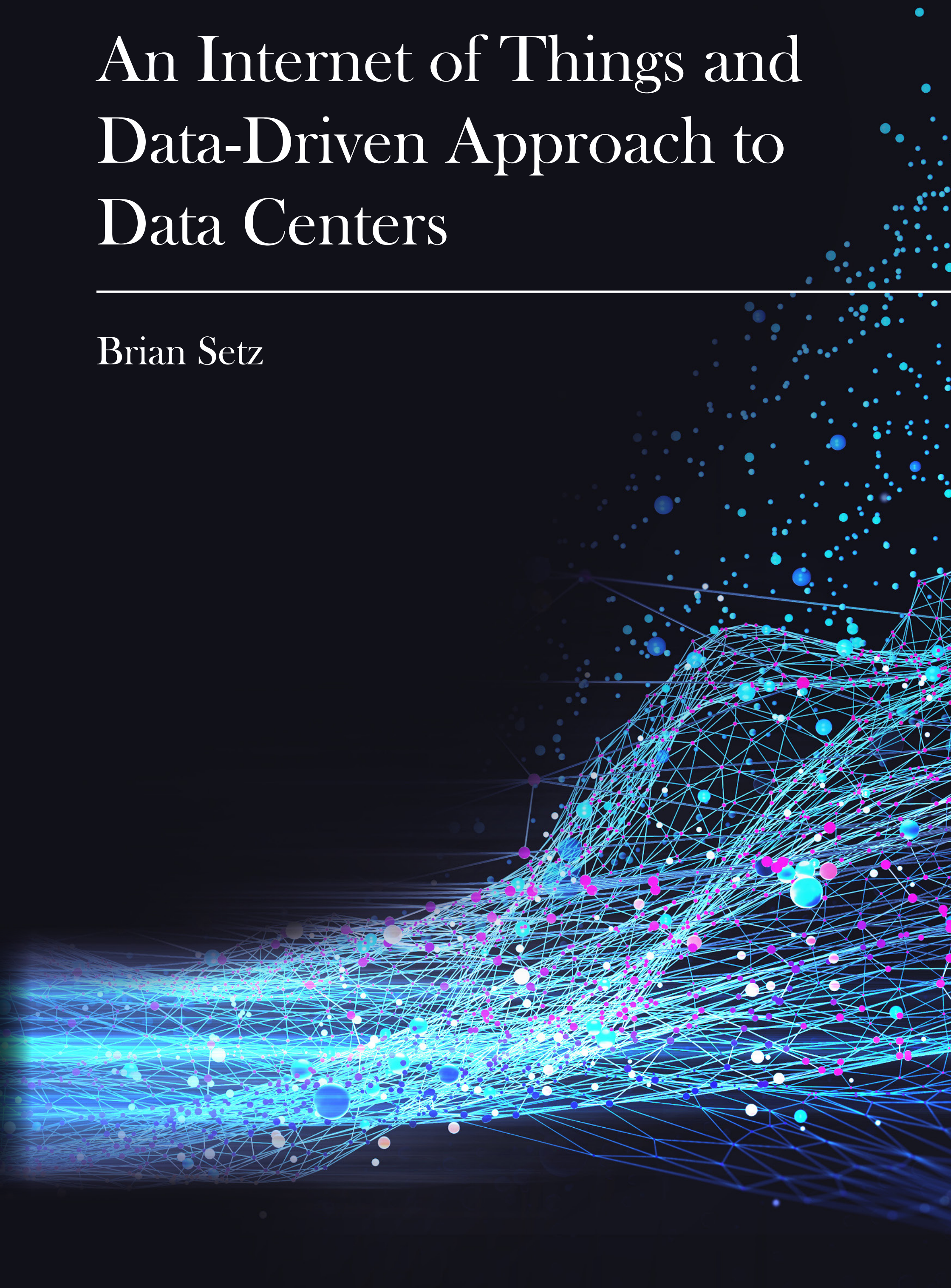
An Internet of Things and Data-Driven Approach to Data Centers

Brian Setz



Universität Stuttgart

An Internet of Things and Data-Driven Approach to Data Centers

Von der Fakultät für Informatik, Elektrotechnik und Informationstechnik der Universität Stuttgart zur Erlangung der Würde eines Doktors der Naturwissenschaften (Dr. rer. nat.) genehmigte Abhandlung

Vorgelegt von

Brian Setz

aus Zuidhorn, den Niederlanden

Hauptberichter:	Prof. Dr. Marco Aiello
Mitberichter:	Prof. Dr. Alexander Lazovik
	Prof. Dr. Massimo Villari
Tag der mündlichen Prüfung:	16.12.2022

Institut für Architektur von Anwendungssystemen
Abteilung Service Computing

2022

The research was partially supported by the Netherlands Organization for Scientific Research (NWO) in the framework of the Indo-Dutch Science Industry Collaboration program with project NextGenSmart DC (629.002.102). Cover art by Siarhei ©.

Contents

ZUSAMMENFASSUNG

Datenzentren sind ein essentieller Teil der modernen Gesellschaft, da sie gewährleisten, dass IKT-Dienste verfügbar und responsiv bleiben. Aktuelle Trends zeigen, dass Datenzentren immer größer werden und damit auch ihr bereits erheblicher Energieverbrauch weiter steigt. Da Regierungen weltweit Schwierigkeiten damit haben ihre Nachhaltigkeitsziele zu erreichen, besteht ein dringender Bedarf nach effizienten und nachhaltigen Datenzentren. Das Internet of Things (IoT)-Paradigma spielt eine wichtige Rolle darin, die Gebäudeeffizienz und -nachhaltigkeit zu steigern. Die Domänen Smart Home und Büro demonstrieren den erfolgreichen Einsatz von IoT zum Einsparen von Energie durch kontinuierliche Überwachung und Optimierung. Die Erweiterung dieser Techniken auf Datenzentren verspricht das Smarte wie bereits bei Smart Homes und Büros ins Datenzentrum zu bringen. Allerdings müssen dazu noch einige Herausforderungen überwunden werden bis das Konzept des smarten und optimierten Datenzentrums umgesetzt werden kann. Diese Arbeit befasst sich mit einigen dieser Herausforderungen aus einem datengetriebenen IoT-Blickwinkel.

Zunächst wird das Konzept eines nachhaltigen Datenzentrums fokussiert. Durch Interviews mit dem Personal von sieben Datenzentren werden die aktuellen Best Practices zu Nachhaltigkeit und ihre Anwendung in Datenzentren untersucht. Es werden zahlreiche Probleme und Verbesserungsmöglichkeiten

identifiziert, darunter die Rolle der Flüssigkühlung. Daraus folgt die Demonstration des Potentials einer Tauchkühlung und eine Analyse der möglichen Energieeinsparungen, die durch den Ersatz klassischer Luftkühlungen erreicht werden können. Andere Faktoren wie Leistungsdichte und Kosten werden dabei ebenfalls berücksichtigt. Um den Effekt der Umsetzung von Best Practices oder der Veränderung von Kühlsystemen zu messen, ist es von äußerster Bedeutsamkeit Messungen durchzuführen. Um zu verstehen was gemessen und auf welchem Level gemessen werden soll, wird eine Taxonomie aus über 100 Datenzetrumsmetriken erstellt. Dabei werden einige Probleme und Defizite in Bezug auf Datenzetrumsmetriken erkannt, darunter die Herausforderungen, die sich Co-Location-Datenzentren stellen müssen und die Notwendigkeit für einen IoT-orientierten Ansatz für die Datenzentrumsüberwachung.

Daher wird die Datenzentrumsüberwachung und die Rolle von IoT betrachtet. Aus der Untersuchung des Datengenerierungspotentials einer Echtzeit- und IoT-getriebenen Datenzentrumsüberwachung sticht der Fakt hervor, dass die generierten Daten durch ihre schiere Größe einen negativen Effekt auf die Netzwerkbandbreite des Datenzentrums haben. Um diesem Problem entgegenzuwirken wird eine Edge-basierte Lösung für die Datenzentrumsüberwachung, die in der Lage ist mit diese Bandbreitenlimitierungen umzugehen und gleichzeitig eine Überwachung mit geringer Latenz ermöglicht, vorgeschlagen. Der Ansatz ist stark in das Datenzentrum integriert, indem er die Vorteile des traditionellen Netzwerklayouts nutzt. IoT-Hubs spielen eine zentrale Rolle bei der Vernetzung von Sensoren und Aktoren und bei der Verarbeitung der Daten, die bei der Überwachung anfallen. Es gibt viele Open-Source-IoT-Hubs und aus einer anfänglichen Auswahl von zwanzig Hubs werden vier im Detail analysiert. Indem jedes der vier Systeme zahlreichen Anwendungsfällen und Kriterien ausgesetzt wird, werden die Stärken und Schwächen jedes Systems identifiziert. Und es wird eine verallgemeinerte IoT-Hub-Architektur basierend auf den Gemeinsamkeiten der vier Systeme aufgebaut. Darüber hinaus wird das Problem der Überwachung von Datenzentren, bei denen der Betreiber des Datenzentrums keinen Zugriff auf das Betriebssystem oder das Gehäuse der IT-Ausrüstung hat, untersucht.

Es wird ein Überwachungsansatz vorgeschlagen bei dem die Privatsphäre gewahrt wird, da nur IoT-Sensoren nötig sind, die außerhalb des Gehäuses platziert werden. Es werden Modelle entwickelt, die in der Lage sind, die CPU-Auslastung und die Temperatur ohne Zugriff auf das Betriebssystem präzise zu schätzen und ermöglichen so eine detaillierte Überwachung von Co-Location-Datenzentren.

Und abschließend wird die Domäne der Büros statt der Domäne der Datenzentren betrachtet. Die Bedeutung intelligenter Reaktionen auf Aktivitäten, die in Gebäuden stattfinden, wird demonstriert indem zahlreiche IoT-Sensoren und -Aktoren in einer Büroumgebung eingesetzt werden und eine Aktivitätserkennung durchgeführt wird, um den Zustand eines Raums oder Bereichs innerhalb eines Raums zu bestimmen. Mit Hilfe von AI Planning reagiert das Gebäude auf Zustandsänderungen, um den Energiebedarf des Gebäudes zu optimieren. Die Ähnlichkeiten in der zugrunde liegenden IoT-Architektur für Gebäude und Datenzentren werden dabei hervorgehoben. Außerdem wird ein Ansatz zur Gerätelastplanung vorgeschlagen, der erneuerbare Energien, Energiespeicher, benachbarte Prosumer, Preissignale und Planungsbeschränkungen berücksichtigt. Der parallele Suchalgorithmus mit einheitlichen Kosten verwendet diese Parameter, um den optimalen Zeitplan zu finden, der die ökonomischen Einsparungen für den Endverbraucher maximiert. Eine Microservice-Architektur wird für die Datenerfassung in Echtzeit verwendet. Die Einsparungen durch die Verfügbarkeit eines lokalen Energiespeichers werden hervorgehoben.

Zusammenfassend lässt sich sagen, dass die Beiträge aus dieser Arbeit den Weg für smarte und nachhaltige Datenzentren durch den Einsatz eines IoT- und datengetrieben Ansatzes ebnen. Des Weiteren wird die Arbeit auf den Bereich der smarten Büros erweitert.

Abstract

Data centers are a fundamental part of modern society, as they ensure that today's ICT services are available and responsive. The current trends show that data centers are increasing in size, which corresponds with an increase in their already significant energy footprint. As governments around the globe struggle to achieve their desired sustainability goals, the need for efficient and sustainable data centers is urgent. The Internet of Things (IoT) paradigm plays an important role in increasing the efficiency and sustainability of buildings. The smart home and office domains demonstrate the successful application of IoT in order to achieve energy savings through continuous monitoring and optimization of the building environment. Extending these techniques to the data center domain promises to bring the smartness previously seen in homes and offices to the data center. However, there are challenges that need to be overcome before the concept of a smart and optimized data center becomes a reality. In this thesis, a number of these challenges are addressed from a data-driven and IoT point of view.

First, we focus on the concept of a sustainable data center. Through interviews with personnel of seven data centers we investigate the current state of sustainability best practices and their application to data centers. Numerous issues and areas for improvement are identified, including the role for liquid cooling. Thus, we show the potential of immersion cooling, and analyze the potential energy savings that can be achieved by replacing traditional air cooling solutions. We also consider other factors such as power density and costs. To measure the effect of implementing best practices or making changes in the cooling system, it is critical to take measurements. To understand what to measure, and at which level to measure it, we construct a taxonomy of over one hundred data center metrics. We also identify several issues and shortcomings with regards to data center metrics, including the challenges co-location data centers face when it comes to data collection, as well as the need for an IoT-oriented approach to data center monitoring.

Thus, we shift our attention to data center monitoring and the role of IoT. By investigating the data generation potential of real-time and IoT-driven data center monitoring, we highlight the fact that the generated data has a negative effect on the data center's networking bandwidth due to its sheer volume. To counter this issue, we propose an edge-based solution to data center monitoring that is able to address these bandwidth limitations while still ensuring low latency monitoring. The approach is tightly integrated with the data center by taking advantage of the traditional network layout. IoT hubs play a central role in enabling interconnectivity between sensors and actuators, and in processing the data that is generated while monitoring. There are many open-source IoT hubs available, and out of an initial selection of twenty hubs, we analyze four in great detail. By subjecting each of the four systems to numerous use cases and criteria, we are able to identify strengths and weaknesses for each system. And we synthesize a generalized IoT hub architecture, based on the commonalities between the four systems. Furthermore, we investigate the problem of monitoring co-location data centers, where the data center operator has no access to the operating system or chassis of the IT equipment. We propose an approach to monitoring that is privacy-preserving by only requiring IoT sensors that are placed external to the chassis. We develop models that are able to accurately estimate the CPU usage and temperature without access to the operating system, enabling detailed monitoring of co-location data centers.

And finally, we switch from the data center domain to the office domain. We demonstrate the importance of intelligent responses to activities that occur within buildings by deploying numerous IoT sensors and actuators in an office environment and performing activity recognition to determine the state of a room or area within a room. With the assistance of AI planning, the building reacts to changes in state to optimize the building's energy footprint. We highlight the similarities in the underlying IoT architecture for buildings and data centers. We also propose a device load scheduling approach that considers renewable energy, energy storage, neighboring prosumers, price signals, and scheduling constraints. Our parallel uniform-cost search algorithm uses these parameters to find the optimal schedule that maximizes the economic savings for the end-user. A micro-service architecture is used for the real-time collection of data. We emphasize the savings that are achieved by having local energy storage available.

To summarize, the contributions that are made in this thesis pave the way towards smart and sustainable data centers supported by an IoT and data-driven approach. Furthermore, we also extend our work to the domain of smart offices.

Acknowledgements

As my PhD journey comes to an end, I look back feeling grateful for the possibilities and opportunities given to me throughout this journey. I was fortunate to experience life in two different countries, at two different universities, while pursuing my PhD. The journey started in the Netherlands at the Distributed Systems group of the University of Groningen, and ends in Germany at the Service Computing department of the University of Stuttgart. My heartfelt thanks and appreciation goes out to everyone who has supported and assisted me on this journey, both in Groningen and in Stuttgart.

My sincere thanks go out to Marco Aiello, my *Promotor* in Groningen and my *Hauptberichter* in Stuttgart. I am very grateful for being given the chance to pursue a PhD under the NWO NextGenSmart DC project. You have always made time to discuss my research and answer any of my questions. During my PhD, you have provided me with critical suggestions and invaluable feedback on my publications and on my thesis. I appreciate the freedom I received from you to steer my research in the direction I wanted, while at the same time correcting my course whenever I started to feel lost. I also enjoyed being involved in the process of setting up a new department at the University of Stuttgart, and developing academic courses from the ground up. I have truly learned a lot from you, thank you.

I would like to thank Alexander Lazovik, my *Copromotor* in Groningen. I have thoroughly enjoyed collaborating with you during my time at the University of Groningen. Your enthusiasm for highly technical discussions was infectious and inspiring. Your valuable feedback has helped me to push my technical abilities to a higher level. Furthermore, your passion for the Scala programming language has stuck with me throughout my journey. I also thoroughly appreciate the many years we spent working together on the Web and Cloud Computing course.

While pursuing my PhD, I have been fortunate to be part of two research groups: the Service Computing department, and the Distributed Systems group. I want to thank all my colleagues at the Service Computing department: Ilche Georgievski, Ebaa Alnazer, Nasiru Aboki, Kawsar Haghshenas, Yuchen Liu, Robin Pesl, Mohammadmehdi Peiro, Denesa Zyberaj, and Marco Aiello. I also want to thank all my colleagues at the Distributed Systems group: Azkario Rizky Pratama, Michel Medema, Tuan Anh Nguyen, Faris Nizamic, Laura Fiorini, Heerko Groefsema, Frank Blaauw, Ang Sha, Talko Dijkhuis, Andrea Pagani, Alexander Lazovik, and Marco Aiello. I was able to collaborate on research with many of you, and for that I am grateful. I also thank the director and founder of the Institute of Architecture of Application Systems, Frank Leymann, for the warm welcome in Stuttgart, and for his feedback during the institute's doctoral conferences.

An extra thank you goes out to the people I shared an office with in Groningen, Azkario and Michel, and in Stuttgart, Ilche and Ebaa, and Nasiru. Azkario, I enjoyed our conversations and it was always entertaining to discover words that Indonesian and Dutch languages have in common. Michel, even though I moved to Stuttgart not long after you became a PhD student, I appreciate our collaborations. Ilche, thank you for the numerous discussions and insights, and also for sharing many a laugh in the office. Ebaa, it was great to have another PhD student to share the office with, and I enjoyed working together with you on the Operating Systems course. Nasiru, thank you for our collaboration on the living lab in our office, and I appreciate your never-ending cheerfulness. I would also like to thank Robin for translating the summary of my thesis to German.

Additionally, I would like to thank the team at the start-up Sustainable Buildings. Thank you Faris and Tuan for giving me the opportunity to be involved in the Sustainable Buildings project, I have learned a lot as your technical advisor and founding shareholder. I also extend my thanks to Vlad Tomashpolskyi for our close technical collaborations and fruitful discussions.

Finally, I want to thank my family for always being supportive in every way possible. First, my sincere gratitude goes out to my late aunt Dea Pol, her immeasurable kindness and support will never be forgotten. My parents, Felix Setz and Hennie Setz, have unconditionally supported my PhD journey. Despite the distance between us, I have always felt very close to the both of you, and you were always there with helpful advice whenever I needed it. To my brother, Kevin Setz, I am grateful that even though I moved to another country, our friendship remained as strong as ever. As my PhD journey comes to an end, I wish you all the best with your own journey into parenthood.

CHAPTER 1

INTRODUCTION

Information and Communications Technology (ICT) permeates all aspects of modern society, from health care and education, to our work environments and the way we socialize. To ensure the availability and responsiveness of ICT services, the supporting infrastructure plays a critical role. Typically, this supporting infrastructure is located in a centralized structure, or groups of structures, known as data centers. A data center is dedicated to the centralized accommodation, operation, and interconnection of ICT equipment while providing data storage, processing, and transportation services [GAB+12]. A data center also encompasses all of the facilities and infrastructures for power distribution, Heating, Ventilation and Air Conditioning (HVAC) control, together with the necessary levels of resilience and security that are required to provide the desired service availability levels. It is therefore not surprising that data centers are one of the most dynamic and critical operations in any business [WB15]. Data centers also form the backbone of the modern Internet, it is their computational, storage, and networking resources that enable many of the services that are present on the World Wide Web today.

Modern data centers are massive in size, covering areas of tens of thousands square meters, housing many thousands of individual server racks [CGC16]. For example, the Tahoe Reno 1 co-location data center owned by Switch and located in Nevada (USA) has 120,000 square meters of data center space, and has a power capacity of 120 MW. Another example is the Tulip Data Center in Bengaluru (India) which has 90,000 square meters of space, 12,000 racks, and a peak power supply capacity of 100MW. Of the modern data centers, the hyper-scale data centers are among the largest in existence. These hyper-scale data centers are enormous warehouse-scale facilities that have emerged in parallel with the growing demand for cloud computing, social media platforms, and big data processing. The growth in hyper-scale cloud data centers is one of the major contributors to the increase in ICT-related energy consumption across the globe. However, it is not only the hyper-scale data centers that are experiencing a growth in numbers, in general data centers are experiencing a steady growth both in number and in size [GSS15]. Furthermore, investments in data center services are expected to grow from \$48.90 billion in 2020 to \$105.6 billion in 2026 [Mor21]. The operational costs of data centers are vastly different from those of other enterprises, as less than 5% of the costs are personnel related. Servers are responsible for 45% of the amortized costs, followed by infrastructure (25%), power draw (15%) and networking (15%) [GHMP08]. As the number of data centers grows and their size increases, the demand on the electrical grid grows proportionally.

The environmental impact of ICT and data centers should not be overlooked. In 2013, the world's ICT infrastructure was estimated to consume 1,500 TWh of electricity, which was roughly 10% of the global energy usage. Furthermore, it was reported that data centers are responsible for 14% of the total ICT energy footprint, as well as 3% of the energy consumption of the United States [Mil13]. The overall energy consumption of data centers has been steadily rising by approximately 7% per year [Jon14]. Some predict that the energy needs of data centers will grow from 286 TWh in 2016 to about 321 TWh in 2030 [KW21]. While others predict that the energy usage of ICT and data centers will reach 3,234 TWh and 974 TWh respectively

[And19]. It is also important to note that data centers are expected to be responsible for 24.7% of the ICT energy footprint by 2030 [WZY+22]. This would be a 10% increase compared to 2013. Within the data center, cooling systems and servers are responsible for 50% and 26% of the energy consumption, which is depicted in Figure 1.1. Power conversion losses, network equipment, and lighting are responsible for 11%, 10%, and 3% of a data center's energy consumption respectively. The continuous growth of the data center market has implications where sustainability is concerned.

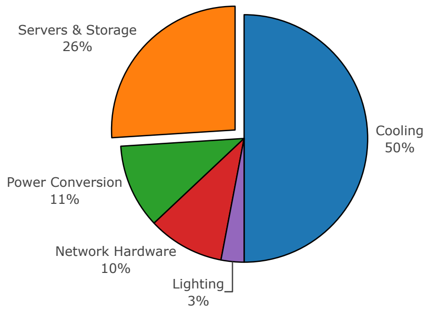


Figure 1.1: Contributors of the energy consumption of data centers [DWF16].

Quantifying the sustainability level of a data center is challenging, as existing metrics typically focus only on a single parameter of the data center [LMG18], such as the facility's water reusage or renewable energy usage. Regardless, it is clear that the energy consumption associated with data centers has significant implications, there is a real risk of energy supply shortage and grid instability as the energy demand keeps increasing [CAB+18]. In fact, the Dutch government banned the construction of new hyper-scale data centers for a period of 9 months in 2022. The ban applies to facilities larger than 100,000 square meters and a power capacity exceeding 70 MW. The reason for this ban is the limited available resources, both in spatial terms, as well as in terms of the power grid. A potential solution to this problem is the use of demand response programs to improve the stability of the grid through the forecasting and shifting of power loads [VCA+20].

However, this requires detailed monitoring and the ability to distinguish between shiftable loads and non-shiftable loads. As governments impose stricter sustainability policies to meet climate goals, the need for a smart and sustainable data center is emerging.

A smart building allows for the monitoring and control of devices and appliances that are connected through a complex network [VPS+19]. The Internet of Things (IoT) has a crucial role in ensuring the connectivity of these devices, as it embraces the idea that devices are always connected to the Internet and are able to provide data and actuate in the physical world [GBMP13]. Homes and offices are becoming a prominent example of this trend as everyday objects are increasingly equipped with digital controllers that are connected to home local networks. Smart buildings are able to optimize their energy consumption by continuously monitoring the state of the environment, understanding what activities are happening in the building, and controlling the IoT devices based on these inputs [NNLA14]. The application of IoT within the context of data center enables *a data-driven and IoT-based approach that can bring the smartness previously seen in homes and offices to the data center domain*.

1.1 Challenges and Research Questions

It is clear that there is a need for an IoT and data-driven approach to data centers to support the concept of an efficient and smart data center. However, there are challenges that need to be overcome to bring the concept of a smart data center closer to reality. In this thesis, we address a number of those challenges. The first step in optimizing the efficiency of data centers is to make Data Center Operators (DCOs) aware of the energy waste and to implement best practices for sustainability. The difficulty lies in understanding the available measures that can be taken in practice, and knowing what issues are addressed by a given measure. Furthermore, it is important to understand which of these measures are used in practice, and which are not.

This assists us with the identification of problem areas and future research directions. Based on these challenges, we define the following research question:

Research Question #1

$\boxed{\text{RQ1}}$ – What measures can be implemented by a data center operator to increase the level of efficiency and sustainability of their data center, and which of these measures are implemented in practice today?

Once we understand the practices that are available to DCOs and the problems they solve, the next challenge is to quantify the results of applying said practices. The data generated from monitoring the data center has a vital role in continuously evaluating the effects of best practices and other policy changes. However, many different parameters can be monitored within the data center environment. The difficulty is to understand what to measure, at what level to measure it, and knowing what standardized measurements are available. We define the second research question as follows:

Research Question #2

$\boxed{\text{RQ2}}$ – How can the effect of policy changes in a data center, such as the implementation of best practices, be measured in a standardized manner?

The data generation potential of large scale IoT deployments is truly astounding. On a world-wide level, the volume of generated IoT data is in the order of Zettabytes [BCC20]. But even on the scale of a single data center, the amount of data that can be generated for monitoring purposes alone is significant, as we show in Chapter 4. In fact, the quantity of monitoring data that has to be transmitted is so large that it has a negative effect on the available bandwidth in the data center. This brings us to our third research question:

Research Question #3

RQ3 – How can the traditional network architecture of a data center be leveraged to support IoT-based real-time monitoring while simultaneously reducing the processing load and bandwidth consumption associated with monitoring?

While collecting data in an enterprise or hyper-scale data center is relatively straight forward, new obstacles arise when attempting to collect data in a co-location data center. In a co-location data center, rack space is rented out and as a result the majority of the IT equipment is owned by third parties. This complicates the situation for DCOs, who no longer have access to the IT equipment, and are unable to directly monitor parameters such as the utilization of the servers. Co-location data centers do enable a unique demand response scheme where customers are charged higher prices during peak energy demand to promote the deferring of workloads [ZLWR15]. However, such an approach does not address the challenges regarding monitoring. Therefore, we pose the following research question:

Research Question #4

RQ4 – With which precision is it possible to monitor servers of a co-location data center while not having access to the operating system or the internals of the server chassis?

Next, we shift our focus from the data center domain to the office domain. At first glance, these two domains appear to be disjoint, but as we will see in this thesis, many parallels exist between smart offices and smart data centers from the IoT point of view. Specifically, we want to understand the role of IoT in supporting smart responses from the building depending on the contextual state, and additionally we focus on the parallels that exist between IoT in offices and data centers. Uncovering these parallels would assist in transferring IoT approaches from the office domain to the data center domain. Thus, the fifth research question is posed as follows:

Research Question #5

RQ5 – What role does the IoT paradigm have in enabling intelligent responses to activities taking place in office buildings, and which similarities and differences exist between IoT architectures for offices and data centers?

The smart grid promises to transform the traditional grid into an intelligent grid that enables actors to cooperate responsively and organically [TA16]. A key feature of the smart grid is the varying energy price signal which is influenced by demand and response, as well as the available energy sources. Taking advantage of these price fluctuations by shifting device loads can result in economic savings for the consumer. Furthermore, IoT allows for the devices to be controlled according to a pre-generated schedule. However, finding an optimal schedule for device loads given scheduling constraints for each device while also considering renewable energy generation, energy storage systems, energy generation by neighboring prosumers, and varying price signals is a complex problem due to the large state space that has to be explored as quickly as possible. This brings us to the final research question:

Research Question #6

RQ6 – How can devices in an office building be scheduled optimally considering the availability of renewable energy, energy storage, neighboring prosumers, varying price signals, and different scheduling constraints, while ensuring timely schedule generation?

1.2 Methodology

The approach taken in this thesis is experimental in nature. The majority of our research follows an empirical methodology. The data that we use is collected from real data centers and real IoT deployments. Furthermore, the research in this thesis is subdivided in three complementary themes: sustainable data centers, monitoring of smart data centers, and smart buildings.

To address research question RQ1, we follow a survey methodology where we interview domain experts using a questionnaire consisting of open questions. Based on the interviews, we quantify the application of best practices across multiple data centers and identify areas for improvement. For research question RQ2, we perform a systematic review and analysis of data center metrics. Through analyzing each metric, we identify inter-metric relationships and potential issues. To answer research question RQ3, we apply an experimental approach on a small scale and extrapolate the results to encompass the entire data center. Additionally, we perform a comparative study between IoT hubs. To evaluate research question RQ4 we take an empirical approach and collect data at a large scale from a real data center. The collected data is used to train and evaluate supervised machine learning models. For the purpose of answering research question RQ5 we perform physical experiments in an office building, supported by an IoT framework we design and develop ourselves. The experiments are performed while the occupants of the building are actively using the affected rooms and areas. And finally, we address RQ6 by empirically collecting data from appliances, devices, and cloud services. We use the collected data in our experiments which simulate the scheduling of devices.

1.3 Contributions

To address the research questions posed in the thesis, six contributions are made. First, we investigate the state of the art in data center best practices. By subjecting seven Dutch and Indian data centers to a questionnaire, interviews, and procedure reviews, we are able to quantify the practical adoption of twenty-three best practices for sustainability. Each best practice is assigned one of five categories, ranging from energy efficiency to storage and networking. This enables us to identify areas where improvements can be made. One area of improvement is liquid cooling, which is why we investigate the state of immersion cooling in greater detail. We summarize our first contribution as follows:

Contribution #1

C1 – An overview of 23 best practices and their adoption in 7 data centers as gathered from interviews with data center operators, and a review of the current state and future potential of immersion cooling.

Data center metrics play a critical role in understanding the effect of implementing the aforementioned best practices. The challenge lies in selecting the appropriate metric for the task at hand. To assist in this process, we identify one-hundred thirty-six metrics to measure the sustainability and efficiency of a data center. Each metric is assigned one of nine categories. We also investigate the relationships between metrics, as there are metrics that directly use or are derived from other metrics. And finally, we uncover numerous open issues, including the need for IoT-based data center monitoring, and the challenges faced by co-location data centers when attempting to monitor their facilities. This results in the following contribution:

Contribution #2

C2 – A taxonomy of 136 sustainability and efficiency metrics across 9 different data center domains, an analysis of the relationships between these metrics, and an evaluation of open issues and challenges related to data center metrics.

Evaluating the large variety of available data center metrics in real time and in great detail requires an IoT-based approach to data center monitoring. The potential volume of data that can be collected on a per server basis is vast, especially when considering the fact that there can be thousands of rack in a data center, each containing dozens of servers. In fact, we show that the amount of data that can be generated is so large that there is a significant negative effect on the data center's network bandwidth. Therefore, we propose to leverage the edge computing paradigm by taking advantage of the data center network hierarchy. Using IoT hubs deployed at the edge, the bandwidth usage is reduced and response times are improved. Furthermore, we perform a detailed analysis of four open-source IoT hubs out of an initial selection of twenty hubs. Using thirteen features and thirty-four criteria, we are able to identify strengths and weaknesses for the four systems, and extract a generic IoT hub architecture based on the commonalities between the four systems. Which leads us to the following contribution:

Contribution #3

C3 – An edge-based architecture taking advantage of the hierarchical network layout to deploy IoT hubs and distribute the data processing load, as well as a detailed analysis of 4 open-source IoT hubs based on 13 features and 34 criteria.

To address the difficulties related to co-location data center monitoring, we develop an approach that enables the monitoring of individual servers despite not having access to the operating system or the server chassis. Instead, all the required measurements are taken external to the servers, thus avoiding any privacy or security compromises with regards to monitoring. As part of our approach, we train and evaluate machine learning models using a data set collected from one-hundred sixty-four servers and consisting of over two billion data points covering thirteen different metrics. In total, we create over ten thousand different models, from individual models for each server, to universal models that are trained using the data of all servers. We summarize our fourth contribution:

Contribution #4

C4 – A privacy-preserving approach to monitoring servers in a co-location data center by only collecting data through sensors that are external to the chassis, and an evaluation of the approach using a data set consisting of 2.5 billion data points collected from 13 metrics collected on 164 servers.

The Internet of Things has an important role in enabling intelligent responses to activities occurring within a building. To demonstrate this, we deploy a multitude of IoT sensors and actuators in an office building, focusing specifically on a restaurant, a social corner with a kitchen, and two offices. The data collected from the sensors is used in the activity recognition process to determine the state of each room or area. An Artificial Intelligence (AI) planning solution is used to adjust the building environment by controlling the deployed actuators based on the recognized states. Our optimization of the building environment results in significant energy saving. We focus on the underlying architecture that supports the deployment of the IoT devices in order to highlight the similarities between IoT in office buildings and IoT in data centers. Therefore, we define the following contribution:

Contribution #5

C5 – A practical evaluation of an IoT deployment in the Bernoulliborg office building using two use cases: one deployed in two offices and a social corner, and one deployed in a restaurant. Furthermore, similarities and differences in the underlying Internet of Things architecture between buildings and data centers are highlighted.

Scheduling device loads in a smart grid environment is a non-trivial problem, especially when considering the varying price signals from the grid and neighboring prosumers, the local generation of renewable energy, as well as energy storage systems. On top of that, each device also has specific scheduling constraints that have to be satisfied. To solve this problem in a timely manner, we propose a parallel uniform-cost search algorithm to explore the complex state space to find the optimal solution that minimizes the energy costs. We design a micro-service architecture that enables real-time prediction of renewable energy generation and day ahead price signals. For the evaluation of our approach we use real-world device data as well as weather data for the prediction of renewable energy generation. And finally, we show that the inclusion of energy storage systems results in additional economic savings when compared to an office environment without energy storage. We summarize our final contribution as follows:

Contribution #6

C6 – A load scheduling approach for devices based on a parallel uniform-cost search algorithm that considers renewable energy, energy storage, neighboring prosumers, price signals, and scheduling constraints when searching for the device load schedule that maximizes economic savings, and a micro-service architecture for real-time data collection for the prediction of renewable energy and price signals.

The six contributions that are listed above have been reported also in eight peer reviewed publications. What follows next is a list of first authored works:

- P1 – B. Setz, K. Haghshenas, and M. Aiello, “Optimizing Energy Costs for Offices Connected to the Smart Grid: Ten Years Later”, manuscript submitted, 2022.
- P2 – B. Setz, S. Graef, D. Ivanova, A. Tiessen, and M. Aiello, “A Comparison of Open-Source Home Automation Systems” in *IEEE Access*, vol. 9, 2021.

- (P3) – B. Setz and M. Aiello, “Towards Real-Time Monitoring of Data Centers using Edge Computing” in *European Conference On Service-Oriented And Cloud Computing*, 2020.

- (P4) – B. Setz, A. Lazovik, and M. Aiello, “A Data-Driven Approach to Monitoring Co-location Data Centers” in *IEEE International Conference on Big Data Intelligence and Computing*, 2019.

- (P5) – V. Reddy, B. Setz, G. Rao, G. Gangadharan, and M. Aiello, “Metrics for Sustainable Data Center” in *IEEE Transactions on Sustainable Computing*, vol. 2, no. 03, 2017[1].

The following works were co-authored:

- (P6) – K. Haghshenas, B.Setz, Y. Bloch, and M. Aiello, “Enough Hot Air: The Role of Immersion Cooling”, submitted, 2022.

- (P7) – V. Reddy, B. Setz, G. Rao, G. Gangadharan, and M. Aiello, “Best Practices for Sustainable Data Centers” in *IT Professional*, vol. 20, no. 05, 2018.

- (P8) – I. Georgievski, T. Nguyen, F. Nizamic, B. Setz, A. Lazovik, and M. Aiello, “Planning meets activity recognition: Service coordination for intelligent buildings”, in *Pervasive and Mobile Computing*, vol. 38, no. 01, 2017.

Authored related works performed during the PhD period, though not included in this thesis:

- (P9) – M. Kalksma, B. Setz, A. Rizky Pratama, I. Georgievski, and M. Aiello, “Mining Sequential Patterns for Appliance Usage Prediction”, in *7th International Conference on Smart Cities and Green ICT Systems (SMARTGREENS)*, 2018.

[1]V. Reddy and B. Setz are both to be considered first authors, as they contributed equally to this manuscript.

- P10 – B. Setz, F. Nizamic, T. Nguyen, A. Lazovik, and M. Aiello, “Power Management of Personal Computers based on User Behaviour”, in *5th International Conference on Smart Cities and Green ICT Systems (SMARTGREENS)*, 2016.

In Figure 1.2, we present an overview of the thesis and visualize how each chapter and section is mapped to one or more research questions (RQ1-RQ6), contributions (C1-C6), and publications (P1-P8).

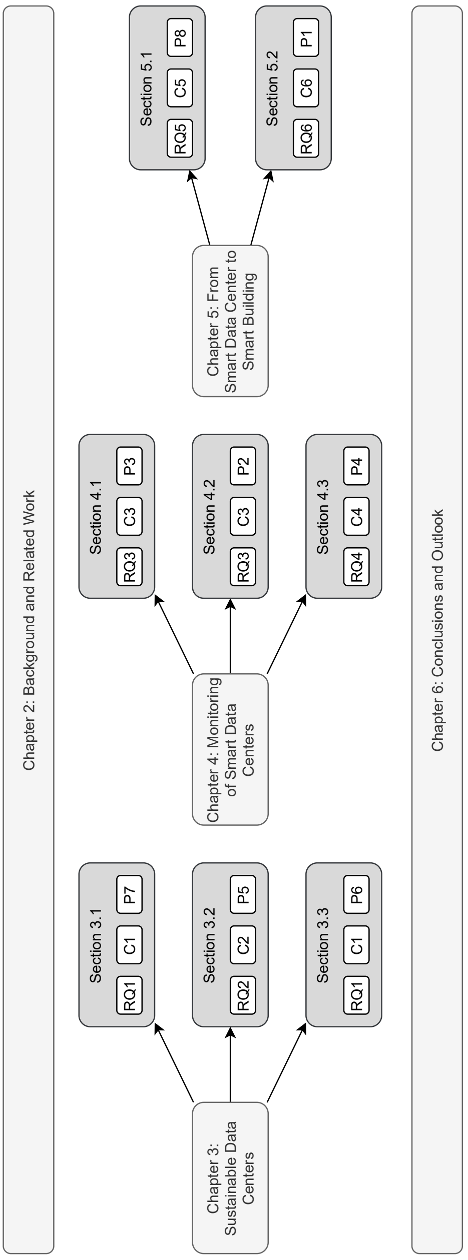
1.4 Structure

The remainder of this thesis is structured as follows:

Chapter 2: Background and Related Work In this chapter, we introduce the core concepts and terminology that are relevant to the work presented here. Furthermore, we discuss the related works in the three key areas.

Chapter 3: Sustainable Data Centers In this chapter, we focus on the efficiency and sustainability of data centers. First, we present an overview of best practices that promote sustainability in data centers. This is followed by a taxonomy and analysis of existing data center metrics. And finally, we also investigate an alternative cooling technique, namely immersion cooling. We address the research questions RQ1 and RQ2 through contributions C1 and C2, as published in our works in P5, P6, and P7.

Chapter 4: Monitoring of Smart Data Centers In this chapter, we investigate the role of the Internet of Things within the context of data center monitoring in order to transform regular data centers into smart data centers. We start by taking advantage of the data center rack layout to promote real-time and performant monitoring, and we identify the need for an IoT hub. Next, we perform a structured analysis of open-source IoT hubs, extract a generic IoT hub architecture, and discuss the strengths and weaknesses of existing hubs. And lastly, we demonstrate an approach for monitoring in



Chapter 2: Background and Related Work
Chapter 3: Sustainable Data Centers
Section 3.1
RQ1
C1
P7
Section 3.2
RQ2
C2
P5
Section 3.3
RQ1
C1
P6
Chapter 4: Monitoring of Smart Data Centers
Section 4.1
RQ3
C3
P3
Section 4.2
RQ3
C3
P2
Section 4.3
RQ4
C4
P4
Chapter 5: From Smart Data Center to Smart Building
Section 5.1
RQ5
C5
P8
Section 5.2
RQ6
C6
P1
Chapter 6: Conclusions and Outlook

Figure 1.2: Mapping of chapters and sections to research questions, contributions, and publications.

co-location data centers that enables privacy-preserved monitoring through the use of IoT. We address the research questions RQ3 and RQ4 through contributions C3 and C4, as published in our works in P2, P3, and P4.

Chapter 5: From Smart Data Center to Smart Building In this chapter, we transition from smart data centers to smart buildings. First, we discuss the case of the Bernoulliborg building, and present our IoT-oriented architecture to support intelligent decision making. And second, we propose a method for device load scheduling in a smart grid environment while considering renewable energy sources, energy storage, and varying price signals. We address the research questions RQ5 and RQ6 through contributions C5 and C6, as published in our works in P1 and P8.

Chapter 6: Conclusions and Outlook In this chapter, we reflect on our research questions and contributions to provide the conclusions to this work. We also present our outlook with regards to future research directions related to the topics discussed in this thesis.

Chapter 2

Background and Related Work

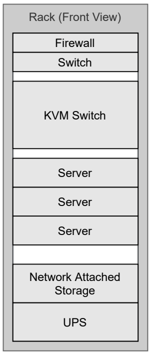
Data centers are complex buildings, consuming vasts amounts of energy while operating under a set of constraints. Some of these constraints are expressed in Service-level Agreements (SLAs), which define constrains regarding uptime guarantees, power supply, temperature ranges, networking bandwidth, and so on. An increasingly important constraint is the sustainability of the data center. As governments around the globe introduce new regulations to reduce the carbon footprint, data centers are increasingly affected. Monitoring the building in detail is a significant step towards optimizing the data center and meeting the new sustainability goals. To fulfill these goals, it is critical to measure in a standardized way the aspects that should be monitored, this is where metrics play an important role. Novel cooling techniques also promise to reduce the environmental impact. The application of IoT approaches within the data center domain enables detailed monitoring that was not possible before. As data centers become smarter, it becomes clear that there are many similarities to smart buildings.

In this chapter, we explore the works that are closely related to the three main aspects of this thesis: sustainability in data centers, IoT-driven monitoring of data centers, and the interconnection between smart data centers and smart buildings. But first, an introduction to data centers and the relevant terminology is given. Next, we look at the concept of sustainable data centers, with a focus on the available metrics to capture the different sustainability characteristics. As data center cooling is the major contributor to the energy footprint, we also explore works related to novel cooling approaches. After exploring sustainability and its related metrics, the focus shifts towards the monitoring of data centers using an IoT approach. In particular, we highlight works related to the monitoring of data centers, the significant role of IoT-hubs, as well as other monitoring-related aspects such as hardware modeling and privacy concerns. And finally, we shift from the data center domain to the smart building domain, while still retaining the focus on sustainability. We show that many parallels exist between smart data centers and smart buildings, particularly with regards to the role that the IoT-paradigm has in both domains.

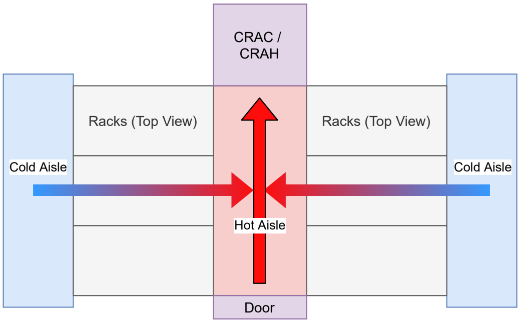
2.1 Data Centers

A data center is a building or cluster of buildings dedicated to housing primarily Information Technology (IT) equipment for data processing, data storage, and network communications, while providing specialized power conversion, reliable and high-quality power signals, a controlled environment in which temperature and humidity are optimal [Gen14]. Despite the fact that data centers house mainly IT equipment, not all space in a data center is dedicated to this purpose. The cooling systems and power distributed systems also require space within the data center buildings. The same holds true for on-site back-up power generation, which typically consists of energy storage and diesel motors. Additional space is also required by fire suppression systems, general storage, and office space for the employees.

The majority of the space is dedicated to the racks. A rack is a metal enclosure with standardized dimensions, typically with a maximum size of 42U, where 'U' stands for rack unit. The dimension of a rack unit is also standardized, as per EIA-310-D and DIN 41494. The IT equipment is mounted in a rack, and depending on the dimensions, takes up one or multiple rack units. This equipment includes computational hardware, networking equipment, storage servers, Power Distribution Unit (PDU), Uninterruptible Power Supply (UPS), as well as rack-mounted keyboards and displays. An example of a typical rack layout that one can find in a data center is displayed in Figure 2.1a. Data centers that rent out space in their server racks are known as co-location data centers. The co-location provides networking, power, and cooling, while the customer provides the hardware that is to be placed in the rented rack space. A challenge regarding co-location data centers is the lack of access to the hardware from the point of view of the data center operator.



(a) A typical data center rack layout.



(b) Hot aisle / cold aisle configuration as viewed from above.

Figure 2.1: Data Center racks and their layouts.

Racks are typically configured in multiple rows consisting of numerous individual racks. A rack has a hot side, and a cold side. The hot side is where the hot air is exhausted from the equipment, and the cold side is where the cold air enters the equipment through the use of forced air ventilation. The racks are positioned in such manner that the hot side of one row of racks faces the hot side of another row of racks. The same holds true for cold sides facing each other. This creates so-called hot aisles and cold aisles. In modern data centers, either the hot aisle or the cold aisle are completely isolated from the main hall. However, in practice open aisle air-based cooling is the dominant approach [NLL18]. Figure 2.1b shows an example of hot aisle isolation, where the hot aisle (in red) is enclosed separately from the cold aisle (in blue). This, in combination with closing any gaps in the server racks, ensures that the mixing of hot and cold air is minimized.

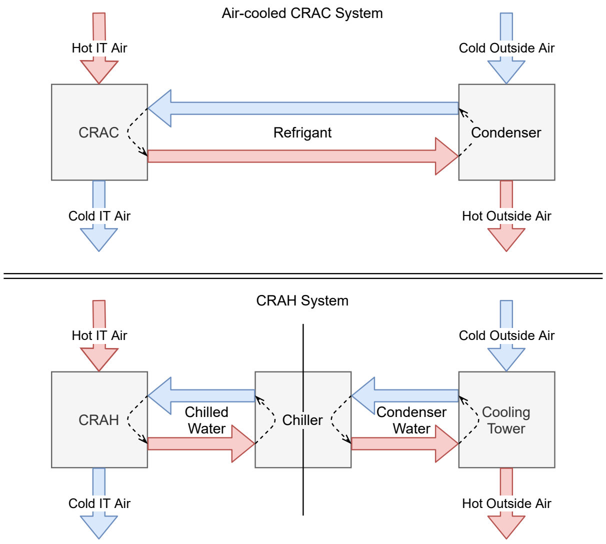


Figure 2.2: Comparison of Computer Room Air Handlers and Computer Room Air Conditioners.

It is typical for racks in a data center to stand on a raised floor. The raised floor allows cold air to enter from underneath the space in which the racks are located, through perforated floor tiles. Typically only the cold aisle has these perforated tiles. The fans inside the IT equipment draw the cold air inside the chassis and then force it over heatsinks covering the processors and other heat-producing hardware. After absorbing the heat of the server components, the warm air is pushed out into the hot aisle due to the air pressure difference. The warm air is then extracted by a Computer Room Air Conditioning (CRAC) unit or a Computer Room Air Handler (CRAH), in order to lower the air temperature to a level where it can be reused again in the cooling cycle [Gen14]. The main differences between CRAC and CRAH systems are illustrated in Figure 2.2. Whereas CRACs function much like residential air conditioners using refrigerants and compressors, CRAHs use a chilled water loop in combination with a chiller and a cooling tower. In general, CRACs are cheaper, while CRAHs are more efficient and reliable.

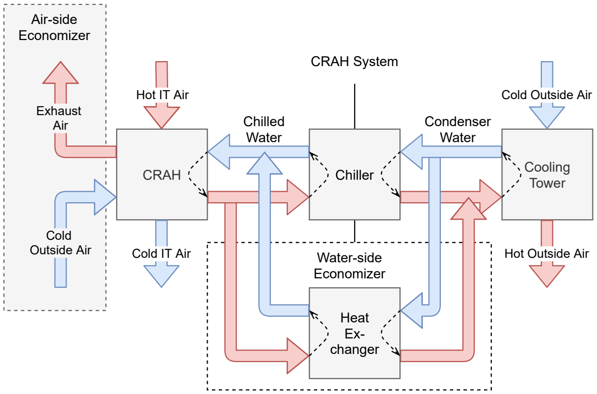


Figure 2.3: Cooling loop with air-side and water-side economizers.

To further increase the energy efficiency of a data center, it is typical to include so-called economizers in the cooling loop. The two economizers typically used in data centers are air-side economizers, and water-side economizers. Figure 2.3 illustrates a cooling loop with both types of economizers. The air-side economizer enables hot air to be exhausted to the outdoor environment, as well as using the cold outside air directly for cooling. This only works if the outside temperature is below a certain threshold. The economizer can be turned off if the outside air temperature is too high. A water-side economizer enables the cooling loop to by-pass the chiller and use a (passive) heat exchanger to cool the chilled water, assuming the condenser water is sufficient cold. On cold days the cooling tower will cool the condenser water to a point where the chiller is no longer required. The efficiency of both types of economizers is highly dependent on the climate in which the data center is located.

To guarantee the high uptime specified in the SLAs, it is important for data center to have a level of redundancy built into their systems. The Tier classification, as defined by the Uptime Institute, gives the data center industry a consistent mechanism for comparing typical facilities based on their up-time and facility performance, as well as pinpointing the available level of redundancy [TIPSB06]. The Tier classification distinguishes between the following categories of tiers:

- **Tier 1:** a facility composed of a single path for power and cooling distribution. It does not contain redundant components and provides at least 99.671% availability.
- **Tier II:** a facility composed of a single path for power and cooling distribution, it contains redundant components and provides at least 99.741% availability.
- **Tier III:** a facility composed of multiple active power and cooling distribution paths, and redundant components with only one active path. It is concurrently maintainable and provides at least 99.982% availability.
- **Tier IV:** facility composes of all components of Tier III and it is fault tolerant, it provides at least 99.995% availability.

2.2 Sustainable Data Centers

In the last years, significant research efforts and technological developments have been devoted to data centers, targeting energy efficiency and eco-friendliness. Meeting sustainability goals has become an important objective when managing data centers for which there exists a multitude of strategies [MSMR21]. To capture the effects of these strategies and the data center management, the first step is to decide which dimensions are relevant, followed by defining the metrics, and finally populating them [DJK+09]. There is a continuous development of novel data center metrics, with an ever-increasing focus on sustainability. For example, the European Union financed an 8-project cluster of over 50 partners to develop new environmental efficiency metrics and methodologies. The projects were All4Green, CoolEmAll, GreenDataNet, RenewIT, GENiC, GEYSER, Dolfin, and DC4Cities. The cluster has published several works in which they analyze existing metrics and also propose novel metrics to assess the performance of data centers [PVC14; SSO+14]. Capozzoli et al. reviewed thermal, power and energy consumption metrics [CCPS15]. Aravanis et al. introduced new metrics for the assessment of flexibility and sustainability of data centers [AVS+15]. In [SSJ+13], Siso et al. propose and evaluate several metrics for the CoolEmAll project.

Energy efficiency metrics are the most prevalent type of metric in the context of data centers. The Green Grid consortium proposed the Power Usage Effectiveness (PUE) [BRPC07], which currently is the dominating metric used and often made public by data centers. However, PUE has some shortcomings, as decreasing the overall power usage may actually result in a higher PUE [Col11]. The Green Grid consortium proposed the Partial Power Usage Effectiveness (pPUE) metric, based on PUE, and the Data Center Infrastructure Efficiency (DCiE) metric [AAFP12] which measures the efficiency of data centers by relating power consumption to IT equipment. PUE and DCiE help data center operators determine the efficiency of the data center, where pPUE measures the energy efficiency of a zone in a data center. The consortium also proposed metrics such as Carbon Usage Effectiveness (CUE) [DAMPT10], Water Usage Effectiveness (WUE) [PABP11], and Electronics Disposal Efficiency (EDE) [BBC+12] to measure the CO_2 footprint, the water consumption per year, and the disposal efficiency of the data centers, respectively. Munteanu et al. proposed two different metrics based on energy consumption and Central Processing Unit (CPU) usage for calculating useful work done by in data centers. [MDBB13]. They proposed EnergeTIC Usage Effectiveness (EUE) considering total Internet Data Centers (IDC) power, IT power and load levels. They also proposed EUE(CPU), EUE(kWh) and EUE(kWh)-IT. Schaeppi et al. explored energy related metrics for IT equipment, data storage and network equipment [SBS+12]. Fiandrino et al. proposed new metrics for computing the energy efficiency of the data center communication systems, processes and protocols which includes the network energy efficiency, network power usage effectiveness and network performance related metrics [FKBZ15]. Daim et al. explored measurable components of a data center and proposed a new metric that fills the gap in measuring the data center equipment power and uses a credit-based system for data centers not meeting the standard [DJK+09].

While the majority of metrics focuses on energy efficiency, there are also other important metrics and standards that focus on other aspects of sustainability. The American Society of Heating Refrigeration and Air Conditioning Engineers (ASHRAE) provides a common set of environmental guidelines for data processing environments, equipment and guidance on server metrics which enables data center operators to optimize temperature and humidity conditions [ASH11; Ash]. Metrics to monitor and control the air flow in a data center are discussed in [CSLC14; TS10]. Chen et al. identified and presented usage-centric green performance indicators at various data center levels [CPH+11]. Wang et al. presented a set of performance metrics for a green data center [WK13]. Their focus is on available benchmarks and on how performance metrics can be used to measure the greenness. Wiboonrat discussed the effect of a data center outage and provided a solution to minimize the data center downtime [Wib08]. The author proposed improvements on the data center topologies to reduce the failure rate.

A detailed analysis of metrics for sustainable data centers can be found in Chapter 3. Measuring, monitoring, and evaluating metrics is critical for determining the present level of efficiency and sustainability, though a more active approach is required to improve the sustainability of data centers. Historical trends show that the number of computations per kilowatt-hour (kWh) has doubled every 1.57 years [KBSW10]. Therefore, upgrading aging IT equipment typically results in an increase of the energy efficiency. Another approach is to optimize the scheduling of workloads [HTGM20; SDCB17]. The optimizations goal can be set to reduce energy consumption, heat output, or the carbon footprint. Scheduling in data centers typically requires power models for individual servers, which in turn requires data regarding a systems power consumption under different load profiles.

One of the most diverse data sets on server power consumption is provided by the SPECpower_ssj2008 benchmark [GKL08]. The benchmark measures performance and power of servers using gradually increasing load levels. The benchmark focuses on collecting measurements, but does not perform predictive modeling. In [BJ12], the authors propose a complete system model for modeling the power consumption of six subsystems: CPU, memory, chipset, Input / Output (I/O), disk, and Graphics Processing Unit (GPU). They have observed an average error of less than 9% per subsystem when evaluating their model. These results are obtained using highly detailed, low level hardware information. The authors of [WDDB12] propose a more abstract approach to modeling power consumption, aiming for data centers in particular. In their model, they consider servers which are running virtual machines, and evaluate how the number of virtual machines influences the server's power consumption. They also include the idle power consumption and the additional load introduced by the hypervisor. Linear regression is applied to obtain the values of the model's parameters. A different approach for estimating power consumption is proposed by Dargie et al. [Dar15]. In their work, the authors model the CPU usage and the power consumption as random variables and exploit the monotonicity property to describe the relationship between these variables. The authors report mean errors between 2% to 5.2%, depending on the data set. Similar to the previous work, only power consumption and CPU usage are compared. In [OKC+10], the authors propose a method to estimate the power consumption of individual CPU cores based on the measured CPU core temperature. They also develop a technique to optimize the throughput on CPUs that have thermal constraints. Their optimization method improved throughput by 4%, when compared to existing temperature-based methods. It is important to note that almost all IT power consumption is converted into heat, which has to be dissipated.

When it comes to sustainable data centers cooling is also an important aspect to consider, and several reviews on cooling solutions have appeared.

One of the earliest reviews considers the thermal aspects of air cooling in data centers [Pat10]. In a review by Ebrahimi et al., different data center cooling technologies are analyzed, and particular attention is devoted to the opportunity of recovering the heat [EJF14]. For example, district heating can be fed from recovered hot air from a data center. Li et al. offer a detailed thermal analysis of cooling solutions, including several strategies based on cold plates, waste heat recovery, and heat pipes [LK15]. Though air cooling is the prevalent cooling method, a more drastic approach to increase the energy efficiency is to migrate away from air cooling and switch to liquid cooling, or even immersion cooling.

Immersion cooling is an approach that uses liquid instead of air to absorb heat from computing hardware. In immersion cooling, components are fully immersed into a dielectric fluid. A dielectric fluid conducts heat while not conducting electricity at all; instead, it acts as an insulator. As this fluid has a much higher heat capacity than air, the heat of all the components is efficiently removed by the liquid [EFV+14; SBS+19]. The use of immersion cooling eliminates the need for components to move air, such as fans. The most commonly used liquids in immersion cooling are white mineral oil, engineered dielectric fluids, and other oils [JMG+21; SBS+19]. Kuncoro et al. review immersion cooling solutions and compare the performance of different types of cooling liquids [KPB+19]. In general, a distinction is made between two types of liquid immersion cooling: single-phase cooling and two-phase cooling, as shown in Figure 2.4.

In single-phase immersion cooling, the fluid remains in liquid form and there is no phase change. Heat emitting components are cooled by the fluid flowing over them, and the heated fluid is transported away. The circulation of the fluid is driven by a pump or by natural convection. In natural convection-driven systems, the heated fluid floats to the top of the tank because it has a higher volume than colder fluid. It then flows, due to more fluid rising to the top, to the side of the tank where it is cooled by a heat exchanger connected to an external loop. As with CRAC and CRAH systems, a cooling tower can be used for heat rejection in the external loop. The density of the cooled liquid increases again and it sinks back to the

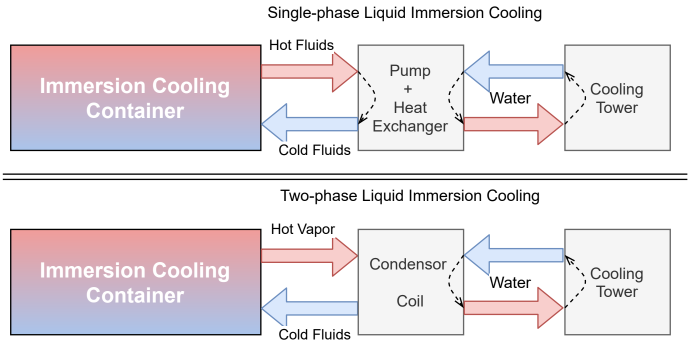

Single-phase Liquid Immersion Cooling
Immersion Cooling Container
Hot Fluids
Cold Fluids
Pump + Heat Exchanger
Water
Cooling Tower
Two-phase Liquid Immersion Cooling
Immersion Cooling Container
Hot Vapor
Cold Fluids
Condensor Coil
Water
Cooling Tower

Figure 2.4: Liquid-based immersion cooling techniques for data centers

bottom of the tank by gravity. In a pump-driven system, the convection is driven by a pump. The pump forces the liquid through an inlet inside the tank and out through an outlet on the opposite side. The liquid is then cooled by flowing through a coolant-to-water heat exchanger.

In two-phase immersion cooling, the coolant changes phase whenever it gets in contact with a heat-producing component. In order to avoid damage to the components, the boiling point of the coolant has to be lower than the critical temperature of the components. While evaporating on a hot component, the gas will float to the top of the tank and will make room for colder liquid coolant. A condenser is located inside the tank above the liquid. Cooling water flows through the condenser to exchange the heat to the external loop. The coolant condenses there and will fall back into the immersion tank where it can absorb heat again [KWF+20].

Some authors were already proposing immersion cooling for data center equipment [Tum10b] in 2010. The work of Kheirabadi et al. focuses on the thermal aspects of server cooling [KG16]. They classify the cooling solutions as air-based or liquid-based. Air-based solutions include CRAC, CRAH, Rear Door Heat Exchanger (RDHx), and Side Car Heat Exchanger (SCHx). Whereas liquid-based solutions are split into two subcategories; direct and indirect. The indirect liquid cooling solutions include: single-phase, two-phase, and heat pipes. And direct solutions include: pool boiling, spray cooling, and jet impingement. The results of the review and the technological comparison show a higher efficiency for liquid cooling-based solutions while maintaining that air-based solutions are a valid option especially if high efficiency is not a top requirement. The chemical and physical effects of immersion cooling on IT equipment are investigated in [SESA16]. The results are particularly relevant to understand the maintenance requirements and life expectancy of the equipment. In addition, several advantages of immersion cooling are identified, such as a decreased chance of overheating and temperature swings, elimination of fan failures, decrease in dust and moisture-related failures, and reduced corrosion. Furthermore, Watson et. al compare two prototypical systems, one based on air cooling and one on immersion cooling, and provide an evaluation of the two based on simulations especially focusing on scalability aspects [WV17]. The results indicate a preference for immersion cooling.

There are many facets to consider where sustainable data centers are concerned. In this section, we have focused on works related to two of those aspects, the importance of metrics for measuring and monitoring purposes, and the role of novel cooling methods such as immersion cooling. Both aspects are explored in detail in Chapter 3.

2.3 Monitoring of Smart Data Centers

Real-time monitoring is a technique that has become more popular in several domains with the emergence of the Internet of Things. In smart grids, for example, real-time monitoring promises to assist in the prevention of severe safety accidents by automatically identifying threats, and it can also contribute to the optimization and sustainability of the production and transport of energy. The authors of [HLW+18] identified that real-time monitoring of smart grids would cause an increase in data that would be too large to handle using the traditional cloud computing paradigm. In their solution they introduce edge computing as a key component of their real time monitoring solution, reducing the network load by more than 50%. The authors of [AP14] also identify the capability of the smart grid to generate substantial amounts of data. Hassan et al. [HGA+18] describe the role that edge computing has in the Internet of Things. They propose a layered model in which millions of IoT devices connected to thousands of edge gateways, which in turn connect to hundreds of cloud data centers. The authors also recognize the need for data abstraction, which uses edge gateways to reduce the volume of the raw data before sending it to the data center. However, deciding which data should be reduced, and by how much, is an open problem according to the authors. It is also important to consider the security of edge gateways, as the distributed nature of edge computing increases the possible attack vectors [SRCL+22]. The application of real-time monitoring in the domain of data centers enables in-detail monitoring of the entire data center at any level, but has similar drawbacks when it comes to the quantity of data that is generated. Utilizing edge computing for data center monitoring promises to address some of these challenges.

When it comes to monitoring data centers, we make some important distinctions. First of all, the difference between external and internal sensing. External sensing is non-intrusive and is typically used to monitor the humidity and temperature per rack, or of the entire room. Another example of external sensing is measuring the power consumption of servers or racks using PDUs. Internal sensing is intrusive to the IT equipment and typically requires either access to the chassis or the operating systems. The work of Hubbell et al. illustrates the difference between external and internal sensing [HMA+15]. They collect 5,000 environmental data points (external sensing) and 3,500 server data points (internal sensing). Their environmental data points include temperature, humidity, air pressure, power consumption, voltages, and amperages. Server data points consist of software versions, CPU load, memory allocations, disk utilization, network and link utilization, storage health and the state of the job scheduler. It is also important to note the difference between hardware sensors and software-defined / virtual sensors. Hardware sensors are standalone hardware-based devices that can be placed in the data center, for example on top of a rack, or in front of IT equipment.

Software-defined sensors and virtual sensors are sensors that execute as software on the IT equipment. For example, monitoring the CPU utilization via the operating system requires software.

The authors of [LH17] propose a real-time monitoring infrastructure using low-power wireless sensors. They employ low-power wireless sensors to monitor the following parameters within a data center: temperature, humidity, airflow, water, security, vibration, differential air pressure, and fire systems. They state that such a system can be rapidly deployed and would enable real-time predictive modeling. In [Wib14], the authors also propose a wireless system architecture for monitoring a data center. However, in their work not only sensing is considered, but also actuation. They propose to control the PDUs and the cooling systems using wirelessly networked sensors and actuators. Medina-Santiago et al. developed a method for real-time monitoring of data centers using an IoT approach [MAG+20]. The data in question is environmental data, such as the temperature and the humidity level. These values are collected, by means of external sensing, every 10 seconds. A total of 1.4 million data points were collected. Their IoT platform is based on a simple web service which accepts data collected by the custom-built sensors. The monitoring takes place on the level of individual racks. When extrapolating these findings to monitoring on the level of individual servers, the amount of collected data increases by an order of magnitude. In the work of [LH17], an approach is proposed for monitoring a data center in real time using low-power wireless hardware sensors. The collected data includes temperature, humidity, airflow, air pressure, water pressure, security status, vibrations, and the state of the fire systems. The need for collecting data from servers for monitoring purposes is also recognized. The authors envision that some type of IoT platform is required for the collection, processing, storage, and management of the data. The envisioned platform is not designed or implemented by the authors.

We can conclude that large scale monitoring of data centers requires an IoT platform, also known as an IoT hub, for interfacing with numerous different types of sensors and handling all of their data. It is also clear that edge computing can play an important role and facilitating the data processing. Obvious parallels can be drawn between IoT hubs that would be required in data centers and those that can be found in smart homes and smart offices. Fortunately, there is a vast body of literature available on IoT hubs, specifically on academic and commercial platforms.

Jerabandi, et al. provide an overview of existing IoT hubs [JK17] . Their focus is on academic works that address very specific problems. Based on the review of 14 systems, the authors identify numerous design issues related to IoT hubs. The issues are: serviceability, scalability, programmability, auto-configuration, centralized vs. decentralized architectures, heterogeneity, transparency, security, open standards, robustness, and energy efficient communication. A generic IoT framework for home automation systems is briefly described. Taiwo, et al. propose a taxonomy of IoT hubs, with the main focus on technology, trends, and challenges [TGE20]. The systems are primarily academic; the results of closed-source studies with limited to no adoption. The taxonomy highlights five components of IoT hubs: (1) application area, (2) automation layers, (3) protocols, (4) platforms, and (5) sensors. The authors also identify trends and challenges in IoT hubs. The current research trends focus on: energy efficiency and energy reduction, privacy and security, and innovative technologies. End-to-end encryption using the Public Key Infrastructure can be used to address some of the security concerns by ensuring that intermediary receivers cannot decipher the messages [AGC+19]. Furthermore, the following challenges are identified in [TGE20]: authentication and authorization, privacy, high cost and incompatibility, and energy management. Derhamy, et al. focus on commercial IoT frameworks [DEDP15]. In their work they analyze 14 different frameworks and platforms. The emphasis is on commercially available IoT frameworks for automation, including: IPSO Alliance, IoTivity, AllJoyn, and Thread. The authors recognize that for a platform or framework to succeed they must: (1) securely expose Application Programming Interfaces (APIs) for third parties, (2) provide protocol interoperability with third party API, as well as protocol extensibility, (3) enable constrained devices to participate, and (4) enable management and governance of heterogeneous networks of device and applications. The authors of [SZZ14] highlight that openness is an important prerequisite for experts to contribute to IoT hubs.

Particularly relevant is the work of Risteska Stojkoska et al. which features a review of the state-of-the-art in Internet of Things applications in smart environments [RT17]. They identified numerous challenges related to this topic, which also apply to the monitoring of data centers. First of all, the use of edge computing shows great promise despite a lack of adoption in existing platforms and frameworks. This is inline with our findings regarding real-time monitoring of data centers, and the need for edge computing. The next challenge relates to big data, specifically regarding the overall performance when handling big data. The scale at which data can be collected in data center indeed enters the domain of big data. Finally, security and privacy is another challenge which is highlighted, which is primarily due to the way data is transmitted wirelessly in smart homes. Where data centers are concerned, security and privacy are always critical, especially considering the vast quantity of potentially sensitive data.

As data is generated and collected in increasingly larger quantities, more opportunities will arise for businesses and governments to combine and process this data to gain new and interesting insights. However, this frequently endangers the privacy of individuals [JGK16]. A significant effort is made in the research on privacy preservation within big data.

A classical approach to preserving privacy in data analytics is applying de-identification. The k-anonymity property of a data set ensures the individual's privacy by preventing identification through linking attacks [MGKV06]. A data set satisfying k-anonymity ensures that every entry is indistinguishable from at least k - 1 other entries, for every combination of attributes (e.g. place of birth, phone number, age) that could uniquely identify an individual when combined with external sources. An extension of the k-anonymity property is the l-diversity model as proposed by [MGKV06], which addresses a number of weaknesses. It utilizes generalization and suppression to further ensure an individual's privacy. For example, the exact age of an individual is not stored, but instead a range is specified. This inevitably leads to a loss of information. A more recent extension of l-diversity, addressing a number of vulnerabilities, is t-closeness [LLV07]. This approach takes into account the distribution of the attributes that can potentially identify an individual. More recent developments include privacy preservation in MapReduce frameworks [DDGS16], anonymization of real-time data streams [KG15], and differential privacy [DR+14]. In many cases this is implemented by distorting the original data set, or by adding additional noise to guarantee an individual's privacy. In the case of real-time data streams, the computational complexity of the anonymization algorithm has to allow for high throughput.

To monitor all dimensions of a data center requires the appropriate monitoring infrastructure. The potential amount of data that can be collected is enormous, and requires interfacing with many different types of power and cooling systems, as well as IT equipment. In Chapter 4, we explore how edge computing can play an important role when monitoring data centers at a large scale. We also analyze multiple open-source IoT hubs. And finally we demonstrate how a co-location data center can be monitored non-intrusively while preserving privacy and security.

2.4 From Smart Data Center to Smart Building

A smart building is one which integrates and accounts for intelligence, sensing, control, and optimization, typically through use of IoT, with adaptability as a core concept in order to optimize for: energy, overall efficiency, comfort, and occupant satisfaction [BMB14]. There exist many parallels between smart buildings and smart data centers. Both environments require an interconnection of sensors and actuators to optimize for a given goal. Whereas smart buildings integrate their IoT hubs with a Building Management System (BMS), a data center on the other hand depends on a Data Center Management (DCM) system for integration. Optimizing for energy efficiency or carbon footprint is something that is important in both domains.

When optimizing for the energy efficiency, the potential savings are significant, and there exist a multitude of different approaches that optimize different aspects of a building. For example, intelligent lighting control has the potential of energy savings up to 58% [NA13]. Yu et al. provide an overview of energy optimization approaches using deep learning [YQZ+21]. The approaches are grouped into three categories: single building, multi building and micro grid. The overall energy savings for most of these works are in the 5-30% range, with some outliers going up as much as to 70%. The authors of [PC21] review the state of the art in AI for smart buildings. They identify five research domains: energy, comfort, safety, design, and maintenance. Their conclusion is that AI will continue to play an important role in the optimization of smart buildings.

The use of AI planning in pervasive computing has been a subject of research in many studies, but only few use planning in actual pervasive computing environments [GA16]. The studies usually describe pervasive computing environments intuitively through scenarios and examples where devices provide some operations or services and requests are issued by people or software components. In some cases, a spatial organization of environments is portrayed too. The basic correspondence that these studies establish between pervasive computing environments and AI planning is by relating services to actions and requests to goals. All other environment-specific knowledge, such as the spatial organization, is integrated into the planning domain model. The use of AI planning for building optimization is one of the open challenges identified by Paola et al. [DPOLR+14]. Kaldeli et al. [KWLA13] take a more transparent approach and provide a formalized transition between their home domain and planning domain model.

In the future, smart buildings and smart data centers are expected to integrate even more tightly with the power grid using modern technology and artificial intelligence, creating a smart grid. The smart grid is an intelligent grid, it is able to store, communicate, and make decisions in a cooperative and responsive manner [TA16]. To improve sustainability, economics, and security, the smart grid aims to integrate the behaviors of consumers, producers, and prosumers. Prosumers do not only consume energy, but also produces energy. In other words, the smart grid allows the different actors on the grid to work together based on signals that are transmitted alongside the energy delivery. The building's BMS can interpret these signals and control the building as necessary to optimize for a certain goal. The authors of [Gha19] highlight the important role of IoT within the context of smart grids, specifically with respect to monitoring energy generation and consumption at the source, integration with smart meters, and coordination of energy storage. The authors also identify challenges, especially with regards to communication between IoT, BMS, and smart grids, as well as interpreting smart grid signals. An example of a typical smart grid signal is the price signal. Throughout the day, the electricity prices fluctuate on the energy markets as demand and supply changes. A smart building is able to take advantage of these fluctuations and schedule appliances, devices, and other loads to perform work at a time during the day which minimizes energy costs while still adhering to constraints, such as comfort.

A common approach to the scheduling of electrical loads is using Mixed-Integer Linear Programming (MILP) models. Paterakis et al. present a method for optimal scheduling of household appliances [PEBC15]. The scheduling is performed under hourly pricing and peak power limiting demand response strategies. The authors utilize a MILP model for optimal scheduling, distinguishing between thermostatically and non-thermostatically controllable devices. The results show that the load factor can be improved significantly, while the economic costs slightly increase. Duman et al. also demonstrate the use of a MILP model for scheduling the operation of shiftable loads [DEGG21]. The distinction is made between time-shiftable loads, power-shiftable loads, and thermostatically controlled loads. Their simulation shows that daily costs can be reduced by 53.2%. A similar method has been proposed by Nan et al. for optimal demand response scheduling in residential communities [NZL18]. A distinction is made between interruptible loads, shiftable loads, and distributed generation. The authors define a MILP problem to be solved in order to find a solution that reduces peak load and peak-valley differences and, in turn, the overall cost of power. The end user costs are reduced by up to 1.52%.

While Mixed-Integer Linear Programming (MILP) appears to be the predominant approach, alternatives have been proposed. Amer et al. propose a home energy management system centered around a multi-objective optimization problem [ASGM21]. A distinction is made between shiftable loads, non-shiftable loads, and active loads. Energy storage and Photo-voltaic (PV) energy resources are also considered. The objective is to balance the benefits for the end-user and the Distribution System Operator (DSO). The results show a reduction in energy cost of 31%, and a reduction in demand peak by 18%. Lu et al. present a different approach for demand response management using reinforcement learning and artificial neural networks [LHY19]. While a distinction between different types of loads is made, energy storage and renewable generation are not considered. The obtained cost savings are between 7.3% and 72.3%. The authors demonstrate that their approach outperforms MILP as the number of iterations increases. The method proposed by Li et al. utilizes deep-reinforcement learning for the optimal scheduling of home appliances [LWH20]. The appliances are split into three categories: deferrable devices, regulatable devices, and critical devices. The approach reduces the overall electricity costs by 31.6%. An approach that is more closely related to the work that is presented in this thesis, is the work of Fioretto et al. [FYP17]. They define a constraint optimization problem, which is solved by a distributed algorithm that divides the problem into individual sub-problems to overcome the complexity of the problem. The savings compared to the baseline use case are over 50%. Finally, Georgievski et al. demonstrated that the optimal scheduling of the operation of devices based on user-defined policies can result into significant savings [GDP+12]. From the economic perspective, the obtained savings are of about 35%. Furthermore, energy savings of 10% are also observed. The occupant satisfaction study shows that comfort and satisfaction is preserved while the system operates the devices.

Whether it is a smart home, a smart office, or a smart data center, the potential for energy savings through optimization is clear, both in theory and in practice. IoT enables many aspects of the building to be monitored and controlled, either through an IoT Hub, a BMS, or a DCM system. In Chapter 5, we focus specifically on smart offices. Two practical cases are evaluated, a kitchen/social corner area, and a restaurant. We also research the role that energy storage plays when scheduling device loads.

Chapter 3

Sustainable Data Centers

Climate change has been an important driver of a continued push towards increasing sustainability all across society. The green house effect is one of the main contributors to climate change, and of the greenhouse gases, carbon dioxide emissions are the main culprit. The burning of coal, oil, and gas for the purpose of electricity generation is causing a sharp rise in green house gas emissions. As data centers are responsible for a significant portion of the global energy consumption [Jon14], the concept of a sustainable data center becomes increasingly important. The focus of this chapter is on sustainable data centers. We stress the necessity of applying best practices, selecting the appropriate metrics to measure sustainability, and exploring novel cooling solutions.

In Section 3.1, we investigate the adoption of best practices for sustainability in data centers. Additionally, we identify a number of recurring issues and the appropriate best practices to apply to counteract them. Next, in Section 3.2, we review the state of the art in data center metrics across numerous dimensions, with an emphasis on metrics that help to promote sustainable decision making. Furthermore, open issues and challenges are identified for each of the dimensions. And finally, Section 3.3 discusses the potential that immersion cooling has compared to traditional cooling systems. We compare air and liquid based approaches in terms of efficiency, cost, and maintenance. The goal of the chapter is to give a deeper understanding of what practices steer data centers towards sustainability, how the effects of these changes can be measured and evaluated, and how novel cooling methods can play a role in the near future. We summarize our findings in Section 3.4.

3.1 Best Practices for Sustainable Data Centers

An important first step towards sustainable data centers is to identify and understand the best practices that can be applied to improve sustainability, and determine to what extent these best practices are followed in reality. This section presents the outcome of our study on such practices. We analyze the current state of affairs in seven data centers in India and the Netherlands using a multiple case study approach [Yin09]. We compare the practices followed in these data centers against the relevant standards on sustainable data centers, identify design issues and operation inefficiencies, and provide recommendations for improvements at various operational levels of the data centers.

3.1.1 Methodology

The features and infrastructure details of the seven data centers partaking in our study are presented in Table 3.1. For each data centers we present details on the number of racks, the size in square meters, the power capacity, the number of security zones, the tier, and the Power Usage Effectiveness (PUE). Among the seven selected data centers, two are co-location data centers and provide various services including public cloud services. Three data centers are privately owned by companies and are used in the financial services sector running banking, financial, and insurance applications. The remaining two data centers are those of academic institutions. The selection was made to cover a wide range of types of data centers.

A uniform and standard data-collection methodology was adopted in each case, which included a standard questionnaire, a review of procedures, benchmarks, and interviews with key personnel. The standard questionnaire was based on the assessment of the following dimensions: energy efficiency, cooling, thermal and air management, greenness, and network and storage. For each dimension, issues and best practices were identified directly in the data center for the purpose of the study. Based on interview transcripts, we developed an ad hoc case study report, which was then distributed and discussed with the interviewees and other staff to gain insights and tailored feedback for the correct understanding of the status of the data center.

3.1.2 Adoption of Best Practices

The following dimensions for best practices are at the core of the present study: energy efficiency, cooling, air and thermal management, greenness, and storage and network. Each practice is categorized accordingly. Based on the interviews that have taken place with the personnel from the seven data centers, Table 3.2 has been populated. Each practice is assigned to a dimension, and each data center is subjected to the practice. In case a data center follows a given best practice, a check mark (✓) is used to indicate this. We will now discuss the practices for each of our dimensions.

Table 3.1: The seven selected data centers and their configuration.

	Racks	**Size**	**Power**	**Zones**	**Tier**	**PUE**
DC1	5 000	21 300 sqm	30 MW	8	4	1.6
DC2	1 400	3 700 sqm	10 MW	6	4	1.7
DC3	800	1 850 sqm	6 MW	6	3	1.6
DC4	3 000	11 600 sqm	20 MW	6	3	1.6
DC5	1 000	2 800 sqm	10 MW	6	3	1.7
DC6	160	325 sqm	450 kW	Redacted	2	1.25
DC7	100	300 sqm	300 kW	Redacted	2	1.25

Table 3.2: The adoption of best practices for sustainability per data center.

	DC1	DC2	DC3	DC4	DC5	DC6	DC7
Energy Efficiency Practices							
Automation tools	✓	✓	✓	✓	✓		
Virtualization and consolidation	✓	✓	✓	✓	✓	✓	✓
Dynamic voltage and frequency scaling	✓	✓	✓	✓	✓	✓	✓
Handling comatose/zombie servers	✓	✓	✓	✓	✓	✓	✓
Raised thermostat set point	✓	✓		✓		✓	✓
Controlled lighting with sensors							
On-site power plant							
Data center infrastructure management tools	✓	✓		✓		✓	✓
Cooling, Thermal, and Air Management Practices							
Custom central air handler	✓		✓	✓		✓	✓
Liquid cooling	✓						
Sensors for chiller plant	✓	✓	✓	✓	✓	✓	✓
Hot aisle and cold aisle containment	✓	✓	✓	✓	✓	✓	✓
LOOP design for chillers	✓		✓	✓	✓	✓	✓
Adjustable speed drive chillers	✓	✓	✓				
Green Practices							
Economizers (free cooling)	✓			✓		✓	✓
Reclaimed water for cooling							
Cooling water re-circulation						✓	✓
Using renewable resources						✓	✓
Storage and network Practices							
Storage tiering	✓	✓	✓	✓	✓	✓	✓
Automatic waste storage management	✓	✓	✓	✓	✓	✓	✓
Centralized control and storage optimization	✓	✓	✓	✓	✓	✓	✓
Software-defined networking	✓	✓		✓		✓	✓
Storage pooling and geo-replication	✓	✓	✓	✓	✓	✓	✓

3.1.2.1 Energy Efficiency Practices

Efficiency is defined as the ratio of the useful work done by a system to the total energy delivered to it. For data centers, the energy efficiency translates to the useful work performed by different subsystems. For example, the energy efficiency of a server is the ratio of energy delivered to the server to the energy used to complete the computational workload. Inefficiencies may arise from power conversion losses, capacitor leakage, or even the cooling fans. What follows next are some of the key practices to improve the energy efficiency in a data center.

Automation Tools The use of data center automation tools helps to automate tasks such as provisioning, configuration, patching, release management, and compliance. Most of the data centers we studied rely on automation tools that enable real-time optimization, reduce error rates, and improve the performance of the applications. As can be seen in Table 3.2, larger data centers rely more heavily on automation tools.

Virtualization and Consolidation Virtualization enables abstracting physical servers in a data center facility along with storage, networking, and other infrastructure devices and equipment. Consolidation combines workloads from different machines into a smaller number of systems when servers are under-utilized and therefore consume more energy as a whole [FNCDR11]. All the data centers in our study are virtualized and use different virtual machine consolidation and placement techniques to reduce power consumption and improve server utilization.

Dynamic Voltage and Frequency Scaling The power management technique known as Dynamic Voltage and Frequency Scaling (DVFS) reduces the power consumption of a processor on the fly by adjusting clock frequency according to current workload, which indirectly leads to a reduction in the supply voltage [MSA+03]. Load scheduling techniques that take advantage of DVFS or directly control the frequencies are able to achieve energy savings by leveraging the changes in supply voltage. All of the interviewed data centers use this best practice.

Handling Comatose / Zombie Servers Comatose or zombie servers are those that run applications that are no longer required or are unused, yet remain plugged in while operating continuously. Data center operators have to audit and identify comatose servers as well as duplicate applications. A report from the Anthesis Consulting Group states that the percentage of comatose servers in data centers is around 25% [KT17], whereas Ngoko et al. [NC17] report that up to 30% of servers are comatose. In our interviews we observed that decommissioning unused servers may result in energy savings of up to 50%. Each of the seven data centers has procedures in place to handle these types of servers.

Controlled Lighting with Sensors Installing a lighting control system in conjunction with more efficient fixtures and occupancy sensors can help reduce energy usage. Only three of the data centers in our study were using resource-friendly timers that dim or turn off lighting when people are not present.

On-Site Power Plant The critical need for clean and economical sources of energy is transforming data centers that are primarily energy consumers to energy producers. On-site renewable power generation is an economical and eco-friendly solution for regions with high electricity prices, and for campus-like facilities that can re-utilize excess heating and cooling [Ahu12; TLCS13]. In these cases, one can utilize the grid power as a backup in combination with on-site generation systems such as gas turbines or diesel engines combined with fuel cells as the primary source. However, none of the data centers in our study used on-site power generation.

Data Center Infrastructure Management Tools Energy monitoring allows greater visibility into overall data center energy usage while providing solutions to maximize server and infrastructure equipment operating efficiency. Many data centers use Data Center Infrastructure Management (DCIM) tools to monitor energy and cooling efficiency and claim 20% savings in operational expenses [GRD13]. IoT also has an important role to play in the monitoring of data centers, as we will see in Chapter 4. The majority of the interviewed data centers utilize DCIM tools.

3.1.2.2 Cooling, Thermal, and Air Management Practices

The data center cooling system ensures that air flows from the raised floor to the air inlets of the IT equipment in order to maintain the desired thermal envelope [BP06]. As cooling is one of the main contributors to a data center's energy footprint, the energy efficiency of a data center can be improved by implementing the appropriate air management practices. What follows are best practices that promote sustainable cooling, thermal, and air management.

Custom Central Air Handler Efficient airflow can be achieved by eliminating bypass and re-circulation air flows, as this is where the airflow is wasted in a data center. All the data centers in our study used horizontal, vertical, under-rack panels, and PVC curtains for isolation with the goal of minimizing the re circulation of hot air. All the data centers also used high-raised floors, overhead cabling, cable grouping, the placing of cable trays below the hot aisle, and cabling within the cabinets and racks to avoid air blockages. The best practice is to have dedicated horizontal airflow rather than a mixture of vertical and horizontal airflow, because dedicated horizontal airflow provide much more uniform distribution. During the inspection of the return air ducts for HVAC, we observed inadequate ceiling height or undersized hot air return plenum in a few of the data centers. Increasing the size of the return duct to match the air handler avoids this problem. Use of high overhead plenum and several feet of clearance under the raised floor provides better maintenance. For efficient airflow management, some of the data centers in our study made use of custom-designed central air handler systems

Liquid Cooling As coolant liquids have a higher thermal conductivity than air, liquid cooling is an interesting solution to data center, and is typically deployed on room-level and row-level systems. The most common type of liquid cooling is direct-to-chip cooling, which uses flexible tubing and copper plates to bring the cooling liquid directly to the heat sources. Only one data center in our study is using liquid cooling, as it is more expensive than air-based cooling. A more extreme solution to liquid cooling is immersion cooling, where the hardware is completely immersed in a dielectric fluid. The state of immersion cooling, as well as the benefits and drawbacks are discussed in Section 3.3.

Sensors for Chiller Plant Monitoring is critical in data centers, not only for IT equipment, but also for the cooling systems. Load-monitoring sensors for chiller plants enable the data center to determine if the current cooling capacity is sufficient, or if it needs to be increased or decreased. All of the data centers we interviewed used chiller plant sensing.

Hot and Cold Aisle Containment When the rack aisles are not contained, there is a risk of by-pass and re-circulation airflow, as shown in Figure 3.1. Ideally, all the hardware in a row of cabinets faces the same way so that hot air is expelled on one side and cold air blows from the other side. And air from the hot and cold aisles should not be allowed to mix. All of the data centers followed a containment approach that allows the proper flow of cold air to the destination while preventing mixing of air, in turn reducing energy consumption.

LOOP Design for Chillers The control of humidity in data centers is essential to achieve high availability and reduce maintenance costs. The level recommended is around 50% or higher. However, data centers without large high-speed fans can safely operate at 40% humidity levels, thus decreasing water and energy consumption [IBM12]. Humidity can be best controlled knowing both inside and outside environmental conditions. Adiabatic humid-

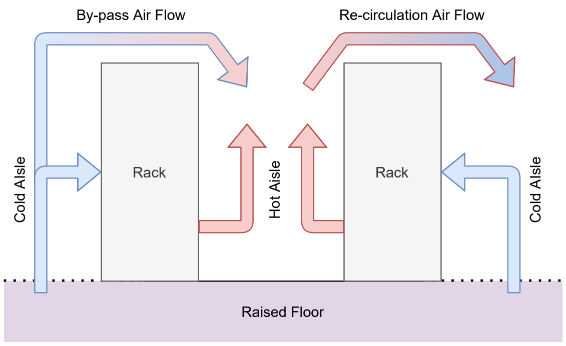


Figure 3.1: Hot aisle/cold aisle air mixing in data centers.

ification technology provides higher efficiency than infrared or isothermal technologies [CH16a]. Many of the data centers used LOOP design, with a median temperature of 10°C to 15°C [Har99]. A LOOP design uses a closed feedback control loop to optimally adjust the parameters of the cooling systems.

Adjustable Speed Drive Chillers Variable frequency fans in the CRAH units would allow for self-adjusting, thus resulting in energy savings. Most of the data centers in our study were operating at conservatively low baseline temperatures, but raising the baseline temperature would save 4% in energy costs with each degree of increase in the temperature set point [Ahu12]. The use of variable speed drive chillers, which slow down their motors to match the varying capacity requirements, can further reduce the energy consumption of the cooling system. Less than half of the interviewed data centers make use of this technology.

3.1.2.3 Green Practices

A green data center incorporates energy-efficient design with high-efficiency power delivery, highly efficient cooling, and increased utilization of renewable energy sources [GLMP13; Mur08]. What follows next are some of the approaches and practices for the development of green data centers.

Economizers (Free Cooling) Data centers can achieve significant energy savings through the use of water-side or air-side economizers. Economizers make an impact when wet bulb temperatures outside the data center are less than 13°C for more than 3,000 hours per year. Four out of seven interviewed data centers used economizers. Some of the data centers in our study claimed up to a 20% decrease in energy costs and a 7% decrease in maintenance costs since deploying economizers. However, the use of economizers depends on the geographical location, weather conditions, and economizer design.

Reclaimed Water for Data Center Cooling The use of reclaimed, or so-called gray water, is neither harmful to the environment nor to human health. Using gray water for cooling is considered more ecologically and environmentally friendly because it reduces demands for ground water and does not require energy for the recycling process at waste-water treatment sites. This best practice was applied by none of the data centers.

Cooling Water Re-circulation Using the same water for several cycles of cooling operations reduces water consumption. The water savings improve data center's efficiency and lower the impact on the environment and on potable water supplies while simultaneously cutting costs. Only two data centers utilized the re-circulation of water for their cooling systems.

Renewable Resources Renewable energy typically comes from solar panels, wind turbines, or hydroelectric installations. As renewable energy production is intermittent in nature and dependent on the geographical location, it is often combined with energy storage facilities. Nevertheless, these are still expensive installations. Most of the data centers in our study were not using green energy.

3.1.2.4 Storage and Network Practices

Generally, a data center is seen as a facility used to house computer systems and associated components, such as telecommunications and storage systems. Storage and networking equipment are critical pieces of IT equipment within the data center. Some of the best practices followed for storage and network communications of a data center are described as follows.

Storage Tiering Tiered storage is an approach to organizing storage that aims to reduce overall storage costs and simultaneously improve availability and performance by ranking data based on business value and access frequency. The way this is achieved is by categorizing the data in different tiers, for example: mission critical data, hot data, warm data, and cold data. Depending on the category, the appropriate storage requirements can be defined. Mission critical data requires high levels of redundancy and backups, hot data depends on highly throughput storage devices, and for cold data the size of storage is more important than the performance. All of the interviewed data centers use storage tiering.

Automatic Waste Storage Management Reducing wasted storage is an important step in decreasing the e-waste of a data center. Managing wasted storage reduces the need for the data center to rapidly expand its storage capabilities. Actively monitoring the storage capacity is important for the automation of wasted storage management. All of the interviewed data centers have procedures or tooling in place to reduce the storage waste.

Centralized Control and Storage Optimization All of the data centers we surveyed have centralized control over the servers, storage, and databases for storage optimization. A solid-state drive helps reduce the energy consumption of spinning disks and handles the enormous demand on storage systems. Some of the data centers in our study were using pooling storage, hybrid storage, and flash caches.

Software-defined Networking Software-defined Networking (SDN) is the virtualization of networking and storage infrastructure, it offers resource flexibility, optimal resource usage, and scalability [DKR13]. All of the data centers had either adopted SDN or are planning to adopt SDN in the near future. Some were using network virtualization, which allows each customer to have their own network with different controller applications and balances the performance, port utilization, and traffic demands. The data centers in our study used automation tools to predict network loads and avoid outages.

Storage Pooling and Geo-replication All of the data centers were using geo-replication for storage backup, e-discovery, and data mapping for archiving data, whereas flash storage was used for specific applications in very few of the data centers in our study.

3.1.3 Recommendations for Data Center Operators

Based on the results of the multiple case study, we have compiled a number of recommendation and general observations. First of all, data center life cycle management helps enterprises understand key management tasks, connections between different phases, and the pitfalls that exist in each phase. Generally, the data center life cycle is comprised of five phases: plan and analyze, design, build, operate, and continuous evaluation. For initial phases, it is better to use reference designs to validate the early project choices and develop system concepts. Considering the whole chain of operations, data center operations become the base layer with the goal of optimizing not only energy and cost, but also of helping with the long-term planning and provisioning of equipment and resources.

Nowadays, cloud computing is of strategic importance, benefiting both providers and their customers. If a new data center is under-utilized, it can act as a cloud provider for other data centers and customers. To accommodate the growing demands of users and other background processes using the same physical resources, data centers are required to make optimal use of all the resources by increasing utilization and visibility. Proper selection of virtual machines for migration minimizes the number of power-on nodes. Designing and implementing fast energy-efficient virtual machine allocation and selection algorithms considering multiple resources can result in energy efficient data centers.

Maintaining a separate, direct current feed to power the telecommunications and storage systems directly will reduce energy consumption, building costs, and conversion losses. DCIM or automation tools can achieve considerable energy savings, ranging from 5 to 20% [GRD13]. Centralized cooling systems in a large-scale data center can be optimized by maintaining a median temperature of 10°C to 15°C, using adjustable-speed drive chillers, storing excess thermal energy, and by installing energy- and load-monitoring sensors. Following the recommendations for chiller plants will have quick return on investment in the order of two to three years due to energy savings.

Hot and cold aisle containment, increasing the data center supply air temperatures, using air-side or water-side economizers, and increasing the room temperature all reduce the cooling capacity requirements of the CRAH units. Furthermore, increasing the temperature of the chilled-water supply can still provide sufficient cooling for a data center while reducing the number of hours of compressor-based cooling.

To improve the PUE, it is important to understand the baseline power consumption of the data center. It is also important to monitor and evaluate compute efficiencies per server type, adjust operations according to peak power utilization, and shift resource usage based on usage profiles. There is a need for DCOs to correlate the infrastructure investments more closely to the actual resource requirements of applications.

Table 3.3 and Table 3.4 summarize the various implementation issues in the dimensions we studied that lead to inefficiencies, as well as the solutions and best practices that were uncovered to avoid these problems and improve the data center's efficiency. In short, optimizing the cooling plant, operational parameters of the data center, uninterruptible power supply load, and zombie servers along with controlled lighting, continuous monitoring, and proper airflow management can improve the energy efficiency of a data center. Placing cold data on slower, larger drives that use less power can

Table 3.3: Issues and solutions for the various dimensions, part 1.

Implementation Issues	Solutions / Best Practices
Energy efficiency	
Some virtual machines are allocated more resources than requested to achieve high performance, leading to inefficient use of resources.	Virtualization and consolidation; decommission servers not performing useful work.
There are several power conversion steps while delivering power to IT equipment, leading to losses in power distribution.	On-site power plant or direct DC feed to power ICT equipment.
Most data centers do not monitor detailed energy use, which can help operators find the actual point of losses and inefficiencies.	Utilize DCIM tools, detailed sub-metering at the component level, IoT-based monitoring.
Power savings and performance requirements can lead to service level agreement SLA violations.	Placement policies to ensure performance and meet SLAs constraints.
Cooling, thermal, and air management	
High-density racks may result in one or more areas of excess temperature known as "hot spots", which result in deteriorated performance.	A global capacity plan as is known from the Information Technology Infrastructure Library (ITIL) best practices.
Inadequate air management can lead to bypass and re-circulation air flow, which reduces the cooling efficiency.	Placing perforated tiles near hot spots, utilizing blanking panels, ensuring hot and cold aisle separation.
Air blockages lead to poor flow of cool air to the server inlets.	PVC curtains for isolation, seal cable cutouts, or other openings in under-floor distribution systems, use overhead cabling technology.

Table 3.4: Issues and solutions for the various dimensions, part 2.

Implementation Issues	Solutions / Best Practices
Cooling, thermal, and air management	
High heat density in racks restrict efficient space utilization.	Central air handler, LOOP design for chillers, adjustable- speed drive chillers and economizers.
Improper configurations lead to inefficient use of chillers.	Installing energy and load monitoring sensors, IoT sensors.
Over-sized cooling infrastructure limits operating capacity.	Increasing the data center supply air temperatures and room temperature.
Inadequate duct size results in poor airflow.	Using waterside economizer and higher temperature- chilled water.
Greenness	
Data centers consume a lot of water but fail to recycle or use water in an efficient manner.	Cooling water re-circulation and using reclaimed water.
Recycling of electronic equipment and other materials are not in place.	Recycling ICT equipment through certified authorities
Storage and Network	
Storage over-provisioning and massive volumes of redundant data lead to inefficient use of storage and consume more energy.	Automating waste storage management.
Increasing the number of virtual machines consumes more storage.	Storage tiering centralized control and optimization.
High throughput and performance requirements limit the efficient use of storage and network.	Joint host-network traffic management and power optimization in SDNs, software-defined flash storage and storage pooling.
Connectivity issues might lead to outages.	Using traffic engineering techniques to alleviate loss and reduce congestion.

also reduce the energy footprint. In our study, DC3 claimed a reduction of 20% in their PUE score by following best practices in cooling, thermal and air management, and storage and network practices. DC1 claimed a PUE reduction of 20% by strictly adhering to energy-efficiency practices such as virtualization, consolidation, and automation tools. DC2 and DC4 claimed a PUE reduction of 10% by effectively following the best practices of energy efficiency and cooling, thermal, and air management.

3.1.4 Conclusions

Best practices play an important role in identifying areas of improvement, and in increasing the data center's operational efficiency. The greenness best practices specifically aim towards increasing the sustainability aspects of the data center. We studied issues in seven data centers in India and the Netherlands to identify and describe which of the best practices are applied with regards to sustainability. In total, 23 best practices were selected, across the domains of cooling-, thermal-, and air-management, as well as energy efficiency, greenness, and storage and networking. Each of the seven data centers were subjected to these best practices, and results show that most of the practices were indeed implemented. However, there were only two data centers which implemented the majority of the greenness practices. Progress can also be made when it comes to using reclaimed water for cooling, re-circulation of cooling water, and the use of renewable resources. Furthermore, none of the interviewed data centers make use of smart lighting controls, or on-site power generation. Implementing all of the listed best practices enables data centers to become more efficient and sustainable. Now that we understand what actions to implement to increase sustainability in data centers, the next question is how can we determine the effect that these best practices have on the data center? This is where the use of the appropriate metrics is critical.

3.2 Metrics for Sustainable Data Centers

Metrics are quantifiable assessments, they are useful for evaluating, comparing, and monitoring performance over time. More formally, metrics are an empirical, objective assignment of numbers, according to a rule derived from a model or theory, to attributes of objects or events with the intent of describing them [FP16]. Metrics often play an important role in the form of Key Performance Indicators, which track the progress towards certain goals. In the context of data centers, metrics enable the data center operators to have a better view on potential inefficiencies and areas of improvement. Data center metrics also allow architects and operators to measure the performance and effects of changes made to subsystems, for example when implementing sustainability practices. Poorly defined metrics will impede business innovation and prevent meeting environmental sustainability goals.

The number of available metrics that can be applied in the context of data centers is vast. To assist in the process of metric selection, we present an analysis of metrics that are commonly used in data centers, starting from the power grid and going all the way up to the service delivery. One of the major contributions presented in this section is the identification of various metrics relating to a data center and the classification based on the different core dimensions of data center operations. We define the core dimensions of data center operations as follows: energy efficiency, cooling, greenness, performance, thermal- and air-management, network, security, storage, and financial impact. An emphasis is put on metrics that measure the performance of the data centers in terms of sustainability. Furthermore, we derive relationships between metrics, and discuss the advantages and disadvantages of each metric in order to expose the research gaps and illustrate the latest research trends in computing the efficiency of a data center. We present a taxonomy of state-of-the-art metrics used in the data center industry which is useful for the researchers and practitioners working on monitoring and improving the energy efficiency of data centers. And finally, we identify issues and open challenges with regards to the state of the art in data center metrics.

3.2.1 A Taxonomy of Data Center Metrics

The continuous monitoring and evaluation of metrics is a critical tool for organizations to gain information and insights to control and optimize their data centers, and to measure the impact of policy changes. To remain competitive with their peers in the industry, organizations must ensure optimal utilization of resources in order to increase efficiency while minimizing environmental impact. This is only possible if there is information available that is meaningful and actionable. Well-defined and organized metrics increase the organization's productivity and assist with making management decisions. The goal of our taxonomy of metrics is to bring order and structure in the heterogeneous landscape of data center metrics, as well as identify relationships between existing metrics, and uncover potential issues and challenges.

For efficient and eco-friendly operation of data centers, we need to monitor all the components of a data center. The components that we have identified are visualized in Figure 3.2. At the top level, we have the entire facility, which encompasses energy and other resources going into IT related components, and into support components such as lighting, HVAC, and offices. The IT power flows to the Power Distribution System (PDS) and UPS, which further distributes the power to IT equipment. The IT equipment consists of servers that are organized into racks. Servers can include application servers, networking equipment such as switches, routers, and storage servers. This classification enables us to assign a category to each metric and group them based on these categories.

Besides metric categories, we also identify the dimension in which the metric operates. The following dimensions emerge as core dimensions: Energy Efficiency, Cooling, Greenness, Performance, Thermal and Air Management, Network, Storage, Security and Financial Impact. Each dimension has many different metrics, across different categories, each with its own approach to measuring performance of a data center, each with its own advantages, drawbacks and limitations. We provide a survey of data center metrics, and for each metric describe the unit in which it is expressed, the objective, the

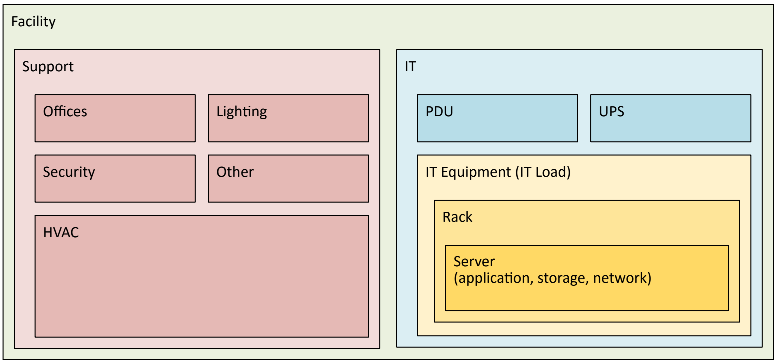


Figure 3.2: The identified metric categories for data centers.

optimal value as well as the scale at which the metric operates. The objective specifies the optimization that should be done for a given metric, which typically means either minimizing or maximizing the metric. The optimal value is the ideal or target value for the metric. Furthermore, there are inter-dependencies between individual metrics, as some are based on or have a strong relationship with other metrics. We identify these relationships and label them as 'uses'-relationships and 'based on'-relationships. The 'uses'-relationship exists when a metric uses another metric directly as input for the calculation. The 'based on'-relationship indicates that a metric is based on the principles of another metric, but does not directly use this metric as input. It is critical to understand these dependencies, as the shortcomings of one metric are not necessarily overcome when it is used in the calculation of another metric. What follows next is a detailed review of metrics per dimension. For a complete description and definition of all metrics we refer to our work in [RSR+17]. Only the abbreviation of the metric are given in the tables, for the full name of the metric we refer to the list of acronyms of this thesis.

Table 3.5: An overview of energy efficiency metrics, part 1.

Acronym	**Unit**	**Objective**	**Optimal**	**Category**	**Ref.**
APC	Ratio	Maximize	1.0	Facility	[AVS+15]
CADE	%	Maximize	1.0	Facility	[KFK08]
CPE	%	Maximize	1.0	Facility	[BM07]
DCA	Ratio	Minimize	$-\infty$	Facility	[AVS+15]
DCcE	%	Maximize	1.0	Server	[Bla10]
DCeP	UW / kWh	Maximize	∞	Facility	[SMR+12]
DCiE	%	Maximize	1.0	Facility	[AAFP12]
DCLD	kW / ft^2	Minimize	0.0	Facility	[Ras05]
DCPD	kW / Rack	Maximize	∞	Rack	[Ras05]
DCPE	UW / Power	Maximize	∞	Facility	[Gre07]
DC-FVER	Ratio	Minimize	1.0	Facility	[NLLS12]
DH-UE	%	Maximize	1.0	Server	[SBK07]
DH-UR	%	Maximize	1.0	Server	[SBK07]
DPPE	Ratio	Maximize	1.0	Facility	[Dpp]
DWPE	Perf / Watt	Maximize	∞	Server	[WAP+14]
EES	Ratio	Maximize	1.0	Facility	[AVS+15]
EWR	Ratio	Minimize	0.0	Facility	[SSO+14]

Table 3.6: An overview of energy efficiency metrics, part 2.

Acronym	Unit	Objective	Optimal	Category	Ref.
GEC	%	Maximize	1.0	Facility	[Dpp]
H-POM	Ratio	Minimize	1.0	IT Equipment	[SBK07]
ITEE	Cap / kW	Maximize	∞	IT Equipment	[Dpp]
ITEU	%	Maximize	1.0	IT Equipment	[Dpp]
OSWE	OS / kW	Maximize	∞	Facility	[HMP+09]
PDE	%	Maximize	1.0	Rack	[LHF14]
PEsavings	Ratio	Maximize	1.0	Facility	[AVS+15]
PUE_{1-4}	Ratio	Minimize	1.0	Facility	[AAFP12; SSO+14]
$\text{PUE}_{scalability}$	%	Maximize	1.0	Facility	[TCF+09]
pPUE	Ratio	Minimize	1.0	Facility	[AAFP12]
PpW	Perf / Watt	Maximize	∞	Server	[Bar05]
ScE	%	Maximize	1.0	Server	[Bla10]
SI-POM	Ratio	Minimize	1.0	Facility	[SBK07]
SPUE	Ratio	Minimize	1.0	Facility	[WAP+14]
SWaP	Ratio	Maximize	∞	Rack	[Gre05]
TUE	Ratio	Minimize	1.0	Facility	[PPH+13]

3.2.1.1 Energy Efficiency Metrics

The energy efficiency of a system is defined as the ratio of useful work done by a system to the total energy delivered to the system. For data centers, the energy efficiency translates into the useful work performed by different subsystems, where useful work is typically the computational load. An overview of available energy efficiency metrics is presented in Table 3.5 and Table 3.6. The unit of each metric is listed, including the objective, optimal value and the category to which it belongs. We analyze the relationship between these metrics and present them in Figure 3.3, where we organize the metrics horizontally based on their category and visualize the relationships that exist among them.

The most popular energy efficiency metric, PUE, is used by a large number of other metrics either directly or as a derivation. For example, Server Power Usage Efficiency (SPUE) and pPUE metrics are based on the same principles as the PUE metric. The Data Center Performance Per Energy (DPPE) metric is also noteworthy as the metric is a combination of four other metrics: DCiE, Green Energy Coefficient (GEC), IT Equipment Energy (ITEE), and IT Equipment Utilization (ITEU). The PUE is defined as the ratio of the total energy consumption of the data center, and the total energy consumption of the IT equipment. Since the IT energy consumption is included in the total energy, the value of PUE will typically be greater or equal to one.

$$PUE = \frac{Total\ Facility\ Energy}{IT\ Equipment\ Energy} \tag{3.1}$$

To calculate the ITEU, one needs to know the exact power used by fans, voltage regulators and other components inside IT equipment. One of the challenges is the accurate measurement of the total energy that goes into IT equipment. Defining coefficients for different types of IT equipment is also challenging, especially in the heterogeneous environments of co-location data centers. To accurately calculate the Operating System Workload Efficiency (OSWE) metric, the number of operating systems needs to be known, including the operating systems of virtual machines. We can conclude that some of these metrics require accurate and very hardware-specific data in order to be useful. In other words, it requires monitoring of the data center at a large scale. In Chapter 4 we will further investigate data center monitoring and explore how data centers can be monitored at a large scale using the principles of IoT.

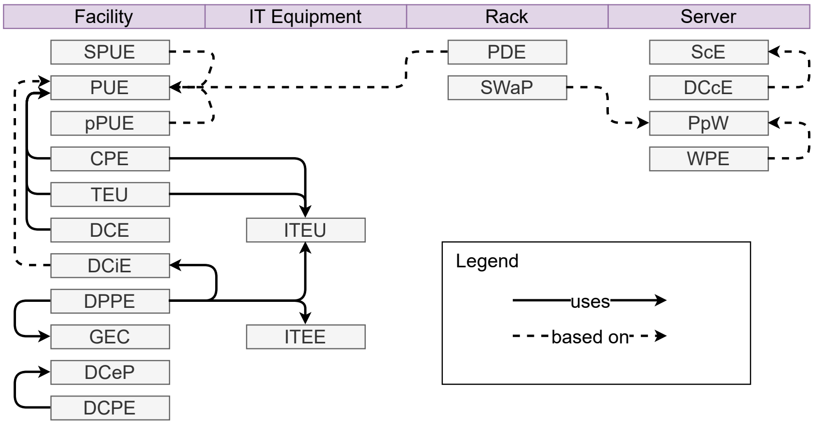


Figure 3.3: Relationships between energy efficiency metrics.

3.2.1.2 Cooling Metrics

Almost all of the electrical energy used for computations in a data center is transformed into thermal energy, or heat. The heat generated by the IT equipment must be controlled and removed to maintain high levels of operational performance and ensure that hardware operates within their environmental envelope. Therefore, cooling plays a vital role in any data center. The complex interconnection of HVAC systems ensures optimal conditions for the computing environment in a data center, guaranteeing the life span, scalability and flexibility of the servers [KK15]. An overview of the available cooling metrics that can be applied in the context of data centers can be found in Table 3.7.

Table 3.7: An overview of cooling metrics.

Acro.	Unit	Objective	Optimal	Category	Ref.
AEUF	%	Maximize	1.0	HVAC	[LC13]
CoP	Ratio	Maximize	∞	Facility	[PP09]
DCCSE	kW/ton	Minimize	0.0	HVAC	[MGGS09]
DCSSF	Ratio	Minimize	1.0	HVAC	[MGGS09]
EER	Ratio	Maximize	∞	Facility	[SSO+14]
HSE	Ratio	Maximize	3.5	HVAC	[Van11]
RI	Ratio	N/A	N/A	HVAC	[VS06]
WEUF	%	Maximize	1.0	HVAC	[Han08]

3.2.1.3 Green Metrics

The reduction of the carbon footprint and greenhouse gases are critical for the future of our society and are therefore becoming subject to governmental regulations and taxes. As a result, the "greenness" of a data center is becoming increasingly important. A green data center is defined as a system in which the mechanical, lighting, electrical and IT equipment are designed to maximize energy efficiency and minimize environmental impact [BBDC+15; Mur10]. Green IT addresses the sustainability issues by improving energy efficiency, lowering greenhouse gas emissions, using renewable resources, and by encouraging reuse, and recycling [MG12]. Table 3.8 presents various green metrics which reflect the greenness of the data center in terms of carbon footprint, heat reuse, efficiency of water consumption and use of renewable energy resources. Figure 3.4 illustrates the relationships between these metrics using the following concepts: reducing resource usage, reusing resources, recycling resources, renewable resource usage. We organize the green metrics horizontally according to the these four concepts and vertically based on the category in which they operate.

Table 3.8: An overview of green metrics.

Acronym	**Unit**	**Objective**	**Optimal**	**Category**	**Ref.**
CO_2S	Ratio	Maximize	1.0	Facility	[AVS+15]
CUE	KgCO_2/kWh	Minimize	0.0	Facility	[DAMPT10]
EDE	%	Maximize	1.0	Facility	[BBC+12]
ERE	%	Minimize	0.0	Facility	[PTV+10]
ERF	%	Maximize	1.0	Facility	[PTV+10]
GEC	%	Maximize	1.0	Facility	[Dpp]
GUF	%	Minimize	0.0	Facility	[AVS+15]
MRR	%	Maximize	1.0	Facility	[Pow11]
ω_{energy}	Ratio	Minimize	0.0	Facility	[SSB+09]
TCE	Pounds of CO_2/kWh	Minimize	0.0	Facility	[Coo07]
TGI	Ratio	N/A	N/A	Facility	[SF12]
WUE	Liters/kWh	Minimize	0.0	Facility	[PABP11]

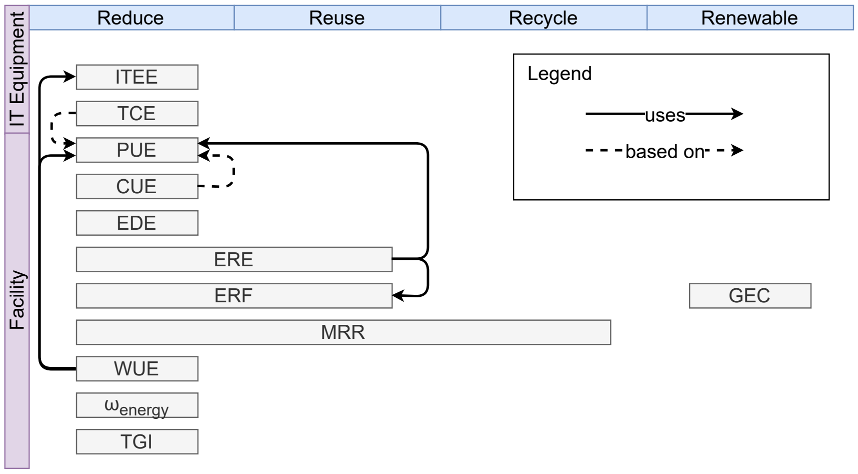


Figure 3.4: Relationships between green metrics.

3.2.1.4 Performance Metrics

The performance of a data center is the total effectiveness of the system, including throughput, response time, and availability [WK13]. Measuring performance and productivity is crucial as sub-optimal performance has operational and financial implications for a data center. When determining the performance of a data center one can encounter several difficulties including: identifying workloads, overhead of performance measurements, energy distribution losses, and measuring the energy consumption at various levels of the data center. Measuring the actual performance and productivity allows data center operators to determine how to further improve the performance and plan for future work loads. An overview of the metrics which measure the performance of various components in data centers is presented in Table 3.9.

Table 3.9: An overview of performance metrics.

Acro.	Unit	Objective	Optimal	Category	Ref.
ACE	Ratio	Maximize	1.0	HVAC	[BDD14]
DCP	Work / W	Maximize	∞	Facility	[ACD08]
DEEPI	Prod. / Watt	Maximize	∞	Facility	[Bri07]
DR	Ratio	Maximize	1.0	Server	[WA12]
EP	Ratio	Maximize	1.0	Server	[RPE11]
FpW	Ops / J	Maximize	∞	Server	[BC10]
IPR	Ratio	Minimize	0.0	Server	[VAG10]
LD	Ratio	Minimize	0.0	Server	[WA12]
LDR	Ratio	Minimize	0.0	Server	[VAG10]
PG	Ratio	Minimize	0.0	Server	[WA12]
SWaP	Ratio	Maximize	∞	Facility	[Gre05]
U_{CPU}	%	Maximize	1.0	Server	[BC10]
U_{DC}	%	Maximize	1.0	Facility	[BP08]
U_{server}	%	Maximize	1.0	Server	[BP08]
UPS_{CF}	Ratio	Optimize	1.4	UPS	[Ras06]
UPS_{EE}	%	Maximize	1.0	UPS	[Giu11]
UPS_{PF}	Ratio	Maximize	1.0	UPS	[Ras06]
UPS_{PFC}	Ratio	Maximize	1.0	UPS	[NO10]
UPS_{SF}	Ratio	Optimize	1.5	UPS	[Ras06]

3.2.1.5 Thermal and Air Management Metrics

Thermal and air management metrics measure environmental conditions of the data center and also determine how air flows within a data center, from cooling units to the vents. These metrics assist with the diagnostic analysis to determine, for example, the amount of re-circulation and by-pass air. In general, the metrics are based on the relationship between air flow rate and ambient temperature. When it comes to thermal and air management, the metrics can be influenced by internal parameters, such as the data center's layout and configuration, as well as external parameters, such as geographical location [WB15]. Metrics like temperature, humidity, dew point and heat flux are used to prevent the over-heating, maintain the humidity levels, capture the current condition of the cooling system, and to assist with making the correct decisions. The dimension, objective, optimal value of the outcomes, and the scale at which these metrics operate are presented in Table 3.10.

Table 3.10: An overview of thermal and air management metrics.

Acro.	**Unit**	**Objective**	**Optimal**	**Category**	**Ref.**
AE	W/cfm	Minimize	0.0	Facility	[MGGS09]
β_{index}	Ratio	N/A	N/A	Rack	[QLL13]
BPR	Ratio	N/A	N/A	Facility	[TS10]
BR	Ratio	N/A	N/A	Facility	[TS10]
CI	%	Maximize	1.0	HVAC	[VS07]
DC	^oC /^oF	Optimize	18-27^oC	Facility	[ASH11]
DP	^oC /^oF	Optimize	17^oC	Facility	[ASH11]
HF	W/m^2	Minimize	0.0	Facility	[ASH11]
IoTemp	%	Minimize	0.0	Rack/Server	[SSO+14]
D^2	Unit	Minimize	0.0	Facility	[NHE+03]
M$_x$	cfm	N/A	N/A	Facility	[TKS09]
NPR	Ratio	N/A	N/A	Facility	[TS10]
RCI	%	Maximize	1.0	Rack	[Her05]
RH	%	Optimize	60%	Facility	[Eva15]
RHI	Ratio	Maximize	1.0	Facility	[SBP02]
RR	Ratio	N/A	N/A	Facility	[TS10]
RTI	%	Optimize	1.0	Rack	[HK07]
SHI	Ratio	Maximize	1.0	Facility	[SBP02]

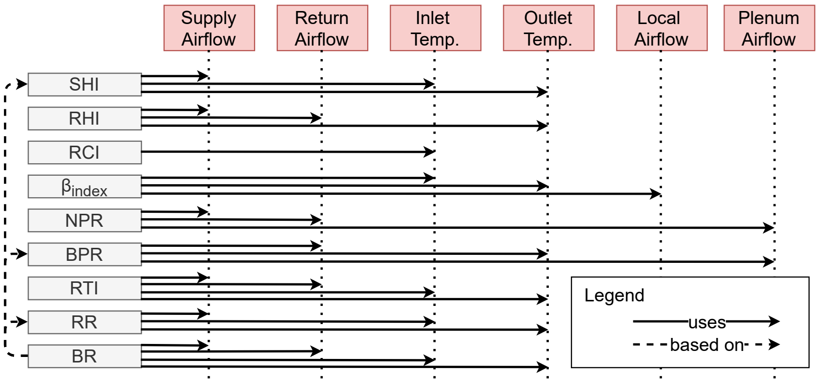


Figure 3.5: Relationship between thermal and air management metrics.

Air management metrics address air flow efficiency and separation of hot and cold air streams. We observed that most of the air management metrics depend on common inputs. We have analyzed these metrics, looking specifically at their inputs, airflow path, and purpose of each metric. The result of this analyses can be seen in Figure 3.5. Noteworthy is the fact that Return Heat Index (RHI) and Supply Heat Index (SHI) differ in airflow paths, and that Balance Ratio (BR) can be developed as a function of Recirculation Ratio (RR) and Bypass Ratio (BPR).

3.2.1.6 Network Metrics

The data center network is a core component of the data center, providing numerous interconnectivity services. Networking equipment is responsible for up to 15% of a data center's amortized cost [GHMP08]. To increase the efficiency of data centers, operators should improve the energy efficiency of their network. At the same time, performance levels should be maintained as decreased network performance harms application performance and can lead to revenue loss. A data center's network performance can typically be characterized using well-known metrics such as bandwidth, Network Power Usage Effectiveness (NPUE), Communication Network Energy Efficiency (CNEE), reliability and throughput [AMW+10]. An overview of the network metrics with the unit of each metric, objective, optimal value and the scale at which these metrics operate are presented in Table 3.11.

Table 3.11: An overview of network metrics.

Acro.	**Unit**	**Objective**	**Optimal**	**Category**	**Ref.**
BJC	b/J	Maximize	∞	IT Equip.	[RM07]
CNEE	J/b	Minimize	0.0	IT Equip.	[FKBZ15]
DS	Ratio	Optimize	1.0	IT Equip.	[CCC12]
ECR-VL	W/Gbps	Minimize	0.0	IT Equip.	[ANK10]
NPUE	Ratio	Minimize	1.0	IT Equip.	[FKBZ15]
NT_{kwh}	b/kWh	Maximize	∞	Facility	[Glo14]
PS	Ratio	Optimize	1.0	IT Equip.	[CCC12]
RS_{max}	Ratio	Maximize	1.0	IT Equip.	[CCC12]
TEER	Ratio	Maximize	∞	IT Equip.	[ATI14]
$U_{network}$	%	Maximize	1.0	IT Equip.	[BP08]

3.2.1.7 Storage Metrics

Efficient and performant storage for cloud data centers can be challenging to achieve as it requires interaction with many components in the infrastructure such as application servers, storage devices, and network equipment. By applying a set of metrics for storage operations in the data centers the storage performance can be increased by continuous monitoring of these metrics [CAP+11]. Overall Storage Efficiency (OSE) and slot utilization, for example, enable insights into how efficiently the storage capacity is utilized. Traditional metrics are unable to capture the improved efficiency achieved by new tools and methods such as trim storage and just-in-time allocations. We perceive the requirement for a single set of metrics that reflects storage utilization across a changing technology base. We analyze and present the current storage metrics along with their units as well as the objective, optimal value of the outcomes and the scale at which these metrics operate in Table 3.12.

Table 3.12: An overview of storage metrics.

Acro.	Unit	Objective	Optimal	Category	Ref.
CEE	GB/W	Maximize	∞	Storage	[Sch09]
LSP	%	Maximize	1.0	Storage	[CAP+11]
OSE	Ratio	Maximize	1.0	Storage	[CAP+11]
RT	ms	Minimize	0.0	Storage	[Sch09]
SU	%	Maximize	1.0	Storage	[Sch09]
$TP_{i/o}$	B/s	Maximize	∞	Storage	[MK91]
U_{mem}	Ratio	Maximize	1.0	Storage	[MK91]
$U_{storage}$	%	Maximize	1.0	Storage	[BP08]

3.2.1.8 Security Metrics

Data centers can be designed to withstand everything from corporate espionage to terrorists, to natural disasters. To ensure high security standards, data centers need to follow several practices and guidelines. Data centers should be built on the right site with walls capable of withstanding explosions. To handle fire break outs, data centers should establish fire compartments and monitor the environment with the help of aspirating smoke detectors. Data centers should have redundant utilities, a buffer zone around the site, a limited number of entry points, plenty of surveillance cameras, etc. In addition, data center employees, customers and visitors should be authenticated at least three times [ANI12]. Security metrics quantify how well security strategies are deployed. A security metric is defined as a system of related dimensions (compared against a standard) enabling quantification of the degrees of freedom from the possibility of suffering damage or loss from malicious attacks [Abb11]. The basic security goals in a data center include authentication, authorization, and data protection.

Most of the data centers have layered security in place. The number of layers of security increases with the tier of the data centers, with Tier 4 data centers having more than 6 security levels [Ctr15; Sca15]. Layered security include perimeter fences equipped with senors, badge access to inner doors, guard escorted visitors, a floor to ceiling turnstile, access card or bio-metric authentication to secure parts of the data center, video surveillance, and locked cages for servers. Some data centers use testing, development and production zones, where production zones have high security and testing zones have less [Yas09].

For full control, it is advised to have security zones in a data center networks to provide better visibility and improve detection performance [Lyo12]. A security zone is created in a network, consisting of a group of IT equipment that have similar access control requirements. Security zones are logical entities that provide isolation and minimize security risks. They are organized as layered trust zones with inner layers having higher levels of security than the outer ones. This layering offers one way communication from higher trust zones to lower trust zones. Furthermore, virtual private networks can be used to manage and protect the networked environment. In a virtualized environment, strict enforcement of security policies may not be possible due to migrations of virtual machines across data centers [BBG10]. Providers and customers should communicate their expectations for security as part of agreement process and component level security controls need to be developed in the shared control model. Table 3.13 lists the metrics for the complexity and performance of firewalls, intrusion detection, and prevention systems.

3.2.1.9 Financial Impact Metrics

Within organizations, financial impact metrics are useful for setting up budgets and measuring project expenses [APP15].In the context of data centers, employing financial metrics in a balanced score-card approach can assist the data center operators with placing other key metrics, such as outage reports and service quality metrics, in a financial perspective. An example of an

Table 3.13: An overview of security metrics.

Acro.	**Unit**	**Objective**	**Optimal**	**Category**	**Ref.**
ACPR	Count	Minimize	0.0	IT Equip.	[AHAS11]
AS	Count	Optimize	-	IT Equip.	[AHAS11]
ATR	b/s	Maximize	∞	IT Equip.	[Sny10]
CC	Count	Maximize	∞	IT Equip.	[AHAS11]
CER	Con/s	Maximize	∞	IT Equip.	[AP03]
CTR	Con/s	Optimize	-	IT Equip.	[AP03]
DeD	Count	Maximize	∞	Facility	[Sny10]
DeP	-	Maximize	1.0	IT Equip.	[BM08]
DTE	Count	Minimize	0.0	IT Equip.	[BM08]
FC	Ratio	Optimize	-	IT Equip.	[AHAS11]
FL	ms	Minimize	0.0	IT Equip.	[Sny10]
HTR	b/s	Maximize	0.0	IT Equip.	[AP03]
IAS	Count	Optimize	-	IT Equip.	[AHAS11]
IPFH	-	Maximize	∞	IT Equip.	[AP03]
ITH	%	Maximize	∞	IT Equip.	[AP03]
RA	Count	Optimize	-	IT Equip.	[AHAS11]
RC	Count	Minimize	0.0	Facility	[BM08]
RCD	Days	Minimize	0.0	IT Equip.	[BM08]
T	Days	Minimize	0.0	IT Equip.	[BM08]
TP_{IP}	b/s	Maximize	∞	IT Equip.	[New99]

important financial metric is Total Cost of Ownership (TCO), which is the main cost driver for IT and represents a significant expense for other units such as cooling and lighting. The metric empowers us to settle on better venture choices and manage demand. Capital expenditure and Operational expenditure indicate the amount of funds required to purchase the physical assets and the cost incurred for making them operational, respectively. Along with other metrics, such as carbon credit and Return On Investment, these assist in the development of an effective business case for data center modernization. An overview of the financial impact metrics is presented in Table 3.14 where the unit of each metric is listed including the objective, optimal value, and the category to which it belongs.

Table 3.14: An overview of financial impact metrics.

Acro.	**Unit**	**Objective**	**Optimal**	**Category**	**Ref.**
A	Ratio	Maximize	1.0	Facility	[Wib08]
BVCI	Dollars	Maximize	∞	Facility	[VPDS11]
CapEx	Dollars	N/A	N/A	Facility	[APP15]
CCr	Tons	Maximize	∞	Facility	[Sch09]
λ_{faults}	Faults/Hour	Minimize	0.0	Facility	[Wib08]
MTBF	Hours	Maximize	∞	Facility	[TA04]
MTTF	Hours	Maximize	∞	Storage	[Wib08]
MTTR	Hours	Minimize	0.0	Facility	[Wib08]
OpEx	Dollars	Minimize	0.0	Facility	[APP15]
ROI	Ratio	Maximize	∞	Facility	[VD13]
TCO	Dollars	N/A	N/A	Facility	[Ras11]

We analyze the relationships between the financial metrics and present them in Figure 3.6. This figure shows that the Total Cost of Ownership (TCO), is calculated as a sum of Capital Expenditure (CapEx) and Operational Expenditure (OpEx) of the data center. Component failure rate (λ) is calculated using Mean Time Between Failures (MTBF). And Availability (A) is calculated using both MTBF and Mean Time To Repair (MTTR).

3.2.2 Open Issues and Research Challenges

There are a multitude of metrics to measure and monitor different aspects of data centers. We have identified a number of open issues and challenges in each of the dimensions. By applying a well-defined set of metrics which measure energy consumption and environmental impact during data center operation, and while making choices at various levels, it is possible for

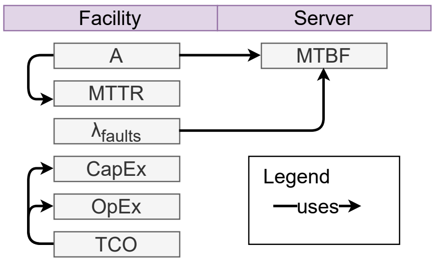


Figure 3.6: Relationship between financial metrics.

data centers to be planned, designed, implemented and operated in an energy-aware and eco-friendly manner. In Tables 3.15 and 3.16, we make a distinction between independent and dependent metrics. Independent metrics do not depend on other metrics; while dependent metrics do depend on other metrics. What follows next is a summary of the open issues we have identified.

When looking at the relationship between the metrics and challenges associated with using them, it becomes apparent that there is no single metric which covers all dimensions of the data center's performance. Even per dimension, there are several metrics promising to provide insights into the same area, through similar or different methods. However, none of these metrics are designed for the purpose of comparing data centers. In practice however, it is the PUE metric that is used for this purpose, despite the fact that it was never intended to be used as a comparison metric [BRPC07]. Instead, the metric was envisioned to be an internal measurement to steer an individual data center towards higher levels of efficiency, by knowing which areas have a low efficiency in terms of energy consumption. There are several problems when using PUE for the purpose of comparing data centers. For example, the IT load of a data center influences the PUE significantly. Furthermore, the PUE is also influenced by the weather and the geographical location of the data center. Therefore, comparisons between data centers using PUE are most often not representative of the actual situation. We see that there is a need for a metric which is designed with comparison in mind from its inception. Ideally, this metric should attempt to normalize the data in such a way that a fair comparison between data centers can be performed. The metric should take into account the overall utilization of data centers, as well as the location and weather.

Table 3.15: Metrics categorization into independent and dependent, part 1.

Independent Metrics	**Dependent Metrics**
Energy Efficiency	
APC, CADE, DCA, DCeP, DCLD, DCPD, DC-FVER, DH-UE, DH-UR, EES, EWR, GEC, H-POM, ITEE, OSWE, PEsavings, PUE, PpW, ScE, SI-POM, TUE.	CPE, DCcE, DCiE, DCPE, DPPE, DWPE, ITEU, PDE, pPUE, PUE$_{scalability}$, SPUE, SWaP.
Cooling	
AEUF, CoP, DCCSE, DCSSF, EER, RI, WEUF.	HSE.
Greenness	
CO_2S, EDE, ERF, GUF, MRR, TGI.	CUE, ERE, GEC, ω_{energy}, TCE, WUE.
Performance	
DCP, DR, EP, FpW, IPR, LD, LDR, PG, U$_{CPU}$, U$_{DC}$, U$_{server}$, UPS$_{CF}$, UPS$_{EE}$, UPS$_{PF}$, UPS$_{PFC}$, UPS$_{SF}$.	ACE, DEEPI, SWaP.
Thermal and Air Management	
AE, CI, DC, DP, HF, IoTemp, D^2, M_x, RCI, RH, RHI, RTI, SHI, β_{index}.	BPR, BR, NPR, RR.
Network	
BJC, CNEE, ECR-VL, NT$_{kwh}$, RS$_{max}$, TEER, U$_{network}$.	DS, NPUE, PS.

Another challenge is the difficulty of capturing the energy efficiency for all of the possible parameters of the data center using a single metric. This can be partially addressed by extending existing metrics, for example Corporate Average Data Center Efficiency (CADE) metric can be extended by considering how efficiently servers, storage, and network equipment are utilized.

Table 3.16: Metrics categorization into independent and dependent, part 2.

Independent Metrics	Dependent Metrics
Storage	
CEE, OSE, RT, SU, TP$_{i/o}$, U$_{mem}$, U$_{storage}$.	LSP.
Security	
ACPR, ATR, CC, CER, CTR, DeD, DeP, DTE, FC, FL, HTR, RA, RC, RCD, T, TP$_{IP}$.	AS, IAS, IPFH, ITH.
Financial Impact	
BVCI, CapEx, MTBF, MTTF, MTTR, OpEx, ROI.	A, CCr, TCO, λ_{faults}.

Data Center Infrastructure Efficiency (DCiE) metric is effective at discovering initial efficiency problems and helps justify the need to implement energy saving changes. However, the DCiE metric varies for each data center as it depends on the IT electrical load, which is a variable and is a site specific function of the IT hardware, software, architecture, load and efficiency. Due to this variability, we can not predict the impact of changes to the data center using DCiE. The Green Index (TGI) metric allows for flexibility in green benchmarking as it can be used and viewed in different ways by its end-users. Even though we have specified the use of the performance-per-watt metric for computing the TGI, it can also be computed with any other energy efficiency metric. TGI does not consider the power consumed outside of the IT equipment context. In case other components need to be included, the metric can be extended. Overhead metrics such as IT Hardware Power Overhead Multiplier (H-POM), Site Infrastructure Power Overhead Multiplier (SI-POM) give an understanding of a data center's energy use considering variations in IT equipment energy consumption and the facility energy consumption. These metrics provide useful insights to the data center operators regarding the efficiency of the different components of a data center, from cooling to electrical losses. However, to accurate evaluate all of these metrics requires in-detail monitoring of the data center. We foresee that IoT plays an important role in achieving the required level of detail with regards to monitoring.

The number of inter-dependencies between metrics is large. It is important to be aware of these relationships, as metrics can have certain limitations that affect other metrics associated with them. When combining existing metrics into new ones, or basing new metrics on existing ones, the flaws of the existing metrics are usually not overcome, and are sometimes even magnified. Therefore, it is useful to understand these shortcomings and know what a metric can and cannot measure. Applying metrics is even more difficult for co-location data centers as the equipment, space and bandwidth are available for rental in these types of data centers. Co-location data centers face an additional challenge, as metrics that define useful work are not usable by co-location centers. This is due to the fact that the facility has no access to the servers, and therefore cannot monitor the load. Even more crucially, the operators cannot define useful work on servers which are not their own. In the next chapter, we will investigate a method to enable monitoring in co-location data centers. We summarize our findings per dimension with respect to data center metrics:

Energy efficiency metrics measure the compute and non-compute energy usage of a data center. These metrics measure the efficiency at various levels of granularity starting from operating system to data center. But it is difficult to measure the energy consumption at operating system level. Also, it is challenging to measure the energy consumption at sub-component level of a data center, as these low level measurements are often not available.

Cooling metrics are used to specify the performance of the Computer Room Air Conditioning (CRAC) / Heating, Ventilation and Air Conditioning (HVAC) units and proper sizing of the cooling units. These metrics also measure the efficiency of the cooling systems. Estimating power and cooling capacity requirements using the ratings found in the specifications of IT equipment may not be accurate. Another issue is that heat densities change within racks, and also differ from one rack to the next.

Green metrics measure the environmental impact of a data center and its components. They highlight the importance of green energy and measure the efficiency of recycling and reuse in a data center. Efficient measurement of these metrics requires capturing regional and seasonal changes, which also enables comparisons with different data centers.

Performance metrics can assist the data center with increasing its productivity. These metrics help to measure IT performance and productivity of the data center and also identify problem areas. Metrics can range from low level UPS performance to high level data center utilization. Across all the components, a single fault may affect many other systems and ultimately decrease the overall performance of the data center. Operators rely on name-plate capacities and modeled load which do not accurately represent the actual capacity requirements. It is challenging to understand in real-time the impact of changes that are made.

Thermal and Air Management metrics monitor environmental conditions inside the data center. These metrics give an overview of how efficiently air flows within a data center and also quantify the extent of cold and hot air mixing. Continuous monitoring of these metrics allows the operators to reduce fan speed and increase cooling set points in real-time, which increases cooling efficiency and energy savings. It is difficult to determine the correct values for temperature and humidity in the data center, as the environment is dynamic and constantly changing.

Network metrics cover the network energy efficiency, network utilization and traffic demands of a data center. Networking equipment is responsible for a large portion of a data center's energy consumption, therefore it is important to optimize the efficiency of the networking equipment.

Security metrics cover aspects such as the firewall performance. These metrics are highly dependent on internal governance, compliance standards and SLAs of the data center in question. Another issue is authorization: the visibility of resources and the control over resources in a data center.

Storage metrics capture the performance of storage operations. These metrics assist the operators in reducing storage cost, improving storage utilization, and increasing the overall storage performance. The distributed nature of cloud computing makes it critical to learn what workloads customers are accessing and the level of importance of the accessed data.

Financial Impact metrics help achieve a data center's financial and strategic objectives. These metrics range from total cost of ownership to return on investments. Measuring business value may vary from one organization to another due to different definitions, and Carbon Credit may vary based on a country's policies.

3.2.3 Conclusions

Metrics are important for planning, designing, building and operating a data center in an efficient manner. When implementing best practices, metrics are a critical tool in understanding the effects of these practices. Our classification of metrics provides deep insights into the state-of-the-art of measuring different data center components. Our study on the most adopted and representative metrics currently in use throughout the data center industry revealed that the use of these metrics is critical to enable monitoring the data center efficiency in a timely manner, aiming to minimize energy consumption and total cost of ownership. Our proposed classification allows for quick access to the right subset of metrics from a huge collection that fits the desired context.

Evaluation of the metrics can be performed either by using manually collected data or data automatically gathered from sensors. We foresee that IoT has an important role to play in the automatic gathering of this data. Energy efficiency, performance, network and storage metrics can be used to increase the operational efficiency. So-called green metrics can be used to decrease the environmental impact of data centers. However, in all cases, it is crucial to use accurate data as input. This reinforces the importance of data center monitoring, in particular real-time monitoring of the different sub-components. We observed that existing metrics are mainly focused on measuring the energy efficiency of IT equipment or facilities. Older facilities may not be able to capture the raw data that feeds today's more sophisticated metrics. There are very few metrics defined which can integrate different components of the data center that have a single numerical value to report the efficiency of the data center in all perspectives. Also, there is no metric which reflects the changes made to a data center and its sub-components. Furthermore, there is a need for new metrics that consider different factors such as the geographical location and age of the data center, in order to allow world-wide comparisons across different data centers.

As there is a wide range of different metrics available for data centers it would be beneficial if there was an automated process to collect, process and analyze the data and use it to automatically calculate all available metrics. Such an automated process can take advantage of the Internet of Things philosophy by connecting numerous sensors together to create one platform. It can also maintain the history of sensor data and provide different types of analytics on top. Such a platform can potentially discover new correlations between data sets. The data can also be used to decide whether the existing technology and equipment can be used more efficiently, for example using improved scheduling algorithms, or whether it is better to replace them with the latest, most efficient technology or equipment. Data collection in co-location data centers is especially challenging, as there is no access to the IT equipment that is not owned by the data center.

Finally, it is important to note that aside from the IT equipment, the main contributor to a data center's energy footprint is the cooling system. So when it comes to sustainability, the cooling-related metrics play an important role in determining the effect of changes made in the cooling system or policies of the data center. An even more impactful approach to increasing the cooling efficiency is migrating away from air-based cooling, and instead explore alternatives such as liquid cooling, or even immersion cooling.

3.3 On Immersion Cooling and Sustainability

Data centers transform energy into useful work, where useful work is typically some computational load. A significant amount of heat is generated as the IT equipment consumes energy and performs these computations as almost all of the supplied electrical energy is converted into heat. To ensure the high level of availability and reliability as typically defined in the data center's SLA, the produced heat needs to be removed and the temperature and humidity should be maintained in a certain range. A cooling system needs to be in place to handle and remove the large amount of heat produced by the IT equipment. The cooling system is one of the main energy consumers in a data center.

Presently, air cooling is the most prominent technique used in data centers, and is responsible for approximately 40% of the data center's total energy consumption [NB17]. In an air-cooled data center, cold air is circulating through the perforated tiles up and into the front of the servers and hot air is pushed out by new cold air coming into the servers. By this method, it is possible to cool server racks with at most 50kW power density [KG16]. In 2018, only 10% of respondents to a survey reported that the power density of some of their racks was above 40kW [Smo19]. However, as more workloads depend on GPUs, the power density also increases. Moreover, recent studies show that as the end of Dennard scaling is reached [DGY+74], and transistor sizes approach their practical limitations, new cooling solutions are needed to maintain the traditional performance improvement trend. Therefore, CPUs and GPUs with higher power consumption are expected to be manufactured in near future [FWKT18; SADK19], and consequently, the power density of racks will increase as well. We now shift our focus towards an alternative to air cooling, namely immersion cooling, and investigate its benefits and drawbacks.

In immersion cooling, IT components are fully immersed in a dielectric fluid that conducts heat and does not conduct electricity, therefore, the heat of all components is fully transferred to the liquid, which reduces the PUE of the data center as the liquid has a better heat transfer coefficient than air.

In fact, the PUE of immersion-cooled data centers is close to perfection, at about 1.02-1.04 [AAH+18; CGB17; EFV+14; MMK17; Sha18], which shows that these centers consume 10-50% less energy compared to their air-cooled counterparts for the same amount of computational load. When compared to the air, common dielectric fluids have a much higher heat capacity. Therefore, immersion cooling allows for more computing power in less space. While the maximum power density per rack for an air-cooled data center is around 50 kW, immersion-cooled counterpart allows for up to 250 kW per rack [Bit; KG16].

Immersion cooling is certainly an efficient solution in terms of computing efficiency and power density. In addition, the capital expenditure for constructing a data center, assuming a constant power density, is lower with immersion cooling compared to air cooling [BTA20]. Despite the benefits, the topics of maintenance and reliability, and specifically the lack of practical information on these aspects, are the primary concerns for the adoption of immersion cooling technologies [Ali18; CH16b; JMG+21; RRC+19; Var19; Vil20]. Maintenance is challenging due to an increased number of IT equipment failures, liquid leakage, and liquid evaporation, which also impose additional operational costs. However, there are several successful implementations of immersion cooling in data centers. For example, one of the biggest players in cloud computing, Microsoft, has already constructed its first immersion-cooled data center [Wes21].

We explore quantitatively the trade-offs between air and immersion cooling technologies and evaluate several aspects of both methods. For this evaluation, we refer to various references, from research studies to practical implementations. Additionally, quantitative analyses on data center efficiency and computing power are presented. This will help data center operators to have a better perspective while selecting the best solution for their specific application. Our investigations show that immersion cooling has significant advantages specifically for high power applications and we expect its adoption to grow. Improving the efficiency of a big data center by a small percentage would considerably impact total energy consumption. In addition, the migration from air to liquid cooling could influence operators of smaller data centers in trusting immersion cooling. However, we also argue that retrofitting an air-cooled data center with liquid cooling is more expensive and is generally not recommended.

3.3.1 Immersion Cooling in Practice

While air cooling is the dominant solution in data centers, several large companies are adopting the immersion cooling technology and a number of startups have appeared offering innovative immersion cooling systems. *Microsoft* has announced its first liquid immersion-cooled data center in Washington, USA [Wes21]. which is used for cloud-based communication platforms such as Microsoft Teams. *Alibaba* also uses Single-Phase Immersion Cooling (1PIC) tanks in its data centers. They have shown that immersion cooling reduces the total power consumption by 36% and helps to achieve a PUE of 1.07 [Zho19]. Another example comes from the *BitFury* group that built a 40+ MW data center that comprises 160 tanks, achieving a PUE of 1.02 using Two-Phase Immersion Cooling (2PIC) [Bit].

Furthermore, some companies are offering immersion-cooled server systems. *Asperitas*, for example, is a Dutch company located in Amsterdam that offers complete liquid immersion cooling solutions to its customers. Their AIC24 server enclosures utilizes single-phase immersion cooling and natural convection, avoiding the use of any mechanical parts such as pumps. The immersion-cooled servers of *Asperitas* are insulated in order to capture all heat produced by the servers in the fluid and to allow for maximum waste heat re-utilization. Each of their enclosures can contain up to 48 servers or 288 GPUs with a footprint of only 60cm x 120cm[1].

An alternative interesting approach was that proposed by the Dutch company *Nerdalize*. The idea was to offer a distributed data center by displacing servers in the residential buildings [NSCT18]. The immersion cooled servers would exchange heat with water that was then used for indoor heating and hot tab water. When the energy savings of heating the water are taken into account, the PUE of such a system would be less than 1.0. The company

[1]Asperitas – Immersion Cooling solutions for data centers, `https://asperitas.com/`

deployed several servers before bankruptcy in 2018. Interestingly, the company which restarted *Nerdalize*, *LeafCloud*, decided to opt for air cooling, mostly due to its lower maintenance costs and failure rates.

Another company offering liquid immersion cooling enclosures in various sizes is *Submer*[1]. All of *Submer*'s products use single-phase immersion cooling and are ranging from cabinet-sized enclosures called microPod, up to megaPod, which are set up inside shipping containers. The microPods are capable of cooling 5 kW of components even in direct sunlight which makes them suitable for companies who want to cool their in-house equipment efficiently. On the other hand, megaPods are targeted for higher computing powers. They can be put in almost any place since there is only electricity and network connection needed. For example, it would be possible to install a megaPod onto or near a building which then supplies the building with heat. Similar to *Nerdalize* and *LeafCloud*, their products are excellent for waste heat re-utilization.

We will now compare air cooling and immersion cooling solutions on several dimensions including computing efficiency, computing density, power density, cost, and maintenance. From the perspective of a DCO, there is a clear link between each dimension and the profit margin.

3.3.2 Computing Efficiency

The PUE is used by data center professionals to determine the energy efficiency of their facility [AAFP12]. It is the most popular metric for measuring the energy efficiency of a data center. Several studies report on the PUE of data centers with specific installations of immersion and air cooling solutions, these are shown in Table 3.17. For air-cooled data centers, reported PUE values range from 1.1 to 2.9 [MMK17; McN13; Mil14; Sha18]. Values close to 1.1 can only be achieved by hyper-scale data center facilities, which are especially optimized for efficient cooling. For example, Google's state of the art air-cooled data centers have the PUE of 1.12 [Mil14]. This means that in these data centers, 89% of the total energy is consumed by IT equipment.

[1] Submer – Data centers that make sense, `https://submer.com/`

Table 3.17: Reported PUE values for air and immersion cooling.

Reference	PUE Air-cooled	PUE Immersion-cooled
[MMK17]	1.1	<1.04
[AAH+18]		1.02
[EFV+14]		1.03 - 1.17
[CGB17]		1.02 - 1.03
[Sha18]	1.7 - 2.9	1.02 - 1.03
[Mil14]	1.12, 1.18	
[McN13]	2.2 - 2.61	

However, worldwide there are only a small number of high efficient data centers with state of the art designs [Jon18]. In addition, average air-cooled data centers have a much higher PUE compared to the most efficient ones. The average PUE has been reported to be between 2.2 and 2.61 for data centers in Singapore, Japan, Hong Kong, and Australia. This shows a significant difference between state of the art hyper-scale and average data centers [McN13].

Studies on immersion cooling data centers have reported consistently better results. For these studies, the PUE falls in the range 1.02 to 1.04 [AAH+18; CGB17; EFV+14; MMK17; Sha18]. The PUE of 1.02 appears to be the sweet spot for immersion cooling. Table 3.17 shows how the various studies agree on the fact that PUE values around 1.02 are achievable with immersion cooling. With a PUE of 1.02, about 98% of the energy consumed by the data center goes to the IT equipment. This is close to perfect efficiency. The maximum reported PUE for immersion-cooled data centers is 1.17 [EFV+14], which was achieved in an experiment for the maximization of cooling capacity without regards for efficiency.

As we mentioned in Section 3.2, the PUE metric is useful for evaluating the efficiency of a data center over time, but it is not suitable for comparing data centers. While PUE offers a reasonable indicator of data center efficiency, it is also desirable to have a value representing the compute capacity in relation to energy consumption. Computing performance is traditionally measured by running a benchmark on the IT equipment and measuring the completion time. An example of such a performance benchmark is measuring the number of Floating Point Operations Per Second (FLOPS) versus the power consumption [BC10]:

$$\begin{aligned} FpW &= \frac{Floating\ Point\ Operations\ /\ Second}{Joules\ /\ Second} \\ &= Floating\ Point\ Operations\ /\ Joule \end{aligned} \tag{3.2}$$

The FLOPS per Watt (FpW) metric only considers the power consumption of the IT equipment. This means changes in the cooling systems are not reflected in the metric because changing the cooling system does not influence the computing performance and the IT equipment's power consumption. Typically, when modifying the cooling systems, the IT energy remains constant while the total energy changes. To calculate the improvement in FpW by changing the cooling method, the knowledge of PUE and FpW can be combined:

$$\eta_{\text{data center}} = \frac{\frac{b_{ops}}{b_{time}}}{\text{IT power}} \cdot \frac{1}{\text{PUE}} = c \cdot \frac{1}{\text{PUE}} \tag{3.3}$$

where b_{ops} and b_{time} represent the number of benchmark operations and benchmark time, respectively. The inverse of PUE represents the part of the energy that has been used by IT equipment. For example, a data center with a PUE of 1.5 uses two-third of its electricity for IT equipment. As the cooling method is changed, the values for b_{ops}, b_{time}, and IT power remain constant, indicated by c. In other words, the overall computing efficiency depends directly on the fraction of power used for IT equipment. The expected improvement in the data center's overall computing efficiency by switching from air cooling to immersion cooling can be calculated by comparing $\frac{1}{\text{PUE}}$ values. For example, when migrating from best practice air cooling (PUE=1.12) to immersion cooling (PUE=1.02), the overall computing efficiency is increased by 9.8%. While this is a significant increase that reduces the operational

Table 3.18: Efficiency changes by migrating between cooling techniques.

To From	**Standard air cooling**	**State of the art air cooling**	**Immersion cooling**
Standard air cooling	-	78.6%	96.1%
State of the art air cooling	-44%	-	9.8%
Immersion cooling	-49%	-8.9%	-

expenditures, migrating from standard air cooling (PUE=2) to immersion cooling (PUE=1.02) increases the computing efficiency by 96.1%, reducing the energy consumption by half. The increase and decrease in computing efficiency when changing cooling method is shown in Table 3.18.

Let us present a numerical example based on one of the most powerful server CPUs currently on the market, the EPYC 7742 as manufactured by Advanced Micro Devices, Inc. (AMD). This processor has a peak power consumption of 225 watts and achieves approximately 3.48 teraFLOPS [Tra19]. This equates to 15.5 gigaFLOPS per watt:

$$\eta_{\text{EPYC7742}} = \frac{3.48 \text{ teraFLOPS}}{225\text{W}} = 15.5 \text{ (gigaFLOPS/W)} \tag{3.4}$$

The CPU is not the only power consuming component of a server. Therefore, to calculate the server computing efficiency, one needs to know the proportion of power consumed by the processor in relation to the server's total power. The computing efficiency of a server is calculated as follows:

$$\eta_{\text{server}} = \eta_{\text{processor}} \cdot p_{\text{processor}} \tag{3.5}$$

where $p_{\text{processor}}$ stands for the proportion of power consumed by the processor in relation to the total server consumption. Various works have reported different power breakdowns for the components of a server. In accordance with the results presented in [GB18], we assume that 50% of the server's power is consumed by its processor. From Equation 3.5, the computing efficiency of the server amounts to 7.73 gigaFLOPS/W. To calculate the computing efficiency of the IT equipment, we consider the power of all IT equipment including storage and network facility. In this way, the computing efficiency of IT equipment is calculated as:

$$\eta_{\text{IT equipment}} = \eta_{\text{server}} \cdot p_{\text{server}} \tag{3.6}$$

where p_{server} stands for the proportion of the server's power in relation to the total IT power. We use p_{server} in Equation 3.6 to take into account the energy consumption of non-computational IT equipment such as storage and networking. The contribution of the servers, storage, and network facility in the total power consumption of a data center has been reported in several works [DWF16; SSS+16]. For our numerical example, we assume the average of reported values, that is 77%. Therefore, $\eta_{\text{IT equipment}}$ is 5.95 gigaFLOPS/W. Finally, the computing efficiency of the data center is calculated by:

$$\eta_{\text{data center}} = \eta_{\text{IT equipment}} \cdot \frac{1}{\text{PUE}} \tag{3.7}$$

As per Equation 3.7, the computing efficiency of a data center cooled by standard air cooling, state of the art air cooling, and immersion cooling is calculated as 2.98, 5.32, and 5.84, respectively. The computing efficiency of an immersion-cooled data center is almost 9% higher than a state of the art air-cooled hyperscale data center. For the average data center, the switch to immersion cooling offers even more improvement. Operators can decrease the power consumption by about 50% without any decrease in computing performance.

3.3.3 Computing and Power Density

Computing density is defined as the amount of computation that a system can offer in relation to its physical size. The metric used to determine the computing density of a data center is $FLOPS/m^3$. This metric considers all aspects of the data center: from the servers to the power management system and up to the cooling equipment. Comparing air-cooled and immersion-cooled in these terms also shows significant differences as the required space for immersion cooling has been reported to be about one third of traditional air cooling method [MMK17].

The increase in computing density offered by immersion cooling comes from various factors. In immersion cooling, there is no space needed in between the racks for airflow. Tubs can be placed right next to each other and the only limitation is the required accessibility for maintenance personnel. In addition, there is significantly less cooling equipment needed, and no raised floors or air vents are required [MMK17]. Unlike air cooling, immersion cooling does require pipes or flexible tubes for the transportation of liquids or gasses. Using immersion cooling, the density increases at both the rack and facility levels [Tum10b]. Furthermore, many air-cooled data centers trade their computing density for efficiency and reliability [Mil14]. The cooling units of these centers are often bigger than they need to be.

Power density is another important factor for data center operators. Power density is low in air-cooled data centers, otherwise, either fan speeds need to be increased or air temperature needs to be lowered, resulting in a decreased energy efficiency. The typical rack-level power density of air-cooled data centers is about 0.018 kW/l - 0.028 kW/l [KG16], while the density of immersion-cooled data centers is between 0.045 kW/l [MMK17] and 0.23 kW/l [GBR+14]. At the extreme, 4 kW/l has been reported to be possible in immersion-cooled data centers with enough coolant flow and compact hardware design [Tum10a; Tum10b]. Table 3.19 presents the data gathered from various references on the power density of air-cooled and immersion-cooled data centers. Since power densities are presented in kW per rack for air-cooled data centers and for immersion cooling there is no typical rack size, the values need to be converted to kW/l.The power density with immersion cooling is almost six times higher than air cooling. Therefore, power density can be another strong motivation for DCOs to consider immersion cooling for their future projects.

Table 3.19: Power densities as reported in the literature.

Reference	**Air cooling density**	**Immersion cooling density**	**Normalized to kW/l**
[MMK17]		14 kW/bathtub	0.045
[GBR+14]		400 W/bathtub	0.23
[AAH+18]		250 kW/rack	0.14
[Tum10b]		4 kW/l	4
[KG16]	33-50 kW/rack		0.018-0.028
[Smo19]	40 kW/rack		0.022

3.3.4 Cost

Important factors to consider when adopting immersion cooling are the CapEx and OpEx for the data center. The CapEx includes the sealed chassis for immersing the IT equipment, dielectric fluids, as well as pumps and tubing. Similar to air-cooled data centers, the OpEx includes electricity, staff, network connection fees, as well as supporting and maintaining the IT equipment.

According to Bunger et al.[BTA20], for air-cooled data centers with a power density of 10 kW per rack, the capital expenditures are $7.02 per watt. For a liquid-cooled data center with a similar power density, the cost is reduced slightly to $6.98 per watt. The benefit of liquid cooling is also that much higher power densities can be achieved. Assuming a power density of 40 kW per rack, the expenditures are further reduced to $6.02 per Watt. Furthermore, when it comes to the operational expenditures, a reduction between 9-20% is expected with regards to the energy cost due to the absence of fans. This is in agreement with the results of Neudorfer et al. who state that a 5-10% reduction of the IT energy consumption is expected due to the avoidance of internal fans [NEKZ16]. The need for fans is also absent for the entire facility, reducing the total energy costs by 15 to 25%. Another point to consider is that the dielectric fluids present in the sealed chassis can be used virtually indefinitely, assuming some form of filtration is present.

Day et al. highlight that when building a new data center and optimizing it for liquid cooling from the ground up, capital expenditure savings can be achieved over air-cooled data centers [DLB19]. On the contrary, retrofitting an air-cooled data center with liquid cooling can result in higher costs. An important exception is retrofitting an air-cooled data center with limited floor space and power capacity. In this case, the increased power density possible with liquid cooling, as well as a reduction in energy consumption, can address both the space and power limitations with one solution.

3.3.5 System Maintenance

The operational costs of a data center includes system maintenance. The cooling method influences the components' environmental conditions and consequently affects the number of and time to failures and overall equipment lifetime. The number of maintenance requests caused by failures has been reported to be almost 6.6% higher for an immersion-cooled data center compared to a traditional air-cooled counterpart [CH16b]. The higher number of maintenance requests associated with immersion cooling results in additional operating costs. In addition, higher failure rates can degrade the components' lifetime, but immersion cooling can compensate for this degradation with lower junction temperatures [JMG+21]. Besides the number of maintenance requests, the maintenance procedure is more challenging with immersion cooling as immersed IT equipment is removed by opening the lid and lifting the equipment out of the tank. This can result in liquid evaporation and spillage.

According to the work presented in [Vil20] and [Ali18], the maintenance overheads and reliability concerns, as well as the leaks and spills, are the top contributors to the low adoption of immersion cooling. Indeed, the enclosure that the racks are immersed in must be sealed perfectly to avoid liquid evaporation/losses. Complete enclosure sealing would mitigate the problem but is not practical. The high number of maintenance requests, compared to the air cooling system, cause access issues and adds operating costs related to compensating the fluid losses [Var19].

The operating costs imposed by the liquid loss have been evaluated in [CH16b]. The cost of the lost liquid divided by the cost of the IT equipment's energy usage has been reported to be 4.68 for a specific implementation. This number shows that the maintenance overhead is a significant drawback associated with immersion cooling. It can even mitigate the energy efficiency improvement of immersion cooling. However, it should be noted that the implementation characteristics including the liquid price and the electricity costs affect the reported value. In [CH16b], these values are set at 75 $\$/liter$ and 0.09 $\$/kWh$, respectively. One-phase immersion cooling is reported to have fewer maintenance needs than two-phase immersion cooling [Var19].

3.3.6 Conclusions

Air cooling is the traditional solution for dissipating heat in data centers. A high energy consumption and low cooling capacity, and consequently limited power density, are the main challenges associated with air cooling. Immersion cooling is emerging as a novel method with many advantages in terms of efficiency, density, and cost. We provide a quantitative comparison of these two approaches and provide an overview of the results presented in the literature. While most data centers around the world depend on air cooling, immersion cooling is recognized as a potential alternative and several cloud providers have already constructed their immersion-cooled data centers. The key findings related to immersion cooling are listed next.

First, based on the PUEs reported in the literature, we conclude that the immersion cooling method consumes less energy and offers a higher computing efficiency. Our analysis shows that a typical immersion-cooled data center consumes almost 50% less energy compared to its air-cooled counterpart. Second, immersion-cooled data centers allow for compact designs, more than three times the density of their air-cooled counterparts. In liquid immersion-cooled data centers, there is no trade-off between efficiency and density. Conversely, air-cooled data centers can only be either-or. Immersion-cooled data centers can be placed in ordinary spaces as they have lower requirements and require less additional cooling infrastructure. Third, while research related to immersion cooling is mostly targeting efficiency, the aspect of power density should not be overlooked. The increase of computing power in a specific volume is even more important than the efficiency improvement. The most conservative figures for immersion cooling are about double density in kW/l on rack-level compared to the maximum possible in air-cooled data centers. Double the density means less than half of the physical space is required to achieve the same compute performance, addressing issues seen in the Netherlands where the physical space is limited. In the extreme case, densities of 4kW/l are possible with IT equipment optimized for liquid immersion cooling. This high-density capability makes immersion cooling the first-choice solution for running high-performance workloads. In addition, it allows manufacturing the CPUs and GPUs with higher frequency and higher power consumption. Fourth, the capital expenditure for an air-cooled data center with a power density of 10 kW per rack is about 4% higher compared to its immersion-cooled counterpart. In addition, a 9-20% operational expenditures reduction is expected with immersion cooling. Fifth and final, the main downside of immersion cooling is the challenges related to the maintenance and reliability concerns. This is a reasonable explanation for why even with significant advantages in efficiency and density, not many companies have switched their cooling solution to immersion cooling, yet.

As we approach the limits of Moore's law and Dennard scaling, the increase in overall performance per watt slows down. If we want to maintain the same rate of increase for computer performance, more efficient cooling techniques are required to promote the sustainability of data centers. Immersion cooling appears to be a solution that should be strongly considered when designing a sustainable data center and we expect its adoption to grow. At the same time, retrofitting any air-cooled data center with liquid cooling does not seem convenient in most cases, if feasible at all. Considering the reliability and maintenance challenges and uncertainties, immersion cooling might not be useful for small and average power densities. On the contrary, immersion cooling appears to be the best solution for high power densities.

3.4 Summary

In this chapter, we have focused on several important aspects of sustainable data centers. First, we identified 23 best practices across 4 different dimensions and subjected 7 data centers to these best practices by means of questionnaires and interviews to the relevant stakeholders. It is clear that not all best practices are implemented, and that the implementation of greenness practices was lacking. We also identified the need for metrics to monitor the impact of best practices. As there is a vast variety of available metrics, we created a taxonomy of metrics covering 9 dimensions. For each dimension we analyzed potential issues and open challenges. From the analysis we conclude that real-time data center monitoring is important, as well as the need for an IoT-based approach for large scale data collection. Additionally, we emphasize that co-location data centers will struggle with data collection from the IT equipment due to a lack of access. And finally, we investigated a more efficient technique for data cooling, namely immersion cooling. While there is no widespread adoption as of yet, it is clear that especially newly built data centers can benefit from using immersion cooling instead of traditional air cooling. We use the metrics from our taxonomy to perform a quantitative analysis of different cooling approaches. Our analysis shows that migrating to immersion cooling can reduce the energy consumption by up to 50% while increasing the maximum power density. In the next chapter, we dive deeper into the issues regarding real-time monitoring, data collection in co-location data center, and the role of IoT in this context.

Chapter 4

Monitoring of Smart Data Centers

Traditionally, data center operators rely on DCIM tools to maintain a high level overview of the data center. Monitoring the data centers is crucial for multiple reasons. First of all, it assists in improving the energy efficiency by discovering comatose or zombie servers. These comatose servers are performing no useful work, yet still consume energy. It is estimated that up to 30% of servers are comatose [NC17]. Monitoring is also critical in preventing outages, which can have a wide-spread global effect [ZB17]. Preventing outages is critical for upholding the Quality of Service (QoS) as is specified in the SLA. Furthermore, monitoring aids the expansion planning process of data centers by predicting future cooling and space requirements as the data center grows. IoT has an important role to play in transforming the contemporary data center into a smart data center, or smart building, by enabling detailed monitoring on a level that was not possible before. In a smart buildings, devices are typically connected to a central hub on the local network, known as an IoT hub. The IoT devices, together with the hub, provide monitoring and control of the physical spaces and instruments of the building, [LL15]. In a smart data center, the IoT hub could be responsible for monitoring the data center temperature and keeping it within a given range in a certain time of the day by controlling the cooling systems. Monitoring and control are the foundation for building automation and, in turn, the enabler of truly smart data centers. This automation also plays an important role in improving the energy efficiency and overall sustainability of buildings [EH12]. It is clear that the amalgamation of monitoring and IoT, applied to the context of data centers, has significant potential.

We investigate the potential of real-time monitoring using an IoT-based approach in Section 4.1. We also uncover some potential issues with monitoring on such a large scale, and turn towards edge computing for solutions. The need for an IoT hub becomes apparent, and an edge gateway architecture is proposed. Next, in Section 4.2, we analyze 20 different IoT hubs, of which four are selected for a more detailed analysis. The strengths and weaknesses are identified, and a generic IoT architecture emerges, which shares many features with the proposed edge gateway architecture. Then, in Section 4.3, we utilize IoT to collect data from a real-world co-location data center to allow privacy-preserving monitoring of servers not owned by the data center. This is one of the challenges uncovered in the previous chapter. And finally, in Section 4.4, a summary is given of the major points uncovered in this chapter.

4.1 Real-Time Monitoring of Data Centers

Real-time monitoring is an approach, typically supported by the IoT paradigm, that enables the continuous streaming and analysis of vast quantities of data [MIT18]. By continuously analyzing the data stream, decision making becomes proactive rather than reactive. A concrete example of this phenomenon in the context of data centers is the detection and prevention of hard disk drive failures through continuously monitoring the data reported by the Self-Monitoring, Analysis and Reporting Technology (S.M.A.R.T.) system. Furthermore, the IoT paradigm enables monitoring of data centers at a scale that was not possible in the past. A wide variety of hardware and virtual sensors can be utilized to collect different types of data which, in turn, can be used to evaluate dozens of sustainability and performance metrics such as those presented in Chapter 3. As a result, the amount of data that can be collected in this environment is of massive proportions: a data center of 100 000 servers, each of which report 50 distinct metrics every second, would result in 300 000 000 data points every minute. Collecting data at such fine granularities enables the real-time monitoring of the data center in its entirety. However, if this data would be collected at the high frequencies required for real-time monitoring, a different problem arises: the quantity of transmitted data would be sufficiently large to negatively impact the data center's network infrastructure. This problem is also observed in real-time monitoring of smart grids [HLW+18]. We foresee that edge computing can play an important role in alleviating the pressure caused by large-scale data generation.

In this section, we investigate how we can leverage a data center's network infrastructure to efficiently monitor a data center in real-time by utilizing the edge computing paradigm. First, we analyze the common network architectures found in data centers. Next, we look at the potential data sources that can be found in a data center in order to determine the size of the raw data and the required network throughput. This is followed by a preliminary design of an edge-based data collection platform that takes advantage of a data center's network infrastructure to reduce the load on the network. Finally, we discuss the results we have obtained thus far.

4.1.1 Data Center Network Infrastructure

The IT equipment of a data center is placed in racks. This equipment, such as servers and switches, often occupies between 1U to 4U of space, with blade server enclosures consuming up to 10U of space. Efficiently connecting all the rack equipment to the network can be a challenge, and the design of the

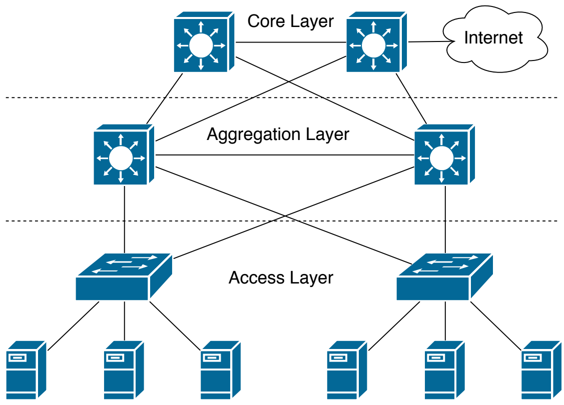


Figure 4.1: An example of a 3-layer data center network architecture.

data center network affects the networking efficiency at which the connected equipment operates. The most widely used network architecture in data centers is the *3-layer data center network architecture* [BCH13], shown in Figure 4.1. As the name suggests, this architecture consists of 3 distinct layers: a core layer at the top, an aggregation layer in the middle, and an access layer at the bottom. Equipment, such as servers, that requires network access is connected to the access layer, usually with 1 or 10 Gigabit links. The access layer is commonly implemented as a network switch located at the top of a rack, known as a Top of Rack (ToR) switch. Alternatively, the switch can be placed at the end of a row of racks, known as an End of Row (EoR) switch. The aggregation layer aggregates the different ToR and EoR switches, to enable network connectivity between racks or rows of racks. The links between ToR and EoR switches are commonly 10 or 40 Gigabit. The aggregation layer switches all connect to the core layer, these links can often be up to 100 Gigabit. The core layer is responsible for providing uplinks to the Internet.

There are also other network architectures currently in use in data center, such as Facebook's data center fabric approach [FA13]. This approach is similar to approaches taken by Google and eBay. The notion of a server pod is introduced, which is essentially a standalone cluster consisting of racks and servers, containing up to 48 ToR switches and 4 special fabric switches. These fabric switches are responsible for interconnecting the servers in a single pod. To connect different pods, a network spine is introduced consisting of up to 48 spine switches per spine plane. This approach is highly scalable, as computational resources can be increased by introducing more pods, and the network capacity can be increased by introducing more spine planes.

Another approach is the Fat Tree data center network [AFLV08]. This approach is similar in design to the 3-layer approach, but provides guarantees regarding the available bandwidth for each server in a rack. This is done by carefully planning the numbers of switches in each layer, and increasing the number of links between individual switches the higher up the hierarchy they are. Any horizontal slice in the network graph has the same amount of bandwidth available.

Despite the significant differences between the available data center network architectures, they all contain an access layer with ToR and EoR switches in one form or another. As we show later in our proposed architecture, these types of switches are excellent candidates to become edge gateways due to their proximity to the servers that are being monitored.

4.1.2 Impact on Network Load

To understand the significance of the additional load that is associated with real-time monitoring of a data center, a number of steps have to be taken. First, the number of servers per rack and the number of racks per data center have to be identified. Next, the data types that can be collected from a server have to be investigated, as well as their data size. And finally, the load on the network that is generated by real-time monitoring has to be calculated.

The number of servers that can be placed inside a rack is not only limited by the size of the servers, but also by the data center's cooling capacity and power limitations. A standard full height rack offers space for up to 40 servers, leaving 2U for other equipment. In practice this number is between 25 to 35 servers per rack. Using blade servers, the density of a rack can be increased much further. A typical high performance 10U blade server enclosure contains 16 servers. This increases the density to 64 servers per rack. There are also 3U blade servers enclosures for low performance blade servers that house 20 blade servers. This results in a maximum density of 260 servers per rack. In all cases we assume there is at least 2U left for the ToR switch and a Keyboard Video Mouse switch.

The largest data center in the world is China's Range International Information Group data center, covering over 500 000 square meters. More commonly, data centers are between 10 000 and 20 000 square meters in size. For example, Google's Dallas data center is 18 000 square meters and contains 9090 server racks [CGC16]. Applying the previously determined server density numbers, it can be extrapolated that a data center containing 9090 server racks can house anywhere between 318 000 and 2 363 400 servers. A report from Gartner estimates that Google had around 2.5 million servers in July 2016, spread across 13 data centers, which equates to around 192 000 servers per data center [ZKS+19].

There are two types of sensors required to monitor a data center: hardware sensors and virtual sensors. The hardware sensors are typically used to monitor the temperature and humidity, as well as power consumption. These measurements can be taken on a global level for the whole data center, but also on an individual server level. Virtual sensors are software-based sensors, such as agents interacting with the operating system to gather information about the CPU, memory, networking interfaces, storage devices, and more. There are software agents available that can collect and transmit this type of data, popular solutions include: Telegraf, StatsD, collectd, Zabbix, Prometheus, and Nagios. In our experiments, Telegraf is used to represent the virtual sensors, because of its popularity and its ability to integrate with a multitude of platforms. Telegraf is a plugin-based software solution for collecting and transmitting a wide variety of data. It consists of four plugin types: input plugins, processor plugins, aggregator plugins, and output plugins. Input plugins collect data from the system, processor plugins transform the data, aggregator plugins aggregate the data, and output plugins transmit the data to other systems. Only the input plugins that collect generic system information are included in our experiments, a full overview of all of the used plugins and their reported metrics is given in Table 4.1.

Table 4.1: Telegraf input plugins and the metrics they collect.

Input Plugin	**Metrics**
cpu	Time the CPU spends in different states (e.g. user time, system time).
disk	Storage device usage in bytes and percentages.
diskio	I/O statistics of reads and writes to storage devices.
kernel	Operating system statistics not covered by other plugins.
mem	Detailed information on Random Access Memory usage.
processes	Total number of processes and their status.
swap	Swap usage in bytes and percentages.
system	Load, uptime, and number of users logged in.
hddtemp	Temperature data from storage devices.
internal	Internal statistics of the Telegraf agent.
kernel_vmstat	Statistics regarding the virtual memory usage.
net	Network usage per network interface.
netstat	TCP and UDP connection state and count.
nstat	Fine-grained networking statistics.
ntpq	NTP query metrics and status.
sensors	Data from hardware sensors (e.g. chipset temperature).
smart	S.M.A.R.T. information.
temp	CPU-related system temperatures.

To measure the bandwidth required to monitor the generic metrics collected by Telegraf, experiments are performed using a real server. The server in question is a Dell PowerEdge R7425 with dual AMD EPYC 7551 32-core processors, 512 GB of Random Access Memory (RAM), and six 960 GB Intel S4510 Solid-state Drives (SSDs). The operating system is Proxmox, a Debian-based virtualization environment. Telegraf is installed on the operat-

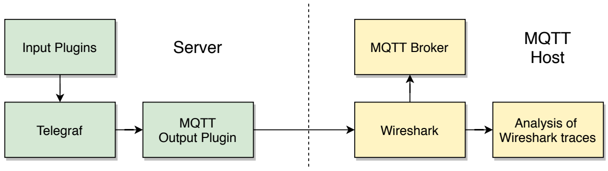


Figure 4.2: Setup to analyze the bandwidth usage of real-time monitoring.

ing system and configured to collect the selected metrics. Message Queue Telemetry Transport (MQTT), a lightweight publish-subscribe network protocol, is configured as the output plugin. An MQTT broker is deployed on a second host. Wireshark, a network packet analyzer, is also installed on this second host in order to monitor the network usage. The traces produced by Wireshark are analyzed to calculate the required bandwidth for real-time monitoring of a data center. An overview of the setup is shown in Figure 4.2.

To determine the load on the infrastructure, network packets were collected for a duration of 600 seconds. During this period, 185 400 messages were sent to the MQTT broker. In total, 55.3 megabytes of data were transmitted, an average of 92.2 kilobytes per second. While seemingly insignificant for one server, however when we extrapolate this and use Google's Dallas data center and a rack density of 25 servers per rack as an example, the total bandwidth would equal 25 servers per rack $\times$ 9090 racks $\times$ 92.2 kB/s = 167.62 Gbit/s. In practice this number is conservative, as the servers per rack density is ever increasing, and data centers are becoming ever larger.

4.1.3 Proposed Edge-based Architecture

One method to reduce the overall load on a data center's network is bringing the computations closer to the source of the data. This reduces the amount of hops required for the data to reach their destination, and in turn limits the load to the access layer instead of overloading the aggregation and core layers. Edge computing has an important role in achieving this reduction in networking load. The architecture we propose is shown in Figure 4.3. As each rack has a ToR switch, the goal is to leverage the computational power of the switch to turn it into an edge gateway. Every edge gateway is responsible for processing and analyzing the data of their rack only. Therefore, the edge gateway would only have to handle the network traffic of a limited amount of servers. The network load for the gateway ranges between 18 Mbit/s and 47 Mbit/s, for 25 servers and 64 servers per rack respectively. At these loads the impact on the switch itself is minimal.

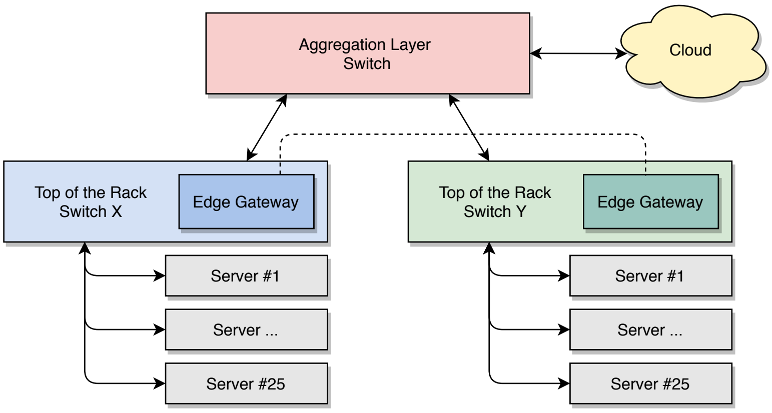


Figure 4.3: Proposed edge-based architecture using Top of the Rack switches.

Because edge gateways are close to the source of the data, the network latency is also greatly reduced. This is crucial for real-time monitoring, as the data center operator should be informed as soon as possible about critical events. The edge gateway can also be used to automatically interact with the servers. For example, when a server is overheating, the gateway could inform the server to reduce the load, or even lower the frequency at which the CPU cores are operating. This allows the edge gateways to act as autonomous agents. The proposed architecture also improves the scalability of the data center. As the data center grows and more racks are placed and filled with servers, the impact that monitoring these new servers has will be minimized as the majority of the data remains at the ToR switch. It also possible for multiple racks to be clustered together, such that the edge gateways of these racks communicate with each other in a peer-to-peer fashion. Another benefit of this approach concerns the privacy. In case a rack is dedicated to processing sensitive data, the edge gateway will ensure that monitoring data collected from these sensitive servers does not leave the rack. Or, when the data does have to be transmitted outside the rack, it is anonymized and privacy sensitive data is removed before it is sent across the network.

A schematic overview of a possible implementation of an edge gateway is shown in Figure 4.4. The hardware sensors and virtual sensors generate data, which is then transmitted to an event queue to facilitate real-time streaming data. Some data streams can be monitored directly, whereas other may have to be processed and aggregated first. Based on the events that are happening, alerts can be sent out. The processed data can then be sent to the cloud, or broadcasted to neighboring edge gateways. There are clear similarities to IoT hubs and gateways, as we will see in Section 4.2.

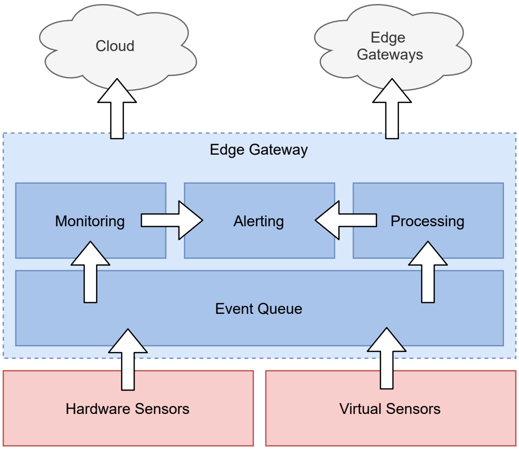


Figure 4.4: A schematic view of the edge gateway.

Using edge computing instead of traditional cloud computing to perform real-time monitoring in data centers has a number of benefits. From reducing the network load, to increasing the responsiveness, enabling autonomous control, as well as improved scalability and privacy. These advantages come at the cost of increased deployment complexity, and more complex ToR switches.

4.1.4 Conclusions

We show that real-time monitoring of a data center comes at a cost: the increase in network traffic is significant enough to influence the networking performance of a data center. We estimated the additional load that is placed on a data center's network, and have shown that this additional load is significant. To counteract this problem, we proposed an architecture based on edge computing that enables real-time monitoring while reducing the required bandwidth, leveraging the network infrastructure of the data center by relying on ToR switches. As the edge gateway's responsibility is to interface with hardware sensors and virtual sensors to process their data, there are clear similarities with IoT hubs. Therefore, it may not be necessary to reinvent the wheel, and instead a selection can be made from one of many available IoT hubs.

4.2 Open-Source Internet of Things Hubs

Today, we are able to talk about smart buildings because the pervasiveness of IoT devices and because the ease of their installation and interoperation has brought them to the masses. This has been a long journey. About 40 years ago, home automation components were expensive and with non standardized interfaces, making them isolated components that would not coordinate and cooperate with other home components [AD08]. Vendors' lock-in was the standard practice. Slowly standards for building automation started to appear and interoperability efforts occurring in other areas of ICT contaminated also the home and office environment. More than 15 years ago, the use of Extensible Markup Language (XML)-based web services standards for resolving the building automation problem was proposed, and an application for supporting the elderly in their own homes was demonstrated [AD08; AZZ05]. Today, the situation is very different. Not only are internetworking and interoperation standards widely adopted, but there are also commercially available IoT hubs and even open source software for device integration. IoT hubs facilitate the integration of products from different vendors. Commercial examples include the Fibaro Home Center 3, Athom Homey, or the Samsung SmartThings Hub v3. While these hubs are easy to install, and easy to use, they do have their limitations. Typically, these hubs support only a few protocols and device types, and have a high total cost of ownership [BLM+11]. Vendor lock-in is also still a challenge that consumers face when using commercial IoT hubs, as these systems often promote the use of devices that are manufactured by the same company. On the contrary, open-source automation systems enables the use of well-developed system free of charge, though quality control may suffer [VR08].

There is no doubt that IoT hubs play an important role in smart homes and smart offices. But we also foresee that smart data centers will heavily rely on IoT hubs for the interconnection and integration of sensors and actuators, as well as data processing and analysis. This requirement was also highlighted in Chapter 3, where we discuss the need for an IoT platforms to automate the collection and analysis of data within data centers. Such a platform, or hub, would be responsible for the interconnection between sensors and actuators found in the data center, as well as perform real-time monitoring tasks to continuously evaluate metrics and alert operators when necessary. Edge computing has an important role to play in these task by optimizing latency and bandwidth utilization in the data center, as the potential for data generation is vast. Regardless of where the data analysis takes place, decentralized at the edge or more centralized in the cloud, the need for an IoT hub to support the concept of a smart data center is clear.

In recent years there has been a significant increase in the number of free and open-source IoT hubs. Their open-source nature allows them to provide support for hundreds, if not thousands, of diverse devices, overcoming the vendor lock-in issues that some of the commercial solutions have. At the same time, there is no cost associated with the software itself, which can often run on cheap single board computers such as the Raspberry Pi. Therefore, the total cost of ownership is also reduced. These types of systems have their own features and limitations, as will become clear later in this section. The challenge today is thus that of selecting the appropriate open-source automation system. First of all, there is a vast number of available systems, each with a varying number of functionalities, different levels of support for devices and protocols, and also variations in overall maturity and quality of the software. Second, while vendor lock-in is not a critical issue with these open platforms, migrating from one system to another system is time consuming due to the lack of migration tools. Finally, open-source projects come and go, which means the longevity of the system also needs to be considered. Projects with fewer contributors and low commit activity in the community are at higher risk of becoming stale or inactive. Therefore, it is important to be able to choose the system that fits the requirements of the users. To the best of our knowledge, there are no works that: (1) present an overview of available systems, (2) identify which of these systems are actively developed, or (3) perform any type of comparisons between these systems to identify differences in functional and non functional requirements, as well as identify gaps in the state of the art.

The work presented in this section is multi-purpose: it is a framework which can be applied to related domains by researchers, it is a tool for the practitioner to help make an informed decision when designing a building automation system, and it is an overview of open-source software for smart buildings for the hobbyist. These goals are achieved by evaluating 20 automation systems based on five core criteria, and making a selection of the top four systems based on the combined score of these criteria. These four systems are then methodologically compared in greater detail. The detailed comparison consists of two views: a use case based analysis to determine what is supported, and a criteria-based analysis that considers other useful aspects such as setup complexity and pricing. This evaluation is practical in nature; to evaluate each system, we installed and configured it on specific hardware. When applicable, the criteria is evaluated using the deployed system. Furthermore, a reference architecture for IoT hubs is identified based on the commonalities that emerge from the analysis of the four systems, and parallels are drawn between the reference architecture and the envisioned edge gateways for smart data centers.

4.2.1 Methodology

The approach that is taken consists of three steps. First, a list of IoT hubs is compiled and ranked. The list of systems is compiled by searching for open-source systems using numerous search engines (Google, Bing, Wikipedia) and online source code management platforms (GitHub, BitBucket, GitLab, Launchpad). The search terms that were used include: IoT Platform, Home Automation, and Building Automation. A selection of the top four systems is made, based on their individual scores. These top four systems are analyzed in great details. Next, a catalog of 13 system features is created based on 17 use cases, and each of the four systems is subjected to this catalog of features in order to determine which features are supported. The final step is an extensive analysis of 34 different criteria to which the four systems are subjected, each of these criteria are scored from 0 to 5. For the evaluation, the systems are deployed on a Lenovo ThinkPad E490 with an Intel Core i5-8265U CPU. What follows is a description for each of the three steps.

Initial Selection There is a wide variety of open-source IoT hubs available. To reduce the number of systems that are subjected to the detailed analysis, we perform an initial selection by ranking each system based on five criteria. Each criteria has a score i where $i \in \mathbb{Z} : i \in [0, 5]$. The criteria scores for each system are summed to obtain the final score. The criteria for the initial selection are as follows:

S1 **Commits**: the number of commits to the source code repository can be an important indicator of the level of activity within a project. While not all commits are of equal importance, their frequency is useful indicator. The commit score is calculated according to Equation 4.1.

$$s(x) = log_{10}(x) \tag{4.1}$$

Where x is the number of commits. When $s(x) > 5$ then $s(x) = 5$.

S2 **Stars**: the number of stars on a source code repository is comparable to the number of likes on social media platforms. These stars are commonly used as a proxy to determine the overall popularity of a repository [PDS16; SN16]. The score for this criteria is calculated in the same manner as the number of commits, using Equation 4.1, where x becomes the number of stars.

S3 **Latest Commit**: the date of the latest commit is checked in order to penalize inactive projects. This is done by looking at the number of years since the latest commit, and applying Equation 4.2.

$$s_{commits}(x) = \begin{cases} 5 - x, & \text{if } x \leq 5 \\ 0, & \text{otherwise} \end{cases} \tag{4.2}$$

Where x is the number of years since the last commit.

S4 **Documentation**: the quality of the documentation is crucial for automation systems. For the initial selection, if the system has any documentation, 5 points are assigned. If the system has no documentation, 0 points are assigned. This is done in order to penalize projects without any documentation. A more detailed review of the documentation will be performed for the top 4 systems, as part of the criteria-based analysis.

S5 **Contributors**: the number of contributors to the system are also taken into account, as this number is important for multiple reasons. First of all, the longevity of the project can be jeopardized when there are only a handful of contributors. Furthermore, there is a positive correlation between the number of contributors and the ease with which the source code of a system can be extended or modified. And finally, it is another indication of the popularity of the system. The score for this criteria is calculated using Equation 4.3.

$$s_{contributors}(x) = \begin{cases} 5, & \text{if } x > 1000 \\ 3, & \text{if } 100 < x \leq 1000 \\ 1, & \text{if } 2 < x \leq 100 \\ 0, & \text{otherwise} \end{cases} \tag{4.3}$$

Where x is the number of contributors to the project.

In case IoT hubs have identical total score after evaluating all criteria, a tie breaker is decided as follows: whichever system has the higher sum of scores for the stars, latest commits, and documentation criteria. In case the scores remain equal, the precise values of Equation 4.1 are used. After determining the overall ranking, the top four systems are selected, and these systems are analyzed in great detail.

Use Case Based Analysis The goal of the use case based analysis is to obtain a high-level overview of the functionality which the four selected IoT hubs offer. The list of features to which each of the selected systems is subjected has been extracted from 17 different use cases. These use cases have been partially selected from a survey by Abbas et al. [Abb18]. The remaining use cases are defined based on collective academic and industrial experience. Each use case requires the system to provide a certain set of features in order to fulfill the requirements. The collection of use cases and the corresponding set of features that have been extracted from the use cases is shown in Table 4.3 and Table 4.4.

The thirteen high-level features, that have been identified based on the uses cases, are divided into five categories: Visualization (F1), Localization (F2), Notification (F3), Data-Handling (F4), Interaction (F5). The high-level features of the four selected systems will be evaluated. When a system supports a feature natively, or provides an official plugin (near-native) then no points are deducted. If the feature is supported only through third-party plugins or applications, or requires a workaround, then 0.5 points are deducted. In case the feature is entirely unsupported, 1 point is deducted. The final score is calculated according to Equation 4.4.

$$s_{features}(x) = \begin{cases} 5 - x, & \text{if } x \leq 5 \\ 0, & \text{otherwise} \end{cases} \tag{4.4}$$

Where x is the number of points that are deducted for features that are not (fully) supported. What follows is the list of features that have been identified and their descriptions. These features were extracted from 17 different uses cases. Each feature is categorized in one of 5 categories.

F1 **Visualization**

F1.1 *ePaper/eInk*: allows the user to employ a paper-white display in a static way, that is, for viewing and not for interactions. Most commonly used for low resolution wall-mounted displays.

F1.2 *Dashboard*: allows the user to create customized dashboards, which can display real-time information (e.g. news feeds, states of other sensors, historical data). Also provides the user with control of various devices from a single location.

F2 **Localization**

F2.1 *Global Positioning System (GPS)-Tracking*: enables the system to track the location of the user, or multiple users, using GPS coordinates.

F2.2 *Presence Detection*: detects the presence of a human in an area of interest. Typically realized with Bluetooth Low Energy beacons, sometimes also realized with motion sensors. This includes the use of geo-fencing.

F3 **Notification**

F3.1 *Mobile Notification*: includes all types of notifications that can be received on mobile devices, for example: e-mail, SMS, Telegram, WhatsApp, push-notifications, text-to-speech, and voice calls.

F4 **Data-Handling**

- F4.1 *Sensor – Read*: forms the foundation of a smart building, allows the system to collect data from IoT devices.
- F4.2 *Device – Actuation*: forms the foundation of a smart building, allows the system to actuate IoT devices.
- F4.3 *Automation rule*: transforms the "IoT building" into a smart building, enables rules to be defined which can be triggered by events.
- F4.4 *Media Streaming*: allows the user to utilize video and audio streams, can be used for surveillance cameras or baby monitors.

F5 **Interaction**

- F5.1 *Mobile Remote Control*: extends feature F4.2, to enable control of any IoT device connected to the system using a mobile application.
- F5.2 *External API Calls*: enables the system to use external services on the Internet. For example, to display weather information to the user or to interact with an external calendar.
- F5.3 *Scheduler*: allows a user to schedule an action to be performed at a specific time. Also enables the creation of recurring events.
- F5.4 *Bio-metric User Authentication*: enables the use of bio-metric authentication. It can be used to secure private information on dashboards via Face ID or fingerprint authentication for door access control.

Criteria-based Analysis In the final step of the analysis, 34 distinct criteria are evaluated for each of the four remaining IoT hubs. Similar to the initial selection, each criteria in this part of the analysis also has a score i where $i \in \mathbb{Z} : i \in [0, 5]$. Where 5 indicates that a criterion has been fully satisfied, and 0 indicates that the criterion is entirely unfulfilled. What follows is the list of criteria and their descriptions. Each criterion is categorized in one of 8 categories.

C1 **Popularity and Community**

C1.1 *Activity:* is the system actively maintained? This is determined by the time of the latest commit to the main repository of the project.

C1.2 *Developer Popularity:* how popular is the system among developers? This is measured using the Stargazers Metric as defined by Jarczyk et al. [JGJ+14]. This metric utilizes the number of stars a source code repository has received.

C1.3 *Overall Popularity:* what is the popularity index of the system? The popularity index is determined using Google Trends[1]. The name of the system is used as the search term. The mean popularity over the last 24 months is used to quantify this criteria.

C2 **Pricing**

C2.1 *Support Plans:* are support plans available, and what is their cost? The cheapest support plan is considered.

C2.2 *Minimum Hardware Requirements:* what are the minimum hardware requirements, and what is the cost associated with these requirement? Prices are gathered from German retailers.

C2.3 *Recommended Hardware Requirements:* what are the recommended hardware requirements, and what is the cost associated with these requirement? Prices are gathered from German retailers.

C3 **Setup Complexity**

C3.1 *System Start-Up Time:* how much time does it take to download and install the system from scratch? The time is measured from download until the moment the system user interface becomes responsive.

C3.2 *Basic Sensor Setup:* how many steps does it take to setup the system and add an MQTT based sensor? After the completion of all the steps, the sensor data should be in the desired format, ready

[1]Google Trends, https://trends.google.com/trends/

to be displayed and used. A step in this context is not individual clicks or keystrokes, but a significant step in the overall process. For example, opening the plug-in marketplace, adding a sensor, and so on.

C4 **User Interface and User Experience**

These criteria will be measured by means of a self-review in the context of a workbench use-case experiment. We use the metrics defined by Sauro et al. [JS16].

C4.1 *Effort:* how much effort is required to create the necessary automation rules to fulfill a given scenario? The scenario is taken from the Smart Home Scenarios data set [Abb18]. Effort is expressed as the number of clicks necessary to finish the tasks.

C4.2 *Task Time:* how much time is required for the user to complete the task from C4.1? The time required to complete the task is measured.

C4.3 *Extensibility:* does the system support extension of the user interface? The number of available user interface widgets in the source code repository or on the marketplace are used to quantify this criteria.

C4.4 *Responsiveness:* does the system provide a user interface that adheres to the responsive web design principles? Responsiveness is the ability to correctly render the user interface on a variety of devices with different display sizes, from smartphones and tablets to 4K televisions.

C5 **Security and Authentication**

C5.1 *User Authentication:* is there any form of user authentication available? If so, what types of user authentication are supported?

C5.2 *Multiple User Accounts:* is it possible to create multiple user accounts within the system?

C5.3 *Authorization Management:* does the system provide role or attribute based access control to limit and control user permissions?

C6 **Extensibility and Support**

C6.1 *Custom Extensions:* does the system offer the possibility to implement custom plug-ins?

C6.2 *Extension Count:* how many extensions are available that extend the system to enable support for different IoT devices and protocols? Other types of extensions, such as user interface extensions and widgets, are excluded.

C6.3 *Quality of Documentation:* the quality of the documentation is defined by six sub-criteria, which are adapted from [CSA14; Sco17; WELL10].

C6.3.1 *Actuality:* does the system provide up-to-date documentation? The date of the most recent change to the documentation is taken.

C6.3.2 *Completeness and Comprehensiveness*: does the documentation cover all the different aspects of the system?

C6.3.3 *Examples:* does the documentation include examples on how to use the system.

C6.3.4 *Findability:* how easy is it to find the documentation? Is the documentation referenced on the homepage or the source code repository of the system?

C6.3.5 *Readability:* does the documentation use clear terms for describing the system and its features?

C6.3.6 *Skimmable:* is the documentation made to skim through quickly? To evaluate skimmability, the following sub-criteria are investigated: (1) informative headlines,(2) short paragraphs, (3) table of content, (4) global index, and (5) glossary.

C6.4 *Variety of Support:* does the system offer a variety of support methods? This is determined by the availability of e-mail support, forum support, and social-media support.

C7 **System Performance**

C7.1 *Concurrency:* how many concurrent MQTT-sensors are supported?

C7.2 *Update Rates:* how many updates per second can the system handle without performance drops?

C7.3 *Scalability:* does the system support horizontal scaling?

C8 **Software Quality**

C8.1 *Code reviews:* does the system require code reviews before new features are added to the source code?

C8.2 *End-to-end Test Metric:* does the system have end-to-end tests?

C8.3 *Formal Code Metric:* how many lines of code does the system have?

C8.4 *Pipeline Support:* does the system use Continuous Integration / Continuous Delivery pipelines?

C8.5 *Quality Checks for Third-Party Plugins:* does the system guarantee a certain level of quality of third-party plug-ins added to the repositories?

C8.6 *Unit Test Metric:* what is the unit test coverage level?

4.2.2 System Selection

The IoT hubs presently available range from recent projects to projects that are quite mature. Based on the proposed methodology, 20 systems are discovered and considered for further analysis. As the number of systems is great, a selection is made to reduce this number to 4 systems. As part of the selection process, each of these systems are evaluated based on the five metrics: commit count, number of stars, date of latest commit, documentation, and number of contributors. A total score for each system is obtained by summing the scores of the individual metrics.

The results of the initial selection process are presented in Table 4.2. The Table is populated with data collected on the 16th of July, 2021. The scores of the individual metrics for each of the 20 systems are shown, as well as the total score of each system. Based on the scoring, the top 4 systems are: Home Assistant, Domoticz, openHAB, and ioBroker. These are the systems that are selected and analyzed in greater detail. What follows next is a brief description of each of the selected system, covering its history, as well as the architecture and conceptual model. And finally, based on the commonalities between the systems, a generic IoT hub architecture is defined.

Table 4.2: Initial selection of IoT hubs.

IoT Hubs	**Rank**	**Score**	Commits	Stars	Latest Commit	Docs	Contributors
Home Assistant [Homb]	1	24	4.6	4.6	5	5	5
Domoticz [Doma]	2	21	4.1	3.5	5	5	3
openHAB [Comb]	3	17	3.2	2.7	5	5	1
ioBroker [ioBb]	4	17	3.2	2.7	5	5	1
HomeGenie [G-L]	5	16	3	2.4	5	5	1
Calaos [Cal]	6	16	3.1	2.2	5	5	1
Wirehome [Kra]	7	16	2.6	2.3	5	5	1
OpenMotics [Ope]	8	16	3.5	1.4	5	5	1
Freedomotic [NCT+]	9	16	3.2	2.6	4	5	1
FHEM [FK]	10	15	4.3	1.1	5	5	0
MisterHouse [Kee]	11	15	3.6	2.3	3	5	1
OpenNetHome [Str]	12	14	2.7	1.6	5	5	0
Ago Control [Kle]	13	14	3.6	0.5	4	5	1
TheThingSystem [MH]	14	12	3.1	2.5	0	5	1
üAutomate [GS]	15	11	2.2	1.1	3	5	0
Neon HomeControl [Gia]	16	11	1.9	1.4	3	5	0
Pytomation [kin]	17	11	3	1.9	0	5	1
Smarthomatic [Fre]	18	10	2.8	1.5	0	5	1
Smart Haus [Off]	19	8	2.1	0.8	5	0	0
AutoBuddy [Wau]	20	8	2.4	1.2	4	0	0

Home Assistant The Home Assistant project was founded by Paulus Schoutsen, and is an open-source system maintained by a worldwide community. It currently provides over 1.500 different integrations. These integrations add support for new devices, adapter protocols, user interface modifications or extensions, and the integration of external services. The configuration of Home Assistant is mainly done through the use of YAML Ain't Markup Language (YAML)-defined configuration files. Though more configuration options are being added to the user interface instead [HH19].

The Home Assistant core and its integrations are written in the Python programming language. The architecture of the system is shown in Figure 4.5. As can be seen in this figure, the event bus is the core of the system, listening to and firing events to other components. One of these components is the State Machine, used to keep track of the state of entities. Each change in a state fires an event that is handled by the event bus. The Timer component generates regular 'time changed' events. The Service Registry allows other components to register services, and allows these services to be discovered.

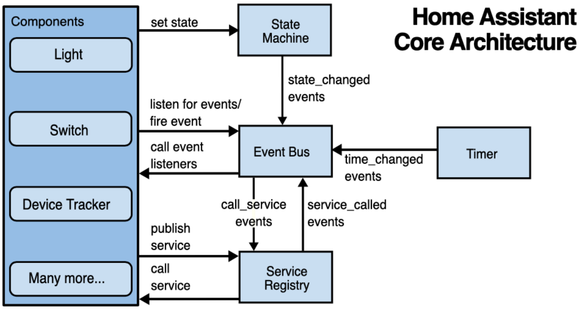


Figure 4.5: Home Assistant core system architecture [Homa].

In Home Assistant, every sensor, controller, and actor is a represented as a device which can be organized into groups. Every device and integration is represented as one or more entities, each having attributes representing the state of the entity. For automation, there is the ability to create automation rules which can be triggered by various events, including state changes of entity attributes, when a user enters a defined area, when the sun is set, or when the host restarts. In addition, the automation rules can have conditions to prevent the execution of the rule if the conditions are not satisfied. Lastly, there are actions that specify what should be executed when the automation rule fires.

Domoticz Domoticz is an IoT hub that is able to monitor and configure a variety of devices. The first version of Domoticz was released in December 2012. Thanks to a responsive user interface, the system is usable on both desktop and mobile devices. It is maintained by a large and active community of developers.

Domoticz is implemented in the C++ programming language. The project also implements its own web server, written in C++ as well. Unfortunately, little documentation is available on the architecture of Domoticz and on the underlying concepts of the project. However, Figure 4.6 shows an example of a typical Domoticz setup. As is shown, sensors and actuators are connected to Domoticz through MQTT. The actuators can be triggered by automation rules defined in an external system such as Node-RED, or Domoticz's own Blockly rule builder. The backend handles the incoming data, which can then be displayed on the Graphical UI (GUI).

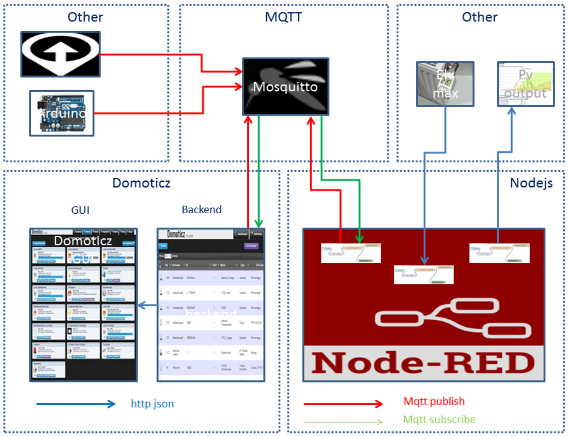


Figure 4.6: Domoticz system architecture [Domb].

openHAB Open Home Automation Bus, commonly known as openHAB, is an open-source automation controller [HHKM17]. The first lines of code were added to the project in 2010. The project does not have a singular creator, instead, it has been implemented by a community of volunteers. The openHAB system is vendor-independent and it works with many protocols and devices. This is one of its main strengths and goals: providing a uniform user experience regardless of the vendors and subsystems it interfaces with.

openHAB is primarily written in the Java programming language. A representation of the openHAB architecture can be seen in Figure 4.7. The figure clearly demonstrates that the Event Bus is the central component of the system, enabling communication between the other openHAB components. Bindings enable uniform communication between the system and the devices or services. Thanks to openHAB's extensibility, there are many different user interfaces available to interact with the system, as well as a Representational State Transfer (REST) API.

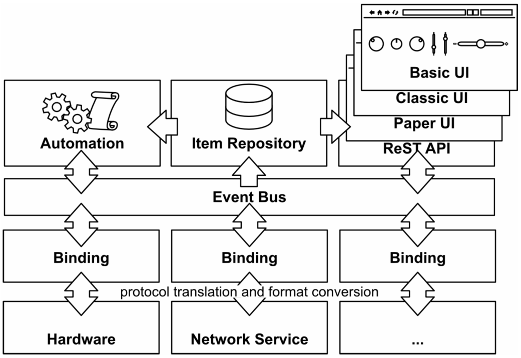


Figure 4.7: openHAB system architecture [HHKM17].

Conceptually, openHAB consists of five important elements: Things, Channels, Bindings, Items, and Links. Things are objects that are physically added to the system, and that can provide one or more functions. A temperature and humidity sensor is a single physical Thing that provides two functionalities: temperature sensing, and humidity sensing. Each functionality of a Thing is exposed through a Channel. Bindings are adapters, they enable access to Things through the system and hide hardware specific details. Items are stateful, and provide functionality that can be used by application or in automation logic. A Link connects one or more Channels to one or more Items. The act of linking Channels and Items enables the functionality provided by an Item to be access through that specific Channel. Figure 4.8 illustrates this connection between Things, Items, Channels, and Links.

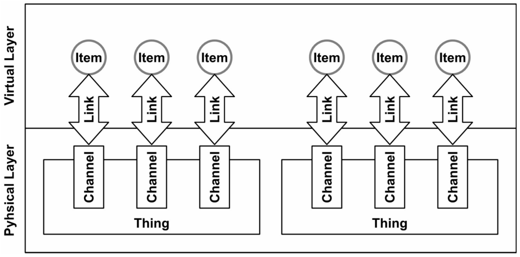


Figure 4.8: openHAB conceptual architecture [HHKM17].

ioBroker The first version of ioBroker was published in 2014 by the company ioBroker GmbH. It is the successor of the CCU.IO project, which was terminated in April 2015 [HH19]. The goal of ioBroker is to integrate heterogeneous smart building devices and systems. At the time of writing, ioBroker offers 350+ adapters to integrate with different devices and systems. The system is non-commercial software, and is developed and maintained by volunteers. One of the main advantages of ioBroker is that all configuration can be done through a web interface. This makes the system accessible to a wide range of users [HH19]. Additionally, ioBroker uses the local API of a device, when this is supported, in order to bypass the online cloud services of the vendors. The benefit of this approach is that sensitive data remains local and any security vulnerabilities that might exist in the cloud service are avoided.

The ioBroker core is primarily written in JavaScript. The adapters are also written in JavaScript, though Typescript can be used as well. The architecture of the system is shown in Figure 4.9. It is clear from the figure

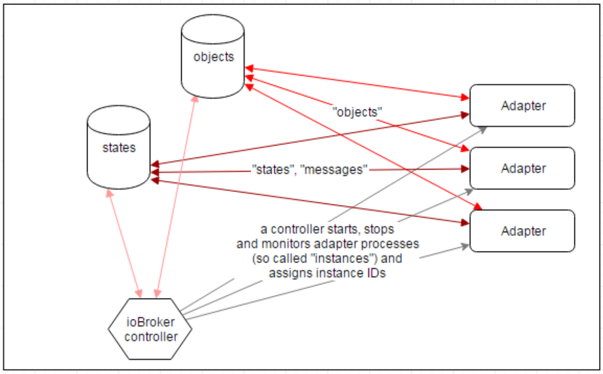


Figure 4.9: ioBroker system architecture [ioBa].

that two databases play an important role: the objects database and the states database. The object database is responsible for storing meta data and configurations, whereas the states database is used to keep track of the state of devices and services. By default in-memory databases are used, but there are adapters available to support other types of databases. Adapters are used to integrate with different IoT devices and systems, resulting in a loosely coupled architecture. The controller is responsible for managing the adapter processes.

From a conceptual viewpoint, ioBroker is an extremely modular system. Each module or adapter is responsible for a specific function. Even the administration user interface is developed as a separate adapter. A central coordinator, also known as the js-controller, is responsible for managing the adapters and realizing the communication between them.

4.2.3 The IoT Hub System Architecture

The architecture of an IoT hub influences the capabilities and the characteristics of the system itself. It is clear that the architectures of the top four systems show a large number of commonalities. On the basis of these, we discuss concepts, components, and designs that are present in multiple systems and can be considered jointly core architectural principles for an IoT hub. Due to a lack of documentation, especially with respect to system architecture, the Domoticz system is not included. This emerging architecture is shown in Figure 4.10. What follows next is a description for each of the components.

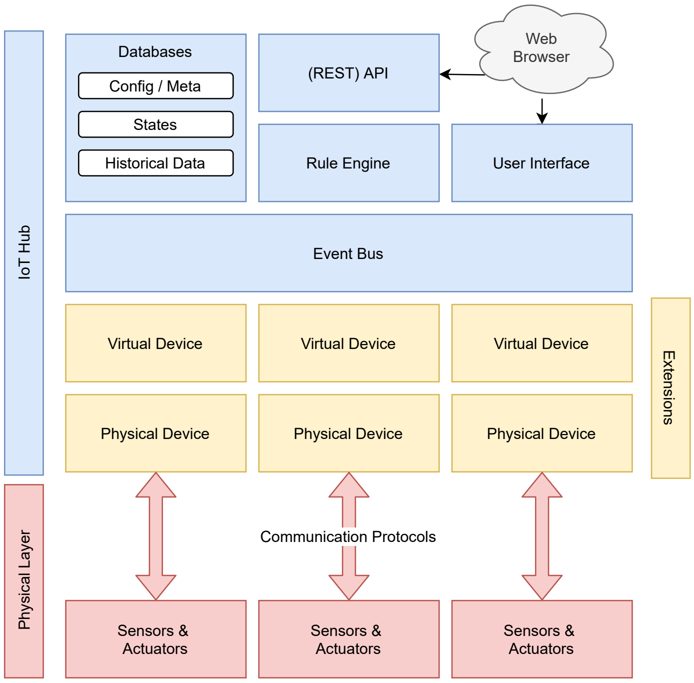


Figure 4.10: The generic IoT Hub architecture.

- *Database* is a critical component to store historical device data. The database is generally also used to store the current state of devices and other entities within the system. Furthermore, it is also used to track all devices and extensions. All three systems, Home Assistant, openHAB, and ioBroker support a multitude of database systems.

- *Web-based User Interface* is the most popular option to interact with the system, though not the only one. While Home Assistant leans heavily on a single user interface which is highly personalize, openHAB and ioBroker support multiple, entirely distinct, user interfaces each with their own strengths and weaknesses.

- *(REST) API* to interact with the system by means other than the web-based user interface. All of the investigated systems have an API of some form and shape to enable interactions with the system, such as triggering actuators or automation rules.

- *Event Bus* is present in both Home Assistant and openHAB, the event bus plays a central role in facilitating asynchronous communication between components. The bus is also used to listen to events generated by devices or by the system itself. On the contrary, ioBroker opts for a direct TCP/IP connection for communication between components.

- *Physical Device,* such as a temperature sensor or relay, needs to be represented in the building automation system. Home Assistant simply refers to a physical device as ’device’, openHAB uses the label ’thing’. ioBroker does not make a clear distinction, though in general physical devices are abstracted and represented as the ’object’ data type.

- *Virtual Device* can represent a physical device that has multiple sensors, and sometimes multiple actuators, to uniformly abstract individual features provided by physical devices. In Home Assistant, a virtual device is called an ’entity’, while openHAB uses the name ’item’. Again, ioBroker does not support such abstraction.

- *Extensions* are critical for IoT hubs given the high dynamicity of the related ecosystem. Extensions allow the systems to offer support for many devices and services. Though, there is no agreement on what they should be called; openHAB talks about ’bindings’, Home Assistant considers them ’integrations’, and ioBroker uses the concept of ’adapters’.

- *Rule Engine* enables the automation of the building to define and execute rules. Home Assistant and openHAB provide their own components to wire together devices and events. ioBroker relies on extensions, such as Blockly and Node-RED. Though both Home Assistant and openHAB also support Node-RED.

Interestingly, many commonalities can be observed between the generic IoT hub architecture, and the edge gateway architecture for smart data centers proposed in Figure 4.4. In both cases there is a need for integrating with many different sensors and actuators, each using a variety of communication protocols. Events are emitted to an event bus or event queue, allowing the rest of the system to interact by monitoring these events and triggering rules when certain preconditions are met. Therefore, IoT hubs are good candidates for becoming the foundation of an edge gateway as we have envisioned for the real-time monitoring of smart data centers.

4.2.4 Use Case Based Analysis

The features supported by the four selected systems largely overlap, though they are not exactly the same. Let us consider the four system under the lenses of the use cases and corresponding features identified in Table 4.3 and Table 4.4. The support for each of these features is determined and translated into a numeric score as shown in Equation 4.4. In order to determine which features are supported, each system is deployed in practice in order to verify the support of each feature. The results are summarized in Table 4.5 where a check mark (✓) indicates native or near-native support of the feature; a circle (O)" indicates that the feature is supported by means of a third-party solution, or that it requires significant effort (e.g. writing custom scripts) from the user; and finally, a dash (-) indicates that the feature is not supported.

F1.1 (ePaper / eInk Display): Home Assistant is the only system that supports ePaper Displays natively. There are third-party alternatives available for Home Assistant, such as Basic-Hass-Dash or HASS eInk Display, which can display the Home Assistant dashboard in an ePaper-friendly manner. The remaining systems only support this feature through third-party solutions.

Table 4.3: Use cases and their corresponding feature requirements, part 1.

Use Cases \ Features	F3.1: Mobile Notifications	F4.3: Automation Rules	F2.1: GPS-Tracking	F4.1: Sensor – Read	F5.1: Mobile R.C.	F4.2: Device – Actuation	F4.4: Media Streaming	F5.2: External API Calls	F1.1: Paper-White Display	F1.2: Dashboard	F2.2: Presence Detection	F5.3: Scheduler	F5.4: Biometric User Auth.
In case of a fire, I would like to be informed and receive notifications on my mobile phone.	✓	✓		✓									
When it is winter and it is cold, I want my heating to turn on before I get home from work.		✓	✓	✓		✓							
Even when my lamps and light fixtures are from different manufacturers, I want to be able to control all of them from one device.					✓	✓							
While doing the groceries I want to be able to access a video stream inside my fridge to check its contents.							✓						
I would like to know when I should empty my rain barrels in order to benefit the most from an upcoming rain storm.	✓	✓		✓				✓					
I want to see, in real-time, much power my solar panels currently generate.		✓		✓					✓	✓			
When I leave the house, and I am the last person to do so, all lights should be turned off in case I forget.		✓	✓			✓					✓		
Every morning at 5:00 AM start brewing my coffee automatically so it will be ready for me when I wake up [Abb18].						✓						✓	

Table 4.4: Use cases and their corresponding feature requirements, part 2.

Features / **Use Cases**	F3.1	F4.3	F2.1	F4.1	F5.1	F4.2	F4.4	F5.2	F1.1	F1.2	F2.2	F5.3	F5.4
If the Carbon Monoxide (CO) detector is low on battery, I would like to receive an email notification that the battery needs to be replaced. This will avoid triggering the alert that goes off when the battery is low, and usually causes panic as people may think there is a CO hazard [Abb18].	✓	✓		✓									
During the day, whenever I walked into the bathroom the light would come on. However, after a certain time, when it is night, when I walk in the bathroom I would want a much softer light to come on [Abb18].		✓				✓					✓		
When the doorbell rings it interacts with both the television and the lights. If the TV is on then when the doorbell rings the TV should mute itself and/ or pause depending on what the input the TV is. Additionally, the lights should change color to indicate someone is at the door, in case the doorbell was not heard [Abb18].		✓		✓		✓							
I want the sprinkler system to water the lawn only when: it is not raining already, and it is between 4 am and 6 am, and the temperature is above 50 degrees Fahrenheit. [Abb18]		✓		✓		✓		✓					
When I am home and shut my windows, I want all of my doors to lock automatically [Abb18].		✓		✓		✓							
I would like to receive a notification when the stock market passes a certain threshold.	✓	✓						✓					
I only want people I know, or people I have given permission, to be able to access my smart building.					✓			✓					✓
My building dashboard should indicate how many unread mails I have.									✓	✓			
My appointments for the day should be visualized on the display of my dynamic calendar, my garbage collection schedule should also be integrated.								✓				✓	

Domoticz has the Dashticz third-party dashboard that could be user on ePaper devices with a browser. For openHAB there is a third-party project called PaPiRus-MQTT, which uses the lightweight message queue MQTT to transmit the data from openHAB to the ePaper device. For ioBroker a solution suggested in the community forums is to use Remote Procedure Call (RPC) calls and a Homematic display.

F1.2 (Dashboard): All systems have native support for dashboards to display data and to control devices. Both Home Assistant and ioBroker make use of the Lovelace User Interface (UI). All systems also support third-party dashboards. For example, by connecting to an InfluxDB time-series database and using the Grafana visualization platform.

F2.1 (GPS-Tracking): Home Assistant includes a native companion app that can be installed on mobile devices to enable detailed tracking information. No native support is provided by Domoticz, though there are workarounds to include GPS data or to use the GeoFence mobile app. openHAB includes native add-ons that offer integration with applications such as OpenPaths and OwnTracks to provide detailed tracking. For ioBroker, the community provides third-party adapters, such as ioBroker.places, which adds support for mobile apps such as OwnTracks.

F2.2 (Presence Detection): All systems provide support for detecting the users presence using Internet Protocol (IP)-based approaches, tracking the presence of mobile devices in the building's network. Additionally, Home Assistant, Domoticz, and openHAB provide native support for Bluetooth Low Energy (BLE) beacons. Third party adapters exist for ioBroker to enable presence detection through BLE.

F3.1 (Mobile Notification): All systems support mobile notifications. Email notifications are supported on all systems, and add-ons are also provided for Telegram notifications. Push notifications are also supported on all systems.

F4.1 (Sensor – Read): All systems support the reading of sensor data from devices.

F4.2 (Device – Actuation): All systems support the actuation of devices.

F4.3 (Automation Rule): The creation of automation rules is an integral part of the dashboard provided by Home Assistant and openHAB. Domoticz and ioBroker rely on scripts written by the user, or the use of Blockly, a third-party visual programming editor.

F4.4 (Media Streaming): This criteria of Media Streaming is evaluated based on the support for the Real Time Streaming Protocol. Home Assistant, openHAB, and ioBroker have out-of-the-box support for it. Domoticz does not offer native support for the protocol, though third-party workarounds do exist.

Table 4.5: Features Overview: ✓ (near-)native support, *O* 3rd-party support, – no support.

System / **Features**	Home Assistant	Domoticz	openHAB	ioBroker
F1.1: ePaper / eInk Display	O	O	O	O
F1.2: Dashboard	✓	✓	✓	✓
F2.1: GPS-Tracking	✓	O	✓	O
F2.2: Presence Detection	✓	✓	✓	✓
F3.1: Mobile Notification	✓	✓	✓	✓
F4.1: Sensor – Read	✓	✓	✓	✓
F4.2: Device – Actuation	✓	✓	✓	✓
F4.3: Automation Rules	✓	✓	✓	✓
F4.4: Media Streaming	✓	O	✓	✓
F5.1: Mobile Remote Control	✓	O	✓	O
F5.2: External API Calls	✓	✓	✓	✓
F5.3: Scheduler	✓	✓	✓	✓
F5.4: Biometric User Auth.	✓	-	O	O
Score	4.5	2	4	3

F5.1 (Remote control): Home Assistant and openHAB provide official mobile applications, for both Android and iOS, that enable the remote control of devices connected to the system. ioBroker only offers an official iOS application, and a third-party Android application. Domoticz provides support for multiple third-party applications, such as Domoticz for Android, and ImperiHome.

F5.2 (External API Calls): All systems have support for external service REST API calls.

F5.3 (Scheduler): Home Assistant, openHAB, and ioBroker offer native solutions for time-based triggers and scheduling. Whereas, Domoticz relies on user-defined scripts or Blockly functionality.

F5.4 (Bio-metric User Authentication): The only system providing native support from bio-metric features is Home Assistant. There are official integrations available for Dlib Face Detect, Facebox, and Microsoft Face Detect, among others. Third-party projects also explain how to integrate fingerprint recognition. For both openHAB and ioBroker, third party projects are available that demonstrate the addition of face recognition. Domoticz offers no support for this feature.

4.2.5 Criteria-based Analysis

To complete the analysis of the system, after having studied the functionalities derived from uses cases, we proceed with checking the features identified in Section 4.2.1. Each criteria is evaluated, and a score between 0 (not fulfilled) and 5 (completely fulfilled) is assigned to each system.

C1.1 (Activity): All four systems are actively maintained. The latest commit to the source code repository for each system was made on the 16th of July, 2021. As a result, the four systems receive the maximum score of 5 for this criterion.

C1.2 (Developer Popularity): We define popularity based on the Stargazers metric as defined by Jarczyk et al. [JGJ+14]:

$$f(x) = log_{10}(x + 10) \tag{4.5}$$

where x is the number of stars a source code repository has received. Applying Equation 4.5 to each of the systems and rounding to the nearest integer value yields the results shown in Table 4.6.

Table 4.6: Popularity among developers.

System	Stars	Stargazers	Score
Home Assistant	33511	4.53	5
Domoticz	2700	3.43	3
openHAB	297	2.49	2
ioBroker	684	2.84	3

C1.3 (Overall Popularity): The *Google Popularity Index* is used to measure overall popularity. Figure 4.11 shows the trend of the relative popularity index over time. The value represents the search interest relative to the highest point on the chart. A value of 100 represents the peak popularity. Scores are assigned based on the results of the latest relative popularity as is shown in Table 4.7, where the most popular system receives the highest score.

Table 4.7: Relative Popularity (2016 - April 2020).

System	Average	Gradient	Latest	Score
Home Assistant	38.92	37.09%	89	5
Domoticz	17.39	0.47%	12	2
openHAB	24.50	2.82%	30	4
ioBroker	9.35	9.39%	19	3

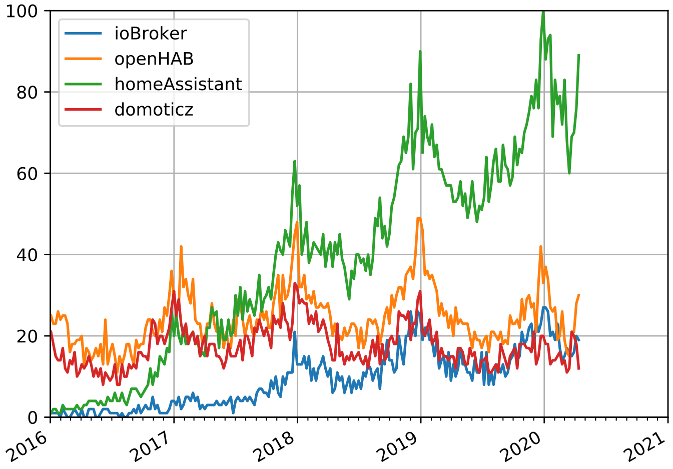


Figure 4.11: Google Popularity Index (2016 - April 2020).

C2.1 (Support Plans): None of the four IoT hubs provide support plans. Instead, the support is provided by the communities in the shape of chat rooms and discussion boards. Therefore, for this category, all of the systems receive zero points.

C2.2 (Minimum Hardware Requirements): Single board computers such as the Raspberry Pi are popular hardware platforms for IoT platforms. Not only do they have a small form factor, they also are one of the most affordable hardware options available for these systems. Table 4.9 shows the mapping between price and score. The minimum hardware requirements for each system, their price, and the score, are shown in Table 4.8. These prices are in accordance with what is expected in the market for 2021.

C2.3 (Recommended Hardware Requirements): Table 4.9 shows the mapping between price and score. The recommended hardware requirements for each system, their price, and the score, are derived and shown in Table 4.10. These prices are in accordance with what is expected in the market for 2021.

Table 4.8: Minimum hardware requirements.

System	Required Hardware	Price	Score
Home Assistant	Raspberry Pi 3 Model B	34.90 - 38.17 Euro	4
Domoticz	Raspberry Pi 1 Model B+	26.99 - 31.92 Euro	5
openHAB	Raspberry Pi 2 Model B	33.90 - 39.79 Euro	4
ioBroker	Raspberry Pi 2 Model B	33.90 - 39.79 Euro	4

Table 4.9: Price to score conversion.

≤ 30 €	≤ 35 €	≤ 40 €	≤ 60 €	≤ 100 €	> 150 €
5	4	3	2	1	0

Table 4.10: Recommended hardware requirements.

System	Recommended Hardware	Criterion C2.2	Score
Home Assistant	Raspberry Pi 4 Model B	38.73 - 39.56 Euro	3
Domoticz	Raspberry Pi 3 Model B	34.90 - 38.17 Euro	4
openHAB	Raspberry Pi 2 Model B	33.90 - 39.79 Euro	4
ioBroker	Raspberry Pi 4 Model B	38.73 - 39.56 Euro	3

C3.1 (System Start-Up Time): Each system provides Docker container images for deployment. Since Docker containers automate many of the installation steps, using these containers is generally the fastest way to setup a system. Therefore, the time to download and and fully initialize these containers is measured. The results are shows in Table 4.11. Scores are assigned as follows, where time is measured in seconds: 5 points for ≤ 40, 4 points for ≤ 80, 3 points for ≤ 120, 2 points for ≤ 180, and 1 point for ≤ 300.

C3.2 (Basic Sensor Setup): To evaluate this criteria, a MQTT broker (Eclipse Mosquito) and a virtual MQTT sensor are set up using Docker containers. The virtual MQTT sensor generates random sensor data. The number of required steps for each system to integrate this MQTT-based

Table 4.11: System start-up time.

System	Mean Setup Time	Score
Home Assistant	85s	3
Domoticz	84s	3
openHAB	69s	4
ioBroker	94s	3

Table 4.12: Interaction steps to setup an MQTT sensor.

System	# Interaction steps	Score
Home Assistant	5	4
Domoticz	3	5
openHAB	8	3
ioBroker	11	2

sensor is measured. The results, shown in Table 4.12 show that setting up the sensors requires a comparable number of steps, the minimum is 3 for Domoticz and the maximum is 11 for ioBroker.

C4.1 (Effort): To measure the effort, we consider the following task from the Smart Home Scenario data set is measured: During the day, whenever I walk into the bathroom the light should come on, and after a certain time, when it is night, I would want a much softer light to come on [Abb18]. All systems are tested with virtual sensors and lighting fixtures, which are set-up ahead of time. The automation rule creation of Domoticz is tested through its integrated rule engine Blockly. The creation of the automation rule and the corresponding scene takes 43 clicks using Domoticz. In turn, openHAB offers an experimental UI-based rule engine [Comc], which is used in this test. Overall, it takes 70 clicks to install the rule engine and create the necessary rules in openHAB. Home Assistant comes with a built-in automation rule editor, which is used in this test, taking 38 clicks in total to setup the rules to fulfill the scenario. Due to an unexpected error within a required module of ioBroker, the score for this system cannot be evaluated. Table 4.13 provides an overview of the results and the scoring, which is assigned by rank.

Table 4.13: Effort of creating the automation rules for the user story.

System	# clicks	Score
Home Assistant	38	5
Domoticz	43	4
openHAB	70	3
ioBroker	-	0

C4.2 (Task Time): The scenario and tasks used to evaluate the Task Time criteria are identical to *C4.1 (Effort)*, and the same setup is used. In our evaluation, the time to complete the task takes 1:39 minutes with Domoticz, 2:16 minutes with openHAB, and 1:00 minute with Home Assistant. Because of an error with a required module, ioBroker cannot be evaluated. An overview of the test result is shown in Table 4.14.

Table 4.14: Completion time of the user story.

System	Duration	Score
Home Assistant	60s	5
Domoticz	99s	4
openHAB	136s	3
ioBroker	-	0

C4.3 (Extensibility): In terms of extensibility, Home Assistant has 25 official widgets available for its Lovelace UI. Panels can also be customized using ReactJS. Domoticz does not have built-in widgets. There are ways the dashboard can be customized, but they are quite limited, such as applying skins and modifying icons. Domoticz can be used with a number of third-party dashboards, including Reacticz, New Frontpage, and Dashticz. openHAB has 13 built-in widgets. It also allows for inclusion of custom widgets, which can then be accessed through the widget gallery. The fourth and final system, ioBroker, has a considerable number of third party widgets, that can be imported into the system. The widgets can also be customized further by changing the Cascading Style Sheets (CSS) attributes. Home Assistant and openHAB both provide official UI widgets and other means to customize the UI, therefore they receive the maximum score of 5. The other two systems, Domoticz and ioBroker, appear to only support third party widgets, thus receiving a score of 3.

C4.4 (Responsiveness): All IoT hubs provide a UI that is responsive, ensuring that it functions on smartphones, tablets, and other mobile devices. However, the UI of Domoticz is lacking with regards to its responsiveness, for example, the automation rule editor is borderline unusable on a mobile device as it does not scale well. Therefore, two points are deducted from its score. To conclude, Home Assistant, Domoticz and openHAB receive full marks, and Domoticz has a score of 3.

C5.1 (User Authentication): Domoticz, openHAB, and ioBroker only offer basic-auth as a login method. Home Assistant is the only system which provides webauthn as an additional login method. In addition, Home Assistant supports Multi-factor Authentication (MFA) and Two-factor Authentication (2FA) for self-hosted solutions. The other systems only support 2FA for their online cloud-hosted solutions. Table 4.15 shows the results. Basic-auth yields a score of 3, additional points are given for webauthn support, as well as MFA/2FA support.

Table 4.15: Protected settings by authentication.

System	basic-auth	MFA/2FA	webauthn	Score
Home Assistant	✓	✓	✓	5
Domoticz	✓	(✓)		3
openHAB	✓	(✓)		3
ioBroker	✓	(✓)		3

C5.2 (Multiple User Accounts): Multiple user accounts are either supported (5 points), or not (0 points). All four systems support multiple user accounts, therefore receiving the full score of 5.

C5.3 (Authorization management): Both Domoticz and ioBroker provide some form of authorization management. Domoticz supports a single admin user and multiple regular users, who can be invited by the admin. ioBroker has a similar authentication concept to Domoticz, there is one admin user and there can be multiple regular users. However, in ioBroker, users be separated into different user groups with different permissions. Currently, Home Assistant does not provide authorization management. Thus, all users have the same level of access. openHAB also does not provide any kind of authorization management. ioBroker receives a score of 5, as it provides the most comprehensive authorization management. Domoticz receives a score of 3, as the features are more limited. Home Assistant and openHAB do not currently have authorization management and therefore receives 0 points.

C6.1 (Custom Extensions): All four systems provide the means for their systems to be extended with additional functionality using custom extensions or plugins. One thing of note is that ioBroker, unlike the other systems, does not provide a well documented framework for implementing extensions. As all systems support custom extensions, they all receive the full score of 5 points.

C6.2 (Extension Count): The number of available extensions for each IoT hub is obtained by analyzing official wiki's and available documentation, as well as plugin repositories or marketplaces, when available. An overview of the number of available extensions for each system can be found in Table 4.16. Score is assigned in accordance with the system ranking.

C6.3 (Quality of Documentation): The quality of documentation is defined by the following sub-criteria: C6.3.1 Actuality, C6.3.2 Completeness / comprehensiveness, C6.3.3 Examples), C6.3.4 Findability, C6.3.5 Readability, and C6.3.6 Skimmability. The overall quality of the documentation is determined as the average over all sub-criteria. The final scores are presented in Table 4.17. What follows next is a more detailed analysis for each of the sub-criteria.

Table 4.16: Extension count.

System	C6.2	Score
Home Assistant	1611	5
Domoticz	79	2
openHAB	323	3
ioBroker	352	4

Table 4.17: Quality of documentation.

System	C6.3.1	C6.3.2	C6.3.3	C6.3.4	C6.3.5	C6.3.6	Score
Home Assistant	5	5	5	5	5	4	5
Domoticz	5	0	3	5	5	4	4
openHAB	5	3	0	5	5	3	4
ioBroker	5	0	0	5	5	4	3

C6.3.1 (Actuality): The actuality of the documentation is determined by the date of the last change. This data was gathered either from the documentation of the system, or from its source code repository. At the time of writing, each of the four system had their documentation updated within the last two days, and therefore they all receive 5 points.

C6.3.2 (Completeness / Comprehensiveness): The completeness and comprehensiveness of the documentation is determined by how often other sources besides the official documentation had to be consulted during the analysis of the IoT hubs. The scores are assigned as follows: 5 for a complete and comprehensive documentation and rarely needing other sources, 3 for a mostly complete documentation where other sources had to be consulted multiple times, and 0 in case there is no documentation or external sources had to be primarily used to perform the analysis. Home Assistant scores full marks for its documentation, openHAB scores 3 points for sufficient but incomplete documentation. Both Domoticz and ioBroker are lacking critical information in their documentation, requiring external sources to be accessed regularly and therefore receive a score of 0.

C6.3.3 (Examples): Home Assistant provides a separate section dedicated to examples of many different use cases. Therefore, it receives the full score for the availability of examples criterion. Domoticz offers a manual that shows how many of its features can be used, individually. Though, it does not provide complete example use cases. Therefore, Domoticz receives a score of 3. Both openHAB and ioBroker do not provide any examples use cases in their documentation, and therefore receive a score of 0.

C6.3.4 (Findability): Since all of the IoT hubs have made it simple to locate the documentation, either through links on their websites or via the source code repositories, each of them receives the full score.

C6.3.5 (Readability): Each of the systems use clear terms for the description of their system and features. It is interesting to note that the same concepts have different terms in different systems, but the terms are well defined. Despite this, it is not difficult to understand the similarities between the systems. Therefore, each of the four systems receives a score of 5 for this criterion.

C6.3.6 (Skimmability): To determine skimmability, the documentation for each system is analyzed in terms of: (1) informative headlines, (2) short paragraphs, (3) table of content, (4) global index, and (5) glossary. If an item is partially or not available, then 1 point is deducted from the score. The results are shown in Table 4.18.

Table 4.18: Quality of documentation - skimmable.

System	(1)	(2)	(3)	(4)	(5)	Score
Home Assistant	✓	✓	-	✓	✓	4
Domoticz	✓	✓	✓	✓	-	4
openHAB	✓	✓	-	✓	-	3
ioBroker	✓	✓	✓	✓	-	4

C6.4 (Variety of Support): Each IoT hub has some presence on social media for support. Home Assistant is on Twitter, Facebook, and Reddit, as well as Discord. Domoticz is on Facebook, Reddit, and Twitter, though the

Twitter account appears to be inactive. openHAB is on Twitter, Facebook, Reddit, and YouTube. Though, both Facebook and YouTube accounts appear to be inactive. ioBroker is on Twitter, Facebook, and Reddit. In this case, the Twitter account appears to be inactive. When it comes to email support, openHAB and ioBroker provide an email address, Home Assistant does too but not for support purposes, and Domoticz has no email contact details. Each of the four systems has a forum available for support. The final score can be found in Table 4.19.

Table 4.19: Variety of support.

System	**Social Media**	**Forum**	**Email**	**Score**
Home Assistant	√	√	-	3
Domoticz	√	√	-	3
openHAB	√	√	√	5
ioBroker	√	√	√	5

C7.1 (Concurrency): To evaluate the concurrency criterion, a virtual MQTT sensor is created and connected to each of the systems. The sensor updates at a frequency of one message per second. The number of virtual sensors is gradually increased until noticeable performance issues arise. These performance issues can include, but are not limited to, out-of-order message processing, data inconsistency, high system response time, UI freezes, or system crashes. The systems are deployed on a Raspberry Pi 3 Model B+ using the pre-configured Docker containers that each system provides. The virtual sensors and MQTT broker are deployed on separate machines.

When testing, ioBroker was able to handle at least 100 concurrent sensors. However, at this point problems do arise when displaying the real-time logs of the incoming sensor data, which becomes notably slow until the page is refreshed. Domoticz was able to handle up to 10 concurrent sensors, at which point the processing of messages was notably delayed, and the order in which the messages were processed became inconsistent. Home Assistant was able to process the data from 70 concurrent sensors. When increasing beyond this number of sensors, Home Assistant is unable to process the messages on time, and the visual representation of the data will be delayed. Unfortunately, openHAB does not support this automated test bench, and each virtual sensor has to be manually configured. Therefore, only 10 sensors were set up. These manual tests indicated that openHAB was able to handle at least 10 concurrent sensors. Table 4.20 shows the results, the score is assigned in accordance with the ranking.

Table 4.20: Maximum concurrent sensors.

System	**C7.1**	**Score**
Home Assistant	70	4
Domoticz	10	2
openHAB	10+	3
ioBroker	100+	5

C7.2 (Update Rates): To evaluate update rates, the same setup as in C7.1 is used, with the difference that there is only one virtual MQTT sensor. However, this single virtual MQTT sensor is able to transmit data at varying rates: 1, 10, and 100 messages per second. Tests show that all of the systems are able to handle a single sensor with update rates of up to 100 messages per second. The results are shown in Table 4.21.

Table 4.21: Maximum update rates.

System	**1/s**	**10/s**	**100/s**	**Score**
Home Assistant	✓	✓	✓	5
Domoticz	✓	✓	✓	5
openHAB	✓	✓	✓	5
ioBroker	✓	✓	✓	5

C7.3 (Scalability): Two of the four IoT hubs that were examined offer horizontal scaling. ioBroker supports a multi-host-mode setup, which makes it possible for multiple instances to operate in parallel. And openHAB supports a special version called openHAB-cloud, that enables off-loading to the cloud or a local cluster. Both ioBroker and openHAB receive a score of 5, whereas Home Assistant and Domoticz receive 0 points due to the lack of scalability.

C8.1 (Code reviews): Each of the four systems require code reviews before a pull or merge requests is accepted. Home Assistant and openHAB provide a contribution document, in which the process to contribute to the source code is detailed. Domoticz and ioBroker are less restrictive, but do enforce code reviews. Therefore, all systems receive the full score.

C8.2 (End-to-End Test Metric): None of the IoT hubs provide end-to-end tests, neither in their main source code repositories, or their UI-specific source code repositories. Therefore, a score of 0 is assigned to each of the systems.

C8.3 (Formal Code Metric): The maintainability of the source code is critical for the longevity of the project. Each system provides a core source code repository which contains the code for the backend of the system. To determine the size of the backend, the Lines of Code (LoC) are determined for each of the core repositories. Scores are assigned in accordance with the LoC, where fewer lines equates a higher score. The results are shown in Table 4.22

Table 4.22: Lines of Code of the core repository.

System	LoC	Score
Home Assistant	542013	2
Domoticz	208295	3
openHAB	127745	4
ioBroker	37017	5

C8.4 (Pipeline Support): Each of the systems use some form of Continuous Integration and Continuous Delivery (CI/CD) pipeline to automate the software engineering process such as testing, and the creation of Docker images. All of the systems use either Travis or GitHub workflows for this purpose. To conclude, each system receives 5 points for this criterion.

C8.5 (Quality checks for 3rd party plugins): Home Assistant provides an elaborate quality scale to determine the quality of 3rd party plugins [Homc]. The focus of Domoticz is more on the technical specification of the plugins, rather than on the process that developers have to follow. openHAB on the other hand, performs static code analysis, and provides a detailed list of requirements for contributions [Coma]. ioBroker ensures that all 3rd party plugins are checked before they are made available in their adapter repository. Additionally, there is a forum on which developers search for members to evaluate the new plugins, and verify that they are functioning. The final scores are as follows: Home Assistant, openHAB, and ioBroker receive full marks for their elaborate checks. Whereas Domoticz receives 3 points as their quality checks are significantly less elaborate.

C8.6 (Unit test metric): Home Assistant makes use of unit tests, though the available information on how they should be written is limited. openHAB provides a detailed explanation about how the unit tests and integration tests should be coded, highlighting the importance of these tests in the project. ioBroker also provides comprehensive documentation on the practices that should be used when writing tests. Domoticz on the contrary does not have unit tests. The three systems that do have unit tests receive a score of 5, whereas Domoticz receives a score of 0.

4.2.6 Discussion

The comparison of the systems on the basis of the use cases and features has highlighted many similarities among the currently available open-source projects, but also some key differences. What follows is a discussion of the findings of the previous sections in terms of use cases realizability and feature possession.

Use Case Based Analysis (F1-F5) The use case analysis has highlighted Home Assistant is the highest scoring system, followed closely by openHAB; ioBroker is in third place, and Domoticz has the lowest score, as summarized in Table 4.5. Each system supports the basic operations that one may expect from an IoT hub, such as reading sensor data, sending commands to actuate devices, and creating automation rules. Though, none of the systems have native support for ePaper and eInk displays. That said, almost all functionality that is not supported natively can be provided by third party extensions or plugins. This highlights the strong points of open-source and extensible IoT hubs, which is extensibility and the ability for anyone to contribute to the project. It is of note to mention that the use cases were defined before any practical evaluation, and that the use cases are considered to be comprehensive for smart buildings, for both residential and non-residential buildings. The fact that all systems are able to support every use case, either natively or through extension, demonstrates the maturity of these systems.

Popularity and Community (C1) The popularity among developers, as well as users, are an important factor to consider when choosing a system, as these factors influence the available support and longevity of the project. Table 4.23 summarizes the results of the Popularity and Community category, which consists of three individual criteria. While all systems are actively developed by their community of developers, there is a clear difference in overall popularity. Home Assistant takes the lead in both Developer Popularity, as well as Overall Popularity. Interestingly, while openHAB appears to be the least popular among developers, their overall popularity is second highest.

Table 4.23: Popularity and community score.

Criterion	**Home Assistant**	**Domoticz**	**openHAB**	**ioBroker**
C1.1	5	5	5	5
C1.2	5	3	2	3
C1.3	5	2	4	3
Average	5	3.3	3.7	3.3
Rank	**1st** (5.0)	**3th** (3.3)	**2nd** (3.75)	**3rd** (3.3)

Pricing (C2) The upfront costs can be a significant hurdle when first purchasing an IoT hub. Deciding on an open-source solution instead of a commercial system can help drastically reduce them, while sacrificing paid support. Table 4.24 provides an overview of the scores related to the Pricing category, considering both support plans and hardware costs. First of all, none of the systems provides paid support plans. For any kind of assistance, one has to rely on the community of developers and users. While commercially available closed-source IoT hubs are expensive (€250-€600), the hardware required to deploy the open-source systems considered in this work are significantly cheaper. All four systems can be deployed on the Raspberry Pi single board computer. The minimum and recommended hardware requirements are nearly identical for all systems. Note that we excluded the cost of the power supply and the non-volatile memory card, as these costs are identical for all four systems.

Table 4.24: Pricing score.

Criterion	**Home Assistant**	**Domoticz**	**openHAB**	**ioBroker**
C2.1	0	0	0	0
C2.2	4	5	4	4
C2.3	3	4	4	3
Average	2.3	3	2.6	2.3
Rank	**2nd** (2.3)	**1st** (3.0)	**4nd** (2.6)	**2nd** (2.3)

Setup Complexity (C3) The complexity of a system is an important aspect to consider, while uncomplicated systems make it easy to connect hardware to the system, a more complex system may offer more options and functionality that can be configured. Table 4.25 presents the results of the Setup Complexity category. While all systems are easy and quick to start from a fresh setup, thanks to Docker support, there are still some differences in the time it takes to start the Docker container. openHAB starts the fastest, in 69 seconds. Whereas ioBroker takes the longest, with 94 seconds. A bigger discrepancy can be found in the number of steps it takes to setup a basic MQTT sensor. In Domoticz this only takes 3 steps, whereas ioBroker requires 11 steps.

Table 4.25: Setup complexity score.

Criterion	**Home Assistant**	**Domoticz**	**openHAB**	**ioBroker**
C3.1	3	3	4	3
C3.2	4	5	3	2
Average	3.5	4	3.5	2.5
Rank	**2th** (3.5)	**1st** (4.0)	**2nd** (3.5)	**4rd** (2.5)

User Interface and User Experience (C4) Many of the interactions with the IoT hub takes place through the web interface provided. Therefore, the experience and functionality that the interface offers are important when deciding which system to adopt. Table 4.26 shows the results for the User Interface (UI) and User Experience (UX) categories. While Home Assistant and Domoticz require nearly the same number of clicks to setup an automation rule (around 40), openHAB requires nearly double that amount (70 clicks). The time it takes to setup an automation rule also widely differs, from 60 seconds for Home Assistant, to 136 seconds for openHAB. When it comes to extensibility of the UI, Home Assistant and openHAB support this functionality natively, whereas the other two systems have third party support for extending the UI. Domoticz is the only system which has a number of issues with its responsive design, making it less prone to be used on mobile and handheld devices. Unfortunately, C4.1 and C4.2 could not be evaluated for ioBroker, due to an error in one of the modules.

Security and Authentication (C5) Home automation systems deal highly sensitive data in terms of privacy, which should be managed with care. Table 4.27 presents the results of the Security and Authentication category, which consists of three criteria. All four systems provide support for basic-

Table 4.26: UI and UX score.

Criterion	Home Assistant	Domoticz	openHAB	ioBroker
C4.1	5	4	3	N/A
C4.2	5	4	3	N/A
C4.3	5	3	5	3
C4.4	5	3	5	5
Average	5	3.75	4	2
Rank	**1st** (5.0)	**3rd** (3.75)	**2nd** (4.0)	**4th** (2.0)

auth. Only Home Assistant provides MFA/2FA and webauthn support for their self-hosted solution. The other systems provide MFA/2FA only in their cloud-hosted solutions. Perhaps unsurprisingly, all four systems have support for multiple user accounts. Authorization management is in general lacking; in most systems there is either only two predefined roles (user and admin), or just one single role that provides everyone with the same level of access. The main exception here is ioBroker, which provides the most detailed level of control over permissions.

Table 4.27: Security and authentication score.

Criterion	Home Assistant	Domoticz	openHAB	ioBroker
C5.1	5	3	3	3
C5.2	5	5	5	5
C5.3	0	3	0	5
Average	3.3	3.6	2.6	4.3
Rank	**3rd** (3.3)	**2nd** (3.6)	**4th** (2.6)	**1st** (4.3)

Extensibility and Support (C6) An IoT hub should provide support for as many devices and protocols as possible. Though in practice not every system supports every device or protocol. While popular protocols, such as

Z-Wave and ZigBee, are widely supported, more obscure or niche devices and protocols may not be supported. Table 4.28 summarize the result for the Extensibility and Support Category. While all systems provide support for extensions, the number of available extensions varies widely. Home Assistant has over 1.600 extensions available, whereas Domoticz has about 80. The quality of the documentation also varies widely, with Home Assistant having excellent documentation, while ioBroker's documentation is clearly lacking, especially when it comes to completeness and the inclusion of examples. All of the systems rely heavily on the community to provide support to users. The support is often provided via forums or social media.

Table 4.28: Extensibility and support score.

Criterion	**Home Assistant**	**Domoticz**	**openHAB**	**ioBroker**
C6.1	5	5	5	5
C6.2	5	2	3	4
C6.3	5	4	4	3
C6.4	3	3	5	5
Average	4.5	3.5	4.3	4.3
Rank	**1st** (4.5)	**4rd** (3.5)	**2st** (4.3)	**2nd** (4.3)

System Performance (C7) Current trends show that there is an increase in the number of devices and appliances that can be connected to IoT hubs [SKS+20]. Therefore, it is important that the system is able to scale to support this increasing number of smart devices. Table 4.29 provides an overview of the results for the System Performance category. The results show that ioBroker is best suited to handle many sensors working concurrently. Domoticz appears to struggle with more than 10. All systems support at least up to 100 messages per second from a single sensor. Horizontal scalability is lacking in Home Assistant and Domoticz, whereas both ioBroker and openHAB support it.

Table 4.29: System performance score.

Criterion	Home Assistant	Domoticz	openHAB	ioBroker
C7.1	4	2	3	5
C7.2	5	5	5	5
C7.3	0	0	5	5
Average	3	2.3	4.3	5
Rank	**3rd** (3.0)	**4th** (2.3)	**2nd** (4.3)	**1st** (5.0)

Software Quality (C8) The quality of software is not trivial to quantify, however, there are indicators to the quality such as the presence of tests (unit, e2e, etc.), lines of code, and the general processes and checks in place when contributing to a project. Table 4.30 summarizes the results for the Software Quality category. Source code reviews are present for all four systems. Interestingly, none of the systems performs end-to-end testing, which could benefit the overall user experience. Though unit tests are performed by all systems with the exception of Domoticz. ioBroker's lines of code are significantly less than for the other systems, with Home Assistant having by far the most lines of code. Quality checks are also in place for third party plugins, though Domoticz's checks are not as elaborate as the checks that are in place for the other systems.

Discussion and limitations While at first glance all four systems appear to offer very similar functionalities, as deduced from the use case based analysis, on closer inspection, several differences emerge. These differences are visualized in Table 4.31, which presents an overview of the satisfaction of the criteria for each one of the top four systems. The results show quite some variance for each criteria going from two to five stars for some of them, e.g., for criteria C7. This indicates that the systems are not equivalent and making an informed choice when selecting one will have consequences for the effort and success of a building automation project. Furthermore,

Table 4.30: System software quality score.

Criterion	Home Assistant	Domoticz	openHAB	ioBroker
C8.1	5	5	5	5
C8.2	0	0	0	0
C8.3	2	3	4	5
C8.4	5	5	5	5
C8.5	5	3	5	5
C8.6	5	0	5	5
Average	3.7	2.7	4	4.2
Rank	**3rd** (3.7)	**4th** (2.7)	**2nd** (4.0)	**1st** (4.2)

the importance of each criteria may differ based on the desired application; some deployments may desire high performance, while others value the support of a wide range of devices. High performance and low resource utilization may be preferable where edge gateways for data centers are concerned. Therefore, the criteria are not weighted: this should be done on a per-deployment basis.

A limitation of our analysis is that obtained results are a snapshot of a specific moment in time. Specifically, the majority of data was collected in the month of July, 2021. The code base of these systems, especially for open-source projects, is highly volatile and ever changing as many contributors are continuously making changes to the source code on a daily basis. That said, the underlying architecture of the systems is fundamental and therefore not as susceptible to continuous changes. We therefore do not expect the common architecture that emerges from the four systems to be subjected to significant changes in the near future. As the systems continuously change, it is challenging to obtain a snapshot of critical privacy and security vulnerabilities. Furthermore, these vulnerabilities are also strongly influenced by the deployment environment: from the connected IoT hardware, to the configuration of the operating system on which the software is installed. Therefore, the analysis of vulnerabilities is not included in the analysis.

Table 4.31: Overview of all categories.

Criterion	**Home Assistant**	**Domoticz**	**openHAB**	**ioBroker**
F1-5	4.5	2	4	3
C1	5	3.3	3.75	3.3
C2	2.3	3	2.6	2.3
C3	3.5	4.0	3.5	2.5
C4	5	3.75	4	2
C5	3.3	3.6	2.6	4.3
C6	4.5	3.5	4.3	4.3
C7	3.0	2.3	4.3	5
C8	3.7	2.7	4	4.2
Average	3.9	3.1	3.7	3.4
Rank	**1st** (3.9)	**4th** (3.1)	**2nd** (3.7)	**3rd** (3.4)

4.2.7 Conclusions

The selection of available IoT hubs is vast, making an informed decision when designing a smart building or smart data center is difficult. Not only because the available systems offer different functionalities, but also because retroactively migrating from one system to another one is not a trivial task. This means that the expected longevity of the open-source project is of importance too. We have addressed these concerns by providing an overview of 20 open-source IoT hubs. Out of the 20 systems, we investigate the four most popular ones in greater detail, where popularity is determined by their commit activity, stars, time of latest commit, documentation availability, and number of contributors.

For each of the four selected systems, their architecture is analyzed and seven distinct components that all systems have in common are identified, compared, and discussed. We deem these components to be essential parts of a modern building automation system architecture. We also uncovered many parallels between the reference architecture and the proposed edge gateways for data centers. Furthermore, the use case based analysis shows that almost all systems support the list of thirteen essential features. The support is either native or through third party extensions, in turn highlighting the importance of an extensible IoT hub. A further analysis is done based on thirty four criteria. This criteria-based analysis reveals the strengths and weaknesses of the selected systems. With Home Assistant presenting a very solid user experience, Domoticz providing an uncomplicated setup, openHAB having strong scores in all categories, and ioBroker offering great system performance. It is for the practitioner or requirements engineer to assign the appropriate weights to each criteria, as each deployment will have distinct requirements. Significant shortcomings among the systems were also identified, and include: the lack of role-based access control, no horizontal scalability, and no options for enterprise or paid support. Finally, we have also described the common architecture that emerges from the design of the four systems, identifying the key components of which an IoT hub is comprised.

The work presented in this section serves several purposes. The basic service is that of inventorizing existing open-source IoT hubs. The second one is to provide meaningful features to actually evaluate these kind of systems. The features and methodology adopted may also be useful for research in other related fields beyond building automation systems. The final, and perhaps, more relevant service is that of acting as a decision support tool for the practitioner and for the developer who intend to deploy and contribute to open-source IoT hubs. It is clear that IoT hubs have an important role in the realization of smart buildings and smart data centers. The next step is to demonstrate the capabilities of IoT in a real-world data center environment.

4.3 Data-Driven Approach to Monitoring Co-location Data Centers

The Internet of Things (IoT) allows for the effortless and inexpensive collection of large quantities of data regarding many different environmental aspects. In the case of data centers, IoT can be applied to the IT infrastructure by employing a wide variety of sensors. From the temperature and humidity levels of entire halls, to detailed thermal maps for individual server racks, to measuring the power consumption of each individual piece of IT equipment. As we have seen in Chapter 3, collecting such a large data set is useful for evaluating the many metrics and key performance indicators that can be applied to data centers. Utilizing IoT for data collection and analysis also leads to improvements in the data center's service levels by use of predictive maintenance [SG16]. Another useful application of IoT in the data center context is detecting so-called comatose or zombie servers. Comatose servers are servers which are no longer in use and serve no useful purpose, yet are responsible for a significant portion of the data center's energy consumption. Detecting and removing these types of servers can greatly increase the energy efficiency of the data center by decreasing the total energy consumption of servers and reducing the cooling load.

The application of IoT in the data center domain leads to a curious clash of requirements. From a data center operator point-of-view, knowing every detail regarding the infrastructure down to individual servers would allow the operator to monitor the entire data center and take well-founded decisions. However, full knowledge of such data does affect privacy and security [MSFM16]. When collecting and processing highly sensitive data, for example medical data, privacy is of the utmost importance. Exposing server metrics to the data center operators is a potential security threat that could make the server vulnerable to attacks. Therefore, it is preferable that the server remains a black box to minimize external access to the operating system. Another aspect to consider is that co-location data centers face challenges when collecting data from servers in their data center. In a co-location center, rack space is rented out to customers who store their own servers and other hardware in the racks they have rented. It is even possible for a single rack to contain servers belonging to numerous distinct customers. Due to privacy concerns, it is not possible for the co-location center to monitor the internal state of these servers, as physical access and operating system access remain exclusively with the customer. This is one of the challenges we have identified in Chapter 3. The question that emerges is: can we employ an Internet of Things approach to monitor the state of a server without requiring direct access to the operating system or server chassis? Having a positive answer would enable non-intrusive monitoring of co-location data centers, as privacy is fully preserved from the data center perspective.

To answer the question, we conduct a large scale experiment where we collect data from over 160 High Performance Computing (HPC) servers in a single data center over a period of 5 months. These servers are used by researchers from the University of Groningen for data-intensive, computationally expensive, and privacy sensitive jobs. During these 5 months, the data set has grown to 2.5 billion data points, which is almost 18 million data points every day. The data set considers a multitude of metrics, ranging from CPU usage to inlet and outlet air temperatures.

The main contribution in this section is the proposal of a method for privacy preserving monitoring of data centers. The objective is to determine the internal state of a server by monitoring external, environmental parameters using IoT. We distinguish between internal and external metrics. Internal metrics require access to the operating system or chassis of the server, while external metrics can be measured without such access. An internal metric is, for example, the CPU usage of a server, as it is measured by the operating system. An example of an external metric is the power consumption of a server, which can be measured externally by a metered Power Distribution Unit (PDU). We utilize these internal and external metrics to design and develop models that depend on external metrics as input features in order to estimate the internal state of a server. We prove that it is feasible to reason about the internal state of servers using external parameters, thus enabling privacy preservation when monitoring co-location data centers.

4.3.1 Data Center

The University of Groningen has two on-campus data centers that house the university's IT infrastructure, in addition to a number of HPC clusters and co-located servers. The servers which we monitor are located in an off-site data center. The cluster in question, also known as the Peregrine cluster, consists of 165 blade servers. Each server has two Intel Xeon E5-2680 v3 CPUs and 128 gigabytes of RAM, bringing the total to 3960 individual CPU cores and 21 terabytes of RAM respectively. Each server has 1 terabyte of internal disk space, and is connected to a storage area network consisting of 463 terabytes. The servers are locally connected by a 56 gigabit per second Infiniband network connection, and are also connected by a 10 gigabit per second ethernet connection to the Internet.

The cluster is utilized by researchers and staff of the University of Groningen to assist in computationally intensive jobs, such as processing large quantities of astronomical data and other Big Data oriented tasks. For example, the cluster is used in the genome analysis of almost 250,000 adults to identify correlations in the human genome between smoking behavior and obesity [JWF+17]. Over the years, the Peregrine cluster has assisted in more than 150 scientific publications.

4.3.1.1 Data Acquisition

Each server in the HPC cluster reports around 150 individual metrics to Ganglia, a monitoring system for HPC applications and Big Data grid computing clusters. We select a subset of 13 metrics for detailed collection and analysis. The selection is made to exclude all aggregated metrics. We make this distinction such that we focus on the raw metrics, which can be used to derive the aggregated metrics if necessary. Table 4.32 shows the selected metrics and their unit. Servers report each metric every 10 seconds to Ganglia, where it is aggregated after 1 hour in order to reduce the storage space requirements. We require the collected data to be as granular as possible. Fortunately, Ganglia exposes the data through a JavaScript Object Notation (JSON) API, which we use to collect and store the raw data with a granularity of 10 seconds. The collection of data started on the 1st of December 2016. Every month we, collect around 560,000,000 individual data points from 164 servers. For this work, a subset of approximately 2,500,000,000 data points was used.

Table 4.32: Selected server metrics.

Metric	**Name**	**Unit**
	Internal Metrics	
T_{cpu}	CPU Temperature	Degrees Celsius
CPU_{user}	CPU Utilization (User)	Percentage
CPU_{sys}	CPU Utilization (System)	Percentage
Mem_{free}	Unused Random Access Memory	Kilobytes
$Proc_{run}$	# Processes Running	Integer
$Proc_{total}$	# Processes Total	Integer
Eth_{in}	Ethernet In	Bytes per Second
Eth_{out}	Ethernet Out	Bytes per Second
Inf_{in}	Infiniband In	Bytes per Second
Inf_{out}	Infiniband Out	Bytes per Second
	External Metrics	
P_{watts}	Power Consumption	Watts
T_{in}	Inlet Air Temperature	Degrees Celsius
T_{out}	Outlet Air Temperature	Degrees Celsius

For the collection, processing, and storage of data, we utilize our own service-oriented IoT hub as shown in Figure 4.12. We developed our own platform as we require a generic, scalable solution for streaming and processing high volume data. The arrows in the figure indicate the general flow of data. The platform is optimized for throughput, as we need to handle tens of millions of data points on a daily basis. To collect the metrics from *Ganglia*, we develop the *Ganglia Service*, that regularly scrapes the data from Ganglia's JSON API. The data is published to *Kafka*, a distributed stream processing platform. The data is consumed by the *Data Collector Service,* which collects the raw data and inserts it into the database. For time-series data storage we employ *Cassandra*, a distributed NoSQL database management system. The data is made accessible by the *Data Access Service* to be consumed by an *External Application*. We also make use of *Spark*, a cluster-computing framework for data processing. The data processed by these Spark Jobs can originate directly from a Kafka topic, from the Cassandra database, or a combination of both. For hierarchically structured data we utilize *OrientDB*, a distributed graph database. For example, the different metrics collected by Ganglia all have their own unique data type identifier. These data types form a type hierarchy which is preserved in OrientDB. The Ganglia Service, and other services which generate data, register these new (data) types in OrientDB. All services also register themselves with *Consul*, a distributed service discovery and key-value store.

Even though we design our own IoT hub, there are still many parallels between our proposed IoT hub and the IoT hubs discussed in Section 4.2. The physical layer is represented by Ganglia and the conversion between physical device and virtual device is done by the Ganglia Service. Kafka takes the role of the event bus. The historical data and device states are stored in Cassandra, and the meta data is stored in OrientDB. And finally, the data access service exposes a REST API for reading data from the Cassandra database.

4.3.1.2 Data Sanitation

We collect the data set using many sensors of which some are internal metrics, such as RAM utilization as reported by the operating system, and some are external metrics, such as the air temperature as measured by hardware sensors. In order to prepare the data set for analysis, we apply the following sanitation steps: (1) combine CPU_{user} and CPU_{system} to get the CPU_{total} such that $0.0 \leq CPU_{tot} \leq 100.0$, (2) subtract T_{in} from T_{out} to obtain T_{diff}, the temperature difference between inlet and outlet, (3) detect and remove outliers / faults introduced by measurements errors, (4) compute

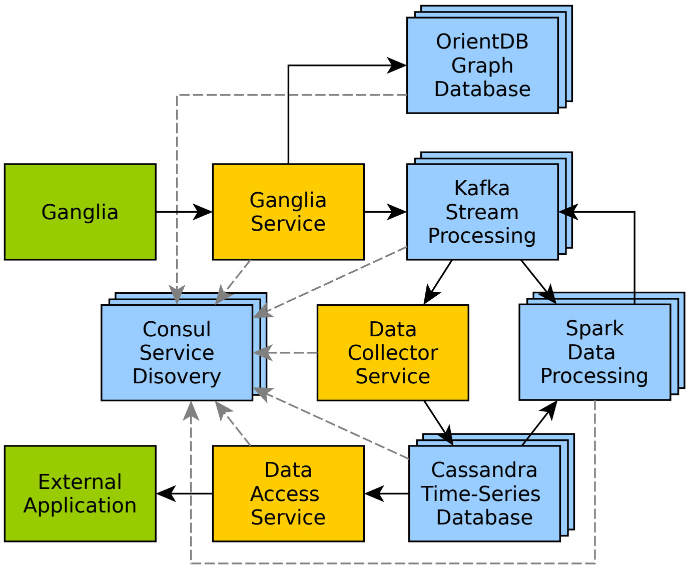


Figure 4.12: Service-oriented IoT hub. Green: external services/applications. Orange: internal services. Blue: core infrastructure. Arrows indicate main data flow.

the correlation between every pair of metrics, and (5) determine the lag between time-series of metrics that will be part of the models.

Table 4.33: Kendall's tau correlation, including standard deviation.

	T_{cpu}	P_{watts}	$Proc_{run}$	CPU_{tot}
P_{watts}	0.76 ± 0.07	-		
$Proc_{run}$	0.54 ± 0.11	0.62 ± 0.13	-	
CPU_{tot}	0.73 ± 0.08	0.83 ± 0.08	0.70 ± 0.13	-
T_{diff}	0.78 ± 0.11	0.82 ± 0.09	0.57 ± 0.12	0.76 ± 0.07

Step 1 and 2 are simple mutations of the data set. In the first step, we combine the CPU usage in user space (CPU_{user}) and kernel space (CPU_{sys}), the combination of these metrics gives us the total CPU usage (CPU_{tot}). In the second step, we subtract the temperature of the cold air entering the server (T_{in}) from the temperature of the hot air exiting the server (T_{out}) in order to obtain the temperature difference (T_{diff}).

In the third step, we remove the faulty measurements from the data set. If a row in the data set does not conform to the following constraints, it is discarded: $T_{cpu} > 0$, $P_{watts} > 0$, and $\sum_{m \in M} m > 0$, where m is a measurement and M is the set of measurements belonging to a row.

Inspection of the data set shows that P_{watts} and T_{cpu} are sometimes 0, while other metrics are not. This indicates a measurement error. Thus, the first and second constraints are required. The final constraint ensures that rows with all zeros are excluded, by verifying that the sum of all metrics is not zero. These all-zero rows appear when the Ganglia monitoring system is unavailable.

For step 4, the goal is to identify which metrics are correlated, and to what extent. This allows us to perform feature selection and use the appropriate set of metrics for our models. We determine the correlation between metrics using Kendall's tau coefficient [Dod08] for each individual node:

$$\tau = \frac{n_c - n_d}{n(n-1)/2} \tag{4.6}$$

Where n_c is the number of concordant pairs, n_d is the number of discordant pairs, and n is the number of pairs. When applied to the data we immediately notice that there are numerous metrics with very weak correlations all-around ($-0.5 < \tau < 0.5$): T_{in}, $Avail_{ram}$, $Proc_{tot}$, Eth_{in}, Eth_{out}, Inf_{in} and Inf_{out}. We discard the metrics with a weak correlation, as they would have little to no contribution to our models.

There is a very high correlation between Inf_{in} and Inf_{out}, the Infiniband connection for inter-node communication. This is likely caused by the fact that when the inter-node communication is used, it is used to transfer data back and forth between nodes. However, for the purpose of this paper we also discard these metrics as they only have a strong correlation among themselves, while they are both considered internal metrics. Table 4.33 provides the resulting mean correlation and the associated standard deviation of the remaining metrics, where the mean is taken over the correlation results for each individual server. The remaining metrics consist of two external metrics (T_{diff} and P_{watts}) and three internal metrics (T_{cpu}, $Proc_{run}$, and CPU_{tot}).

Table 4.34: Metrics characteristics.

	$\mathbf{T_{cpu}}$	$\mathbf{P_{watts}}$	$\mathbf{Proc_{run}}$	$\mathbf{CPU_{tot}}$	$\mathbf{T_{diff}}$
Mean	56.06	263.73	18.02	62.36	13.96
Std. Dev.	12.51	70.24	24.12	37.43	3.29
Min.	24.00	77.00	0.00	0.00	0.00
25%	34.00	196.00	12.00	29.22	11.00
50%	43.00	280.00	22.00	79.20	15.00
75%	60.00	336.00	24.00	100.00	17.00
Max.	100.00	448.00	2867.00	100.00	25.00

The characteristics of the remaining metrics are described in Table 4.34. For each of the metrics we identify the mean value and corresponding standard deviation. We also include minimum and maximum values observed, and the percentile statistics. Based on these characteristics we can conclude that the data set is representable, as it covers the full range of possible values that one can expect for all of the metrics. For example, CPU temperature (T_{cpu}) ranges from $24.0°C$ to $100.0°C$, as the thermal throttling threshold is commonly set between $90°C$ and $100°C$. The power consumption (P_{watts}) also has a decent spread, ranging from 77 watts, when idle, to 448 watts, under full load. The number of user-space processes running ($Proc_{run}$) shows a large spike as the maximum observed value: 2867 processes running. Perhaps there was a mistake in one of the submitted jobs, starting too many processes. The CPU usage (CPU_{tot}) is not as evenly spread as other metrics, we commonly observe the CPU usage to be at a 100%. This is expected for a

HPC cluster, as Big Data jobs are CPU intensive. The temperature difference between inlet and outlet air (T_{diff}) varies from $0.0°C$ to $25.0°C$.

Now that we have reduced the set of metrics to a set of highly correlated metrics, we make our final selection with regards to which external metrics we will utilize when modeling the internal state of a server. The metrics that describe the internal state of a server are CPU_{tot}, T_{cpu}, and $Proc_{run}$. However, we exclude $Proc_{run}$ as it is too specific to the tasks that a server is performing, and therefore does not generalize well in different environments. For example, a single process could have a CPU usage of 100%, while the same hold true for 100 processes with 1% CPU usage each. Furthermore, the correlation between $Proc_{run}$ and the external metrics are significantly lower than the other correlations we have observed in Table 4.33.

The internal metrics that remain are CPU_{tot} and T_{cpu}, representing the state of the server. The external metrics that remain are P_{watts} and T_{diff}, as these can be measured externally. This leaves the following models to be explored:

- Model CPU_{tot} using T_{diff} and / or P_{watts}
- Model T_{cpu} using T_{diff} and / or P_{watts}
- Model T_{diff} using P_{watts}

Thus in total we have 7 models to evaluate. The last model utilizes one external metric (P_{watts}) to estimate another external metric (T_{diff}). This model does not provide any insights in the internal state of a server, but is interesting none the less from a data center perspective, as it estimates the increase in temperature which has to mitigated by the cooling system.

Finally, we consider the notion of lag between two discrete time-series, this relates to step 5. To demonstrate the effect of lag, we normalize the previously selected time-series and visualize them in Figure 4.13. We can clearly observe that there is a certain amount of lag between the time-series. The cause of this lag is twofold: there is a delay between individual observations, and certain time-series have a natural tendency to lag behind.

For example, an increase in T_{cpu} would eventually lead to an increase in T_{diff} (as exhaust temperatures increase). However, it takes a certain amount of time for the heat to dissipate, which causes lag between the two time-series.

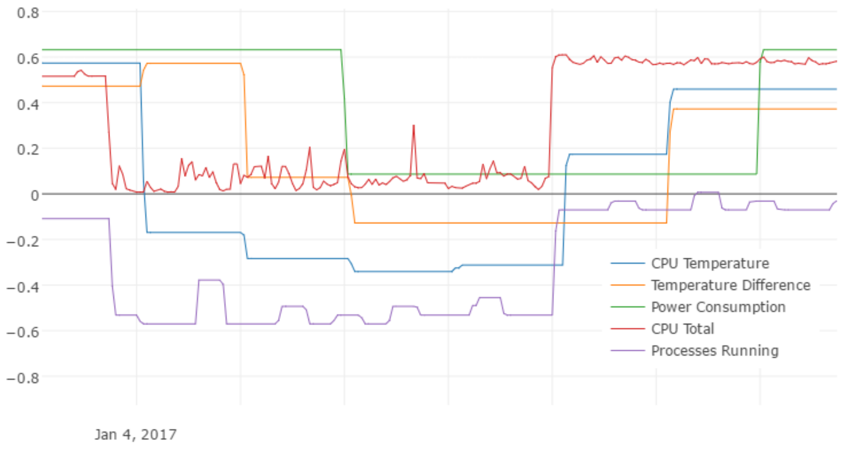


Figure 4.13: Lag observed between selected, normalized time-series ($[-1,1]$).

Intuitively, decreasing the lag between time-series increases the number of concordant pairs, and consequentially increases the correlation. A higher correlation between time-series is of benefit when modeling the data. To determine the exact amount of lag between two metrics, we look at the cross-correlation. Given two discrete time-series $x[m]$ and $y[m]$, the cross-correlation is defined as [RG75]:

$$R_{xy}(k) = \sum_{m=-\infty}^{\infty} x[m]y[m-k] \tag{4.7}$$

Where $k \in \mathbb{Z}$, and $-\infty \leq k \leq \infty$. Parameter k is also known as the lag parameter. For each of the selected metrics we determine the lag that exists between them by performing cross-correlation for every pair of time-series and selecting the k which maximizes (since the time-series are positively correlated) $R_{xy}(k)$ such that:

$$lag_{units} = \arg\max_{k} R_{xy}(k) \tag{4.8}$$

We calculate the lag individually for every server and adjust the time-series accordingly. Shifting the time-series based on the lag further increases Kendall's tau correlation by the values show in Table 4.35.

We collected data from 164 servers, of which 15 servers recorded unusable data due to faulty configurations. Resulting in a data set consisting of 149 servers in total. We also identified the external metrics that are to be used to model the internal state of a server, leaving 7 distinct models to investigate.

Table 4.35: Increase in correlation between metrics.

	$\mathbf{T_{diff}}$	$\mathbf{P_{watts}}$
$\mathbf{CPU_{tot}}$	+0.007	+0.015
$\mathbf{T_{cpu}}$	+0.003	+0.006
$\mathbf{T_{diff}}$	-	+0.000

4.3.2 Models and Evaluation

In Section 4.3.1, we concluded that we have seven candidate models to estimate a server's internal state. In this section, we describe how we design and train these candidate models, and summarize the results for evaluating each model. We distinguish between two types of models: the individual models and the universal model. Individual models are models which we train separately for each individual server, using only the subset of data that belongs to a specific server. For the universal model, we train and evaluate one global model using all available data, of every server. In both cases, the training and evaluation process is almost identical.

Inspection of the data set shows that there is a near-linear correlation between the external metrics (P_{watts}, T_{diff}) and the internal metrics (CPU_{tot}, T_{cpu}). Therefore, we define a linear regression model of at most two parameters (Eq. 4.9):

$$\hat{y} = w_0 + w_1 x_1 + w_2 x_2 \tag{4.9}$$

Where x_1 and x_2 are the parameters of the model, w_0, w_1, w_2 are the weights, and $\hat{y}$ is the estimated value. Next, since the data is not fully linear, we transform the features into polynomial features. For every model, we generate all polynomial features from the 1st degree to the 10th degree. Given two features x_1 and x_2, the transformation to polynomial features of the second degree is shown in Equation 4.10.

$$z = [x_1, x_2, x_1 x_2, x_1^2, x_2^2] \tag{4.10}$$

The resulting model is a linear regression model in which the features are polynomial (Eq. 4.11). Thus we can apply any linear regression technique, such as ordinary least squares fitting, to obtain our polynomial model.

$$\hat{y} = w_0 + w_1 z_1 + \cdots + w_5 z_5 \tag{4.11}$$

To train our models we apply the Ridge Regression method, which is expressed as ordinary least squares, the first term in Equation 4.12, with an additional L2 regularization term, the second term in the equation. The α parameter controls the strength of the regularization.

$$\min_{w} ||Xw - y||^2 + ||\alpha w||^2 \tag{4.12}$$

Finally, to evaluate how well our models generalize, we apply k-fold cross-validation with $k = 10$ [Koh+95]. We evaluate both the Root Mean Squared Error (RMSE) and the Coefficient of Determination (R^2) obtained using cross-validation. Next, we evaluate the individual models, followed by the universal models. The results are discussed in Section 4.3.3. The tables which summarize the results distinguish between $RMSE_i$ and R_i^2 for the individual models, and $RMSE_u$ and R_u^2 for the universal model.

4.3.2.1 Individual Models

To evaluate all of the individual models we train models for each of the 149 servers. For each of the seven models, we train 10 distinct iterations where we vary the polynomial degree from 1 to 10. In total we train ($149 \times 7 \times 10 =$) 10,430 distinct polynomial models. When we summarize the results, we discard the results from polynomial degrees 7 till 10, as in nearly all cases the models start to overfit at the 5th or 6th degree. The exception to this observation are commonly models with only a single feature. For these models, as the number of degrees approaches the number of data points, the error approaches 0.

Individual Total CPU Usage (CPU_{tot}) First, we evaluate the results for modeling CPU_{tot}, the total CPU usage. The summary of the results can be found in Table 4.36. In this table we list the mean $RMSE_i$ and the mean R_i^2 metrics, as well as their standard deviations, for every selection of input features. The input features are the external parameters as selected in Section 4.3.1. For feature P_{watts}, we observe a significant improvement in $RMSE_i$ and R_i^2 between the 1st and 3rd degrees, with $RMSE = 8.13$ and $R^2 = 0.95$. For feature T_{diff}, the same improvement is observed at the 3rd degree, where $RMSE_i = 12.45$ and $R_i^2 = 0.89$. Furthermore, we can clearly see that the models start to overfit from the 6th degree, as the standard deviation of $RMSE_i$ increases, and the R_i^2 significantly decreases. The last features we evaluate to model CPU_{tot} is the combination of P_{watts} and T_{diff}. As observed in previous cases, a notable improvement happens until the 3rd degree, where $RMSE_i = 7.68$ and $R_i^2 = 0.96$. Improvements are observed until the 5th degree, after which the models overfit as the increased standard deviation of the $RMSE_i$ and the lower R_i^2 signify. The optimal result is obtained using both P_{watts} and T_{diff} as features to model the CPU_{tot} with a polynomial of the 5th degree. This results in a mean $RMSE_i$ of 7.35%.

Individual CPU Temperature (T_{cpu}) Next, we analyze the results we obtained when modeling the T_{cpu}, the CPU temperature. A summary of these results is shown in Table 4.37. Again we list the mean $RMSE_i$ and R_i^2 that was obtained individually for all 149 servers. First, we model T_{cpu} using only P_{watts} as a feature. Again, we observe a significant improvement at the 3rd degree, where the $RMSE_i = 4.74$ and the $R_i^2 = 0.85$. A similar decrease in $RMSE_i$ is noted when only applying feature T_{diff}. At the 3rd degree we observe $RMSE_i = 3.51$ and $R^2 = 0.92$. At the 6th degree the models start to overfit; the $RMSE_i$ increases, as does the standard deviation. When using both P_{watts} and T_{diff} as features, we observe $RMSE_i = 3.23$ and $R_i^2 = 0.93$ for the 3rd degree. Significant over-fitting occurs after the 5th degree, with R_i^2 becoming negative which indicates that the model fits the data extremely poor. The optimal results are obtained when using both P_{watts} and T_{diff} as features, a polynomial of the 5th degree models T_{cpu} with a mean $RMSE_i$ of $3.17°C$.

Individual Temperature Difference (T_{diff}) Finally, we assess the results of modeling T_{diff}, the difference between inlet and outlet temperature of a server. The results are summarized in Table 4.38 In this case, there is only one feature that has a strong correlation, that is P_{watts}. We do not observe a significant increase at the 3rd degree, as we noted in previous models. The $RMSE_i$ and R_i^2 improve ever so slightly as the number of degrees increases. The optimal results are observed at the 10th degree, where the mean $RMSE_i$ of the models is $1.07°C$. It is likely that these models continue to improve slowly as the number of degrees increases.

Table 4.36: Summary of modeling CPU_{tot}, $i =$ individual models, and $u =$ universal model.

Degree	**1st**	**2nd**	**3rd**	**4th**	**5th**	**6th**
Features: P_{watts}						
$RMSE_i$	10.02 ± 0.84	9.95 ± 0.87	8.13 ± 0.85	8.08 ± 0.85	8.05 ± 0.86	8.02 ± 0.86
R_i^2	0.92 ± 0.03	0.93 ± 0.03	0.95 ± 0.03	0.95 ± 0.03	0.95 ± 0.03	0.95 ± 0.03
$RMSE_u$	10.68 ± 0.53	10.68 ± 0.53	8.71 ± 0.34	8.67 ± 0.32	8.67 ± 0.32	8.66 ± 0.32
R_u^2	0.96 ± 0.01	0.96 ± 0.01	0.97 ± 0.00	0.97 ± 0.00	0.97 ± 0.00	0.97 ± 0.00
Features: T_{diff}						
$RMSE_i$	14.25 ± 1.06	14.16 ± 1.18	12.45 ± 1.16	12.37 ± 1.21	12.35 ± 1.26	13.13 ± 5.77
R_i^2	0.85 ± 0.04	0.85 ± 0.03	0.89 ± 0.03	0.89 ± 0.03	0.89 ± 0.03	0.63 ± 2.10
$RMSE_u$	15.68 ± 0.92	15.66 ± 0.88	13.68 ± 0.87	13.64 ± 0.89	13.67 ± 0.95	13.51 ± 0.85
R_u^2	0.91 ± 0.02	0.91 ± 0.01	0.93 ± 0.01	0.93 ± 0.01	0.93 ± 0.01	0.94 ± 0.01
Features: P_{watts} and T_{diff}						
$RMSE_i$	9.71 ± 0.83	9.5 ± 0.86	7.68 ± 0.8	7.51 ± 0.79	7.35 ± 0.81	8.14 ± 5.89
R_i^2	0.93 ± 0.02	0.93 ± 0.02	0.96 ± 0.02	0.96 ± 0.02	0.96 ± 0.02	0.68 ± 2.27
$RMSE_u$	10.43 ± 0.47	10.30 ± 0.46	8.38 ± 0.32	8.30 ± 0.32	8.14 ± 0.30	9.10 ± 2.97
R_u^2	0.96 ± 0.01	0.96 ± 0.01	0.98 ± 0.00	0.98 ± 0.00	0.98 ± 0.00	0.97 ± 0.03

Table 4.37: Summary of modeling T_{cpu}, i = individual models, and u = universal model.

Degree	**1st**	**2nd**	**3rd**	**4th**	**5th**	**6th**
Features: P_{watts}						
$RMSE_i$	4.96 ± 0.54	4.84 ± 0.56	4.74 ± 0.55	4.72 ± 0.55	4.67 ± 0.55	4.65 ± 0.55
R_i^2	0.84 ± 0.04	0.85 ± 0.04	0.85 ± 0.04	0.85 ± 0.04	0.86 ± 0.04	0.86 ± 0.04
$RMSE_u$	5.32 ± 0.23	5.23 ± 0.25	5.17 ± 0.25	5.17 ± 0.25	5.13 ± 0.23	5.13 ± 0.23
R_u^2	0.91 ± 0.01	0.91 ± 0.01	0.92 ± 0.01	0.92 ± 0.01	0.92 ± 0.01	0.92 ± 0.01
Features: T_{diff}						
$RMSE_i$	4.04 ± 0.35	3.88 ± 0.39	3.51 ± 0.36	3.4 ± 0.35	3.39 ± 0.37	3.47 ± 0.71
R_i^2	0.89 ± 0.04	0.90 ± 0.03	0.92 ± 0.03	0.92 ± 0.02	0.92 ± 0.03	0.90 ± 0.17
$RMSE_u$	4.69 ± 0.30	4.61 ± 0.35	4.22 ± 0.31	4.15 ± 0.33	4.13 ± 0.34	4.13 ± 0.35
R_u^2	0.93 ± 0.01	0.93 ± 0.01	0.95 ± 0.01	0.95 ± 0.01	0.95 ± 0.01	0.95 ± 0.01
Features: P_{watts} and T_{diff}						
$RMSE_i$	3.90 ± 0.36	3.66 ± 0.39	3.23 ± 0.34	3.17 ± 0.36	3.21 ± 0.78	4.79 ± 10.17
R_i^2	0.90 ± 0.03	0.91 ± 0.02	0.93 ± 0.02	0.93 ± 0.03	0.89 ± 0.33	-6.58 ± 60.01
$RMSE_u$	4.42 ± 0.23	4.26 ± 0.28	3.88 ± 0.24	3.84 ± 0.25	3.85 ± 0.35	3.86 ± 0.34
R_u^2	0.94 ± 0.01	0.94 ± 0.01	0.95 ± 0.01	0.95 ± 0.01	0.95 ± 0.01	0.95 ± 0.01

Table 4.38: Summary of modeling T_{diff}, i = individual models, and u = universal model.

Degree	**1st**	**2nd**	**3rd**	**4th**	**5th**	**6th**	**...**	**10th**
Features: P_{watts}								
$RMSE_i$	1.10 ± 0.10	1.10 ± 0.10	1.09 ± 0.10	1.09 ± 0.10	1.09 ± 0.10	1.08 ± 0.10	...	1.07 ± 0.10
R_i^2	0.88 ± 0.04	0.88 ± 0.04	0.88 ± 0.04	0.89 ± 0.04	0.89 ± 0.04	0.89 ± 0.04	...	0.89 ± 0.03
$RMSE_u$	1.26 ± 0.10	1.26 ± 0.10	1.25 ± 0.10	1.25 ± 0.10	1.25 ± 0.10	1.25 ± 0.10	...	1.25 ± 0.10
R_u^2	0.93 ± 0.01	0.93 ± 0.01	0.93 ± 0.01	0.93 ± 0.01	0.93 ± 0.01	0.93 ± 0.01	...	0.93 ± 0.01

4.3.2.2 Universal Model

After evaluating all of the individual models in Section 4.3.2.1, it becomes apparent that the individual models appear to be very similar. Therefore, the next step is to reduce the ten-thousand individual models to less than a hundred universal models. Considering that large data centers can contain several hundreds of thousands of servers [Gen14], it would be advantageous to develop a universal model that represents a significant subset of servers. For each of the 7 models we train 10 distinct iterations where we vary the polynomial degree from 1 to 10. In total we train ($7 \times 10 =$) 70 distinct polynomial models.

Due to the sheer size of the complete data set (2.5 billion data points) compared to the data set for an individual server (15 million data points), we have to modify the approach to learning our model, as we cannot fit all 2.5 billion data points simultaneously. Instead of Ridge Regression, we fit the polynomial models using Stochastic Gradient Descent with L2 regularization [Bot10]. We partially fit the data of every individual server, until all the training data has been fitted.

Universal Total CPU Usage (CPU_{tot}) As before with the individual models, we first analyze the results for modeling CPU_{tot} with one universal model using three sets of features. When only using feature P_{watts}, we observe a significant decrease in $RMSE_u$ at the 3rd degree, where $RMSE_u = 8.71$ and $R_u^2 = 0.97$. For feature T_{diff} we record $RMSE_u = 13.68$ and $R_u^2 = 0.93$ at the 3rd degree, which shows the most significant improvement as well. And finally, when using both P_{watts} and T_{diff} as features, we obtain $RMSE_u = 8.38$ and $R^2 = 0.98$ at the 3rd degree. However in this case the the 5th degree shows slightly better results, whereas the 6th degree shows signs of overfitting. These results are in line with the results of the individual models for CPU_{tot}. We observe the optimal results using a polynomial of the 5th degree while using both P_{watts} and T_{diff} as features. This yields a $RMSE_u$ of 8.14%.

Universal CPU Temperature (T_{cpu}) We model T_{cpu} using the same three sets of features as used when modeling CPU_{tot}. First we use P_{watts} as a feature, which yields significant improvements up until the 3rd degree ($RMSE_u = 5.17$ and $R_u^2 = 0.92$), after which the $RMSE$ decreases much slower. Selecting T_{diff} as a feature yields better results, at the 3rd degree we obtain $RMSE_u = 4.22$ with $R_u^2 = 0.95$. After the 3rd degree we observe minor improvements in $RMSE$. When utilizing both P_{watts} and T_{diff} as features we get $RMSE_u = 3.88$ and $R_u^2 = 0.95$ at the 3rd degree, improving at the 4th degree after which the error slowly increases. The best results are obtained when using both features, a 4th degree polynomial gives a $RMSE_u$ of $3.85°C$.

Universal Temperature Difference (T_{diff}) The last universal model we train and evaluate is for modeling T_{diff}. The best result is observed when using a polynomial between the 3rd and 10th degree. This yields a $RMSE_u = 1.25$ and $R_u^2 = 0.93$. The error remains nearly constant after the 3rd degree, which results in an optimal $RMSE_u$ of $1.25°C$.

4.3.3 Discussion

Based on the evaluation of the models, we conclude that the individual models are, when compared to the universal model, better at estimating the CPU usage, CPU temperature, and the difference between inlet and outlet air temperature. This is as expected, as each server will have some variations that are included in its individual model. These variations can be caused by the positioning of the server in the rack. However, the universal model comes very close to the individual models in terms of $RMSE$. Both the individual and the universal models demonstrate a similar behavior: as the number of polynomial degrees increases beyond the 6th degree, the models tend to overfit. We remark that after the 3rd and 4th degree there is little to no improvement in terms of $RMSE$ and R^2. Therefore the 3rd and 4th degree strike a good balance between computational complexity, which grows exponentially as number of polynomial degrees increases, and the $RMSE$ and R^2 scores.

Without knowing anything about the internal state of a server, we prove that we can still estimate the CPU usage (CPU_{tot}) using external metrics. In our data set, the CPU usage ranges from 0% to 100%. Exclusively monitoring the power consumption (P_{watts}) of the server yields an error of 8.02% (individual) or 8.66% (universal) when estimating the CPU usage. When we estimate the CPU usage based on the temperature difference (T_{diff}) between inlet air and outlet air, we get less accurate results with an error of 12.35% (individual) or 13.51% (universal). When considering both power consumption and temperature difference the error is 7.35% (individual) or 8.14% (universal).

We also verify that we can estimate the CPU temperature (T_{cpu}) based on external parameters. The CPU temperatures range from $24°C$ to $100°C$ in our data set. When determining the CPU temperature using the server's power consumption we obtain an error of $4.65°C$ (individual) or $5.13°C$ (universal). Using the temperature difference between inlet and outlet air to estimate the CPU temperature yields a better result, with an error of $3.39°C$ (individual) or $4.13°C$ (universal). Estimating the CPU temperature using both power consumption and temperature difference results in an error of $3.17°C$ (individual) or $3.84°C$ (universal).

Additionally, we also demonstrate that we can estimate the difference between inlet and outlet temperatures without requiring access to the server. While this is not important for determining a server's internal state, it is useful from the data center perspective as it relates directly to the cooling load of a data center. The temperature difference between the inlet and outlet air of a server ranges from 0° to 25° within our data set. Given the power consumption of a server, our model yields an error of $1.07°C$ (individual) or $1.25°C$ (universal) when estimating the temperature difference.

Our results confirm that we can monitor the internal state of servers without requiring access to the operating system or server chassis. This enables privacy-preserved monitoring of co-location data centers as servers processing highly sensitive data can be considered black boxes, yet we're still able to estimate the internal state. We merely have to deploy inexpensive networked sensors to measure the temperature difference between inlet and outlet air. In most cases, the IT equipment's power consumption is already monitored in data centers.

4.3.4 Conclusions

We have provided a positive answer to the question we posed in the beginning of this section: can we employ an Internet of Things (IoT) approach to monitor the state of a server without requiring direct access to the operating system or server chassis, to enable privacy-preserved monitoring in co-location data centers? Our answer is based on 2.5 billion data points collected from 164 individual servers, covering 13 distinct metrics. We determined the correlation between each of the metrics, and identified external metrics that estimate well the internal state of servers. We outlined our approach to modeling the relationship between internal and external metrics, where we distinguished between individual models and universal models. In total, we trained and evaluated 10,430 individual models and 70 universal models. The results validate the vision that using external measurements to estimate the internal state of a server is feasible.

This enables privacy-preserved monitoring, as we demonstrate that we do not need to access the server's operating system or chassis to monitor the server's internal state. Instead, we can employ IoT to measure the inlet and outlet air temperature, and the power consumption of a server. Using these inexpensive sensors, the operators of data centers can monitor their infrastructures, ensuring the privacy from a data center point of view as direct access to the server is no longer required. It also enables co-location data centers to evaluate metrics that require information about server utilization, and assists DCOs in the expansion planning process.

4.4 Summary

We have set out to uncover the potential of data center monitoring using IoT. To achieve this, we first investigated the data generation capabilities of a smart data center. According to our scenario, a reasonably-sized data center can generate almost 170 Gbit/s of monitoring data. As this would have a significant impact on the data center's network load, we proposed an edge-based approach to data center monitoring, which uses an IoT hub at the edge to process data on a per-rack basis. There is a significant number of freely available open-source IoT hubs. To assist in the selection process, we have developed a framework for the evaluation of IoT hubs, and subjected 20 systems to this framework, of which four were studied in greater detail by subjecting each system to 13 features and 34 criteria analysis. The resulting analysis uncovered many of the strengths and weaknesses of each system, and of IoT hubs in general. We also discussed the common architecture that emerged from the analysis, and drew parallels with the proposed edge gateway architecture. And finally, we demonstrated the potential of IoT-driven data center monitoring by applying it to a real-world data center. In total, 2.5 billion data points were collected from over 160 servers, covering 13 different metrics. The data was collected with the help of a tailor-made IoT hub, which was influenced by the generic IoT hub architecture. Using the data collected from monitoring the servers, we were able to train models to estimate the internal state of the server using only externally measured data points. This enables co-location data centers to monitor servers that are not owned by them. We emphasize that IoT hubs do not only play an important role in smart data centers, but are of course important in any type of smart building. In the next chapter, we will shift our focus to smart buildings and the parallels between smart buildings and smart data centers will become clear.

Chapter 5

From Smart Data Center to Smart Building

There are two emerging paradigms that play an important role in transforming regular buildings into smart buildings: the smart power grid and the digital interface to everyday objects known as the Internet of Things. On the one hand, the smart grid aims to integrate the behaviors and actions of all connected actors, be it consumers, producers, or prosumers by leveraging information and communication technologies. Local energy generation and storage, economic efficiency, and sustainability are some of the key features associated with the concept of the smart grid [TMOSP19]. On the other hand, the IoT paradigm enables a wide variety of devices and machines to interact with one another programmatically and autonomously [LL15]. This allows for the establishment of smart buildings, where devices can be used to sense and control, and in turn influence the energy consumption of a building by reacting to different signals while satisfying the occupants needs and maintaining comfort levels.

The signals to which a smart building can react are multiple. For example, a smart building can react to the behavior of its occupants by learning usage patterns of rooms, and ensuring comfortable temperature levels before and while the room is expected to be in use. The benefits are multiple, high comfort levels are maintained as the room is already at a comfortable temperature before its expected use, and when the room is not in use no unnecessary cooling or heating takes place. Room occupancy and user behavior can also be used to control the lighting systems of a building. Signals that are external to the building also play an important role. The current weather conditions and the forecasted weather signals can be used to optimize the heating and cooling of the building. The smart grid paradigm promises to introduce another important external signal: the real time price of energy as provided by different producers, such as power plants, and by prosumers, such as neighboring buildings with renewable generation and energy storage. This signal is mostly influenced by weather, fossil fuel prices and current demand [PA15; WLJ12]. End users receive an incentive to actively participate by changing and rescheduling their load patterns based on current prices, while not being forced to [FA19; GMRH16]. This differs from demand response schemes where the users leaves the ultimate control of its consumption to their energy provider [Dar13].

The present chapter shifts our focus from smart data centers to smart buildings, between which many parallels exist. In Section 5.1, we investigate a practical example of transforming parts of a traditional building into a smart building, with the principal goal of reducing the energy consumption while maintaining user comfort. Two offices, a social corner, and a restaurant are included in the experiments. Each area is equipped with smart sensors and actuators, which enables our system to take control in order to optimize the environment. In Section 5.2 we research an approach that optimally schedules appliances and devices in order to minimize costs. A parallel uniform-cost search algorithm for finding the optimal solution is proposed, and a micro-service architecture is designed for integration with the building management system. The evaluation considers eight schedulable devices, as well as the inclusion of renewable energy sources and energy storage. Our findings are summarized in Section 5.3, where we also highlight similarities between smart buildings and smart data centers.

5.1 An Intelligent Building: the Bernoulliborg

An intelligent building is a building that provides a responsive, effective, and supportive environment for its occupants, supported by an automated building management system for controlling a wide variety of devices [Oma18]. Examples of devices that can be found in buildings include actuators, such as controllable ceiling lamps, movement detectors, and light sensors. The emergence of IoT-enabled devices has made it easy to remotely control and monitor different aspects of a building. Traditionally, the building management system is responsible for heating, ventilation and air condition control, lighting control, hot water control, and electricity control. As building management systems migrate towards service-oriented architectures and IoT devices are more prevalent, the system is able to trigger a single service or a predefined sequence of services [AD08; DLGL+13]. However, the challenge is to understand the context of the building and its occupants. For example, a traditional building management systems may use a motion sensor to turn on lights in a room when it is occupied. But the system does not have a deep understanding of the context and state of the room and its occupants. Activity recognition is an important step to understanding the context, a room in which there is a single occupant that is reading a book has different lighting requirements than the same room with multiple occupants performing work on personal computers. In short, the building is unable to anticipate the occupants behavior in order to provide intelligent responses that may even lead to energy savings.

To introduce intelligent responses, a search through the high number of possible contextual states most occur. Finding the compositions of services that will satisfy the requirements needed for performing the activities is a complex task, especially when additional constraints exist, such as reducing the energy consumption. The challenge is to address the building coordination problem by performing service composition automatically and dynamically, while identifying the occupant's activities as early as possible during the coordination. Doing this should maximize occupant comfort and minimize occupant disruption under the constraints of minimizing the energy consumption. Once a sequence of services that anticipates the activities of occupants is found, the next step is to execute the services.

The main problem we address is how to efficiently coordinate and execute services, and also anticipate occupant activities, while the environment changes due to external factors, or as services happen to fail at execution time. Hierarchical Task Network (HTN) planning and activity recognition play a central role in solving this problem. To understand which services to compose, activity recognition is needed to identify occupant activities as soon as they occur. The combination of planning and activity recognition outputs a sequence of services that needs to be executed under the uncertainty of a building environment. A critical component to the solution we propose is the underlying architecture. We resort to a service-oriented architecture, where composition, orchestration, and activity recognition components are all encapsulated in services. Furthermore, we standardize the communication between services, and enable the automatic discovery of services. We develop our own gateways for integrating and interacting with IoT devices through a RESTful API. To evaluate our approach, we deploy our solution in the Bernoulliborg building at the University of Groningen in the Netherlands. The Bernoulliborg is an office building of more than 10,000 m^2 erected in 2008. The building has 180 offices, 16 lecture rooms, 8 meeting rooms, 6 social corners, and a restaurant situated on the ground floor.

In this section, we focus on the design, implementation, and the evaluation of our work as published in [GNN+17], with an emphasis on the service-oriented IoT point of view. For details, specifically regarding HTN planning, domain modeling, and activity recognition algorithms, we refer to [GNN+17] and [Geo15]. What follows next is a description of the approach taken, as well as the architecture and implementation of the supporting framework to solve the aforementioned building coordination problem. This is followed by a description of the deployment and the practical evaluation using two different use cases in the Bernoulliborg building. The first case concerns the deployment of our solution in two offices and one social corner, and the second case concerns the deployment in the restaurant. Our results show both energy savings, as well as economic savings, while maintaining high levels of end-user satisfaction.

5.1.1 Approach

To enable intelligent responses from a building management system requires several limitation to be addressed. First, preprogrammable timers and set points simply cannot address the requirements of buildings, such as dynamism, service availability, uncertainty, scalability, and support for standards, protocols and policies. Second, the existing IoT hubs appear to be limited in terms of providing only basic device interoperation [KWLA13]. Third, rule-based approaches, which provide more complex control functionalities, are characterized by several drawbacks: 1) a lack of flexibility as all possible situations that may happen in a building environment need to be predicted and covered by a large number of carefully designed rules; 2) a lack of systematicity as the building environment becomes more robust, more rules are added without any enforced systematic steps resulting in a cluttered building management system; 3) a limited service order as the creation of ordered or partially ordered adaptations appears to be a challenge (for example, given an objective to close a window, a rule-based approach might create a coordination consisting of, first, pulling down the blinds, and then closing the window); 4) a lack of re-usability as reuse, modifications and maintenance across different types of building environments becomes difficult. Two spaces may happen to have some commonalities, but if those spaces serve different purposes, say a meeting room and a social corner, then most of their situations will differ.

Our approach to the building coordination problem is based on an application of AI planning to compose services. AI planning provides powerful means for searching and composing 'best' sequences of actions in many domains [GNT04]. Approaches based on AI planning are able to address the challenges of building environments and to overcome the drawbacks of rule-based approaches. In fact, AI planning naturally supports the evolution of environments in pervasive computing – it is easily adaptable and flexible approach [KWLA13]. The domain model is maintained and modified in an organized and conceptual way – the purpose of each action is always known. Furthermore, situations that require execution order among some or all actions in some coordination can be easily achieved. Finally, the same domain model can be suitable with minor modifications to a wide range of buildings.

Since we expect that the service coordination will be initiated as early as occupant activities occur, the goal should correspond to the occupant activities in terms of requirements needed to be satisfied for the comfortable occupant performance. This means that to solve a planning problem, we need to identify the current occupant activities. Identifying activities relates to the problem of transforming low-level sensor data into high-level and meaningful information over which AI planning is well suited to reason. Fortunately, we can turn to human activity recognition to solve this problem. Occupant activities in buildings are generally well known and often limited in number, so we can use a predefined set of activities to recognize the occurring ones. In particular, we enforce ontology-based reasoning on sensor data to preprocess and classify the occupant activities [NRA14]. Once a set of activities is identified, we may establish a correspondence with the planning goal. A direct match can be established if each activity corresponds to some procedure without specifying details on what has to hold in final state for each activity. For support of procedural goals in AI planning, we can turn to HTN planning [EHN94].

We solve the problem of execution of services by using the concept of orchestration from service-oriented computing, which is the process of coordinating and executing services with the purpose to carry out the computed plans [Erl07]. We propose an advanced orchestration model and a strategy to coordinate the receipt of events, the planing for new HTN planning problems, and the execution of their corresponding plans based on an effective and modern approach for plan orchestration in dynamic and uncertain environments [KLA16]. With our solution, building management systems can anticipate occupant behavior and provide dynamic and intelligent responses to different building situations. Recalling the example of lights in office buildings, a management system based on our solution will understand that Theodore is working with a computer and that the level of natural light for this activity is sufficient according to the building standard. It will thus not turn on the lights. Moreover, different compositions of services can be computed for the same objective depending on the current contextual state. Since the coordination of services is performed at runtime, building management systems can consider availability of services dynamically. Finally, our solution supports heterogeneous devices and is easily customizable to different building management systems.

5.1.2 Architecture and Implementation

The realization of our proposed approach requires a platform to be in place that is able to interface with IoT sensors and actuators, similar to the IoT hubs discussed in Chapter 4. However, contemporary IoT hubs lack the capabilities that enable a building to exhibit intelligent responses in a changing building environment as they are limited by their rule-based approach. To address this limitation, we propose our own architecture which has additional services that enable activity recognition and service orchestration. With the orchestration, we can complete a whole operation cycle of buildings' operation, starting from the collection of data through sensors; processing it into context information; composing acting services to coordinate the building environment by anticipating the occupants activities; until the execution of acting services upon devices representing actuators. In Figure 5.1 we show the architectural design of our proposed solution to the building coordination problem. The arrows in the figure indicate the main data flow. Communication is done by passing JSON objects between components, either via the message queue, or via REST services. All services are automatically registered with ZooKeeper, enabling service discovery. What follows next is a description of the main components, including implementation details. It

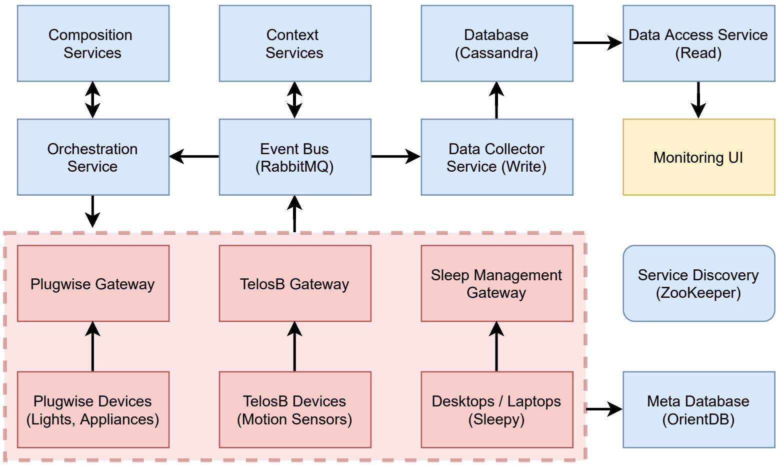


Figure 5.1: High-level overview of the architecture design.

will become clear that there are many similarities between our proposed architecture and the generic architecture for IoT hubs as proposed in the previous chapter.

Gateway Services and Devices Understanding the activities in the building and providing an appropriate response requires our system to interface with a wide variety of sensors and actuators. The gateway services that are part of our architecture are responsible for the communication between the devices and the rest of our system. Each communication protocol has its own gateway. The Plugwise devices communicate with the Plugwise gateway, and TelosB devices communicate with the TelosB gateway. When a protocol allows for any sort of control, the gateway exposes this control in the form of a RESTful API. In practical terms, this allows the Orchestration service to send REST requests to the Plugwise gateway to turn the devices on and off. Each gateway is also responsible for registering meta data about the devices in the Meta Database.

We utilize wireless TelosB-based sensors produced by Advantic Systems. These sensors are integrated with an on-board Passive Infrared (PIR) based

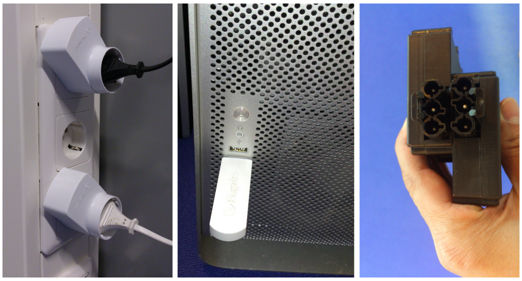


Figure 5.2: Power meters and a receiver device.

movement detectors, microphones, FlexiForce pressure sensors, and light sensors. The devices are programmed in nesC and run on the TinyOS 2.1.1 embedded operating systems. As power meters, we use Plugwise products shown in Figure 5.2, consisting of plug-in adapters that fit between a device and the power socket. The adapters can measure the real-time power consumption of the device plugged in, and can turn on and off the device on request. The adapters form a wireless ZigBee mesh network around a coordinator. The network communicates with the base station through a link provided by a USB-based receiver device. To monitor activities on workstations, we use the Sleep Management gateway, and the Sleepy software that is deployed on the workstations. For more details on Sleepy, we refer to our work in [SNLA16]. The Gateway services are implemented in the Scala programming language (www.scala-lang.org), and run on an Intel Next Unit of Computing (NUC) thin client device, which is deployed in the restaurant itself.

Context Services The Context services consists of a Context Processing (CP) service and an Activity Recognition (AR) service. These services provide the system with the ability to process the sensor data received from the gateways. This is achieved by listening to events on the event queue, and processing them as they are received. CP is responsible for ensuring a consistent view over the ambient environmental condition, such as the

Table 5.1: Description of activities considered for the evaluation of our system

Activities	**Definition**	**Artifacts**
Working with PC	Person uses PC, PC and screen are turned on, keyboard/mouse are manipulated	Chair, PC, screen, keyboard/mouse
Working without PC	Person sits at a desk, PC and screen are turned off	Chair
Having meeting	Two or more persons discuss	Chair, microphone, PC
Presence	Person is active in an area but no further specific activity is recognized	Motion sensor
Absence	User is absent from an area	No artifacts involved
Having coffee	Persons have lunch/coffee or just a break	Microwave, movement detector

natural light level. That is, raw sensor readings, coming from a light sensor in the unit of lux, are calibrated and correlated in a standard form for the light condition at different locations. To calibrate and correlate the readings, we collect historical data from the sensors associated with the ground truth about the natural light levels. Then, the CP uses this information to classify the incoming raw sensor readings into appropriate natural light conditions. The AR service is responsible for processing the data related to occupant activities. For example, the movement data coming from PIR sensors is used to reason over for the presence activity. The occupant presence of an area can be determined from one or many PIR sensors, depending on the pre-configuration of the environment. Table 5.1 shows the definitions and involved artifacts of the activities considered for the evaluation of our system.

We also implement a mechanism that adapts the timeout of the presence activity depending on the time of the day. For example, in the Bernoulliborg restaurant, in the early morning, when the chance of occupants staying long in some area is low, the timeout is set as low as 5 minutes, while during the lunch time, when occupants would stay long for their lunch, the timeout is set to 30 minutes. The time values are adjusted based on observations.

Meta and Time Series Databases Two different database are used to store two types of building information. One involves descriptions of devices deployed in the building, their device types and locations within the environment, their data type and value ranges, the layout of the building, and any other meta information. We use Neo4j to store the environment's meta data. Neo4j is an open-source graph database with features from both document and graph database systems and excels in scalability, availability, performance, and price. Our Neo4j database contains static information about the building. The other type of information is dynamic and involves the data coming periodically from devices directly or from the Context services. This concerns mainly time series data. Therefore, we use Cassandra to store time series data coming from sensor readings. Cassandra is a NoSQL database that delivers fast performance for systems involving time series or big data.

Composition Services The Composition services are composed of three individual services: 1) a modeling service, which provide capabilities that focus on the planning entities used to form an HTN planning problem; 2) a problem-solving service, which have capabilities bounded within the process of solving planning problems; and 3) an utility service, which include capabilities related to message conversion, storage of domain models, communication with other components, and exception handling. We encode the domain model using Hierarchical Planning Definition Language (HPDL) [CFOGPP06]. We assume that a domain model specified in HPDL is predefined. The HPDL problem and domain descriptions are then transformed into programming-level constructs through the HPDL processor. This processor uses parser combinators to transform HPDL descriptions into Scala objects. The HPDL domain and problem specifications constitute a HTN planning problem which is provided as input to the Composition component.

We use our own HTN planner, called Scalable Hierarchical (SH) planning system. We provide planning as a service by implementing SH's capabilities as REST resources (Web services) [RR07]. Upon receiving a request with appropriate arguments, SH may check, for example, the correctness of an HPDL domain or problem, the consistency of a required problem and domain, or search for a solution. The planner replies to an interested component with an answer appropriate to the situation. For example, SH may provide a plan in JSON format, or it may show a syntax failure at a specific position in the domain encoding.

Orchestration Service Since building environments are expected to support continuous orchestration, we need to adopt a strategy that enables long-running runtime activities of the Orchestration component. The Orchestration service receives events from the Context services continuously and reacts immediately and accordingly. When necessary, the Orchestration uses the planning services to create an HTN planning problem and to compute a plan. If plan is found, the Orchestration executes it by executing REST calls on the Gateway services.

Since the orchestration service is stateful, it maintains a model of the environment, including the planning state, domain, task network and orchestration plan; and, in order to support our design assumption regarding concurrent use and updates of the state. The orchestrator receives messages asynchronously and reacts to them by making local decisions, creating other actors to handle specific and/or concurrent messages, sending new messages, and deciding what to do upon the next message received. The model of the environment is represented by an implementation of the restaurant. The orchestrator populates a specific environment by retrieving the information from the databases. The set of variables, their types, locations, and properties are gathered from Neo4j. The initial values of variables are gathered from Cassandra. Then, the orchestrator subscribes to RabbitMQ, and awaits for messages, that is, events. Upon each event, the orchestrator creates a corresponding HTN planning problem and invokes the core planning service

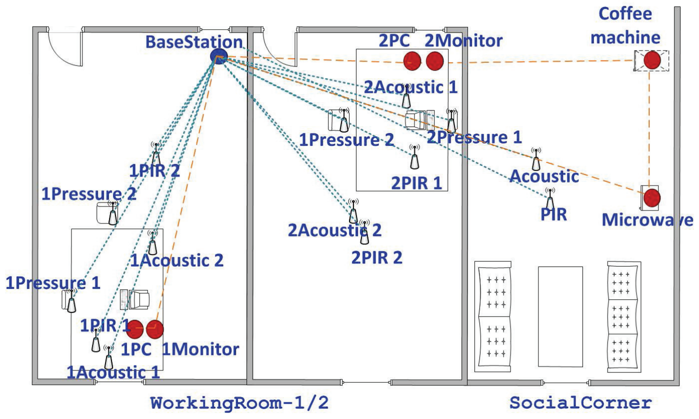


Figure 5.3: Schematic representation of the offices, social corner, and deployed devices.

of SH. When a plan is found, the orchestrator translates the plan steps into actuating services and uses the gateway services implemented as REST resources for execution.

5.1.3 Evaluation: Offices and Social Corner

First we evaluated the deployment in the two offices and the social corner. This area covers a total of 60 m^2. Each space has several small and large windows which let a large amount of natural light into the spaces. The offices are typically used by two occupants, while the social corner is continuously used by many occupants from the surrounding rooms. The layout of the offices and social corner together with the network of power meters attached to appliances and simple sensors is illustrated in Figure 5.3. In particular, each space has three controllable light fixtures (or lamps). The lamps consume 32 W each. Each lamp is equipped with a Plugwise plugs, which make the lamps controllable but also provides information about the power consumption of the fixtures.

We consider simulated control of the workstations of the PhD candidates in the two offices. A workstation consists of a Personal Computer (PC) with an 24” LCD screen. The power consumption of a workstation when actively in use is 120 W, and only 5W while in sleep mode. We attach Plugwise power meters to each PC and screen to detect the power state and measure the power consumption. In addition, we take into account the use of a microwave at the social corner. The microwave is also associated with a Plugwise power meter to measure its real-time power consumption. We installed 14 more sensors to estimate chair’s occupancy, to measure the human voice level, and to detect people’s movement. Finally, we simulate a natural light sensor by using weather conditions retrieved from Freemeteo[1]. We assume that clear weather equates to light levels of 700 lux, a few clouds to 550 lux, partly cloudy skies to 500 lux, cloudy skies to 425 lux, light snow to 400 lux, and mist indicates 350 lux.

We conducted tests on the system during the working hours of three days. We analyze two scenarios to show the potential for energy saving. In the first scenario, we consider manual control, meaning that occupants have manual control over their workstations and lamps. In the second scenario, we consider automated control where the planner takes into account both the natural light level and recognized occupant activities to control the workstations and lamps automatically. Here we do not actually control the workstations, but we assume the use of the Sleepy software that enables bringing workstations into sleep mode or waking them up on planner’s demand, as is presented in our work in [SNLA16].

Energy savings For the manual control scenario, we consider that the two workstations and lamps at all three areas are turned on during all of the working hours, given the fact that it is common for office workers to leave their computers on even when they are not using them and no power management options are enabled [All07; All09; CM10; WRMB+06]. The average energy consumption is then 0.528 kWh. For the automated

[1]Freemeteo--TheWeather,https://freemeteo.com

control, the workstations are in active mode only if the activity "working with PC" is taking place, otherwise they are brought to sleep mode. Lamps are controlled in accordance with the European standard on lighting in indoor work places. In other words, for the activities *working with PC, having a meeting*, and *being present* require lower levels of light at 500 lux, while the activity *working without PC* requires a higher light level of 750 lux. In the automated control mode, our system provides an intelligent coordination of the use of workstations and lamps with respect to the activities of occupants and the natural light level.

Figure 5.4 illustrates the power consumption of both the manual control and our system at every minute on two different days; day one and day two. The results of day three are very similar to day two. The red dotted line indicates the base-line power consumption associated with manual control, whereas the green line indicated the consumption using automated control. A significant reduction of energy consumption is evident in the case of automated control by taking into consideration the natural light level and turning off as many lamps as possible, and bringing workstations into sleep mode as soon as occupants do not work with their PC. In particular, no lamps were needed on the first day , while on the second day only five lamps were turned on for 11 minutes and 24 minutes, respectively. Similarly, two workstations were needed in total for 5.13 hours on the first day, and 8.47 hours on the second day.

Table 5.2 shows the potential for energy savings of our system. When compared to the manual control, the combination of activity recognition with planning brings a significant potential for energy reduction of 75.5%, including 46.9% and 98.5% savings from controlling the workstations and lamps, respectively.

Performance We evaluate the accuracy of the activity recognition that our system performs. The system is considered accurate if it is able to recognize and classify occupant activities correctly. During the tests on the three separate days, the system recognized six different activities performed by

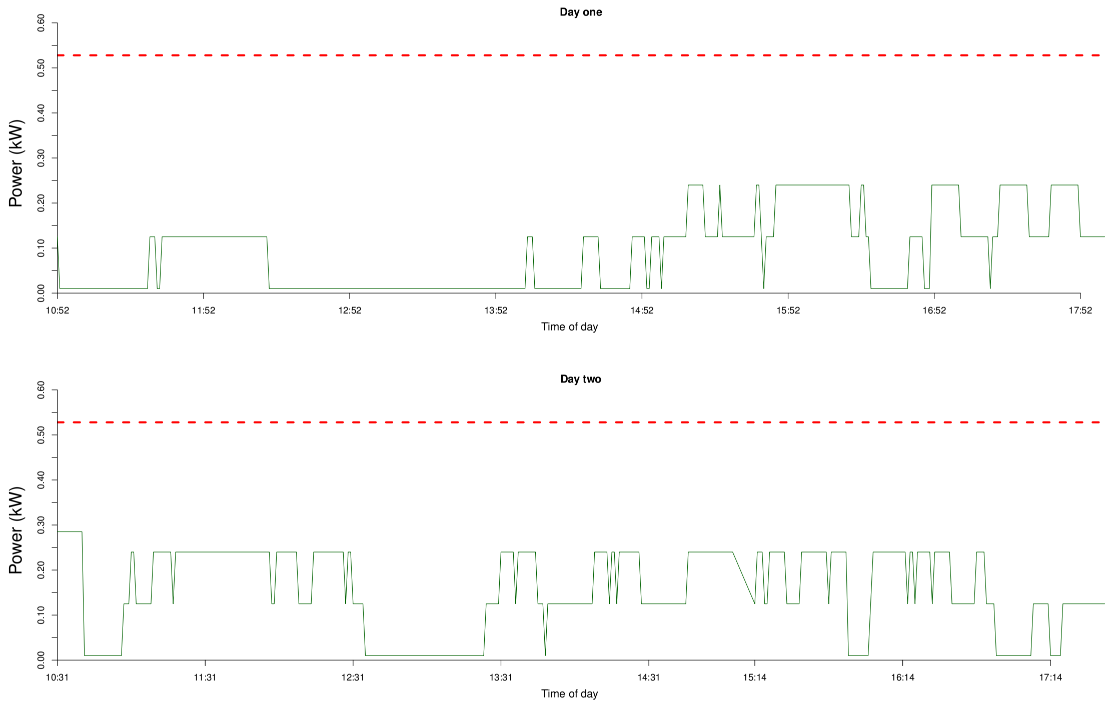


Figure 5.4: Comparison of power consumption between manual control and automated control.

Table 5.2: Potential for energy savings in the offices and social corner.

Date	**Device**	**Total time (hours)**		**Consumption (kWh)**	
		Manual control	Automated control	Manual control	Automated control
Day one	Lamps	64.98	0	2.08	0
	Workstation active	14.44	5.13	1.73	0.62
	Workstation sleep	0	9.23	0	0.05
Day two	Lamps	64.53	0.92	2.06	0.03
	Workstation active	14.34	8.45	1.72	1.01
	Workstation sleep	0	5.5	0	0.03
Day three	Lamps	63	2	2.02	0.06
	Workstation active	14	8.27	1.68	0.99
	Workstation sleep	0	5.82	0	0.03
Total energy consumption of lamps				**6.16**	**0.09**
Total energy consumption of workstations				**5.13**	**2.72**
Total energy consumption				**11.29**	**2.82**
Energy saved from lamps				**6.07 kWh (98.5%)**	
Energy saved from workstations				**2.41 kWh (46.9%)**	
Total energy saved				**8.48 kWh (75.05%)**	

Table 5.3: Ground truth of the occurrences of activity instances at each area.

Activity	**Area**		
	WorkingRoom-1	*WorkingRoom-2*	*SocialCorner*
Absence	70	129	843
Presence	232	112	0
Working without PC	256	139	0
Working with PC	640	754	358
Having meeting	3	67	0
Having coffee	0	0	0

the two occupants in the two offices and using the social corner. Using pen and paper, the occupants took accurate notes of the actual activities happening every minute, which are then used as the ground truth for the evaluation. Table 5.3 shows the real occurrences of activity instances at each area. We collected ground truth composed of 1201 activity instances with a granularity of one minute. The overall success rates of the activity recognition service for *WorkingRoom-1*, *WorkingRoom-2*, and *SocialCorner* are 80.85%, 76.35% and 99.17%, respectively. On the other hand, the total number of minutes of incorrect recognition for *WorkingRoom-1*, *WorkingRoom-2*, and *SocialCorner* are 230, 284 and 56, respectively. A detailed analysis of these results is presented in [NRA14].

5.1.4 Evaluation: Restaurant

The restaurant of the Bernoulliborg covers a total area of 252 m^2 with a capacity of 200 sitting places. Two photos of the restaurant are presented in Figure 5.5. The restaurant has glass walls on three sides, allowing for a significant amount of natural light to enter when the weather conditions allow for it. The restaurant area is used for lunch from 11:30 a.m. until 2:00 p.m. Outside of these hours, the area is used by staff, students or other visitors for work, meetings, or other purposes.

The restaurant area is an open space divided in two sections by design, in between the sections is a stairway. We make use of this division in our set-up.

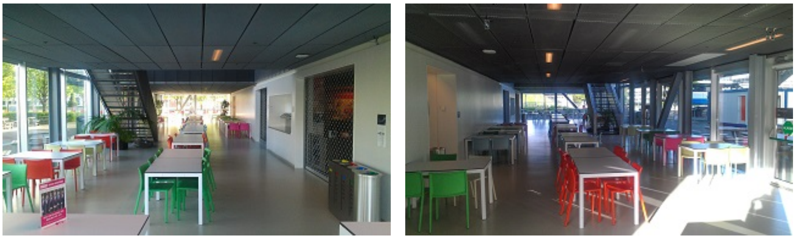


Figure 5.5: Overview of the restaurant from the east and west sides.

The layout, together with the locations of deployed sensors and power meters, is illustrated in Figure 5.6. In particular, each section has 15 controllable light fixtures (or lamps), making 30 in total. All lamps are fluorescent Philips tubes. There are several light fixtures that are uncontrollable and represent security lamps. While we do not control these, we take into account the light that they provide. In addition, there are two types of controllable lamps. The first type are large lamps that have a power consumption of 38W each, and the second one are small lamps, each of which has a power consumption of 18W. These lamps are controllable via the Plugwise plugs, which also provide information about the fixture's power consumption. We installed 15 additional sensors, one to measure the natural light level, and the rest to detect people's movement. We divide each section of the restaurant into smaller spaces, called areas. The areas are not necessarily of the same size, and we have positioned the embedded movement sensors in each area such that they cover most of the space of the respective area.

We conducted tests on the system over the course of five weeks in the months of February, March, and April, involving measurements from Monday to Sunday. These months are cold and dark months, meaning that we experience some of the worst possible conditions to save energy. In the last three weeks in February, we recorded measures of energy consumption of lamps in order to understand the typical behavior of manual control of lamps in the restaurant under manual control. This enables us to define a baseline. In the last week of March and first week of April, we allowed for automated control of the environment by using our system. Thus, manual control was

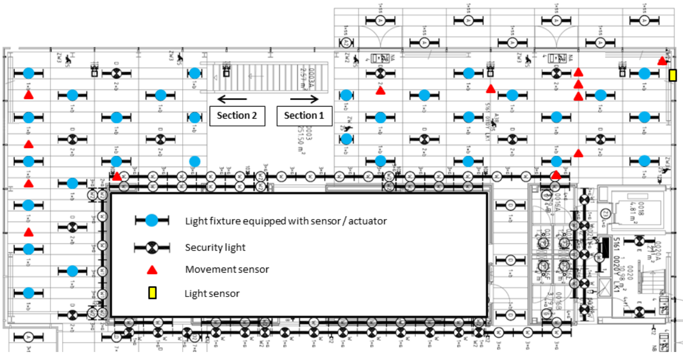


Figure 5.6: Schematic representation of the restaurant and deployed devices. The legend only describes the symbols of interest to the experiment.

disabled and the system was running continuously without interruptions during these two weeks. For the purpose of comparison, we simulate control of lamps based only on the information coming from the movement sensors during the period of automated control. We use the same setup of sensors and lamps as in the case of automated control. We consider two types of activities: presence and absence. To summarize, we consider three situations in our evaluation: manual control, motion sensor control, and automated control.

Energy savings Observing the measurements gathered in February, when there is manual control in the restaurant, we found that the average time point when the lamps are turned on by the building cleaners is 6:30 a.m; the lamps stay turned on until around 8 p.m., when they are usually switched off by the security personnel. The average consumption per working day in the restaurant is 14 kWh. On weekends, there is no manual control of the lamps, thus they are always off. The use of our system results in intelligent coordination of the restaurant with respect to the natural light and presence of people. This means that there are a plenty of possibilities for lamps to be turned off, which enables energy savings. Figure 5.7 shows the daily average

Table 5.4: Daily total energy consumption (kWh) and savings with respect to the use of motion sensors during two weeks of using our system.

Day	Motion sensors	Our system	Savings
Week one			
Monday	3.63	0.37	89.6%
Tuesday	3.94	0.79	79.8%
Wednesday	3.97	3.97	0%
Thursday	3.56	1.34	62.2%
Friday	5.19	4.10	21.1%
Week two			
Monday	3.16	0.73	76.8%
Tuesday	3.83	1.21	68.3%
Wednesday	4.27	0.63	85.3%
Thursday	3.98	0.60	84.8%

electricity consumption when the lamps are controlled manually, when the lamps are triggered by motion sensors, and when our automated control system is used in the restaurant. Table 5.4 shows the percentage of savings for each day when compared to using only motion sensors. Wednesday and Friday of the first week have lower savings compared to other days, which is due to the worse weather conditions in these two days. In summary, the average savings of electricity between the scenario of manual control and motion sensors is 71%, the average savings between the scenario of manual control and the one with our system is 89%, and the average savings between the scenario of motion sensors and our system is 61%.

Economic savings The average daily cost when our automated system is used is €0.34, assuming €0.22 as the kWh tariff. Even in the worst case during working hours while weather conditions are worse than average and the restaurant is visited more than usual, the cost resulting from the use of our system is always less than the cost associated with manual control. On average, seven months of automated control costs as much as one month of

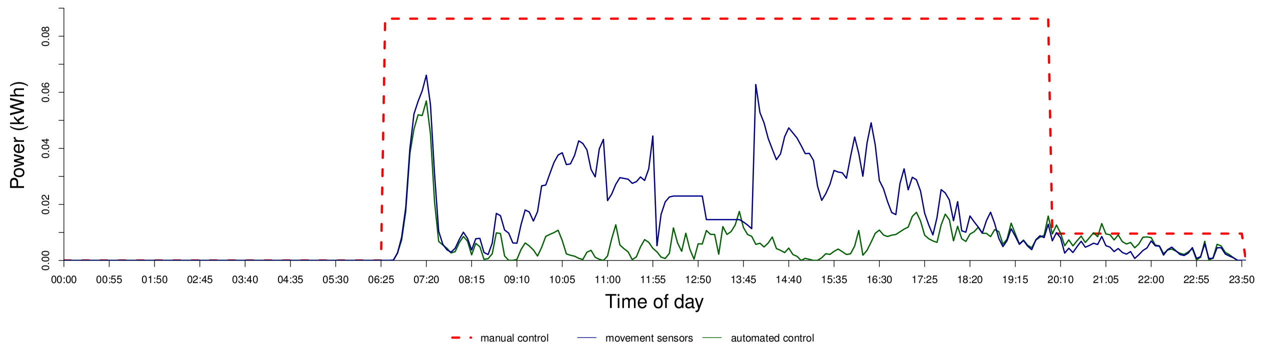


Figure 5.7: Comparison of daily average electricity consumption between using manual control, motion sensors and our system to control lamps.

manual control. In other words, while manual control costs € 746 annually, we only pay € 88 annually when our system is used in the restaurant.

Usability To evaluate the usability, we prepared a questionnaire following the guidelines in [GA16]. In particular, we identify two groups of users, one experiencing the system during lunch, and another one outside lunchtime. The attitude of occupants towards our system, such as the use of sensors, switching lamps (more often than usual), and automation of tasks, defines their acceptability. The need for occupants to understand how to use our system defines learnability. The satisfaction of users with the overall system is related to system effectiveness, and user satisfaction with the time the system takes to perform its tasks relates to system efficiency. Occupants may have different requirements for the system. For example, occupants having lunch may not have the same expectations for the level of light as compared to the ones reading or working in the restaurant outside lunchtime. Finally, we use two Likert scales each with five levels to have an informed way of defining the actual usability of the system [Lik32]. Our questionnaire has 24 questions, two with multiple choices and the rest with the items on the Likert scales.

We conducted the survey on the group of occupants experiencing the system during lunch (L), and the one outside lunchtime when working/reading (W). The group L shows partial awareness with one third of the participants stating that they are aware of the system. The perception of the group L is that our system saves energy (65%), and considers the natural light level (54%) and people's presence (72%). Regarding the learnability, 43% of the group state that it is easy to use the system, while more than half are neutral or do not know the answer to this question. The majority of the participants find the system to be acceptable, while 17% think that the system causes distractions. As for the system efficiency, the majority of the participants do not know if the system reacts immediately to changes or their answer is neutral. When it comes to the effectiveness, 59% of the participants state that they are satisfied with the system, 31% are neutral and only 6% are dissatisfied. Finally, the majority of participants state that they find the system to be useful.

The group W also shows moderate awareness with a bit more than one third of the participants stating that they are aware of the system. The perception of the group W is that this system saves energy (78%), and considers people's presence (67%). Half of the participants think that the system also considers the natural light level. Regarding the learnability, 67% of the group state that it is easy to use the system, while 33% are neutral or do not know the answer to this question. 83% of the participants find system to be acceptable, while none of the participants think that the system causes distractions. As for the efficiency of the system, the majority of the participants do not know if the system reacts immediately to changes or their answer is neutral (72%). When it comes to the effectiveness, 72% of the participants state that they are satisfied or very satisfied with the system, 11% are neutral and none of the participants are dissatisfied. Finally, the majority of the participants (83%) state that they find the system to be useful.

5.1.5 Discussion

We discuss first the case of the deployment at the two offices and coffee corner. Using real-world data, the outcome of the activity recognition process is gathered. The control of the lights is simulated in our experiments, there is no actual execution of the actions defined in the plan. It should also be noted that the weather conditions were optimal with regards to natural light levels. Therefore, the high percentage of energy savings is related to the fact that the lamps could be turned off most of the time. The savings achieved by controlling workstations and displays are realized by monitoring the activities of the occupants. It is a known issue that energy waste is caused by leaving computers on. According to [All07], 31% of office workers assume that their computer has power management options enabled, but, in reality, it is often the case that such options are not active. In fact, the sleep mode is disabled by default at the Bernoulliborg. While our assumption to have workstations and lamps turned on continuously during working hours may seem hypothetical at first, it does reflects the real-world conditions.

Since the main deployment and use of our system was performed in the restaurant, we provide a more extensive discussion for this case. During the first weeks, there were numerous challenges in the deployment at the restaurant. First of all, the TelosB sensor network had to be reset regularly, and the Plugwise plugs had some interference with the metal roof tiling of the restaurant, causing messages to be dropped in transit. While these reliability issues should be addressed in the long term, the reported savings are still valid. If these issues were not present, slightly higher savings may have been observed. For the detection of activities, a timeout was enforced to prevent triggering new activities too frequently. Concretely, during peak hours at the restaurant the timeout was increased to prevent occupants from having to manually trigger motion sensors in case they are sitting at the table and little to no movement is detected. Ideally, the timeouts should be adjusted dynamically, as was also done for the Sleepy software [SNLA16]. In the case of the restaurant, we settled for three different timeout values depending on the time of day.

We do consider the power consumption of the additional IoT devices and hardware required to make the system function: 30 Plugwise devices, 14 sensors, one thin client, and one server that consume 3.3 W, 6.7 W, 4 W, and 365 W, respectively. The consumption from the sensors and thin client is negligible compared to the savings. The consumption of the server is more difficult to quantify, as the server does not only perform calculations related to the restaurant deployment, but is also used for other research tasks. The costs associated with procuring the devices and hardware are however not negligible. If we consider the equipment that needs to be bought and deployed, the costs for 30 Plugwise devices, 14 sensors, one thin client, and one server are €990, €2040, €200, and €2700, respectively. Then, the payback period for the investment in our system would be nine years. But once again, this concerns a server that is not exclusively used for the services running in the restaurant.

In addition to the performance evaluation, we conducted a usability study.

The study provided insights into the attitude and satisfaction of the restaurant occupants with the usefulness and effectiveness of our system. Beside the realistic nature of the setting, the participants are not a random sample of the building population. The participants have specific social and academic characteristics that are correlated to the university environment. Particularly, the participants are well informed about sustainability issues and behavior. In addition, the participants are familiar with automated control and sensors in buildings. These factors prevent us from generalizing to all building population. Moreover, even though the participants were exposed to a mandatory use of our system, from the results it can be argued that these participants are highly motivated to engage in environmentally friendly behavior and will therefore be more inclined to accept our system than a general building population. We thus have a second reason to be careful about generalizing to the building population.

And finally, the system and underlying architecture was designed from the ground up to be extensible and adaptable to other buildings, regardless of the type of building. Communication between services is standardized, and the gateway approach to device integration enables any device with an open protocol to be integrated into the system. Furthermore, thanks to the standardization of interfaces and a service-based approach, it is possible to change the implementation of core services. In fact, we have done so in Chapter 4, where we replaced Neo4J with OrientDB, RabbitMQ with Kafka, and ZooKeeper with Consul.

5.1.6 Conclusions

We address the building coordination problem automatically by combining AI planning, activity recognition, and the IoT paradigm. We presented our architecture to support intelligent decision making. It enables the execution of computed plans to address the building coordination problem. Our proposed architecture is able to fully support the capabilities of building management systems: from sensing and recognizing occupant activities, planning for composing services, to executing services upon devices. This is achieved by standardizing communication between the different services, and by following a gateway approach to integrate and interact with external systems and hardware.

The developed system is deployed in the offices, the social corner and the restaurant of the Bernoulliborg building. We showed that the combination of HTN planning, activity recognition, and IoT can bring savings that go beyond those associated with traditional rule-based lighting control provided by existing automation solutions. The return on investment for the adoption of our solution in the restaurant is nine years. By design, the system has the capabilities to support different occupant activities and to be applied in diverse types of rooms. The system can also be integrated with any device or building management system that has an open communication protocol. While we could not generalize the results of our usability study to the whole building population, we can conclude that environmentally motivated occupants are satisfied with the system and ready to accept intelligent systems as ours.

The results show energy savings of up to 80% and economic savings of up to € 658 per year, while at the same time 83% of restaurant occupants find the system useful and up to 72% are satisfied with it. As government regulations regarding climate and sustainability become increasingly more restrictive, achieving even higher energy savings is a worthwhile objective to pursue. What we did not consider in this section is the inclusion of local renewable energy generation, energy storage systems, and varying energy price signals as the future smart grid promises to provide. The inclusion of these aspects has the potential to further increase savings. Therefore, in the next section we will investigate the scheduling of device loads for the purpose of saving energy while considering energy generation, storage, and price signals.

5.2 Optimizing Energy Costs for Smart Grid Connected Offices

The dynamic electricity price signal represents the electricity price as it changes over time depending on the demand and supply of energy, as well as other market factors. As a reaction to the signal, a building can operate subsystems by switching them on or off, and postpone their operation to times where the signal is expected to be more favorable. However, deciding when to schedule a subsystem load in order to minimize costs is a non trivial task, especially when local energy generation and storage add uncertainty to the overall set of possible decisions to make. For example, the heating of a meeting room could be anticipated if energy prices are expected to increase considerably just before the scheduled time of a meeting.

In 2012, Georgievski et al. proposed an architecture and optimization strategy to control office appliances based on user needs and dynamic prices [GDP+12]. In the last ten years, we have witnessed a widespread adoption of IoT hubs to monitor and control appliances and building systems, IoT and data science approaches to building context and human activity recognition, and rapid developments in electricity storage technologies. This provides both challenges and opportunities for building energy optimization.

The challenges are that more uncertainty is present in the system and that the complexity of monitoring and decision making increases. The opportunities come from the major flexibility and controllability of the system, enabled through IoT hubs. Therefore, we approach the problem considering the current state of the art and resort to improved techniques for the optimal scheduling of appliance and device loads in office buildings. Specifically, we present an approach to find the optimal load schedule that minimizes costs while considering (1) price signals from the grid and prosumers, (2) local energy generation, (3) local energy storage, and (4) the scheduling constraints of typical office devices. Based on this information, an optimization problem is formulated. The resulting search space is large, as for each time slot there are many permutations of possible actions that can be taken. The contributions made and presented in this section include:

- A general optimal device scheduling algorithm for smart grid enabled buildings that include photovoltaic panels, micro wind turbines, and battery energy storage;
- A parallel uniform-cost search to find the optimal schedule that minimizes overall costs while ensuring performance when navigating complex search spaces;
- A micro-service architecture for the integration of building management systems with independent cloud services, enabling a loosely coupled, modular systems that can be deployed in virtually any building; and
- A real-world data evaluation for the scheduled devices and for the prediction of renewable energy generation.

5.2.1 Approach

First we define the energy optimization problem in terms of the relationships between devices, their desired behavior, and the available energy sources. This is followed by the models of the energy sources, in the present case: wind turbines, photovoltaic panels, and prosumers on the same local grid. Energy storage in the form of batteries is also modeled as part of the optimization problem. The last part of the problem definition relates to the user policies associated to the office devices. Such policies drive the way in which each one can be scheduled. Finally, we describe the algorithm to find the optimal schedule.

Problem Definition Given the devices to be scheduled, the scheduling policy for every device, and the available energy sources, the goal is to find an optimal day-ahead schedule X that minimizes the costs while ensuring all of the constraints as defined by the device policies are satisfied. The following problem definition is based and extends the definitions from the work of Georgievski et al. [GDP+12].

For every time slot t with length $t.duration$, $ES(t) = \{es_i\}$ represents the available energy sources at time slot t. An energy source consists of a tuple $es_i = \langle cost, energy \rangle$, where cost is the cost per kWh, and energy is the available energy that the energy source can provide at time t.

The set of all building devices that are amenable to scheduling is D. For each device $d_i \in D$ there exists a tuple $d_i = \langle d_{id}, S_i \rangle$ where d_{id} is the identifier of the device, and S_i is the set of available states in which the device can be. Every state $s_{ij} \in S$ consists of the tuple $\langle s_{id}, power \rangle$ where s_{id} is the identifier of the state, and $power$ is the power consumption in that particular state.

Each device d_i has associated one or more scheduling policies that determine the scheduling behavior for the device. The set P contains all of the policies that are associated with devices. Each policy $p_i \in P$ is represented by the triple $p_i = \langle p_{id}, type, parameters \rangle$ where p_{id} is the identifier of the policy, $type$ is the policy type, and $parameters$ are the specific parameters for that type of policy.

A schedule $X = \{x_{ti}\}$ is a set of values per device per time slot, where each value $x_{ti} \in S_i$ denotes the state that a device i assumes at time t. Then we define the office optimization problem as follows: *find a schedule* $X = \{x_{ti}\}, \forall t \in T, \forall i \in D$ *that is optimal*. A schedule is optimal if and only if:

$$\sum_{t \in T} cost(t, e_t) \rightarrow min, \forall p \in P : satisfied(p, X) \tag{5.1}$$

where

$$e_t = \sum_{d_i \in D} x_{ti}.power * t.duration \tag{5.2}$$

In other words, the schedule is optimal when the devices are scheduled in such a way that the policy constraints for each device are satisfied and the total costs of operation are minimal.

Energy Sources: Wind Turbines One of the energy sources that is considered is local on-site energy generation using wind turbines. The following mathematical model is used to represent the wind turbine's power output and is based on the work of Xia *et al.* [XAW13]:

$$P_w = \frac{1}{2}\rho C_p A v_w^3 \tag{5.3}$$

where:

P_w	power output (W);
ρ	air density (kg/m^3);
C_p	power coefficient;
A	wind turbine swept area (m^2);
v_w	wind speed (m/s).

The wind turbine swept area and the power coefficient are dependent on the particular wind turbine model. As per the Betz Limit, the theoretical maximum power coefficient is 0.59, though in practice the power coefficient is generally in the 0.35-0.45 range [The10]. To determine the air density, first the saturation vapor pressure must be calculated. We resort to the work of Herman Wobus who has provided an approximating polynomial to calculate the vapor pressure [GW03]:

$$\begin{aligned} &p_s = e_{so}/C^8 \\ &C = c_0 + T_d(c_1 + T_d(c_2 + T_d(c_3 + T_d(c_4 + \\ &\quad T_d(c_5 + T_d(c_6 + T_d(c_7 + T_d(c_8 + T_d c9)))))))) \end{aligned} \tag{5.4}$$

where:

p_s	saturation vapor pressure (hPa);
e_{so}	known constant (6.1078);

$c_0 - c_9$	known constants;
T_d	dew point (°C).

Once the saturation vapor pressure is known, the air density can be calculated according to the following equation:

$$\begin{aligned} p_d &= p - p_s \\ \rho &= \frac{100p_d}{R_{da}T_{ok}} + \frac{100p_s}{R_{wv}T_{ok}} \end{aligned} \tag{5.5}$$

where:

p_d	dry air pressure (hPa);
p	air pressure (hPa);
p_s	saturation vapor pressure (hPa);
ρ	air density (kg/m^3);
R_{da}	dry air gas constant (J/kgK);
R_{wv}	water vapor gas constant (J/kgK);
T_{ok}	outside air temperature (K).

And finally, the cut-in and cut-out air speeds of the wind turbine have to be taken into account. If the air speed is above (cut-out) or below (cut-in) a certain value, the output of the turbine will be zero:

$$P_{turbine} = \begin{cases} P_w & \text{if } v_{in} \leq v_w \leq v_{out} \\ 0 & \text{otherwise} \end{cases} \tag{5.6}$$

where:

$P_{turbine}$	turbine power output (W);
v_{in}	cut-in air speed (m/s);

v_{out}	cut-out air speed (m/s).

The cut-in and cut-out speed are dependent on the turbine specifications.

Energy Sources: Photovoltaic Panels Photovoltaic panels are a common local generation means. We use the following formulation to model the photovoltaic panel's power output based on the work of Chen *et al.* [CGW12]:

$$P_{pv} = \eta S I (1 - 0.005(T_{oc} - 25)) \tag{5.7}$$

where:

P_{pv}	photovoltaic power output. (W);
η	panel efficiency (%);
S	array area (m^2);
I	solar irradiation (W/m^2);
T_{oc}	outside air temperature (°C).

The array area and panel efficiency depend on the configuration and on the specifications of the photovoltaic panels. The solar irradiation and outside air temperature depend on the location and weather. This model assumes that the panels utilize a two-axis maximum power point tracker. As a result, the surface of the panel is always perpendicular to the sunlight. Therefore, the direct normal solar irradiance is used in the calculation of the panel's power output.

Energy Sources: Prosumers and DSO The final energy sources considered in this work are the energy provided by neighboring prosumers and by the DSO. These are both seen as energy providers independently of the underlying technology used. For instance the prosumer could also have photovoltaic panels or storage systems, while the DSO also has access to the energy coming from the transmission infrastructure. For both, we need to define the cost of the energy, and the amount of energy available during a given time slot. For prosumers, the available energy (kWh) is sampled from a normal distribution:

$$X \sim \mathcal{U}_{[a,b]} \tag{5.8}$$

whereas the cost of the energy (€/kWh) is sampled from a Gaussian distribution:

$$X \sim \mathcal{N}(\mu, \sigma^2) \tag{5.9}$$

As typical in practice, we assume that the DSO can always fulfill the demand of the system. Though, the cost of energy from the DSO will typically be higher than those of prosumers, especially in times of high demand. As for the cost of energy supplied by the DSO, this is modeled after the day-ahead energy market prices for DSOs, which are then scaled to values more representative of real-world end-user prices.

Energy Storage: Battery Energy storage provides for important flexibility in complex smart energy systems. It can take many shapes and sizes, from fly wheels, to rechargeable batteries, to hydroelectric dams. In this work, residential energy storage in the form of household batteries is considered as it is the most easy and likely to be deployed in an office building. The battery should adhere to the following constraints, based on the definitions by Chen et al. [CGW12]. If $C(t)$ is the charge of the battery at time t, P_t^c the power charged to the battery, P_t^d the power discharged from the battery, and η is the charging efficiency, then:

$$\begin{aligned} \text{Charge: } C(t+1) &= C(t) + \frac{\Delta t P_t^c}{\eta} \\ \text{Discharge: } C(t+1) &= C(t) - \frac{\Delta t P_t^d}{\eta} \end{aligned} \tag{5.10}$$

under the maximum charge and discharge limits:

$$\begin{aligned} &0 \leq P_t^c \leq P_t^{c,max} \\ &0 \leq P_t^c \leq P_t^{c,max} \end{aligned} \tag{5.11}$$

energy storage limits:

$$C_{min} \leq C(t) \leq C_{max} \tag{5.12}$$

and the start and end limits:

$$C(t_{start}) = C(t) = C(t_{end}) \tag{5.13}$$

where $P_t^{c,max}$ and $P_t^{d,max}$ are the maximum charge rate and maximum discharge rate respectively, C_{min} and C_{max} are the minimum and maximum energy that the battery can store, and $C(t_{start})$ and $C(t_{end})$ represent the charge of the battery at the beginning and end of the scheduling period.

Scheduling Policies The way devices operate varies depending on their main function and user needs. This influences the way they are scheduled. To represent this, we utilize the policies defined in [GDP+12]. This set of policies is extended by us by including a battery policy. The policies are then translated into scheduling constraints when solving the optimization problem.

- *Total Policy*: defines the total amount of time a device d_i should be in state s_{ij}. For example, for any device with a battery that needs to be recharged, this is the time it takes to fully charge it. The amount of time is fixed, though it need not be continuous.

- *Continuous Policy*: similar to the *total* policy, if defines the total amount of time a device d_i should be in state s_{ij}. The key difference being that once the device is in state s_{ij}, it should remain in this state for the given amount of time without interruptions.
- *Repeat Policy*: defines the amount of time a device d_i should be in state s_{ij} with a certain periodicity. For example, a freezer that should be scheduled repeatedly every hour to maintain the appropriate temperature.
- *Multiple Policy*: combines the *continuous* and *repeat* policies. This policy is for devices that need to be in state s_{ij} for a certain number of times, but should not be interrupted before a certain deadline once they enter that state. For example, a printer can have multiple jobs to complete, but cannot be interrupted while executing the job.
- *Strict Policy*: defines a schedule that is known ahead of time, determining exactly when device d_i should be in state s_{ij}, for every available time slot. For example, a safety light that should be on from dusk till dawn. It would not be acceptable to only turn the light on for 15 minutes every hour.
- *Pattern Policy*: similar to the *strict* policy, it defines ahead of time what the expected usage of a device is. However, this policy does not offer the possibility of the device to be controlled, it simply provides the expected usage pattern.
- *Sleep Policy*: defines a no-op policy, during the entire specified period, device d_i should be in state s_{ij} and should not be operated upon. For example, this is useful when a device should be turned off for a period of time in order to save energy.
- *Battery Policy*: indicates that a device d_i is able to act as short-term energy storage, allowing for charging and discharging of energy. Examples of devices that can have the battery policy include stationary

home energy storage devices, as well as connected electric vehicles.

The formal definitions for each policy, except for the *battery* policy which we introduce for the purpose of our approach, can be found in [Deg14].

Uniform-cost Search The goal of our approach is to find the optimal solution to the previously defined problem. Optimal in this case, refers to the solution that minimizes the overall costs. The cost of the solution can be defined in multiple ways, for example, as the total cost of electricity, or as the carbon footprint associated with the energy consumption. To solve the problem optimally, the state space has to be explored. The state space can be interpreted as a weighted graph, where each node is a permutation of the state space, edges represent the transformation from one state to another and have an associated cost, the depth of the graph is equal to the number of time slots. Uniform-cost search is an uninformed search algorithm used for traversing graphs, finding the path with the lowest cost from the root node to the destination node [RN10]. In the context of this work, uniform-cost search expands the state space until either the optimal device schedule is found, or no schedule is found. Algorithm 5.1 describes the uniform cost-search algorithm, including the heuristics that are applied for this specific problem.

The algorithm is initialized by creating an empty node and adding it to the priority queue. Each node contains a permutation of states, and the cost associated with being in this state. The priority queue ensures that the node with the lowest cost is the first node in the queue. Next, the algorithm continues looping through the queue until a solution has been found, or until the queue is empty and no solution is found. In every iteration, the node with the lowest cost is removed from the queue, and a check takes place to verify if this node contains the solution. In case the node only contains a partial solution, the next set of states are expanded. For each state in this set, a new child node is created. Two heuristics are applied to reduce the complexity of the search space. If the child node passes the heuristic checks, it is added to the priority queue.

Algorithm 5.1 Uniform-cost Search

```
1: procedure UniformCostSearch(problem)
2:     initialNode ← new(node)            ▷ node.state = ∅, node.cost = 0
3:     queue ← push(queue, initialNode)                  ▷ priority queue
4:     visited ← ∅
5:     while notEmpty(queue) do
6:         node ← pop(queue)
7:         if goalReached(problem, node.state) then
8:             return solution(node)
9:         visited ← insert(visited, node)
10:        for newState in expand(problem, node.state) do
11:            c ← createChild(problem, node, newState)
12:            if policiesViolated(c.state) then
13:                continue
14:            else if c.state not in queue or visisted then
15:                queue ← push(queue, c)
16:            else if c.state in queue with higher cost then
17:                queue ← replace(queue, c.state, c)
18:            else
19:                continue
20:        end for
21:    end while
22:    failure
23: end procedure
```

The first heuristic check is to determine whether the newly expanded states violate any policies. If a policy is violated, the state is invalid and should not be explored further. For example, a device with a *sleep* policy from 01:00 AM till 06:00 AM should not be scheduled during this given period. If the device is scheduled during this period in the newly expanded state, then the state violates the policy and therefore the child node is pruned from the search space.

The second heuristic checks if there are partial solutions that have already been explored and that have the same outcome (cost and number of time slots) as the newly expanded state. In case the outcome is the same, but the cost is lower, the original node with the same outcome is replaced by the newly created child node. In case the outcome of both nodes are identical, the child node is pruned from the search space.

5.2.2 Design, Architecture, and Implementation

The cost optimization of smart grid enabled buildings requires the availability of a wide variety of data sources. For example, the reliable prediction of the renewable generation for the next day is based on the weather forecast. Weather services vary considerably in terms of the predicted features (radiation, temperature, wind, humidity, etc.), the spatial granularity, and the temporal resolution. The micro-service architecture that we design and propose enables the decoupling of the system's components, where each loosely coupled service adheres to the single responsibility principle [Bak17]. Furthermore, this architectural design promotes modularity, allowing services to be easily replaced by alternative services, even at runtime [KMM18]. An example is replacing the weather forecast of provider A, by the forecasting service of provider B, because provider B provides better local forecasts at an increased resolution without affecting the overall operation of the system.

The proposed micro-service architecture is shown in Figure 5.8. It consists of nine distinct micro-services, identified by the rounded rectangles. The arrows indicate the inter-dependencies between them. Four of these services depend on external components, which can be cloud services (e.g. DarkSky for weather data), or data repositories (e.g. MongoDB for storing device data). Communication between services is exclusively done via the REST architectural style, as each REST service exposes a REST API that accepts Hypertext Transfer Protocol (HTTP) requests. The Content-Type of each HTTP response is *application/json*. The deployment of the micro-services is handled by means of containerization, where each service is hosted within individual Docker container. What follows next is a description of each micro-service.

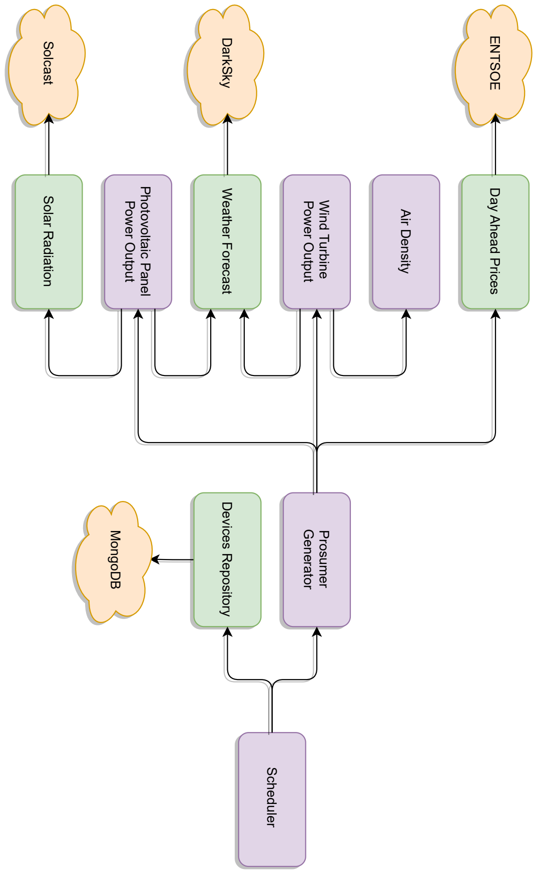


Figure 5.8: Microservice architecture. The green colored shapes represent services that can easily be exchanged to support a different data source. The purple shapes are the core services. The orange shapes are external services. The arrows represent the interdependencies.

Day Ahead Prices Service: requires an Energy Identification Code to identify the energy market that is used, and a date for which the day ahead prices should be retrieved. This service uses the cloud API provided by the European Network of Transmission System Operators for Electricity (ENTSO-E) transparency platform[1] in order to obtain realistic and real-time day ahead prices. The services returns the hourly day ahead prices for the specified date.

Air Density Service: requires the temperature, air pressure, and dew point in order to calculate the air density. This service implements the Herman Wobus vapor pressure polynomial to calculate the saturation vapor pressure at the dew point temperature, which can then be used to calculate the pressure of dry air. Finally, the service returns the air density for the given parameters.

Wind Turbine Power Output Service: requires the turbine's blade radius, wind speed, air density, and power coefficient to calculate the turbine's power output. This service also requires the cut-in and cut-out wind speed of the turbine to be provided. Based on these parameters, the turbine power output is given.

Weather Forecast Service: requires the latitude and longitude of the location for which the weather forecast should be obtained. This service returns the temperature, dew point, humidity, air pressure, and wind speed. The weather service used in this work is DarkSky[2], as it provides high temporal resolution forecasts for our desired location.

Photovoltaic Panel Power Output Service: requires the direct normal solar irradiance, temperature, area of the panel, and the efficiency of the panel. The service assumes that a maximum power point tracker is used. Based on the parameters provided to the service, the expected power output is returned.

[1]ENTSO-E – data and information on wholesale energy generation, transmission, and consumption, `https://transparency.entsoe.eu/`

[2]DarkSky – hyper-local weather information, `https://darksky.net/`

Solar Radiation Service: requires the latitude and longitude of the location for which the solar radiation should be collected. The solar irradiance data is provided by the cloud API from Solcast[1]. This service returns the direct normal irradiance, diffused horizontal irradiance, and global horizontal irradiance.

Prosumer Generator Service: requires the latitude and longitude of the location for which the data for simulated prosumers should be generated, as well as the specifications of the available PV and/or wind turbine resources. This service generates a list of energy sources which includes prosumers representing the neighboring buildings connected to the smart grid. Each prosumer is able to deliver a certain amount of energy for a certain cost during a certain period of time. Added to this list of prosumers are the available renewable resources of the building in question, as well as an energy source representing the traditional power grid in case the energy provided by the renewables and prosumers is insufficient to satisfy the demand.

Device Repository Service: returns the list of devices which should be scheduled, their power consumption in different states, as well as the policy for each device. The devices, their power states and their policies are stored in the document-oriented database MongoDB.

Scheduler Service: requires a list of energy sources (renewables, prosumers, and grid), a list of devices to schedule, the initial charge of the battery, as well as the algorithm that should be used. The modularity allows for selecting among versions of the uniform grid-search problem algorithm. The service returns the day ahead schedule of the devices with a 15 minute granularity, this predetermines the power state that each of the devices should be in for every 15 minute interval. Some meta-data on the performance of the algorithm is also included in the data that is returned by this service.

[1]Solcast – global solar irradiance data and PV system power output data, `https://solcast.com/`

5.2.3 Data Structures

There are three critical JSON data structures that are used in our approach; two which have to be provided to the scheduler, and one which is generated by the scheduler. First, a data structure with the available energy sources should be provided to the scheduler. This structure is shown in Listing 5.1, and contains the available energy sources, the amount of energy they can provide during a given period of time, and the cost for consuming the energy. The expression of the cost is flexible, it can be expressed in economic terms, but it is also possible to express the cost in terms of the carbon footprint of the energy source or any other meaningful totally ordered group.

```
[{
  "start_time": "2021-10-20T12:00:00Z",
  "duration": "01:00:00",
  "energy_sources": [
    { "name": "turbine", "cost": 0.15,
      "supply": 2.5 },
    { "name": "photovoltaic", "cost": 0.2,
      "supply": 1.2 },
    { "name": "prosumer-1", "cost": 0.3,
      "supply": 0.5 },
    ...
    { "name": "grid", "cost": 0.35,
      "supply": 10 }
  ]
},
...
{
  "start_time": "2021-10-20T13:00:00Z",
  ...
}]
```

Listing 5.1: An example of the data structure containing energy sources.

The second data structure which should be provided to the scheduler is the list of devices that are to be scheduled. This data structure is visible in Listing 5.2 and contains each device, the states in which the device can be, as well as the cost for being in that state, and the policies that are assigned to the device. In this example, there is no cost associated with charging and discharging the battery, though it is possible to assign a cost for these states that represents the degradation of the battery.

```
[{
  "name": "battery",
  "id": "BATTERY",
  "initial_state": 0,
  "states": [
    { "id": 1, "name": "charging",
      "cost": 0.0},
    { "id": 0, "name": "discharging",
      "cost": 0.0}
  ],
  "policies": [
    { "policy": "BATTERY",
      "capacity": 4.0, "rate": 1.0 }
  ]
},
...
{
  "name": "microwave",
  ...
}]
```

Listing 5.2: An example of the data structure containing devices.

The two previous data structures provide the scheduler with all the required information to solve the problem of finding an optimal schedule given the constraints that are applied to the devices. Once the solution has been found, the scheduler generates the third data structure: a schedule, which is shown in Listing 5.3. The schedule includes the start and end times of the schedule, also known as the scheduling horizon. It also includes the cost associated with executing this schedule. And finally, the schedule defines for each device the state it should be in for any given point in time. In this example, the fridge should be in state 0 (off) from 00:00, and in state 1 (on) from 00:30. To maintain the temperature of the fridge, it should be turned on regularly. This behavior is captured in the device policy of the fridge.

```
{
  "start_time":  "2021-10-20T00:00:00Z",
  "end_time": "2021-10-21T00:00:00Z",
  "cost": 15159,
  "schedule" : [
    {
      "id": "FRIDGE",
      "actions": [{
        "time": "00:00",
        "state": 0
      },{
        "time": "00:30",
        "state": 1
      },
      ...
      ]
    },
    ...
    {
      "id": "BOILER",
      ...
    }
  ]
}
```

Listing 5.3: An example of the data structure containing a schedule.

Parallel Uniform-cost Search As the number of devices and/or the scheduling horizon increase, the branching factor of the graph also increases. To counteract this increase in complexity, which leads to execution times that are too high for practical use of the system, a parallelized uniform-cost search was designed. While the core of Algorithm 5.1 remains the same, a number of important changes are made, Algorithm 5.2. Two global states are maintained: the priority queue, and the visited list. Each thread also maintains a local priority queue and visited list. When the local queue is empty, the thread takes a node from the global queue and starts expanding the state space while populating its local queue. Once the local queue reaches a certain threshold, it is merged with the global queue, emptied, and a new node is taken from the global queue. When a solution is found by a thread, all other threads pause their work and synchronize their local queue with the global queue. Next, the thread that found the solution checks the global queue to verify there is no node with a lower cost. If indeed there is no node with a lower cost, the solution is returned and all threads terminate.

5.2.4 Evaluation

To evaluate the quality and possibilities of the proposed architecture and scheduling approach, we consider offices. The specific setup consists of individual offices and a shared kitchen area, each with numerous devices. Smart plugs are used to collect power data from the devices situated in the offices and the kitchen area. This data is used to define the devices' profiles required by the scheduler. The renewables energy sources and energy storage are modeled after commercially available products. Combined with real-world weather data, the power output of the renewables is computed. The device profiles and energy sources are then used as input for the scheduler. Three different cases are considered; for each case we evaluate it with and without the availability of energy storage. For each generated schedule, the economic cost, the energy demand profile, and the battery charge are analyzed.

Algorithm 5.2 Parallel Uniform-cost Search

```
procedure UCS_PARALLEL(problem, threads)
    initialNode ← new(node)
    globalQueue ← push(queue, initialNode)
    globalVisited ← $\emptyset$
    solutionFound ← False
    for i ← 1 to threads do                          ▷ Starts a new thread
        queue ← pop(globalQueue)
        visited ← $\emptyset$
        while not solutionFound do
            node ← pop(queue)
            if goalReached(problem, node.state) then
                synchronizeThreads(threads)
                if peek(queue).cost > node.cost then
                    solutionFound ← True
                    return solution(node)
            Lines 11 to 23 from Alg. 5.1
            if length(queue) $\leq$ thresholdA then
                merge(globalQueue, queue)
                queue ← $\emptyset$
                queue ← pop(globalQueue)
                merge(globalVisited, visited)
                visited ← $\emptyset$
        end while
    end for
    failure
end procedure
```

Devices Two power states for each controllable device are considered. These states are defined based on the historical power data of the devices. The historical data was collected over a period of eight months, with measurements taken every 10 seconds. The K-means clustering algorithm [Mac67] is applied to specify the boundary between different power states based on the power values and the density of the points. In this work, 8 different devices are scheduled: a coffee machine, a fridge, a laptop, a microwave, a printer, a display, a thin client, and a battery based energy storage device.

As an example, Figure 5.9.(a) shows the historical power data and its histogram for the display device. As is seen from the histogram, as well as examined by the K-means inertia plot and elbow method [Ng12], the optimum number of clusters for the data set is two. Figure 5.9.(b) shows the clustering result highlighting the centroids of the clusters. The boundary between the two clusters is placed between the two centroids. Table 5.5 shows the cluster centroids for different devices with K-means clustering algorithm. S0 and S1 in this table, stand for two different power states.

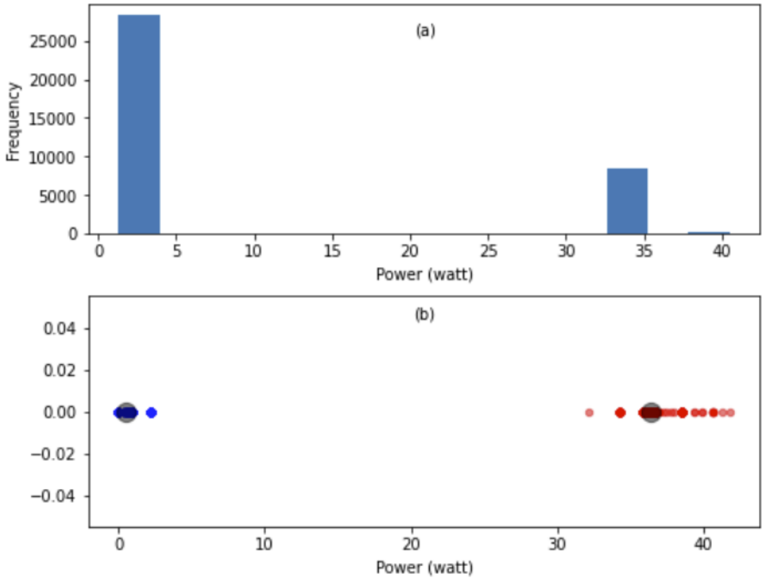


Figure 5.9: Display device data. (a) Histogram of power data (b) Clustering result of power data.

Each device needs to have a policy assigned to it in order for the scheduler to understand the constraints under which to schedule it. An overview of each device and the assigned policy is shown in Table 5.6. The coffee machine and the fridge require the *repeat* policy, as these devices need to be repeatedly turned on to maintain their temperature. The laptop uses the *total* policy as it needs to be charged for a total amount of time. The

Table 5.5: Centroids of the power states for different devices.

Device	Power state S0 (watt)	Power state S1 (watt)
Coffee Machine	22.48	2555.84
Fridge	0.05	50.8
Laptop	14.12	76.61
Microwave	4.3	1567.71
Printer	11.13	654.59
Display	0.63	36.36
Thin Client	0.07	10.52

microwave follows a predefined *pattern* based on the usage of this device by the users; it is used more during lunch time. The printer has multiple printing jobs that need to be performed, therefore it has a *multiple* policy. The display and its associated thin client are only on during office hours, and therefore require a *strict* policy. From the perspective of the scheduler, the battery energy storage is just another device that can be scheduled, though this device does not only consume energy, but can also return energy at a later point in time. The energy storage is assigned the *battery* policy.

The total daily energy demand for these devices is approximately 15.6 kWh. The majority of this demand originates from the coffee machine, the printer, and the microwave. The coffee machine needs to repeatedly reheat water to maintain the required temperature, the printer has numerous large printing jobs that need to be completed, and the microwave is used heavily during lunch breaks.

Energy Sources and Storage For local energy generation, the building has a single micro wind turbine, and a PV panel installation. The building also has battery storage. The geographical location of the building affects the energy sources, as the renewable energy generation is influenced by the weather, and the prosumers are constructed based on the day-ahead energy of the energy market. We choose as location Stuttgart, Germany.

Table 5.6: Device policies as assigned to each device.

Device	Policy
Coffee Machine	REPEAT
Fridge	REPEAT
Laptop	TOTAL
Microwave	PATTERN
Printer	MULTIPLE
Display	STRICT
Thin Client	STRICT
Battery Energy Storage	BATTERY

The wind turbine is modeled after a Sonkyo Energy Windspot 1.5, for which the technical specifications are shown in Table 5.7. The power coefficient is derived from the specifications as the manufacturer does not provide it explicitly. It should be noted that typically the specifications are optimistic, and that in practice the power coefficient will be lower. However, for the purpose of this evaluation the differences between theoretical and practical performance are not of major importance. The cost of locally generated wind energy is fixed at 0.08 €/kWh.

Table 5.7: Windspot 1.5 - technical specifications.

Rotor Swept Area	12.88 m^2
Rated Power	1.5 kW
Rated Speed	12 m/s
Cut In Speed	3 m/s
Survival Speed	60 m/s
Power Coefficient	0.11

For the PV panels, it is assumed that the building is equipped with two-axis maximum power point tracking solar panel installation. The specifications of the panels, as shown in Table 5.8, are based on the HiTech Solar 250Wp Black 60 cells panel. The installation consists of 6 panels, with a total of 1.5 kW peak power, and an area of 9.9 m^2. The cost is set to 0.06 €/kWh.

Table 5.8: HiTech Solar 250Wp - technical specifications.

Dimensions	1x1.65 m
Panel Area	1.65 m^2
Cells	60
Technology	Monocrystalline
Maximum Power	250 Wp
Panel Efficiency	15.30 %

As for the battery-based energy storage, the assumption is that the building has a single battery storage installation. The assumed energy storage is based on AlphaESS SMILE3. The technical specifications of the battery energy storage are given in Table 5.9.

Table 5.9: SMILE3 - technical specifications.

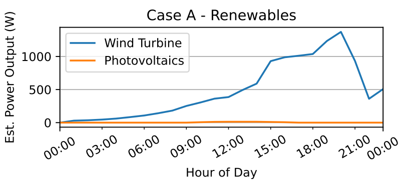
Capacity	2.8 kWh
Charging/Discharging Current	60 A
Charging/Discharging Power	3000 W
Depth of Discharge	95 %

Both the prosumers and the grid are modeled using the day ahead prices of the German electricity market, as published on the ENTSO-E Transparency Platform. To model the grid, the hourly cost of energy is to be equal to the hourly day-ahead prices as obtained from ENTSO-E. For the purpose of this evaluation, we assume that the grid is always able to satisfy the energy demand of the building, in case the renewables and prosumers are not able to. The obtained grid prices are normalized between €0.40/kWh and €0.60/kWh. The prosumers are also modeled based on the hourly day-ahead prices. It is assumed there are 10 different prosumers available at all time. To introduce some variability, the hourly energy cost for each prosumer is sampled from Equation 5.9, where $\mu = \frac{grid\_price}{1.5}$ and $\sigma = 0.025$. The available energy for each prosumer is sampled from Equation 5.8, where $a = 0$ and $b = 1$.

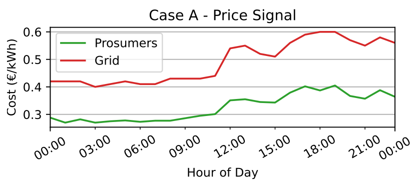
Results (Economic) Three distinct cases are considered in order to evaluate the economic benefits of scheduling devices with and without energy storage in the presence of locally generated renewable energy. Each case has a unique price signal, modeled on the day ahead prices of the German electricity market. Furthermore, each case also has a unique renewable energy profile which is dependent on the weather forecast for that day. The renewable energy profile and the price signal for Case A (February 5th, 2022) are shown in Figures 5.10a and 5.10b, respectively. Case A is a cloudy and extremely windy day, especially after 14:00. The price signal for case A shows that the price is relatively low until around 11:00. The prosumer signal is the average price signal of the 10 available prosumers. Case B (February 6th, 2022), as shown in Figures 5.10c and 5.10d, is a windy day with some PV generation throughout the day. And finally, Case C (February 8th, 2022) is a sunny day with wind speeds below the cut-in wind speed of the turbine, as can be seen in Figure 5.10e. The price signal is low for the majority of the day, until approximately 16:00, as shown in Figure 5.10f.

The energy sources and their prices, as well as the list of devices, form the input of the scheduler. In total, six schedules are generated. Two schedules for each of the three cases, one with energy storage, and one without energy storage. The values of energy sources do not change for each individual case, the only change is the inclusion and exclusion of the energy storage. The scheduling horizon is set to 24 hours, the length of each scheduling period is set to 15 minutes, and the energy storage is empty at the start of the day.

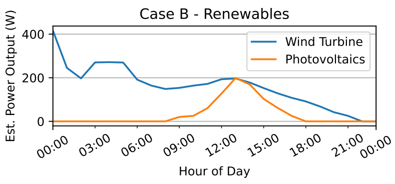
Figure 5.11a, 5.11b, and 5.12a show the cumulative charge of the energy storage device (battery), the cumulative cost, and the cost per period, for case A. The cumulative charge is only applicable to the schedule with energy



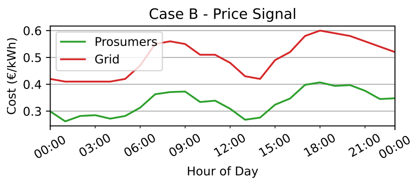
(a) Case A, renewable energy production



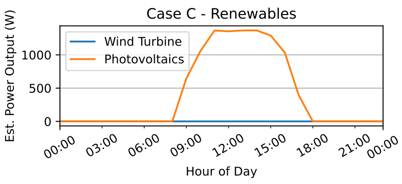
(b) Case A, grid and prosumers price signal



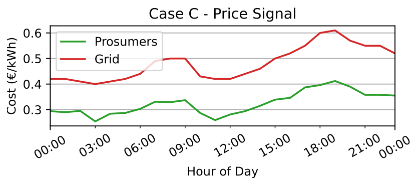
(c) Case B, renewable energy production



(d) Case B, grid and prosumers price signal



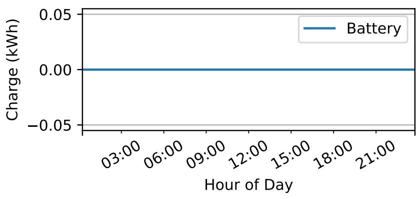
(e) Case C, renewable energy production



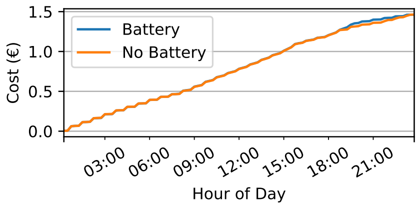
(f) Case C, grid and prosumers price signal

Figure 5.10: The renewable energy production and price signals for each of the three considered cases.

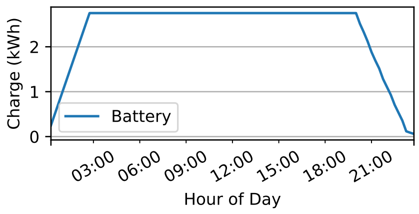
storage. The remaining two figures visualize the results for both the schedule with energy storage, and the schedule without energy storage. It is apparent from Figure 5.11a that the optimal schedule, i.e. the schedule with the lowest cost, does not charge or discharge the battery. The reason for this behavior is the availability of cheap locally generated renewable energy (wind) throughout the day. The final cumulative cost is identical for both schedules, though the scheduling of devices is slightly different as can be seen in Figure 5.12a, where the cost per period differs slightly from 18:00 onward. This difference can be explained by the parallel nature of the algorithm used by the scheduler, it is possible that the search space is explored in a different



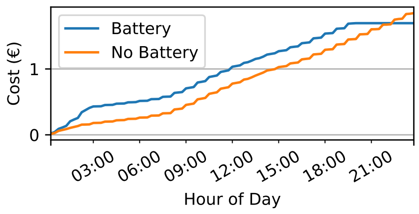
(a) Case A, cummulative charge



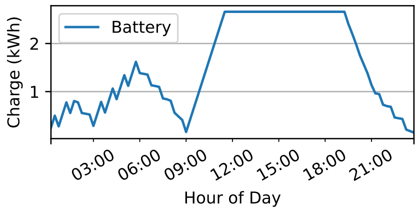
(b) Case A, cummulative cost



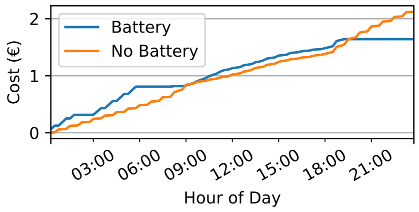
(c) Case B, cummulative charge



(d) Case B, cummulative cost

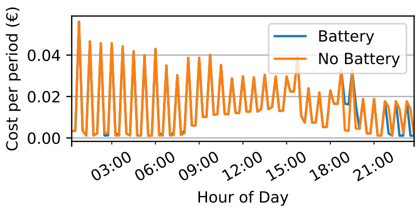


(e) Case C, cummulative charge

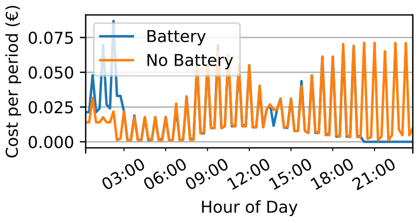


(f) Case C, cummulative cost

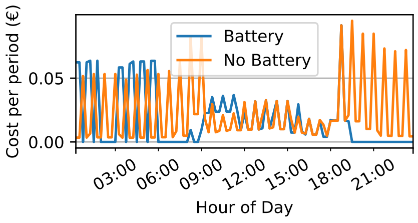
Figure 5.11: The cumulative battery charge and cumulative cost associated with the generated schedules.



(a) Case A, cost per period



(b) Case B, cost per period



(c) Case C, cost per period

Figure 5.12: The cost per period associated with the generated schedules.

order, though this clearly does not affect the goal of minimizing total cost. The peaks in the cost per period graph are caused by the fridge and coffee machine repeatedly turning on and off with a certain periodicity.

The schedules for case B clearly exhibit different behavior when compared to case A. Figure 5.11c demonstrates this, the battery is charged in the beginning of the day, and is not discharged until the end of the day. When looking at the availability of locally generated renewable energy, and the price signal, it becomes clear why case B behaves in such way. Until approximately 19:00 there is sufficient renewable energy available to meet most of the demand. Simultaneously, the price signal remains high after 17:00. Therefore, the battery is charged in the early hours of the day, as cheap locally generated renewable energy is available, and is discharged after 19:00, as the renewable energy generation is not sufficient, and the price signal as high.

For case C, as visible in Figure 5.11e, the charge of the battery fluctuates between 00:00 and 09:00, after which it is fully charged, and remains fully charged until 19:00. From Figure 5.11f one notices that the cumulative costs of the schedule which includes the battery are significantly lower. These savings are obtained after 19:00, when the price signal is high and there is no renewable energy available, as visible in Figure 5.12c.

An overview of the economic costs associated to each of the 6 schedules is shown in Table 5.10. The price signal and availability of locally generated renewable energy greatly affect the potential economic benefits of utilizing energy storage. Regardless of these factors, the scheduler is always able to find the optimal schedule, whether this includes utilizing energy storage or not.

Table 5.10: Overview of economic savings.

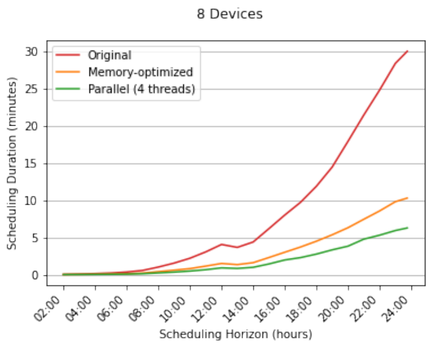
Case	**Battery?**	**Cost (€)**	**Cost Difference**
A	No	1.46	-
A	Yes	1.46	-
B	No	1.84	-
B	Yes	1.69	-€0.15 (8.15%)
C	No	2.12	-
C	Yes	1.64	-€0.48 (22.64%)

Results (Performance) When optimizing energy it is important that the system is fast enough to be used in the building and that it does not consume more resources than it saves. To study this we perform the analysis on a Dell PowerEdge R7425 server with two AMD EPYC 7551 2.0GHz 32-core processors, 512GB of RAM, and read-optimized SSDs with a throughput of 560MB/s. The bottleneck of this test bed is the relatively low clock speed of the AMD EPYC 7551, which limits the number of operations per second when executing the uniform grid search.

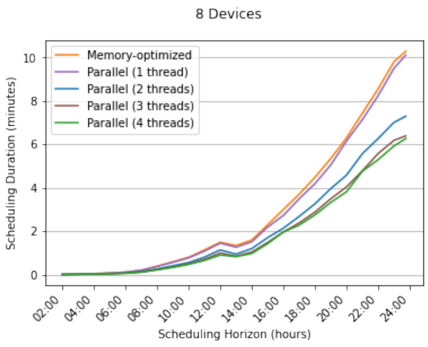
To evaluate the performance of the algorithms, a set of test cases is generated by varying numerous parameters. These parameters include: the number of devices, scheduling horizon, number of threads, number of iterations, and algorithm variation. The number of devices ranges from 1 to 12. The scheduling horizon ranges from 2 hours to 24 hours, in 1 hour increments. The number of threads ranges from 1 to 16. The number of iterations is set to 10, and indicates how many times the same test case is run. And finally, the algorithm variations include the original algorithm from the work of Georgievski et al. [GDP+12], a variation of the original algorithm that minimizes memory operations, and the parallel algorithm proposed in this work. The original algorithm was modified to support battery device policy and the multitude of micro-services. The result is 27.600 test cases.

Due to the high dimensionality of the resulting data, it is not possible to visualize all of the results in their entirety. Therefore, we focus on a number of specific cases which characterize the general performance of the approach. Figure 5.13a shows the mean performance when scheduling 8 devices with a varying scheduling horizon going from 2 to 24 hours. This case is selected because the scheduling problem is complex. The time it takes to find the optimal schedule (scheduling duration) is measured in minutes. It is clear that the memory-optimized and parallel algorithms perform considerably better than the original algorithm. The more complex the search space (the broader the scheduling horizon), the more pronounced the difference becomes. At the 24 hour scheduling horizon, the original algorithm (30 minutes) is 2.9 times slower compared to memory-optimized (10.2 minutes), and 4.7 times slower compared to parallel (6.2 minutes) with 4 threads.

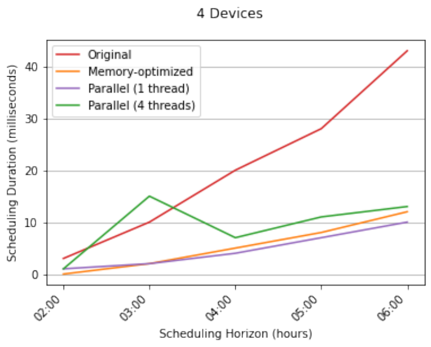
Figure 5.13b shows the same results as the previous figure, except the original algorithm is excluded, and the results for the varying thread count parameters are added. This highlights the performance differences between number of threads. The memory-optimized and parallel (1 thread) algorithms have identical performance (10.2 minutes at 24:00 hours scheduling horizon), as is expected due to the fact that the parallel algorithm is derived from the memory-optimized one. Increasing the thread count from 1 to 2, improves the performance by a factor of 1.4 (7.2 minutes at 24:00).



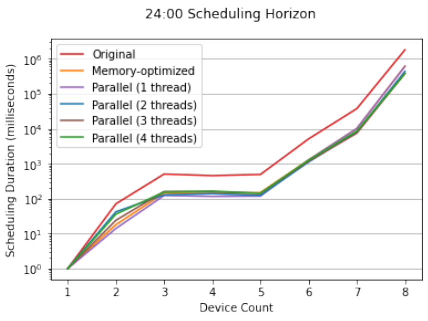
(a) 8 devices, 24 hours.



(b) 8 devices, 24 hours, excluding original.



(c) 4 devices, 6 hours.



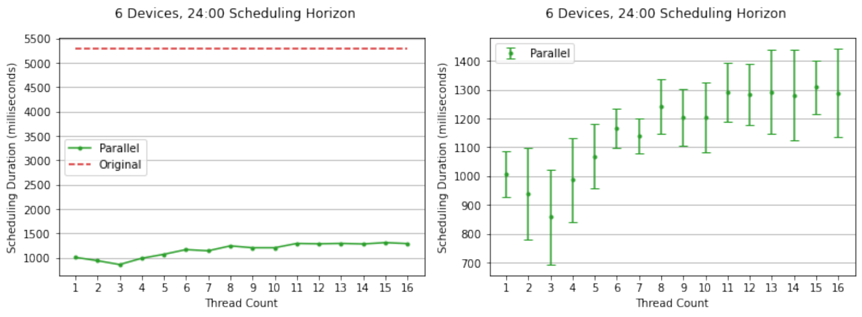
(d) 1 to 8 devices, 24 hours.

Figure 5.13: Performance of the variations of the uniform grid search algorithm.

When increasing the thread count to 3 or 4, the performance improves by a factor of 1.6 (6.2 minutes at 24:00) for both cases when compared to the memory-optimized results. This is due to the implementation of the concurrent priority queue. As this implementation requires synchronization when adding and removing nodes to the queue, acquiring the lock becomes the bottleneck.

In Figure 5.13c, the case of 4 devices and a 06:00 hour scheduling horizon is considered. This case is selected because complexity-wise it is the opposite of the previous one: the search space is small. When the search space is small, the overhead of synchronizing 4 threads becomes larger than the time it takes for the single threaded variations of the algorithm to find the solution. This is clearly visible at 03:00, where the parallel algorithms (with 4 threads) requires 15 milliseconds to complete, while parallel (with 1 thread) requires only 2 milliseconds. In practice, the search space is typically more complex, and a difference of 13 milliseconds is negligible in the broader perspective of the system's execution time.

In the three preceding cases the number of devices is fixed. Figure 5.13d (logarithmic y-axis) demonstrates the effect of increasing the device count, and therefore increasing the search space. The complexity does not change significantly when adding the 4th and 5th devices, this is due to the policy that is assigned to these devices. Some policies (such as *pattern*, *strict*, and *idle*) only minimally increase the search space. When ignoring the 4th and 5th devices, as the number of devices increases, the scheduling duration increases exponentially.



(a) 8 devices, 24 hours. (b) 8 devices, 24 hours, parallel algorithm only.

Figure 5.14: Thread counts vs. performance.

To clearly illustrate the bottleneck problem caused by the locking mechanism of the queue, the number of devices is reduced to 6, the thread is set to range from 1 to 16, and the scheduling horizon is fixed at 24:00 hours. Figure 5.14a visualizes the mean scheduling duration in milliseconds with varying thread counts. The original algorithm (red dashed line) is included for reference. For this particular scheduling problem, it is clear that 3 threads provides the fastest scheduling duration. As the thread count grows beyond 3, the synchronization mechanism in the queue becomes the bottleneck. In addition, Figure 5.14b shows the standard deviation for each thread count obtained after 10 iterations.

5.2.5 Discussion, Limitations and Outlook

Previously, Georgievski et al. demonstrated the concept of scheduling devices against a price signal in order to decrease energy cost in an office building [GDP+12]. The present work expands on that experience and improves upon it in multiple ways. First of all, by adding real time modeling of renewable energy generation, as well as the inclusion of energy storage. Second, the modeling of energy sources is supported by a loosely coupled, modular micro-service architecture which enables the system to adapt to different service providers. Furthermore, real world data is used for the scheduled devices, and for the prediction of renewable energy generation. And finally, significant performance enhancements are made by introducing a parallel uniform cost-search algorithm.

From the economic perspective, the inclusion of energy storage coupled with device scheduling can result in significant cost savings. However, as shown by the presented results, this depends greatly on the price signal and on the availability of locally generated renewable energy. For example, when sufficient amounts of energy are generated throughout the day, the battery is not scheduled to charge, as charging is not free and the number of charge cycles of a battery is limited. Therefore, in the worst case, the economic benefit of adding energy storage is null on a daily basis. Nevertheless, in several cases conditions are beneficial for energy storage, for example when price signals fluctuate and the locally generated renewable energy is not sufficient to meet the demand, then there are significant cost savings to be made. The results show cost savings of up to 8.15% per day, going as high as 22.64% under the more favorable conditions. While the evaluation focuses on the economic perspective, the algorithm is agnostic with regards to the costs. In other words, instead of providing a list of energy sources and their cost expressed in euros per kWh, one could provide instead the cost expressed as grams of CO_2 equivalent emissions per unit of power (gCO2eq per kWh) to improve the sustainability rather then the economic costs.

From the performance perspective, the memory optimizations and parallelization of the uniform cost-search algorithm result in significant performance gains. The original algorithm requires up to 30 minutes to schedule 8 devices with a scheduling horizon of 24 hours. The memory optimizations alone reduce this time to 10 minutes. Further gains are made by parallelizing the uniform-cost search algorithm, completing the scheduling in 6.2 minutes with 4 threads. It is also possible to spread the load over multiple machines using distributed agents, where each agent solves a part of the problem. However, accessing the shared resources requires synchronization, and therefore needs to be executed sequentially. This sequentially executed section of the algorithm becomes the bottleneck. The bottleneck is a manifestation of Amdahl's law [Amd07], which in the context of parallel programming describes that the maximum possible speed-up is disproportionately limited by the sequential section of the application.

Finding the optimal solution to a scheduling problem is a time consuming task. In our approach, the scheduler operates on 15 minute intervals with a scheduling horizon of 24 hours. Decreasing the size of the intervals, expanding the scheduling horizon, and increasing the number of scheduled devices all exponentially increase the complexity of the problem. To counteract these limitations, the scheduling problem can be divided into sub-problems, which are solved individually and later combined into one schedule. While the local schedules are optimal, the global schedule is likely not to be.

The energy consumption associated with the schedule generation is not included in the economic evaluation since it is negligible, as we compute next. The time it takes to solve the scheduling problems is in the order of several minutes. The system only needs to be available during this time period.

The execution of the schedule can be performed by a simple embedded device. As per the performance evaluation, the scheduling of 8 devices requires at most 6.2 minutes when using 4 parallel threads. The 32-core AMD EPYC 7551 CPU is responsible for the majority of the system's power consumption. Assuming each of the 4 parallel threads utilizes 100% of their allocated CPU core, the overall CPU utilization is 12.5%. According to the SPECpower_ssj2008 benchmarks [Spe], this would equate to a power consumption of 100W. Therefore, generating the schedule requires 0.01 kWh. We can conclude that the cost of the energy consumption of the system itself is insignificant with respect to the overall economic savings.

From a system architectural point of view, we proposed a micro-service architecture that encapsulates the three critical components in several micro-services. This enables the system to leverage the real time data provided by weather services and the ENTSO-E transparency platform to generate realistic price signals on the fly. Furthermore, a performance-oriented version of the uniform cost-search algorithm is proposed, which parallelizes the search for the optimal schedule. The performance evaluation shows that this parallel algorithm decreases the time to find the optimal schedule by a factor of 4.7 compared to the previous approach. And finally, this work evaluates the economic benefits of including local energy storage in the scheduling problem.

5.3 Summary

In this chapter, our focus has shifted in this chapter from smart data centers to smart buildings. The treatment has highlighted that many parallels exist between the two. In both cases sustainability is an important driver behind the need for smartness and intelligent decision making. The ability to control the building and influence its environment through IoT-enabled devices is required to achieve the desired level of control. And, as we have seen in the previous chapter, IoT devices also play a critical role in the real-time monitoring of data centers. Furthermore, the IoT platform developed in the case of the Bernoulliborg is nearly identical to the one used in Chapter 4 for the monitoring of data centers. This is not a coincidence, as the system for monitoring data centers is an adaptation of the system deployed in the Bernoulliborg. The underlying concepts and protocols are identical, though a number of core technologies have been superseded by more suitable alternatives. While the focus in this chapter was on scheduling appliance loads, the scheduling of loads is also important in the context of data centers. Though, in data centers it is typically application workloads that are scheduled on servers. Generally, these workloads are split into deferrable and non-deferrable workload types. This distinction can also be made in buildings, where some loads can be deferred, such as turning on the washing machine, and others cannot, such as turning on lights in an occupied room.

Where the Bernoulliborg is considered, we have demonstrated in a very practical manner the benefits of smartness, but also emphasized the selection of sensors and actuators that are necessary to achieve the required level of control and optimization. The results show that the annual economic savings are significant, up to € 658, and energy savings of up to 80% were also observed. The service-oriented architecture combined with AI planning, activity recognition, and the IoT paradigm was essential to obtain these savings. Furthermore, the majority of building occupants had a positive response to the experiments that were performed. We also investigated the scheduling of device loads in an office environment while considering renewable energy generation and energy storage, as well as varying price signals. For this purpose, we developed a parallel uniform-cost search algorithm that significantly outperforms traditional uniform-cost search by speeding up the search by 4.7 times. Each of the subsystems of our proposed architecture was encapsulated in a micro-service. Our evaluation included eight devices, two renewable energy source, and one energy storage device. By optimally scheduling the device loads based on the given constraints we were able to obtain up to 22.64% economic savings. While the area of application has changed in this chapter from data centers to buildings, the techniques and tools that we have used are highly related and relevant in both domains.

Chapter 6

Conclusions and Outlook

Data centers are steadily growing in size and number, and their overall environmental footprint is increasing correspondingly. This trend conflicts with modern policy making, which is focused on sustainability and on achieving climate goals. The use of IoT in data centers for monitoring, and subsequently to support optimization, is a promising method to improve the efficiency of data centers. In this thesis, we looked at data centers through the lenses of IoT and data-driven approaches. We also extended our research to the domain of office building optimization. In Chapter 1, we identified numerous challenges regarding these research topics and defined corresponding research questions. In the following sections, we will reflect on each of our research questions and how the present work answers them.

In Section 6.1, we discuss research questions RQ1 and RQ2, which are both related to sustainable data centers. Next, in Section 6.2, we shift our focus to the monitoring of smart data centers and review research questions RQ3 and RQ4. The final two research questions, RQ5 and RQ6, are related to smart buildings and are discussed in Section 6.3. And finally, we provide our vision on possible future research directions in Section 6.4.

6.1 Sustainable Data Centers

As sustainability is becoming increasingly important in modern society, the need for sustainable data centers is pressing. Understanding which actions can be taken to increase the data center's efficiency is an important step towards sustainability. This leads us to our first research question:

> RQ1 – *What measures can be implemented by a data center operator to increase the level of efficiency and sustainability of their data center, and which of these measures are implemented in practice today?*

We addressed research question RQ1 in contribution C1. Through our interviews with key personnel from 7 data centers, we investigated the adoption of 23 best practices which can be implemented to increase the efficiency and sustainability. Almost all data centers follow the best practices for storage and networking. The majority of the cooling, thermal, and air management practices are also implemented in practice. Though, liquid cooling is only used in one data center. Only slightly more than half of the energy efficiency practices are used. None of the data centers use automated lighting solutions in combination with occupancy sensors to reduce their energy footprint. Furthermore, none of the data centers use on-site power generation which can reduce conversion losses and allows for the re-use of the generated heat. The area of green practices is where most improvements can be made; none of the data centers implemented all of the green practices. Specifically, none of the data centers use reclaimed water, and only a few utilize water-side or air-side economizers. The re-circulation of cooling water and the use of renewable resources is also severely lacking.

Additionally, we emphasized the use of immersion cooling as a measure to increase efficiency and sustainability. Where traditionally air-cooled data centers have a PUE between 1.1 and 2.9, immersion-cooled data centers yield significant improvements with PUEs ranging from 1.02 to 1.04. Other benefits of immersion cooling are a significantly higher power density, allowing for more computational capacity per cubic meter. The operational costs are also reduced due to the absence of internal fans. While retrofitting an existing data center with immersion cooling is not practical or cost efficient, designing a new data center with immersion cooling in mind promises to significantly increase the data center's efficiency.

To summarize, there is a multitude of measures available that can be taken by data center operators to increase the level of efficiency and sustainability of their data center. We have shown that in practice not all of these measures are implemented, and that particularly the area of green practices has significant room for improvement. Once a given measure is implemented, its effectiveness on the data center needs to be evaluated. This brings us to the next research question:

> RQ2 – *How can the effect of policy changes in a data center, such as the implementation of best practices, be measured in a standardized manner?*

We addressed research question RQ2 in contribution C2. Data center metrics play a critical role in evaluating the state of the data center and monitoring the effect of policy changes. By developing a taxonomy of data center metrics, we obtained an overview of 136 metrics across 9 categories. For each metric we reported their unit, optimization objective, optimal value, and at which level of the data center they operate. Furthermore, we have analyzed the relationships between metrics, specifically focusing on metrics that re-use existing metrics or are derived from other metrics. Energy efficiency metrics are heavily depended upon each other, with many of these metrics re-using other energy efficiency metrics. The green metrics category also contains a number of inter-metric relationships, but more crucially it is clear that there is a lack of green metrics focused on recycling and renewable energy sources. Where thermal and air management metrics are concerned, the relationships are not so much between the metrics, but between the input parameters of the metrics, 9 of these metrics depend on the same 6 input parameters.

We also uncovered numerous issues and challenges. First of all, there is no metric that encapsulates all aspects of a data center, nor are there metrics that specifically enable the comparison of data centers. In practice, PUE is used for comparisons, though the PUE metric was never intended for this purpose and has numerous flaws that prevent fair comparisons. For example, PUE is heavily influenced by the IT load, the geographical location, and the weather. Other issues include the challenge of defining what work in the data center is *useful* work, as many metrics depend on this notion of usefulness. Furthermore, co-location data centers have the additional challenge of not being able to evaluate metrics that depend on data collected from IT equipment due to a lack of access. We also identified the need for IoT-driven in-detail monitoring of data centers to enable the continuous evaluation of metrics.

In short, the effect of policy changes can be measured through a plethora of standardized data center metrics that are available to be used at different levels of the data center. However, one has to be cognizant of the inter-relationships and weaknesses of the chosen metrics, as the flaws that exist within one metric are not necessarily overcome when it is re-used in another metric.

6.2 Monitoring of Smart Data Centers

The amount of data that can be collected through the use of IoT-assisted monitoring is truly astounding. In fact, the transmission of such vast quantities of data has a negative impact on the networking bandwidth. To overcome this problem, we posed the following research question:

> RQ3 – *How can the traditional network architecture of a data center be leveraged to support IoT-based real-time monitoring while simultaneously reducing the processing load and bandwidth consumption associated with monitoring?*

We addressed research question RQ3 in contribution C3. By analyzing the data generation potential of real-time server monitoring in data centers we have quantified that the impact on the network is significant.

Our experiments demonstrated that each server can continuously generate monitoring data at 92.2 kilobyte per second. Assuming Google's Dallas data center consisting of 9090 server racks has a density of 25 server per rack, we extrapolate the total bandwidth consumption to be 167.62 gigabit per second. Thus, we proposed an edge-based approach, taking advantage of the data center's network architecture. By co-locating an edge gateway with the ToR switch in every rack, we can process the data at the edge, reducing the bandwidth requirement to 18 megabit per second for 25 server per rack, or 47 megabit per second for 64 servers per rack. Furthermore, we see clear parallels between the proposed edge gateway and IoT hubs, especially when edge gateways are also utilized to connect to external hardware sensors.

There is a multitude of freely available open-source IoT hubs. Initially, we selected 20 IoT hubs and, based on 5 parameters, we reduced our selection to the 4 highest scoring systems: Home Assistant, Domoticz, openHAB, and ioBroker. We analyzed the architecture of each system and extracted a generic IoT hub architecture based on the commonalities. The parallels between the edge gateway and generic IoT hub are immediately apparent. We defined 17 use cases to extract 13 features that are required to support all of the use cases. Almost all of the four systems support each feature, though some systems rely on 3rd party plugins. We also subjected each system to 34 criteria ranging from popularity and community to performance and support. Our results present Home Assistant as the victor, though the results are more nuanced. While Home Assistant offers a well rounded experience, Domoticz provides an uncomplicated setup, ioBroker has excellent performance, and openHAB as strong scores across the board.

To conclude, we leverage the network architecture of a data center by co-locating edge gateways with the ToR switches which allow the processing of monitoring data to take place at the edge, reducing the bandwidth consumption as well as the latency. Furthermore, IoT hubs are perfect candidates to be co-located with the ToR switches, as they offer the functionality that is required to process data and monitor the data centers. By investigating 4 IoT hubs in detail, we uncovered the strengths and weaknesses of each system, assisting in the selection process. While the approach we presented is feasible in traditional data centers, co-location data centers face the challenge of not having access to the IT equipment. Thus, we reach our fourth research question:

> $\boxed{\text{RQ4}}$ – *With which precision is it possible to monitor servers of a co-location data center while not having access to the operating system or the internals of the server chassis?*

We addressed research question $\boxed{\text{RQ4}}$ in contribution $\boxed{\text{C4}}$. We monitored 13 metrics of 164 servers in a data center, which resulted in a data set consisting of 2.5 billion data points. To replicate the conditions of monitoring in a co-location data center, our aim was to monitor the CPU utilization and temperature of servers using only measurement taking external to the server, such as power consumption and the temperature difference between the inlet and outlet air. This would enable privacy-preserved monitoring. In our approach we used linear regression model with polynomial features, as we observed near linear correlations between metrics. By training and evaluating 10,470 different models using k-fold cross validation we determined that we can determine the CPU utilization with an error between 7.35% and 8.14% by only using power consumption and the temperature difference as inputs to the model. Similarly, the CPU temperature can be determined with an error between 3.17$°C$ and 3.84$°C$. And finally, we modeled the outlet temperature using power consumption and inlet temperature as input parameters and obtained an error between 1.07$°C$ and 1.25$°C$.

In conclusion, we can monitor the CPU utilization and CPU temperature of servers with the given degree of precision despite using only the data collected externally to the server. Therefore, we do not require access to the operating system or server chassis. Our approach relies on IoT devices to measure and monitor the required external parameters, such as the power consumption, as well as the inlet and outlet air temperatures.

6.3 From Smart Data Center to Smart Building

Next, we shift our attention to smart buildings. Simply deploying IoT sensors and actuators in a building does not inherently create an environment that is able to intelligently respond to the behavior of its occupants. Yet IoT does play an important role in both smart building and smart data centers alike. This brings us to the next research question:

> RQ5 – *What role does the IoT paradigm have in enabling intelligent responses to activities taking place in office buildings, and which similarities and differences exist between IoT architectures for offices and data centers?*

We addressed research question RQ5 in contribution C5. We deployed a multitude of IoT devices in an office building, including passive infrared sensors, light intensity sensors, smart plugs, and even our own custom software that transforms workstations into virtual sensors. While the intelligent responses were primarily enabled by activity recognition and AI planning, the input data required for both these processes was supplied by IoT devices. Therefore, it is critical that the data supplied by the IoT sensors is accurate and timely. However, in our deployment in the restaurant area, we encountered multiple interference issues with two types of sensor networks. The microwave in the social corner area was also causing interference when in use. Identifying these sources of interference and deciding on the type of network, wireless or wired, is an important part of the IoT deployment process. Despite these challenges, we observed energy savings of up to 75.5% in the offices and social corner, and up to 89% in the restaurant.

Throughout the thesis, we have illustrated a number of different architectures. First, we proposed the edge gateway architecture for data center monitoring. Next, we defined a generic IoT hub architecture, followed by a service-oriented architecture for collecting, processing, and storing server monitoring data. We also developed an architecture for smart buildings, with additional services that enable intelligent control. And finally, we used a micro-service architecture to collect weather data, solar radiation data, and day ahead energy prices from cloud services to be used in the generation of device load schedules. There are many obvious similarities between these IoT-oriented architecture for data centers and offices. For example, in both domains there are many different types of sensors and actuators, and thus there is a need to standardize the integration with different sensor technologies. The storage and processing of data, as well as responding to state changes is important in both cases. A key difference is in the types of sensors and actuators that can be found in each domain. While in the office domain it is typically physical sensors, such as motion sensors, light sensors, and smart plugs, in the data center domain the majority of the sensors are defined in software by monitoring agents deployed on IT equipment. Another important difference is in the scheduling of loads in each domain. In the office domain, the loads are typically from a single device, such as a fridge, a coffeemaker, or a microwave. In the data center domain it is also possible to schedule loads, but these loads are typically workloads or processes running on a server. It is possible to consolidate multiple workloads on a single server, to defer their execution, or even to move workloads on a geographical scale.

In summary, while IoT devices on their own do not provide an overarching sense of intelligence or smartness, the combination of IoT with techniques such as activity recognition and AI planning does enable the building to intelligently respond to activities occurring within it. There are also many similarities in the IoT architectures found in data centers and offices, though there are significant differences in the types of sensors that can be found in each domain, as well as in the way loads can be scheduled. The concept of a smart grid increases the complexity of load scheduling problems, especially when you include varying price signals, the generation and storage of energy, and scheduling constraints for each device. This leads us to the sixth and final research question:

> $\boxed{\text{RQ6}}$ – *How can devices in an office building be scheduled optimally considering the availability of renewable energy, energy storage, neighboring prosumers, varying price signals, and different scheduling constraints, while ensuring timely schedule generation?*

We addressed research question $\boxed{\text{RQ6}}$ in contribution $\boxed{\text{C6}}$. We defined an optimization problem for the optimal scheduling of device loads given the scheduling constraints for each device, price signals from the grid and neighboring prosumers, renewable energy generation, and the availability of energy storage. Optimality in this case means reducing the economic costs to the minimum necessary to satisfy the user's needs. We developed an extension of the uniform-cost search algorithm which is able to parallelize the task of finding the optimal solution to the optimization problem in order to speed up the search process. Furthermore, a micro-service architecture was designed to collect data from cloud services and to integrate with the BMS. We then used real-world data for the evaluation of our approach. The evaluation focused on the difference between schedules that use energy storage, and schedules that do not. When using energy storage, savings between 8.15% and 22.64% were observed. Furthermore, the optimization and parallelization of the algorithm has reduced the search time from 30 minutes to 6.2 minutes.

To reflect on the research question, parallel uniform-cost search enables us to find the optimal load schedule for devices given constraints, price signals, energy generation, and energy storage. Furthermore, the parallelization of the search task enables the timely generation of the device load schedule.

6.4 Outlook

IoT-driven approaches for data centers and buildings are the subjects of active research. It is therefore not surprising that the topics discussed in this thesis are open to be researched further. We highlight a number of specific research directions that the present work opens to further investigation. First of all, more research, and especially data, is required to evaluate the effect of particular best practices on the efficiency of the data center. A cost-benefit analysis of each best practice would greatly assist with the selection and implementation of practices. Though, this would require sensitive data to be made publicly available. Where data center metrics are concerned, there is a clear need for a metric or multiple metrics that enable the fair comparison of data centers. Presently, the PUE metric is misused for this purpose, as PUE was only ever developed to be used for comparisons of a single data center with itself over time. Furthermore, most green metrics focus on reducing and reusing resources while there is a clear lack of metrics focused on recycling and renewables.

A practical evaluation of IoT-based data center monitoring encompassing the entire facility is necessary. The challenge lies in the fact that data centers are critical components of the ICT infrastructure, making facility-wide experiments difficult. This is a challenge we also faced in our own research. One way to counter this problem is through the use of simulators. There are simulators for workload scheduling and virtual machine placement, but there is a lack of simulators that are able to capture the data center in its entirety. Especially where environmental conditions and cooling systems are concerned. Additionally, data center metrics could be tightly integrated with such a simulator. We also foresee research opportunities in generalizing the models we developed for monitoring co-location data centers in order to include other CPU types, and even GPUs.

Where smart building and smart grids are concerned, the inclusion of electric vehicles in the device load scheduling problem would be interesting, as electric vehicles are essentially mobile energy storage systems which can be charged and discharged according to unique scheduling constraints. Another interesting direction is to perform the scheduling in the domain of data centers, where workloads can be scheduled instead of device loads. One can envision a HPC cluster where users specify the constraints of their computational jobs, or where jobs are characterized automatically and constraints are assigned accordingly. The jobs can then be scheduled based on the available resources and price signals in order to minimize a given cost function. Regardless, it seems inevitable that data centers have to become increasingly more sustainable, and that IoT and data-driven approaches will play a central role in this process.

Bibliography

[AAFP12] V. Avelar, D. Azevedo, A. French, E. N. Power. ‘PUE™: A Comprehensive Examination of the Metric’. In: *The Green Grid, White paper #49* (2012) (cit. on pp. 41, 74, 75, 99).

[AAH+18] X. An, M. Arora, W. Huang, W. C. Brantley, J. L. Greathouse. ‘3D Numerical Analysis of Two-Phase Immersion Cooling for Electronic Components’. In: *17th IEEE Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems (ITherm)*. 2018, pp. 609–614 (cit. on pp. 97, 100, 105).

[ACD08] D. Anderson, T. Cader, T. Darby. ‘A Framework for Data Center Energy Productivity’. In: *The Green Grid, White paper #13* (2008) (cit. on p. 81).

[AD08] M. Aiello, S. Dustdar. ‘Are our homes ready for services? A domotic infrastructure based on the Web service stack’. In: *Pervasive and Mobile Computing* 4.4 (2008), pp. 506–525 (cit. on pp. 121, 193).

[AFLV08] M. Al-Fares, A. Loukissas, A. Vahdat. ‘A Scalable, Commodity Data Center Network Architecture’. In: *SIGCOMM Computer Communication Review* 38.4 (2008), pp. 63–74 (cit. on p. 115).

[AGC+19] A. Anand, A. Galletta, A. Celesti, M. Fazio, M. Villari. ‘A secure inter-domain communication for IoT devices’. In: *IEEE International Conference on Cloud Engineering (IC2E)*. 2019, pp. 235–240 (cit. on p. 49).

[AHAS11] S. Al-Haj, E. Al-Shaer. 'Measuring firewall security'. In: *Proceedings of 4th Symposium on Configuration Analytics and Automation (SAFE-CONFIG)*. 2011, pp. 1–4 (cit. on p. 87).

[AMW+10] D. Abts, M. R. Marty, P. M. Wells, P. Klausler, H. Liu. 'Energy proportional datacenter networks'. In: *ACM SIGARCH Computer Architecture News*. Vol. 38. 3. ACM. 2010, pp. 338–347 (cit. on p. 83).

[ANI12] ANIXER. 'The Four Layers of Data Cente Physical Security for a Comprehensive and Integrated Approach'. In: *ANIXER, White Paper* (2012) (cit. on p. 85).

[ANK10] A. Alimian, B. Nordman, D. Kharitonov. 'Network and telecom equipment – energy and performance assessment'. In: *ECR initiative Draft 3.0.1* (2010). URL: `https://ecrinitiative.org/pdfs` (cit. on p. 84).

[AP03] M. Arregoces, M. Portolani. 'Performance Metrics of Data Center Devices'. In: *Data Center Fundamentals*. Cisco Press, 2003, pp. 919–960 (cit. on p. 87).

[AP14] M. Aiello, G. A. Pagani. 'The smart grid's data generating potentials'. In: *2014 Federated Conference on Computer Science and Information Systems*. 2014, pp. 9–16 (cit. on p. 47).

[APP15] APPTIO. 'IT Financial Metrics Primer: Eleven Essential Metrics for Optimizing the Business Value of IT'. In: *APPTIO, White paper* (2015) (cit. on pp. 86, 88).

[ASGM21] A. Amer, K. Shaban, A. Gaouda, A. Massoud. 'Home Energy Management System Embedded with a Multi-Objective Demand Response Optimization Model to Benefit Customers and Operators'. In: *Energies* 14.2 (2021) (cit. on p. 53).

[ASH11] ASHRAE. 'Thermal Guidelines for Data Processing Environments – Expanded Data Center Classes and Usage Guidance'. In: *ASHRAE Technical Committee 9.9* (2011) (cit. on pp. 42, 82).

[ATI14] ATIS. 'Energy Efficiency For Telecommunication Equipment: Methodology For Measurement & Reporting – Transport & Optical Access Requirements'. In: *ANSI Webstore* (2014) (cit. on p. 84).

[AVS+15] A. I. Aravanis, A. Voulkidis, J. Salom, J. Townley, V. Georgiadou, A. Oleksiak, M. R. Porto, F. Roudet, T. Zahariadis. 'Metrics for Assessing Flexibility and Sustainability of Next Generation Data Centers'. In: *IEEE Globecom Workshops*. 2015, pp. 1–6 (cit. on pp. 41, 74, 75, 79).

[AZZ05] M. Aiello, M. Zanoni, A. Zolet. 'Exploring Web-Service Notification: Building a scalable domotic infrastructure'. In: *Dr. Dobb's Journal: Software Tools for the Professional Developer, CMP, 371:48-51* (2005) (cit. on p. 121).

[Abb11] Z. Abbadi. 'Security Metrics – What Can We Measure?' In: *OWASP PDF Archive* (2011) (cit. on p. 85).

[Abb18] T. Abbas. 'Smart Home Scenarios'. In: *Figshare* (2018) (cit. on pp. 126, 130, 142, 143, 150).

[Ahu12] N. Ahuja. 'Datacenter power savings through high ambient datacenter operation: CFD modeling study'. In: *28th Annual IEEE Semiconductor Thermal Measurement and Management Symposium (SEMI-THERM)*. IEEE. 2012, pp. 104–107 (cit. on pp. 60, 63).

[Ali18] Alibaba. *Immersion Cooling for Green Computing*. `https://www.opencompute.org/files/Immersion-Cooling-for-Green-Computing-V1.0.pdf`. 2018 (cit. on pp. 97, 107).

[All07] Alliance to Save Energy. 'PC Energy Report 2007: United States'. 2007 (cit. on pp. 204, 214).

[All09] Alliance to Save Energy. 'PC Energy Report 2009: United States, United Kingdom, Germany'. 2009 (cit. on p. 204).

[Amd07] G. M. Amdahl. 'Computer Architecture and Amdahl's Law'. In: *IEEE Solid-State Circuits Society Newsletter* 12.3 (2007), pp. 4–9 (cit. on p. 251).

[And19] A. Andrae. 'Comparison of Several Simplistic High-Level Approaches for Estimating the Global Energy and Electricity Use of ICT Networks and Data Centers'. In: *International Journal of Green Technology* 5 (2019), pp. 50–63 (cit. on p. 21).

[Ash] ‘Design Considerations for Datacom Equipment Centeres’. In: *Technical report, American Society of Heating Refrigeration and Air Conditioning Engineers* (2005) (cit. on p. 42).

[BBC+12] M. Banks, E. Benjamin, T. Calderwood, R. G. Llera, J. Pflueger. ‘Electronics Disposal Efficiency (Ede): An IT Recycling Metric for Enterprises and Data Centers’. In: *The Green Grid, White Paper #53* (2012) (cit. on pp. 41, 79).

[BBDC+15] R. Basmadjian, P. Bouvry, G. Da Costa, L. Gyarmati, D. Kliazovich, S. Lafond, L. Lefevre, H. De, J.-M. P. Meer, R. Pries, et al. ‘Green Data Centers’. In: *Large-Scale Distributed Systems and Energy Efficiency: A Holistic View* (2015), pp. 159–196 (cit. on p. 78).

[BBG10] R. Buyya, J. Broberg, A. M. Goscinski. *Cloud Computing: Principles and Paradigms*. Vol. 87. John Wiley & Sons, 2010 (cit. on p. 86).

[BC10] C. Bekas, A. Curioni. ‘A new energy aware performance metric’. In: *Computer Science - Research and Development* 25 (2010), pp. 187–195 (cit. on pp. 81, 101).

[BCC20] M. Bansal, I. Chana, S. Clarke. ‘A Survey on IoT Big Data: Current Status, 13 V’s Challenges, and Future Directions’. In: *ACM Computing Surveys* 53.6 (2020) (cit. on p. 23).

[BCH13] L. A. Barroso, J. Clidaras, U. Hölzle. *The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines*. 2013 (cit. on p. 114).

[BDD14] M. Bana, A. Docca, S. Devis. ‘From Compromised to Optimized-10 million Saved in One Data Center’. In: *Future Facilities Ltd., White paper FFL-004* (2014) (cit. on p. 81).

[BJ12] W. L. Bircher, L. K. John. ‘Complete System Power Estimation Using Processor Performance Events’. In: *IEEE Transactions on Computers* 61.4 (2012), pp. 563–577 (cit. on p. 43).

[BLM+11] A. B. Brush, B. Lee, R. Mahajan, S. Agarwal, S. Saroiu, C. Dixon. ‘Home automation in the wild: challenges and opportunities’. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI ’11*. ACM Press, 2011, p. 2115 (cit. on p. 122).

[BM07] C. L. Belady, C. G. Malone. ‘Metrics and an Infrastructure Model to Evaluate Data Center Efficiency’. In: *ASME 2007 International Electronic Packaging Technical Conference and Exhibition*. American Society of Mechanical Engineers. 2007, pp. 751–755 (cit. on p. 74).

[BM08] W. Boyer, M. McQueen. ‘Ideal Based Cyber Security Technical Metrics for Control Systems’. In: *Critical Information Infrastructures Security*. Springer, 2008, pp. 246–260 (cit. on p. 87).

[BMB14] A. H. Buckman, M. Mayfield, S. B. Beck. ‘What is a Smart Building?’ In: *Smart and Sustainable Built Environment* 3.2 (2014), pp. 92–109 (cit. on p. 51).

[BP06] M. Beitelmal, C. Patel. *Model-Based Approach for Optimizing a Data Center Centralized Cooling System*. Tech. rep. HP Labs, 2006 (cit. on p. 61).

[BP08] C. Belady, M Patterson. ‘Green Grid Productivity Indicator’. In: *The Green Grid, White Paper #15* (2008) (cit. on pp. 81, 84, 85).

[BRPC07] C. Belady, A. Rawson, J. Pfleuger, T. Cader. ‘The Green Grid Data Center Power Efficiency Metrics: PUE and DCiE’. In: *The Green Grid, White Paper #06* (2007) (cit. on pp. 41, 89).

[BTA20] R. Bunger, W. Torell, V. Avelar. ‘Capital Cost Analysis is of Immersive Liquid-Cooled vs. Air-Cooled Large Data Centers’. In: *Schneider Electric, White Paper #282* (2020) (cit. on pp. 97, 105).

[Bak17] K. Bakshi. ‘Microservices-based software architecture and approaches’. In: *IEEE Aerospace Conference*. 2017, pp. 1–8 (cit. on p. 229).

[Bar05] L. A. Barroso. ‘The Price of Performance: An Economic Case for Chip Multiprocessing’. In: *ACM Queue* 3.7 (2005), pp. 48–53 (cit. on p. 75).

[Bit] ‘Two-Phase Immersion Cooling A revolution in data center efficiency’. In: 3M™ Novec™ Engineered Fluids. 2015 (cit. on pp. 97, 98).

[Bla10] M. Blackburn. ‘The Green Grid Data Center Compute Efficiency Metric: DCcE’. In: *The Green Grid, White Paper #34* (2010) (cit. on pp. 74, 75).

[Bot10] L. Bottou. 'Large-Scale Machine Learning with Stochastic Gradient Descent'. In: *Proceedings of COMPSTAT'2010*. 2010, pp. 177–186 (cit. on p. 186).

[Bri07] K. G. Brill. 'Data Center Energy Efficiency and Productivity'. In: *The Uptime Institute, White Paper* (2007) (cit. on p. 81).

[CAB+18] T. Cioara, I. Anghel, M. Bertoncini, I. Salomie, D. Arnone, M. Mammina, T.-H. Velivassaki, M. Antal. 'Optimized flexibility management enacting Data Centres participation in Smart Demand Response programs'. In: *Future Generation Computer Systems* 78 (2018), pp. 330–342 (cit. on p. 21).

[CAP+11] S. Chahal, I. K. Anandarao, S. C. Planner, I. C. Peters, I. I. Director, S. Healy, I. N. Wayman, S. Engineer, I. S. Owen. 'Implementing Cloud Storage Metrics to Improve IT Efficiency and Capacity Management'. In: *IT@Intel White Paper* (2011) (cit. on pp. 84, 85).

[CCC12] R. S. Couto, M. E. M. Campista, L. H. M. Costa. 'A reliability analysis of datacenter topologies'. In: *IEEE Global Communications Conference (GLOBECOM)*. 2012, pp. 1890–1895 (cit. on p. 84).

[CCPS15] A. Capozzoli, M. Chinnici, M. Perino, G. Serale. 'Review on Performance Metrics for Energy Efficiency in Data Center: The Role of Thermal Management'. In: *Energy Efficient Data Centers*. 2015, pp. 135–151 (cit. on p. 41).

[CFOGPP06] L. A. Castillo, J. Fernández-Olivares, Ó. García-Pérez, F. Palao. 'Efficiently handling temporal knowledge in an HTN planner'. In: *International Conference on Automated Planning and Scheduling (ICAPS)*. 2006, pp. 63–72 (cit. on p. 201).

[CGB17] S. Chandrasekaran, J. Gess, S. Bhavnani. 'Effect of subcooling, flow rate and surface characteristics on flow boiling performance of high performance liquid cooled immersion server model'. In: *16th IEEE Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems (ITherm)*. IEEE. 2017, pp. 905–912 (cit. on pp. 97, 100).

[CGC16] T. Chen, X. Gao, G. Chen. 'The features, hardware, and architectures of data center networks: A survey'. In: *Journal of Parallel and Distributed Computing* 96 (2016), pp. 45 –74 (cit. on pp. 20, 116).

[CGW12] S. X. Chen, H. B. Gooi, M. Q. Wang. 'Sizing of Energy Storage for Microgrids'. In: *IEEE Transactions on Smart Grid* 3.1 (2012), pp. 142–151 (cit. on pp. 223, 224).

[CH16a] J. Colby, J. Herzing. 'The Why, What, And How Of Data Center Humidification'. In: *Engineered Systems* (2016) (cit. on p. 63).

[CH16b] H. Coles, M. Herrlin. *Immersion Cooling of Electronics in DoD Installations*. Tech. rep. LBNL-1005666. Berkeley National Laboratories, 2016 (cit. on pp. 97, 106, 107).

[CM10] L. Chiaraviglio, M. Mellia. 'PoliSave: Efficient power management of campus PCs'. In: *SoftCOM 2010, 18th International Conference on Software, Telecommunications and Computer Networks*. 2010, pp. 82–87 (cit. on p. 204).

[CPH+11] D. Chen, B. Pernici, E. Henis, R. I. Kat, D. Sotnikov, C. Cappiello, A. M. Ferreira, M. Vitali, T. Jiang, J. Liu. 'Usage Centric Green Performance Indicators'. In: *ACM SIGMETRICS Performance Evaluation Review* 39.3 (2011), pp. 92–96 (cit. on p. 42).

[CSA14] N. Carvalho, A. Simões, J. Almeida. 'DMOSS: Open Source Software Documentation Assessment'. In: *Computer Science and Information Systems*. Vol. 11. 2014, pp. 1197–1207 (cit. on p. 131).

[CSLC14] A. Capozzoli, G. Serale, L. Liuzzo, M. Chinnici. 'Thermal Metrics for Data Centers: A Critical Review'. In: *Energy Procedia* 62 (2014), pp. 391–400 (cit. on p. 42).

[Cal] Calaos. *Calaos – Open Source Home Automation*. URL: `https://calaos.fr/` (cit. on p. 133).

[Col11] D. Cole. 'Data Center Energy Efficiency – Looking Beyond PUE'. In: *No Limits Software, White Paper #4* (2011) (cit. on p. 41).

[Coma] openHAB Community. *Contributing to the Development of openHAB*. URL: `https://openhab.org/docs/developer/contributing.html` (cit. on p. 159).

[Comb] openHAB Community. *Empowering the smart home*. URL: `http://openhab.org/` (cit. on p. 133).

[Comc] openHAB Community. *Next-Generation Rule Engine*. URL: `https://openhab.org/docs/configuration/rules-ng.html` (cit. on p. 150).

[Coo07] A. Cook. 'Technology Carbon Efficiency'. In: *CS Technology, White Paper* (2007) (cit. on p. 79).

[Ctr15] CtrlS Datacenter. 'Data Center Security Strategies'. In: *CtrlS Datacenter, White Paper* (2015) (cit. on p. 86).

[DAMPT10] C. B. Dan Azevedo, J. P. Michael Patterson, R. Tipley. 'Carbon Usage Effectiveness (CUE): A Green Grid Data CenterSustainability Metric'. In: *The Green Grid, White Paper #32* (2010) (cit. on pp. 41, 79).

[DDGS16] P. Derbeko, S. Dolev, E. Gudes, S. Sharma. 'Security and privacy aspects in MapReduce on clouds: A survey'. In: *Computer Science Review* 20 (2016), pp. 1 –28 (cit. on p. 51).

[DEDP15] H. Derhamy, J. Eliasson, J. Delsing, P. Priller. 'A survey of commercial frameworks for the Internet of Things'. In: *IEEE 20th Conference on Emerging Technologies Factory Automation (ETFA)*. 2015, pp. 1–8 (cit. on p. 49).

[DEGG21] A. C. Duman, H. S. Erden, Ömer Gönül, Önder Güler. 'A home energy management system with an integrated smart thermostat for demand response in smart grids'. In: *Sustainable Cities and Society* 65 (2021), p. 102639 (cit. on p. 53).

[DGY+74] R. H. Dennard, F. H. Gaensslen, H.-N. Yu, V. L. Rideout, E. Bassous, A. R. LeBlanc. 'Design of ion-implanted MOSFET's with very small physical dimensions'. In: *IEEE Journal of Solid-State Circuits* 9.5 (1974), pp. 256–268 (cit. on p. 96).

[DJK+09] T. Daim, J. Justice, M. Krampits, M. Letts, G. Subramanian, M. Thirumalai. 'Data center metrics: An energy efficiency model for information technology managers'. In: *Management of Environmental Quality* 20.6 (2009), pp. 712–731 (cit. on pp. 41, 42).

[DKR13] D. Drutskoy, E. Keller, J. Rexford. 'Scalable network virtualization in software-defined networks'. In: *Internet Computing, IEEE* 17.2 (2013), pp. 20–27 (cit. on p. 66).

[DLB19] T. Day, P. Lin, R. Bunger. 'Liquid Cooling Technologies for Data Centers and Edge Applications'. In: *Schneider Electric, White Paper #265* (2019) (cit. on p. 106).

[DLGL+13] V. Degeler, L. I. Lopera Gonzalez, M. Leva, P. Shrubsole, S. Bonomi, O. Amft, A. Lazovik. 'Service-Oriented Architecture for Smart Environments'. In: *IEEE 6th International Conference on Service-Oriented Computing and Applications*. 2013, pp. 99–104 (cit. on p. 193).

[DPOLR+14] A. De Paola, M. Ortolani, G. Lo Re, G. Anastasi, S. K. Das. 'Intelligent management systems for energy efficiency in buildings: A survey'. In: *ACM Comput. Surv.* 47.1 (2014), 13:1–13:38 (cit. on p. 52).

[DR+14] C. Dwork, A. Roth, et al. 'The Algorithmic Foundationsof Differential Privacy'. In: *Foundations and Trends® in Theoretical Computer Science* 9.3–4 (2014), pp. 211–407 (cit. on p. 51).

[DWF16] M. Dayarathna, Y. Wen, R. Fan. 'Data Center Energy Consumption Modeling: A Survey'. In: *IEEE Communications Surveys & Tutorials* 18.1 (2016), pp. 732–794 (cit. on pp. 21, 103).

[Dar13] S. J. Darby. 'Load management at home: advantages and drawbacks of some 'active demand side' options'. In: *Proceedings of the Institution of Mechanical Engineers, Part A: Journal of Power and Energy* 227.1 (2013), pp. 9–17 (cit. on p. 192).

[Dar15] W. Dargie. 'A Stochastic Model for Estimating the Power Consumption of a Processor'. In: *IEEE Transactions on Computers* 64.5 (2015), pp. 1311–1322 (cit. on p. 43).

[Deg14] V. Degeler. 'Dynamic Rule-Based Reasoning in Smart Environments'. English. PhD thesis. University of Groningen, 2014 (cit. on p. 227).

[Dod08] Y. Dodge. 'The Concise Encyclopedia of Statistics'. In: *The Concise Encyclopedia of Statistics*. 2008. Chap. Kendall Rank Correlation Coefficient, pp. 278–281 (cit. on p. 175).

[Doma] Domoticz. *Domoticz – Control at your finger tips*. URL: `https://domoticz.com/` (cit. on p. 133).

[Domb] Domoticz. *Domoticz – MQTT Architecture*. URL: `https://domoticz.com/wiki/MQTT` (cit. on p. 135).

[Dpp] 'New Data Center Energy Efficiency Evaluation Index – DPPE Measurement Guidelines'. In: *Green IT Promotion Council, White paper (Ver 2.05)* (2012) (cit. on pp. 74, 75, 79).

[EFV+14] R. Eiland, J. Fernandes, M. Vallejo, D. Agonafer, V. Mulay. 'Flow Rate and inlet temperature considerations for direct immersion of a single server in mineral oil'. In: *Fourteenth Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems (ITherm)*. IEEE. 2014, pp. 706–714 (cit. on pp. 44, 97, 100).

[EH12] A. ElShafee, K. A. Hamed. 'Design and implementation of a WIFI based home automation system'. In: *World academy of science, engineering and technology* 68 (2012), pp. 2177–2180 (cit. on p. 112).

[EHN94] K. Erol, J. Hendler, D. S. Nau. 'HTN planning: Complexity and expressivity'. In: *AAAI National Conference on Artificial Intelligence*. 1994, pp. 1123–1128 (cit. on p. 196).

[EJF14] K. Ebrahimi, G. F. Jones, A. S. Fleischer. 'A review of data center cooling technology, operating conditions and the corresponding low-grade waste heat recovery opportunities'. In: *Renewable and Sustainable Energy Reviews* 31 (2014), pp. 622–638 (cit. on p. 44).

[Erl07] T. Erl. *SOA principles of service design*. Prentice Hall PTR, 2007 (cit. on p. 196).

[Eva15] T. Evans. 'Humidification Strategies for Data Centers and Network Rooms'. In: *Schneider Electric, White Paper #58 v3* (2015) (cit. on p. 82).

[FA13] N. Farrington, A. Andreyev. 'Facebook's data center network architecture'. In: *Optical Interconnects Conference*. 2013, pp. 49–50 (cit. on p. 114).

[FA19] L. Fiorini, M. Aiello. 'Energy management for user's thermal and power needs: A survey'. In: *Energy Reports* 5 (2019), pp. 1048–1076 (cit. on p. 192).

[FK] FHEM e.V., R. König. *FHEM SmartHome-Server*. URL: `https://wiki.fhem.de/` (cit. on p. 133).

[FKBZ15] C. Fiandrino, D. Kliazovich, P. Bouvry, A. Zomaya. 'Performance and Energy Efficiency Metrics for Communication Systems of Cloud Computing Data Centers'. In: *IEEE Transactions on Cloud Computing* PP.99 (2015) (cit. on pp. 42, 84).

[FNCDR11] T. C. Ferreto, M. A. Netto, R. N. Calheiros, C. A. De Rose. 'Server consolidation with migration control for virtualized data centers'. In: *Future Generation Computer Systems* 27.8 (2011), pp. 1027–1034 (cit. on p. 59).

[FP16] A. M. Ferreira, B. Pernici. 'Managing the complex data center environment: an Integrated Energy-aware Framework'. In: *Computing* 98 (2016), pp. 709–749 (cit. on p. 71).

[FWKT18] Y. Fan, C. Winkel, D. Kulkarni, W. Tian. 'Analytical Design Methodology for Liquid Based Cooling Solution for High TDP CPUs'. In: *17th IEEE Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems (ITherm)*. IEEE. 2018, pp. 582–586 (cit. on p. 96).

[FYP17] F. Fioretto, W. Yeoh, E. Pontelli. 'A Multiagent System Approach to Scheduling Devices in Smart Homes'. In: *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*. AAMAS '17. 2017, 981–989 (cit. on p. 54).

[Fre] U. Freese. *Smarthomatic*. URL: `https://smarthomatic.org/` (cit. on p. 133).

[G-L] G-Labs Open Source Factory. *HomeGenie*. URL: `http://homegenie.it/` (cit. on p. 133).

[GA16] I. Georgievski, M. Aiello. 'Automated Planning for Ubiquitous Computing'. In: *ACM Computing Surveys* 49.4 (2016) (cit. on pp. 52, 213).

[GAB+12] C. Garnier, M. Aggar, M. Banks, J. Dietrich, B. Shatten, M. Stutz, E. Tong-Viet. 'Data Center Life Cycle Assessment Guidelines'. In: *The Green Grid, White Paper #45* (2012) (cit. on p. 19).

[GB18] S. S. Gill, R. Buyya. 'A Taxonomy and Future Directions for Sustainable Cloud Computing: 360 Degree View'. In: *ACM Computing Surveys (CSUR)* 51.5 (2018), pp. 1–33 (cit. on p. 102).

[GBMP13] J. Gubbi, R. Buyya, S. Marusic, M. Palaniswami. ‘Internet of Things (IoT): A vision, architectural elements, and future directions’. In: *Future Generation Computer Systems* 29.7 (2013), pp. 1645 –1660 (cit. on p. 22).

[GBR+14] J. Gess, S. Bhavnani, B. Ramakrishnan, R. W. Johnson, D. Harris, R. Knight, M. Hamilton, C. Ellis. ‘Investigation and characterization of a high performance, small form factor, modular liquid immersion cooled server model’. In: *2014 Semiconductor Thermal Measurement and Management Symposium (SEMI-THERM)*. IEEE. 2014, pp. 8–16 (cit. on pp. 104, 105).

[GDP+12] I. Georgievski, V. Degeler, G. A. Pagani, T. A. Nguyen, A. Lazovik, M. Aiello. ‘Optimizing Energy Costs for Offices Connected to the Smart Grid’. In: *IEEE Transactions on Smart Grid* 3.4 (2012), pp. 2273–2285 (cit. on pp. 54, 218, 219, 225, 247, 250).

[GHMP08] A. Greenberg, J. Hamilton, D. A. Maltz, P. Patel. ‘The Cost of a Cloud: Research Problems in Data Center Networks’. In: *SIGCOMM Computer Communications Review* 39.1 (2008), pp. 68–73 (cit. on pp. 20, 83).

[GKL08] L. D. Gray, A. Kumar, H. H. Li. ‘Workload Characterization of the SPECpower_ssj2008 Benchmark’. In: *SPEC International Performance Evaluation Workshop* (2008), pp. 262–282 (cit. on p. 43).

[GLMP13] Q. Gu, P. Lago, H. Muccini, S. Potenza. ‘A categorization of green practices used by Dutch data centers’. In: *Procedia Computer Science* 19 (2013). The 4th International Conference on Ambient Systems, Networks and Technologies (ANT 2013), the 3rd International Conference on Sustainable Energy Information Technology (SEIT-2013), pp. 770–776 (cit. on p. 63).

[GMRH16] A. Gholian, H. Mohsenian-Rad, Y. Hua. ‘Optimal Industrial Load Control in Smart Grid’. In: *IEEE Transactions on Smart Grid* 7.5 (2016), pp. 2305–2316 (cit. on p. 192).

[GNN+17] I. Georgievski, T. A. Nguyen, F. Nizamic, B. Setz, A. Lazovik, M. Aiello. ‘Planning meets activity recognition: Service coordination for intelligent buildings’. In: *Pervasive and Mobile Computing* 38.1 (2017), pp. 110–139 (cit. on p. 194).

[GNT04] M. Ghallab, D. S. Nau, P. Traverso. *Automated planning: Theory & practice*. Morgan Kaufmann Publishers Inc., 2004 (cit. on p. 195).

[GRD13] P. Gilbert, K. Ramakrishnan, R. Diersen. 'From Data Center Metrics to Data Center Analytics: How to Unlock the Full Business Value of DCIM'. In: *CA Technologies, White Paper* (2013) (cit. on pp. 61, 67).

[GS] V. Goel, S. Sharma. *üAutomate – Intelligent Home Automation System*. URL: `https://uautomate.herokuapp.com` (cit. on p. 133).

[GSS15] C. Gough, I. Steiner, W. Saunders. *Energy Efficient Servers: Blueprints for Data Center Optimization*. Apress, Berkeley, CA, 2015. Chap. Why Data Center Efficiency Matters, pp. 1–20 (cit. on p. 20).

[GW03] C. Gerald, P. Wheatley. *Applied Numerical Analysis*. 7th Edition. Pearson Education Inc., 2003 (cit. on p. 221).

[Gen14] H. Geng. *Data Center Handbook*. John Wiley & Sons, 2014 (cit. on pp. 36, 39, 186).

[Geo15] I. Georgievski. 'Coordinating services embedded everywhere via hierarchical planning'. PhD thesis. University of Groningen, 2015 (cit. on p. 194).

[Gha19] A. Ghasempour. 'Internet of Things in Smart Grid: Architecture, Applications, Services, Key Technologies, and Challenges'. In: *Inventions* 4.1 (2019) (cit. on p. 53).

[Gia] T. Giachi. *Neon.HomeControl*. URL: `https://neon-home-control.readthedocs.io/` (cit. on p. 133).

[Giu11] L. Giuntini. 'Modeling UPS efficiency as a function of load'. In: *2011 International Conference on Power Engineering, Energy and Electrical Drives*. IEEE. 2011, pp. 1–6 (cit. on p. 81).

[Glo14] Global Metrics Harmonization Task Force. 'Harmonizing Global Metrics for Data Center Energy Efficiency'. In: *The Green Grid, White Paper* (2014) (cit. on p. 84).

[Gre05] D. GreenHill. 'SWaP Space Watts and Performance'. In: *US EPA Energystar Power Efficiency Forum* (2005) (cit. on pp. 75, 81).

[Gre07] Green Grid Industry Consortium. 'Green Grid Metrics Describing Data Center Power Efficiency'. In: *Green Grid Technical Committee White Paper* (2007) (cit. on p. 74).

[HGA+18] N. Hassan, S. Gillani, E. Ahmed, I. Yaqoob, M. Imran. ‘The Role of Edge Computing in Internet of Things’. In: *IEEE Communications Magazine* 56.11 (2018), pp. 110–115 (cit. on p. 47).

[HH19] P. Hüwe, S. Hüwe. *IoT at Home*. Carl Hanser Verlag GmbH & Co. KG, 2019 (cit. on pp. 134, 137).

[HHKM17] F. Heimgaertner, S. Hettich, O. Kohlbacher, M. Menth. ‘Scaling home automation to public buildings: A distributed multiuser setup for OpenHAB 2’. In: *2017 Global Internet of Things Summit (GIoTS)*. 2017, pp. 1–6 (cit. on pp. 136, 137).

[HK07] Herrlin, M. K. ‘Improved data center energy efficiency and thermal performance by advanced airflow analysis’. In: *Digital Power Forum*. 2007, pp. 10–12 (cit. on p. 82).

[HLW+18] Y. Huang, Y. Lu, F. Wang, X. Fan, J. Liu, V. C. M. Leung. ‘An Edge Computing Framework for Real-Time Monitoring in Smart Grid’. In: *IEEE International Conference on Industrial Internet (ICII)*. 2018, pp. 99–108 (cit. on pp. 46, 113).

[HMA+15] M. Hubbell, A. Moran, W. Arcand, D. Bestor, B. Bergeron, C. Byun, V. Gadepally, P. Michaleas, J. Mullen, A. Prout, A. Reuther, et al. ‘Big Data strategies for Data Center Infrastructure management using a 3D gaming platform’. In: *IEEE High Performance Extreme Computing Conference (HPEC)*. 2015, pp. 1–6 (cit. on p. 47).

[HMP+09] J. Haas, M. Monroe, J. Pflueger, J. Pouchet, P. Snelling, A. Rawson, F. Rawson. ‘Proxy Proposals for Measuring Data Center Productivity’. In: *The Green Grid, White Paper #17* (2009) (cit. on p. 75).

[HTGM20] K. Haghshenas, S. Taheri, M. Goudarzi, S. Mohammadi. ‘Infrastructure Aware Heterogeneous-Workloads Scheduling for Data Center Energy Cost Minimization’. In: *IEEE Transactions on Cloud Computing* 10.02 (2020), pp. 972–983 (cit. on p. 42).

[Han08] S. Hanson. ‘"Free" Cooling Using Water Economizers’. In: *TRANE – engineers newsletter* 37.3 (2008) (cit. on p. 78).

[Har99] T. Hartman. ‘"LOOP" Chiller Plant Design Dramatically Lowers Chilled Water Costs’. In: *Renewable and Advanced Energy Systems for the 21st Century*. 1999 (cit. on p. 63).

[Her05] M. Herrlin. 'Rack Cooling Effectiveness in Data Centers and Telecom Central Offices:The Rack Cooling Index (RCI)'. In: *ASHRAE Transactions*. Vol. 111. 2. 2005 (cit. on p. 82).

[Homa] Home Assistant Inc. *Home Assistant – Architecture*. URL: `https://developers.home-assistant.io/docs/architecture/core` (cit. on p. 134).

[Homb] Home Assistant Inc. *Home Assistant – Awaken your home*. URL: `https://home-assistant.io/` (cit. on p. 133).

[Homc] Home Assistant Inc. *Home Assistant – Quality Scale*. URL: `https://www.home-assistant.io/docs/quality_scale/` (cit. on p. 159).

[IBM12] IBM Labs. *Data Center Operational Efficiency Best Practices*. Tech. rep. IBM Global Technology Services, 2012 (cit. on p. 62).

[JGJ+14] O. Jarczyk, B. Gruszka, S. Jaroszewicz, L. Bukowski, A. Wierzbicki. 'GitHub Projects. Quality Analysis of Open-Source Software'. In: *International Conference on Social Informatics* (2014), pp. 80–94 (cit. on pp. 129, 147).

[JGK16] P. Jain, M. Gyanchandani, N. Khare. 'Big data privacy: a technological perspective and review'. In: *Journal of Big Data* 3.1 (2016), p. 25 (cit. on p. 50).

[JK17] M. Jerabandi, M. M. Kodabagi. 'A review on home automation system'. In: *2017 International Conference On Smart Technologies For Smart Nation (SmartTechCon)*. 2017, pp. 1411–1415 (cit. on p. 49).

[JMG+21] M. Jalili, I. Manousakis, Í. Goiri, P. A. Misra, A. Raniwala, H. Alissa, B. Ramakrishnan, P. Tuma, C. Belady, M. Fontoura, et al. 'Cost-Efficient Overclocking in Immersion-Cooled Datacenters'. In: *ACM/IEEE 48th Annual International Symposium on Computer Architecture (ISCA)*. 2021, pp. 623–636 (cit. on pp. 44, 97, 106).

[JS16] J. R. L. Jeff Sauro. *Quantifying the User Experience: Practical Statistics for User Research*. 2nd Edition. Morgan Kaufmann, 2016 (cit. on p. 130).

[JWF+17] A. Justice, T. Winkler, M. Feitosa, M. Graff, V. Fisher, K. Young, L. Cupples, et al. 'Genome-wide meta-analysis of 241,258 adults accounting for smoking behaviour identifies novel loci for obesity traits'. In: *Nature Communications* 8 (2017), p. 14977 (cit. on p. 171).

[Jon14] P. Jones. *DCDi Industry Census 2015*. Tech. rep. Data Center Dynamics, 2014 (cit. on pp. 20, 55).

[Jon18] N. Jones. 'The Information Factories'. In: *Nature* 561.7722 (2018), pp. 163–6 (cit. on p. 100).

[KBSW10] J. Koomey, S. Berard, M. Sanchez, H. Wong. 'Implications of Historical Trends in the Electrical Efficiency of Computing'. In: *IEEE Annals of the History of Computing* 33.3 (2010), pp. 46–54 (cit. on p. 42).

[KFK08] J. M. Kaplan, W. Forrest, N. Kindler. *Revolutionizing Data Center Energy Efficiency*. Tech. rep. McKinsey & Company, 2008 (cit. on p. 74).

[KG15] R. Kotecha, S. Garg. 'Data streams and privacy: Two emerging issues in data classification'. In: *2015 5th Nirma University International Conference on Engineering (NUiCONE)*. 2015, pp. 1–6 (cit. on p. 51).

[KG16] A. C. Kheirabadi, D. Groulx. 'Cooling of server electronics: A design review of existing technology'. In: *Applied Thermal Engineering* 105 (2016), pp. 622–638 (cit. on pp. 45, 96, 97, 104, 105).

[KK15] M. U. S. Khan, S. U. Khan. 'Handbook on Data Centers'. In: Springer New York, 2015. Chap. Smart Data Center, pp. 247–262 (cit. on p. 77).

[KLA16] E. Kaldeli, A. Lazovik, M. Aiello. 'Domain-independent planning for services in uncertain and dynamic environments'. In: *Artificial Intelligence* 236.7 (2016), pp. 30–64 (cit. on p. 197).

[KMM18] M. Kalske, N. Mäkitalo, T. Mikkonen. 'Challenges When Moving from Monolith to Microservice Architecture'. In: *Current Trends in Web Engineering*. Cham, 2018, pp. 32–47 (cit. on p. 229).

[KPB+19] I. Kuncoro, N. Pambudi, M. Biddinika, I Widiastuti, M Hijriawan, K. Wibowo. 'Immersion cooling as the next technology for data center cooling: A review'. In: *Journal of Physics: Conference Series*. Vol. 1402. 4. 2019, p. 044057 (cit. on p. 44).

[KT17] J. Koomey, J. Taylor. 'Zombie / Comatose Servers Redux'. In: *Koomey Analytics and Anthesis* (2017) (cit. on p. 60).

[KW21] M. Koot, F. Wijnhoven. 'Usage impact on data center electricity needs: A system dynamic forecasting model'. In: *Applied Energy* 291 (2021), p. 116798 (cit. on p. 20).

[KWF+20] B. B. Kanbur, C. Wu, S. Fan, W. Tong, F. Duan. 'Two-phase liquid-immersion data center cooling system: Experimental performance and thermoeconomic analysis'. In: *International Journal of Refrigeration* 118 (2020), pp. 290–301 (cit. on p. 45).

[KWLA13] E. Kaldeli, E. U. Warriach, A. Lazovik, M. Aiello. 'Coordinating the Web of services for a smart home'. In: *ACM Tran. on the Web* 7.2 (2013) (cit. on pp. 52, 195, 196).

[Kee] K. R. Keegan. *MisterHouse – It Knows Kung-Fu, an open source home automation program*. URL: `http://misterhouse.sourceforge.net/` (cit. on p. 133).

[Kle] H. Klein. *Ago Control – a framework for device control*. URL: `https://www.mysensors.org/controller/agocontrol` (cit. on p. 133).

[Koh+95] R. Kohavi et al. 'A study of cross-validation and bootstrap for accuracy estimation and model selection'. In: *Ijcai*. Vol. 14. 2. 1995, pp. 1137–1145 (cit. on p. 181).

[Kra] C. Kratky. *Wirehome.Core – an open source Home Automation system for .NET Core*. URL: `https://github.com/chkr1011/Wirehome.Core/` (cit. on p. 133).

[LC13] K.-P. Lee, H.-L. Chen. 'Analysis of energy saving potential of air-side free cooling for data centers in worldwide climate zones'. In: *Energy and Buildings* 64 (2013), pp. 103–112 (cit. on p. 78).

[LH17] M. Levy, J. O. Hallstrom. 'A new approach to data center infrastructure monitoring and management (DCIMM)'. In: *IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC)*. 2017, pp. 1–6 (cit. on p. 48).

[LHF14] B. Lajevardi, K. R. Haapala, J. F.Junker. 'An Energy Efficiency Metric for Data Center Assessment'. In: *2014 Industrial and Systems Engineering Research Conference*. 2014 (cit. on p. 75).

[LHY19] R. Lu, S. H. Hong, M. Yu. 'Demand Response for Home Energy Management Using Reinforcement Learning and Artificial Neural Network'. In: *IEEE Transactions on Smart Grid* 10.6 (2019), pp. 6629–6639 (cit. on p. 54).

[LK15] Z. Li, S. G. Kandlikar. 'Current status and future trends in data-center cooling technologies'. In: *Heat Transfer Engineering* 36.6 (2015), pp. 523–538 (cit. on p. 44).

[LL15] I. Lee, K. Lee. 'The Internet of Things (IoT): Applications, investments, and challenges for enterprises'. In: *Business Horizons* 58.4 (2015), pp. 431–440 (cit. on pp. 112, 191).

[LLV07] N. Li, T. Li, S. Venkatasubramanian. 't-Closeness: Privacy Beyond k-Anonymity and l-Diversity'. In: *IEEE 23rd International Conference on Data Engineering*. 2007, pp. 106–115 (cit. on p. 50).

[LMG18] G. Lykou, D. Mentzelioti, D. Gritzalis. 'A new methodology toward effectively assessing data center sustainability'. In: *Computers & Security* 76 (2018), pp. 327–340 (cit. on p. 21).

[LWH20] H. Li, Z. Wan, H. He. 'Real-Time Residential Demand Response'. In: *IEEE Transactions on Smart Grid* 11.5 (2020), pp. 4144–4154 (cit. on p. 54).

[Lik32] R. Likert. 'A technique for the measurement of attitudes'. In: *Archives of Psychology* 22.140 (1932), pp. 1–55 (cit. on p. 213).

[Lyo12] C. Lyons. 'Enterprise IT Security Architecture Security Zones: Network Security Zone Standards'. In: *Security Enhancement Project (SEP)* (2012) (cit. on p. 86).

[MAG+20] A. Medina-Santiago, A. D. P. Azucena, J. M. Gómez-Zea, J. A. Jesús-Magaña, M. de la Luz Valdez-Ramos, E. Sosa-Silva, F. Falcón-Pérez. 'Adaptive Model IoT for Monitoring in Data Centers'. In: *IEEE Access* 8 (2020), pp. 5622–5634 (cit. on p. 48).

[MDBB13] I. Munteanu, V. Debusschere, S. Bergeon, S. Bacha. 'Efficiency metrics for qualification of datacenters in terms of useful workload'. In: *IEEE Grenoble Conference*. IEEE. 2013, pp. 1–6 (cit. on p. 41).

[MG12] S. Murugesan, G. R. Gangadharan. *Harnessing Green IT: Principles and Practices*. Wiley - IEEE Press, 2012 (cit. on p. 78).

[MGGS09] P. Mathew, S. Ganguly, S. Greenberg, D. Sartor. 'Self-benchmarking Guide for Data Centers: Metrics, Benchmarks, Actions'. In: *Lawrence Berkeley National Laboratory, Technical Report* (2009) (cit. on pp. 78, 82).

[MGKV06] A. Machanavajjhala, J. Gehrke, D. Kifer, M. Venkitasubramaniam. 'L-diversity: privacy beyond k-anonymity'. In: *22nd International Conference on Data Engineering (ICDE'06)*. 2006, pp. 24–24 (cit. on p. 50).

[MH] B. Morgan, B. T. Hill. *The Thing System – Take control of things*. URL: `http://thethingsystem.com/` (cit. on p. 133).

[MIT18] MIT Technology Review. 'How to Unlock the Power of IoT with Real-Time Monitoring'. In: *MIT Technology Review, White Paper* (2018) (cit. on p. 112).

[MK91] E. L. Miller, R. H. Katz. 'Input/output behavior of supercomputing applications'. In: *1991 ACM/IEEE conference on Supercomputing*. ACM. 1991, pp. 567–576 (cit. on p. 85).

[MMK17] M. Matsuoka, K. Matsuda, H. Kubo. 'Liquid immersion cooling technology with natural convection in data center'. In: *IEEE 6th international conference on Cloud networking (CloudNet)*. IEEE. 2017, pp. 1–7 (cit. on pp. 97, 99, 100, 104, 105).

[MSA+03] G. Magklis, G. Semeraro, D. H. Albonesi, S. G. Dropsho, S. Dwarkadas, M. L. Scott. 'Dynamic frequency and voltage scaling for a multiple-clock-domain microprocessor'. In: *IEEE Micro* 23.6 (2003), pp. 62–68 (cit. on p. 59).

[MSFM16] J. Moreno, M. A. Serrano, E. Fernández-Medina. 'Main Issues in Big Data Security'. In: *Future Internet* 8.3 (2016) (cit. on p. 169).

[MSMR21] M. Manganelli, A. Soldati, L. Martirano, S. Ramakrishna. 'Strategies for Improving the Sustainability of Data Centers via Energy Mix, Energy Conservation, and Circular Energy'. In: *Sustainability* 13.11 (2021) (cit. on p. 40).

[Mac67] J. MacQueen. ‘Some methods for classification and analysis of multivariate observations’. In: *Fifth Berkeley Symposium on Mathematical Statistics and Probability*. Vol. 1. 14. 1967, pp. 281–297 (cit. on p. 237).

[McN13] A. McNevin. ‘APAC data center survey reveals high PUE figures across the region’. In: *Data Center Dynamics* (2013) (cit. on pp. 99, 100).

[Mil13] M. P. Mills. ‘The Cloud Begins With Coal- An overview of the electricity used by the global digital ecosystem’. In: *Tecnical report, Digital Power Group* (2013) (cit. on p. 20).

[Mil14] R. Miller. ‘Inside SuperNAP 8: Switch’s Tier IV Data Fortress’. In: *DataCenter Knowledge* (2014) (cit. on pp. 99, 100, 104).

[Mor21] Mordor Intelligence. *Data Center Services Market – Growth, Trends, Covid-19 Impact, and Forecasts (2022-2027)*. Tech. rep. Mordor Intelligence, 2021 (cit. on p. 20).

[Mur08] S. Murugesan. ‘Harnessing Green IT: Principles and Practices’. In: *IEEE IT Professional* 10.1 (2008), pp. 24–33 (cit. on p. 63).

[Mur10] S. Murugesan. ‘Making IT Green’. In: *IEEE IT Professional* (2010), pp. 4–5 (cit. on p. 78).

[NA13] T. A. Nguyen, M. Aiello. ‘Energy intelligent buildings based on user activity: A survey’. In: *Energy and Buildings* 56 (2013), pp. 244–257 (cit. on p. 51).

[NB17] J. Ni, X. Bai. ‘A review of air conditioning energy performance in data centers’. In: *Renewable and Sustainable Energy Reviews* 67 (2017), pp. 625–640 (cit. on p. 96).

[NC17] Y. Ngoko, C. Cérin. ‘Reducing the Number of Comatose Servers: Automatic Tuning as an Opportunistic Cloud-Service’. In: *IEEE International Conference on Services Computing (SCC)*. 2017, pp. 487–490 (cit. on pp. 60, 111).

[NCT+] E. Nicoletti, M. Cicolella, G. Pulido de Torres, A. Mengoli, M. Mazzoni. *Freedomotic – an open source, flexible and secure Internet of Things development framework*. URL: https://freedomotic-platform.com/ (cit. on p. 133).

[NEKZ16] J. Neudorfer, M. Ellsworth, D. Kulkarni, H. Zien. ‘Liquid Cooling Technology Update’. In: *The Green Grid, White Paper #70* (2016) (cit. on p. 106).

[NHE+03] M. Norota, H. Hayama, M. Enai, T. Mori, M. Kishita. ‘Research on efficiency of air conditioning system for data-center’. In: *The 25th International Telecommunications Energy Conference, 2003. INTELEC ’03.* 2003, pp. 147–151 (cit. on p. 82).

[NLL18] C. Nadjahi, H. Louahlia, S. Lemasson. ‘A review of thermal management and innovative cooling strategies for data center’. In: *Sustainable Computing: Informatics and Systems* 19 (2018), pp. 14–28 (cit. on p. 38).

[NLLS12] L. Newcombe, Z. Limbuwala, P. Latham, V. Smith. ‘Data Centre Fixed to Variable Energy Ratio metric DC-FVER’. In: *BCS – The Chartered Institute for IT* (2012) (cit. on p. 74).

[NNLA14] F. Nizamic, T. A. Nguyen, A. Lazovik, M. Aiello. ‘GreenMind - An Architecture and Realization for Energy Smart Buildings’. In: *2014 conference ICT for Sustainability*. 2014, pp. 20–29 (cit. on p. 22).

[NO10] J. Neudorfer, F. J. Ohlhorst. ‘Data Center Efficiency Metrics and Methods’. In: *TechTarget Inc.* (2010) (cit. on p. 81).

[NRA14] T. A. Nguyen, A. Raspitzu, M. Aiello. ‘Ontology-based office activity recognition with applications for energy savings’. In: *Journal of Ambient Intelligence and Humanized Computing* 5.5 (2014), pp. 667–681 (cit. on pp. 196, 208).

[NSCT18] Y. Ngoko, N. Saintherant, C. Cerin, D. Trystram. ‘How Future Buildings Could Redefine Distributed Computing’. In: *IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*. IEEE. 2018, pp. 1232–1240 (cit. on p. 98).

[NZL18] S. Nan, M. Zhou, G. Li. ‘Optimal residential community demand response scheduling in smart grid’. In: *Applied Energy* 210 (2018), pp. 1280–1289 (cit. on p. 53).

[New99] D. Newman. ‘Benchmarking Terminology for Firewall Performance’. In: *Network Working Group, The Internet Society* (1999) (cit. on p. 87).

[Ng12] A. Ng. 'Clustering with the k-means algorithm'. In: *Machine Learning* (2012) (cit. on p. 238).

[OKC+10] D. Oh, N. S. Kim, C. C. P. Chen, A. Davoodi, Y. H. Hu. 'Runtime temperature-based power estimation for optimizing throughput of thermal-constrained multi-core processors'. In: *2010 15th Asia and South Pacific Design Automation Conference (ASP-DAC)*. 2010, pp. 593–599 (cit. on p. 43).

[Off] A. Off. *Smart Haus – How to build the most robust and secure home automation system*. URL: `https://www.freecodecamp.org/news/the-most-robust-and-secure-home-automation-system-6d0ddbb39f29/` (cit. on p. 133).

[Oma18] O. Omar. 'Intelligent building, definitions, factors and evaluation criteria of selection'. In: *Alexandria Engineering Journal* 57.4 (2018), pp. 2903–2910 (cit. on p. 193).

[Ope] OpenMotics BV. *OpenMotics – an open, modular, wired and affordable software and hardware platform for automating a home, building or district*. URL: `https://openmotics.com/en/` (cit. on p. 133).

[PA15] G. A. Pagani, M. Aiello. 'Generating Realistic Dynamic Prices and Services for the Smart Grid'. In: *IEEE Systems Journal* 9.1 (2015), pp. 191–198 (cit. on p. 192).

[PABP11] M. Patterson, D Azevedo, C Belady, J Pouchet. 'Water Usage Effectiveness (WUE): a Green Grid Data Center Sustainability Metric'. In: *The Green Grid, White Paper #35* (2011) (cit. on pp. 41, 79).

[PC21] R. Panchalingam, K. C. Chan. 'A state-of-the-art review on artificial intelligence for Smart Buildings'. In: *Intelligent Buildings International* 13.4 (2021), pp. 203–226 (cit. on p. 52).

[PDS16] M. Papamichail, T. Diamantopoulos, A. Symeonidis. 'User-Perceived Source Code Quality Estimation Based on Static Analysis Metrics'. In: *IEEE International Conference on Software Quality, Reliability and Security (QRS)*. 2016, pp. 100–107 (cit. on p. 125).

[PEBC15] N. G. Paterakis, O. Erdinç, A. G. Bakirtzis, J. P. S. Catalão. 'Optimal Household Appliances Scheduling Under Day-Ahead Pricing and Load-Shaping Demand Response Strategies'. In: *IEEE Transactions on Industrial Informatics* 11.6 (2015), pp. 1509–1519 (cit. on p. 53).

[PP09] E. Pakbaznia, M. Pedram. 'Minimizing data center cooling and server power costs'. In: *ACM/IEEE international symposium on Low power electronics and design*. ACM. 2009, pp. 145–150 (cit. on p. 78).

[PPH+13] M. K. Patterson, S. W. Poole, C.-H. Hsu, D. Maxwell, W. Tschudi, H. Coles, D. J. Martinez, N. Bates. 'TUE, a New Energy-Efficiency Metric Applied at ORNL's Jaguar'. In: *2013 International Supercomputing Conference (ISC)*. 2013, pp. 372–382 (cit. on p. 75).

[PTV+10] M. Patterson, B. Tschudi, O. Vangeet, J Cooley, D Azevedo. 'ERE: A Metric for Measuring the Benefit of Reuse Energy from a Data Center'. In: *The Green Grid, White Paper #29* (2010) (cit. on p. 79).

[PVC14] M. R. Porto, S. S. Vives, A. F. Caro. 'Cluster Activities Task 3'. In: *EU SmartCities Cluster* (2014) (cit. on p. 41).

[Pat10] S. V. Patankar. 'Airflow and Cooling in a Data Center'. In: *Journal of Heat Transfer* 132.7 (2010) (cit. on p. 44).

[Pow11] E. N. Power. 'Recycling Ratios: The Next Step for Data Center Sustainability'. In: *Emerson Network Power, White Paper* (2011) (cit. on p. 79).

[QLL13] X. Qian, Z. Li, Z. Li. 'A thermal environmental analysis method for data centers'. In: *International Journal of Heat and Mass Transfer* 62 (2013), pp. 579–585 (cit. on p. 82).

[RG75] L. R. Rabiner, B. Gold. *Theory and Application of Digital Signal Processing*. 1st Edition. Prentice Hall, 1975 (cit. on p. 178).

[RM07] V. Rodoplu, T. Meng. 'Bits-per-Joule Capacity of Energy-Limited Wireless Networks'. In: *IEEE Transactions on Wireless Communications* 6.3 (2007), pp. 857–865 (cit. on p. 84).

[RN10] S. Russell, P. Norvig. *Artificial Intelligence: A Modern Approach*. 3rd Edition. Prentice Hall, 2010 (cit. on p. 227).

[RPE11] F. Ryckbosch, S. Polfliet, L. Eeckhout. 'Trends in Server Energy Proportionality'. In: *Computer* 44 (2011), pp. 69–72 (cit. on p. 81).

[RR07] L. Richardson, S. Ruby. *RESTful Web Services*. 1st Edition. O'Reilly Media, Inc., 2007 (cit. on p. 202).

[RRC+19] S. Ramdas, P. Rajmane, T. Chauhan, A. Misrak, D. Agonafer. 'Impact of Immersion Cooling on Thermo-Mechanical Properties of PCB's and Reliability of Electronic Packages'. In: *ASME 2019 International Technical Conference and Exhibition on Packaging and Integration of Electronic and Photonic Microsystems*. American Society of Mechanical Engineers. 2019 (cit. on p. 97).

[RSR+17] V. D. Reddy, B. Setz, G. S. V. Rao, G. Gangadharan, M. Aiello. 'Metrics for Sustainable Data Centers'. In: *IEEE Transactions on Sustainable Computing* 2.3 (2017), pp. 290–303 (cit. on p. 73).

[RT17] B. L. Risteska Stojkoska, K. V. Trivodaliev. 'A review of Internet of Things for smart home: Challenges and solutions'. In: *Journal of Cleaner Production* 140 (2017), pp. 1454 –1464 (cit. on p. 50).

[Ras05] N. Rasmussen. 'Guidelines for Specification of Data Center Power Density'. In: *Schneider Electric, White Paper #120* (2005) (cit. on p. 74).

[Ras06] N. Rasmussen. 'Understanding Power Factor, Crest Factor, and Surge Factor'. In: *Schneider Electric, White Paper #1* (2006) (cit. on p. 81).

[Ras11] N. Rasmussen. 'Determining Total Cost of Ownership for Data Center and Network Room Infrastructure'. In: *Schneider Electric, White Paper #6* (2011) (cit. on p. 88).

[SADK19] Y. Sun, N. B. Agostini, S. Dong, D. Kaeli. 'Summarizing CPU and GPU design trends with product data'. In: *arXiv preprint* abs/1911.11313 (2019) (cit. on p. 96).

[SBK07] J. R. Stanley, K. Brill, J Koomey. 'Four Metrics Define Data Center "Greenness"'. In: *Uptime Institute, White Paper* (2007) (cit. on pp. 74, 75).

[SBP02] R. K. Sharma, C. E. Bash, C. D. Patel. 'Dimensionless Parameters for Evaluation of Thermal Design and Performance of Large-scale Data Centers'. In: *8th AIAA/ASME Joint Thermophysics and Heat Transfer Conference*. 2002 (cit. on p. 82).

[SBS+12] B. Schaeppi, T. Bogner, A. Schloesser, L. Stobbe, M. D. De Asuncao. 'Metrics for energy efficiency assessment in data centers and server rooms'. In: *2012 Electronics Goes Green 2012+*. 2012, pp. 1–6 (cit. on p. 42).

[SBS+19] P. A. Shinde, P. V. Bansode, S. Saini, R. Kasukurthy, T. Chauhan, J. M. Shah, D. Agonafer. 'Experimental Analysis for Optimization of Thermal Performance of a Server in Single Phase Immersion Cooling'. In: *ASME 2019 International Technical Conference and Exhibition on Packaging and Integration of Electronic and Photonic Microsystems*. American Society of Mechanical Engineers. 2019 (cit. on p. 44).

[SDCB17] J. Son, A. V. Dastjerdi, R. N. Calheiros, R. Buyya. 'SLA-Aware and Energy-Efficient Dynamic Overbooking in SDN-Based Cloud Data Centers'. In: *IEEE Transactions on Sustainable Computing* 2.2 (2017), pp. 76–89 (cit. on p. 42).

[SESA16] J. M. Shah, R. Eiland, A. Siddarth, D. Agonafer. 'Effects of mineral oil immersion cooling on IT equipment reliability and reliability enhancements to data center operations'. In: *15th IEEE Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems (ITherm)*. IEEE. 2016, pp. 316–325 (cit. on p. 46).

[SF12] B. Subramaniam, W.-c. Feng. 'The Green Index: A Metric for Evaluating System-Wide Energy Efficiency in HPC Systems'. In: *IEEE 26th International Parallel and Distributed Processing Symposium Workshops & PhD Forum*. 2012, pp. 1007–1013 (cit. on p. 79).

[SG16] V. K. Singh, J. Guo. 'Improving Service Levels Using Internet of Things Infrastructure in Data Centers'. In: *IEEE International Conference on Smart Computing (SMARTCOMP)*. 2016, pp. 1–3 (cit. on p. 169).

[SKS+20] K. Shafique, B. A. Khawaja, F. Sabir, S. Qazi, M. Mustaqim. 'Internet of Things (IoT) for Next-Generation Smart Systems: A Review of Current Challenges, Future Trends and Prospects for Emerging 5G-IoT Scenarios'. In: *IEEE Access* 8 (2020), pp. 23022–23040 (cit. on p. 164).

[SMR+12] L. H. Sego, A. Márquez, A. Rawson, T. Cader, K. Fox, W. I. Gustafson Jr, C. J. Mundy. 'Implementing the Data Center Energy Productivity Metric'. In: *ACM Journal on Emerging Technologies in Computing Systems* 8.4 (2012), p. 30 (cit. on p. 74).

[SN16] A. Sanatinia, G. Noubir. 'On GitHub's Programming Languages'. In: *arXiv preprint* abs/1603.00431 (2016). arXiv: `1603.00431` (cit. on p. 125).

[SNLA16] B. Setz, F. Nizamic, A. Lazovik, M. Aiello. 'Power Management of Personal Computers based on User Behaviour'. In: *International Conference on Smart Cities and Green ICT Systems*. 2016, pp. 409–416 (cit. on pp. 199, 204, 215).

[SRCL+22] N. A. Sulieman, L. Ricciardi Celsi, W. Li, A. Zomaya, M. Villari. 'Edge-Oriented Computing: A Survey on Research and Use Cases'. In: *Energies* 15.2 (2022) (cit. on p. 47).

[SSB+09] R. Sharma, A. Shah, C. Bash, T. Christian, C. Patel. 'Water efficiency management in datacenters: metrics and methodology'. In: *IEEE International Symposium on Sustainable Systems and Technology*. IEEE. 2009, pp. 1–6 (cit. on p. 79).

[SSJ+13] L. Sisó, J. Salom, M. Jarus, A. Oleksiak, T. Zilio. 'Energy and Heat-Aware Metrics for Data Centers: Metrics Analysis in the Framework of CoolEmAll Project'. In: *2013 International Conference on Cloud and Green Computing*. 2013, pp. 428–434 (cit. on p. 41).

[SSO+14] L. Siso, J. Salom, E. Oro, G. D. Costa, T. Zilio. 'CoolEmAll D5.6 Final metrics and benchmarks'. In: *IRIT – Institut de Recherche en Informatique de Toulouse* (2014) (cit. on pp. 41, 74, 75, 78, 82).

[SSS+16] A. Shehabi, S. Smith, D. Sartor, R. Brown, M. Herrlin, J. Koomey, E. Masanet, N. Horner, I. Azevedo, W. Lintner. 'United States Data Center Energy Usage Report'. In: *LBNL-1005775* (2016) (cit. on p. 103).

[SZZ14] L. Smirek, G. Zimmermann, D. Ziegler. 'Towards Universally Usable Smart Homes – How Can MyUI, URC and openHAB Contribute to an Adaptive User Interface Platform?' In: *International Conference on Advances in Human-oriented and Personalized Mechanisms, Tech-*

nologies, and Services (CENTRIC 2014) (2014), pp. 29–38 (cit. on p. 49).

[Sca15] S. D. Scalet. *19 ways to build physical security into your data center – Mantraps, access control systems, bollards and surveillance*. 2015 (cit. on p. 86).

[Sch09] G. Schulz. *The Green and Virtual Data Center*. 1st Edition. Auerbach Publications, 2009 (cit. on pp. 85, 88).

[Sco17] A. D. Scott. *Collaborative Web Development*. OReilly Media, Inc., 2017 (cit. on p. 131).

[Sha18] J. M. Shah. 'Characterizing contamination to expand ASHRAE envelope in airside economization and thermal and reliability in immersion cooling of data centers'. PhD thesis. The University of Texas at Arlington, 2018 (cit. on pp. 97, 99, 100).

[Smo19] M. Smolaks. 'Power density – the real benchmark of a data centre'. In: *Virtus Data Centres* (2019) (cit. on pp. 96, 105).

[Sny10] J. Snyder. 'Firewalls in the Data Center: Main Strategies and Metrics'. In: *Opus One, White Paper* (2010) (cit. on p. 87).

[Spe] *SPEC Power Benchmark, SPECpower_ssj® 2008*. URL: `https://www.spec.org/power_ssj2008/` (cit. on p. 252).

[Str] S. Strömberg. *OpenNetHome – an open source software/hardware project for home automation*. URL: `https://opennethome.org/` (cit. on p. 133).

[TA04] W. Torell, V. Avelar. 'Mean Time Between Failure: Explanation and Standards'. In: *Schneider Electric, White Paper #78* (2004) (cit. on p. 88).

[TA16] M. L. Tuballa, M. L. Abundo. 'A review of the development of Smart Grid technologies'. In: *Renewable and Sustainable Energy Reviews* 59 (2016), pp. 710 –725 (cit. on pp. 25, 52).

[TCF+09] R. Tipley, T. Cader, J. Froedge, R. Tozer, R. Wofford. 'PUE Scalability Metric and Statistical Analysis'. In: *Green Grid, White Paper* (2009) (cit. on p. 75).

[TGE20] O. Taiwo, L. A. Gabralla, A. E. Ezugwu. 'Smart Home Automation: Taxonomy, Composition, Challenges and Future Direction'. In: *Computational Science and Its Applications – ICCSA 2020*. Cham: Springer International Publishing, 2020, pp. 878–894 (cit. on p. 49).

[TIPSB06] W. P. Turner IV, J. PE, P. Seader, K. Brill. 'Tier classifications Define Site Infrastructure Performance'. In: *Uptime Institute, White Paper* (2006) (cit. on p. 40).

[TKS09] R. Tozer, C. Kurkjian, M. Salim. 'Air Management Metrics in Data Centers'. In: *ASHRAE Transactions* 115.1 (2009) (cit. on p. 82).

[TLCS13] J. Tu, L. Lu, M. Chen, R. K. Sitaraman. 'Dynamic Provisioning in Next-Generation Data Centers with on-Site Power Production'. In: *Fourth International Conference on Future Energy Systems*. 2013, 137–148 (cit. on p. 60).

[TMOSP19] P. E. Teixeira Martins, M. Oleskovicz, A. L. da Silva Pessoa. 'A Survey on Smart Grids: concerns, advances, and trends'. In: *IEEE PES Innovative Smart Grid Technologies Conference - Latin America (ISGT Latin America)*. 2019, pp. 1–6 (cit. on p. 191).

[TS10] R. Tozer, M. Salim. 'Data center air management metrics – practical approach'. In: *12th IEEE Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems*. 2010, pp. 1–8 (cit. on pp. 42, 82).

[The10] The Royal Academy of Engineering. 'Wind Turbine Power Calculations'. In: *Mechanical and Electrical Engineering / Power Industry* (2010) (cit. on p. 221).

[Tra19] T. Trader. *AMD Launches Epyc Rome, First 7nm CPU*. `https://www.hpcwire.com/2019/08/08/amd-launches-epyc-rome-first-7nm-cpu/`. 2019 (cit. on p. 102).

[Tum10a] P. Tuma. 'Open Bath Immersion Cooling In Data Centers: A New Twist On An Old Idea'. In: *Electronics Cooling* (2010), p. 10 (cit. on p. 104).

[Tum10b] P. E. Tuma. ‘The merits of open bath immersion cooling of datacom equipment’. In: *26th Annual IEEE Semiconductor Thermal Measurement and Management Symposium (SEMI-THERM)*. 2010, pp. 123–131 (cit. on pp. 45, 104, 105).

[VAG10] G. Varsamopoulos, Z. Abbasi, S. K. Gupta. ‘Trends and effects of energy proportionality on server provisioning in data centers’. In: *2010 International Conference on High Performance Computing* (2010), pp. 1–11 (cit. on p. 81).

[VCA+20] A. V. Vesa, T. Cioara, I. Anghel, M. Antal, C. Pop, B. Iancu, I. Salomie, V. T. Dadarlat. ‘Energy Flexibility Prediction for Data Center Engagement in Demand Response Programs’. In: *Sustainability* 12.4 (2020) (cit. on p. 21).

[VD13] S. Vargas, R. Dines. ‘Calculate The ROI Of Data Center Investments’. In: *Forrester, White Paper* (2013) (cit. on p. 88).

[VPDS11] R. L. Villars, R. Perry, J. Daly, J. Scaramella. ‘Measuring the Business Value of Converged Infrastructure in the Data Center’. In: *International Data Corporation, White Paper* (2011) (cit. on p. 88).

[VPS+19] A. Verma, S. Prakash, V. Srivastava, A. Kumar, S. C. Mukhopadhyay. ‘Sensing, Controlling, and IoT Infrastructure in Smart Building: A Review’. In: *IEEE Sensors Journal* 19.20 (2019), pp. 9036–9046 (cit. on p. 22).

[VR08] V. Vukovic, L. Raković. ‘Open Source Approach in Software Development - Advantages and Disadvantages’. In: *International Scientific Journal of Management Information Systems* 3 (2008) (cit. on p. 122).

[VS06] J. W. Vangilder, S. K. Shrivastava. ‘Real-Time Prediction of Rack-Cooling Performance’. In: *ASHRAE Transactions* 112 (2006), pp. 151–162 (cit. on p. 78).

[VS07] J. W. Vangilder, S. K. Shrivastava. ‘Capture Index: An Airflow-Based Rack Cooling Performance Metric’. In: *ASHRAE Transactions* 113.1 (2007) (cit. on p. 82).

[Van11] O. VanGeet. ‘Best Practices Guide for Energy-Efficient Data Center Design’. In: *US Department of Energy Federal Energy Management Program* (2011) (cit. on p. 78).

[Var19] D. Varma. ‘Two-Phase Versus Single-Phase Immersion Cooling’. In: *Green Revolution Cooling* (2019) (cit. on pp. 97, 107).

[Vil20] H. Villa. ‘Liquid Cooling vs. Immersion Cooling Deployment’. In: *Rittal – The System* (2020) (cit. on pp. 97, 107).

[WA12] D. Wong, M. Annavaram. ‘Knightshift: Scaling the energy proportionality wall through server-level heterogeneity’. In: *45th Annual IEEE/ACM International Symposium on Microarchitecture* (2012), pp. 119–130 (cit. on p. 81).

[WAP+14] T. Wilde, A. Auweter, M. Patterson, H. Shoukourian, H. Huber, A. Bode, D. Labrenz, C. Cavazzoni. ‘DWPE, a new data center energy-efficiency metric bridging the gap between infrastructure and workload’. In: *2014 International Conference on High Performance Computing & Simulation (HPCS)*. 2014, pp. 893–901 (cit. on pp. 74, 75).

[WB15] C. Wu, R. Buyya. *Cloud Data Centers and Cost Modeling: A Complete Guide To Planning, Designing and Building a Cloud Data Center*. Morgan Kaufmann, 2015 (cit. on pp. 19, 82).

[WDDB12] G. Warkozek, E. Drayer, V. Debusschere, S. Bacha. ‘A new approach to model energy consumption of servers in data centers’. In: *IEEE International Conference on Industrial Technology*. 2012, pp. 211–216 (cit. on p. 43).

[WELL10] A. Wingkvist, M. Ericsson, R. Lincke, W. Löwe. ‘A Metrics-Based Approach to Technical Documentation Quality’. In: *7th International Conference on the Quality of Information and Communications Technology, QUATIC 2010*. 2010, pp. 476–481 (cit. on p. 131).

[WK13] L. Wang, S. U. Khan. ‘Review of performance metrics for green data centers: a taxonomy study’. In: *The Journal of Supercomputing* 63.3 (2013), pp. 639–656 (cit. on pp. 42, 80).

[WLJ12] Q. Wang, M. Liu, R. Jain. ‘Dynamic pricing of power in Smart-Grid networks’. In: *IEEE 51st IEEE Conference on Decision and Control (CDC)*. 2012, pp. 1099–1104 (cit. on p. 192).

[WRMB+06] C. A. Webber, J. A. Roberson, M. C. McWhinney, R. E. Brown, et al. 'After-hours power status of office equipment in the {USA}'. In: *Energy* 31.14 (2006), pp. 2823–2838 (cit. on p. 204).

[WV17] B. Watson, V. K. Venkiteswaran. 'Universal Cooling of Data Centres: A CFD Analysis'. In: *Energy Procedia, 9th International Conference on Applied Energy* 142 (2017), pp. 2711–2720 (cit. on p. 46).

[WZY+22] P. Wang, P. Zhong, M. Yu, Y. Pu, S. Zhang, P. Yu. 'Trends in energy consumption under the multi-stage development of ICT: Evidence in China from 2001 to 2030'. In: *Energy Reports* 8 (2022), pp. 8981–8995 (cit. on p. 21).

[Wau] F. Wautier. *AutoBuddy – a would-be home automation system*. URL: `http://frawau.github.io/AutoBuddy/` (cit. on p. 133).

[Wes21] S. Weston. 'Microsoft is submerging servers in boiling liquid to prevent Teams outages'. In: *ITPro.* (2021) (cit. on pp. 97, 98).

[Wib08] M. Wiboonrat. 'An Empirical Study on Data Center System Failure Diagnosis'. In: *2008 The Third International Conference on Internet Monitoring and Protection*. IEEE. 2008, pp. 103–108 (cit. on pp. 42, 88).

[Wib14] M. Wiboonrat. 'Data center infrastructure management WLAN networks for monitoring and controlling systems'. In: *The International Conference on Information Networking 2014 (ICOIN2014)*. 2014, pp. 226–231 (cit. on p. 48).

[XAW13] Y. Xia, K. Ahmed, B. Williams. 'Wind Turbine Power Coefficient Analysis of a New Maximum Power Point Tracking Technique'. In: *IEEE Transactions On Industrial Electronics* 60.3 (2013), pp. 1122–1132 (cit. on p. 221).

[YQZ+21] L. Yu, S. Qin, M. Zhang, C. Shen, T. Jiang, X. Guan. 'A Review of Deep Reinforcement Learning for Smart Building Energy Management'. In: *IEEE Internet of Things Journal* 8.15 (2021), pp. 12046–12063 (cit. on p. 52).

[Yas09] R. Yasin. *5 steps to secure your data center*. 2009. URL: `https://gcn.com/articles/2009/11/30/5-steps-to-a-secure-data-center.aspx` (cit. on p. 86).

[Yin09] R. K. Yin. *Case Study Research: Design and Methods*. 5th Edition. Sage Publications, 2009 (cit. on p. 56).

[ZB17] A. Zeck, J. Bouroudjian. ‘Real-World Experience with a Multicloud Exchange’. In: *IEEE Cloud Computing* 4.4 (2017), pp. 6–11 (cit. on p. 111).

[ZKS+19] S. K. Zaman, A. U. R. Khan, J. Shuja, T. Maqsood, S. Mustafa, et al. ‘A Systems Overview of Commercial Data Centers: Initial Energy and Cost Analysis’. In: *International Journal of Information Technology and Web Engineering* 14 (2019), pp. 42–65 (cit. on p. 116).

[ZLWR15] L. Zhang, Z. Li, C. Wu, S. Ren. ‘Online Electricity Cost Saving Algorithms for Co-Location Data Centers’. In: *IEEE Journal on Selected Areas in Communications* 33.12 (2015), pp. 2906–2919 (cit. on p. 24).

[Zho19] Y. Zhong. ‘A Large Scale Deployment Experience Using Immersion Cooling in Datacenter’. In: Alibaba Group: Open Compute Project Summit, 2019 (cit. on p. 98).

[ioBa] ioBroker GmbH. *ioBroker – Architecture*. URL: `https://github.com/ioBroker/ioBroker/blob/master/img/architecture.png` (cit. on p. 138).

[ioBb] ioBroker GmbH. *ioBroker – Automate your life, open source automation platform*. URL: `https://www.iobroker.net/` (cit. on p. 133).

[kin] king-dopey on GitHub. *Pytomation*. URL: `http://www.pytomation.com/` (cit. on p. 133).

All URLs were verified on 15.08.2022.

List of Figures

List of Tables

Acronyms

1PIC	Single-Phase Immersion Cooling
2FA	Two-factor Authentication
2PIC	Two-Phase Immersion Cooling
ACE	Availability, Capacity, and Efficiency
ACPR	Average Comparisons Per Rule
AEUF	Air Economizer Utilization Factor
AE	Airflow Efficiency
AI	Artificial Intelligence
AMD	Advanced Micro Devices, Inc.
APC	Adaptability Power Curve
API	Application Programming Interface
AR	Activity Recognition
ASHRAE	American Society of Heating Refrigeration and Air Conditioning Engineers
AS	Accessibility Surface
ATR	Application Transaction Rate
A	Availability
β_{index}	Recirculation Index
BJC	Bits per Joule Capacity
BLE	Bluetooth Low Energy

BMS	Building Management System
BPR	Bypass Ratio
BR	Balance Ratio
BVCI	Business Value of Converged Infrastructure
CADE	Corporate Average Data Center Efficiency
CCr	Carbon Credit
CC	Concurrent Connections
CEE	Capacity Energy Efficiency
CER	Connection Establishment Rate
CI/CD	Continuous Integration and Continuous Delivery
CI	Capture Index
CNEE	Communication Network Energy Efficiency
CO_2S	CO_2 Savings
CO	Carbon Monoxide
CPE	Compute Power Efficiency
CPU	Central Processing Unit
CP	Context Processing
CRAC	Computer Room Air Conditioning
CRAH	Computer Room Air Handler
CSS	Cascading Style Sheets
CTR	Connection Tear down Rate
CUE	Carbon Usage Effectiveness
CapEx	Capital Expenditure
CoP	Coefficient of Performance Ensemble
D^2	Generalized Mahalanobis Distance
DC-FVER	Data Center Fixed to Variable Energy Ratio
DCA	DCAdapt
DCCSE	Data Center Cooling System Efficiency
DCIM	Data Center Infrastructure Management
DCLD	Data Center Lighting Density
DCM	Data Center Management
DCPD	Data Center Power Density
DCPE	Data Center Performance Efficiency

DCP	Data Center Productivity
DCSSF	Data center Cooling System Sizing Factor
DCcE	Data Center Compute Efficiency
DCeP	Data Center Energy Productivity
DCiE	Data Center Infrastructure Efficiency
DC	Data Center
DCO	Data Center Operator
DEEPI	Data Center Energy Efficiency and Productivity Index
DH-UE	Deployed Hardware Utilization Efficiency
DH-UR	Deployed Hardware Utilization Ratio
DPPE	Data Center Performance Per Energy
DP	Dew Point
DR	Dynamic Range
DSO	Distribution System Operator
DS	Diameter Stretch
DTE	Data Transmission Exposure
DVFS	Dynamic Voltage and Frequency Scaling
DWPE	Data center Workload Power Efficiency
DeD	Defense Depth
DeP	Detection Performance
ECR-VL	Energy Consumption Rating Variable Load
EDE	Electronics Disposal Efficiency
EER	Energy Efficiency Ratio
EES	Energy ExpenseS
ENTSO-E	European Network of Transmission System Operators for Electricity
EP	Energy Proportionality
ERE	Energy Reuse Effectiveness
ERF	Energy Reuse Factor
EUE	EnergeTIC Usage Effectiveness
EWR	Energy Wasted Ratio
EoR	End of Row
FC	Firewall Complexity

FLOPS	Floating Point Operations Per Second
FL	Firewall Latency
FpW	FLOPS per Watt
GEC	Green Energy Coefficient
GPS	Global Positioning System
GPU	Graphics Processing Unit
GUF	Grid Utilization Factor
GUI	Graphical UI
H-POM	IT Hardware Power Overhead Multiplier
HF	Heat Flux
HPC	High Performance Computing
HPDL	Hierarchical Planning Definition Language
HSE	HVAC System Effectiveness
HTN	Hierarchical Task Network
HTR	HTTP Transfer Rate
HTTP	Hypertext Transfer Protocol
HVAC	Heating, Ventilation and Air Conditioning
I/O	Input / Output
IAS	Interface Accessibility Surface
ICT	Information and Communications Technology
IDC	Internet Data Centers
IPFH	IP Fragmentation Handling
IPR	Idle-to-peak Power Ratio
IP	Internet Protocol
ITEE	IT Equipment Energy
ITEU	IT Equipment Utilization
ITH	Illegal Traffic Handling
ITIL	Information Technology Infrastructure Library
IT	Information Technology
IoTemp	Imbalance of Temperature
IoT	Internet of Things
JSON	JavaScript Object Notation
kWh	kilowatt-hour

λ_{faults}	Faults per Hour
LDR	Linear Deviation Ratio
LD	Linear Deviation
LSP	Low-cost Storage Percentage
LoC	Lines of Code
M$_x$	Mass Flow of x
MFA	Multi-factor Authentication
MILP	Mixed-Integer Linear Programming
MQTT	Message Queue Telemetry Transport
MRR	Material Recycling Ratio
MTBF	Mean Time Between Failures
MTTF	Mean Time To Failure
MTTR	Mean Time To Repair
N/A	Not Applicable or Not Available
NPR	Negative Pressure Ratio
NPUE	Network Power Usage Effectiveness
NT$_{kwh}$	Network Traffic per kWh
NTP	Network Time Protocol
NUC	Next Unit of Computing
ω_{energy}	Water Usage Energy
OSE	Overall Storage Efficiency
OSWE	Operating System Workload Efficiency
OpEx	Operational Expenditure
PC	Personal Computer
PDE	Power Density Efficiency
PDS	Power Distribution System
PDU	Power Distribution Unit
PEsavings	Primary Energy Savings
PG	Proportionality Gap
PIR	Passive Infrared
pPUE	Partial Power Usage Effectiveness
PS	Path Stretch
PUE$_{scalability}$	Power Usage Effectiveness Scalability

PUE	Power Usage Effectiveness
PV	Photo-voltaic
PpW	Performance per Watt
QoS	Quality of Service
R^2	Coefficient of Determination
RAM	Random Access Memory
RA	Rule Area
RCD	Rogue Change Days
RCI	Rack Cooling Index
RC	Reachability Count
RDHx	Rear Door Heat Exchanger
REST	Representational State Transfer
RHI	Return Heat Index
RH	Relative Humidity
RI	Recirculation Index
RMSE	Root Mean Squared Error
ROI	Return On Investment
RPC	Remote Procedure Call
RR	Recirculation Ratio
RS_{max}	Maximum Relative Size
RTI	Return Temperature Index
RT	Response Time
S.M.A.R.T.	Self-Monitoring, Analysis and Reporting Technology
SCHx	Side Car Heat Exchanger
SDN	Software-defined Networking
SHI	Supply Heat Index
SH	Scalable Hierarchical
SI-POM	Site Infrastructure Power Overhead Multiplier
SLA	Service-level Agreement
SMS	Short Message Service
SPUE	Server Power Usage Efficiency
SSD	Solid-state Drive
SU	Slot Utilization

SWaP	Space, Watts and Performance
ScE	Server Compute Efficiency
TCE	Technology Carbon Efficiency
TCO	Total Cost of Ownership
TCP	Transmission Control Protocol
TEER	Telecommunications Energy Efficiency Ratio
TGI	The Green Index
TP$_{IP}$	Throughput IP
TP$_{i/o}$	Throughput I/O
TUE	Total-Power Usage Effectiveness
ToR	Top of Rack
T	Vulnerability Exposure
U$_{CPU}$	CPU Utilization
U$_{DC}$	Data Center Utilization
U$_{mem}$	Memory Utilization
U$_{network}$	Network Utilization
U$_{server}$	Server Utilization
U$_{storage}$	Storage Usage
UDP	User Datagram Protocol
UI	User Interface
UPS$_{CF}$	Uninterruptible Power Supply Crest Factor
UPS$_{EE}$	Uninterruptible Power Supply Energy Efficiency
UPS$_{PFC}$	Uninterruptible Power Supply Power Factor Corrected
UPS$_{PF}$	Uninterruptible Power Supply Power Factor
UPS$_{SF}$	Uninterruptible Power Supply Surge Factor
UPS	Uninterruptible Power Supply
UX	User Experience
WEUF	Water Economizer Utilization Factor
WUE	Water Usage Effectiveness
XML	Extensible Markup Language
YAML	YAML Ain't Markup Language

Brian Setz

An Internet of Things and Data-Driven Approach to Data Centers

2022

Universität Stuttgart

Data centers are a fundamental part of modern society, as they ensure that today's ICT services are available and responsive. The current trends show that data centers are increasing in size, which corresponds with an increase in their already significant energy footprint. As governments around the globe struggle to achieve their desired sustainability goals, the need for efficient and sustainable data centers is urgent. The Internet of Things (IoT) paradigm plays an important role in increasing the efficiency and sustainability of buildings. The smart home and office domains demonstrate the successful application of IoT in order to achieve energy savings through continuous monitoring and optimization of the building environment. Extending these techniques to the data center domain promises to bring the smartness previously seen in homes and offices to the data center. However, there are challenges that need to be overcome before the concept of a smart and optimized data center becomes a reality. In this thesis, a number of these challenges are addressed from a data-driven and IoT point of view.