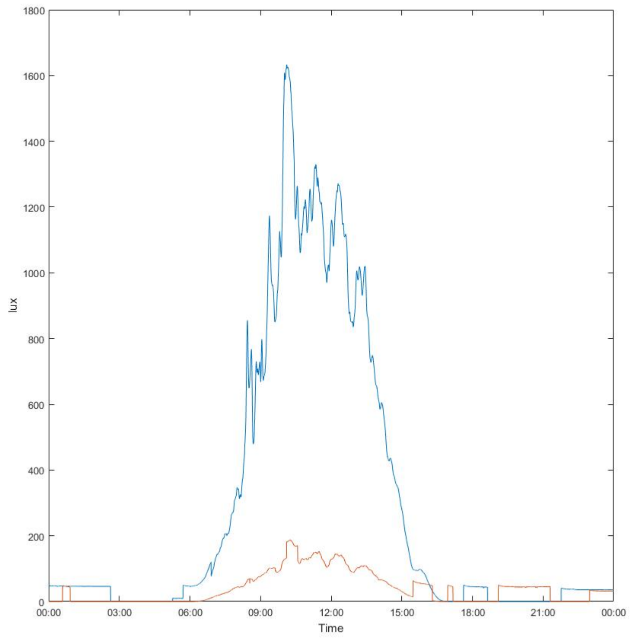

Figure IV.3. A graph of the data from light sensor in the rooms. Axis of abscissae: Time of the day. Axis of ordinates: Lux

employees, which will be checked by a simple small bot in Telegram. Thus, we will improve our result in order to create a small AI system that will be responsible for the control of lighting, temperature, electricity and comfort level in the room.

This will lead to the creation of a unified platform, unified control center, written in Jolie that will allow:

1) To develop a system based on microservices using Jolie,
2) To incorporate new devices according to one of three protocols,
3) To track the preferences of students and staff in the environment (Innopolis University),
4) To control the flow of electricity to the premises and, if possible, to suggest ways to reduce costs,
5) Develop the idea of Human-Building Interaction and technologies on campus and the entire city.

On the software engineering side, instead, all the development process have to be streamlined and organized in a more formal way, from requirements elicitation to deploy and testing. Formal techniques will be here of major help [24], [25].

VI. Conclusions

In this paper we presented preliminary steps toward the transformation of the Innopolis University building and students dorms into an effective smart building. Tracking some key environmental values emphasized the need for an optimization to lead to both to energy save and improved living conditions. In the future it is not impossible to imagine buildings capable to adapt and self-configure depending on environmental conditions and human needs, in the same way as modern software shows the same flexibility [21].

Jolie demonstrated to be flexible and simple enough for working with microservices and the Internet of Things: code is easy to write, to deploy and devices are easy t connect. The overall scenario looks promising to be repicated in several projects related to smart homes and cities.

A large obstacle to the development is existence of a narrow set of tools for the job, and limited documentation, that will require specific attention on a case to case basis. However, the Internet community and students of many European Universities supported us in our enterprise. In the future, this work will allow us to create a system of HBI that is easier to use and more affordable.

Acknowledgements

The authors received logistic and financial support by Innopolis University. We would like to thank Daniel Johnston and Ruslan Tushov for linguistic assistance and all the colleagues and students who made this work possible.

References

[1] EU Project SENSORIA. Accessed April 2016. http://www.sensoria-ist.eu/.
[2] Internet of things global standards initiative. http://www.itu.int/en/ITU-T/gsi/iot/Pages/default.aspx. Online; accessed 25th of October 2016.
[3] Internet of things (iot): number of connected devices worldwide from 2012 to 2020. https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/. Online; accessed 25th of October 2016.
[4] Jolie Programming Language. Accessed April 2016. http://www.jolie-lang.org/.

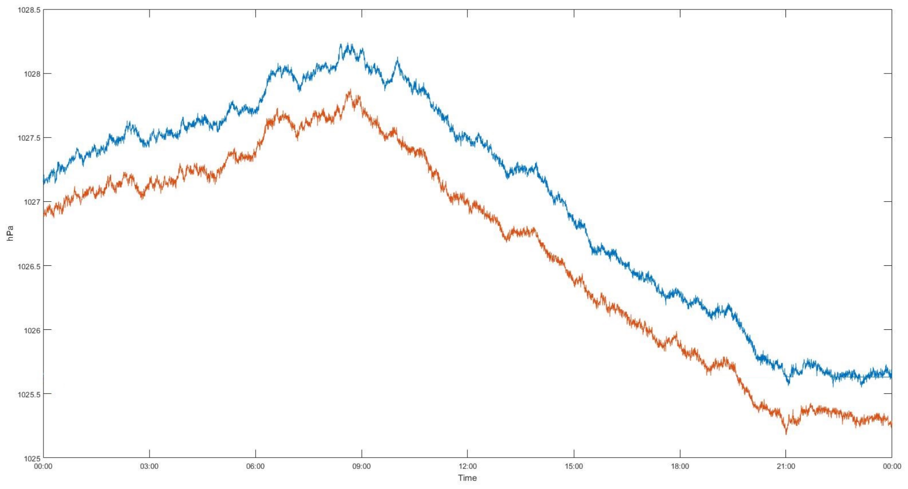

Figure IV.4. A graph of the pressure in the rooms. Axis of abscissae: Time of the day. Axis of ordinates: millibars

[5] Relative humidity and your home. http://www.thermastor.com/information/relative-humidity-and-your-home.aspx. Online; accessed 25th of October 2016.

[6] WS-BPEL OASIS Web Services Business Process Execution Language. accessed April 2016. http://docs.oasis-open.org/wsbpel/2.0/wsbpel-specification-draft.html.

[7] Kanna Al Falahi, Nikolaos Mavridis, and Yacine Atif. Social networks and recommender systems: a world of current and future synergies. In *Computational Social Networks*, pages 445–465. Springer, 2012.

[8] Marco Carbone and Fabrizio Montesi. Deadlock-freedom-by-design: multiparty asynchronous global programming. In *POPL*, pages 263–274, 2013.

[9] Nicola Dragoni, Manuel Mazzara, Saverio Giallorenzo, Fabrizio Montesi, Alberto Luch Lafuente, Ruslan Mustafin, and Larisa Safina. Microservices: yesterday, today, and tomorrow. In *Present and Ulterior Software Engineering*. Springer Berlin Heidelberg, 2017.

[10] M. Fowler. Microservice Trade-Offs. http://martinfowler.com/articles/microservice-trade-offs.html, (2015).

[11] C. Guidi, I. Lanese, F. Montesi, and G. Zavattaro. Dynamic error handling in service oriented applications. *Fundam. Inform.*, 95(1):73–102, 2009.

[12] C. Guidi, R. Lucchi, G. Zavattaro, N. Busi, and R. Gorrieri. Sock: a calculus for service oriented computing. In *ICSOC, volume 4294 of LNCS*, pages 327–338. Springer, 2006.

[13] Abdulhamid Haidar, Haiwei Dong, and Nikolaos Mavridis. Image-based date fruit classification. In *Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2012 4th International Congress on*, pages 357–363. IEEE, 2012.

[14] Jermsak Jermsurawong, Mian Umair Ahsan, Abdulhamid Haidar, Haiwei Dong, and Nikolaos Mavridis. Car parking vacancy detection and its application in 24-hour statistical analysis. In *Frontiers of Information Technology (FIT), 2012 10th International Conference on*, pages 84–90. IEEE, 2012.

[15] R. Lucchi and M. Mazzara. A pi-calculus based semantics for WS-BPEL. *J. Log. Algebr. Program.*, 70(1):96–118, 2007.

[16] Larisa Safina Ivan Lanese Manuel Mazzara, Ruslan Mustafin. Towards microservices and beyond: An incoming paradigm shift in distributed computing. *arXiv preprint arXiv:1610.01778*, 2016.

[17] Ilaria Baroni Marco Nalin and Manuel Mazzara. A holistic infrastructure to support elderlies' independent living. *Encyclopedia of E-Health and Telemedicine, IGI Global*, 2016.

[18] Nikolaos Mavridis, Thirimachos Bourlai, and Dimitri Ognibenes. The human-robot cloud: Situated collective intelligence on demand. *Cyber Technology in Automation, Control, and Intelligent Systems (CYBER), 2012 IEEE International Conference*.

[19] Nikolaos Mavridis, Stasinos Konstantopoulos, Ioannis A. Vetsikas, I. Heldal, Pythagoras Karampiperis, G. Mathiason, Serge Thill, K. Stathis, and Vangelis Karkaletsis. CLIC: A framework for distributed, on-demand, human-machine cognitive systems. *CoRR*, abs/1312.2242, 2013.

[20] Nikolaos Mavridis, Georgios Pierris, Chiraz BenAbdelkader, Aleksandar Krstikj, and Christos Karaiskos. Smart buildings and the human-machine cloud. In *GCC Conference and Exhibition (GCCCE), 2015 IEEE 8th. IEEE, 2015*.

[21] Dragoni Nicola Zhou Mu. Mazzara, Manuel. Dependable workflow reconfiguration in WS-BPEL. In *Proceedings of the 5th Nordic Workshop on Dependability and Security*, 2011.

[22] M. Mazzara, F. Abouzaid, N. Dragoni, and A. Bhattacharyya. Toward design, modelling and analysis of dynamic workflow reconfigurations - A process algebra perspective. In *Web Services and Formal Methods - 8th International Workshop, WS-FM*, pages 64–78, 2011.

[23] Manuel Mazzara. *Towards Abstractions for Web Services Composition*. PhD thesis, University of Bologna, 2006.

[24] Manuel Mazzara. Deriving specifications of dependable systems: toward a method. In *12th European Workshop on Dependable Computing (EWDC)*, 2009.

[25] Manuel Mazzara. On methods for the formal specification of fault tolerant systems. In *4th International Conference on Dependability (DEPEND)*, 2011.

[26] R. Robin Milner. *A calculus of communicating systems*. Lecture notes in computer science. Springer-Verlag, Berlin, New York, 1980.

[27] Robin Milner, Joachim Parrow, and David Walker. A calculus of mobile processes, I and II. *Information and Computation*, 100(1):1–40,41–77, September 1992.

[28] F. Montesi and M. Carbone. Programming Services with Correlation Sets. In *Proc. of Service-Oriented Computing - 9th International Conference, ICSOC*, pages 125–141, 2011.

[29] Fabrizio Montesi. JOLIE: a Service-oriented Programming Language. Master's thesis, University of Bologna, 2010.

[30] Fabrizio Montesi. Process-aware web programming with Jolie. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, SAC '13, pages 761–763, New York, NY, USA, 2013. ACM.

[31] Fabrizio Montesi, Claudio Guidi, and Gianluigi Zavattaro. Service-oriented programming with jolie. In *Web Services Foundations*, pages 81–107. 2014.

[32] Renato Jorge Caleira Nunes. Home automation - a step towards better energy management. In *INTERNATIONAL CONFERENCE ON RENEWABLE ENERGIES AND POWER QUALITY (ICREPQ 2017)*.

[33] Mila Dalla Preda, Saverio Giallorenzo, Ivan Lanese, Jacopo Mauro, and Maurizio Gabbrielli. AIOCJ: A choreographic framework for safe adaptive distributed applications. In *Software Language Engineering - 7th International Conference, SLE 2014, Västerås, Sweden, September 15-16, 2014. Proceedings*, pages 161–170, 2014.