Pós-Graduação em Ciência da Computação

Eltton Tullyo Silva Araujo

Availability and Reliability Evaluation of an IoT System:
Hierarchical Modelling for Smart Buildings

Universidade Federal de Pernambuco
posgraduacao@cin.ufpe.br
http://cin.ufpe.br/~posgraduacao

Recife
2020

Eltton Tullyo Silva Araujo

Availability and Reliability Evaluation of an IoT System:
Hierarchical Modelling for Smart Buildings

A M.Sc. Dissertation presented to the Center for Informatics of Federal University of Pernambuco in partial fulfillment of the requirements for the degree of Master of Science in Computer Science.

Concentration Area: Performance and Dependability Evaluation
Advisor: Paulo Romero Martins Maciel

Recife
2020

I dedicate this work to my parents who always bent over backward to make me get to where I am today.

ACKNOWLEDGEMENTS

Thanks God for allowing me to reach this moment, giving me opportunities, health and strength to overcome adversities. I would like to thank my parents that bent over backward to make me a successful person, they are my inspiration. **Fernando Marcos Araujo and Maria Alcy Silva Araujo** are and will always be my heros. They are the greatest warriors that I ever met and I really proud of having they as my parents. Today if I see further it is because I stand on the shoulders of a giants, and this giants it are my parents for sure.

I would also like to thank my lab friends (MoDCS group), which always tried to help me and responded to all my questions without hesitation or delays. They also built a really nice learning environment, where everyone is more than welcomed. My roommate Paulo for being such a nice person.

I am really grateful to Paulo Maciel for opening the doors of the CIn-UFPE to me. If someone asked me three years ago if I would be qualified to study there, I would totally say no. Therefore, I am really grateful for this opportunity. Paulo Maciel is also a hardworking Professor and a great inspiration for all his students.

To finish, I would like to thank the financial support from the funding agency CAPES, which was essential for concluding this stage of my life.

"Security is mostly a superstition. It does not exist in nature, nor do the children of men as a whole experience it. Avoiding danger is no safer in the long run than outright exposure. Life is either a daring adventure, or nothing."

(KELLER, 2005)

ABSTRACT

The Internet has been undergoing significant transformations and changing the world around it. We can also see the internet is used in many areas and innumerable purposes, even by objects. This evolution leads to the definition of the Internet of Things (IoT) concept. It is a system composed of storage resources, sensor devices, controllers, applications, and network infrastructure, to provide some service to its users. Since IoT comprises heterogeneous components, the creation of these systems, as well as the communication and maintenance of their components, becomes a complex task. There are growing interest and efforts of academia and industry in areas such as smart cities, smart homes, and industry 4.0, which may be seen as IoT-related concepts. This dissertation presents a sensitivity analysis of a system of IoT, considering distinct infrastructures and models to assess their availability metrics. A case study was carried out based on different scenario considering a smart building. The proposed models enable us to estimate measures such as steady-state availability. Thus, it was possible to notice a gain of availability in the system when applying redundancy to a local management infrastructure. Besides, there was a gain of availability when considering cloud computing infrastructure to manage a system comprising two or more buildings. Sensitivity analysis revealed the components that most affect the availability. I also performed a comparative analysis of budget costs between on-premises and cloud-based management infrastructures. This study considered a smart photovoltaic system, where the previous reviews served as the basis for the proposed system. Besides, was evaluate the deployment cost and obtained economy. Thus, it was possible to notice that there was no considerable gain of availability in the system when applying grid-tie solar power or off-grid solar power. The grid-tie solar power system is cheaper than the off-grid solar power system, even though it produces more energy. However, in our research, we were able to observe that the off-grid solar power system recovers the applied financial investment in smaller interval of time.

Keywords: Internet of Things, Dependability model, Smart Building.

RESUMO

A Internet está passando por transformações significativas e mudando o mundo ao seu redor. Também podemos ver que a internet é usada em muitas áreas e inúmeros propósitos, mesmo por objetos. Essa evolução leva à definição do conceito de Internet das Coisas (IoT). É um sistema composto por recursos de armazenamento, dispositivos sensores, controladores, aplicativos e infraestrutura de rede, a fim de fornecer algum serviço aos seus usuários. Como a IoT compreende componentes heterogêneos, a criação desses sistemas, bem como a comunicação e manutenção de seus componentes, torna-se uma tarefa complexa. Há um crescente interesse e esforços da academia e da indústria em áreas como cidades inteligentes, casas inteligentes e indústria 4.0, que podem ser vistos como conceitos relacionados à IoT. Esta dissertação apresenta uma análise de sensibilidade de um sistema de IoT, considerando diferentes infraestruturas e modelos para avaliar suas métricas de disponibilidade. Um estudo de caso foi realizado com base em diferentes cenários, considerando um edifício inteligente. Os modelos propostos nos permitem estimar medidas como a disponibilidade em estado estacionário. Assim, foi possível notar um ganho de disponibilidade no sistema ao aplicar redundância a uma infraestrutura de gerenciamento local. Além disso, houve um ganho de disponibilidade ao considerar a infraestrutura de computação em nuvem para gerenciar um sistema composto por dois ou mais edifícios. A análise de sensibilidade revelou os componentes que mais afetam a disponibilidade. Também realizei uma análise comparativa dos custos orçamentários entre as infra-estruturas de gerenciamento locais e baseadas na nuvem. Este estudo considerou um sistema fotovoltaico inteligente, onde as análises anteriores serviram de base para o sistema proposto. Além disso, foi avaliado o custo de implantação e a economia obtida. Assim, foi possível perceber que não houve ganho considerável de disponibilidade no sistema ao aplicar a energia solar vinculada à rede ou a energia solar fora da rede. O sistema de energia solar vinculado à rede é mais barato do que o sistema de energia solar fora da rede, embora produza mais energia. Porém, em nossa pesquisa, pudemos observar que o sistema de energia solar off-grid recupera o investimento financeiro aplicado em menor intervalo de tempo.

Palavras-chaves: Internet das coisas, Modelo de dependabilidade, edifício inteligente

LIST OF FIGURES

Figure 1 – Dependability tree (based in (AVIZIENIS et al., 2001)) 21
Figure 2 – Reliability Block Diagram example 26
Figure 3 – Markov Chain example 29
Figure 4 – Petri Net components - **a**) Token. (**b**) Transition. (**c**) Place. (**d**) Arc. . 31
Figure 5 – Petri Net example 32
Figure 6 – Components added with SPN - **a**) Exponential Transitions. (**b**) Inhibit Arcs. 34
Figure 7 – Systems that compose a smart building 46
Figure 8 – Generic smart building 47
Figure 9 – Communication between components 48
Figure 10 – Proposed IoT environment 49
Figure 11 – Local management infrastructure 49
Figure 12 – Local management infrastructure with redundancy 50
Figure 13 – Cloud computing management infrastructure 50
Figure 14 – SPN to entire system with local management 52
Figure 15 – SPN to entire redundant local system 54
Figure 16 – SPN to entire cloud system 56
Figure 17 – Common components in a photovoltaic system 59
Figure 18 – Grid-tie photovoltaic system main components 60
Figure 19 – Hybrid photovoltaic system main components 60
Figure 20 – Proposed IoT environment 61
Figure 21 – Building's sensors 61
Figure 22 – Communication between sensors and cloud computing 62
Figure 23 – Local management infrastructure 65
Figure 24 – SPN to entire system with local management 66
Figure 25 – Downtime difference between cloud and local infrastructure in hours . . 72
Figure 26 – Sensitivity of system availability with respect to $MTTF_F$ (IoT floor environment failure) 73
Figure 27 – Sensitivity of system availability with respect to $MTTR_F$ (IoT floor environment failure) 74
Figure 28 – New sensitivity of system availability with respect to $MTTF_F$ (IoT floor environment failure) 75
Figure 29 – Sensitivity of system availability with respect to $MTTF_{HW/OS}$ (hardware and operating system failure) 76
Figure 30 – Sensitivity of system availability with respect to $MTTF_{DB}$ (database component failure) 77

Figure 31 – Sensitivity of system availability with respect to $MTTF_{DM}$ (device manage component failure) 78
Figure 32 – Sensitivity of system availability with respect to $MTTF_{HW/OS/DC}$ (hardware, operating system, and docker failure) 79
Figure 33 – Sensitivity of system availability with respect to $MTTR_{DM}$ (device manage repair) 80
Figure 34 – Sensitivity of system availability with respect to $MTTR_{DB}$ (database repair) 80
Figure 35 – Equipment cost 87
Figure 36 – (a) KW consumed (b) Value spent 88
Figure 37 – Years to offset all the investment 88

LIST OF TABLES

Table 1 – The most relevant related works comparison. 45
Table 2 – Input values for SPN 69
Table 3 – Comparison between local infrastructures 70
Table 4 – Availability metrics results to building with 15 floors 70
Table 5 – Comparison between local and cloud with cold-standby in relation of cost - 3 years 71
Table 6 – Comparison between local and cloud with cold-standby 72
Table 7 – Sensitivity ranking for local infrastructure without redundancy 73
Table 8 – Sensitivity analysis for local infrastructure without redundancy and requirement of 14 out of 15 floors working 74
Table 9 – Sensitivity analysis for local infrastructure with redundancy 76
Table 10 – Sensitivity analysis to local infrastructure with redundancy and requirement of 14 out of 15 floors working 77
Table 11 – Sensitivity analysis to cloud computing infrastructure 78
Table 12 – Sensitivity analysis to cloud computing infrastructure and requirement of 14 out of 15 floors working 79
Table 13 – Annual electricity Peak Demand (KW) 81
Table 14 – Daily heat stroke, monthly average 83
Table 15 – Input values for SPN 85
Table 16 – Availability metrics results to whole system 86
Table 17 – Equipment cost 86
Table 18 – Energy produced/Reduction cost 87

CONTENTS

1 INTRODUCTION 14
1.1 MOTIVATION AND JUSTIFICATION 16
1.2 OBJECTIVES 17
1.3 STRUCTURE OF THE DISSERTATION 18

2 BACKGROUND 19
2.1 INTERNET OF THINGS 19
2.2 DEPENDABILITY 20
2.2.1 Threats 21
2.2.2 Means 22
2.2.3 Attributes 23
2.3 MODELING 25
2.3.1 Reliability Block Diagram - RBD 26
2.3.2 Markov Chain 29
2.3.3 Petri Nets 30
2.3.4 Stochastic Petri Net 32
2.4 SENSITIVITY ANALYSIS 34
2.5 SOLAR POWER SYSTEM 34
2.6 FINAL REMARKS 37

3 RELATED WORKS 38
3.1 DEPENDABILITY ON INTERNET OF THINGS 38
3.2 DEPENDABILITY ON CLOUD COMPUTING 40
3.3 DEPENDABILITY ON PHOTOVOLTAIC SYSTEM 42
3.3.1 Analysis of related works 44
3.4 FINAL REMARKS 45

4 SMART BUILDING SYSTEMS 46
4.1 SMART BUILDING 46
4.1.1 Generic Smart Building 47
4.1.2 Proposed generic Architecture to smart building 48
4.1.3 Availability and Relibility Related Metrics 50
4.1.3.1 Availability for a single floor IoT environment 51
4.1.3.2 SPN Availability Model for Entire System 51
4.1.3.3 SPN Availability Model for Entire Redundant System 54

4.1.3.4 SPN availability model for system with redundant cloud computing management . . . 55
4.2 MANAGEMENT PHOTOVOLTAIC SYSTEM . . . 59
4.2.1 Photovoltaic system . . . 59
4.2.2 Smart Photovoltaic System . . . 60
4.2.3 Availability and Relibility Related Metrics . . . 62
4.2.3.1 Model of the Building . . . 62
4.2.3.2 Model of the Hybrid Solar System . . . 63
4.2.3.3 Model of Grid-tied Solar System . . . 64
4.2.3.4 SPN Availability Model of the Entire System . . . 65

5 CASE STUDIES . . . 69
5.1 GENERIC SMART BUILDING . . . 69
5.1.1 Availability evaluate in a generic architecture to smart building . . . 70
5.1.2 Comparison between non-redundant and redundant local infrastructures . . . 70
5.1.3 Comparison between local and cloud infrastructures . . . 71
5.1.4 Sensitivity analysis . . . 72
5.2 SMART PHOTOVOLTAIC SYSTEM . . . 80
5.2.1 Photovoltaic systems . . . 81
5.2.1.1 Hybrid solar system . . . 81
5.2.1.2 Grid-tie solar power system . . . 83
5.2.2 Results obtained with the proposed models . . . 84
5.2.2.1 Autonomous energy management system: Availability . . . 85
5.2.2.2 Autonomous energy management system: Deploying cost . . . 86
5.2.2.3 Autonomous energy management system: Cost reduction . . . 86
5.3 FINAL REMARKS . . . 88

6 CONCLUSIONS AND FUTURE WORK . . . 89
6.1 CONTRIBUTIONS . . . 90
6.2 FUTURE WORK . . . 91

REFERENCES . . . 92

1 INTRODUCTION

The evolution of communication systems and devices brought a significant number of new applications. In this mobile context, the number of smart devices or objects capable of communicating and computing, such as sensors, home appliances, and smartphones, has been growing (STOJKOSKA; TRIVODALIEV, 2017). Therefore, it is necessary to create a heterogeneous communication network capable of connecting several types of objects. This paradigm is popularly known as the Internet of Things (IoT). S. Haller (HALLER, 2009) defines that physical objects are seamlessly integrated into the information network and can become active participants in business processes, and services are available to interact with these 'smart objects' over the Internet. IoT allows users to access and pull data more quickly and intuitively. Thus, with a large amount of information generated by sensors and actuators, many new services may be created in many different areas, for example, industrial automation, health, traffic, environmental, agriculture, and livestock, among others (ZANELLA et al., 2014).

IoT devices have limited computational resources, especially in terms of processing and storage capacity (SATYANARAYANAN et al., 2009). Cloud computing is an alternative, as one of its key features is its ability to provide abundant computing resources to its customers (ARMBRUST et al., 2010). However, despite the high heterogeneity and wide distribution of feature-rich devices that IoT and cloud computing can offer. It should be noted that the variety of components in this scenario can be challenging, especially from the perspective of service dependability. That is, it must be ensured that each component that is part of the service architecture has its proper functioning, mitigating the commitment of the service as a whole.

In Smart Cities, the urban space is a stage for the intensive use of context-sensitive communication and information technologies. This information mainly helps urban management and social aspects. (COCCHIA, 2014). These interaction flows employ infrastructure and services, leveraging the use of information, communication for strategic goals of urban planning and management, to meet social needs along with the economic needs of society (NAM; PARDO, 2011).

In general, projects involving Smart Cities can be subdivided into areas of expertise, such as Smart Governance and Education, which aims to improve government operations and services through the application of technology (SCHOLL; SCHOLL, 2014)(JOHNSTON; HANSEN, 2011), Smart Health-care projects addressing health aspects (CATARINUCCI et al., 2015)(MANOGARAN et al., 2018), Smart Mobility cares about aspects related to traffic (BENEVOLO; DAMERI; D'AURIA, 2016)(NING et al., 2017). Estimates for the Smart Cities market size by 2025 may reach 2.57 trillion dollars, implying that more and more research should be done to improve concepts and technology (RESEARCH, 2018 (accessed November

28, 2018)). Another area related to Smart Cities is the smart building, where several applications are possible, such as energy optimization and safety (MINOLI; SOHRABY; OCCHIOGROSSO, 2017).

When we talk about smart buildings, we talk about the convergence of technologies and the management of building infrastructure. This means applying IT management solutions to maintain, improve efficiency, and reduce expenses such as electricity. (MCGLINN et al., 2010) define Smart Buildings as "a subset of smart environments" where smart environments are "able to acquire and apply knowledge about the environment and its inhabitants in order to improve their experience in that environment" (COOK; DAS, 2007). In this context, green buildings can be inserted. The green building is a building or any space or environment built with social, environmental, and economic sustainability in mind, from conception, construction, and throughout its operation (KIBERT, 2016). Environmental benefits include rational use and reduced extraction of natural resources. Besides, imply the reduction and efficient consumption of water and energy, the use of materials, and technologies with low environmental impact, among others. In the social sphere, the benefits are: improved occupant health security, social inclusion, and increased sense of community, social and environmental awareness, increased employee productivity - due to increased satisfaction and well-being, among many others. In this aspect, we consider many automation opportunities, employing a set of sensors to improve the security and reduction of energy consumption.

Some researchers have been assessing several aspects of IoT systems. In (ŞTEFAN; OTTO; ALEXANDRINA, 2017), the authors proposed the evaluation of dependability in a smart home system. This study was considered the data network paths of the IoT devices to a storage server. These paths were considered with and without redundancy. The result showed that the maximum availability of 97,47% was reached using redundancy in all device paths. In (ARAUJO et al., 2016), the authors had two objectives. The first one was to identify the most critical components of a mHealth system through parametric sensitivity analysis, observing the availability. The second one was to propose and design an extended mHealth architecture with higher availability and lower downtime period. The proposed solution reduced the downtime by 41% compared to the baseline architecture.

This work aims to identify the components that most affect the availability of an IoT system. Therefore, this work presents a hierarchical model, comprising closed-form equations and Stochastic Petri Nets (SPNs) models to represent and evaluate the proposed systems. Thus, a smart building system was chosen for analysis, considering local management infrastructure without redundancy, and it is compared to a local infrastructure with redundancy. In this context, it was also considered a cloud computing infrastructure. The deployment costs of the considered systems were also calculated. Besides, the availability was evaluated in a scenery specific considering two energy power smart systems. The result of this comparison showed that downtime using an on-grid solar power

smart system is similar to the hybrid solar energy power smart system. The deploying cost was also compared between the two systems. The result of this comparison showed that downtime using an on-grid solar power smart system is similar to the hybrid one. Besides, the deploying cost comparing the two systems showed that the on-grid is cheaper than the hybrid. Besides, our analysis showed that the off-grid solar power smart system would have a faster financial return.

1.1 MOTIVATION AND JUSTIFICATION

Many organizations are being attracted by the opportunities that the Internet of Things offers to lower their costs. Besides, the freedom that technology offers to the managers of those organizations focuses solely on strategies rather than managing resources, among other things. There is also great interest from governments for IoT services cities, which shows the importance of this trend in the world economy (WYLD, 2009) and quality of life of its citizens (SU; LI; FU, 2011). Several application domains will be affected by the emergence of IoT. The major domains of IoT are Energy, Smart City, Transportation, Smart Home, Environment, Supply Chain, and Health Care (GAZIS et al., 2015).

In such environments, high availability and performance are factors of great importance to be considered. Whereas, even a short downtime can result in a significant financial loss for the companies providing the services (SHARMA; CHEN; PARK, 2017). Besides, unavailability can also affect customer reliability. Usually, high availability and performance are often achieved through components replication. In cloud computing systems, this is commonly obtained by replicating the server and storage system. However, it is known that IoT systems are composed of several distinct components like sensors, storage systems, among others. Then, to achieve high availability and performance, it is necessary to know the components that most affect these metrics. Some researchers are directing their efforts in research to analyze the impact these components in the availability and performance of the various IoT systems. These studies aimed at increasing users' confidence and satisfaction on these services (SHARMA et al., 2017).

Achieving satisfactory levels of dependability in IoT environments is not a trivial task. However, to evaluate these levels, the use of models is considered useful as it predicts the components' behavior. In order to computationally represent the main aspects of systems in smart building, modeling can provide support for dependability planning and evaluation, abstracting technical complexities (JAIN et al., 1990). Evaluating the attributes of dependability through models is a viable means to propose improvements to systems. Therefore, modeling is a means of evaluating various systems, including those that require high availability of services (KIM; MACHIDA; TRIVEDI, 2009).

The adoption of models for supporting planning and smart buildings' design contributes to achieving services high availability and information loss prevention. It also avoids other unintended consequences that only tend to occur when the service is un-

available (KIM; MACHIDA; TRIVEDI, 2009). Issues such as availability have become crucial factors in ensuring the continuity of this type of service (SILVA et al., 2013).

In this scenario, it is important to create solutions that can meet the demand for information generated even more by addressing vital information for building management. Straightforwardly, it is indispensable to analyze all the characteristics that allow the system availability. Thus, this work's primary motivation is to define a formal model to monitor and evaluate the availability of a generic smart building system.

1.2 OBJECTIVES

This research's main objective is the proposition of models for the availability evaluation to support smart building's infrastructure planning. Initially, a baseline architecture is proposed for a smart housing building. In this architecture are considered some sensors, as smoke, motion, and gas. The sensors were input into each apartment and the hallway of each floor. Moreover, data collected by sensors are sent to a necessary local management infrastructure with minimum requirements. The next step was to insert redundancy in management infrastructure and compare the baseline architecture results. Then, the system was scaled to support more than one buildings, which was considered a local management infrastructure with redundancy in each building at first. Sequentially, we evaluate a cloud computing infrastructure capable of managing all buildings. To evaluate the availability of all architectures proposed, we used mathematical equations and SPN models. However, in addition to considering availability, the deployment cost was evaluated to all architectures. The results achieved for the best architecture, considering the cost and availability, were chosen to be used in a specific scenario. The specific scenario consists of evaluating the availability of a smart photovoltaic system on a housing building. This case study compared a smart photovoltaic system connected on-grid with a smart photovoltaic system with batteries.

In other words, there are some specific objectives, described as follows:

- Design and construction of basic infrastructure for IoT systems to support smart building;
- Generation of architectural models of buildings;
- Extension of architectural models of buildings to include energy, environmental and mobility aspects of the population;
- Obtaining estimates of annual energy consumption using the extended architectural model.

1.3 STRUCTURE OF THE DISSERTATION

This work is structured as follows. Chapter 2 introduces basic concepts of the Internet of Things, dependability, among others that are fundamental to understand this dissertation. Chapter 3 depicts some related works. The Architectures are presented in detail in Chapter 4, comprising the components, features, and a general view. Chapter 5 presents some experiments to evaluate the proposed architectures and the results that were achieved. Finally, Chapter 6 draws some conclusions of the dissertation and discusses some directions for future works.

2 BACKGROUND

Dependability is an important measure used to qualify several systems, such as aviation, systems of weather, automotive, health, and more. Internet of Things has various aspects of research, such as communication, security, architecture, etc. This chapter presents a background to the main components of the proposed strategy. This study is mainly about dependability in IoT systems based on a smart building. This work was used different types of management infrastructures and systems, such as smart energy, light, and security. Therefore, this Chapter, we give a brief explanation of dependability. Besides, are presented some concepts about the Internet of Things and sensitivity analysis. Lastly, is explain briefly about the photovoltaic system.

2.1 INTERNET OF THINGS

The concept of the Internet of things or only IoT emerged from the evolution of several areas such as microelectronics, sensing, communication, and distributed systems. However, in a nutshell, the Internet of Things is nothing more than an extension of the current Internet, which enables controlling and monitoring daily routine on smart objects, but with computational and communication skills, to connect to the Internet (LEA, 2018). However, in order to understand whats the IoT is, the following concepts are presented.

The first concept consists of the identification of the objects; that is, each object must have a unique identification number (UID) and an Internet Protocol (IP) address (GREENGARD, 2015). In turn, some technologies are used for this fact, as is the case of RFID (Radio-Frequency Identification), NFC (Near Field Communication), and IP (Internet Protocol address), in this case, only possible with the emergence of IPv6. These objects connect via cords, wired, and wireless technology, including satellites, cellular networks, Wi-Fi, and Bluetooth. The last concept defines what are "connected devices". In which, are devices that exchange data using some internet standard and obtain some benefit with this (GREENGARD, 2015). There are two types of connected devices, physical-first and digital-first. The physical-first are objects that usually do not generate or communicate digital data unless that be manipulated. The digital-first are objects capable of generating and communicating digital data for further use, inherently, and by design (GREENGARD, 2015). Therefore, the Internet of Things is too powerful because it connects physical-first objects and items and connects them to digital-first devices, including computers and software applications.

There are many aspects to consider when creating an IoT system. The first one is related to sensing and power. The larger part of IoT consists of capturing action or events of the real-world. Sometimes this involves big data generated from a temperature sensor

or perhaps an actuator moving on a lock. Therefore, project designer should define which data will be generated from devices. Besides, the IoT devices commonly have a low energy storage capacity, making this a critical factor in the project (LEA, 2018). Another critical part of the IoT system is data communication. This involves significant technologies to move data from the remotest and most hostile environment to the largest data centers such as Google, Amazon, Microsoft, and IBM. Besides, this aspect also considers Internet protocols and routing, such as HTTMP, SNMP, MQTT, and COAP. The project designer should define how communication between devices and storage systems will take place. The wrong choice can generate a significant impact on system performance and security (LEA, 2018).

Fog and edge computing, analytics, and machine learning also are aspects consider in IoT systems. A project designer should plan the data flow and service infrastructure. Understanding the architectures' constraints is essential to make a sound judgment on how a system will deploy and scale. Analytics and rules engines are used when the analog data captured by devices are transformed into digital data, which can generate actions (LEA, 2018). At last, the vulnerability should be an important aspect consider in IoT systems since most IoT systems will be in public or very remote areas, in moving vehicles or even inside a person, that makes the IoT a big, or even the most prominent target, for cyber attacks (LEA, 2018).

The IoT systems can be inserted in many social areas, and each one has an impact differ. The industrial and manufacturing are the largest segments in the overall IoT space, considering the number of devices connected to the value these services bring to the industry (LEA, 2018). The consumer segment was the first to adopt the IoT. This came into form as a connected coffee pot at a university in the 1990s. Now millions of homes have smart objects like light bulbs, assistants, among others. The energy segment is gaining more notoriety in this regard (LEA, 2018). A significant amount of research and development has focused on consumer and commercial energy monitors such as smart electric meters that communicate over low-power and long-range protocols to reveal real-time energy usage (LEA, 2018). This work presented a smart energy system, which adopts the sun as one of the energy sources.

2.2 DEPENDABILITY

In (AVIZIENIS et al., 2001) the term dependability is defined as the ability of a computational system to deliver or provide a service fairly and reliably. High-level definition considers dependability as a large black box that can only be analyzed from the system user's point of view, and its evaluation will be based on the expected behavior of this box, and the responses it provides to each entry made. Therefore, dependability is directly related to reliability, which can be defined as the probability of a system running without failure. By dealing with a broad term, despite the short definition, the dependability as-

sessment usually interrelates with the study of a set of **Threats** that prevent the delivery of the service in the desired way; of **Means** that are responsible for guaranteeing the results of the **Attributes** which, in turn, are nothing more than metrics related to the delivered service offered of the already mentioned just and reliable way. The properties can be schematized through a tree, as in Figure 1

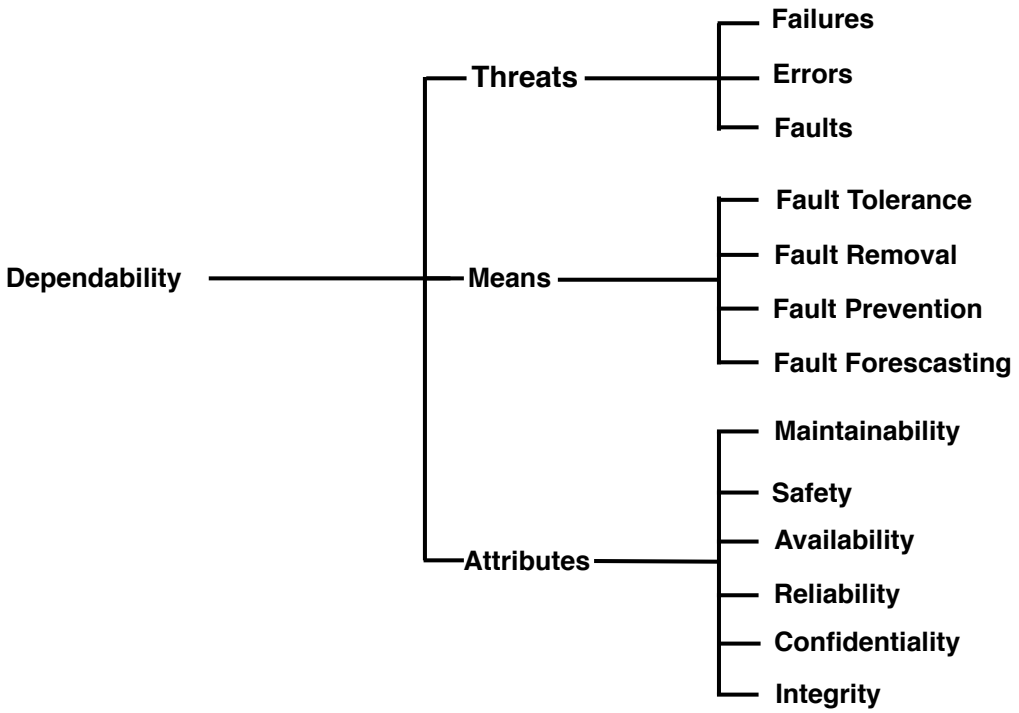


Figure 1 – Dependability tree (based in (AVIZIENIS et al., 2001))

The three main branches of the dependability tree (Threats, Means, and Attributes) are described in the following sections, and their relations with this dissertation are pointed out in the text.

2.2.1 Threats

There are three significant threats to dependability, failures, errors, and faults, presented in Figure 1. These threats are responsible for preventing the delivery of the service in the desired manner.

Errors are the software itself's threats and usually caused by bad programming in the development process. As well as, they can be even more rooted, having passed as unnoticed throughout the planning stage of the system; In general, errors can lead to a failure state. In turn, this refers to the time when an error reaches the service interface and changes its operation. Thus, becoming visible to users; While faults are said to be the probable or hypothetical causes that led to the occurrence of the error (AVIZIENIS et al., 2001). There are techniques to mitigate the impact of failures in computer systems. These techniques are in the branch next to one of the threats, and call themselves Means of Dependability.

2.2.2 Means

Prevention, **Tolerance**, **Removal**, and **Failure Prediction** are techniques that reduce the impact of Failures. The combination of these four techniques is used to provide a service fairly and reliably. Thus, it enables the system to reach the dependability metrics from which it was previously agreed (AVIZIENIS et al., 2001).

- **Fault Prevention**: failure prevention is the first of the means of dependability. It can be achieved through the adoption of quality control techniques. However, these techniques must be applied in the planning and manufacturing phases of the system. In which, all necessary hardware and software components must be considered. The sets of these make up the whole system (AVIZIENIS et al., 2001). Performing a requirements survey and evaluation step can be considered a quality control technique. As well as the adoption of structured programming for the system development process can be used. Another technique is the application of frequent tests on the system. The sets of these activities tend to be enough for a considerable reduction in failures in the future. However, in the occurrence of failures, it becomes indispensable to adopt some mechanism that ensures that the system is not unavailable to its users. Through of adoption of these, the system can continue delivering the service proposed manner correctly. These mechanisms are explained next.

- **Fault Tolerance**: consists of proposing redundant mechanisms that allow the correct delivery of services, even in the presence of faulty components (AVIZIENIS et al., 2001). A system capable of detecting the occurrence of errors automatically and the location of this event. There are several types of redundancy with Hot-standby and Cold-standby. In the first classification, the redundant modules in standby are working in sync with the operational module, without their computation being considered in the system, and, if a failure is detected, it is ready to become operational immediately. In cold standby, the redundant modules are turned off, and only when a fault occurs will they be activated.

- **Fault Remove**: is a constant process carried out during the development of a system and throughout its lifetime (AVIZIENIS et al., 2001). In contrast, we spend more time in the phases of primary analysis and validation. A shorter time tends to be spent on removing failures in the future. Therefore, this can directly lead to an increase in the availability of the product or service if we adopt mechanisms capable of predicting failures with the maximum possible accuracy. We could remove them before they occur, mitigating their impact on service delivery. This is the medium of dependability explained next.

- **Fault Forecasting**: when using techniques for the behavioral analysis of a system, it is possible to predict its behavior over some time. These techniques have a certain

degree of accuracy and confidence, capable of revealing when the service may no longer be offered correctly (AVIZIENIS et al., 2001). Therefore, with estimated values, it is possible to reduce and mitigate the impacts of a system failure occurrence. Furthermore, through the application of corrective and preventive maintenance, it is possible to increase the continuity in the service provision. This results in a more reliable system. Reliability is one of the six dependability attributes presented in the next Section.

2.2.3 Attributes

Dependability has many attributes. One of these attributes is systems or services´ availability. Such an attribute is central to the focus of this dissertation.

- **Availability**: the availability can be summarized as the probability of a system is operational; that is, it is accessible to its users within established previously period (COMMITTEE et al., 1990). The availability can be calculated using the Mean Time to Failure (MTTF), along with the Mean Time To Repair (MTTR), as is presented in Equation 2.1 (TRIVEDI, 1982).

$$A = \frac{MTTF}{MTTF + MTTR} \tag{2.1}$$

The MTTF of a system can be calculated through the integration of reliability as a function of time, Equation 2.2, (RAUSAND; HØYLAND, 2003). While determining the systems´ mean time to repair depends directly on the result from the previous derivative and is represented by Equation 2.3.

$$MTTF = \int_0^t R(t)\partial t \tag{2.2}$$

$$MTTR = MTTF \ \times \ (\frac{UA}{A}) \tag{2.3}$$

In Equation 2.3 the availability is given by (A) and unavailability is given by (UA = 1 - A). The system's downtime period can be understood as the period in which the service was unavailable within a time interval. The unavailability calculation considers the relationship between the time taken for a maintenance team to move to the location of the failure and the detection of its occurrence. The period of locomotion is known as non-repair time (NRT). Moreover, the time of repair is known as the time to repair (TTR). In this way, we can calculate the downtime of service through the expression Downtime = NTR + TTR, (MACIEL; TRIVEDI; KIM, 2010).

The system availability may also be represented through the time that the system tends to become available or Uptime and of its unavailability period or Downtime. These values are generally used and agreed in the SLA (Service Level Agreement). Therefore, availability is a metric of great importance for service providers and customers. Equation 2.4 represents the calculation of availability for a system or service (TRIVEDI, 1982).

$$A = \frac{E[Uptime]}{E[Uptime] + E[Downtime]} \tag{2.4}$$

where A equals availability and E to the expected value.

The availability also can be presented in the function of the number of nines (NN), as show Equation 2.5 (MACIEL; TRIVEDI; KIM, 2010). Therefore, the biggest the quantity, the greater its availability. Thus, a service with 99.44% availability can also be interpreted as a system with 2.25 nines of availability, for example.

$$NN = -log_{10}(UA) \tag{2.5}$$

It is worth pointing out that it is virtually impossible to guarantee the availability of 100%, not even the largest companies offering infrastructure and services on the Internet announce such high availability.

- **Reliability**: is the probability that the system has performed its function up to a predetermined and uninterrupted time limit (TRIVEDI, 1982). The reliability of a system can be measured similarly to the calculation of availability. However, the rates related to the repair of the same are not considered. This occurs because the service is considered down when there is a component failure. Thus, reliability can be calculated through Equation 2.6.

$$R(t) = P(T > t), t \geq 0 \tag{2.6}$$

Where the random variable T be the lifetime or the time to failure of a component. Moreover, t is the predetermined time-out in a range of [0, t] (KUO; ZUO, 2003).

- **Security**: is a metric directly related to the reliability of the system, since it depends on the nonexistence of interruptions in its service. It can be said that a system is safe when the occurrence of catastrophic failures is unlikely. However, it is not possible to say the same about the existence of imminent dangers (AVIZIENIS et al., 2004).

- **Integrity**: a system of integrity is one where improper changes will not be applied. Thus, we can say that a system's integrity is nothing more than a prerequisite for availability, reliability, and security metrics (AVIZIENIS et al., 2001).

- **Maintainability**: is the ability of a system to receive repairs and maintenance after failures occur. However, the maintainability is the probability of a system being repaired after failure at some point (AVIZIENIS et al., 2001).

- **Confidentiality**: a system is said to be confidential when there is no disclosure of data and information not correctly authorized. Thus, it aims to keep what it should be and to remain safe, stealthy (AVIZIENIS et al., 2001).

2.3 MODELING

In this section I refer to the model as a computational model in terms of dependability. According to (RAUSAND; HØYLAND, 2003) a model describe the essential features of the system, but do not necessarily have to be exact in all details. A model, in most cases, approximates the behavior of the real system, in which quality is given precisely by the ability demonstrated to predict the response of the system (output variables) to a stimulus (input variables). The level of abstraction of the model must be consistent with the objectives of the analysis, and the inclusion of many details may impact the complexity of the model, as well as its evaluation. On the other hand, the omission of important characteristics can result in inaccurate data. To according (TRIVEDI; BOBBIO, 2017), the modeling formalism can be classified into three types:

- Non-state-space (or combinatorial) models: *"Solutions of these models can be determined relatively efficiently with the assumptions that the components are statistically independent. They provide high analytical tractability since the underlying state space need not be generated, but the modeling power is low as dependence is not allowed"*. Reliability Block Diagram (explained in this work), Reliability Graph, and Fault Tree are example of this kind of model.

- State-space models: *"These models can account for statistical as well as state and time dependencies and hence possess high modeling power but less analytical tractability as they suffer from the curse of dimensionality. State-space methods are broadly classified into Markovian (homogeneous and non-homogeneous) and non-Markovian methods"*. Markov Chains and Petri Nets are example of this kinf of model, and both are explained in this work.

- Multi-level models: *"These combine the modeling power of state-space models with the efficiency of non-state-space models"*.

For the dependability modeling there are two categories of models that can be used: those that are classified as Combinatorial Models, which represent the conditions under which a system may be at fault, or be operational, in terms of structural relations between its components; and the Space-Based Models of States, which represent the behavior of

the system through states and events that cause transitions between states (MACIEL et al., 2012).

2.3.1 Reliability Block Diagram - RBD

RBD is a type of model initially proposed for the reliability analysis of systems and their interactions. This technique allows the use of combinations to model the behavior of a system and its modules. Thus, through this type of model, it is possible to produce a reliable analysis. The RBD also had its extended characteristics to the calculation of the availability of the computational systems. These systems can be small and straightforward, as well as large and complex (RAUSAND; HØYLAND, 2003).

Through RBD models, it is possible to describe the interaction between the system's various components using only reliability blocks. However, each block can represent a single component of the system or sub-components that make up a subsystem. In an RBD-type model, a system is only available if it is possible to trace a line, which extends from the origin vertex (Beginning) to its final vertex (End) (SMITH, 2017). All the components within this dashed line are in the state of availability, that is, functional.

A typical example possible of an RBD model is shown in Figure 2. This figure represents a system with four interconnected components. It is possible to observe the two main types of blocks usable within this type of model: in series such as Components 1 and 4 and Parallel as Components 2 and 3. This system will be available, as already mentioned, only if the line between beginning and end can be traced in at least one of the possible paths.

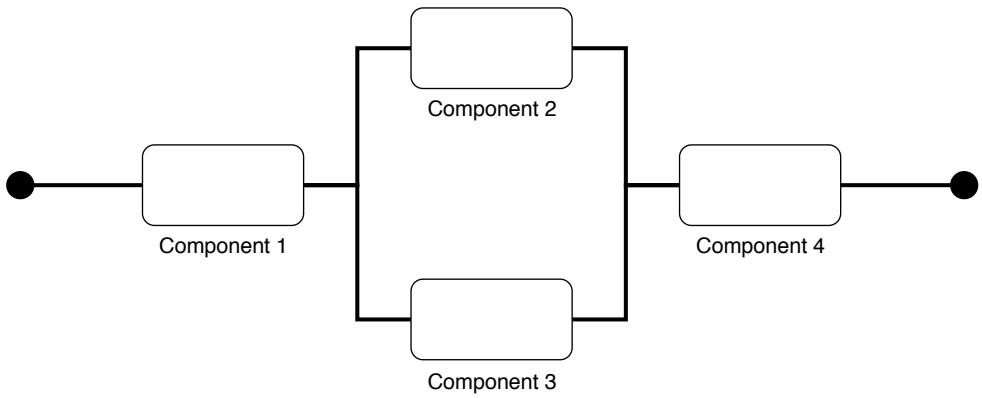


Figure 2 – Reliability Block Diagram example

The structural function provides a basic foundation for understanding the concepts of paths and cuts (KUO; ZHU, 2012) within an RBD; it is necessary to establish and determine a structural function. This function must be able to evaluate the availability of the system based on its modes of operation (TRIVEDI, 1982). We considered a system S with two components: 1 and 2. According to the state of each of its components, the

system can be in a state of failure or operational. The state of component i a random variable x_i gives, such that:

$$x = \begin{cases} 0, & if\ the\ component\ is\ not\ working \\ 1, & if\ the\ component\ is\ available \end{cases} \tag{2.7}$$

The Reliability and availability of our system if the components are connected in series can be calculated through Equation 2.8.

$$R_S(t) = R_1(t) \times R_2(t) \tag{2.8}$$

Where $R_1(t)$ is the reliability of component 1, and $R_2(t)$ is the reliability of component 2. Already for a system in series with n-elements the Equation 2.9 (TRIVEDI, 1982) is adopted.

$$R_S(t) = \prod_{i=1}^{n} R_i(t) \tag{2.9}$$

For a block sequence in parallel, Equation 2.10 can be used (TRIVEDI, 1982), in which, n represents the number of elements in parallel.

$$R_P(t) = 1 - \prod_{i=1}^{n} (1 - R_i(t)) \tag{2.10}$$

Another typical structure of RBD models is k-out-of-n. This structure is generally used when a set of components of the same type is distributed in parallel. In that, a quantity k of components can go into a failed state, and still, the system will remain in operation. This structure can be seen in Equation 2.11, which calculates its reliability (TRIVEDI, 1982).

$$R_{K-out-of-n}(t) = \sum_{i=k}^{n} P(X = i) \tag{2.11}$$

An RBD structure of order N consists of n ordering components from 1 to n. Where the set of components can be denoted by:

$$C = \{1, 2, ..., n\} \tag{2.12}$$

The **Path set** is a subset of components contained in C (we will denote it by P), where if all components are working, it means that the system is also working. More formally, the respective path set of a state vector is defined by Equation 2.13 (MACIEL et al., 2012). A path set is considered minimal, when it cannot be reduced, without losing the status of a path set. Therefore, a path vector x is called minimal path vector if $\phi(x) = 0$, for

any $y < x$, and the respective path set is named minimal path set (MACIEL et al., 2012). The minimal path (MP) can be defined by Equation 2.14

$$P(x) = \{c_i | \phi(x) = 1, x_i = 1, c_i \in C\} \tag{2.13}$$

$$MP(x) = \{c_i | c_i \in P(x), \phi(x) = 0, \forall y < x\} \tag{2.14}$$

In relation to Figure 2, where the component set is $C = \{1, 2, 3, 4\}$, the path set can be denoted by:

$$P = \{\{1, 2, 4\}, \{1, 3, 4\}, \{1, 2, 3, 4\}\} \tag{2.15}$$

The **Cut set** (we will denote by K) is a subset of components in C, where if a component fails, the whole system will fail. More formally, a state vector x is named a cut vector is $\phi(x) = 0$, and the respective set of faulty components is defined as cut set, as showed in Equation 2.16 (MACIEL et al., 2012). A cut set is considered minimal, when it cannot be reduced, without losing the status of a cut set. Therefore, a cut vector x is called minimal cut vector if $\phi(x) = 1$, for any $y > x$, and the respective path set is named minimal cut set. The minimal cut set (MK) can be defined by Equation 2.17 (MACIEL et al., 2012).

$$K(x) = \{c_i | \phi(x) = 0, x_i = 0, c_i \in C\} \tag{2.16}$$

$$MK(x) = \{c_i | c_i \in P(x), \phi(x) = 1, \forall y > x\} \tag{2.17}$$

In relation to Figure 2, where the component set is $C = \{1, 2, 3, 4\}$, the cut set can be denoted by:

$$K = \{\{1\}, \{4\}, \{1, 4\}, \{2, 3\}, \{1, 2, 3\}, \{2, 3, 4\}, \{1, 2, 3, 4\}\} \tag{2.18}$$

It is thanks to this high-level view that RBD-type models are widely used in the representation of systems. However, it is challenging to demonstrate the existence of operating prerequisites between components within a system with them. However, it is only possible to represent independent modules. Thus, for the representation of systems considered more complex, more refined modeling techniques are required, as is SPN and CTMC.

2.3.2 Markov Chain

Markov chains are techniques designed for stochastic modeling. This type of model can describe the system's operation based on a set of states and transitions. These transitions can describe failure and repair events of a system or component.

The Markov chains are more convenient than the RBDs when we want to describe the dynamic properties of the systems (BOLCH et al., 2006). In a Markov chain, each transition represents a stochastic process X(t), which is a set of random variables defined over the same space of probabilities. This set of variables is capable of assuming values in the space of states $S_i \in S$ (CASSANDRAS; LAFORTUNE et al., 2009). Thus, if the set T which variable belongs is discrete, with t = 1,2,3, ..., the process is called a discrete parameter process or discrete-time. However, if T is a continuous set, we have a continuous parameter process or continuous time. Assuming geometric distributions Discrete Time Markov Chain (DTMC) or exponential Continuous Time Markov Chain (CTMC) respectively (NORRIS, 1998).

The stochastic process is classified as a Markov process if, for all $t_0 < t_1 < ... < t_n < t_{n+1}$, and for all $X(t_0)$, $X(t_1)$, $X(t_1)$, $X(t_2)$,...,$X(t_n)$, $X(t_{n+1})$ the conditional distribution of $X(t_{n+1})$ depends only on the last previous value $X(t_n)$ and not on the values that precede it $X(t_0)$, $X(t_1)$,...,$X(t_{n-1})$. This means that for any real number $X(t_0)$, $X(t_1)$, $X(t_1)$, $X(t_2)$,...,$X(t_n)$, $X(t_{n+1})$, $P(X_{n+1} = S_{n+1} \mid X_n = S_n, X_{n-1} = S_{n-1},...,X_0 = S_0) = P(X_{n+1} = S_{n+1} \mid X_n = S_n)$ (BOLCH et al., 2006). This feature is also titled as the absence of memory and is commonly found in distributions of the same type. Just like distributions that resemble exponential or geometric distributions. The absence of memory ensures that the values of the current state in a Markov chain are totally independent of the values of the previous states (TRIVEDI, 1982).

Through a Markov chain, it is possible to perform the behavioral modeling of a system graphically using only states and transitions. Figure 3 shows an example of a Markov chain with only two states: UP and DOWN. In which we have an operant state and another non-operant state of a generic system. Just as we have two arcs: Failure and Repair represent the transition from one state to another.

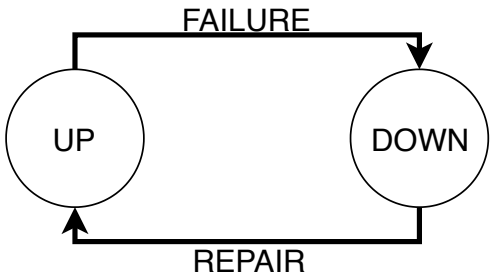


Figure 3 – Markov Chain example

In addition to the graphical representation similar to the finite automata, the Continuous Time Markov Chains (CTMC) is represented by a matrix. This matrix is called

the transition rate matrix Q. In this matrix, details and information describing all states' transitions within a Markov chain are contained. This information is used during the process of solving a Markovian model. The elements located outside the main diagonal represent the occurrence rate of the events that occur in the Markov chain states' transition. The main diagonal elements represent the values necessary for the sum of elements in each line to equal zero, such that:

$$Q = \begin{pmatrix} q_{ii} & q_{ij} \\ q_{ji} & q_{jj} \end{pmatrix} \tag{2.19}$$

The probability of each state provides a steady-state solution based on $\pi.\mathrm{Q} = 0$. Q is the state matrix, and π is the eigenvector corresponding to the unit eigenvalue of the transition matrix, resulting in a null vector. It is important to emphasize that the sum of the probability vector elements π must always be equal to 1, that is,$||\pi|| = 1$ (BOLCH et al., 2006). The state transition probabilities are calculated from Equation 2.20.

$$P_{i,j}(s,t) = PX(t) = j \mid X(s) = i \tag{2.20}$$

For time-dependent, i.e., transient-type solutions, long run times are considered. Thus, it can be shown that the probability of the states converges to constant values (BOLCH et al., 2006). The Markov chain's transient behavior provides us with performance and dependability information on the initial instants of the system (BOLCH et al., 2006). Thanks to these characteristics, the Markov chains have a greater power of representation than the RBDs (BOLCH et al., 2006).

More complex systems with large amounts of states sometimes have their graphical evaluation even more complex. However, it requires the use of approaches that have some type of process automation. Just as it has the same power of representation, these are the SPNs, explained next.

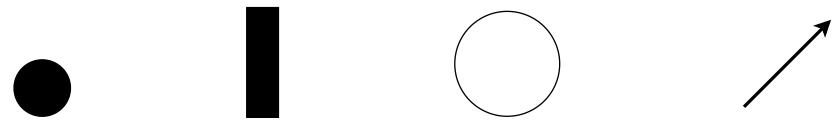
2.3.3 Petri Nets

Carl Adam Petri created the concept of PN in his Ph.D. thesis entitled "Kommunikation mit Automaten" (Communication with Automata) in 1962 (PETRI, 1962). Since then, this formalism has been widely used in different areas, such as Computer Science, Electrical Engineering, Administration, and Chemistry. The PN presents itself as a graphical and mathematical modeling tool that can be applied to solve various characteristics and approaches (MURATA et al., 1989). It allows observing the current state of the system and others possible that the system may be. Several derivations of the classical PN model have been developed over time. In which, stand out the timed (MERLIN; FARBER, 1976), stochastic (MARSAN et al., 1995), and high-level (JENSEN; ROZENBERG, 2012).

According to (GIRAULT; VALK, 2013), PN is a set of formalism that has a graphic representation. They can also provide refinement and abstraction mechanisms of great

importance for the design of complex systems. Besides, several tools are available that facilitate their modeling, analysis, and verification. Resulting in high applicability to the exact sciences. The PNs are a directed graph, where places and transitions are their vertices, interconnected through directed arcs. If an arc's origin is a place, its destination must necessarily be a transition and vice versa. Graphically, PNs are represented by:

- Tokens: represent the state the system is in at a given moment, Figure 4(a);
- Transitions: represent the active elements of the network, that is, the actions performed by the system. For a transition to be enabled, all of its preconditions must be satisfied. However, if a precondition is not satisfied, the transition will be disabled. Once the transition satisfies all the preconditions, this transition may be fired. This action removes a certain amount of tokens of a place and placing it in others, Figure 4(b);
- Places: represent the passive elements of the network. Its primary function is token storage. In which, are removed and added as transitions are fired, Figure 4(c);
- Arcs: represent the flow of tokens through the network. In the graphical representation, transitions and places can be connected by multiple arcs (multi-valued arcs). In which they can be compressed into a single labeled arc, Figure 4(d).



(a) (b) (c) (d)

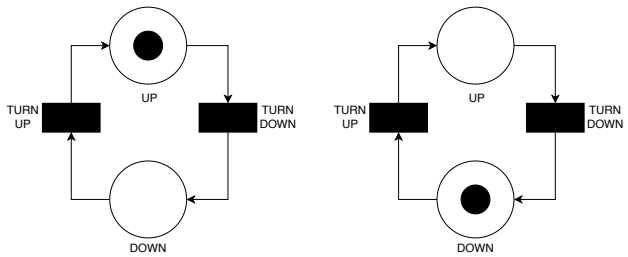
Figure 4 – Petri Net components - **a**) Token. (**b**) Transition. (**c**) Place. (**d**) Arc.

Formally, Petri Nets can be defined as follows. PN = (P, T, F, W, M0), where:

- P $\{P_1, P_2, ..., P_n\}$ is the set of places;
- T $\{T_1, T_2, ..., T_n\}$ is the set of transitions;
- F $\subseteq (P \times T) \cup (P \times T)$ is set of arcs;
- W: F$\rightarrow \{0, 1, 2, 3, ..., n\}$ is the function of weighting the arcs;
- M_0: P$\rightarrow \{0, 1, 2, 3, ..., n\}$ is the initial marking.

Figure 5 shows an example of a PN. In which, the operation of a server is modeled. In this model, places and transitions represent, respectively, server states (up or down). Furthermore, actions that change the server state (turn on or turn off). The current state

of the model is represented by a token in place corresponding to the model's current state, as described in Figure 5(a). Since the current state of the model is working, the only transition that can be fired is to turn off. Once this transition is triggered, then the model will move from the upstate (see Figure 5(a)) to the downstate (see Figure 5(b)).



(a) Server Up. (b) Server Down.

Figure 5 – Petri Net example

2.3.4 Stochastic Petri Net

Stochastic models are techniques for modeling and evaluation of computational systems. As well, they can also be associated with dependability attributes, such as availability and reliability. These models are classified in combinatorial (no state space) and state-space models (GERMAN et al., 2000). Combinatorial models can capture the conditions that lead to a system to fail. The models with state-space represent the behavior of the system based on the occurrence of events. In which, they lead to the execution of failure and repair routines, which can be represented by rates or distribution functions (GERMAN et al., 2000). State-based models allow a more complex representation of systems than combinatorial models. However, they require greater computational power to be evaluated.

In this study, Stochastic Petri Nets (SPN) was adopted to deal with random time. Besides, it can be used for performance and dependability modeling. The time associated with SPNs is associated with transitions that are exponentially distributed on timed transitions and zero on immediate transition (MURATA et al., 1989).

According to to (MARSAN et al., 1995), the execution of activities can be modeled through this type of model. The timed transitions associate times so that their activation period will correspond to the activity execution period. Moreover, the transition firing corresponds to the end. Priorities can be set for each transition to be fired. Furthermore, the immediate transitions have higher priority than timed transitions. The priorities will be paramount when conflict and confusion occur.

The SPNs are defined by (LINDEMANN, 1998) as follows. SPN = (P, T, I, O, H, $\prod$, G, M_0, Atts), where:

- P $\{P_1, P_2, ..., P_n\}$ is the set of places;

- T $\{T_1, T_2, ..., T_n\}$ is the set of transitions;
- I $\in (\mathbb{N}^n \to \mathbb{N})^{n \times m}$ is the matrix representing the multiplicity of input arcs (which may be mark-dependent). Where, the input i_{JK} of I shows the multiplicity of the arc (which may be marking dependent) of the place P_J for the T_K. $[A \subseteq (P \times T) \cup (T \times P)]$;
- O $\in (\mathbb{N}^n \to \mathbb{N})^{n \times m}$ is the matrix representing the multiplicity of output arcs (which may be mark-dependent). Input o_{JK} of O shows the multiplicity of the arc (which may be marking dependent) from place P_J to transition T_K;
- H $\in (\mathbb{N}^n \to \mathbb{N})^{n \times m}$ is the matrix representing the inhibitor arcs (which may be mark-dependent). Input h_{jk} of H shows the multiplicity of the inhibit arc (which may be dependent on marking) from place P_J to transition T_K;
- $\prod \in \mathbb{N}^m$ is the vector that assigns a priority level to each transition;
- G $\in (\mathbb{N}^n \to \{true,\ false\})^m$ is the vector that associates a related guard condition with place marking with each transition;
- $M_0 \in \mathbb{N}^n$ is the vector that associates an initial mark of each place (initial state).
- Atts $= (Dist, W, Markdep, Police, Concurrency)^m$ includes the attribute set for transitions, where:

  Dist $\in \mathbb{N}^m \to f$ is a probability distribution function associated with the time of a transition (this distribution may be mark-dependent) (the domain of f is $[0, \infty)$);

  W $\in \mathbb{N}^m \to \mathbb{R}^+$ is a weight function (this distribution may be mark-dependent);

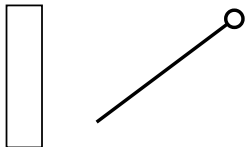
  $Markdep \in \{constant, enabdep\}$ where the probability distribution associated with the time of a transition can be independent (constant) or dependent on the mark (*enabdep* - the distribution depends on the actual enabling condition);

  $Policy \in \{prd, prs\}$ is the memory policy adopted by the transition (*prd* - *preemptive repeat different*, default value, meaning identical to race enabling policy; *prs* - *preemptive resume*, corresponds to age memory policy);

  $Concurrency \in \{ss, is\}$, is the degree of concurrency of transitions, where ss is the single-server semantics and is represents infinity server semantics;

In SPN models, in turn, transitions are fired following the interleaving semantics of actions (GERMAN et al., 2000). This semantics defines that transitions are fired one by one, even if the state comprises non-conflicting immediate transitions.

Figure 6 illustrates the elements of Stochastic Petri Nets (SPNs) that extend to Petri Nets (PNs). The Inhibit arcs are used to prevent the fire of network transitions. Similar to traditional arcs, they can also have a weight associated.



(a) (b)

Figure 6 – Components added with SPN - **a**) Exponential Transitions. (**b**) Inhibit Arcs.

2.4 SENSITIVITY ANALYSIS

The sensitivity analysis consists of studying the effect that the variation of input data can cause in the results. When a small variation in a parameter changes drastically, the model results, the model is said to be very sensitive to this parameter. Therefore, there are many reasons to perform sensitivity analysis, such as finding and eliminating insignificant parameters in the model (SALTELLI et al., 2004). Still, there are several different ways of conducting sensitivity analysis, which may generate different results. At the end of the sensitivity analysis, it is common to construct a ranking with the parameters of decreasing form, where the most sensitive parameter is one that, through its variation, changes the model output to the largest extent.

Since there are several ways to perform the sensitivity analysis, there are several mathematical methods that can be used for this. In this dissertation the percentage difference sensitivity analysis technique is employed, characterized by a sensitivity index known as $S_\theta(Y)$, which indicates the impact of a given measure Y with respect to a parameter θ. Equation 2.21 shows how the percentage difference index is calculated for a metric $Y(\theta)$, where *max* $Y(\theta)$ and *min* $Y(\theta)$ are the maximum and minimum output values, respectively, computed when varying the parameter θ over the range of its n possible values of interest. If $Y(\theta)$is known to vary monotonically, so only the extreme values of θ (i.e., θ_1 and θ_n) may be used to compute *max* $Y(\theta)$; *min* $Y(\theta)$,and subsequently $S_\theta(Y(\theta))$ (HAMBY et al., 1994).

$$S_\theta(Y) = \frac{max\{Y(\theta)\} - min\{Y(\theta)\}}{max\{Y(\theta)\}} \tag{2.21}$$

The $S_\theta(Y)$ computation is done while the values of the model parameters are fixed, carried out for each input parameter until all parameters have been analyzed, and the one with the most significant impact be found.

2.5 SOLAR POWER SYSTEM

The solar power system directly supplies appliances that will use energy and are generally built for a specific local purpose. There are two basic types concerning solar energy systems, on-grid (grid-connected) and off-grid (Stand-alone) (REINDERS et al., 2017). The

off-grid solar power is widely used in remote locations as it is often the most economical and practical way to get electricity in these places. This kind of system is generally composed of a solar panel or panel array, a charge controller, a battery or battery pack, and an inverter (O'CONNOR, 2016). The on-grid solar power is generally composed of an inverter, solar panels, string box, and AC isolator. This system kind is used in places that already have electricity with the purpose to supply part of the energy that will be consumed (VILLALVA, 2012). There is also the solar hybrid system, which connects one or more energy sources besides solar energy (REINDERS et al., 2017). This work compares a hybrid solar power and an on-grid solar power system. In addition, the methodologies described in (VILLALVA, 2012) were adopted for the sizing of the solar systems present in this work.

The use of batteries in an off-grid or hybrid system is common. Therefore, in this type of system, the first step is determine the size the battery bank. The batteries used in the off-grid and the hybrid solar power systems store energy generated by solar panels. For determining the number of batteries of the system, it is necessary to know the amount of energy that should be stored (Wh - Watt-hour); besides, the system output voltage should be defined. According to (VILLALVA, 2012) the stored energy can be calculated through the Equation 2.22.

$$E_S = \frac{E_C}{D_D}, \tag{2.22}$$

where E_C is the energy consumed (Wh) daily, and D_D is the discharge depth allowed (%). The next step is to define the batteries number connected in serial. If we intend to calculate the batteries number connected in serial, we use the Equation 2.23 (VILLALVA, 2012).

$$N_{BS} = \frac{S_V}{B_V}, \tag{2.23}$$

considering that the S_V is the output voltage of the system (V - Volt), and B_V is the output voltage of a battery (V).

However, it is also needed to determine the batterie´s number connected in parallel. The battery bank load capacity will be used to determines the number of batteries connected in parallel (Ah - Ampere hour). It can be determined through the Equation 2.24 (VILLALVA, 2012).

$$C_{BS} = \frac{E_S}{V_B}, \tag{2.24}$$

the V_B is the output voltage of the battery bank (V). On the other hand, the number of batteries connected in parallel can be calculated through the Equation 2.25 (VILLALVA,

2012).

$$N_{BP} = \frac{C_{BS}}{C_B}, \tag{2.25}$$

C_B is the battery load capacity (Ah).

The photovoltaic panels are components used in all kinds of solar power systems. They can convert energy directly solar on electrical energy. The photovoltaic panels are components used in all kinds of solar power systems. They can convert directly energy solar on electrical energy (FAHRENBRUCH; BUBE, 2012). Therefore, the next step is to determine the number of panels needed to generate the energy intend to consume. To know the number of solar panels to generate required energy, the designer can use two methods, daily heat stroke or maximum module current (VILLALVA, 2012). This work considers only the daily heat stroke method because we adopted the charge controller with MPPT (Maximum Power Point Tracking). Besides, the on-grid system always has an inverter with the MPPT system. Equation 2.26 (VILLALVA, 2012) shows how it is calculated the produced energy (Wh) daily by one solar panel in this method.

$$E_P = E_s \times A_m \times \eta_m, \tag{2.26}$$

where, E_s is daily heat stroke[Wh/m²/daily]. A_m is module area (m²). Finally, η_m is module efficiency (%).

For estimating the number of batteries required, the system needs to know the amount of energy that should be stored. Besides, the voltage of the system that can be 12V, 24V, or 48V should be defined. The stored energy can be calculated through Equation 2.22.

The maximum module current method considers that it is impossible to have the maximum use of solar energy. Then, energy production is limited by the operating point imposed by the batteries. To use this method is needed to use some information available by the manufacturer in STC (Standard Test Condition) or NOCT (Nominal Operating Cell Temperature) (VILLALVA, 2012). The use of NOCT is most appropriate in this case because it reflects approximately the actual conditions of operation. The use of STC can result in the value above the obtained under real conditions (VILLALVA, 2012).

According to to (VILLALVA, 2012), all photovoltaic system with batteries should employ a charge controller or charge regulator. The charge controller is responsible for connecting the panels photovoltaic and the battery correctly. Thereby, it avoids that the battery is overcharged or discharge overly. The charge controllers are common with PWM (Pulse Width Modulation) or MPPT (Maximum Power Point Tracking). The charge controller's choice depends on two parameters: operating voltage and maximum current provided by the set of solar panels. The maximum current can be calculated through Equation 2.27 (VILLALVA, 2012).

$$C_M = I_{SC} \times P_P \times F_S, \tag{2.27}$$

where I_{SC} is the module short circuit current (A). P_P is the amount of the panels connected in parallel. Finally, the F_S is the security factor (%) used to ensure that the maximum current will not be exceeded under any circumstances (VILLALVA, 2012).

The inverter is the electrical equipment able to convert continuous current to alternating current (AC) (ŞEN, 2008). There are three kinds of inverters. The square wave and modified sine wave inverters are equipment in which produce output voltage with of semi-square waveform. This kind of inverter is used to provide energy to equipment, not sensible the voltage distortion (SUKHATME; NAYAK, 2017). This kind of inverter is used to provide energy to equipment, not sensible the voltage distortion (TIWARI; TIWARI et al., 2017). The lamps and home appliances are examples of this kind of equipment. To critical systems as hospitals and telecommunication system is used pure sine wave inverter. This kind of inverter produces output voltage with an almost perfect sine wave format. The inverter is the component needed in the solar power system and is chosen according to input and output voltage. It should also supply the total power of the equipment to be fed (SUKHATME; NAYAK, 2017). According to to (ŞEN, 2008), all on-grid solar power system should employ a String box and an AC isolator. String box is protective equipment that isolates the photovoltaic energy production system to prevent the risk of spreading electrical accidents, such as short circuits and electrical surges. The AC isolator is the equipment responsible for connecting the inverter and electrical network safely (VILLALVA, 2012). This equipment, as well as the string box, are protection components for on-grid solar power systems.

2.6 FINAL REMARKS

This chapter provided theoretical foundations that are not exhaustive but essential for the reader's awareness regarding the building blocks that compose this dissertation. The background on stochastic models, the Internet of things, sensitivity analysis, and dependability allows understanding the proposal of this dissertation. We also explained the autonomous photovoltaic system, which will be used further in a case study.

3 RELATED WORKS

This chapter presents related works about dependability in cloud computing, the internet of things, and photovoltaic systems. All these works represents a base to evaluate our work about dependability in a smart building. Concerning related work, we have determined four different ways to compare our work with other areas' achievements. First of all, we compare which group the proposals belong (e.g., cloud computing, internet of thing, and photovoltaic system). Second of all, which work has some redundancy. Third, which work use models to evaluate the dependability. Finally, we compared which dependability metric the authors were interested in (e.g., availability (A), reliability (R)). In this section, some works related to our study are introduced. This section aims to provide an overview of the published works on the topic in question, so at the end of this section, it is possible to find an effective conclusion showing the relevance of the work proposed here, considering the others presented.

3.1 DEPENDABILITY ON INTERNET OF THINGS

Some works have been proposing a dependability approach in IoT systems. In (MACEDO; GUEDES; SILVA, 2014) the authors proposed some redundancy models based on Markov Chains to evaluate availability and reliability in IoT applications, aiming only at the aspects of connectivity between the devices. The models proposed in their work address redundancy concepts such as perfect exchange, not perfect exchange, and standby failure. The authors relate that the model is proposed initially for the network devices, however, the idea can be adapted to evaluate scenarios more complex assuming any routing strategy and topology types. From these mathematical models is possible to analyze the implications of each type or the amount of redundancy enough to obtain a desired dependability requirements. The results showed the advantages to integrate redundancy aspects for the system considered availability. In this aspect, the passive approaches presented results more satisfactory when compared with the active approaches. According to the authors, this occurs because the components in the active redundancy operate in parallel and thus are more likely to fail.

The work proposed in (KAMYOD, 2018) aims to evaluate reliability in a smart agriculture system. The studied IoT communication architectures can be applied to a system to collect and process agricultural data for automatically control the usage of water in a small and medium-sized farm. In order to perform such an evaluation, the Riverbed Optimized Network Engineering Tools (OPNET) was employed considering two different computational visions, with and without using a cloud computing infrastructure for data storage, as well as communication using Wi-Fi and Ethernet. The reliability effects were

simulated and compared at a different number of sensor nodes. The results showed that when increasing the number of sensor nodes, the end to end reliability of the entire system is affected at the beginning of the operating period especially for the IoT communication type II which involve longer communication path and devices. Besides, was observed a delay in the communication path between sensor nodes and the Wi-Fi router. The authors suggests improve the reliability and performance of the system increasing the performance of the first transmission gateway between the sensor nodes and the server. In addition, apply redundancy to devices, load balance or increasing the transmission bandwidth can directly reduce the re-transmission rate and increase the system reliability.

Other related work is presented in (LI et al., 2012), where the authors proposed to formally analyze the reliability and aspects that define the cost of composing an IoT system with Service-Oriented Computing (SOC). For this, Finite State Machines (FSM) (WAGNER et al., 2006) and Markov Decision Process (MDP) models (PUTERMAN, 2014) were used to represent the composition of services and to specify the reliability of operations, respectively. In resume the authors related that the probabilistic model (MDP) checking can be applied to verify and analyze the quality properties of the service composition. This approach, the success probability of the service can be calculated with the current IoT settings of the devices. The authors also noted that real-time constraint is another key point in IoT, since the devices in IoT provide and consume real-time data about the real world. The results presented in this work can aid in decision making, alerting when it will be necessary to deploy a spare candidate service because of the reliability of the selected primary service is not high enough.

Work with similar aspects to our proposed approach is presented in (ANDRADE; NOGUEIRA, 2018). The authors adopted Stochastic Petri Nets (SPN) models to represent a solution for disaster recovery (DR) in IoT systems and had a smart healthcare system case study. The models enabled the authors to calculate metrics such as availability and downtime. The SPN used can represent a disaster recovery solution, various failure-recovery behaviors, distinct network communication, dependency between different components, and data replication. The results presented by authors indicated the disaster recovery solution considerably reduces the annual downtime (more than 4h) and also the number of lost requests (more than 21%). The sensitivity analysis also performed, the results showed that the majority of the parameters ranked in the highest positions of the sensitivity ranking are related to the front-end components. A significant reduction in the downtime was observed from the inclusion of a redundant component and the adoption of efficient fail over and fail back mechanisms. The results unveiled a reduction in the downtime from 17.84 to 11.91h.

In (KHARCHENKO et al., 2017), the authors developed and researched Markov models and assessing the availability of Instrumentation and control systems which are a part of the smart building automation system (BAS). The sets of faults and vulnerabil-

ities are considered as separate and disjoint ones. Markov models of BAS architecture with occurred software faults and attacked vulnerabilities considering three maintenance strategies are systematized and researched. These strategies are based on recovery without maintenance, maintenance with common, and separate activities on reliability and vulnerabilities. The architecture of these strategies depends on the kind of maintenance (common or separate). The result of these strategies gives different ways for recovering the system taking into account the customer requirements as the maximum value of availability during the minimum time.

3.2 DEPENDABILITY ON CLOUD COMPUTING

There are more work in literature related with dependability evaluation in cloud computing than dependability in IoT. In (MATOS et al., 2017), the authors proposed the availability evaluation of warm-standby redundancy for private cloud, in which they used as a case study the Eucalyptus framework (DANTAS et al., 2015). The availability evaluation for redundant private clouds, was represented by RBDs and Markov chains, hierarchically assembled. Also, the work proposed an approach for performing parametric sensitivity analysis on hierarchical models comprising RBD and CTMC models. The results presented by authors showed that the Cloud Manager Subsystem (composed of Eucalyptus Cloud Controller and Walrus components) is the "availability bottleneck" for the analyzed architectures, despite the existence of a spare warm-standby Cloud Manager. Besides that, in availability aspects the non-exponential distributions has a small but significant effect on results, when comparing to the model with only exponential rates. The steady-state availability measure changes from 0.99996937 (analytical model) to 0.99997974(simulation model)

Another study proposed in (MELO et al., 2018), aims to evaluate availability in a cloud computing framework using the hyper-converged paradigm. The hyper-convergence on OpenStack cloud computing platform utilizes a distributed storage mechanism to reduce the expenses with storage device. The work analyzes the availability of different structures, such as structures without redundancy mechanism, double redundancy, triple redundancy, and hyper-converged. The comparison was made considering all of these case studies, and the triple redundancy structure obtained the highest percentage of system availability. To performed the evaluate was used a Video on Demand (VoD) system. The evaluation results of four different architectures show that is better to provide a hyper-convergent infrastructure than a full double redundant one,even with the same amount of equipment.The annual downtime of hyper-convergent and double redundant differs of more than sixty times.

In (GHOSH et al., 2014), the authors performed availability analyzes on a cloud infrastructure, where physical machines are grouped into three types of resource provisioning sets: hot, warm, and cold. This division was made based on energy consumption and the

characteristics of waiting in the provisioning of resources. Thus, a stochastic modeling approach is used to analyze these systems' dependability in the IaaS provisioning cloud in the three modes mentioned above, through a Markov chain. The authors related that work provided some contributions as; state the availability assessment problem for an IaaS cloud, an interacting sub-models approach to solve the largeness problem of the monolithic availability model, and the overall model solution is obtained by fixed-point iteration over individual sub-model solutions. The results presented by authors showed how scalability issues for a monolithic model can be resolved by means of interacting sub-models or by means of simulation. Besides, interacting sub-models approach quickly provides model solutions facilitating scalability without significantly compromising the accuracy. Simulation provides results that closely match with monolithic and interacting sub-models approaches and, for large systems, results are obtained faster.

In (AZAIEZ; CHAINBI; GHEDIRA, 2019) the focus is on the availability attribute of dependability's concept and its evaluation in a Cloud simulation environment. The proposed solution was a hybrid fault tolerance strategy that combines checkpoint and replication strategies to ensure Cloud resources availability. The experiments were performed although the CloudSim, most used simulator in the Cloud computing field. The first step of experiment was evaluate the virtual machine fault tolerance. In this aspect the authors tried in every simulation to increase the total number of Cloudlets submitted to Cloud. After a fault injection into one of primary VMs during the execution of Cloudlets, the reaction of each strategy has been observed. The second experiment was evaluate the physical machine (PM) fault tolerance. In this experiment, the authors tried in every simulation to increase the number of failure injection in different PMs that host a number of VMs running Cloudlets. The evaluation results have shown that the presented strategy allows, on the one hand, to ensure availability and reliability of Cloud applications and, on the other hand, decrease Cloud infrastructure workload and execution time.

In (NAIK, 2016) was proposed an intuitive approach based on Computational Intelligence (CI) for enhancing its dependability. The proposed CI-based approach predicts the host of a manager node's possible failure by observing its abnormal behavior. Thus, this indication can automatically trigger the process of creating a new manager node or promoting an existing node as a manager for enhancing the dependability of Docker Swarm. To evaluate the availability of a multi-cloud system, the authors proposed a simulation based on docker swarm, VirtualBox, and Mac OS X. Finally, the authors proposed a CI-based solution for enhancing the dependability of a multi-cloud system using Docker Swarm. The results showed that the proposed CI-based approach is promising to assess and control the external failure and maintain Docker Swarm cluster availability. However, it is a preliminary model and needs further expansion and testing to determine this approach's effectiveness.

In (ABOHAMAMA; ALRAHMAWY; ELSOUD, 2018) presents a framework that employs

different fault tolerance techniques, including preemptive migration, checkpoint/restart, and replication, for improving the dependability of public cloud computing environments to host real-time applications reliably. As a part of the framework, a replication-based fault-tolerant soft real-time scheduling algorithm (FTRTS) has been designed and implemented to minimize the rate of missing deadlines, makespan, and load imbalance. The performance of the algorithm has been evaluated through several experiments and compared to other standard scheduling algorithms. The results demonstrated that FTRTS outperforms all other algorithms, especially regarding Harvest Time Missing Rate. Additionally, FTRTS achieved a good compromise among all design objectives compared to other scheduling algorithms.

In (Dantas et al., 2012), the authors investigated the benefits of a warm standby redundancy mechanism in a cloud storage environment on the Eucalyptus platform. A hierarchical and heterogeneous modeling approach based on reliability block diagrams and Markov chains to represent as basic and redundant architectures. Both hardware and software failures are considered in our analytical models, which are also used to obtain closed-form equations for computing the availability of the cloud infrastructure. It is also noted that the authors compare the availability of the two architectures. The results showed an enhanced dependability of the proposed redundant system, evidenced by the growth from 2 to 4 nines in the availability, as well as a decrease in the annual downtime from 46 hours to only 43 minutes. The results also demonstrate that the simple replacement of hardware by more reliable machines will not produce such a big improvement in system availability.

In (NGUYEN et al., 2019), the authors proposed a hierarchical modeling framework for reliability and availability evaluation of tree-based data center networks. Two representative data center networks based on three-tier and fat-tree topologies are modeled and analyzed in a comprehensive manner. In resume, the authors list some contributions as; a three-layer hierarchical modeling frame-work specifically for the availability/reliability evaluation, hierarchical and heterogeneous models to not only capture the overall network topologies, and comprehensive analyses and evaluation of different metrics of interest including reliability and availability. The results presented by authors showed The that the distribution of active nodes in the network can enhance the availability/reliability of cloud computing systems. Furthermore, the MTTF and MTTR of physical servers are the major impacting factors, whereas those of links are important in maintaining high availability for the system.

3.3 DEPENDABILITY ON PHOTOVOLTAIC SYSTEM

In (SONG et al., 2012a), the authors proposed to investigate the effect of energy storage systems on the availability of micro-grids with different architectures in the presence of renewable energy sources. The proposed method uses a Markov chain model to properly

model the charging and discharging processes in the energy storage system, which makes it possible to understand how renewable energy sources and energy storage function together to provide power to the load and affect micro grids availability. This model also enables calculating the effect of an energy storage system connected to a micro-grid with multiple energy sources and loads. The results are presented using actual solar PV generation and load profile data. Simulation results show that the proposed method provides a more realistic estimation of the system availability compared to the two-state model which may give overestimated availability values.

In (KWASINSKI et al., 2012), the authors discussed how micro-grids availability during natural disasters and in their aftermath can be assessed. The analysis focuses on two critical groups of components that allow micro-grids to improve power supply availability: distributed generators and local energy storage. Analysis of micro-grids availability is performed based on Markov state-space models and calculated using minimal cut sets approximations. A key contribution of this paper is to focus on representing the effect of two critical aspects affecting micro-grid availability during natural disasters and in their aftermath: lifelines performance and local energy storage contribution. The results presented by authors showed that theoretical calculations confirmed with Monte Carlo simulations seem to show that in the context of natural disasters micro-grids may achieve availability much higher than conventional grids, making micro-grids a prime option for developing advanced smart grids.

In (SONG et al., 2012b), the authors proposed a Markov-chain-based energy storage model to evaluate the power supply availability of photovoltaic generation. The proposed work models energy states in a photovoltaic-energy storage system in order to understand the nature of charge/discharge rates for energy storage that affect the system's power output. The authors related that the developed model allows PV generation to become more dispatchable with optimal-sized energy storage. Hence, the proposed model can be used for a PV-energy storage system not only to meet the certain availability but to avoid over-sizing or under-sizing capacity, which leads to increased system cost or inadequate availability, respectively. Furthermore, the proposed model provides a better understanding of how energy storage charge and discharge rates are expected to occur in a PV-energy storage system and affect the power supply availability of PV generation plants.

In (CHARKI et al., 2017), the authors proposed a multi-level methodology to estimate the availability and the lifetime of a PV system using stochastic Petri networks and taking into account the time distribution to failure and repair. The following components – module, wires, and inverter – are modeled with a Petri network. The evolution of the MTBF, availability, and outages over a 30-year period is simulated for different series/parallel configurations of connected PV modules. The studied system is a grid-connected photovoltaic system with modules connected in a series-parallel configuration and balance-of-system

(BOS) components which are not photovoltaic: an inverter which transforms the direct current from the photovoltaic array to alternating current, wires which transmit the continuous energy from PV modules to an inverter (DC wires) and a wire which transmits the alternating energy from the inverter to the electric grid (AC wire). The authors related that the influence of the number of modules in series on the availability estimation is not significant. However, the mean time between fail-ures decreases with an increasing number of series. The proposed methodology can be used for optimizing the performance of the installation of a PV system. The influence of each failure mode can be analyzed in order to improve each component of the PV system.

In (ZINI; MANGEANT; MERTEN, 2011) was presented as a method to analyze and quantify the reliability of large scale grid-connected photovoltaic systems. This methodology is based on fault tree analysis using an exponential distribution to model the fault probability density function. The main results presented considered one year of system operation. It was detected that for a 100 kW_p system with 23 string protections (considered a single subsystem) have 97.79% probabilities of functioning without failures, while the inverter only 88.25%. However, for a 2.5 MW_p system with 572 string protections (again considered a single subsystem) have 57.36% probabilities of functioning without failures, while the 24 inverters only 4.98%. If considering twenty years of operations, the reliability estimates drop radically. For a 2.5 MW_p system, faults will occur with more than 99% probability to the photovoltaic modules, string protections, inverters, and AC circuit breakers.

3.3.1 Analysis of related works

Analyzing the related works, we can observe that dependability is extensively studied in the literature. Several dependability strategies are proposed to increase the availability, reliability, and other metrics. Comparing the related works to our proposal, we can observe that our proposal encompasses several subsystems, while related works focus on only one. Table 1 shows a comparison between the most relevant related works and our proposal. We also verified if these works were used some analytical model to evaluate the dependability metrics. Besides, if the work proposed a sensitivity analysis of the system to identify system bottlenecks. Finally, we verified which dependability metrics the authors used in your works. It is worth mentioning that columns A means Availability, R means Reliability, and C means the cost.

This study is relevant because it compares different infrastructures to evaluate availability in entire IoT systems, whereas there are a few works that evaluate dependability in the IoT system. Besides, the studies presented do not evaluate all subsystems that can compose the entire IoT system. This dissertation focuses on assisting smart building designers in achieving a high availability rate and preventing information loss in the entire system. It also avoids other unintended consequences that only tend to occur when the service is unavailable. Some different management infrastructures are presented as the

Table 1 – The most relevant related works comparison.

	A. Models	Sensitivity Analysis	Metric
Macedo, Guedes e Silva (2014)	CTMC	No	A/R
Kamyod (2018)	No	No	R
Li et al. (2012)	CTMC	No	R/C
Andrade e Nogueira (2018)	SPN	Yes	A/C
Kharchenko et al. (2017)	CTMC	No	A
Matos et al. (2017)	CTMC/SPN/RBD	Yes	A
Melo et al. (2018)	SPN	No	A
Ghosh et al. (2014)	CTMC/SPN	No	A
Azaiez, Chainbi e Ghedira (2019)	No	No	R
Naik (2016)	No	No	A
Song et al. (2012a)	CTMC	No	A
Kwasinski et al. (2012)	CTMC	No	A
Charki et al. (2017)	SPN	No	A
Proposal Work	SPN/RBD	Yes	A/C

base of our proposal to IoT generic system: local no redundancy, local with redundancy, and cloud computing with redundancy. The management infrastructures used were chosen based on related works. The local management infrastructure presented in this dissertation has the highest availability levels when considered one building. Besides, it has the lowest deployment cost when it is compared with cloud computing infrastructure. Therefore, this proposal used local redundancy infrastructure in a smart photovoltaic system to have the best cost-benefit.

3.4 FINAL REMARKS

In this chapter, it was presented the most relevant related works for our proposal. However, there are many studies in the literature about the dependability mechanism, most of them not being applied in environments internet of things. It may be a problem because IoT comprises heterogeneous components, the creation of these systems, and the communication and maintenance of their components, becomes a complex task. It consists in the lack of cost-benefit analysis of the same systems considering several different scenarios. So this dissertation presents a dependability study for smart building environments, in which it was considered several sceneries to evaluate the cost-benefit of the system. This study aims at supporting designers to plan, build, manage, and maintain intelligent building systems by taking better advantage of available resources.

4 SMART BUILDING SYSTEMS

This chapter describes the proposed architectures for smart buildings. This chapter is divided into two sections. The first section presents a set of systems and a generic architecture to a smart building. This architecture is used to evaluate three distinct management infrastructure in different scenarios. Besides, it is used to identify the components that most affect the system's availability. The second section presents the set of components and architecture to the smart photovoltaic system. We consider a smart building and different types of photovoltaic systems. This architecture is used to evaluate the impact of adopting the photovoltaic system in a smart building.

4.1 SMART BUILDING

Smart buildings use technology to share information about what is happening in the building and then optimize construction performance. In this type of environment, each system that makes up smart buildings has a unique purpose. All data obtained through these systems are sent to a management system. This management system can make decisions in real-time, periodically, or with the help of humans. In this respect, we can subdivide the systems that make up the building into security, safety, comfort, and others, as shown in Figure 7 below.

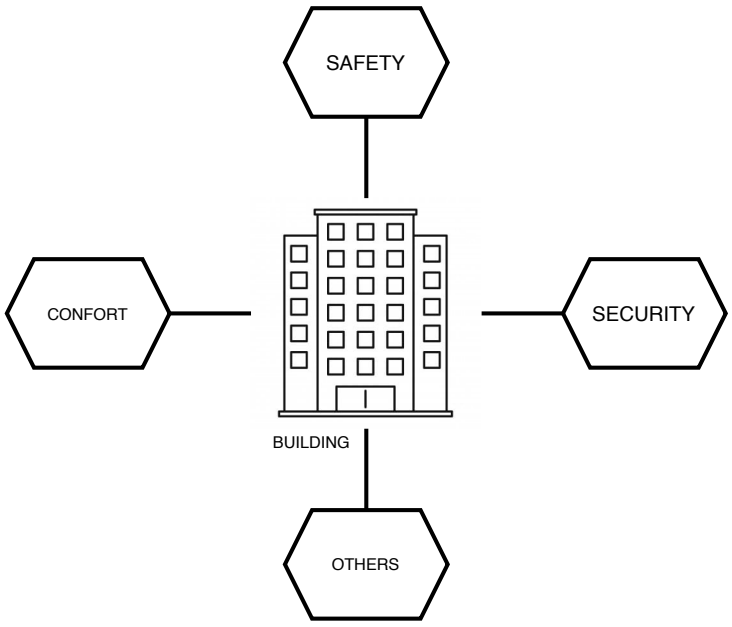


Figure 7 – Systems that compose a smart building

Protection systems may, for example, deal with fire detection systems and sprinkler control. As well as public address systems in which it consists of microphones, amplifiers, speakers, and related equipment, to increase the volume of the human voice or sound

equipment. This type of system is widely used in public buildings. Security systems may control magnetic cards, digital, voice, CCTV (Closed-Circuit Television), intruder detection, and intercom systems are also contained in the security aspect.

The systems in the scope of comfort aim to increase the occupants' level of satisfaction. Considering such an aspect, we may have several types of systems. Examples of such systems are, for instance, luminosity control system, elevators, power metering, and electricity control, HVAC (Heating, Ventilating, and Air Conditioning), which controls the temperature of the environment. In industrial environments, it is ubiquitous to need a PLC (Programmable Logic Controller) system, which performs automation, control, and monitoring functions of machines and industrial processes of different types and complexity levels. Besides, we may also consider many other classes of systems.

4.1.1 Generic Smart Building

This work presents a generic architecture for a smart build. This architecture can help us better understand the main components and how they work. Also, this architecture serves to identify possible bottlenecks in the system. As well as to identify better strategies for different scenarios in this context. Figure 8 shows the common elements found in this type of system.

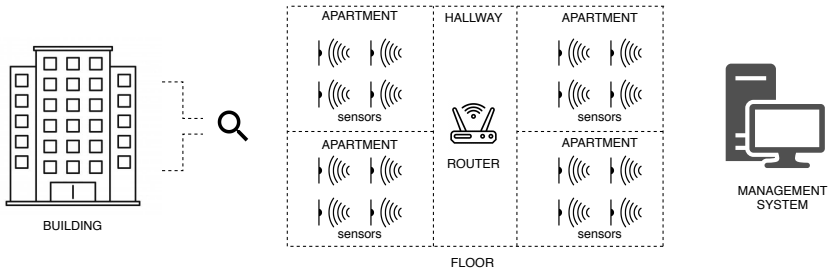


Figure 8 – Generic smart building

Figure 8 shows data transferring between components of the system. The sensors are responsible for capturing data from the environment for the proposed system. As this architecture is generic, the sensors represent any of the systems mentioned above. Therefore, these sensors can be an intruder detection system or an HVAC system. The studies applied to this architecture can be extended to any system contained in the same environment.

For object communication, we combine the Many-to-One and One-to-Many paradigm (TAO et al., 2014). The Many-to-One paradigm is the most common in IoT systems. However, to perform the data collection, it is necessary to create a routing tree, where the root would be the base station, in this case, the management system. The One-to-Many paradigm is known as data dissemination, in which base stations commonly send commands to one or more network objects. Therefore, with the combination of these

paradigms, it is possible having confirmation about the data sent since there is bidirectional communication between the base and the nodes. The communication between the system components is shown in Figure 9.

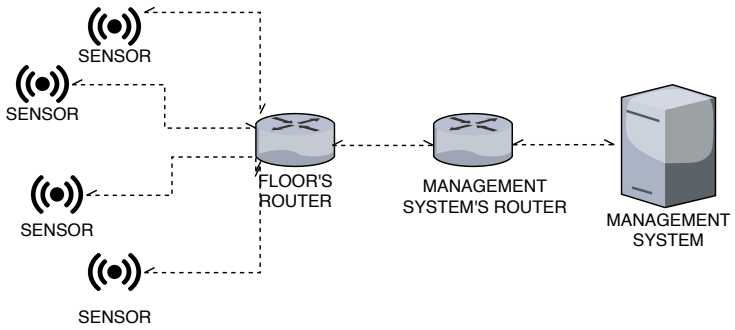


Figure 9 – Communication between components

It is worth mentioning that communication between the devices and the management system is provided through Wi-Fi communication, a fairly common wireless solution. Defined by the IEEE 802.11 standard (BIANCHI, 2000). This standard has already undergone several updates; nowadays, the latest commercial (or in-production) version is the IEEE 802.11ac that defines transmission rates of 600 Mbps or 1300 Mbps (ONG et al., 2011). This type of communication has a drawback for IoT devices because it consumes a considerable amount of energy. Usually, communication technologies that have low power consumption, for example, ZigBee(HAN; LIM, 2010) is preferred. Considering that the devices in question have little capacity for energy storage, Wi-Fi technology was chosen because, in our Case Study, the controlled environment is directly connected to the building's power-line so that the energy consumption factor can be neglected in the analysis.

4.1.2 Proposed generic Architecture to smart building

In this section, a generic infrastructure to smart housing building is proposed. We divide the architecture of the proposed system into two aspects. The first aspect is composed of devices responsible for capturing data from the environment and the components responsible for transmitting this data collected to a management center. The second contains the infrastructure responsible for storing and managing the data and managing those devices.

For object communication, we consider the Many-to-One paradigm (TAO et al., 2014); it is the most common paradigm in IoT systems. However, to perform the data collection, it is necessary to create a routing tree, where the root would be the base station, in this case, the router on the floor. The base station sends the collected data to the router present in the management structure. Thus, the router of the management structure sends the received data to the management center. Therefore, it is impossible to confirm the data sent in this paradigm since there is no bidirectional communication between the base and the nodes. The communication between the system components is shown in Figure 9.

We considered, at first, a local management structure as sufficient for the application. Afterward, we included a redundant management structure to improve system availability. Finally, we adopted tackling interoperability, scalability, availability, and maintainability.

Figure 10 depicts a single floor of the building, which has a gas sensor, smoke sensor, motion sensor. These sensors aim to ensure the residents' safety, preventing them from fires and explosions, and reducing the building's energy costs. All sensor data will be sent to a router, which will send the received data to a management structure.

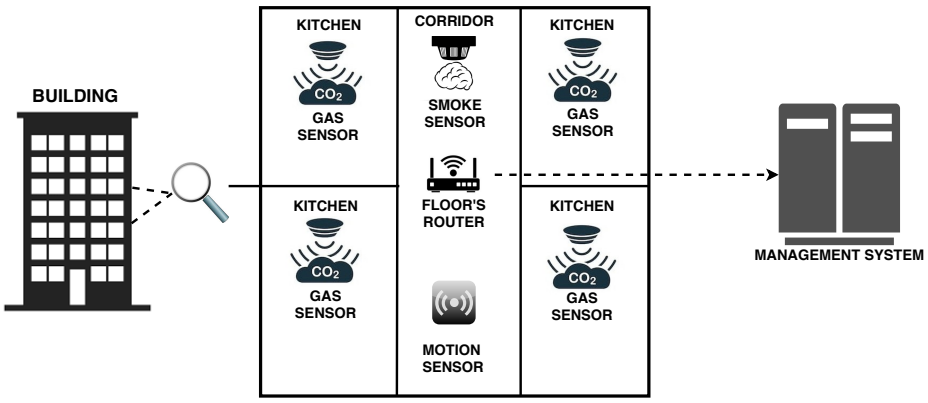


Figure 10 – Proposed IoT environment

Figure 23 shows the basic infrastructure that receives data from the infrastructure mentioned in Figure 10. This local infrastructure contains the minimum requirements for the proposed system; hence it has few mechanisms to recover from failures. Thus, the system has limited components and resources, and the management can only be done locally. Accordingly, in order to obtain availability metrics, we have considered five components, router, hardware, operating system, device manager, and database. The device manager is an application responsible for managing each device as well as visualizing the data analytically, e.g., Thingsboard (CHEBOTKO; KASHLEV; LU, 2015). The database is an application where all data coming from intelligent devices, e.g., Cassandra database (CADAVID et al., 2018) is stored.

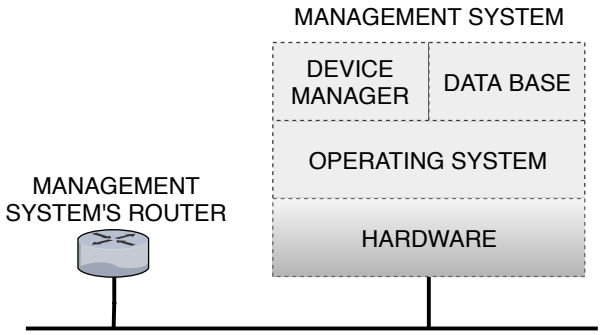


Figure 11 – Local management infrastructure

Another scenario is depicted in Figure 12, which shows the necessary infrastructure with redundancy that receives data from the infrastructure mentioned in Figure 10. How-

ever, to guarantee greater availability, a cold standby redundancy system was adopted, so the secondary machine acts as a backup of the main machine. Therefore, the necessary operating system and software environment setup will only be made after the first machine's primary break down.

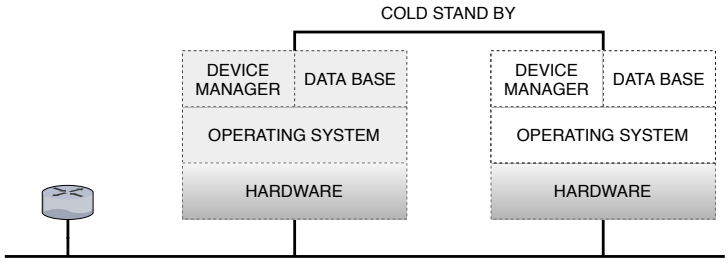


Figure 12 – Local management infrastructure with redundancy

However, if we consider multiple buildings, the replication of the local infrastructure becomes a costly activity. As well as, for interoperability and maintainability, keeping several management infrastructures becomes a complex task. Therefore, we adopted another scenario where the structure is shown in Figure 10 sends data to a management structure based on cloud computing shown in Figure 13. This new structure can consider numerous smart buildings, from different organizations, geographically separated, and managed by a single organization. However, just as in the local management structure, cloud computing infrastructure also contains cold standby redundancy to increase its availability. The cloud infrastructure is composed of least two hosts: one host includes the components already mentioned previously, and also has docker. Docker is a containerization technology for creating and using containers, which act as modular and lightweight virtual machines. The other host is a manager or controller, depending on the specific cloud framework nomenclature responsible for managing all virtualized servers. Furthermore, finally, we have a router responsible for being the gateway between the servers and the outside world; it is through it that the data will be transmitted to the servers.

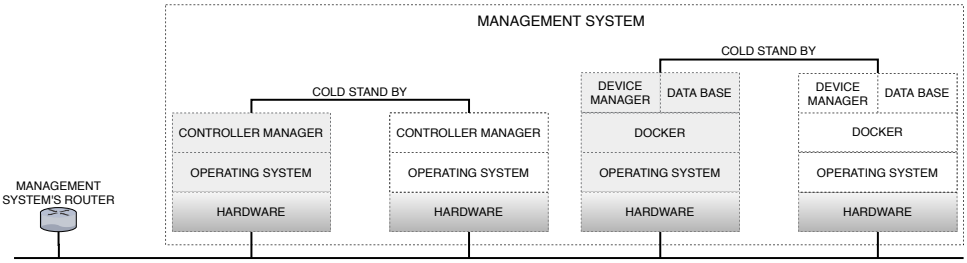


Figure 13 – Cloud computing management infrastructure

4.1.3 Availability and Relibility Related Metrics

This section describes the proposed models used for steady-state availability evaluation. Stochastic Petri Net was used to represent the relationship between the infrastructure

components, both of them previously presented in Section 4.1.2. Mercury tool was adopted for SPN modeling and sensitivity analysis (SILVA et al., 2013).

4.1.3.1 Availability for a single floor IoT environment

This Section describes the proposed availability model. Figure 10 represents the environment proposed, which comprises some components. The GAS component represents the gas sensor located in each apartment's kitchen on a single floor. The MOTION component represents the sensor of motion, which is in the hallway on each building floor. The SMOKE component represents the smoke sensor, which is also on each floor. Lastly, the ROUTER component represents the router, which receives data from all sensors and sends it to the management structure. The floor subsystem's availability, which comprises the mentioned seven components, is obtained through Equation 4.1.

$$A_f = \frac{MTTF}{MTTF + MTTR} \tag{4.1}$$

We consider that the $MTTF$ and $MTTR$ (mean time to failure and mean time to repair the floor subsystem) are exponentially distributed. Therefore, the $MTTF$ can be calculated through Equation 4.2. Moreover, the system's $MTTR$ can be obtained from the reported data of the repair team.

$$MTTF = \int_0^\infty R(t)dt \tag{4.2}$$

Reliability can be calculated by adopting Equation 4.3, where λ_{eq} is of the subsystem the failure rate. In the case of exponential time to failures and a non-redundant system, λ_{eq} can be calculated by summing up all failure rates, as shown by Equation 4.4. Where λ_{GAS}, λ_{MOTION}, λ_{SMOKE}, and λ_{ROUTER} are the failure rate of the gas, motion, and smoke sensors and router respectively.

$$R(t) = e^{\lambda_{eq}t} \tag{4.3}$$

$$\lambda_{eq} = (X \times \lambda_{GAS}) + (Y \times \lambda_{MOTION}) + (Z \times \lambda_{SMOKE}) + \lambda_{ROUTER} \tag{4.4}$$

where X represents the number of gas sensors, Y the number of motion sensors, and Z the number of smoke sensors.

4.1.3.2 SPN Availability Model for Entire System

This section describes the proposed availability model for the entire system considering a local management infrastructure. A Stochastic Petri Net (SPN), depicted in Figure 14, was built to represent the management system, IoT environment, and the router responsible

for sending the environment data to the management system. The SPN model was used because it allows transitions with values exponentially distributed, which is necessary to represent the proposed system accurately. Besides, it enables the representation of system behavior and evaluation of required metrics only and concisely.

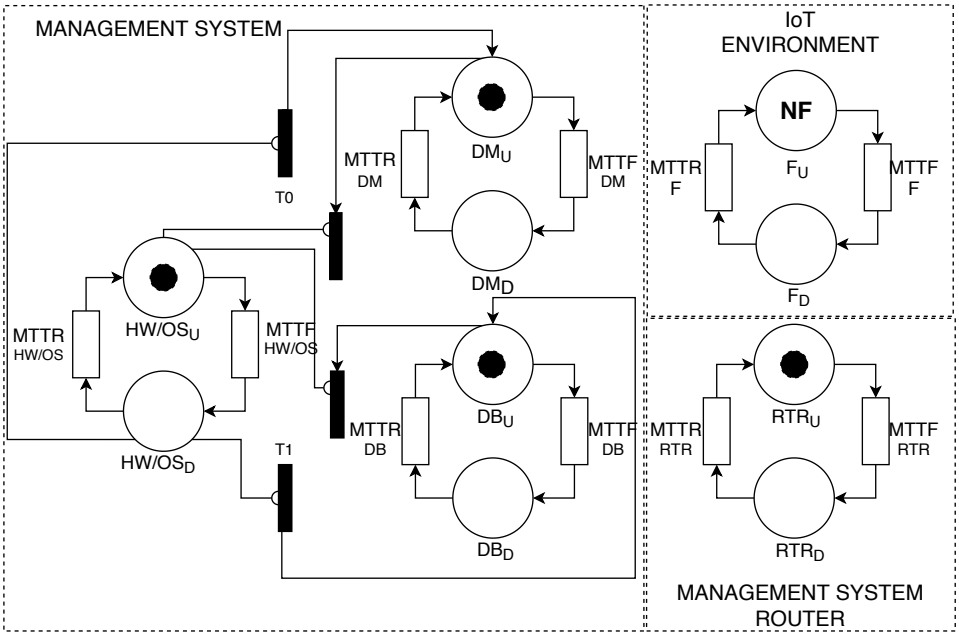


Figure 14 – SPN to entire system with local management

The management system comprises Hardware and Operating System (HW/OS), Device Manager (DM), and Database (DB). We assume, for the initial system state, that the server is working, which is represented by a token in the places: HW/OS_U (hardware and operating system are up), DM_U (device manager is up), and DB_U(database is up). Each component has an MTTF (Mean Time To Failure), whose value is assigned to the delay of its respective transition: $MTTF_{HW/OS}$ (MTTF hardware and operating system), $MTTF_{DM}$ (device manages MTTF), and $MTTF_{DB}$ (database's MTTF). These transitions follow a single-server semantics, and their firing delays are exponentially distributed. The single-server semantics does not allow multiple firing events when the degree of activation is greater than one, i.e., the transition fires only once at a time. Therefore, when a component fails, the corresponding transition consumes a token from its "up" place to its "down" place. The failure of those components is denoted by the following places: DM_D (DM is down), DB_D (DB is down), and HW/OS_D (hardware and operating system are down). When a component is down, it might be repaired by the maintenance team. Each component has an MTTR (Mean Time To Repair), in some transitions in the model: $MTTR_{HW/OS}$ (MTTR hardware and operating system), $MTTR_{DM}$ (device manages MTTR), and $MTTR_{DB}$ (database's MTTR). These transitions also follow a single-server semantics, and their firing delays are exponentially distributed. Therefore, when the component is failed, the transition representing the failure is fire according to its MTTR. Firing this transition takes a token from place "down" and adds a token in

place “up”. The $T0$ and $T1$ are immediate transitions. The $T0$ have a guard expression, where the transition only will fired when there is no token in DM_D and DM_U places. The $T1$ also have a guard expression, where the transition only will fired when there is no token in DB_D and DB_U places. The $MTTF_{HW/OS}$ can be calculated by integrating of reliability over the interval (0, ∞).

$$MTTF_{HW/OS} = \int_0^\infty R_{hw/os}(t)dt \tag{4.5}$$

Reliability can be obtained by Equation 4.6, where λ_{eq} is the immediate failure rate of all subsystem components. The λ_{eq} can be obtained by summing all immediate failure rates of all components, as shown by Equation 4.7. Where λ_{HW} and λ_{OS} are the failure rate of the hardware, and operating system, respectively.

$$R_{hw/os}(t) = e^{\lambda_{eq}t} \tag{4.6}$$

$$\lambda_{eq} = \lambda_{HW} + \lambda_{OS} \tag{4.7}$$

F_U and F_D places represent the IoT environment. The F_U place represents that all components of the IoT environment are active. Whereas the Section 4.1.3.1 represents just one floor of the building, this SPN model represents all floors we want, through the amount of token in the F_U place represent by X. The MTTF and MTTR of the IoT environment on one floor are represented in transitions $MTTF_F$ and $MTTR_F$, respectively. The firing delay of $MTTF_F$ transition corresponds to an exponential distribution, and the transition follows an infinite-server semantics, which enables the transition to fire multiple times simultaneously if there are enough tokens and the timing of events demand that. Therefore, it properly represents the fact that each floor subsystem's failure is an independent one from each other. The $MTTR_F$ transition corresponds to an exponential distribution and follows a single-server semantics. The router that sends data to the management system is represented by RTR_U and RTR_D places, which means active and inactive. The failure and repair events for this component are represented by $MTTF_{RTR}$ and $MTTR_{RTR}$ transitions, respectively. The black transitions are immediate; therefore, these transitions fire the as soon as enabled. Expression 4.8 represents the availability of the entire system.

$$\begin{aligned} A_{non-red} = P\left(\#DM_U = 1\right)\ ANDP\left(\#DB_U = 1\right)\ AND \\ P\left(\#RTR_U = 1\right)\ AND\ P\left(\#F_U = X\right). \end{aligned} \tag{4.8}$$

where X is the number of building's floor.

P means probability, and # means the number of tokens in a determined place. Therefore, the system is available when the device manager (DM), database (DB), router (RTR),

and all floors (F) are up. The annual uptime and downtime can be computed using Equations 4.9 and 4.10, respectively, considering that 8760 corresponds to the number of hours in a year.

$$Uptime = A \times 8760\ h \tag{4.9}$$

$$Downtime = 8760\ h - Uptime \tag{4.10}$$

4.1.3.3 SPN Availability Model for Entire Redundant System

This Section describes the proposed availability model for the entire system considering a redundant local management infrastructure. A Stochastic Petri Net (SPN), depicted in Figure 15, was built to represent the cold-standby redundancy system, it is characterized by considering a non-negligible time for switching between the machines in the occurrence of a failure in one of them. The main machine will always have priority over the secondary. The model also represents the IoT environment and the router responsible for sending the environment data to the management system.

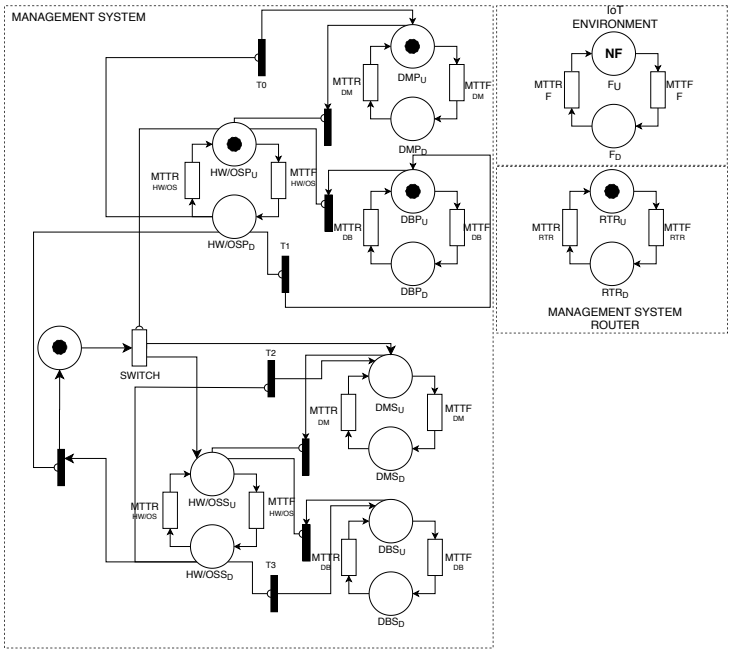


Figure 15 – SPN to entire redundant local system

The redundancy system comprises Hardware and Operating System (HW/OS), Device Manager (DM), and Database (DB). Almost all places and transitions in the model of Figure 15 are equivalent to the model presented in Figure 14. However, the cold standby component is also similarly represented. We assume that at the initial state that the primary server is working, which is represented by a token in the places HW/OSP_U (hardware and operating system are up), DMP_U (device manager is up), and DBP_U

(database is up). The failure of those components is denoted by tokens in the following places: DMP_D (DM is down), DBP_D (DB is down), and HW/OSP_D (hardware and operating are down). However, when the primary server (hardware and operating system) fails, the SWITCH transition fires, making the secondary server to exit the inactive state to enter the active state. This transition represents the time to turn on the secondary server, and transfer the workload to it, which shall be less than the time to repair the primary server. The second server is represented by the places: HW/OSS_U (hardware and operating system are up), DMS_U (device manager is up), and DBS_U (database is up). The failure of those components is denoted by the following places: DMS_D (DM is down), DBS_D (DB is down), and HW/OSS_D (hardware and operating are down). The $T0$, $T1$, $T2$, and $T3$ are immediate transitions. The $T0$ have a guard expression, where the transition only will fired when there is no token in DMP_D and DMP_U places. The $T1$ also have a guard expression, where the transition only will fired when there is no token in DBP_D and DBP_U places. The $T2$ have a guard expression, where the transition only will fired when there is no token in DMS_D and DMS_U places. The $T3$ also have a guard expression, where the transition only will fired when there is no token in DBS_D and DBS_U places. Expression 4.11 represents the availability of the entire system.

$$\begin{aligned} A_{red} = P\left(\#DMP_U = 1 \; OR \; \#DMS_U = 1\right) \; AND \\ P\left(\#DBP_U = 1 \; OR \; \#DBS_U = 1\right) \; AND \\ P\left(\#RTR_U = 1\right) \; AND \; P\left(\#F_U = X\right) \end{aligned} \tag{4.11}$$

4.1.3.4 SPN availability model for system with redundant cloud computing management

Another objective of this paper is to evaluate distinct management infrastructures that should deal better with many buildings. Therefore, we also propose an availability model for the entire system considering a cloud-based management infrastructure. The device management and data storage systems are hosted in the cloud. An SPN depicted in Figure 16, was built to represent this system. The model composition represents the IoT environment, the router, controller system, and management system in cold standby redundancy.

The redundancy system comprises Device Manager (DM), Database (DB), Hardware, Operating System, and Docker (HW/OS/DC). We assume, for the initial system state, that the primary server is working, which is represented by a token in the places: $HW/OS/DCP_U$ (hardware, operating system, and docker are up), DMP_U (device manager is up), and DBP_U(database is up). Each component has an MTTF (Mean Time To Failure), whose value is assigned to the delay of its respective transition: $MTTF_{HW/OS/DC}$ (MTTF hardware, operating system, and docker), $MTTF_{DM}$ (device manages MTTF), and $MTTF_{DB}$ (database's MTTF). These transitions follow a single-server semantics, and their firing delays are exponentially distributed. Therefore, when a component fails, the

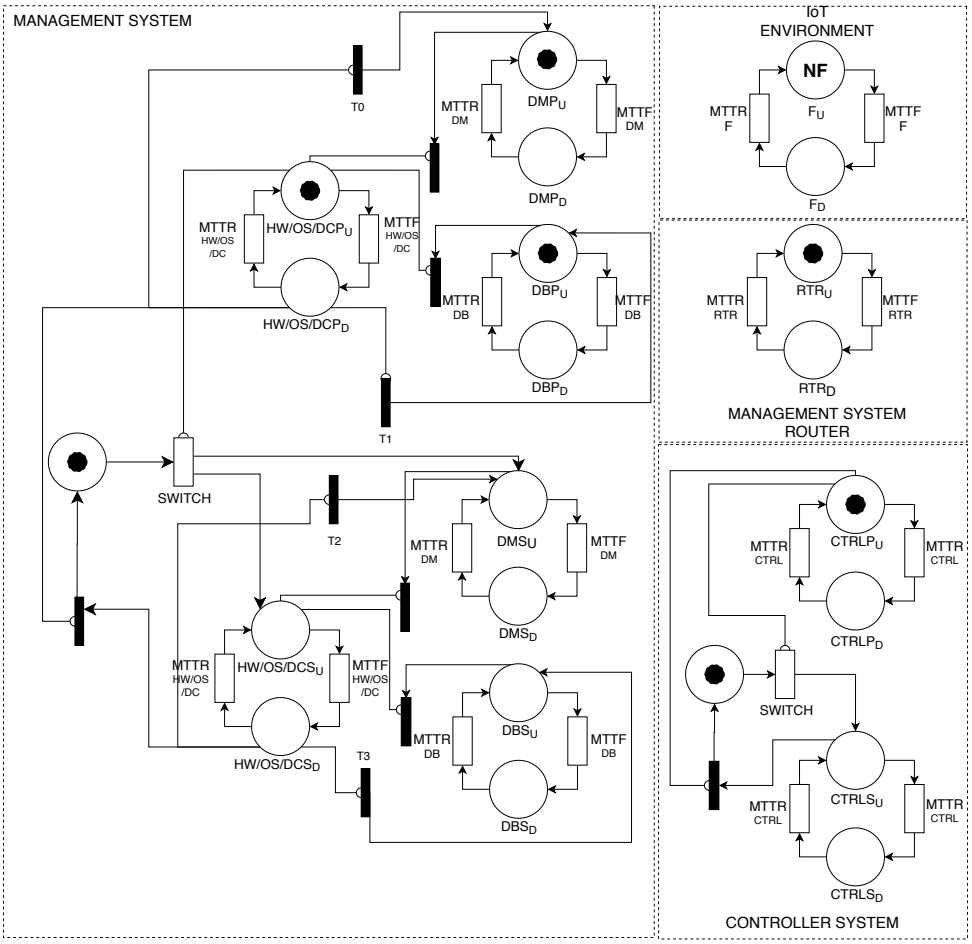


Figure 16 – SPN to entire cloud system

corresponding transition consumes a token from its “up” place to its “down” place. The failure of those components is denoted by the following places: DMP_D (DM is down), DBP_D (DB is down), and $HW/OS/DCP_D$ (HW/OS/DC are down). When a component is down, it might be repaired by the maintenance team. Each component has an MTTR (Mean Time To Repair); those rates are in some transitions in the model, they are $MTTR_{HW/OS/DC}$ (MTTR hardware, operating system, and docker), $MTTR_{DM}$ (device manages MTTR), and $MTTR_{DB}$ (database's MTTR). These transitions also follow a single-server semantics, and their firing delays are exponentially distributed. Therefore, when the component is down, the transition may fire according to its MTTR, pulling out a token from the "down" place to the "up" place. However, when the primary server (HW/OS/DCP) fails, the switch time transition fires, making the secondary server to exit the inactive state to enter the active state. This transition represents the time to turn on the secondary server and transfer the workload to it, which is less than the time to repair the primary server's hardware. The second server is represented by the places: $HW/OS/DCS_U$ (hardware, operating system, and docker are up), DMS_U (device manager is up), and DBS_U (database is up). The failure of those components is denoted by the following places: DMS_D (DM is down), DBS_D (DB is down), and $HW/OS/DCS_D$

(hardware, operating system, and docker are down). The $T0$, $T1$, $T2$, and $T3$ are immediate transitions. The $T0$ have a guard expression, where the transition only will fired when there is no token in DMP_D and DMP_U places. The $T1$ also have a guard expression, where the transition only will fired when there is no token in DBP_D and DBP_U places. The $T2$ have a guard expression, where the transition only will fired when there is no token in DMS_D and DMS_U places. The $T3$ also have a guard expression, where the transition only will fired when there is no token in DBS_D and DBS_U places. The $MTTF_{HW/OS/DC}$ of a subsystem can be calculated through Equation 4.12, while the system's $MTTR_F$ can be obtained from the data reported by the repair team.

$$MTTF_{HW/OS/DC} = \int_0^\infty R_{hw/os/dc}(t)dt \tag{4.12}$$

Reliability can be obtained by Equation 4.13, where λ_{eq} is the immediate failure rate of all subsystem components. The λ_{eq} can be obtained by summing all immediate failure rates of all components, as shown by Equation 4.14. Where λ_{HW}, λ_{OS}, and λ_{DC} are the failure rate of the hardware, operating system, and docker, respectively.

$$R_{hw/os/dc}(t) = e^{\lambda_{eq}t} \tag{4.13}$$

$$\lambda_{eq} = \lambda_{HW} + \lambda_{OS} + \lambda_{DC} \tag{4.14}$$

The Controller subsystem is represented by $CTRLP_U$, $CTRLS_U$, $CTRLP_D$, and $CTRLS_D$ places in cold standby redundancy. This redundancy is composed of Hardware (HW), Operating System (OS), and Controller Manage (CM). We consider that the $MTTF_{CTRL}$ and $MTTR_{CTRL}$ values of the components are exponentially distributed. The $MTTF_{CTRL}$ of this subsystem can be calculated through Equation 4.15, while the system's MTTRs can be obtained from the data reported by the repair team.

$$MTTF_{CTRL} = \int_0^\infty R_c(t)dt \tag{4.15}$$

Reliability can be obtained by Equation 4.16, where λ_{eq} is the immediate failure rate of all subsystem components. The λ_{eq} can be obtained by summing all immediate failure rates of all components, as shown by Equation 4.17. Where λ_{HW}, λ_{OS}, and λ_{CM} are the failure rate of the hardware, operating system, and controller manager, respectively.

$$R_c(t) = e^{\lambda_{eq}t} \tag{4.16}$$

$$\lambda_{eq} = \lambda_{HW} + \lambda_{OS} + \lambda_{CM} \tag{4.17}$$

For the initial system state, we assume that the primary controller is working, represented by a token in the place: $CTRLP_U$ (the controller is up). The controller has an MTTF (Mean Time To Failure), whose value is assigned to the delay of its respective transition: $MTTF_{CTRL}$ (controller's MTTF). Therefore, when a component fails, the corresponding transition consumes a token from its “up" place to its “down" place. The failure of that component is denoted in the place: $CTRLP_D$ (CTRL is down). When the controller is down, it might be repaired by the maintenance team. The controller has an MTTR (Mean Time To Repair), that rate is in transition in the model: $MTTR_{CTRL}$ (controller's MTTR). Both $CTRL_{MTTF}$ and $MTTR_{CTRL}$ firing delays are exponentially distributed, and these transitions follow a single-server semantics. Therefore, when the component is down, the transition may fire according to its MTTR, pulling out a token from the "down" place to the "up" place. However, when the primary controller fails, the switch time transition fires, making the secondary controller exit the inactive state to enter the active state. The secondary controller is represented by the places: $CTRLS_U$ (the controller is up) and $CTRLS_D$ (controller down).

The IoT environment is represented by F_U and F_D places. The F_U place represents that all components of the IoT environment are active. Whereas the Section 4.1.3.1 represents just one floor of the building, this SPN model represents all floors we want, through the amount of token in the F_U place represent by X. The MTTF and MTTR of the IoT environment on one floor are represented in transitions $MTTF_F$ and $MTTR_F$, respectively. The $MTTF_F$ transition firing delay corresponds to an exponential distribution, which has an infinite-server semantics. The $MTTR_F$ firing delay corresponds to an exponential distribution, which has a single-server semantics. The router that sends data to the storage system is represented by RTR_U and RTR_D places, active and inactive. $MTTF_{RTR}$ and $MTTR_{RTR}$ transitions, respectively, represent the failure and repair events for this component. The black transitions are immediate; therefore, these transitions fires as soon as enabled. Expression 4.18 represents the availability of the entire system.

$$\begin{aligned} A_{cloud} = P\left(\#DMP_U = 1 \; OR \; \#DMS_U = 1\right) \; AND \\ P\left(\#DBP_U = 1 \; OR \; \#DBS_U = 1\right) \; AND \\ P\left(\#CTRLP_U = 1 \; OR \; \#CTRLS_U = 1\right) \; AND \\ P\left(\#F_U = NF\right) \; AND \; P\left(\#RTR_U = 1\right) \end{aligned} \tag{4.18}$$

Therefore, the system is available when the device manager (DM), database (DB), router (RTR), controller (CTRL), and all floors (F) are up. The uptime and downtime can be computed using Equations 4.19 and 4.20, respectively.

$$Uptime = A_{cloud} \times 8760 \; h \tag{4.19}$$

$$Downtime = 8760\ h - Uptime \tag{4.20}$$

4.2 MANAGEMENT PHOTOVOLTAIC SYSTEM

In this section, an intelligent architecture for energy management in photovoltaic systems is proposed. However, firstly is presented architecture general of the photovoltaic systems, then we evolved the concept to an autonomous energy management system for photovoltaic systems.

4.2.1 Photovoltaic system

In this Section we presented a general architecture photovoltaic system. This architecture includes the main components of grid-tie, and hybrid photovoltaic systems. Figure 17 shows the common components found in these type of system. The solar panel is responsible for capturing energy from sunlight and transforming it into electrical energy, this element is indispensable in all photovoltaic systems. The battery is responsible for storing the energy captured by the solar panel, all hybrid photovoltaic system have this component. The utility grid, source of energy used when solar energy cannot supply the demand. This alternative energy source is used in hybrid and grid-tie systems. The building represents the set of elements that will consume electricity. It is worth mentioning that it is possible to have others sources of energy in addition to the previously mentioned.

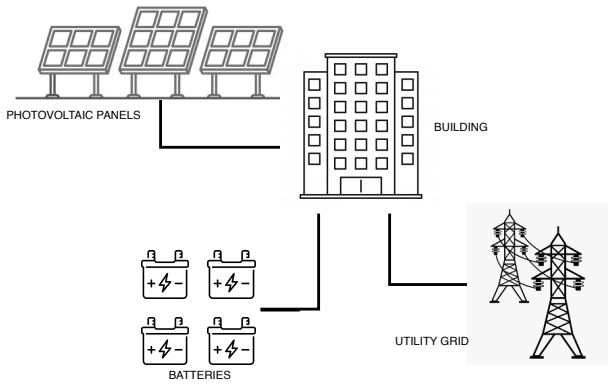


Figure 17 – Common components in a photovoltaic system

However, as mentioned earlier, not all photovoltaic systems have all of these components. For example, a grid-tie Systems do not have batteries. This makes this type of system the cheapest among the other types. Besides, depending on the tariff system adopted, the power grid can be used as a virtual battery. Figure 18 shows the components needed in this type of photovoltaic system. The inverter is the electrical equipment able to convert direct current to alternating current. The String box is protective equipment

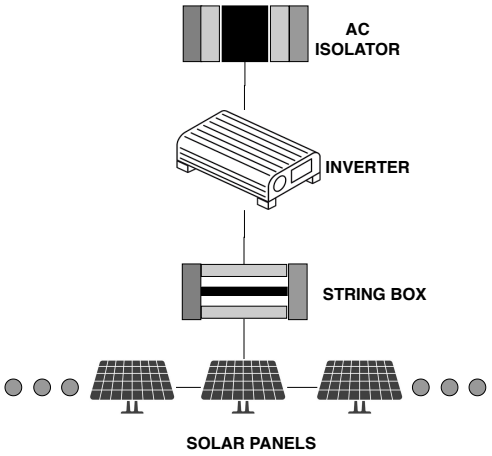


Figure 18 – Grid-tie photovoltaic system main components

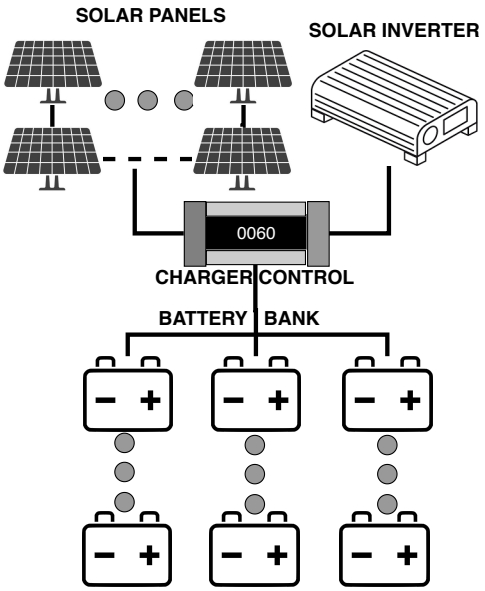


Figure 19 – Hybrid photovoltaic system main components

that isolates the photovoltaic energy production system, to prevent the risk of spreading electrical accidents. The AC isolator is the equipment responsible for connecting the inverter and electrical network safely.

Hybrid solar systems are an alternative for those who desire have more than one energy source. However, this type of system is the most expensive than grid-tie system, mainly due to the number of batteries used to support the entire energy demand. On the other hand, it allows access to energy, even if there is a power failure utility grid. Figure 19 shows the necessary components in this type of system. The charge controller is responsible to connect the solar panels and the batteries correctly.

In this work, we considered a grid-tie and Hybrid systems to smart photovoltaic system. The following Section presents our proposal for energy management based on Internet of Things.

4.2.2 Smart Photovoltaic System

This section presents an autonomous management system for the photovoltaic system. This system adopts technologies based on cloud computing and the Internet of things to perform the energy management of the building. This is generic architecture, and can be adapted for any energy management system that contains the same principles. Figure 20 presents a macro view of the autonomous system for energy management. This system includes all the components presented in the previous section (with the presence of sensors). As well, it considers a cloud computing infrastructure and one switch power, more details are provided below.

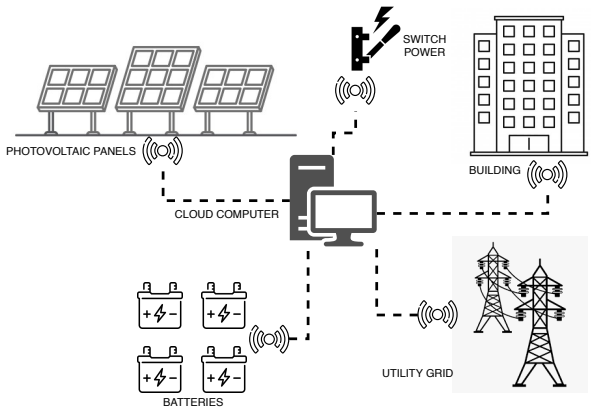


Figure 20 – Proposed IoT environment

The sensor in the photovoltaic panels captures information about the amount of energy produces. Therefore, it is possible to determine whether the energy produced by the panels is sufficient to supply the building in real-time (grid-tie system). Therefore on rainy and cloudy days, where there is little or no incidence of particles of sunlight, the solar system will remain off. Besides, it is easy to identify any defects in the panels. The sensor located in the battery pack has the purpose of informing about the charge level. It is then possible to determine whether the stored energy can supply the building's demand (hybrid system). If the amount of energy stored in the battery is insufficient, the system will remain off, and the utility grid will remain turn on. The sensor located in the utility grid provides information on the amount of energy used. As well, it is possible to identify if there is a problem with the power supply. All information collected by the sensors is sent to cloud computing infrastructure, where the coordination system will make the appropriate decision for each situation Figure 21 shows the sensors used in the building.

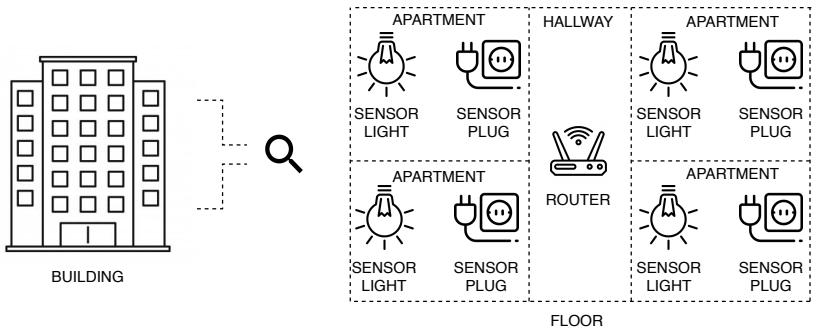


Figure 21 – Building's sensors

The sensors present in the building capture information about the amount of energy being used to supply lights and electronic equipment. This information is necessary so that the control system (cloud computing infrastructure) can identify when it will be possible to use energy from the photovoltaic system. When possible, the control system (cloud computing) will send a signal to the actuator located on the switch power to activate

the power supply directly from the photovoltaic panels (grid-tie system) or battery pack (hybrid system). Likewise, when it is not possible, the control system determines the actuator to activate the supply of energy via the utility grid. All data is transmitted through the routers present in the building.

The adoption of sensors makes it possible to recognize the environment, making them powerful instruments for understanding complex things and responding to them efficiently. The sensors can communicate through different protocols, in this work we adopt the technology ZigBee protocol based on the IEEE 802.15.4 standard and MQTT (MQ Telemetry Transport) a lightweight messaging protocol for small sensors to carry out the communication between components. Figure 22 shows the communication between the device and cloud computing. There is bidirectional communication between all components. However, this not occurs totally in practice. The sensors present in the photovoltaic system, batteries, and utility grid only send data. While the sensors located in building send and receive data, and smart power switch only receive data.

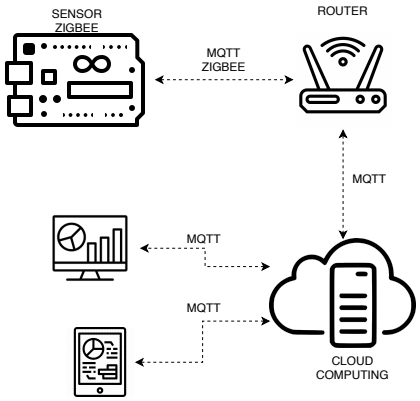


Figure 22 – Communication between sensors and cloud computing

4.2.3 Availability and Relibility Related Metrics

This Section describes the proposed models used for steady-state availability evaluation. Stochastic Petri Net and mathematical equations were used to represent the relationship between the components previously presented in Section 4.2.2. Mercury tool was also adopted for SPN modeling analysis (SILVA et al., 2015).

4.2.3.1 Model of the Building

This Section describes the proposed model to calculate the MTTF to a single floor. The components that composed a single floor were described in Figure 21. Therefore, we have smart plug, smart light, and router. We consider that the $MTTF_F$ (Mean Time To Failure) and $MTTR_F$ (Mean Time To Repair) of the floor subsystem are exponentially distributed. Therefore, the $MTTF_F$ can be calculated through the integration of reliability

as a function of time, as described in Equation 4.21, while the system's $MTTR_F$ can be obtained from the reported data of the repair team.

$$MTTF_F = \int_0^\infty R_f(t)dt \tag{4.21}$$

Reliability can be obtained by Equation 4.22, where λ_{eq} is the immediate failure rate of the subsystem, considering that a single failure in any component brings the whole subsystem down. The λ_{eq} can be obtained by summing all immediate failure rates of all components, as shown by Equation 4.23, where λ_{PLUG}, λ_{LIGHT}, and λ_{ROUTER} are the failure rate of the smart plugs, light meter sensors and router respectively.

$$R_F(t) = e^{\lambda_{eq}t} \tag{4.22}$$

$$\lambda_{eq} = (X \times \lambda_{PLUG}) + (Y \times \lambda_{LIGHT}) + (Z \times \lambda_{ROUTER}) \tag{4.23}$$

where X represents the number of smart plugs, Y the number of light meter sensors, and Z the number of routers.

It is worth mentioning that the mathematical models presented in this Section are generic. Therefore, it allows each designer to represent the appropriate environment for each situation.

4.2.3.2 Model of the Hybrid Solar System

This Section describes the proposed model to calculate the MTTF to a hybrid solar power system. Figure 19 describe the environment proposed, with the exception of the sensors present on the battery bank, on photovoltaic panels, and on the utility grid, as shown in figure 20. Therefore, we have photovoltaic panels, batteries, inverter, charge control, sensor on panels, battery bank, and utility grid. I consider that the $MTTF_{PV}$ and $MTTR_{PV}$ are the MTTF and MTTR of phothovoltaic subsystem, and are exponentially distributed. Therefore, the $MTTF_{PV}$ can be calculated through the integral of reliability as a function of time, as described in Equation 4.24, while the system's $MTTR_{PV}$ can be obtained from the reported data of the repair team.

$$MTTF_{PV} = \int_0^\infty R_f(t)dt \tag{4.24}$$

Reliability can be obtained by Equation 4.25, where λ_{eq} is the immediate failure rate of the subsystem, considering that a single failure in any component brings the whole subsystem down. The λ_{eq} can be obtained by summing all immediate failure rates of all components, as shown by Equation 4.26, where λ_{PANEL}, λ_{CTRL}, $\lambda_{BATTERY}$, $\lambda_{INVERTER}$,

λ_{PS}, λ_{BS}, and λ_{US} are the failure rate of the solar panel, charge controller, battery, inverter,panel sensor, battery sensor, and utility grid sensor, respectively.

$$R_P(t) = e^{\lambda_{eq}t} \tag{4.25}$$

$$\begin{aligned} \lambda_{eq} = (X \times \lambda_{PANEL}) + (Y \times \lambda_{BATTERY}) + \lambda_{US} + \\ (Z \times \lambda_{INVERTER}) + (W \times \lambda_{CTRL}) + \lambda_{PS} + \lambda_{BS} \end{aligned} \tag{4.26}$$

where X represents the number of panels, Y the number of batteries, Z the number of inverters, and W number of charger controls.

As in the previous Section, the mathematical model presented here is generic. Therefore, the designer can manipulate the equations to represent a hybrid photovoltaic park as needed.

4.2.3.3 Model of Grid-tied Solar System

This Section describes the proposed model to calculate the MTTF to an on-grid solar power system. Figure 18 describe the environment proposed, except the sensors present on the on photovoltaic panels and the utility grid, as shown in figure 20. Therefore, we have photovoltaic panels, an AC isolator, an inverter, a string box sensor on panels, and a utility grid sensor. We consider that the $MTTF_{PV}$ and $MTTR_{PV}$ are exponentially distributed. Therefore, the $MTTF_{PV}$ can be calculated from Equation 4.27 and the system's $MTTR_{PV}$ can be obtained from the failure reported data.

$$MTTF_{PV} = \int_0^\infty R_f(t)dt \tag{4.27}$$

Reliability can be obtained by Equation 4.28, where λ_{eq} is the immediate failure rate of the subsystem, considering that a single failure in any component brings the whole subsystem down. The λ_{eq} can be obtained by summing all immediate failure rates of all components, as shown by Equation 4.29, where λ_{PANEL}, λ_{ISO}, λ_{STR}, $\lambda_{INVERTER}$, λ_{PS}, and λ_{US} are the failure rate of the solar panel, AC isolator, string box, solar inverter, panel sensor, and utility grid sensor respectively.

$$R_P(t) = e^{\lambda_{eq}t} \tag{4.28}$$

$$\begin{aligned} \lambda_{eq} = (X \times \lambda_{PANEL}) + (Y \times \lambda_{ISO}) + (Z \times \lambda_{STR}) \\ + (W \times \lambda_{INVERTER}) + \lambda_{PS} + \lambda_{US} \end{aligned} \tag{4.29}$$

where X represents the number of panels, Y number of AC isolators, Z number of string box, and W number of inverters.

4.2.3.4 SPN Availability Model of the Entire System

This Section describes the proposed availability model for the entire system considering a redundant local management infrastructure. Figure 23 shows the management system infrastructure.

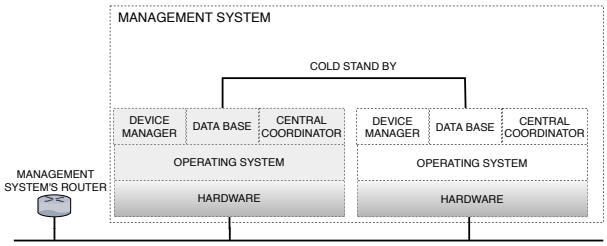


Figure 23 – Local management infrastructure

This local infrastructure contains the minimum requirements with redundancy for the operation of the proposed system. However, to ensure greater availability, we adopt a cold standby redundancy system, so the secondary machine acts as a backup of the main machine. Therefore, the necessary O.S. and software environment setup will only be made after the first machine's primary break down. Accordingly, to obtain availability metrics, a router, hardware, operating system, device manager, database, and finally, central coordinator, software that analyzes the energy consumption and the energy generated by the solar energy system to activate or deactivate the solar energy system.

A Stochastic Petri Net (SPN), depicted in Figure 24, was built to represent the cold-standby redundancy system, which is characterized by considering a non-negligible time for switching between the machines in the occurrence of a failure in one of them. The main machine will always have priority over the secondary. The model also represents the IoT environment and solar power system. Therefore, it is possible to use the "same" model to evaluate both hybryd and Grid-tie system. However, it is necessary to use the corresponding MTTF and MTTR values for each solar power system model. The smart switch power and the router responsible for sending the environment data to the management system are also presented in the model. The SPN model was used because it allows transitions with delay values exponentially distributed, which is necessary to accurately represent the proposed system. Besides, it enables the representation of system behavior and evaluation of required metrics only and concisely.

The redundancy system comprises Hardware and Operating System (HW/OS), Device Manager (DM), Central controller (CC), and Database (DB). We assume, for the initial system state, that the primary server is working, which is represented by a token in the places: HW/OSP_U (hardware and operating system are up), DMP_U (device manager is up), CCP_U (the central controller is up), and DBP_U (database is up). Each component has an MTTF, whose value is assigned to delay of its respective transition: $MTTF_{HW/OS}$ (MTTF hardware and operating system), $MTTF_{DM}$ (device manages MTTF), $MTTF_{CC}$

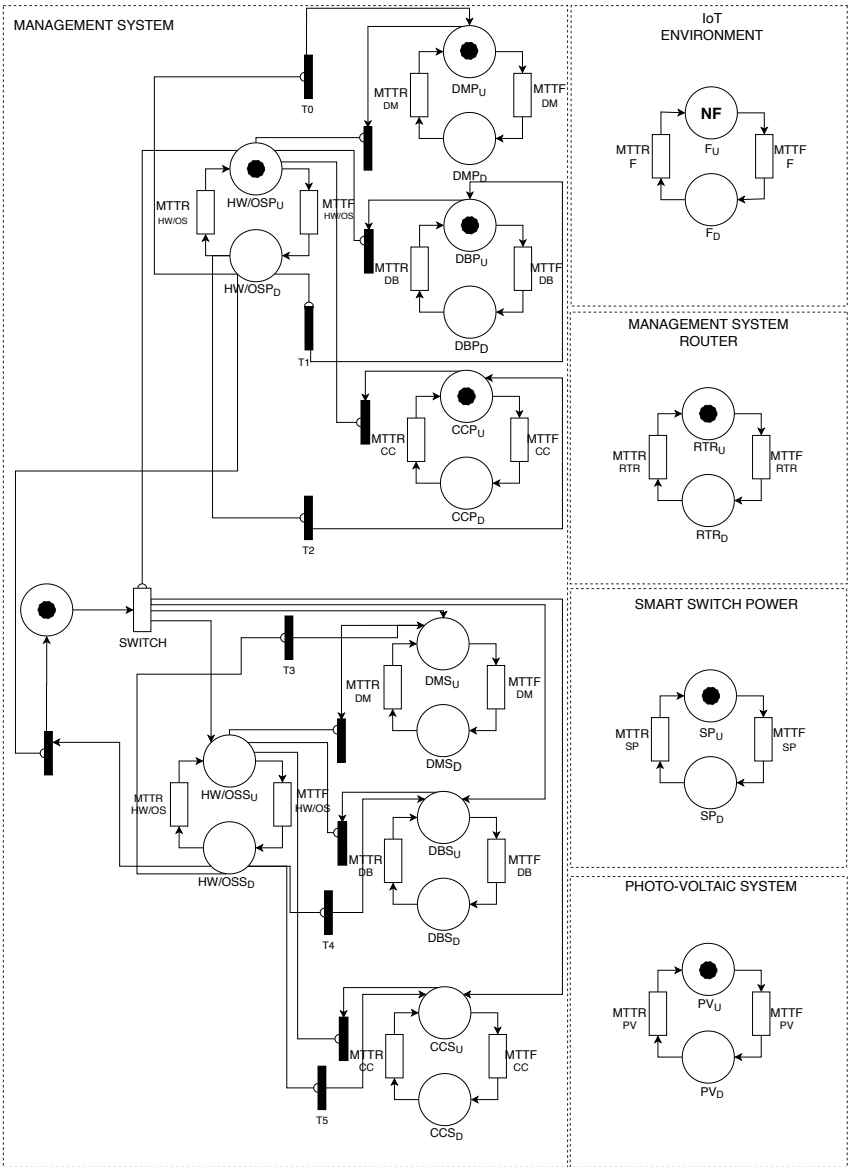


Figure 24 – SPN to entire system with local management

(central controller's MTTF), and $MTTF_{DB}$ (database's MTTF). These transitions follow a single-server semantic, and their firing delays are exponentially distributed. The single-server semantic does not allow multiple firing events when the degree of activation is more significant than one, i.e., the transition fires only once at a time. Therefore, when a component fails, the corresponding transition consumes a token from its "up" place to its "down" place. The failure of those components is denoted by the following places: DMP_D (DM is down), CCP_D (CC is down), DBP_D (DB is down), and HW/OSP_D (hardware and operating are down). When a component is down, it might be repaired by the maintenance team. Each component has an MTTR, in some transitions in the model: $MTTR_{HW/OS}$ (MTTR hardware and operating system), $MTTR_{DM}$ (device manages MTTR), $MTTR_{CC}$ (central controller's MTTR), and $MTTR_{DB}$ (database's MTTR). These transitions also follow a single-server semantic, and their firing delays are exponentially distributed. Therefore,

when the component is down the transition may fire according to its MTTR, pulling out a token from the “down" place to the “up" place. However, when the primary server (hardware and operating system) fails, the SWITCH transition fires, making the secondary server to leave the inactive state to enter the active state. This transition represents the time to turn on the secondary server, and transfer the workload to it, which shall be less than the time to repair the primary server. The second server is represented by the places: HW/OSS_U (hardware and operating system are up), DMS_U (device manager is up), CCS_U (the central controller is up), and DBS_U (database is up). The failure of those components is denoted by the following places: DMS_D (DM is down), CCS_D (CC is down), DBS_D (DB is down) and HW/OSS_D (hardware and operating are down). The $T0$, $T1$, $T2$, $T3$, $T4$, and $T5$ are immediate transitions. The $T0$ have a guard expression, where the transition only will fired when there is no token in DMP_D and DMP_U places. The $T1$ also have a guard expression, where the transition only will fired when there is no token in DBP_D and DBP_U places. The $T2$ as well have a guard expression, where the transition only will fired when there is no token in CCP_D and CCP_U places. The $T3$ have a guard expression, where the transition only will fired when there is no token in DMS_D and DMS_U places. The $T4$ also have a guard expression, where the transition only will fired when there is no token in DBS_D and DBS_U places. The $T5$ as well have a guard expression, where the transition only will fired when there is no token in CCS_D and CCS_U places. The $MTTF_{HW/OS}$ is obtained from Equation 4.30, whereas the system's MTTRs is obtained through the repair team specifies reliability server.

$$MTTF_{HW/OS} = \int_0^\infty R_{hw/os}(t)dt \tag{4.30}$$

Reliability can be obtained by Equation 4.31, where λ_{eq} is the immediate failure rate of all subsystem components. The λ_{eq} can be obtained by summing all immediate failure rates of all components, as shown by Equation 4.32, where λ_{HW} and λ_{OS} are the failure rate of the hardware, and operating system, respectively.

$$R_{hw/os}(t) = e^{\lambda_{eq}t} \tag{4.31}$$

$$\lambda_{eq} = \lambda_{HW} + \lambda_{OS} \tag{4.32}$$

The building environment is represented by F_U and F_D places. The F_U place represents that all components of the building environment are active. Whereas the Figure 21 describe only one floor of the building, this SPN model represents all floors we want, through the amount of token in the F_U place represent by X. The MTTF and MTTR of the building environment on a floor are represented in transitions $MTTF_F$ and $MTTR_F$, respectively. The firing delay of $MTTF_F$ transition corresponds to an exponential distribution, and the transition follows an infinite-server semantic. Therefore, it represents

appropriately the fact that the failure of each floor subsystem is independent of each other. The $MTTR_F$ transition corresponds to an exponential distribution and follows a single-server semantic, because we only consider a maintenance team.

The solar power system is represented by PV_U and PV_D places. The PV_U place represents that all components of the solar power system are active. The MTTF and MTTR of the solar power system are represented in transitions $MTTF_{PV}$ and $MTTR_{PV}$, respectively. These transitions also follow a single-server semantic, and their firing delays are exponentially distributed. Whereas, the Section 4.2.3.2 and the Section 4.2.3.3 presented how the values of the transitions were obtained. The smart switch power sensor, which exchanges the electrical energy to solar energy and vice and versa, is represented by SP_U and SP_D places; it means active and inactive, respectively. $MTTF_{RTR}$ and $MTTR_{RTR}$ transitions, respectively, represent the failure and repair events for this component. These transitions also follow a single-server semantic, and their firing delays are exponentially distributed. The router on the cloud computing system, which sends data to the management system, is represented by RTR_U and RTR_D places, which means active and inactive. $MTTF_{RTR}$ and $MTTR_{RTR}$ transitions, respectively, represent the failure and repair events for this component. These transitions also follow a single-server semantic, and their firing delays are exponentially distributed. The black transitions are immediate; therefore, these transitions fire the as soon as they are enabled. Expression 4.33 represents the availability of the entire system.

$$\begin{aligned} A = P\left(\#DMP_U = 1 \; OR \; \#DMS_U = 1\right) \; AND \\ P\left(\#CCP_U = 1 \; OR \; \#CCS_U = 1\right) \; AND \\ P\left(\#DBP_U = 1 \; OR \; \#DBS_U = 1\right) \; AND \\ P\left(\#RTR_U = 1\right) \; AND \; P\left(\#F_U = NF\right) \; AND \\ P\left(\#PV_U = 1\right) \; AND \; P\left(\#SP_U = 1\right) \end{aligned} \tag{4.33}$$

P means probability, and # means the number of tokens in a determined place. Therefore, the system is available when the device manager (DM), central controller (CC) database (DB), router (RTR), solar power system (PV), smart switch power (SP), and all floors (F) are up. The annual uptime and downtime can be computed using Equations 4.34 and 4.35, respectively, considering that 8760 corresponds to the number of hours in a year.

$$Uptime = A \times 8760 \; h \tag{4.34}$$

$$Downtime = 8760 \; h - Uptime \tag{4.35}$$

5 CASE STUDIES

This chapter describes the case studies proposed in this work. Chapter 4 presents some case studies of generic architecture in a smart building. The objective is to evaluate the availability of three distinct management infrastructures in different scenarios. Besides, the work presents a sensitivity analysis to identify the components that most affected availability. The second Section presents case studies to the smart photovoltaic system. In which, consider a grid-tied and hybrid photovoltaic system. The evaluation impact of adopting the photovoltaic system in an intelligent building considered the architectures presented in the previous chapter.

5.1 GENERIC SMART BUILDING

To evaluate the generic smart building system proposed in Section 4.1.2, I considered a building with fifteen floors, containing four apartments per floor. Each floor has one gas sensor, capturing data about the possible gas leak in the kitchen. Also, there are one motion sensors on the hallway to identify when the light will turn on. Besides, there is one smoke sensor to identify incidences. The router present on each floor sent data to the management system. The models that used to evaluate the architecture are described in Section 4.1.3.

In order to evaluate our models, some input values are needed. These values can be found in the literature. The MTTF and MTTR values of each component are extracted from (DANTAS et al., 2012)(COOPER; FARRELL, 2007)(NETACARD, 2014)(KIM; MACHIDA; TRIVEDI, 2009). The input values for each of the previous models can be seen in Table 2.

Table 2 – Input values for SPN

Component	**MTTF (hours)**	**MTTR (hours)**
Hardware	61320	8
Operating System	1440	1
Sensors (G, S and M)	300000	1
Docker	2900	1
Controller Manager	700	1
Data Base	1440	1
Device Manager	700	1
Router	26000	8

5.1.1 Availability evaluate in a generic architecture to smart building

Three case studies are presented in this Section to evaluate the IoT system employing the proposed models described in Section 4.1.2. The main objective is to estimate the system availability and identify the components that most affect the availability considering distinct parameter values for the whole system. However, availability is estimated using a local management infrastructure and a cloud computing management infrastructure separately, and then we compare them. The local infrastructure's acquisition costs with redundancy and the cloud computing infrastructure also calculated, considering a three-year time interval.

5.1.2 Comparison between non-redundant and redundant local infrastructures

Table 3 – Comparison between local infrastructures

	Downtime Hours	
Amount of Floors	**No Redundancy**	**Redundancy**
15	62.19	53.10
14	47.14	39.95
13	46.27	38.33
12	46.27	38.33

Table 3 shows the model results for the smart building system employing a local management infrastructure without redundancy and cold-standby redundancy. At first, considered 15 floors, and all the components on all floors have to be working. Then, gradually reduced the amount necessary for floors; for example, at least fourteen out of the fifteen floors worked. The downtime results in fifteen floors are $62.19\,h$ for the infrastructure without redundancy, and $53.10\,h$ to infrastructure with redundancy. Thus, there was an availability increase from $99.29\,\%$ to $99.39\,\%$ between both infrastructures, as shown in Table 4. Lastly, when are considering at least fourteen, thirteen, and twelve, the downtime was always smaller in the infrastructure with cold-standby redundancy than without redundancy. The downtime converges to $47.14\,h$ on infrastructure without redundancy and $39.95\,h$ to infrastructure with redundancy, as noticed on the similar results for 13 and 12 floors.

Table 4 – Availability metrics results to building with 15 floors

Metric	**No Redundancy**	**Redundancy**
Uptime	8697.81 h	8706.90 h
Availability	99.29%	99.39%

5.1.3 Comparison between local and cloud infrastructures

This Section describes the comparison between the cloud computing structure and local structure. This comparison takes into account the availability and acquisition cost. The evaluation considers just the structures with redundancy, and the acquisition costs calculate for three years.

To perform the cost analysis, we consider two management servers with one processor comprising two cores, $8\,GB$ (gigabytes) of main memory, and $1\,TB$ (terabyte) of hard disk memory as well as; we are also considering one router of medium size to the local infrastructure with redundancy. The infrastructure Total Cost of Ownership (TCO) can obtain through Equation 5.1, where n is the total amount of components.

$$TCO = \sum_{k=1}^{n}(CV_k \times MP_k) \tag{5.1}$$

The component k acquisition value represented by CV_k and MP_k represents the percentage of maintenance cost to the equipment per year. To calculate the cost of acquisition of the cloud computing infrastructure, we consider two management servers with the same characteristics adopted in local infrastructure. The controller servers also have the same characteristics except that they do not have storage capacity. The value of the acquisition to cloud computing infrastructure performs through the AWS Total Cost of Ownership (TCO) Calculators (AMAZOM, 2019).

Table 5 – Comparison between local and cloud with cold-standby in relation of cost - 3 years

	US$	
Amount of buildings	**local**	**cloud**
1	4,808.50	7,782.00
2	9,617.00	7,782.00
3	14,425.50	7,782.00
4	19,234.00	7,782.00
5	24,042.50	7,782.00

Table 5 presented the cost comparison between the local infrastructure with redundancy and the cloud computing infrastructure. When we consider a single building, the local infrastructure has a smaller cost of acquisition: $US\$\,4,808.50$, which is almost 40 % less than the cost in cloud computing infrastructure ($US\$\,7,782.00$) in three years. Nevertheless, when we consider more than one building, the cloud computing infrastructure has a smaller cost than the local infrastructure. The high cost of local infrastructure occurs due to the need to multiply the local infrastructure, while the cloud computing infrastructure stays the same for all buildings analyzed. We also compared the infrastructure with the availabilities.

Table 6 – Comparison between local and cloud with cold-standby

	Downtime (h)	
Amount of buildings	**local**	**cloud**
1	53.10	61.67
2	106.54	81.44
3	159.33	101.24
4	211.79	121.35
5	263.94	141.83

Table 6 shows the model results for the smart building system employing local and cloud computing. We adopted all buildings with fifteen floors in this analysis, and all floors must be working. The availability analysis showed that considering a single building. The downtime is smaller in the local infrastructure than cloud computing infrastructure, 53.10 h, and 61.67 h, respectively. However, the scenario changes when there is more than one building. When considers two or more buildings, the downtime in the cloud computing infrastructure is smaller than the downtime in local infrastructure, and the difference between them grows almost linearly. Figure 25 shows that the downtime difference between both infrastructures is directly proportional to the number of buildings.

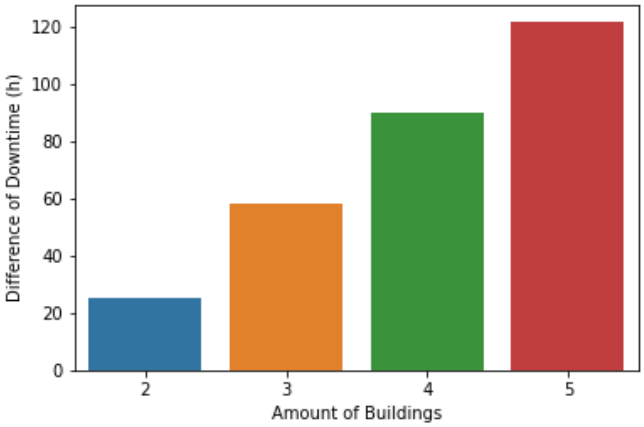


Figure 25 – Downtime difference between cloud and local infrastructure in hours

5.1.4 Sensitivity analysis

To improve the results obtained through the availability evaluation, identified the system bottlenecks employing sensitivity analysis. This section describes the sensitivity analysis in all infrastructures proposed in this paper. Hence, we can use Equation 2.21 in each model. In this work, we divide the sensitivity analysis into two steps. In the first step, we considered that all components on all floors must be working. In the second step, we considered that all components in at least fourteen floors are working. Table 7 shows

the sensitivity ranking, considering all smart building system components for a local management infrastructure without redundancy.

Table 7 – Sensitivity ranking for local infrastructure without redundancy

Component	Sensitivity Index
$MTTF_F$	0.00769700
$MTTR_F$	0.00624340
$MTTF_{HW/OS}$	0.00313548
$MTTF_{DM}$	0.001845736
$MTTF_{DB}$	0.001845718
$MTTR_{DM}$	0.001685539
$MTTR_{DB}$	0.001685531
$MTTR_{HW/OS}$	0.000732489
$MTTR_{RTR}$	0.000273396
$MTTF_{RTR}$	0.000040983

The results presented in Table 7 show that the IoT floor environment is the component that most affects availability considering both MTTF and MTTR. Figure 26 presents the relation between the MTTF of the IoT environment and the smart building system availability. When this parameter reaches its lowest value, just below $8,000\,h$, the overall system availability decreases to its lowest value, approximately 98.8 %. The opposite occurs when the parameter reaches its highest value, about $25,000h$. The availability reaches its highest value, about 99.4 %.

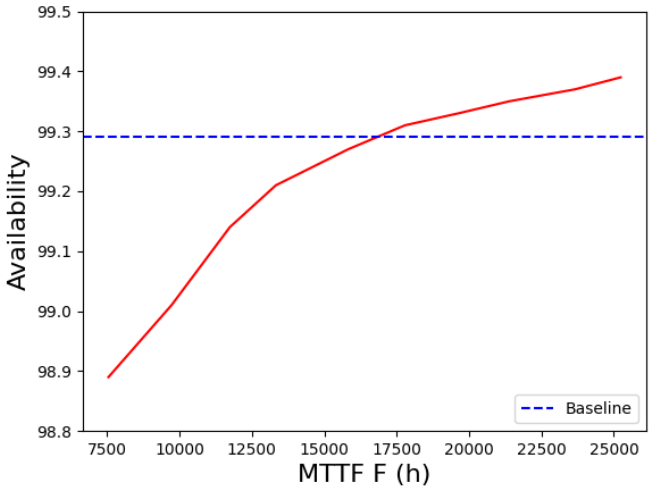


Figure 26 – Sensitivity of system availability with respect to $MTTF_F$ (IoT floor environment failure)

The second parameter in the ranking is the MTTR of the IoT environment, and Figure 27 presents the relation with the availability. When the MTTR reaches its smallest

value, below $3\,h$, the availability grows to above $99.4\,\%$. The opposite occurs when the MTTR reaches to its most significant value, above $8\,h$. The availability decreases by to below $99.15\,\%$.

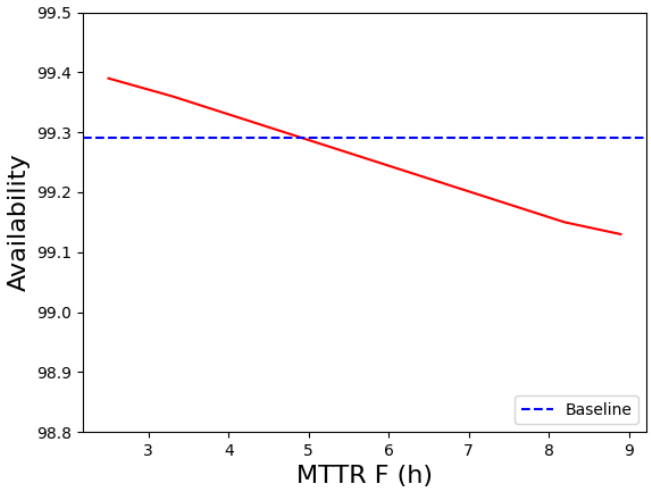


Figure 27 – Sensitivity of system availability with respect to $MTTR_F$ (IoT floor environment failure)

The IoT environment is composed of sensors and routers; thus, this work performed a sensitivity analysis described in Equation 2.21 for this environment. The results showed that the router is the component with the highest impact in the IoT environment, and all sensors have the same sensitivity index. We also performed The sensitivity analysis considering that at least fourteen floors must be working. The new analysis of the same infrastructure presented different results compared with the values presented in Table 7.

Table 8 – Sensitivity analysis for local infrastructure without redundancy and requirement of 14 out of 15 floors working

Component	**Sensitivity Index**
$MTTF_{HW/OS}$	0.002413548
$MTTF_{DM}$	0.001845736
$MTTF_{DB}$	0.001845718
$MTTR_{DM}$	0.001685539
$MTTR_{DB}$	0.001685531
$MTTR_{HW/OS}$	0.000732489
$MTTR_{RTR}$	0.000273396
$MTTF_F$	0.000080051
$MTTR_F$	0.000045057
$MTTF_{RTR}$	0.000040986

Table 8 shows that by decreasing the minimum number of floors working, the IoT floor environment's sensitivity index drops to the bottom of the sensitivity ranking. The

decrease occurs because a single failure in just one of the many IoT components will not bring the entire system down. The same occurs with the new relation between the MTTR of the IoT environment. Figure 28 presents the relation between the MTTF of the IoT environment and the smart building system availability. When the MTTF reaches its smallest value, below 7500 h, the availability decreases to below 99.44 %. The opposite occurs when the MTTF reaches to its most significant value, above 25000 h. The availability grows up to above 99.46 %.

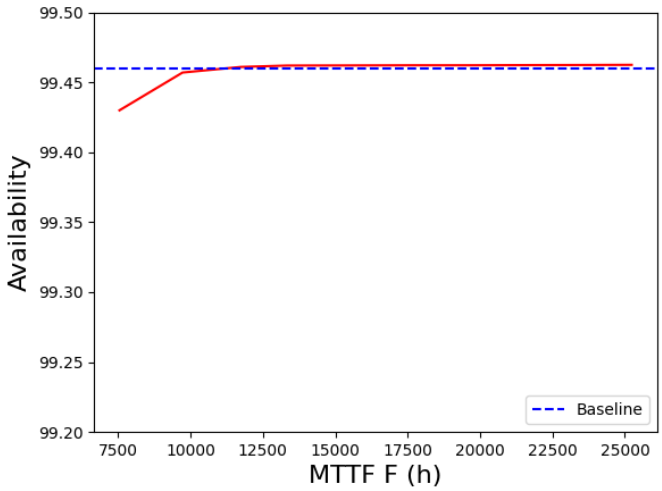


Figure 28 – New sensitivity of system availability with respect to $MTTF_F$ (IoT floor environment failure)

Therefore, the $MTTF_{HW/OS}$ is the component that significantly impacts this new scenario's availability. This new scenario shows that the management system's redundancy should be implemented with a high priority to improve availability. Figure 29 presents the relation between the hardware and operating system's MTTF and the smart building system availability. When the MTTF reaches its smallest value, below 1000 h, the availability decreases to below 99.29 %. The opposite occurs when the MTTF reaches to its most significant value, above 3800 h. The availability grows up to above 99.6 %.

The author also performed the sensitivity analysis in the local infrastructure with redundancy. The results presented in Table 9 show that the IoT environment is the component that most affects availability, considering both MTTF and MTTR. Thus, we can observe that the IoT environment impacts the entire smart building system's availability when all floors are working.

The sensitivity analysis performed, considering at least fourteen floors, must be working. Table 10 shows that by decreasing the minimum number of floors working, the sensitivity index of the IoT environment drops to the bottom of the ranking, as it also occurred in the analysis for the environment without redundancy. In this scenario, the MTTFs of database and device manager are the components with the highest impact on availability.

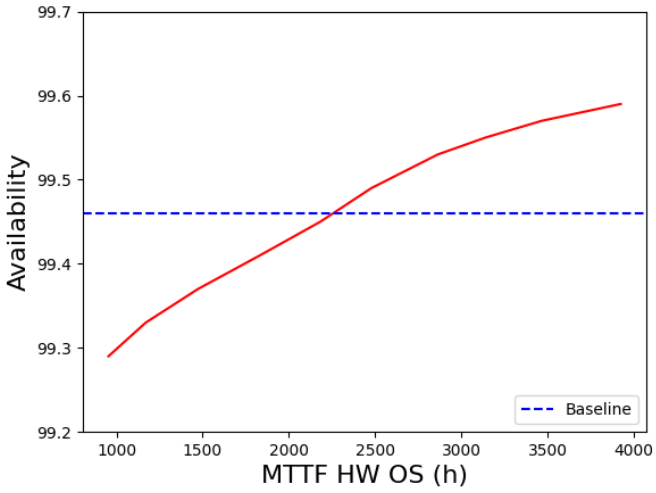


Figure 29 – Sensitivity of system availability with respect to $MTTF_{HW/OS}$ (hardware and operating system failure)

Table 9 – Sensitivity analysis for local infrastructure with redundancy

Component	**Sensitivity Index**
$MTTF_F$	0.006569700
$MTTR_F$	0.004924340
$MTTF_{DB}$	0.000962114
$MTTF_{DM}$	0.000962103
$MTTR_{DB}$	0.000658652
$MTTR_{DM}$	0.000658627
$SWITCH$	0.000109401
$MTTF_{HW/OS}$	0.000099881
$MTTR_{HW/OS}$	0.000090854
$MTTR_{RTR}$	0.000040958
$MTTF_{RTR}$	0.000027342

It is noteworthy that the $MTTF_{HW/OS}$ is not so crucial in this scenario, as there is no redundancy applied.

Figure 30 presents the relation between the MTTF of the database and the smart building system availability. When this parameter reaches its lowest value, just below $800\,h$, the overall system availability decreases to its lowest value, approximately $99.43\,\%$. The opposite occurs when the parameter reaches its highest value, above $2,000\,h$. The availability reaches its highest value, about $99.65\,\%$. The same occurs with the MTTF device manage, and the smart building system availability, as shown in Figure 31.

The author also performed a sensitivity analysis in the cloud computing infrastructure. The results presented in Table 11 show that the IoT environment is the component that most affects availability, considering both MTTF and MTTR. In all the analyzed infras-

Table 10 – Sensitivity analysis to local infrastructure with redundancy and requirement of 14 out of 15 floors working

Component	Sensibility Index
$MTTF_{DB}$	0.000962112
$MTTF_{DM}$	0.000962102
$MTTR_{DB}$	0.000658653
$MTTR_{DM}$	0.000658631
$SWITCH$	0.000109401
$MTTR_{HW/OS}$	0.000099885
$MTTF_{HW/OS}$	0.000090851
$MTTF_{F}$	0.000080089
$MTTR_{F}$	0.000045072
$MTTF_{RTR}$	0.000040986
$MTTR_{RTR}$	0.000273396

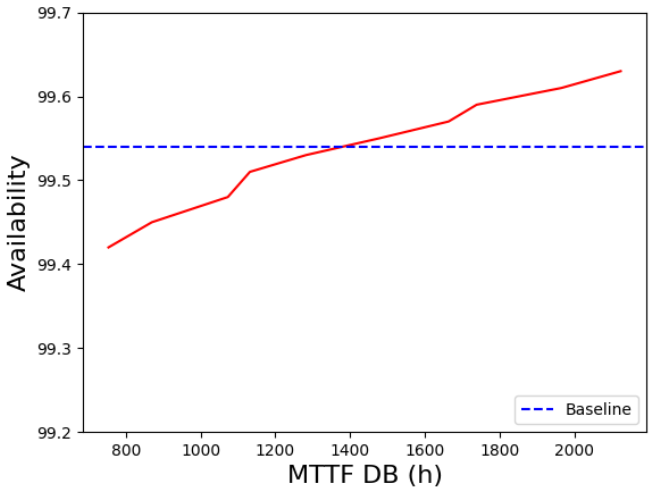


Figure 30 – Sensitivity of system availability with respect to $MTTF_{DB}$ (database component failure)

tructures, the MTTF and MTTR of the IoT environment were the components that most impacted availability considering that the fifteen floors of the building are working. This IoT importance is due to the number of components added to each floor of the building, and the requirement of all of them being functional simultaneously.

The sensitivity analysis performed also considered at least fourteen floors are working. Table 12 shows that the IoT environment has not the same importance in this scenario, as expected when we look to the previous scenarios' behavior. The $MTTF_{HW/OS/DC}$ is the component that causes the most significant effect on system availability, followed by the MTTRs of the device manager and the database. This effect on the system availability is significantly different from the corresponding scenario with local infrastructure, shown in Table 10, where the MTTRs of those components were less critical than the MTTFs.

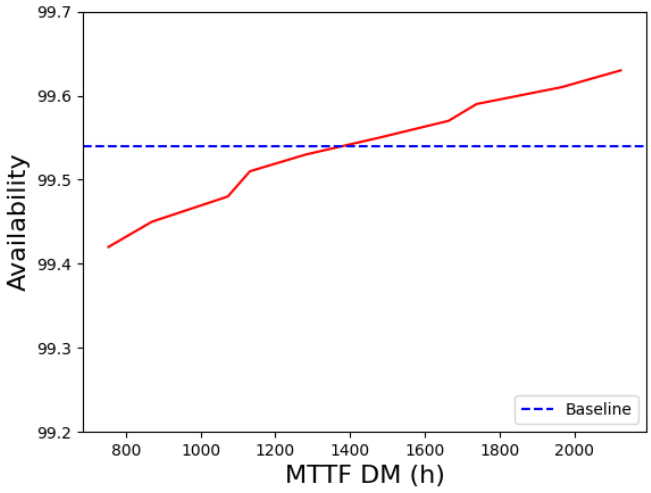


Figure 31 – Sensitivity of system availability with respect to $MTTF_{DM}$ (device manage component failure)

Table 11 – Sensitivity analysis to cloud computing infrastructure

Component	**Sensibility Index**
$MTTF_F$	0.006569700
$MTTR_F$	0.004924340
$MTTF_{HW/OS/DC}$	0.002186142
$MTTR_{DM}$	0.001897046
$MTTR_{DB}$	0.001897037
$MTTF_{DB}$	0.001845361
$MTTF_{DM}$	0.001843284
$MTTR_{RTR}$	0.000273383
$MTTF_{CTRL}$	0.000266536
$SWITCH$	0.000241295
$MTTR_{RTR}$	0.000040979
$MTTR_{CTRL}$	0.000030054
$MTTR_{HW/OS/DC}$	0.000006405

Therefore, the development of efficient techniques and tools for failure detection and recovery targeted at these software components should be among the highest priority actions.

Figure 32 presents the relation between the MTTF of hardware, operating system, docker, and smart building system availability. When the MTTF reaches its smallest value, below 480 h, the availability decreases to below 99.35 %. The opposite occurs when the MTTF reaches to its most significant value, above 1435 h. The availability grows up to above 99.55 %.

Figure 33 presents the relation between the MTTR of the device manager and the smart building system availability. When this parameter reaches its lowest value, just

Table 12 – Sensitivity analysis to cloud computing infrastructure and requirement of 14 out of 15 floors working

Component	Sensibility Index
$MTTF_{HW/OS/DC}$	0.002186142
$MTTR_{DM}$	0.001897046
$MTTR_{DB}$	0.001897037
$MTTF_{DB}$	0.001845361
$MTTF_{DM}$	0.001843284
$MTTR_{RTR}$	0.000273383
$MTTF_{CTRL}$	0.000266536
$SWITCH$	0.000241295
$MTTF_{F}$	0.000080091
$MTTR_{F}$	0.000045035
$MTTF_{RTR}$	0.000040982
$MTTR_{CTRL}$	0.000030054
$MTTR_{HW/OS/DC}$	0.000006405

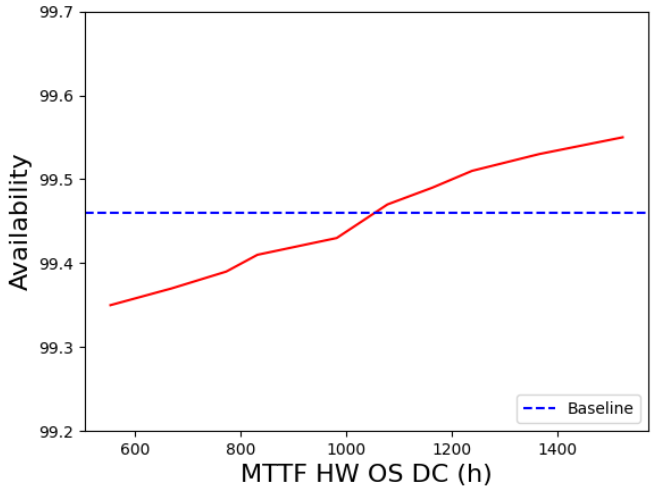


Figure 32 – Sensitivity of system availability with respect to $MTTF_{HW/OS/DC}$ (hardware, operating system, and docker failure)

0.5 h, the overall system availability reaches its highest value, about 99.53 %. The opposite occurs when the parameter reaches its highest value, about 1.4 h. The availability decreases to its lowest value, about 99.39 %. The same occurs with the MTTR database and the smart building system availability, as shown in Figure 34.

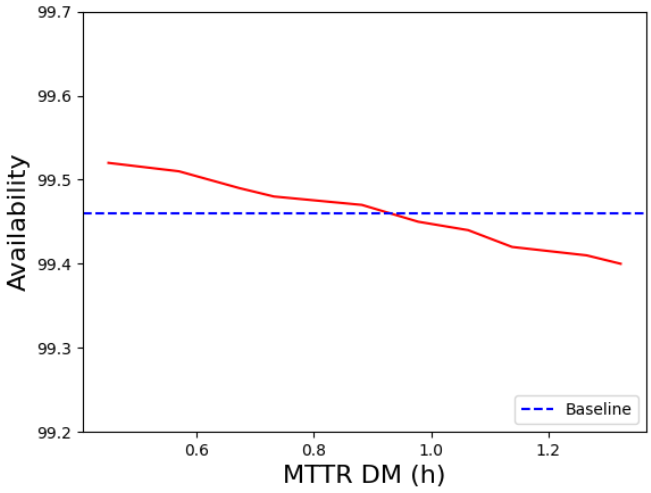


Figure 33 – Sensitivity of system availability with respect to $MTTR_{DM}$ (device manage repair)

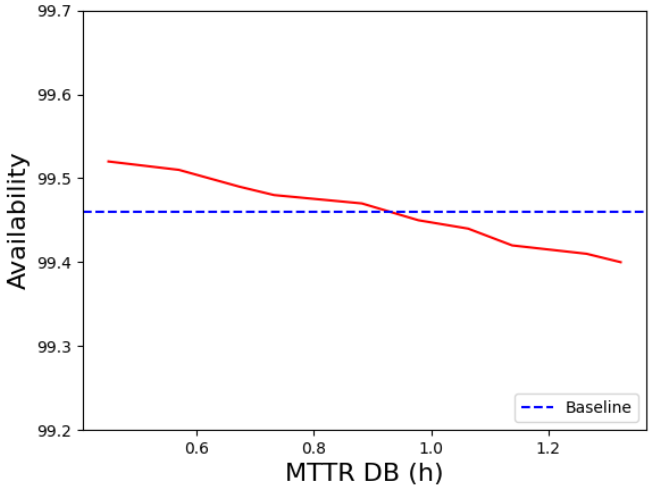


Figure 34 – Sensitivity of system availability with respect to $MTTR_{DB}$ (database repair)

5.2 SMART PHOTOVOLTAIC SYSTEM

This work used a grid-tie and a hybrid photovoltaic system to evaluate the photovoltaic energy management system proposed in Section 4.2.2. Section 5.2.1, presents two distinctions case study. The proposed approach in this paper takes into account a smart building system for housing. It is worth mentioning that we are considering a building in Recife, Pernambuco, Brazil. Therefore, all variables that influence photovoltaic projects such as, heatstroke, fares, and temperature, are according to the region, but it can easily apply to other cities and countries.

This work is considered a building with fifteen floors, containing four apartments per floor. In each apartment have three smart plugs. These sensors aim at capturing data

Table 13 – Annual electricity Peak Demand (KW)

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
Cooling	182.20	181.17	184.03	182.42	175.51	169.83	167.39	166.18	171.06	172.99	180.88	184.04
Lighting	117.91	117.91	117.91	117.91	117.91	117.91	117.91	117.91	117.91	117.91	117.91	117.91
Equipment	76.509	76.509	76.509	76.509	76.509	76.509	76.509	76.509	76.509	76.509	76.509	76.509
Fans	11.776	11.754	11.554	11.756	11.330	11.255	10.909	10.851	11.091	11.091	11.709	11.810
Total	388.40	387.35	390.01	388.60	381.26	375.51	372.73	371.45	376.58	378.61	387.02	390.31

about the energy consumption of the electrical equipment. Also, there are smart sensors to measure the energy consumed from the lights. Each apartment has a sensor of this type, as well as in the hallway of each floor. The management system uses these data to predict when the solar power system can be used instead of electrical energy. The router is responsible for sent data to the management system.

5.2.1 Photovoltaic systems

This Section defines the approaches used for the autonomous energy management system when we adopt a hybrid and a grid-tie photovoltaic system. More details below.

5.2.1.1 Hybrid solar system

The central idea to adopt a solution hybrid to the photovoltaic system is to determine the period that solar energy will use. In this work, we determined that the autonomous power management system will activate the power supply through the batteries only during peak hours, when possible. Peak hours are, by definition, the period of the day when peak demand for electricity occurs. In Recife, due to the population's habits, this time usually occurs between $5:30\,pm$ and $8:30\,pm$, that is, we do not measure the consumption of the whole day; we are only taking into consideration the period between $5:30\,pm$ and $8:30\,pm$, i.e., the peak hours.

To determine the solar power system's size is necessary to know the amount of energy we intend to consume. The author used the OpenStudio software (GUGLIELMETTI; MACUMBER; LONG, 2011), in which we model a building with parameters previously mentioned. However, we inserted some electrical equipment and their usage time to determine the energy that it spends with cooling, interior lighting, interior equipment, and fans. The model of the building considers the climate zone of Recife, State of Pernambuco, Brazil. The simulation results show in Table 13 for the peak period.

Figure 19 shows the components set in an off-grid or hybrid solar system. The system is composed of batteries, solar inverters, charge controllers, and solar panels. To determine the battery bank needs to know the amount of average energy consumed daily. The

daily consumption calculates through Equation 5.2, where $4588.05\,kWh$ is the annual consumption of energy in peak time, and 365 the number of days the year.

$$E_C = \frac{4588,05\,kWh}{365} \tag{5.2}$$

The energy that we will store in the battery bank can calculate through Equation 2.22 to determine the energy total stored is needed to define the limit of discharge of the battery bank, we choose $50\,\%$ as a limit of discharge. Thereby, we have a balance between energy consumption and the life cycle of the batteries.

To determine the number of batteries is necessary to define the system's operating voltage. We defined the operating voltage as $48\,V$. Therefore, the batteries connected in series should produce $48\,V$. Each battery in our system has $12\,V$, and $220\,Ah$. This work used four batteries in series to produce the voltage needed. The battery bank load capacity calculates through Equation 2.24, where the system's operating voltage is $48\,V$. Equation 2.25 determines the number of batteries connected in parallel, where $220\,Ah$ is the battery load capacity of each one.

To ensure the necessities of the system, we round the result to the next integer value. Therefore, we have a total of twelve batteries. The designer connected three ensemble batteries in parallel, with four batteries each. To determine the number of solar panels, we used the daily heat stroke. In this method, only it is valid when considered the use of charge controllers with MPPT resources. The first step is to know the module characteristics; the module chosen has $1.98\,m$ tall, $0.992\,m$ wide, and $17\,\%$ efficiency. The energy produced daily by one solar panel can calculate through Equation 2.26, where $4.28\,KWh$ is the average daily solar radiation in June. As June has the smaller daily average of solar radiation of the year in Brazil, our system will produce at least what we expect for that month Table 14 shows the daily average of solar radiation in the whole year.

The modules number needed can be calculated through Equation 5.3. The result shows that we need 8.8 panels to supply the energy consumed daily. Therefore, considering the energy produced and charge controller output voltage, we adopted nine photovoltaic panels. It is worth point out that the chosen module has $37.3\,V$ output voltage.

$$N_M = \frac{E_C}{E_P} \tag{5.3}$$

The author used the operation voltage and the electric current provides by the solar modules to choose the charge controller. Equation 5.4 calculates the electric current, where 3 is the amount of the panels connected in parallel, and $9.33\,A$ is the module short circuit current standard test conditions. Therefore, we chose one charge controller that supports an input voltage of $111.9\,V$, output voltage $48\,V$, and maximum current $27.99\,A$.

$$C_M = 3 \times 9.33\,A \tag{5.4}$$

Table 14 – Daily heat stroke, monthly average

Month	[kWh/m2.daily]
Jan	5.84
Feb	5.94
Mar	5.88
Apr	5.15
May	4.47
Jun	**4.18**
Jul	4.30
Aug	5.03
Sep	5.45
Oct	5.78
Nov	6.05
Dec	6.06

The voltage of specified input and output is determinant to choose the solar inverter. Besides, the inverter should support the total power of the equipment and lights of the building. Equation 5.5 calculates the total power, where $12570\,Wh$ is the energy consumed in peak time daily, and 3 is the duration (in hours) of the peak time in Recife, Brazil.

$$P_T = \frac{12570\,W}{3\,h} \tag{5.5}$$

Therefore, we chose one inverter that supports the output power $4190\,W$, $48\,V$ input voltage, and $220\,V$ output voltage. In summary, all components of the hybrid photovoltaic system are:

- 9 panels photovoltaic;
- 12 batteries;
- 1 charge controller;
- 1 inverter

5.2.1.2 Grid-tie solar power system

The main idea to adopt a grid-tie solution for a photovoltaic system is to make the most of the energy generated in real-time. Therefore, we assume that when the photovoltaic panels' energy is sufficient to supply the building's demand, the autonomous management system will disconnect the connection to the utility grid and connect the connection with the photovoltaic system. It means the energy produced supplies the building when energy tariff is cheaper ($8:00\,am - 5:30\,pm$). Figure 18 shows the set components that composed

a grid-tie photovoltaic system. The system is composed of a solar inverter, string boxes, solar panels, and an AC isolator.

A way to determine the energy amount that we want to produce is to consider the space available for the installation of solar panels. In this work, we considered that space available for the installation of solar panels support nine panels. The grid-tie solar power system has an inverter with MPPT. Therefore, the best way to calculate the energy produced by solar panels is by using the daily heat stroke method. It is necessary to know the solar panel model used in the system. We chose the same model used in the hybrid solar power system. The same model will help us to have a better comparison between grid-tie and off-grid solar power systems. Equation 2.26 calculates the energy produced daily by all solar panels.

The first step to inverter sizing is to know if we can connect all panels in series. Equation 5.6calculates the maximum output voltage of the string, where $45.6\,V$ is the panel open-circuit voltage in STC mode, and 9 is the panels' amount. Finally, we used $10\,\%$ as a security factor in ensuring the proper functioning of the component. The last step is to know if the insert supports the panels' max power. Equation 5.7calculates the max power, where $330\,W$ is the panel power, and 9 is the number of panels.

$$V_{OC,string} = 45.6\,V \times 9 \times 1.1 \tag{5.6}$$

$$P_{max} = 330\,W \times 9 \tag{5.7}$$

Therefore, the invert chosen should support at least an input voltage of $451.44\,V$, and an input power of $2970\,W$. The author chooses the string box and AC isolator using the same values obtained to size the inverter. In summary, all components of the hybrid photovoltaic system are:

- 9 panels photovoltaic;
- 1 string boxes;
- 1 AC isolator;
- 1 inverter

5.2.2 Results obtained with the proposed models

This case study focuses on assessing the availability of an intelligent energy management system and its impacts. To perform this, we evaluated the autonomous energy management system in three aspects; availability, acquisition cost, and financial return. Therefore, this work compares an autonomous energy management system applied to a grid-tie photovoltaic system and a hybrid photovoltaic system.

The availability models presented in Section 4.2.3 are used to assess availability for the autonomous energy management system. It is worth mentioning that availability was calculated for each autonomous management system. Some parameter values are required to evaluate the models. These values can be found in the literature. The MTTF and MTTR values of each component are extracted from (DANTAS et al., 2012; COOPER; FARRELL, 2007; CISCO..., 2020; KIM; MACHIDA; TRIVEDI, 2009; ROHOUMA; MOLOKHIA; ESURI, 2007; ZINI; MANGEANT; MERTEN, 2011; CARRASCO; NARVARTE; LORENZO, 2013). The input values for each of the previous models can be seen in Table 15.

Table 15 – Input values for SPN

Component	**MTTF (hours)**	**MTTR (hours)**
Hardware	61320	8
Operating System	1440	1
Sensors/Actuators	300000	1
Docker	2900	1
Controller Manager	700	1
Data Base	1440	1
Device Manager	700	1
Router	26000	8
Inverter	24820	8
Charge Controller	70080	8
Solar Panel	219000	8
Battery	47829	8

5.2.2.1 Autonomous energy management system: Availability

This section presents the availability analysis of the autonomous energy management system. To perform this, we compared the availability of an autonomous energy management system with a grid-tie photovoltaic system with a hybrid photovoltaic system. Table 16 shows the results of the dependability metrics of the autonomous energy management system of a building. To evaluate the availability was considered 15 floors, and that all the components on all floors have to be working. Besides, all components of the photovoltaic system and management infrastructure have to be working. The results showed that both autonomous energy management system downtime considering fifteen floors is more than $36\,h$ for the whole system. Therefore, it implies two days of unavailability approximately. It shows that the system will supply the energy demand in 363 days of the year. We obtained values similar to both systems; this occurs due to the high MTTF values of the solar power system components. Therefore, the solar power system has few impacts on the availability of the whole system.

Table 16 – Availability metrics results to whole system

Metric	**hybrid**	**grid-tie**
Uptime	8722.27 h	8722.29 h
Downtime	37.72 h	37.70 h
Availability	99.5694 %	99.5696 %

5.2.2.2 Autonomous energy management system: Deploying cost

This Section describes the deploying cost of an autonomous energy management system. The analysis considers the three subsystems (building components, management infrastructure, and photovoltaic system components). Table 17 shows the results of the acquisition cost to the smart solar power system to a building. The building IoT subsystem has an acquisition cost of the $2,312.93\,US\$$. The subsystem is composed of the 197 sensors/actuators and 15 routers. The solar power subsystem has an acquisition cost of the $6,554.28\,US\$$ to the hybrid system. The hybrid system comprises nine solar panels, one inverter, 12 batteries, one charge controller, and three sensors. The grid-tie solar power system has an acquisition cost of the $3,983.05\,US\$$. The subsystem comprises nine solar panels, one inverter, one string box, one AC isolator, and three sensors. The management subsystem has an acquisition cost of the $4,808.50\,US\$$. The total hybrid solar power system acquisition cost to the whole system is $13,656.24\,US\$$ o supply the needed demand energy in peak time. The grid-tie solar power system acquisition cost to the whole system is $12,742.10\,US\$$. Figure 35 also summarizes the values in Table 17.

Table 17 – Equipment cost

Components	**hybrid US$**	**grid-tie US$**
IoT components	2,293.46	2,293.46
Management infrastructure	4,808.50	4,808.50
solar components	6,554.28	3,983.05
Total	**13,656.24**	**11,085.01**

5.2.2.3 Autonomous energy management system: Cost reduction

This Section presents the cost reduction obtained by adopting the autonomous energy management system for both cases. Table 18 the energy reduction by the two autonomous energy management systems, with the respective cost. As mentioned in the previous Sections, the autonomous energy management system with a hybrid photovoltaic system will activate during peak hours (5:30 pm to 8:30 pm). Furthermore, the autonomous energy management system with a grid-tie photovoltaic system will activate during the "outside the range" period (8:30 am to 5:30 pm). Therefore, to know the real cost reduction

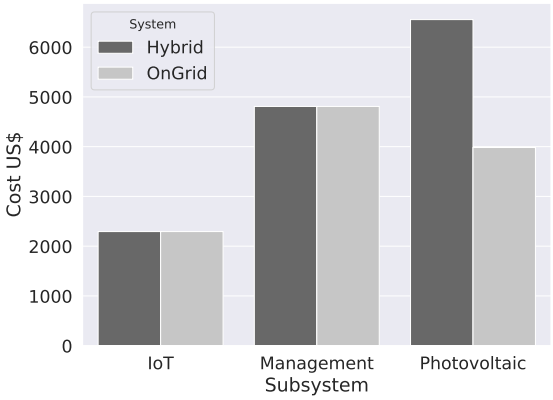


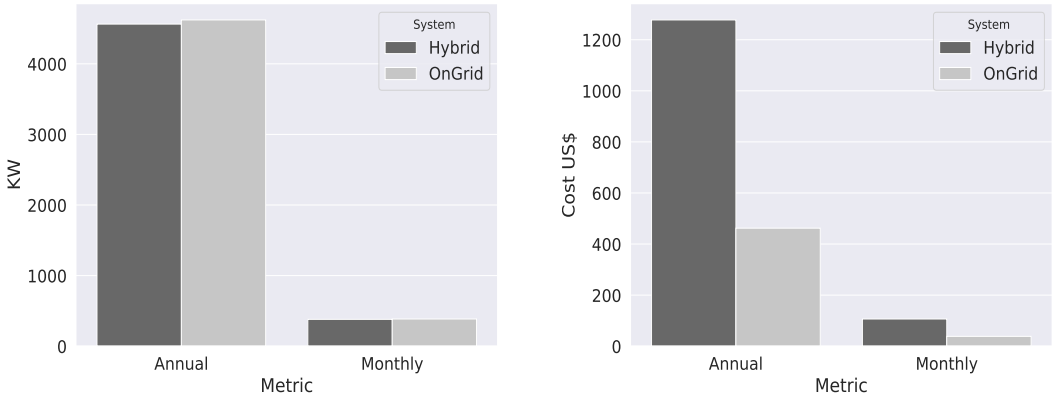
Figure 35 – Equipment cost

with energy, it was considered the fare amount in Recife, Pernambuco, Brazil in peak time, and "outside of range". However, the currency using in this work is the American dollar, and the fare value is 1.1584 R\$ to peak time, and 0.4639 R\$ to “outside of range" (Brazilian currency) per kilowatt. Therefore, it was necessary to convert the fare value to the American dollar. According to the dollar price from December 9, 2019, the fair values are 0.28 US\$ to peak time. Ans 0.10 US\$ to ”outside of range” by the kilowatt. Figure 36 also summarizes the values in Table 18.

Table 18 – Energy produced/Reduction cost

	hybrid		**grid-tie**	
Metric	**Average - KWh**	**US$**	**Average - KWh**	**US$**
Annual	4562.91	1277.61	4621.78	462.17
Monthly	380.24	106.46	385.14	38.51
Daily	12.67	3.54	12.83	1.28

The autonomous energy management system with a photovoltaic hybrid system can replace the electrical energy with solar energy in 363 days of the year (availability). The energy reduction cost is $1,277.61\,US$\$ annually, adopting this system. However, itis necessary to consider the acquisition cost to know the real gain with autonomous energy management systems. Considering the acquisition cost and the value saved with autonomous energy management system adoption, we have a total of 10.68 years to offset all the investment. However, according to to (VILLALVA, 2012), the battery cycle life is from 5 to 10 years, then it is necessary to add to the acquisition cost another battery bank. Therefore, we have a real total of 13.46 years to offset all the investment. The autonomous energy management system with a grid-tie photovoltaic system also can be used in 363 days of the year. The system deployment reduces the value spent with energy in 462.17 US\$ annually. However, it is necessary to consider the acquisition cost to know the real gain with autonomous energy management systems adoption. Then, considering the acquisition cost



(a) a (b) b

Figure 36 – (a) KW consumed (b) Value spent

and the value saved with smart energy system adoption, we have a total of 23.98 years to offset all the investment. Figure 37 shows a total of years to offset all investments in both systems.

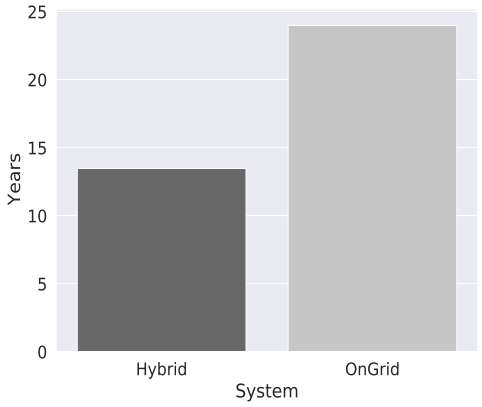


Figure 37 – Years to offset all the investment

5.3 FINAL REMARKS

This chapter presented two case studies that show the proposed models and the availability impact in smart building management. The first case study compares two local infrastructures with redundancy and without redundancy. The second case study compares a local infrastructure with a cloud computing infrastructure. The third case study identifies the components that most affect the availability. The fourth case study compares the availability between an on-grid and a hybrid smart energy system. The last case study analyzes the availability impact in the energy cost annual of a smart building in both systems.

6 CONCLUSIONS AND FUTURE WORK

Through the technological advances of recent years and the popularization of smart devices, society increasingly adopts the internet, paving the way for the explosion of smart cities. This concept allows data sharing between computational machines and everyday objects. One of the areas within smart cities is smart building. However, this computational paradigm's constraints, such as the instability of wireless connections, may lead to the unavailability of the service provided. Designers and administrators need to design and manage their systems to keep the service highly available.

This study presented an essential and significant hierarchical modeling for generic and specific smart building systems. Thus, this work carried out an availability assessment study of an IoT environment to provide smart building services. To achieve improvements in availability, as well as assisting the planning of infrastructures within the studied context.

We investigated three different architectures to verify the feasibility of each concerning availability and cost-effectiveness for generic systems. The results of this study served to build the architecture for a smart energy system. This work proposed the representation of these architectures through hierarchical modeling with SPN and mathematical equations.

Initially, we investigated the first architecture defined as Base Architecture, which has the main objective of modeling and validating the proposed service. The proposed system's annual uptime with local management is about 9 hours larger with redundancy than without redundancy. Both structures have two-nines (i.e., about 99.0 %) in the availability metric. This availability means the system is available in almost 362 days of the year. We also compared the local redundancy infrastructure with a cloud computing redundancy infrastructure. In this analysis, we noted that the local infrastructure is better than the cloud computing infrastructure when there is only one building. When we considered more than one building, the cloud computing infrastructure has a more massive availability than the local infrastructure. We performed a sensitivity analysis of our models to identify bottlenecks in the system. This analysis showed that the IoT environment components are at the top of the sensitivity ranking for all infrastructure setups considering that all fifteen floors must be working. However, considering all the infrastructures analyzed and the variation of the minimum number of functional floors required, we can see that database and device management have always been among the components that most affect availability in all scenarios. To increase the system availability, some strategies can be made together or separately, which will depend on the budget available to invest and the existence of more efficient equipment, software, or strategies to deploy. One alternative is acquiring software for the device manager and database that offers more significant values of MTTF. Another alternative is investing in the maintenance team, fault detection tools,

and spare components to decrease the time to repair the applications.

Finally, this work applied the redundant local infrastructure for a smart photovoltaic system. The study results showed annual uptime of the proposed environment with an on-grid solar power system similar to the hybrid one, about $8710\,h$. This availability obtained means the system is available in almost 363 days of the year. The work also compared the whole system's deployment cost between an on-grid and hybrid solar power system. In this analysis, the author noted that the whole system with an on-grid solar power configuration is cheaper than the whole system with the hybrid solar power one. This difference in value occurs because the on-grid solar power system has fewer components than the hybrid solar power one. This value creates a difference in deployment cost of $2,571.23\,US\$$. However, the hybrid solar power system has less time to recover investment than the on-grid solar one. It happens due to the kilowatt value at a peaky time (hybrid power system up) is more expensive than the kilowatt value "outside of range" (on-grid solar power system up), which creates a difference in the value saved with the electricity of $815.44\,US\$$ annually. It is worth mentioning that we considered the best-case scenario to charge the battery bank for around ten years.

Therefore, the approach used and the results obtained in this research contribute to a common framework for planning future IoT solutions in smart building environments that use resources similar to those analyzed in this work.

6.1 CONTRIBUTIONS

In summary, the main contributions of this dissertation are:

- The work proposed models through an approach using hierarchical SPN modeling and mathematical equation. These models represent an IoT service infrastructure in a smart building. Through these models, it was possible to analyze dependability metrics of this service such as availability and annual downtime;

- The parametric sensitivity analysis helps to improve system availability. It is possible to verify through availability bottlenecks and propose variations in the proposed architectures to improve service availability. Besides, the sensitivity analysis result can use as a guide other works that analyze similar infrastructures;

- This work proposed two extensions of Base Architecture as a solution to increase and optimize service availability. The author obtained better results applying redundancy to the most critical components. It was also possible to evaluate the impact of this redundancy on the increase of system availability. Where cloud computing architecture provided a better level of availability, considering more than one building. The architectures and their respective models that represent them can be used in architectural extensions, within the same environment analyzed, in future works;

Thus, the objectives described in Section 1.2 were achieved and in addition to the contributions mentioned above, an article was published based on the results of this research:

- Araujo, E., Dantas, J., Matos, R., Pereira, P., Maciel, P. (2019, October). Dependability Evaluation of an IoT System: A Hierarchical Modelling Approach. In 2019 IEEE International Conference on Systems, Man and Cybernetics (SMC) (pp. 2121-2126). IEEE.

6.2 FUTURE WORK

In this section, we list some issues that can be investigated in future work:

- Analyze other dependability attributes in the same proposed environment, such as Security, Reliability, and Integrity;
- Performance studies on the proposed architectures. Performance analysis, for example, may assess in different scenarios that which best presents the relationship between availability and performance;

REFERENCES

ABOHAMAMA, A.; ALRAHMAWY, M.; ELSOUD, M. A. Improving the dependability of cloud environment for hosting real time applications. *Ain Shams Engineering Journal*, Elsevier, v. 9, n. 4, p. 3335–3346, 2018.

AMAZOM. *AWS Total Cost of Ownership (TCO) Calculators*. 2019. Available in: https://aws.amazon.com/tco-calculator/?nc2=h$_q l_p r$.

ANDRADE, E.; NOGUEIRA, B. Dependability evaluation of a disaster recovery solution for iot infrastructures. *The Journal of Supercomputing*, Springer, p. 1–22, 2018.

ARAUJO, C.; SILVA, F.; COSTA, I.; VAZ, F.; KOSTA, S.; MACIEL, P. Supporting availability evaluation in mcc-based mhealth planning. *Electronics Letters*, IET, v. 52, n. 20, p. 1663–1665, 2016.

ARMBRUST, M.; FOX, A.; GRIFFITH, R.; JOSEPH, A. D.; KATZ, R.; KONWINSKI, A.; LEE, G.; PATTERSON, D.; RABKIN, A.; STOICA, I. et al. A view of cloud computing. *Communications*, ACM, v. 53, n. 4, p. 50–58, 2010.

AVIZIENIS, A.; LAPRIE, J.-C.; RANDELL, B. et al. *Fundamental concepts of dependability*. [S.l.]: University of Newcastle upon Tyne, Computing Science, 2001.

AVIZIENIS, A.; LAPRIE, J.-C.; RANDELL, B.; LANDWEHR, C. Basic concepts and taxonomy of dependable and secure computing. *IEEE transactions on dependable and secure computing*, IEEE, v. 1, n. 1, p. 11–33, 2004.

AZAIEZ, M.; CHAINBI, W.; GHEDIRA, K. Hybrid fault tolerance model for cloud dependability. In: IEEE. *2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*. [S.l.], 2019. p. 2436–2444.

BENEVOLO, C.; DAMERI, R. P.; D'AURIA, B. Smart mobility in smart city. In: *Empowering Organizations*. [S.l.]: Springer, 2016. p. 13–28.

BIANCHI, G. Performance analysis of the ieee 802.11 distributed coordination function. *IEEE Journal on selected areas in communications*, IEEE, v. 18, n. 3, p. 535–547, 2000.

BOLCH, G.; GREINER, S.; MEER, H. D.; TRIVEDI, K. S. et al. *Queueing networks and Markov chains: modeling and performance evaluation with computer science applications*. [S.l.]: John Wiley & Sons, 2006.

CADAVID, H.; GARZÓN, W.; PÉREZ, A.; LÓPEZ, G.; MENDIVELSO, C.; RAMÍREZ, C. Towards a smart farming platform: From iot-based crop sensing to data analytics. In: SPRINGER. *Colombian Conference on Computing*. [S.l.], 2018. p. 237–251.

CARRASCO, L. M.; NARVARTE, L.; LORENZO, E. Operational costs of a 13,000 solar home systems rural electrification programme. *Renewable and Sustainable Energy Reviews*, Elsevier, v. 20, p. 1–7, 2013.

CASSANDRAS, C. G.; LAFORTUNE, S. et al. *Introduction to discrete event systems*. [S.l.]: Springer Science & Business Media, 2009.

CATARINUCCI, L.; DONNO, D. D.; MAINETTI, L.; PALANO, L.; PATRONO, L.; STEFANIZZI, M. L.; TARRICONE, L. An iot-aware architecture for smart healthcare systems. *IEEE Internet of Things Journal*, IEEE, v. 2, n. 6, p. 515–526, 2015.

CHARKI, A.; LOGERAIS, P.-O.; BIGAUD, D.; KÉBÉ, C. M.; NDIAYE, A. Lifetime assessment of a photovoltaic system using stochastic petri nets. *International Journal of Modelling and Simulation*, Taylor & Francis, v. 37, n. 3, p. 149–155, 2017.

CHEBOTKO, A.; KASHLEV, A.; LU, S. A big data modeling methodology for apache cassandra. In: IEEE. *Big Data (BigData Congress), 2015 IEEE International Congress on*. [S.l.], 2015. p. 238–245.

CISCO Packet Tracer - Networking Simulation Tool. 2020. Disponível em: <https://www.netacad.com/courses/packet-tracer>.

COCCHIA, A. Smart and digital city: A systematic literature review. In: *Smart city*. [S.l.]: Springer, 2014. p. 13–43.

COMMITTEE, I. S. C. et al. Ieee standard glossary of software engineering terminology (ieee std 610.12-1990). los alamitos. *CA: IEEE Computer Society*, v. 169, 1990.

COOK, D. J.; DAS, S. K. How smart are our environments? an updated look at the state of the art. *Pervasive and mobile computing*, Elsevier, v. 3, n. 2, p. 53–73, 2007.

COOPER, T.; FARRELL, R. Value-chain engineering of a tower-top cellular base station system. In: IEEE. *Vehicular Technology Conference, 2007. VTC2007-Spring. IEEE 65th*. [S.l.], 2007. p. 3184–3188.

Dantas, J.; Matos, R.; Araujo, J.; Maciel, P. An availability model for eucalyptus platform: An analysis of warm-standy replication mechanism. In: *2012 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. [S.l.: s.n.], 2012. p. 1664–1669.

DANTAS, J.; MATOS, R.; ARAUJO, J.; MACIEL, P. Models for dependability analysis of cloud computing architectures for eucalyptus platform. *International Transactions on Systems Science and Applications*, v. 8, p. 13–25, Dec. 2012. ISSN 2051-5642.

DANTAS, J.; MATOS, R.; ARAUJO, J.; MACIEL, P. Eucalyptus-based private clouds: availability modeling and comparison to the cost of a public cloud. *Computing*, Springer, v. 97, n. 11, p. 1121–1140, 2015.

FAHRENBRUCH, A.; BUBE, R. *Fundamentals of solar cells: photovoltaic solar energy conversion*. [S.l.]: Elsevier, 2012.

GAZIS, V.; GOERTZ, M.; HUBER, M.; LEONARDI, A.; MATHIOUDAKIS, K.; WIESMAIER, A.; ZEIGER, F. Short paper: Iot: Challenges, projects, architectures. In: IEEE. *2015 18th International Conference on Intelligence in Next Generation Networks*. [S.l.], 2015. p. 145–147.

GERMAN, R. et al. *Performance analysis of communication systems with non-Markovian stochastic Petri nets*. [S.l.]: John Wiley & Sons, Inc., 2000.

GHOSH, R.; LONGO, F.; FRATTINI, F.; RUSSO, S.; TRIVEDI, K. S. Scalable analytics for iaas cloud availability. *IEEE Transactions on Cloud Computing*, IEEE, v. 2, n. 1, p. 57–70, 2014.

GIRAULT, C.; VALK, R. *Petri nets for systems engineering: a guide to modeling, verification, and applications*. [S.l.]: Springer Science & Business Media, 2013.

GREENGARD, S. *The internet of things*. [S.l.]: MIT press, 2015.

GUGLIELMETTI, R.; MACUMBER, D.; LONG, N. *OpenStudio: an open source integrated analysis platform*. [S.l.], 2011.

HALLER, S. Internet of things: an integral part of the future internet. *Prague: SAP Research*, 2009.

HAMBY, D. et al. A review of techniques for parameter sensitivity analysis of environmental models. *Environmental monitoring and assessment*, Springer, v. 32, n. 2, p. 135–154, 1994.

HAN, D.-M.; LIM, J.-H. Smart home energy management system using ieee 802.15. 4 and zigbee. *IEEE Transactions on Consumer Electronics*, IEEE, v. 56, n. 3, 2010.

JAIN, R. et al. *The art of computer systems performance analysis: techniques for experimental design, measurement, simulation, and modeling*. [S.l.]: John Wiley & Sons, 1990.

JENSEN, K.; ROZENBERG, G. *High-level Petri nets: theory and application*. [S.l.]: Springer Science & Business Media, 2012.

JOHNSTON, E. W.; HANSEN, D. L. Design lessons for smart governance infrastructures. *Transforming American governance: Rebooting the public square*, p. 197–212, 2011.

KAMYOD, C. End-to-end reliability analysis of an iot based smart agriculture. In: IEEE. *Digital Arts, Media and Technology (ICDAMT), 2018 International Conference on*. [S.l.], 2018. p. 258–261.

KELLER, H. *Helen keller: Selected writings*. [S.l.]: NYU Press, 2005. v. 2.

KHARCHENKO, V.; PONOCHOVNYI, Y.; ABDULMUNEM, A.-S. M. Q.; ANDRASHOV, A. Availability models and maintenance strategies for smart building automation systems considering attacks on component vulnerabilities. In: *Advances in Dependability Engineering of Complex Systems*. [S.l.]: Springer, 2017. p. 186–195.

KIBERT, C. J. *Sustainable construction: green building design and delivery*. [S.l.]: John Wiley & Sons, 2016.

KIM, D. S.; MACHIDA, F.; TRIVEDI, K. S. Availability modeling and analysis of a virtualized system. In: IEEE. *2009 15th IEEE Pacific Rim International Symposium on Dependable Computing*. [S.l.], 2009. p. 365–371.

KUO, W.; ZHU, X. *Importance measures in reliability, risk, and optimization: principles and applications*. [S.l.]: John Wiley & Sons, 2012.

KUO, W.; ZUO, M. J. *Optimal reliability modeling: principles and applications*. [S.l.]: John Wiley & Sons, 2003.

KWASINSKI, A.; KRISHNAMURTHY, V.; SONG, J.; SHARMA, R. Availability evaluation of micro-grids for resistant power supply during natural disasters. *IEEE Transactions on Smart Grid*, IEEE, v. 3, n. 4, p. 2007–2018, 2012.

LEA, P. *Internet of Things for Architects: Architecting IoT solutions by implementing sensors, communication infrastructure, edge computing, analytics, and security*. [S.l.]: Packt Publishing Ltd, 2018.

LI, L.; JIN, Z.; LI, G.; ZHENG, L.; WEI, Q. Modeling and analyzing the reliability and cost of service composition in the iot: A probabilistic approach. In: IEEE. *Web Services (ICWS), 2012 IEEE 19th International Conference on*. [S.l.], 2012. p. 584–591.

LINDEMANN, C. Performance modelling with deterministic and stochastic petri nets. *ACM sigmetrics performance evaluation review*, Acm, v. 26, n. 2, p. 3, 1998.

MACEDO, D.; GUEDES, L. A.; SILVA, I. A dependability evaluation for internet of things incorporating redundancy aspects. In: IEEE. *Networking, Sensing and Control (ICNSC), 2014 IEEE 11th International Conference on*. [S.l.], 2014. p. 417–422.

MACIEL, P.; TRIVEDI, K.; KIM, D. Dependability modeling in: Performance and dependability in service computing: Concepts, techniques and research directions. *Hershey: IGI Global, Pennsylvania, USA*, v. 13, 2010.

MACIEL, P. R.; TRIVEDI, K. S.; MATIAS, R.; KIM, D. S. Dependability modeling. In: *Performance and Dependability in Service Computing: Concepts, Techniques and Research Directions*. [S.l.]: IGI Global, 2012. p. 53–97.

MANOGARAN, G.; VARATHARAJAN, R.; LOPEZ, D.; KUMAR, P. M.; SUNDARASEKAR, R.; THOTA, C. A new architecture of internet of things and big data ecosystem for secured smart healthcare monitoring and alerting system. *Future Generation Computer Systems*, Elsevier, v. 82, p. 375–387, 2018.

MARSAN, M. A.; BALBO, G.; CONTE, G.; DONATELLI, S.; FRANCESCHINIS, G. *Modelling with generalized stochastic Petri nets*. [S.l.]: Wiley New York, 1995. v. 292.

MATOS, R.; DANTAS, J.; ARAUJO, J.; TRIVEDI, K. S.; MACIEL, P. Redundant eucalyptus private clouds: availability modeling and sensitivity analysis. *Journal of Grid Computing*, Springer, v. 15, n. 1, p. 1–22, 2017.

MCGLINN, K.; O'NEILL, E.; GIBNEY, A.; O'SULLIVAN, D.; LEWIS, D. Simcon: A tool to support rapid evaluation of smart building application design using context simulation and virtual reality. *J. UCS*, v. 16, n. 15, p. 1992–2018, 2010.

MELO, C.; DANTAS, J.; OLIVEIRA, A.; OLIVEIRA, D.; FÉ, I.; ARAUJO, J.; MATOS, R.; MACIEL, P. Availability models for hyper-converged cloud computing infrastructures. In: IEEE. *Systems Conference (SysCon), 2018 Annual IEEE International*. [S.l.], 2018. p. 1–7.

MERLIN, P.; FARBER, D. Recoverability of communication protocols-implications of a theoretical study. *IEEE transactions on Communications*, IEEE, v. 24, n. 9, p. 1036–1043, 1976.

MINOLI, D.; SOHRABY, K.; OCCHIOGROSSO, B. Iot considerations, requirements, and architectures for smart buildings—energy optimization and next-generation building management systems. *IEEE Internet of Things Journal*, IEEE, v. 4, n. 1, p. 269–283, 2017.

MURATA, T. et al. Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, IEEE, v. 77, n. 4, p. 541–580, 1989.

NAIK, N. Applying computational intelligence for enhancing the dependability of multi-cloud systems using docker swarm. In: IEEE. *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*. [S.l.], 2016. p. 1–7.

NAM, T.; PARDO, T. A. Conceptualizing smart city with dimensions of technology, people, and institutions. In: ACM. *Proceedings of the 12th annual international digital government research conference: digital government innovation in challenging times*. [S.l.], 2011. p. 282–291.

NETACARD. *Cisco Packet Tracer*. 2014. Netacad.com, Inc. Available in: https://www.netacad.com/pt-br/courses/packet-tracer.

NGUYEN, T. A.; MIN, D.; CHOI, E.; TRAN, T. D. Reliability and availability evaluation for cloud data center networks using hierarchical models. *IEEE Access*, IEEE, v. 7, p. 9273–9313, 2019.

NING, Z.; XIA, F.; ULLAH, N.; KONG, X.; HU, X. Vehicular social networks: Enabling smart mobility. *IEEE Communications Magazine*, v. 55, n. 5, p. 16–55, 2017.

NORRIS, J. R. *Markov chains*. [S.l.]: Cambridge university press, 1998.

O'CONNOR, J. P. *Off Grid Solar: A Handbook for Photovoltaics with Lead-Acid or Lithium-Ion Batteries*. [S.l.]: Joseph P. O'Connor, 2016.

ONG, E. H.; KNECKT, J.; ALANEN, O.; CHANG, Z.; HUOVINEN, T.; NIHTILÄ, T. Ieee 802.11 ac: Enhancements for very high throughput wlans. In: IEEE. *Personal indoor and mobile radio communications (PIMRC), 2011 IEEE 22nd international symposium on*. [S.l.], 2011. p. 849–853.

PETRI, C. Kommunikation mit automaten (phd thesis). *Institut für Instrumentelle Mathematik, Bonn, Germany*, 1962.

PUTERMAN, M. L. *Markov decision processes: discrete stochastic dynamic programming*. [S.l.]: John Wiley & Sons, 2014.

RAUSAND, M.; HØYLAND, A. *System reliability theory: models, statistical methods, and applications*. [S.l.: s.n.], 2003. v. 396.

REINDERS, A.; VERLINDEN, P.; SARK, W. V.; FREUNDLICH, A. *Photovoltaic solar energy: from fundamentals to applications*. [S.l.]: John Wiley & Sons, 2017.

RESEARCH, G. V. *Smart Cities Market Size Worth 2.57 Trillion By 2025 | CAGR: 18.4*. [S.l.], 2018 (accessed November 28, 2018).

ROHOUMA, W.; MOLOKHIA, I.; ESURI, A. Comparative study of different pv modules configuration reliability. *Desalination*, Elsevier, v. 209, n. 1-3, p. 122–128, 2007.

SALTELLI, A.; TARANTOLA, S.; CAMPOLONGO, F.; RATTO, M. *Sensitivity analysis in practice: a guide to assessing scientific models*. [S.l.]: Wiley Online Library, 2004. v. 1.

SATYANARAYANAN, M.; BAHL, V.; CACERES, R.; DAVIES, N. The case for vm-based cloudlets in mobile computing. *IEEE pervasive Computing*, 2009.

SCHOLL, H. J.; SCHOLL, M. C. Smart governance: A roadmap for research and practice. *iConference 2014 Proceedings*, iSchools, 2014.

ŞEN, Z. *Solar energy fundamentals and modeling techniques: atmosphere, environment, climate change and renewable energy*. [S.l.]: Springer, 2008.

SHARMA, P. K.; CHEN, M.-Y.; PARK, J. H. A software defined fog node based distributed blockchain cloud architecture for iot. *IEEE Access*, IEEE, v. 6, p. 115–124, 2017.

SHARMA, P. K.; SINGH, S.; JEONG, Y.-S.; PARK, J. H. Distblocknet: A distributed blockchains-based secure sdn architecture for iot networks. *IEEE Communications Magazine*, IEEE, v. 55, n. 9, p. 78–85, 2017.

SILVA, B.; CALLOU, G.; TAVARES, E.; MACIEL, P.; FIGUEIREDO, J.; SOUSA, E.; ARAUJO, C.; MAGNANI, F.; NEVES, F. et al. Astro: An integrated environment for dependability and sustainability evaluation. *Sustainable computing: informatics and systems*, Elsevier, v. 3, n. 1, p. 1–17, 2013.

SILVA, B.; MATOS, R.; CALLOU, G.; FIGUEIREDO, J.; OLIVEIRA, D.; FERREIRA, J.; DANTAS, J.; LOBO, A.; ALVES, V.; MACIEL, P. Mercury: An integrated environment for performance and dependability evaluation of general systems. In: *Proceedings of Industrial Track at 45th Dependable Systems and Networks Conference, DSN*. [S.l.: s.n.], 2015.

SILVA, I.; LEANDRO, R.; MACEDO, D.; GUEDES, L. A. A dependability evaluation tool for the internet of things. *Computers & Electrical Engineering*, Elsevier, v. 39, n. 7, p. 2005–2018, 2013.

SMITH, D. J. *Reliability, maintainability and risk: practical methods for engineers*. [S.l.]: Butterworth-Heinemann, 2017.

SONG, J.; BOZCHALUI, M. C.; KWASINSKI, A.; SHARMA, R. Microgrids availability evaluation using a markov chain energy storage model: a comparison study in system architectures. In: IEEE. *Pes T&d 2012*. [S.l.], 2012. p. 1–6.

SONG, J.; KRISHNAMURTHY, V.; KWASINSKI, A.; SHARMA, R. Development of a markov-chain-based energy storage model for power supply availability assessment of photovoltaic generation plants. *IEEE Transactions on Sustainable Energy*, IEEE, v. 4, n. 2, p. 491–500, 2012.

ŞTEFAN, V.-K.; OTTO, P.; ALEXANDRINA, P. M. Considerations regarding the dependability of internet of things. In: IEEE. *Engineering of Modern Electric Systems (EMES), 2017 14th International Conference on*. [S.l.], 2017. p. 145–148.

STOJKOSKA, B. L. R.; TRIVODALIEV, K. V. A review of internet of things for smart home: Challenges and solutions. *Journal of Cleaner Production*, Elsevier, v. 140, p. 1454–1464, 2017.

SU, K.; LI, J.; FU, H. Smart city and the applications. In: IEEE. *2011 international conference on electronics, communications and control (ICECC)*. [S.l.], 2011. p. 1028–1031.

SUKHATME, S. P.; NAYAK, J. *Solar energy*. [S.l.]: McGraw-Hill Education, 2017.

TAO, F.; ZUO, Y.; XU, L. D.; ZHANG, L. Iot-based intelligent perception and access of manufacturing resource toward cloud manufacturing. *IEEE Trans. Industrial Informatics*, v. 10, n. 2, p. 1547–1557, 2014.

TIWARI, G.; TIWARI, A. et al. Handbook of solar energy. *Singapore: Sprinter*, Springer, 2017.

TRIVEDI, K. S. *Probability and statistics with reliability, queuing, and computer science applications*. [S.l.]: Wiley Online Library, 1982. v. 13.

TRIVEDI, K. S.; BOBBIO, A. *Reliability and availability engineering: modeling, analysis, and applications*. [S.l.]: Cambridge University Press, 2017.

VILLALVA, J. R. G. M. G. *Energia Solar Fotovoltaica - Conceitos e Aplicações*. [S.l.]: Saraiva Educação SA, 2012. ISBN 978-85-365-0978-5.

WAGNER, F.; SCHMUKI, R.; WAGNER, T.; WOLSTENHOLME, P. *Modeling software with finite state machines: a practical approach*. [S.l.]: CRC Press, 2006.

WYLD, D. C. *Moving to the cloud: An introduction to cloud computing in government*. [S.l.]: IBM Center for the Business of Government, 2009.

ZANELLA, A.; BUI, N.; CASTELLANI, A.; VANGELISTA, L.; ZORZI, M. Internet of things for smart cities. *IEEE Internet of Things journal*, IEEE, v. 1, n. 1, p. 22–32, 2014.

ZINI, G.; MANGEANT, C.; MERTEN, J. Reliability of large-scale grid-connected photovoltaic systems. *Renewable Energy*, Elsevier, v. 36, n. 9, p. 2334–2340, 2011.