Leandro Oliveira Freitas

Uncertainty and Incompleteness Handling in Context-Aware systems

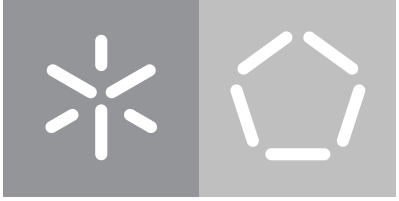
UMinho | 2021

Universidade do Minho
Escola de Engenharia

Leandro Oliveira Freitas

Uncertainty and Incompleteness Handling in Context-Aware systems

agosto de 2021



Universidade do Minho
Escola de Engenharia

Leandro Oliveira Freitas

Uncertainty and Incompleteness Handling in Context-Aware Systems

Doctoral Thesis

Doctorate in Informatics

Work supervised by

Paulo Jorge Freitas de Oliveira Novais

Pedro Rangel Henriques

August, 2021

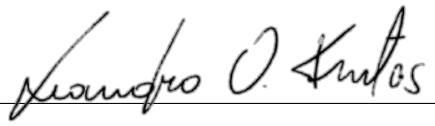
STATEMENT OF INTEGRITY

I hereby declare having conducted this academic work with integrity. I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration.

I further declare that I have fully acknowledged the Code of Ethical Conduct of the Universidade do Minho.

Santa Maria, 11/08/2021

(Location) (Date)



(Leandro Oliveira Freitas)

To my beloved son Arthur.

Acknowledgements

Many people have cheered for me during this challenging period of the Ph.D. course. Without their support, I certainly would not be writing these words today.

I would like to start by thanking Professor Paulo Novais and Professor Pedro Rangel Henriques. They were always much more than supervisors. They are an inspiration for me as a professor and as a person. Thank you for believing in my potential and giving me all the assistance and conditions to develop the project. Professor Paulo, your vision is brilliant. Thank you for providing me the chance to participate in conferences and be there with me. Thank you for the lunches and for sharing with me your way of working and what motivates you. Professor Pedro, thank you for the countless conversations, in person, and by phone. Your advices were more important than you can imagine. Our meetings were a class for me. It is incredible how you understand everything about every subject. Thank you for cheering me up when I was feeling dispirited and lost. Thank you for the dinners and the stories, and your kindness. You were paramount for me to be here.

The friends I made at the ISLab cannot be forgotten. António, Bruno, Dalila, Fábio, Filipe, Francisco, Leonardo, Luís, Marco, Pedro and Ricardo, thank you for your friendship at the university and during the conferences.

At the personal level, first, I would like to thank my beloved wife, Greice, who accepted the unknown of living overseas to take our doctorate courses. Thank you for your partnership and the many comforting times you provided me. Besides all, THANK YOU for giving me the most beautiful present in the world! Our son Arthur arrived during the last year of the course, transforming our lives and making this journey even more extraordinary! I dedicate this work to you, my son. It is impossible to express in words how much I love you both.

I also would like to thank my mother, Ladir, and my sister Alice. You have always encouraged me, even at a distance, asking how I was doing far from home and sending your prayers and good thoughts. My father Paulo (in memoriam) was my biggest inspiration and encourager to go ahead with dreams. I am sure that he still looks after me. I love you all!

My wife's family should also be remembered. My parents-in-law, Eloi and Geísa, my brothers-in-law, Alessandra and Fabiano, and my beautiful nieces, Isabella and Beatriz, you are always so kind.

Without friends, we are no one. So, at this point, I would like to mention my old friends from Brazil. Every time we met over the Internet, they were always kind enough to ask how I was doing at work. Namely, I have to cite Giovani Librelotto, who supervised me during the graduation course and master's degree

and now is more than a professor, a close friend. Also, the dear friends that I made in Portugal from whom I was very welcomed, Ricardo Martini, Renato and Isabela, Bruno and Karine, Lázaro, Alice Balbé, and João Marco, Maiara and Luis Felipe, Ravanello and Tatiana, Jaisso and Cristiane, and Carlos. I am sure that without our discussions on the weekends among some drinks, this period that I lived abroad would not be the same. Thank you!

At last, I would like to thank my colleagues from the Polytechnic School of the Federal University of Santa Maria. They always encouraged me and spared no effort for me to license from my activities to make this dream come true.

Abstract

Ambient Intelligence (AmI) solutions are capable of acting autonomously to benefit human beings. This field of investigation is directly related to other domains, aiming to improve user experience by developing context-aware applications. Context information is naturally dynamic, uncertain, or incomplete. Thus, it is crucial to study ways to handle problems in context data to build applications that run autonomously. The main objective of this Ph.D. thesis is to identify elements that result in the characterization of uncertain contexts. For that, sources and types of uncertainty in intelligent environments were identified. A framework with three layers to validate context data was proposed. The first one defines an attribute grammar to ensure the structure of the data. Restrictions (expressed as logical formulas) associated with the rules of the grammar enforce the system to work with well-defined values. Pieces of context data should have quality, be relevant and be complete to be used by a context-aware system. Thus, the second layer analyses the quality of the data gathered by sensors. A set of premises was used to ensure the relevance of the data regarding its quality and usefulness. The last layer tackles uncertainty (e.g., missing values, obsolete, irrelevant, or ambiguous) and removes it from the context analysis. A classification provided by a decision tree was used to grant relevant data to replace the faulty ones. Thus, the framework ensures that the data being used to process context is well-structured, has high quality, and is as complete and updated as possible. In worst-case scenarios, it can, at least, identifying the source of problems. The validation of the proposal was conducted through a series of case studies developed based on two public datasets. Their content refers to ordinary Activity Daily Living (ADL) tasks, and they were built with actual data collected in test environments. Even considering that the datasets have distinct structures, the results evidenced verified an improvement in the quality of the context data.

Keywords: Context awareness, identification of uncertainty, uncertainty handling.

Resumo

Soluções de *Ambient Intelligence (AmI)* são capazes de actuar de forma autônoma em benefício do ser humano. A área de investigação é relacionada a outros domínios para melhorar a experiência do usuário através da criação de aplicações cientes do contexto. Informações de contexto são naturalmente dinâmicas, incertas ou incompletas. Assim, é crucial estudar maneiras de lidar com problemas em dados para construir aplicações que rodam de forma autônoma. O objetivo principal desta tese de doutoramento é identificar elementos que resultam na caracterização de contextos incertos. Para isso, foram identificadas fontes e tipos de incertezas em ambientes inteligentes. Um framework com três camadas para validar dados de contexto foi proposto. O primeira define uma gramática de atributos para garantir a estrutura dos dados. Restrições (expressas como fórmulas lógicas) associadas às regras da gramática obrigam o sistema a funcionar com valores bem definidos. Dados de contexto devem ter qualidade, serem relevantes e estarem completos para serem usados por um sistema ciente de contexto. Assim, a segunda camada analisa a qualidade dos dados coletados pelos sensores. Um conjunto de premissas foi usdo para garantir a relevância dos dados quanto à sua qualidade e utilidade. A última camada trata a incerteza (*e.g.,*, valores ausentes, irrelevantes ou ambíguos) e remove-a da análise de contexto. Uma árvore de decisão foi usada para fornecer dados relevantes e substituir os defeituosos. A estrutura garante dados bem estruturados, de alta qualidade e tão completos e atualizados quanto possível. No pior dos cenários, pode, pelo menos, identificar a origem dos problemas. A proposta foi validada por meio de uma série de estudos de caso desenvolvidos com base em dois *datasets* públicos, com tarefas comuns de atividades diárias (ADL). Foram criados com dados reais coletados em ambientes de teste. Mesmo considerando que os *datasets* possuem estruturas distintas, os resultados evidenciaram uma melhora na qualidade dos dados.

Palavras-chave: Sensibilidade de contexto, identificação de incerteza, tratamento de incerteza.

Contents

List of Figures

List of Tables

List of Algorithms

Acronyms

AAL Ambient Assisted Living

ADL Activity Daily Living

AG Attribute Grammar

AI Artificial Intelligence

AmI Ambient Intelligence

API Application Programming Interface

AST Abstract Syntax Tree

CART Classification And Regression Tree

CC Context Conditions

CF Certainty Factory Algebra

CFG Context-Free Grammar

DDT Decorated Derivation Tree

DT Decision Tree

EV Evaluation Rules

IoT Internet of Things

LHS Left Hand Side

MEL Maximum Entropy Learning

ML Machine Learning

QoC Quality of Context

QoD Quality of Devices

QoS Quality of Service

RHS Right Hand Side

RSS Rich Site Summary

TR Translation Rules

UQ Uncertainty Quantification

XML eXtensible Markup Language

Chapter 1

Introduction

A contextualization of the field of investigation approached in this document is described here. Concepts about the main topics are presented. The problem of research with the hypothesis to solve it is presented. The methodology used to develop the research follows the main and specific objectives. Also, a running example is characterized aiming to facilitate the understanding of the topics. At last, some relevant works that can be related to this one are listed.

The paradigm of Ambient Intelligence (AmI) is getting special attention in researches due to the evolution of hardware and software technologies and, consequently, new services to assist users are emerging. AmI is directly connected to other fields like Intelligent Environments, which aims the development of systems capable of observe, acquire and use data from the environment and from its users, seeking to improve the experience among them (Cook and Das, 2007).

According to Cook et al. (2009), intelligent environments should be sensible, allowing the system to identify the current state of entities and react under different situations, considering specially the user. They must be adaptable, flexible and assume different behaviours to anticipate situations. The environment should also be transparent, absorbing concepts of pervasive computing to work non-intrusively, avoiding the explicit interaction with the user. Intelligent environments should have devices capturing and exchanging data and helping system to adapt itself to situations, assuming characteristics of ubiquitous computing.

According to Aztiria et al. (2013), in intelligent environments the user is the focus for the development of services. Considering this, AmI is directly related to the People-Centric Computing paradigm, which aims to increase the participation of the user inside environments full of intelligent devices. This paradigm supports the idea of users acting not only as final client, but also as contributors Delmastro et al. (2016). Their actions are monitored and used as source of information to improve the system behaviour.

AmI is a result of the intersection of different fields of research. Artificial Intelligence (AI) is considered one of the most influencers, once many of its issues have essential contribution to the consolidation of AmI (Ramos et al., 2008). The intelligence of systems for these domains is related to Activity Learning and Recognition (Sensoring/Perception), Context-Awareness, Knowledge Representation and Reasoning, Multi-agent Systems and Actuation.

Another contribution of AI refers to the level of actuation. An environment should be smart enough to influence its users, creating alternatives to assist them and incentive them to reach their goals. In this field, robotics emerges as a contribution through autonomous robots specialized in perform specific tasks. In addition to users' assistance, robots also help the management of environment. At last, it is expected the actuation of robots under an affective perspective, with social role, of companionship or emotional assistance. Intelligent environments should have the ability of learning, not only from users' needs or requirements, but also, from the natural evolution of their preferences or desires. These domains learn observing the user and their apprenticeship allows the development of dynamic environments (Novais and Carneiro, 2016). Considering this, one of the main goals for the consolidation of context-aware systems and intelligent environments is the development of applications that are aware of the user's actions and behaviour. Such paradigm aims to adapt the available technology for a better assistance, where computing devices will be able not only to sense what the user is doing, but also, what he wants for the near future.

Thus, it is possible to conclude that intelligent environments have a high level of dynamism and should be able to perceive, adapt and act properly in different situations. In other words, they should have a system aware of the changes that occur in the context of the environment. According to Nugroho (2015), context is an abstraction of environmental elements that have influence on the definition of the applications behaviour.

Context-aware systems use events to evidence changes in the environment. When the system identifies new states of entities, new events are created. Actions are defined as response to events and are used to characterize the behaviour of applications or services. To develop context-aware systems it is necessary to adapt the concept of event, allowing it to be programmed or manipulated by applications (Nugroho, 2015).

Context-aware systems use data provided by different types of sensors. The role of these sensors is to monitor the environment and detect relevant data. Due to its dynamicity, information in intelligent environment is naturally incomplete and, many times, ambiguous, outdated or not accurate. While modelling such domains it is crucial to define a clear abstraction of what sort of data will be necessary for the system to operate satisfactorily. Without a good abstraction, the data collected by sensors may be irrelevant or more generic (level of accuracy) than it should, becoming incomplete or not trustful. The relevance of the data refers to its usefulness to be used in the interpretation of contexts.

The incompleteness means that the data signs sent by sensors are not enough to help systems to build a complete scenario that reflects what is happening in the environment. The lack of necessary data to create models leads to uncertainty, which can be defined as any partial, dubious, vague, outdated or nebulous data that is used by the system as knowledge base. The use of such data results in service orchestrations that instead of assist users, may affect them negatively.

Considering this scenario, there are, in literature, several approaches trying to minimize the uncertainty of information. This document discusses some of them in order to contribute with the consolidation of intelligent environments and Ambient Intelligence.

This document intends to present a Ph.D. thesis developed under the Programa Doutoral em Informática at University of Minho, with emphasis on Context-Aware Computing and Ambient Intelligence.

1.1 Motivation

Context-aware systems are being slowly embedded in our daily life. Nowadays it is easy to find home assistants with a good level of awareness of its surrounding. Intelligent systems of cars perform detailed monitoring of components and assist drivers and passengers. Activity recognition in smart houses allows computing systems to adapt to the environment according to users' requirements. These solutions use different types of sensors providing valuable data for the systems.

This work discusses concepts related to several fields of investigation, such as formal grammars, context awareness, Quality of Context, and uncertainty present in intelligent environments. The development of this Ph.D. thesis is motivated by the possibility of contributing to the advancement of these areas. To analyse the technology behind these incredible solutions available and to help to strengthen them is inspiring.

At last, the possibility of finding answers for some questions that are still open is instigating. The results may provide relevant contributions for the state of the art of domains of investigation.

1.2 Open Research Questions

According to the scope of this Ph.D. thesis, it is possible to evidence issues to be addressed for the improvement of context-aware systems. The analysis of incomplete or uncertain data is an obstacle to overcome. The following open research questions were identified and will be elucidated by achieving the objectives of this work.

- *Is the data collected from the sensors enough to create a context-aware computational model? Do they assure the correct execution of services even with the gap of data caused by the dynamism of the environment considering temporal aspects?*
  A common approach used by context-aware systems when facing a lack of data is to assume a behaviour considering only the facts evidenced and ignoring the missing data. This scenario can lead the system to execute equivocate services;

- *Is it possible to identify uncertainty during the detection of scenarios? Is it possible to create a classification model allowing machine processing?*
  Identifying the source of uncertainty when dealing with context data could allow the proposition of

solutions to prevent it. The recognition of tasks in Activity Daily Living domains and their classification according to similar features can be used in this process;

- How to ensure the definition of contexts from incomplete information to correct, dynamic, and autonomous decision making? Strategies for minimizing the impact of uncertainty in context-aware systems can be defined. Identifying sources of uncertainty and the explicit requisition for the user to provide more data in a specific situation are seen as efficient approaches.

1.3 Research Hypothesis

Considering the context of this Ph.D. project, it is possible to evidence that there are issues to be addressed in the Ambient Intelligence field of research. The analysis of incomplete or uncertain data could be seen as an obstacle to overcome.

Thus, by ensuring a well-structured, high quality, complete and updated context data, the tendency is that:

- The perception of the system increases once the decision-making will be based on consistent and relevant data;
- Wrong service execution, caused by misinterpretation of context, decreases;
- Identification of changes related to the user's behaviour is facilitated;
- Decisions based on default values can be mitigated.

Summing up, the research hypothesis developed in order to conduct this research project states that it is possible to cope with the incompleteness and uncertainty of data, from different perspectives, in context-aware systems. With this, the accuracy of decision-making process in intelligent environments tends to be improved.

1.4 Objectives

The main goal of this Ph.D. work is to search for the identification of elements that result in the characterization of uncertain contexts in intelligent environments. Also, finding ways to overcome the incorrect behaviour of services provided in such contexts.

To achieve the main objective, the following specific objectives should also be achieved:

- *To find ways of ensuring that the context-aware system uses only well-structured data.*
  Issues related to the structure of data may impact its correct usage by context-aware systems. If the system cannot comprehend the content (meaning) of the data correctly, it may ignore this data and make wrong decisions;

- *To analyse approaches of semantic modeling of intelligent environments, seeking to identify the type of characteristics context data should have to be considered relevant.*
  The analysis of the relevance of data is complex for systems with multiple goals, where its behaviour should change according to the changes identified in the environment. A piece of data may seem useless for a specific service to be executed but crucial for another. Thus, it raises the necessity of formalizing the importance of context data, according to a set of requirements;

- *To define a classification model to face the problem of uncertainty caused by lack of data or low quality of it.*
  The use of default values can tackle the uncertainty caused by missing or irrelevant data. They represent data that were useful in situations from the past. The classification provided by machine learning algorithms can be used to label tasks from Activity Daily Living that share similar features. After that, they can fill gaps of data, improving its relevance for the system;

- *To propose a framework capable of identifying causes of incorrect service execution through the analysis of context data and service requirements, overcoming the issues reported in the afore-described items.*
  The main goal of the framework is to improve the accuracy of decision-making by providing three different layers of data validation. The first refers to verifying its structure through the restrictions and rules of an attribute grammar. The second analyses its quality against a set of pre-defined parameters. The third aims to cope with sources of uncertainty by checking hardware issues that may prevent the system from accessing the data, as well as missing information;

- *To conduct experiments through context analysis, generating scenarios, to validate the proposal with real data from smart houses.*
  Two public datasets with distinct structures were used as the basis for the development of the experiments. From them, case studies were created, representing situations of intelligent environments. The validation of these cases is discussed regarding aspects related to elements that cause uncertainty in intelligent environments.

1.5 Research Methodology

Even with some works in the literature discussing if Computer Sciences is really a science or an engineering, according to Hassani (2017), researches on this field can have theoretical or experimental nature or even both of them. Besides that, just like in other areas, Computer Sciences researches may use quantitative or qualitative methods.

Considering this, aiming to achieve the objectives and according to the criteria of classification of researches defined by Silveira and Córdova (2009), it is intended to perform a Qualitative Research, through an investigation and understanding of the object of study (context awareness and uncertainty handling in

Ambient Intelligence), collecting data without formal and structured instruments, analysing related concepts in an organized way. Still according to the authors, the nature of the research fits in the Applied Category, once it seeks a solution for a problem in the Computer Science area, more specifically, Artificial Intelligence and Context-Awareness of Intelligent Environments, through the definition of guidelines for a classification of levels of uncertainty, considering priorities of context.

Regarding the proposed objectives, this research will have an Exploratory, Descriptive and Explicative approaches, according to the classification of Silveira and Córdova (2009) and Gil (2010). Exploratory characteristics will be evidenced in the first phase, along the study of the state of art of the themes involved, including concepts, technologies and related works. In this phase, it is intended to analyse the current scenario regarding to knowledge representation of Intelligent Environments, aiming to identify issues related to the use of uncertain information (incomplete, outdated or incorrect) in the orchestration of services and applications in such domains. Considering the scope of this work, the following definition is being assumed for the expression *orchestration*.

Definition of orchestration: *The process of the system to organize, arrange or adapt the applications and services that compose it according to context changes, i.e., due to modifications regarding the environment's configuration.*

During the process, it is also intended the identification of aspects for modelling a set of entities and relations, regarding the semantic of the domain. Nowadays, approaches about context identification consider mostly syntactic aspects of the entities.

It is believed that, due to the autonomous acting of applications and services in an intelligent environment, where the focus is to find ways of pervasive interaction and assistance to the user, it is necessary to create a semantic representation of entities and their relationships. The study of the state of art aims to help us to acquire a more accurate knowledge about the subject of study. This process is fundamental to the progress of the research to the descriptive phase.

In the next phase, the research assumes a Descriptive approach, through the characterization of the problem. That will be a result of the identification of the causes of incorrect services and applications behaviour. Among the relevant factors that can influence the definition of the problem, one of the most important is the lack of information and interpretation flaws of the data context by computational systems.

An Explicative Research aims the identification of the involved factors in the phenomenon occurrence, describing the motives of it through presentation of results (Gil, 2010). This characteristic fits in the third phase of the work, where a model will be proposed to classify uncertainty levels from the causes discovered the phenomenon identification. The achievement of this goal will be done through the identification and proposal of guidelines that could be used as basis for the model definition, considering the context priority criteria in Intelligent Environments.

Still in the Explicative phase and based on the semantic representation and on the classification of uncertainty levels, it is intended the formalization of a model for context analysis capable of identifying

outdated or incomplete data, proposing possible alternatives from current situations. Lastly, the results will be validated through experiments conducted within research groups.

1.6 Running Example: Human Activities Monitoring

In order to validate the proposal, it was chosen a dataset created in Cook and Schmitter-Edgecombe (2009). It refers to *Smart Apartment Activity Daily Living Monitoring* data in a testbed environment. The authors monitored five activities developed by fifty volunteers. The activities are *phone calling, washing hands, cooking, eating* and *cleaning*. During the analysis, the volunteers were asked to answer a phone call (where the monitoring started) where they received information about their next activities. The resulting dataset had six thousand and four hundred and ninety-seven registers, acquired in an interval of thirteen days. The following sensors were used during the research:

- Motion sensors: to monitor the user's movements inside the house. They were installed in all rooms of the house;
- Item sensors: to detect objects in use by the volunteer to make the phone call, medicine dispenser and cook the meal;
- Water and burner sensors: to detect user's actions while cooking, washing hands and dishes.

The authors of Cook and Schmitter-Edgecombe (2009) provided fifty different datasets, one for each of the volunteers plus one containing a compilation of all of them. This compilation was chosen to be used in the experiments once it provides more information about the same activities. Thus, it was possible to analyse more data, create more inference rules and identify patterns of behaviours that can be used for future machine learning.

All the participants were asked to execute the same activities flow. First, they should make a *phone call*, then *wash hands*, after that *cook* a meal, *eat* the meal and *clean* the objects used. This particularity is important because the result analysis in the following experiment will present some relevant discrepancies in the execution of the activities, which can be related to several aspects, including the user's health and safety.

It is important to emphasize that, for the conduction of the experiments, the *phone call* activity was used for the volunteers to receive information about the tasks they should execute after finishing the call. In other words, the phone call instructions were the same for all of them. Besides that, all the participants were asked to cook the same meal (oatmeal), following the same recipe. This is important because one of the parameters taken into consideration for the experiment analysis is the time spent in each activity. And, specifically for these two, the time may naturally vary. For instance, if the user receives a phone call from a relative that he does not meet for a long time, they probably will spend more time chatting than if the call is from a colleague from his work. Considering the cooking activity, the user would probably spend less time preparing a snack, than if he is cooking a complete meal.

A preparation of the data was performed, in order to be used in the scope of this research. During this process, some data reshaped, specifically in the identification codes of the sensors and the names of activities. Besides that, blank fields were filled with proper values.

No other information about the work developed by Cook and Schmitter-Edgecombe (2009) was used in the research presented here. The experiments were created based on an independent research problem, with independent goals. Besides that, for the development of this work, it was assumed that the information from the dataset refers to the same user, i.e., supposing that only one person was monitored. This makes possible to simulate different behaviours of the same user, assuming that he can change the way he performs the same tasks, taking into consideration the context, preferences and needs.

1.7 Related Work

One of the main contributions of this work is the description of an attribute grammar applied to the intelligent environment. To the best of the authors' knowledge, there are no similar approaches connecting these two fields of research. There are works in the literature with relevant contributions that can be related to the one presented here, even though not using attribute grammars to model human activities. Most of them apply decomposition of tasks into subtasks and actions, creating a hierarchy, and considering temporal aspects.

In Giese et al. (2008), the authors state that formal modelling of tasks helps users to perform activities with computational support. For instance, it is possible to define rankings of priority of activities to be performed according to the user's context, considering, for example, safety and social skills. The authors consider three parameters: simulation of time and conditions, referring to preconditions for tasks and possible obstacles; spatial behavior, that analyzes the location where the tasks will be performed, the objects used and the possibility of changing actors from one location to another, and; analyzing communication, referring to the hierarchy and dependence.

In Bolton et al. (2011), the authors describe the syntax and semantics of a function model capable of integrating task analysis with formal verification called EOFM (Enhanced Operation Function Model), based on Extensible Mark-up Language (XML). The model standardizes changes related to activities, and it defines the behavior of them according to their goals, based on the analysis of state and derived variables.

In Mori et al. (2002) is presented the ConcurTaskTree Environment (CTTE), in which the goal is to provide means for the management of task models for cooperative applications. Among the features, it is interesting to highlight the precise definition of temporal logics for task and their relations; the creation of categories such as user tasks (human routine activities), application tasks (related to the system), interaction tasks (relating user and the system) and abstract tasks (tasks belonging to more than one category) (Mori et al., 2002).

In D'Aniello et al. (2017) a situation awareness framework is proposed. The main goal is to use an approach for adaptive goal selection, where users can choose actions to take aiming to achieve goals, based on their desire, but also considering the cohesive options. The framework is integrated into an

intelligent environment, and context data acquired by sensors feed it. It uses reinforcement learning to understand and improve the reward of the suggested options for the achievement of goals. The proposal is validated through a case study with a prototypical system.

In Djoudi et al. (2016), the authors propose a framework to facilitate the conception of context-aware systems. For that, they merged two modelling approaches, Model-Driven Engineering and Formal Methods. The former is used to identify the entities of context-aware system with the relations between them to create meta models. The latter is used to create a formal domain specific language, defined as CTXs-Maude, to apply reconfiguration and verification techniques. A transformation technique with bidirectional mapping was used to merge them. According to the authors, the specifications generated are executable, which allows the validation of systems developed with this framework. The framework is validated through a running example.

Uncertainty evidenced in activity recognition is tackled in Noor et al. (2016) through an approach combining ontology and Dempster-Shafer theory. The authors state that ontology-based solutions are limited due to problems related to the observations of human actions. By quantifying this and taking into account additional context information, it is possible to define belief states, in order to improve the accuracy of the recognition process. Degrees of belief are also used in Speculative Computation (Satoh, 2005) to fill gaps of context data that generate uncertain scenarios. As well as the approach presented here in this paper, the authors of Noor et al. (2016) consider an activity as being a sequence of actions. However, here activities are represented through attribute grammars instead of ontology.

In Sarker et al. (2020) Principal Component Analysis (PCA) is used to handle with context-aware applications of smartphone usage. The authors describe the problem of high number of dimensions of contexts and how applications' data affect this. A model called "ContextPCA"is presented to deal with it. The main idea is to use PCA to separate components (symmetric from asymmetric) aiming to reduce the dimension of specific situations of context. This reduction was achieved through the combination of PAC approach with a decision tree to classify the most important patterns of the datasets. The paper presents a comparison between the proposed approach and the traditional decision trees, by analysing random registers.

Still using decision tree to analyse context of users with smartphones, in Sarker et al. (2019) it is proposed an approach called BehavDT. The main goal is to provide decision making for specific context situations and also more generic scenarios where the context is not taken into account. The approach uses the sensors (hardware and software) of the smartphone to build the dataset that is used to train and test the decision tree. It considers aspects time, space and social of the user to create contexts. The developed tree uses a behaviour-oriented generalization approach. With this, not only the leaf nodes are used for decision making, but also the internal ones. Thus, all the questions used in the branches to separate the data (and, after that, to classify it) are context-value tests. The nodes contain the answers for theses questions. Behaviour patterns are used for the classification of the data. At last, the authors state that the prediction accuracy of the BehavDT model is higher than a conventional tree.

A framework to adapt the context of users of a smart home through voice commands is proposed in

Chahuara et al. (2017). This system analyses data from physical sensors to perform the reasoning. Uncertain facts are dealt using Markov Logic Network approach. Identification of context changes is detected through the analysis of physical sensors installed in the smart house. A priori knowledge and learning weights were applied aiming to overcome uncertain context scenarios. A hierarchical knowledge model based on ontology was used to create the structure of the context-aware environment. This allows an interoperability of concepts among the modules of the system. Also, it facilitates the processing of the sensor data once a variety of devices was used creating a heterogeneous environment. The main focus of the framework is to act reactively when taking actions and making decisions as response to user's voice commands and proactively when the system detects risky situations. In both cases it considers uncertainties in the context. According to the authors, the validation of the proposal is done in a real-world smart home with naive users. They used weights to identify and measure uncertainty. The evidenced results showed 85% of correct decision making which shows that the system is flexible and adaptive in most of context situations monitored, even when facing uncertain data.

In Huo et al. (2020) it is proposed a mixed $\alpha - \beta$ network model, as result of CNN approach, using uncertainty quantification (UQ), aiming to improve human activities recognition. The α network is responsible for the distribution of weights over the contexts while the β network aims the modelling of the activity recognition under one specific context. Pre-training clustering on the dataset was performed to ensure the stability of the training process. The framework has the goal of learn and deal with uncertain contexts and, also, analyse the likelihood of possible user activities within a context. The UQ is based on the Maximum Entropy Learning (MEL) principal. Considering that, the authors state that the model is able to refuse to make predictions if the level of uncertainty is out of thresholds. The validation was performed with one public dataset and one in-house, where human motion were monitored. The main goal of the study is to create baselines for the improvement of context discovery independently of the chosen modelling approach.

1.8 Structure of the document

Chapter 2 discusses concepts of context-awareness pointing some guidelines for designing, detection, modelling and processing context. Issues about problems related to uncertain data in intelligent environments, considering quality of context and presentation of uncertainty are addressed in Chapter 3.

Chapter 4 presents a discussion about decision tree algorithms emphasizing their characteristics and how they can be used to classify data from Activity Daily Living domain. Chapter 5 introduces formal grammars, more specifically Context-Free and Attribute grammars. Their formal structure is reviewed and the application to intelligent environments are proposed.

Chapter 6 presents the proposal of this Ph.D. work. A framework is described. The modules that compose it are deeply explained. The approach for coping with uncertainty from different sources is detailed. Case studies are described in Chapter 7 aiming the validation of the work. They were created based on two public datasets and represent real data acquisition in test environments.

At last, the conclusions of the work are presented in Chapter 8 along with direct and additional contributions. Future possibilities of research are suggested.

1.9 Roadmad of reading

The main goal of this section is to provide shortcuts for specific subjects covered by this document according the readers' interest. Although *context awareness* and *uncertainty* being naturally embedded in all Chapters of this dissertation, the former is better discussed in Chapter 2, 3, and 6. Requirements, applications and validation are discussed there. The latter is detailed in 3, focusing in the importance of its identification and the insurance of the quality of the data as well as how to deal with it in intelligent environments. Chapters 6 and 7 also discuss it.

If the interest is for *decision trees*, then Chapters 4 and 6 should be read. They describe several approaches for the implementation and the application of *CART* algorithm to two datasets used as examples. If *attribute grammars* is the concern, concepts for implementation and the relation with intelligent enviornments are discussed in Chapters 5 and 6, including its application to context validation.

If the quest is for the proposal and its validation, they are presented, respectively, in Chapters 6 and 7. A deep discussion about the phases of development of the approach are described there. The understanding of formalization of context and the importance of identification of uncertainty are debated and the validation of the work developed is explained.

At last, if the Reader looks for an overview of the thesis with its results, he should analyse Chapter 8. A summarization of the work with its main and additional contributions is presented. A synopsis of the achieved results is described and directions for future research are suggested.

Chapter 2

Context Awareness

Context-aware computing is a wide field of investigation with many different points of view, depending on the approaches used to address problems. This Chapter describes essential concepts about the topic considering the work that was developed. It could be aggregated to others when necessary, according to the domain of interest. Issues related to principles for design context and modeling are addressed and techniques of detections of relevant data. Finally, forms of processing context are presented, using a scenario taken from the running example.

Task modeling is seen as the first step for consolidating context-aware computing in several ways. Formal modeling allows validation of context data, preventing ambiguous analysis and minimizing problems related to mining low-quality data. However, it is not an easy target to accomplish, and there are several approaches to deal with this in the literature.

Context awareness represents the connection between the environment and the computational structure. It refers to identifying the current state of users and other entities and how they can influence the behaviour of applications. Such identification can be achieved using different types of sensors like location monitoring, vital signs, level of stress, or fatigue, among others. In order to go further on the definition of the theme, it is crucial to define the term *context*, as well. Thus, first, this Chapter presents the definition of this field. After that, context-aware computing is discussed with its characteristics.

2.1 Interpretation of Context

The concept of context includes the user's emotional state, focus on an ongoing task, location or orientation, time, and any other relevant entities (objects and people) in an instant of time. However, create applications considering this concept is difficult (Dey, 1998).

Context is any type of data that can be used to characterize the current state of entities (person, place, or object). Any relevant data for the interaction between users and applications in an intelligent environment is seen as an entity (Dey and Abowd, 1999). Considering the definitions proposed by (Schilit et al., 1994) and (Chihani et al., 2011), it is possible to conclude that context is related to the changes that happen in the environment, where:

- Computing environment includes processing and displaying capabilities, quality of the available services input and output devices, network capacity, computing costs, and battery performance;
- Social environment includes user characteristics(physical, emotional, occupational), location, surrounding people, and social context;
- Physical environment considers temperature and levels of loudness and light.

Considering this, for the development of this research, the following definition of context is assumed.

Definition of context: *it is believed that the inner state of the user is not considered context but the result of one. For instance, the emotional state of a user is the result of what happens to him. Thus, the context of an entity is anything external to it that can be used to influence its behaviour (Freitas et al., 2020).*

According to Liu et al. (2010), processing context is complex because it involves identifying new states of entities and using complex inference rules, considering the semantics of relations that connect them. Inference rules allow the deduction of new knowledge based on what is already represented. The representation of the knowledge of a domain can be done through different approaches and technologies. It is necessary to analyse aspects like the level of expressiveness required and the purpose of the model. In section 2.5 it is described the most common approaches for knowledge representation for context-aware systems.

Considering the running example of Chapter 1, an inference rule can be applied in a situation where the user goes to the dining room to eat the meal after having it cooked, letting the stove burning (state = On). Based on information about the user's location, the system infers that he is not cooking anymore, but the burner's sensor status is still On. Thus, it can send an alert informing him about this situation, preventing a possible principle of fire in the house. In order to process context and use inference rules, it is necessary to represent the knowledge of the domain by defining how the environment is seen and how it can be manipulated.

2.2 Context-Aware Computing

Context-aware computing is a paradigm where applications and services can take advantage of user's environmental data, such as his location and current activity, time of the day, other users, and intelligent devices. In the running example, this can be evidenced when the system uses different sensors to monitor the user's activities. Identifying his location and status of objects and devices allows the system to adapt the computational infrastructure to assist the user and show relevant alerts and notifications. The user experience can be enhanced by enabling devices and applications that automatically sense and adapt themselves to the changes in surrounding physical and operational environments. Environmental data is used to create a context for the interaction between users and devices (Musumba and Nyongesa, 2013).

Context awareness can be defined through two perspectives (Musumba and Nyongesa, 2013), (Long et al., 1996):

- Active context awareness: adapts the behaviour of applications according to the sensed context. For instance, when the system detects that the temperature is out of a pre-defined range;
- Passive context awareness: presents new or updated context information to users or persists the context to be retrieved later in time. For instance, when the system asks the user for direct input of data to confirm a context.

Intelligent systems are supposed to have the capability of using external data sources to adapt their behaviour according to this data. Considering specifically context-aware systems, they must read the data and adapt themselves to changes identified on it, even if the data is imperfect (incomplete, outdated, ambiguous) (Camara et al., 2018).

The development of context-aware systems requires a good knowledge representation allied with a set of well-defined inference rules and artificial intelligence techniques. This type of system should analyse what happens in the environment, identify critical situations aiming to assist users, and learn from the environment. Thus, the more the system acts, the more it should learn and, consequently, its autonomous acting would be improved. It works like a cycle. Thus, Machine Learning approaches can be applied, aiming to improve the knowledge of the domain. Chapter 4 will discuss the approach used in the development of this thesis.

Most of the researches about context awareness use a bottom-up approach to propose frameworks and architectures. Their primary focus is to capture contextual data and find a way to understand it and handle it properly. This must be done automatically, with implicit inputs to adapt the behaviour of the applications (Musumba and Nyongesa, 2013).

According to Perera et al. (2014), context awareness has three main features that must be taken into account:

- Presentation of information to the user based on the context: personal devices should present information based on users' location. For example, if he is at work, the device could show his tasks for that day;

- Automatic execution of services: the system should allow machine-to-machine communication and exchange of data. The data used as the basis for the service execution must be completely correct to avoid misinterpretation and execution of wrong services. It is known that a context-aware system should have different types of sensors monitoring and collecting data from the environment. However, if the system does not know what to do with the data, they are useless;
- Goals of the services: the context needs are essential for collecting, fusion, and analysis of pertinent information.

There are several definitions of context and proposed applications of context awareness systems. According to Perera et al. (2014), one of the approaches says that context information can be classified in location, identity, time, and activity. These are considered primary context types, i.e., they are collected and analysed as they were detected, without any kind of data fusion operations or manipulation. From the primary context types, it is possible to achieve secondary context types of data. They have the same classification of the primary, but they result from analysis and/ or processing of primary types. The authors highlight that specific data can be considered primary in one context and secondary in another. The verification of whether data is primary or secondary context may be crucial for the correct execution of services.

Activity recognition has to monitor the user's movements to understand his actions to build models capable of being processed. According to Liu et al. (2017) there are several challenges to overcome in this field. Often, the routines of people in the real world may involve the execution of multiple tasks simultaneously. Situations with these characteristics are challenging for computing systems to understand. In other words, many of the researches focus on single tasks per time. Another problem discussed in the literature refers to the grouping of similar information. The development of context-aware applications needs to create a model that can adapt itself to the dynamicity of the domain without losing the power of processing. The popularization of context data in information systems brings challenges that must be overcome. The trend is that the amount of data to be processed increases considerably compared with systems that use conventional inputs. Besides that, intelligent environments should be full of sensors monitoring objects, users, and other entities. Probably, more than one sensor would be capable of collecting the same data. Considering that sensors have specific objectives, the system must decide from which sensor it should use the data collect, according to the current context.

2.3 Principles for Design Context

The normal development of context-aware applications consider parameters like available resources (including sensors), the total of users, and features of the system. There are several works in the literature, like *Automatic context data life cycle management framework*, Ramparany et al. (2007), Bernardos et al. (2008) and Perera et al. (2014), where the authors define guidelines for the development of context-aware systems, such as:

1. The architecture should have components with the ability to perform a set of tasks independently of other components and layers;

2. Possibility of new components, with new functionalities, to be added and removed without interfering in the architecture structure as a whole. In other words, the architecture of context-aware systems should be extensible. This feature is important since, over time, new relations may emerge, others may become more/less complex, and the system should treat this new information with the same consistency. Besides that, a context-aware system should be developed to be available in the most used platforms with the possibility of implementation in different types of devices, such as smartphones, laptops, smartwatch or wearable computers (Bauer et al., 1998);

3. Compatibility with different hardware of sensors and devices, once there still are no development patterns. They should also be used in different operational systems (Perera et al., 2014).

One of the main concerns of context-aware computing is its ubiquity, i.e., its ability to interact with the user naturally. Hence, context-aware systems should have an Application Programming Interface (API) that allows users to access its functionalities easily. The API should support interoperability among different devices. Thus, a context model should be shared among different layers of the system and also with applications (Perera et al., 2014).

Context-aware systems should be able to build models to work with based on different sources of data. These models should adapt themselves over time according to changes in the context. The context built based on data from the environment should be stored independently from the system itself, facilitating updates. The monitoring and detection of events in the environment are crucial for the proper functioning of the system. A well-defined model for the monitoring and detection of relevant changes helps the execution of appropriated services (Perera et al., 2014).

2.4 Acquisition of Context

Context acquisition is concerned with getting data to characterize the task the user is trying to accomplish. However, daily tasks may have very similar steps even if they have entirely different goals. Thus, it is difficult to determine what the user is trying to do. Applications should use explicit and implicit input of data allowing them to have levels of awareness (Musumba and Nyongesa, 2013).

The level of awareness of a system is directly related to the accuracy of the environment's model. The more information the database has, the better will be the learning of the system. In an ideal context modeling, the system should be able to acquire the necessary data automatically. However, considering that it is not possible to sense all of the context data, it is still necessary for the user to provide it manually, with explicit or more traditional inputs. This may lead to a problem about how the user provides the data. Many times, he may desire something but does not express himself adequately. Then, the system starts to work with incorrect data and, possibly, will trigger applications that are not related to what the user wants.

According to Perera et al. (2014), there are several techniques for the detection of relevant data in intelligent environments, like:

1. Reactive actuation: the system makes requests to the sensor. It characterizes events that occur only once, and the acquisition of data must be at the time it occurs;
2. Proactive actuation: the sensor sends data to the system every time it detects something (Pietschmann et al., 2008). It characterizes events that tend to repeat within a period.

In both cases, the periodicity of the requests or detection may vary between instant detection or periodically.

To acquire the data, sensors need to detect more than once and analyse it to find a pattern. Sensors detect environmental changes and send them directly to the system. In this case, there is no layer between them to process the signs. Software clients have access to the same contextual information. In this case, there is an access management component in the context server to facilitate the concurrence of multiple accesses.

Data can also be acquired based on the source of the context (Chen et al., 2004). It can be provided directly from the sensor device through communication with the system. Usually, this method is applied to detect local information. It can also be acquired indirectly, through middlewares (Perera et al., 2014). In this case, the middleware is responsible for intermediate communication between the sensor device and the context-aware system. Thus, the software is characterized as a source of information as well. It takes the data provided by physical sensors, analyses it, and, if possible, derives new data. The context-aware system should validate the result of this process before being used in decision makings. Besides that, information can be acquired from different external context sources, like databases, RSS, and web services (Perera et al., 2014).

Nowadays, there are several types of sensors to monitor and detect context information. According to Indulska and Sutton (2003), they can be categorized in physical, virtual, and logical sensors. The first is capable of detecting data without any external intervention. There are plenty of devices equipped with this kind of sensor. Data generated is considered as low-level context, i.e., it is related to simple data (Perera et al., 2014). On the other hand, the virtual sensors do not detect context data without intervention. They search for data from other sources through web services and process it as sensor data. Logical sensors use physical and virtual sensors to produce, through web services, richer context information.

Although one of the premises of context awareness is the natural interaction between system and user, in some cases, context information can be manually provided. This fact happens, for example, when the user wants to personalize a notification application not to bother him in one specific night.

2.5 Context Modeling

Context-aware systems must use models that reflect the knowledge of an intelligent environment. According to Yanwei et al. (2011), these models can be classified as static, with a limited and previously defined set

of information, or dynamic, discovering and manipulating new information over time.

The nature of context-aware applications is distributed once it uses data from different sources (Dey, 1998). The modeling process needs to consider aspects like heterogeneity of entities, mobility of users, relations among entities, periodic events, uncertainty, reasoning, usability, and efficiency of the context (Bettini et al., 2010), (Perera et al., 2014). Besides that, according to Pradeep et al. (2021), context modeling refers to structures that facilitate the management of data that composes a situation.

The context modeling starts with the definition of entities with their characteristics (attributes, restrictions, and relations). This process should be performed using a known context as a basis. It is essential to highlight that one of the main issues on context modeling refers to subjectivity. The same set of data detected may be valid in one situation and not in another. Thus, the more complete is the data about a context, the more accurate the model will be built based on that. After modeling a new context, it should be integrated with the rest of the context information in a repository.

According to Musumba and Nyongesa (2013), context can be modeled using the following approaches:

- Key-value: this is the simplest structure for applications. The pairs of key-value represent the characteristics of services. Discovery of services is performed through the match of these pairs and specific algorithms;

- Mark-up: uses a hierarchical structure with tags to create profiles to represent the mark-up scheme model.

- Object-oriented: Uses all the concepts of object-oriented paradigm to build context-aware applications. Context data can be seen as objects allowing encapsulation of processing details;

- Logic-based: the context model is defined through facts, expressions, and rules, giving a formal characteristic for the model. The management (addition, removal, and updating) of new facts is performed through a logic-based system. It allows to have inference rules to derive new information about the context;

- Ontology-based: uses ontologies to represent the entities and their relations. The advantage of this method is the level of expressiveness, including the hierarchical structure and the possibility of reasoning (Strang and Linnhoff-Popien, 2004).

The design of context-aware systems should always try to minimize the mismatch level of awareness. This mismatch can be minimized if the user knows what can influence the system. The user should be aware of the context information used by the system.

The definition of services of a context-aware system is directly related to requirements about the current state of entities. Although, many times, the signs collected by sensors do not provide enough data to fill such requirements. That happens because, under the computational perspective, the context information is naturally dynamic, uncertain, or incomplete. The result is an inexact connection between the service and what is provided by the environment (Vanrompay et al., 2009). The description of the context and services

requirements uses similarity comparison, which analyses the necessary and provided data individually, and also global comparison, which analyses the context as a whole (Vanrompay et al., 2009).

Hence, it is possible to verify that one of the main challenges of modeling context is handling uncertainty. A context-aware system should obtain enough data to build complete contexts, i.e., to identify all information present in the environment. This fact is crucial for a correct orchestration of events with context-aware services. If the sensors cannot identify a situation or if the context models do not represent a real scenario of the environment, the services that would be executed may not be the more appropriate.

2.6 Context Processing

Besides the precise understanding of the domain through the development of a well-defined context modeling, it is necessary to apply algorithms to process it in order to infer new knowledge. This section describes how machine learning algorithms can be used to improve context-aware systems.

Data classification provided by machine learning approaches is widely used to process data and can be applied to several business fields like e-commerce, telecommunication, and medicine. Different machine learning algorithms, such as binary and multiclass classification, regression, association rules, and other approaches (Smola and Vishwanathan, 2008), (Alpaydin, 2014). The main idea is to analyse a model composed of parameters, find patterns of behaviours among them, and suggest an improved output. Through the analysis of these patterns, it is possible to create predictions about future situations, assuming that what will happen is not much different from what is represented in the dataset, so it is supposed to be correct. Besides the predictive model, machine learning can improve or expand the knowledge about the dataset, presenting descriptive characteristics (Alpaydin, 2014).

Machine learning allows the creation of association facts, i.e., based on the analysis of individuals' behaviours of the dataset, it possible to deduce related action patterns. For instance, considering the running example from Chapter 1 that there is a high incidence that, in the period of the night, the user not cleaning the objects he used, after eating his meal, i.e., frequently, he forgets to do the dishes after dinner. The systems analyses his behaviour and identifies a pattern using the association rule P(E, N | C = false), where E represents the activity of eating, N is the period of the night, and C is the activity of cleaning the objects. If the value related to this rule is very high, for instance, P(E, N | C = false) = 0.85, it indicates a probability of 85% of the chances of this scenario being true. This might mean that the user does not like to do the dishes at night. Alternatively, if the percentage is, for instance, around 65%, it might indicate that he tends to forget to do it, sometimes. Thus, depending on the probability range, the system could remind the user to clean the objects he used after dinner.

Another use of machine learning aims to deal with the classification of probability. This approach can be divided into Binary or Multiclass Classification. In the former, a given pattern from a context is associated with one of two values, for instance, 0 or 1. In the latter, the pattern can be associated with a more significant range of values (Smola and Vishwanathan, 2008).

The classification analysis can be applied to running example. The subjects were asked to perform five tasks in a pre-defined sequence (phone call, wash hands, cook a meal, eat it, and clean). Assuming this scenario as a house with a context-aware system and the user performing his daily tasks, these five activities could be seen as a pattern of behaviour. I.e., the user usually performs them in this order. Thus, if he does not perform one of them in one specific moment, the system would be able to identify this anomaly and use classification analysis to understand the impact of it in the user's routine by defining probabilities for this situation. Multiclass classification can be applied when more than two values are accepted (e.g., Low, Medium, High). Binary classification is used in situations where only two values are allowed (e.g., True False). The following code presents the structure of the rule.

$$P (R \mid PrAct, CAct)$$

Where:

- **R** is the risk value identifier;
- **PrAct** is the pre-requirement, i.e., the previous activity;
- **CAct** is the current activity that the user is performing.

Using information from past situations, it is possible to predict the probability of the patient returns on time, assuming the premise that if in the past was true, in the future, it will also be true (Alpaydin, 2014).

Considering a scenario where the user, after finishing the *phone call* activity, goes directly to the kitchen to *cook* his meal, it is possible to create an association rule to classify the risk of he forgetting to *wash hands*. The resulting application of the rule in this case is P (R=0 | WashHands = True, Cook = True) = 0.90. This means that in 90% of the cases where the user is cooking, he accomplished the task of washing hands. The other 10%, represented by P (R=0 | WashHands = False, Cook = True) = 0.10, could be seen as an anomaly, or a case of uncertainty, and thus, must be properly addressed. The simplest way of doing that is by sending a notification to the user alerting him.

Despite been a simple example, this can be extended to more complex situations where the monitoring of the user is related to his health condition, for instance.

2.7 Summary

The main goal of this Chapter was to discuss characteristics related to context-aware computing. It introduced the term *context* within the scope of this thesis, emphasizing that it is *anything external* of users, but with relevance enough to influence their behavior. Different visions of intelligent environments (computing, social and physical environments) were described, highlighting characteristics used in context analysis.

The paradigm of context awareness and context-aware computing are discussed by the analysis of *active* and *passive* perspectives. Features such as presentation of information, automatic execution of services, and goals are characterized. Principles for designing context-aware computing are described.

The importance of the accurate acquisition of data was also tackled in this Chapter. *Reactive* and Proactive detection were characterized. The importance of the correct explicit provision of desires and data by the user was discussed, aiming to improve the decision-making and adaptation of services. Besides that, different modeling approaches were described, highlighting the importance of choosing the most appropriate one according to the system's goals.

At last, machine learning approaches were introduced, presenting how they can contribute to context processing. Supervised and unsupervised learning was defined. The running example was used to explain the characteristics of binary and multiclass classification.

Despite the approach chosen to reason over context data, uncertainty is a common challenge to overcome. It may impact context-aware systems in several ways, for instance, in the adaptation capability. Different approaches describe problems of this sort, caused by lack of reasoning precision or knowledge ambiguity (Bobek and Nalepa, 2017).

The following Chapter addresses the problem of uncertainty, which must be considered when dealing with context-aware computing.

Chapter 3

Uncertainty

Important topics related to uncertainty in context-aware systems are discussed in this Chapter. A list of requirements is described, along with their importance when building sensitive systems. Despite the wide scope of the theme, the main focus here is to describe concepts related to the work developed. Two main types of uncertainty are discussed. The quality of context data to minimize (or prevent) nebulous scenarios is addressed through the specification of parameters to be analyzed. Identification of sources of uncertainty can be used to tackle this problem. This Chapter discusses the topic, highlighting the importance of the presentation of it to users

Context-awareness has as a premise to sense user and environmental data and apply to it inference rules in order to automatically learn the situation of users and adjust the behaviour of computing applications to it (Lim and Dey, 2011), (Dey et al., 2001). However, the more the system learns, the complex the inferences become because the number of rules tends to grow (Weiser and Brown, 1997) considerably. Context-aware systems must deal only with accurate data in order to make correct decisions considering real-world situations because their main goal is not only to assist users in daily tasks but also to ensure their safety in intelligent environments (Aloulou et al., 2015). However, context data not always reflect real situations as it should. Hardware, energy, and communication faults are some of the obstacles faced by context-aware systems. In the majority, these problems are related to sensors, including battery and communication failures (Liu et al., 2009). Wrong domain knowledge mapping and not well-defined inference rules also affect the performance of the system (Aloulou et al., 2015).

Due to the dynamic nature of AmI, it is essential to find ways to handle inaccurate and incomplete information correctly, as its analysis can result in the construction of uncertain contexts. Uncertainty in

human activities recognition is seen as one of the main obstacles to overcome for the consolidation of context-aware systems (Freitas et al., 2019c). In intelligent environments, it can be defined as any incomplete, contradictory, vague, or outdated information (Ben Yaghlane et al., 2008). Nebulous information, susceptible to temporal changes, is also considered uncertain.

As already mentioned, another difficulty that may arise in those systems is the incapacity to acquire or process all the relevant data to characterize the context entirely. Nevertheless, context-aware systems should be able to reason over data even if it has pieces missing or if it contains imprecisions or noises, without compromise the result (Michalakis et al., 2021).

Information is partial when only some of the queries in the reasoning layer can be answered. However, to build a context model aiming to assist users, these answers should be as complete as possible. Besides that, the data could not be as reliable as it should or generate a conflicting result. In this case, different types of sensors can generate the same kind of information from an entity but to be used in different scenarios. The data collected from them could produce conflicting data to the context model, where the same data represents more than one kind of information, depending on the situation (Bhatnagar and Kanal, 1986).

Detected information could be partially correct or even absent but still should be considered for reasoning. This way, Multi-Agent Systems provide a generic model with the necessary flexibility, with different levels of autonomy and dependency for all components of intelligent environments.

The dynamism of a domain centred on the user is a complex subject under different perspectives. It is crucial to consider aspects related to their state of mind, including stress, fatigue, personal preferences, and external factors like temperature and time of the day. Such factors can influence the user causing changes in their acting patterns within a short space of time. The result of this scenario is that the knowledge base starts to work with a high level of uncertainty, without the assurance that the mapped context is correct and up-to-date.

Many times, uncertainty is neglected. Systems assume that the existing data is enough to represent the domain's knowledge syntactically and semantically. Thus it is not possible to ensure that will make the most appropriate decisions at all times. On the other way around, explicit representation of uncertainty allows the quantification of its interference in the dataset Camara et al. (2018). At last, by providing different data sources to the system, with a broader diversity of sensors, optimized reasoning algorithms may be developed, helping to decrease its negative impact on users Tian et al. (2018).

The uncertainty originated in an intelligent environment could be the result of a variability phenomenon, i.e., due to changes in the user behaviour, making them act unexpectedly (Mokhtar et al., 2006). Another cause refers to the lack of complete data about specific situations, increasing the complexity of the system's learning process. Besides the subjectivity generated from these two sources, the quality of the computational service must remain the same. The uncertainty identified in dynamics environments should not be ignored. The user behaviour should be analysed to seek alternatives to fill information gaps evidenced during the process of building contexts (Boulkrinat et al., 2014).

Context information is naturally incomplete and uncertain and could be the result of different causes,

e.g., interpretation problems of the signs sent by sensors (Amdouni et al., 2014). Different manufacturers may use specific concepts and identifiers to name the same entity or relation.

One of the main problems researchers face in using probabilistic and machine learning to deal with context is that they take a long time to learn (Bobek and Nalepa, 2017). According to Aloulou et al. (2015), the inference process in context-aware systems is a complex task due to problems of technological failures and incomplete knowledge, once the nature of the data is dynamic, ambiguous, and imperfect itself.

Several researches are being developed in the sense of dealing with the uncertainty of context-aware systems. In Bobek and Nalepa (2017), the authors defend that mobile context-aware systems may have to deal with a significant quantity of data, while this can change fast in a short period, becoming outdated consequently, useless for being used for the building of context-aware models. Besides that, they say that the conventional approach for modeling context-aware systems cannot be applied in mobile environments. Thus, they propose four requirements that should be followed for the building of such systems:

- Intelligibility: the user should be aware of what the system is doing and decide to change it, if necessary;
- Robustness: due to its dynamic nature, it should be able to adapt itself to different environments;
- Privacy: all the user's data should be secure and accessible only by authorized people;
- Efficiency: mobile context-aware systems should use the available resources the best way they can, with good responsiveness.

Besides the definition of these requirements, the authors also highlighted the importance of dealing with the uncertainty that mobile context-aware systems may generate. They presented a project named KnowMe to validate the proposal, describing all the phases of the development.

An alternative to face situations of an incomplete set of data is to speculate information about the future. According to Oliveira (2017), Speculative Computing refers to the use of defaults values as input for processing and generation of output with previsions of future situations. These default values can be known data from a set or, if this data is not available, it can be defined by the user as a valid value for that attribute. In the case of context-aware systems, by using these default beliefs as input, speculative computation helps to improve the reasoning over a context model, reducing the time of decision making and execution of applications.

There are two types of uncertainty in which context-aware systems should deal with (Helton, 1997), (Senge et al., 2014).

- **Aleatory uncertainty (stochastic or type A)** is associated with hardware problems, including devices, energy, and communication failures. Problems from any of these natures may result in the system behaving randomly. This sort of uncertainty refers to problems of interpretation of the signs sent by sensors and statistics variability;

- **Epistemic uncertainty (subjective or type B)** is related to the level of domain knowledge. This includes problems of execution of reasoning due to lack of context data, making it impossible for the system to identify a current scenario and classify users' behavior.

The more information the system has access to, the lower will be the level of epistemic uncertainty (Senge et al., 2014). Thus, this type of uncertainty tends to be reduced over time using techniques for system learning. According to Bobek and Nalepa (2017), reducing epistemic uncertainty helps to deal with aleatory uncertainty. When the user interacts with an application, e.g., inserting data, he is helping the system to learn, and consequently, he is helping to reduce epistemic uncertainty (Bobek and Nalepa, 2017).

The following sections present important issues to be approached to address uncertainty in context-aware systems within the scope of this work, namely Quality of Context, Identification, and Presentation of uncertainty.

3.1 Quality of Context

Data provided by physical sensors and resulted from processing in context-aware systems can vary according to its usefulness. The quality of data provided by these sources to be used in the orchestration of services is defined as Quality of Context (QoC) (Buchholz and Schiffers, 2003), (Sheikh et al., 2008). It refers to any data about the quality or reliability of context data used by the context-aware system. This includes the level of accuracy and completeness of how the data describes an entity of the real world at the time it was detected (Kim and Lee, 2006). To complement this concept, in Manzoor et al. (2008) the authors proposed the assignments of weights for the attributes, using them as metrics to measure the level of completeness of the information.

According to Buchholz and Schiffers (2003), QoC is directly related to Quality of Services (QoS) and Quality of Devices (QoD). QoS strategies should be considered for the appropriate execution of applications, considering the goals of the system. Besides that, the data is gathered from sensors installed in the environment. Thus, the quality of these devices (QoD) may influence the provision of data for the system.

The minimization or complete elimination of uncertainty context can be achieved through the identification of context patterns that originate the uncertainty (Xu and Cheung, 2006). For this, it is necessary to identify ambiguous, inconsistent and contradictory context data and compare them to real-world statements (Aloulou et al., 2015).

In the running example of section 1.6, a situation that shows the importance of the context's quality can be evidenced when the user changes his behaviours. For instance, if he decides to cook a more elaborate meal for himself or receive friends for dinner, he probably would spend more time in the kitchen with the stove burner ON. The system may classify this situation as uncertain since the time the burner's state in ON is less than what is being monitored at this specific moment. By analysing this situation, the system could infer that the location and burner's sensors are not functioning correctly, and it starts an interaction

with the user to warn him about that. The quality of context is that the system misinterpreted context data and, because of that, disturbed the user with undesired notifications.

One way to compare uncertain data with real-world statements is through semantic modeling of the knowledge, allowing the definition of human profiles and the representation of the domain according to the system logic. Besides that, semantic modeling allows deeper processing of the content through formal models and, consequently, improving software development capable of achieving a better understanding of the data Villanueva et al. (2016). The importance of this approach relies on the fact that the syntactical analysis is not enough to be applied in dynamic scenarios, such as intelligent environments and context-aware systems. This scenario happens because similar situations may compose the same dataset when in reality, small facts differentiate them and may result in unexpected behaviour of the system.

According to Aloulou et al. (2015), the state of an entity or situation is considered absolute truth if there is no level of uncertainty associated with it. However, in context-aware systems, usually, the context data has a level of certainty and uncertainty associated, measured numerically.

Uncertain information can be represented in numeric models through approximation values within an established error margin (Bhatnagar and Kanal, 1986). Independent of the chosen methodology, usually, a relation between two instances has external aspects that must be considered once they may be relevant for the correct understanding of the situation.

Data used in any context-aware system must be represented to be computed. According to Bhatnagar and Kanal (1986), this representation must have characteristics of similarity with what it represents in the real world and should be as simple as possible, avoiding any ambiguous meaning. The system must analyse all the possible scenarios that match it and decide and orchestrate the execution of services. To ensure that services will be the most appropriate for each situation, the data brought to the system must be complete.

In Buchholz and Schiffers (2003) was defined as a set of parameters to analyze the QoC. The first one refers to the *accuracy* of the data. The more accurate a piece of context information describes the real-world environment, the higher is its quality. This precision should be measurable in order to be computed. This is important because physical sensors are electronic devices that can fail and provide incorrect data. Thus, context-aware systems should be able to analyze data aiming to verify its level of correctness by comparing it with pre-defined sets of expected values for each sensor. The accuracy of context data includes the veracity of past occurrences of it among the software agents. For instance, if a specific software agent frequently has to deal with inaccurate data, its relevance must be analyzed carefully to avoid compromising reasoning and decision making.

Intelligent environments such as intelligent houses are generally characterized by their dynamicity. The *usefulness* of context data can change rapidly, i.e., one specific context data can be static for an extended period, and another can become outdated in a matter of seconds. Thus, the system must use *up-to-date* information whenever possible to ensure proper decision-making.

QoC can also be classified through the definition of another set of parameters (Hoyos et al., 2016). The type of *acquisition* is essential to analyze the source of data. This includes not only physical sensors

but also the result of processing data. Thus, it is possible to assume that the software can also be seen as a data source. Another parameter to be considered is the *types of representation* of context data. This is related to aspects like units, formats, and ranges that a set of data will have to ensure that the system will correctly interpret it. At last, context data should be analyzed regarding its *importance* for the software agents, i.e., how useful and relevant it is considering the goals of the system.

The approaches provided by Buchholz and Schiffers (2003) and Hoyos et al. (2016) complement each other and have a substantial contribution to ensure that sensor data is consistent, complete, and relevant and, thus, can be used by context-aware systems. The following tuple formalizes it:

$$\text{QoC} = \langle A, C, Acq, R, I \rangle$$

Where:

- *A* refers to the accuracy of the data;
- *C* represents the set of acceptable values for one specific situation;
- *Acq* is the source of the data;
- *R* refers to the format of the data;
- *I* stores a value that ranks its usefulness.

QoC of a piece of data is ensured after the validation of each element and can be represented as an intersection of them, as follows:

$$\text{QoC} = \text{A} \cap \text{C} \cap \text{Acq} \cap \text{R} \cap \text{I}$$

These parameters are essential when dealing with context-aware systems. Any problem related to them may affect the orchestration of services and applications. If the system uses low-quality data, the user experience may be affected drastically (Buchholz and Schiffers, 2003).

The quality of context information is directly related to the system's goals, i.e., depending on the type of adaptation that the system may assume. The same set of data may or may not be relevant (Hoyos et al., 2016). The level of quality of a situation is restricted to the quality of the set of context data. Different software agents from the same system may use the same set of data to build different scenarios (Buchholz and Schiffers, 2003).

Considering the scope of this work, scenarios refer to the system's understanding of situations. Context-aware systems must be able to build scenarios with a high level of accuracy, considering the situations of the environment.

The QoC influences the analysis of each of these scenarios to represent the real-world situation with more precision. Besides that, the dissemination of useful QoC data prevents the system from using out-dated information and minimizes problems caused by low-QoC data. Figure 3.1 presents a generic scheme of the dissemination of context data after analysis of its quality.

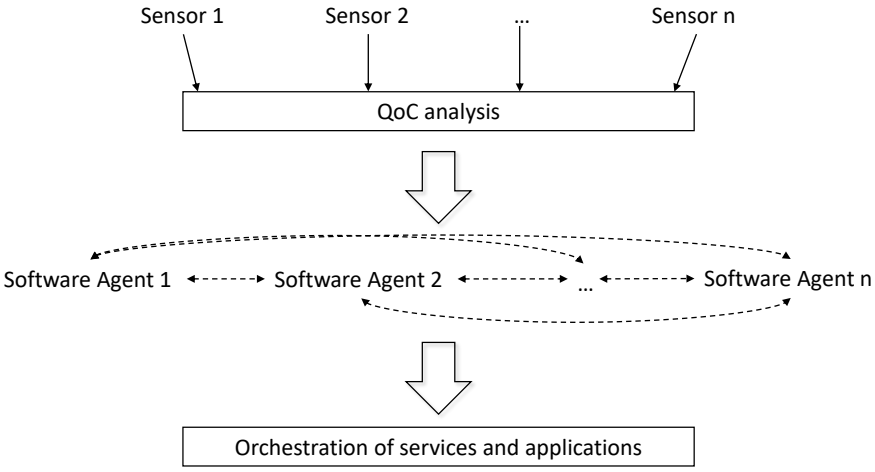


Figure 3.1 Context data sharing among software agents.

After receiving data from the environment, the system performs an analysis to ensure its quality. This includes all the features described above. Then, the sets of data are sent to their respective software agents, where they are properly used and can be shared with others whenever requested. While waiting for the response of requesting information from another agent, the former may use speculative computation belief values to fill in the required information and not interrupt processing (Satoh, 2005), (Oliveira, 2017) — more details on this approach in Freitas et al. (2019a).

Uncertain scenarios are created from the use of low-quality of context data. This is one of the main obstacles to overcome for the improvement of context-aware systems. The following section presents a proposal to tackle this problem, where the main goal is to identify elements that cause uncertainty.

3.2 Identification of Uncertain Situations

Context-aware systems in ADL environments often face imperfect information. Considering the scope of this paper, we characterize imperfect information as any data provided by physical sensors that do not reflect what happens in the environment with a satisfactory level of accuracy. This includes outdated, wrong, or missing data. The concept of accuracy may change according to the goals of the system. Besides that, assuming that systems can be seen as a source of information, after processing raw data, they also can generate imperfect information.

The uncertainty generated by incomplete context data may compromise the correct interpretation of real-world situations. Even considering a system with consistent and well-defined goals, if the data used as input does not have good quality, processing will not have the most appropriate output.

The minimization of negative impacts caused by low-quality context data must be addressed. The approach presented in this paper highlights the importance of the identification of uncertain situations.

According to Panayotov et al. (2018), uncertainty can be identified through the analysis of some features present in the context. Domains that involve monitoring human activities can be highly dynamic. Inputs used by context-aware systems may present slight differences even when they represent the same

context. Such systems should be flexible enough to calculate this and understand these discrepancies. This allows the system to create different scenarios that possibly represent the same situation. Thus, it is possible to create ranks considering the accuracy of the sensed situation regarding the real world. The process of identifying the sources of uncertainty may have high computation costs and time-consuming. The analysis of incomplete data must be performed for each entity that composes a situation.

According to Uusitalo et al. (2015), and Regan et al. (2002), uncertainty has several sources and can be categorized. Each of the types can not be isolated analyzed, especially in dynamic environments. It is natural for context-aware systems to deal with a certain level of **randomness**. In other words, it is not possible to precise with one hundred percent accuracy what happens in a situation even though probabilistic models have been used to improve these outputs. Besides that, problems related to the correct functioning of sensors also must be tackled. These electronic devices can generate inaccurate data when not well-calibrated, for instance. Thus, context-aware systems must validate environmental data before using it as input for reasoning. Wrong **measurements** of the weight of users, the temperature of the environment, or the exact position of them are amongst the standard errors. This type of error affects the sample data from the environment that the system uses to build situations (**systematic**). Consequently, it will not reflect the real world and will affect the orchestration of services. Still, according to Uusitalo et al. (2015), it is difficult to find errors related to context data after the definition of situations based on inaccurate measurements.

The natural **dynamicity** of context data affects its usefulness for the interpretation of situations. This is another problem that may generate uncertainty once the system uses obsolete data as input for decision-making. The definition of ranges of acceptable values front the same scenario is seen as an excellent approach to tackle this problem.

Another source of uncertainty is related to the **modeling**, i.e., creating systems capable of understanding what happens in intelligent environments. It is often impossible to evidence all necessary variables that involve the entities in such dynamic and productive environments. This problem affects the system because, if it is not well-defined, the range of services and applications will be fewer than it could. In other words, the system will not be as aware of the context as it could.

These types of uncertainty can also be categorized within **aleatoric** and **epistemic** classification. **Randomness** and **Measurements** categories can be associated with the *aleatoric* type once it includes any problems related to hardware failures or problems of communication. This may affect the behavior of the system, making it act unexpectedly. **Systematic** and **modeling** categories can be associated with the *epistemic* type due to its characteristics of representing problems of interpretation of the context, i.e., the system does not have enough knowledge resources to resolve what happens in the environment.

If the system faces situations where it cannot understand what happens in the environment, it can ask the user for help to clarify some missing points. Considering this statement, Chen et al. (2018) proposes another classification for uncertainty in context-aware systems. They present a tool where the user can help solving ambiguous situations.

Besides that, different from Helton (1997), where two main types of uncertainty are defined (*Aleatoric*:

and *Epistemic*), here in Chen et al. (2018), spatial-temporal aspects are taken into consideration to divide uncertainty into: *Point Of View*, including problems related to the conversion of sensor data into global position; *Temporal*, related to time lapses; *Transaction Attribute*, considering that attribute values may have different meanings depending on the source; *Location*, referring to problems with GPS logs with missing data, and; *Identity*, including problems with user's data. Although defining other sets of uncertainty categories, it is easy to say that all of them can be related to *Aleatoric* or *Epistemic*.

3.3 Presentation of Uncertainty

Besides acting autonomously, applications should also give feedback on their behaviour to users (Bellotti and Edwards, 2001). Nowadays, mobile applications interact with the user aiming to confirm an inference (e.g., applications that ask the user to confirm his location). Thus, interaction with the user is one of the ways of dealing with uncertainty (Lim and Dey, 2011).

One of the main contributions of displaying information (highly reliable or uncertain) to the user lies in the fact that this helps him to trust the system (Yan et al., 2010).

According to Lim and Dey (2011), it is essential to consider a high probability of dealing with uncertain scenarios when designing context-aware applications, once it is complicated to map all the context data. Depending on the level of uncertainty, a good approach is to ask the user to help the system solve data gaps.

Considering the scenario presented in section 3.1, where the system misinterpreted the fact that the user was not leaving the house, but only passing near the front door, maybe it would be less intrusive if the system had requested more information to build a complete context. For example, instead of emitting alerts suggesting that the user forgot the stove burner on, it could ask him if he was still cooking. This way, the semantic involved would be improved, once instead of disturbing the user with recurring notifications, the system would understand the situation and take this as a feasible behaviour, even being different from the usual.

Another important thing is that the applications should never minimize the importance of data. If there is any incompleteness in a scenario, the user should be notified (Kittur et al., 2008). Thus, the user should always be aware of what the system knows about his context (Bellotti and Edwards, 2001). Modeling uncertainty through probabilistic models is a topic of research in many projects. However, only a few of them focus on the importance of displaying the uncertainty and use it to interact with the user (Lim and Dey, 2011).

According to Antifakos et al. (2004), if the level of certainty in a specific context is high, the level of uncertainty can help to speed the performance of a task if displayed to the user to analyse. In high or medium levels of certainty, the users can still improve the performance of tasks. However, according to Rukzio et al. (2006), if the level of certainty is low, the level of uncertainty to the user is not a good approach once he has to analyse much more data in order to prepare the system to help him.

Fuzzy logic can be used to create models to minimize uncertainty issues. However, it allows only to reduce problems caused by inaccurate information provided by humans (Bobek and Nalepa, 2017). Certainty Factors Algebra (CF) are widely used to deal with problems of uncertainty in rules-based systems (Bobek and Nalepa, 2017). It uses cumulative and disjunctive rules aiming at the same goal. The cumulative rules are composed of independent statements, and the disjunctive is composed of dependent statements (Bobek and Nalepa, 2017).

Thus, it is possible to conclude that there are critical challenges to overcome the uncertainty in intelligent environments. It is essential to guarantee accurate modeling of the available services with a formal description of their goals and requirements for execution. Besides that, the service invocation can verify that the reliability of the output information should be guaranteed. At last, it is necessary to extend the conventional composition models verifying levels of probability for each service related to the detected context. The correct definition of each of these steps contributes to more accurate service orchestration, even if working with uncertain context data.

3.4 Handling Uncertain Situations

There is a wide range of possible problems that result in uncertain situations in intelligent environments. The most common is related to the lack of data for the system to comprehend what actions the user is performing and, consequently, how to adapt its services according to this context. Besides that, the actions that could be used to delimit the beginning and finishing of activities sometimes are not explicit enough. The overlapping of activities is part of the natural human behaviour and is defined by Kwon et al. (2019) as *label jitter*. Activities performed by the user without changing location and/ or using some house appliances may present this characteristic.

For the system to recognize the activity that is being executed, it is crucial to identify the initial and final action and moment in time of it. However, smooth transitioning between activities in sequence may be difficult to identify. One way of dealing with this problem is by creating different levels of abstractions when modeling the activity. Figure 3.2 presents this scenario.

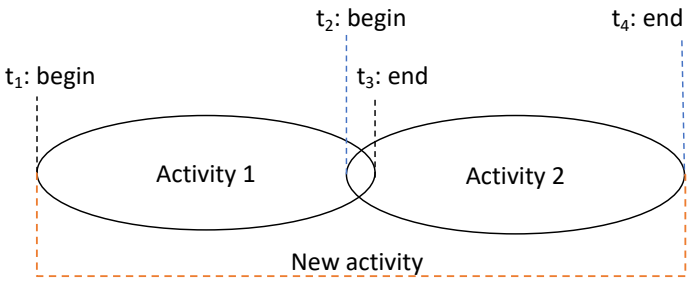


Figure 3.2: Creation of different levels of abstractions for activities.

The image shows the grouping of small activities performed in sequence, where the transition moment is not precise. It is essential to highlight that this new resulting activity may also suffer from the label jitter

uncertainty in a broader dimension. Thus, a verification of the necessity of identification of the activity's boundary must be done. This approach can be used allied with classification algorithms, aiming at the identification of smaller activities.

Still considering figure 3.2, activities 1 and 2 could be defined as atomic, when they represent small (or simple) tasks that cannot be divided into smaller ones Liu et al. (2016). This sort of activity can be easily defined through graphical models. However, complex tasks become a challenge to overcome. Complex activities can usually share time and resources and, thus, must be analysed as concurrent.

Another obstacle related to uncertainty in context-aware systems is related to modeling limitations. Models and approaches presented nowadays are limited, lacking expressiveness to map complex temporal relations among different activities Liu et al. (2016), including the transitions. The authors discuss the advantages of using a semantic-based approach over graphical modeling. Limited power to represent complex relations, the difficulty of defining more than one type of takes in the same network are some issues faced by graphical models. Besides that, this type of approach considers only three-time point relations: precedence, equality, and following of evidence.

The possibility of reuse and expressiveness provided by semantic approaches made them gain attention over the years. However, in some cases, uncertainty is also challenging to address. In Liu et al. (2016) it is addressed the main obstacles when dealing with modeling limitations that can lead to uncertain scenarios. The first refers to high-level classifications, considering different levels of abstractions to identify activities. Predictions are addressed by considering the probability of an event to occur in the future if it happened in the past. In order to do that, past event verifications are also analysed by observing their interval. Concurrent activities are also monitored.

Different causes may affect the correct receiving of sensor data by the context-aware system. Examples include communication problems that may compromise the data, hardware issues such as battery and activation Noor et al. (2016). The consequence is that the system will not have all the needed data to identify the activity.

Thus, several approaches in the literature tackle uncertainty in activity recognition. Dempster-Shafer's theory, for instance, allows the definition of weights for unknown facts Noor et al. (2016). This phenomenon is called total ignorance and, when allied with contextual evidence, allows the definition of degrees of beliefs. Another widely used approach refers to mathematical solutions, such as probabilistic theory, which also uses degrees of beliefs to deal with unknown facts.

In both cases, the main goal is to deal with the uncertainty that comes from the incompleteness of context data. In other words, when, for some reason, the system cannot gather all the necessary data to recognize the activity. In this paper, the main goal is to use a decision tree to classify human activities. The resulting classification will be used to provide data to fill gaps in contextual situations.

Default values can be used to complement structures that model situations. These values must be pre-defined and should pass through validation in order to be used to fill gaps in data. In order to recognize an activity, the system must receive a set of data sent by physical sensors. The processing of them can generate new data that also will help the recognition. Figure 3.3 illustrate this situation.

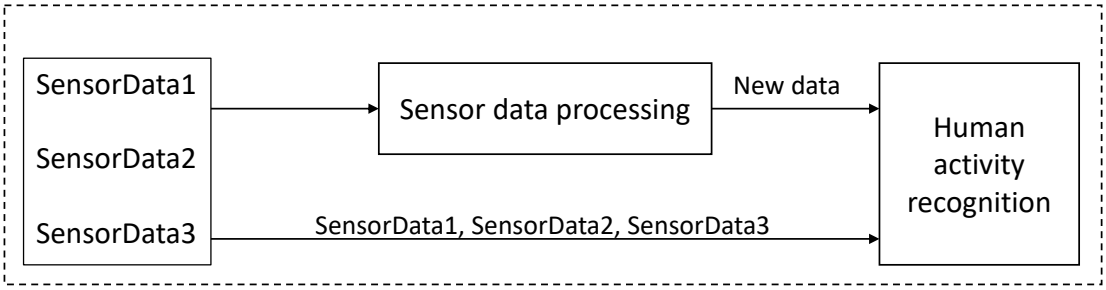


Figure 3.3: Processing as context data source.

The third phase of activity recognition can be composed of many algorithms. For instance, decision trees would be able to classify previous activities and fill gaps of data with default values (figure 3.4).

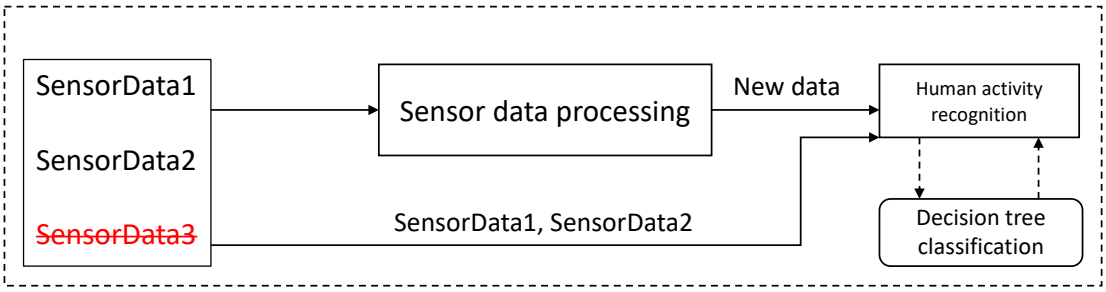


Figure 3.4: Default values based on decision tree classification

The image presents a hypotactic scenario where the system did not have access to all context data. It expects data from three physical sensors but received from only two of them. The system processes the known data and tries to derive complete information that helps the recognition of an activity. Meanwhile, a call to a decision tree is created to verify if it is possible to fill the gap of the missing value with default values. Thus the context recognition would not be performed with less data than necessary.

3.5 Summary

Uncertainty in a context-aware system is characterized by the use of incomplete, ambiguous, or outdated data. This Chapter started by evidencing issues related to the capturing and processing of context data. Problems related to these sources may lead to uncertainty in context-aware systems.

Problems related to the capturing and processing of data that leads to uncertain situations were evidenced. Requirements to model and develop context-aware systems were addressed once they may prevent problems of this source. Two types of uncertainty were discussed, *aleatoric* and *epistemic*, as well as the possibility of using belief values to fill gaps of data, as proposed by *speculative computing*.

The relation of Quality of Context, Quality of Services, and Quality of Devices was evidenced. The advantages of semantic modeling approaches to deal with uncertainty were presented, highlighting the importance of numeric representation for probabilistic processing. Parameters like accuracy, usefulness,

up-to-dateness, and types of acquisition and representation of context data were characterized along with the proposition of a formal representation of them as QoC.

The importance of identifying sources of problems that lead to uncertain models was discussed. By analyzing the level of randomness and identifying inaccurate measurements, it is possible to tackle this problem. The natural dynamicity of the context data of intelligent environments should also be taken into account when creating representations of the real-world.

At last, the importance of presenting uncertain situations to users, aiming to either confirm an inference or to solve problems of lack of context data, was addressed. Users should interact with context-aware systems naturally. Thus, the more they trust the systems, the more they will feel comfortable using them. This results from consistent displaying of information. Relevant notifications based on the correct processing of context data contribute to the consolidation of this phenomenon. Hence, the relation between the probability of certain and uncertain context was discussed with its impact on the presentation of information. Some related works with different approaches were briefly described.

Decision trees will be discussed in the next Chapter as an introduction to the approach used in this thesis.

Chapter 4

Applying Machine Learning to Context Awareness

The main goal of this Chapter is to summarize some of the existing Decision Tree algorithms (ID3, C4.5, and CART). They are characterized, and scenarios of the running example are applied to the CART algorithm to facilitate its understanding once used in this Ph.D. work. Details related to the GINI impurity and Information Gain are presented. At last, the pruning of trees is discussed.

The process of learning involves different visions in the literature. It encompasses the transformation of data into information, then into knowledge towards wisdom. Referring to the general process of learning, the AI founder and co-founder of the MIT AI Laboratory, Marvin Minsky, stated that "*Learning is making useful changes in our minds.*". This way, intelligent systems with learning capacities can identify and absorb changes regarding the context of the data and adapt their features to deal with such scenarios correctly.

According to Herbert Alexander Simon, Nobel Prize in Economics (1978), "Machine learning is concerned with computer programs that automatically improve their performance through experience". Therefore, ML provides the possibility of improving the efficiency of computing systems by analysing data characteristics and transforming it into useful information. By understanding the behaviour of the data, it is possible to use it for further analysis and decision-making. The same author also states that "Learning denotes changes in the system that are adaptive in the sense that they enable the system to do the same task (or tasks drawn from a population of similar tasks) more effectively the next time". The efficiency can be evidenced through different perspectives, such as finding the best path to achieve a goal or ignoring unnecessary features to process a task.

Computer systems capable of learning are characterized by the ability to improve themselves automatically, based on their experiences (Mitchell, 1997). Thus, the learning process refers to consider processing from the past and use this experience to improve future tasks.

Machine learning (ML) approaches are in evidence nowadays due to, among other things, their capacity of providing well-structured data for the improvement of autonomous decision-making. A wide range of algorithms allows data to be classified and grouped according to similar features. Also, ML can be used in data analysis for forecasting future situations based on known data.

Several fields use solutions provided by ML, for instance, pattern recognition, entertainment, computer vision, and others (El Naqa and Murphy, 2015). Context awareness can also be benefited from it. Specifically, systems for the domain of ADL can classify, recognize and predict human activities based on the pattern of behaviour of the user. Machine learning can be divided into three main types, supervised, unsupervised, and reinforcement, according to the characteristics of the algorithms. Figure 4.1 (Anisha and Polati, 2021) provides an overview of the types of ML with respective possibilities of fields of application.

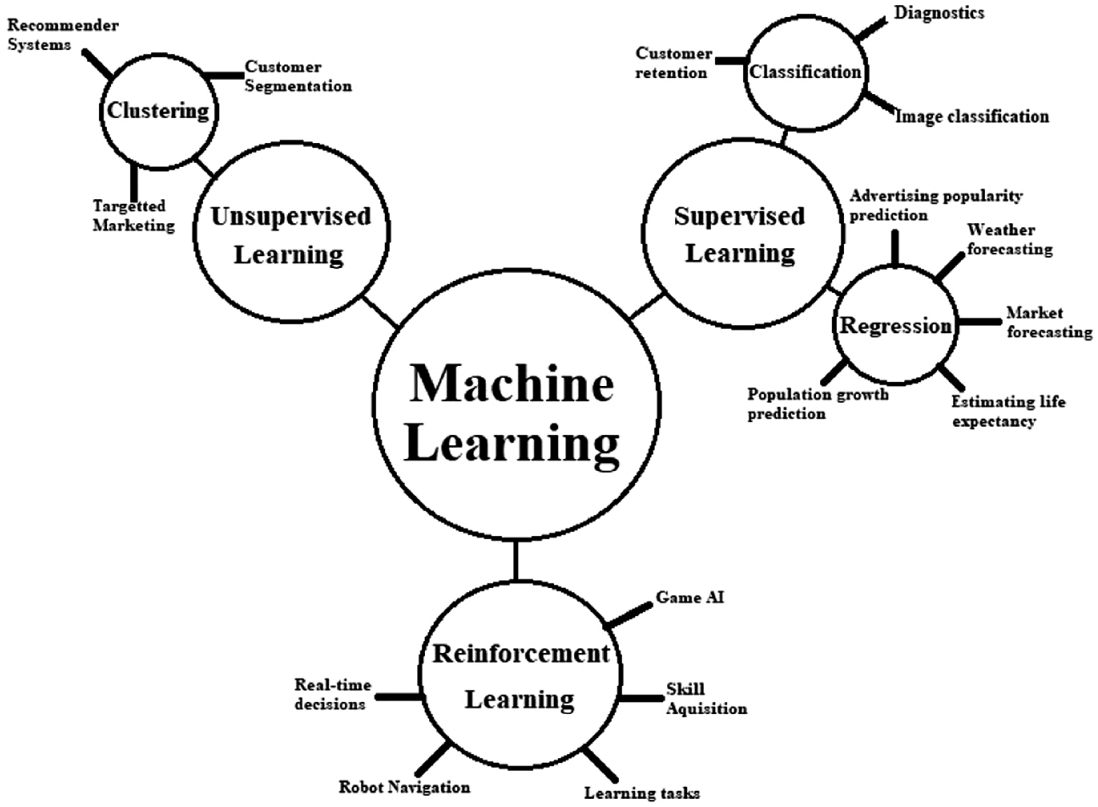


Figure 4.1 Overview of machine learning (Anisha and Polati, 2021).

The approach used in supervised learning takes into consideration labeled data for the training. The examples of the dataset are composed of features that are used to characterize them. These features are analysed individually and by grouping them with the aim of identifying which label should be applied to the example. Thus, based on that, it is possible to predict new inputs from the same universe. Supervised algorithms need labeled data for the input once all the learning process is based on that (Canepa, 2016). Large datasets with a good amount of features tend to improve the accuracy of the predictions.

Unsupervised learning uses data without labels to process a dataset. In this case, the main goal is not to classify examples with a high level of accuracy for each of them. The idea is to separate those with similar features into clusters. It is necessary to define a list of items to be used as limits between the clusters. In other words, these items determine to which of the clusters an example belongs to. Different from supervised learning, where the total number of labels is known from the beginning, here the total of clusters can vary, according to the similarities found in the examples (Alpaydin, 2014).

At last, reinforcement learning, like the unsupervised, also does not consider labeled data during the process. However, the approach here is based on scores. The higher is the similarity of a set of features for a specific element, the more its scores. Thus, the process rewards those with more common characteristics and punishes those with less common characteristics. The goal is to find the path for maximizing the rewards based on the features (Canepa, 2016).

The main goal of this Ph.D. thesis is to cope with uncertain and incomplete data used by context-aware systems. Supervised learning classification of data can help mitigate the lack of context data. Decision tree solutions raise as an efficient alternative, providing an exemplary structure of the data. After classifying the dataset, its result can fill gaps when the system cannot access the data from sensors.

Thus, the following section provides an overview of the most used algorithms for building decision trees and presents the development of one tree to be applied to the datasets of this work.

4.1 Decision Trees

Decision Tree (DT) is one of the most used approaches to deal with supervised learning, and it is based on Information Theory. There are different algorithms to implement them, such as ID3, C4.5, and CART. Each of them uses different approaches. Decision Trees are mainly used for classification problems through the identification of similar elements and their characteristics.

In DTs, the input refers to sets of examples represented by registers of one dataset. The dataset comprises a group of columns representing the features of the attributes and one column representing the label. This label refers to the prediction of the data (classification). Sometimes the dataset cannot be entirely separated (100%). Some of the examples could fit in more than one label.

The main goal of DT is to partition a set of examples into subsets, aiming to separate them into more homogeneous groups, according to the classification labels. The subsets are then partitioned into other subsets to improve this homogeneity. The process is repeated until the data is as classified as it can get.

The internal nodes of the tree (including the root) use values of features as parameters to separate the data. They represent tests and are used to perform the classification.

Some DT algorithms use entropy to calculate the amount of information provided by the examples of the dataset. Also, information gain is used to estimate which feature should be considered to separate the data into subsets in the nodes. The entropy, i.e., information provided by the distribution (P = $p_1, p_2, ..., p_n$), is estimated by (Hssina et al., 2014) :

$$Entropy(p) = -\sum_{i=1}^{n} p_i \log_2(p_i)$$

The result is used to find the gain of information, as follows:

$$IG(p,T) = Entropy(p) - \sum_{j=1}^{n} (p_j * Entropy(p_j))$$

Where:

- p_j represents all values for the feature T in the dataset. I.e., all valid values for that column.

The feature with higher information gain will be used in the current node of the tree. This process is performed with values from the training set.

The following sections describe the characteristics of different DT algorithms, namely ID3, C4.5, and CART.

4.1.1 ID 3 Algorithm

The ID3 algorithm (Quinlan, 1986) uses the information gain result to classify the test data. It was the first proposed for this goal and has some limitations compared to more recent (C4.5 and CART). For instance, it does not deal with uncertain data (missing values), nor can it classify discrete attributes. The ID3 algorithm uses entropy and information gain to find the best relation of feature/value to be used in the node.

To find the relation feature/value in the root node, it is necessary to apply the entropy and information gain to all possibilities. This means that every value of all columns (features) of the dataset must be considered. For instance, considering a dataset that classifies daily activities that a user performs along an ordinary day at home (table 4.1, adapted from Ordónez et al. (2013) and described in details in Chapter 6), the entropy would be the following:

- Total of examples: 14;
- Number of different activities: 5;
- Total of features: 3;
- Label for classification: Activity.

$$Entropy(S) = -\frac{3}{14} * \log_2(\frac{3}{14}) - \frac{3}{14} * \log_2(\frac{3}{14}) - \frac{2}{14} * \log_2(\frac{2}{14}) - \frac{4}{14} * \log_2(\frac{4}{14}) - \frac{2}{14} * \log_2(\frac{2}{14})$$

After finding the entropy, it is necessary to calculate the gain of information for each value of a specific feature. For instance *Place*:

Location	Type	Place	Activity
Door	PIR	Living	Grooming
Door	PIR	Living	SpareTime_TV
Maindoor	Magnetic	Entrance	Leaving
Door	PIR	Bedroom	Sleeping
Fridge	Magnetic	Kitchen	Snack
Door	PIR	Bedroom	Sleeping
Door	PIR	Bedroom	Sleeping
Maindoor	Magnetic	Entrance	Leaving
Door	PIR	Bedroom	Sleeping
Door	PIR	Living	Grooming
Microwave	Electric	Kitchen	Snack
Door	PIR	Living	Grooming
Door	PIR	Living	SpareTime_TV
Seat	PIR	Living	SpareTime_TV

Table 4.1: Dataset with routine activities examples.

$$Gain(S, Place) = Entropy(S) - \frac{6}{14} * Entropy(Living) - \frac{2}{14} * Entropy(Entrance)$$

$$-\frac{4}{14} * Entropy(Bedroom) - \frac{2}{14} * Entropy(Kitchen)$$

This process must be performed for every feature of the dataset. The one with the highest gain should be used in the node (starting from the root).

According to Hssina et al. (2014), the ID3 algorithm presents limitations regarding features with a high number of different values. Thus, this should be considered before choosing this approach.

4.1.2 C 4.5 Algorithm

The C4.5 algorithm (Quinlan, 1993) overcome some challenges faced by ID3. It uses information gain to find a gain ratio. This is taken into account to define the feature/value that should be used in each node to split the data. The gain ratio is estimated as follows:

$$GainRatio(p, T) = \frac{Gain(p, T)}{SplitInfo(p, T)}$$

Where:

$$SplitInfo(p, T) = -\sum_{j=1}^{n} p'(\frac{j}{p}) log_2(p'(\frac{j}{p}))$$

The $p'\frac{j}{p}$ refers to the proportion of element in a specific position(p). This approach does not consider the distribution of elements in the dataset. The splitting of data is performed considering the gain ratio, which defines how much impurity it may have. Features with a higher level of gain ratio tend to provide more homogeneous splitting, i.e., with less impurity. In other words, at each node, the gain ratio is measured

for each feature/value, and those with better gain are used to separate the current dataset into subsets. These subsets should contain more homogeneous data once the process of classification is going further.

Another characteristic of C4.5 is the possibility of dealing with uncertain values by using only examples defined for that feature. The classification is performed by estimating different outcomes. the new criteria to find the gain is the following (Hssina et al., 2014):

$$Gain(p) = F(Entropy(p) - Info(p, T)$$

Where:

$$Info(p, T) = \sum_{j=1}^{n} (p_j * Entropy(p))$$

The C4.5 algorithm deals with continuous data by assuming that an attribute represents an interval of values. These values must be analysed in ascending sequence, i.e., the interval should be ordered. After that, a valid value is taken as a parameter (K), and the group is divided into two subsets: one containing those that are lower or equal to K and; one with higher values. The gain and ratio gain is calculated in all the splits.

4.1.3 CART Algorithm

CART (Classification And Regression Tree) algorithm (Breiman et al., 1984) uses GINI Impurity and Information Gain to perform this quantification. GINI Impurity quantifies the uncertainty in the tree nodes, i.e., how uncertain is the question of each feature. Information Gain verifies how much of the uncertainty found by GINI Impurity can be reduced. By calculating the GINI Impurity and Information Gain, finding the most appropriated featured to be used in questions of each node is possible. Based on this, the tree nodes are created recursively, and in each of them, a question is made, and, consequently, a subgroup is divided into two other subgroups. When there are no further questions to be done in a node, it means that it represents a leaf of the tree.

First, it is necessary to define a root for the tree. While creating the tree, all the nodes should receive a list with lines (examples) of the dataset as input. These sets of examples may represent subgroups of the entire dataset. The root node always receives the entire dataset that is being used as the training set.

Each node should contain yes/no questions about one of the features (e.g., Is the Type *Magnetic*?, Is the Place *Entrance*?, etc.). Based on the answer, the dataset is divided into two subgroups. One containing only the examples in which the answer was true and the other with the examples with an incorrect response.

The main goal of these questions is to find the purest possible distribution in each of the nodes. Thus, to build an effective tree, it is necessary to find appropriate questions for each node. To do that, it is necessary to quantify questions and verify how they contribute to the distribution of the labels.

To find an appropriate question, it is necessary to iterate on all the values of each feature of the examples received by the node. Considering that, apart from the label, all other features (columns) can be used as a basis to create a question. Features can be used in questions of different nodes, only changing

their value. For instance, considering table 4.1, one node "A"can use the feature Type and the value Magnetic to create the question: "Is the Type *Magnetic*?". In another node, "B"the same feature is used, changing only the value to PIR, creating the question: "Is the Place *Entrance*?". Figure 4.2 presents this scenario.

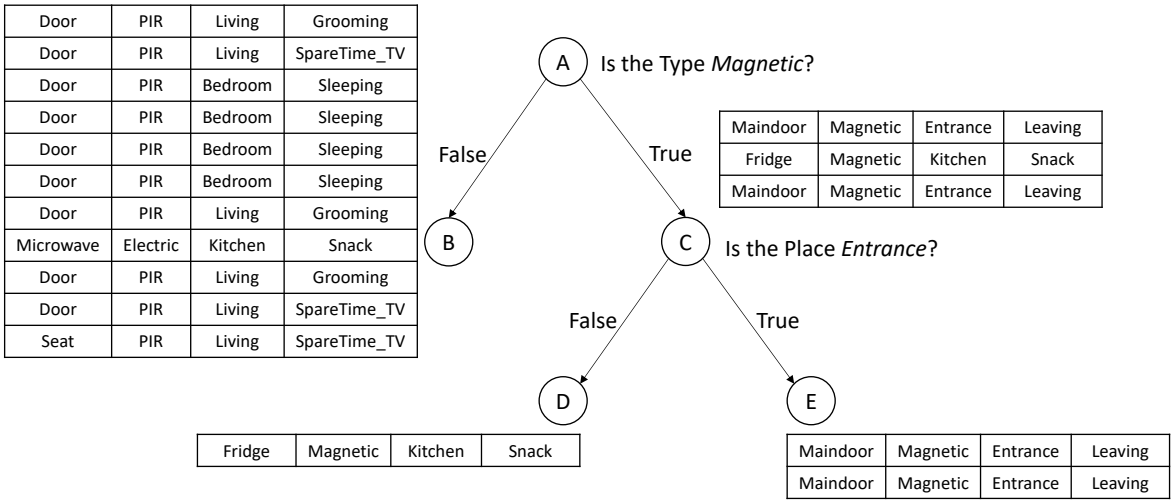


Figure 4.2 Example of Decision Tree.

From this image, it is possible to evidence that, with only two questions (nodes A and C), two types of activity were identified from a total of five different activities. The best questions are those that reduce the most of uncertainty of the answers. As stated above, GINI Impurity and Information Gain help this process. The quantification of uncertainty (GINI Impurity) varies from 0 to 1, where values near zero represent less uncertainty and values near one represents more uncertainty.

GINI Impurity refers to the probability of incorrectness of an aleatoric relation between one label and one example based on a specific feature as criteria. For instance, assuming a group of five different objects and another group with five different labels representing the name for these objects, the aleatoric relation between one object of each group has a probability of 80% of being incorrect, i.e., 1 - 1/5 = 0,8. And, consequently, only 20% of being correct, i.e., 1 - 0,8 = 0,2.

However, most of the time, in realistic situations, the proportions of the groups will not be the same. In table 4.1, there are fourteen examples to only five labels. In these cases, GINI Impurity is the result of the sum of the probabilities for each feature, as follows:

$$G(k) = \sum_{i=1}^{n} P(i) * (1 - P(i))$$

After finding the GINI Impurity value for the Yes and for the No questions, it is necessary to calculate the average among them.

The information gain results from the difference between the impurity of the entire training set and the average of the impurities of the yes/no questions. The feature with more information gain should be used

as the basis for questions in the node.

4.1.4 Pruning

The main goal of pruning a DT is to minimize the prediction error rate (Hssina et al., 2014). Many times, processing training data leads to overfitting. This impacts the analysis of new data over time. Pruning a DT refers to the process of eliminating some of the branches that have a low contribution to the data classification. This improves the accuracy of the prediction and reduces the complexity of the classification process (Hssina et al., 2014).

Pessimistic associated error is used to analyse the total instances, considering that a specific amount of them do not appear in the most regular classes. Considering that, the estimated error is calculated by:

$$EstimateError = NK$$

Where:

- N is the total of examples;
- K is the total of errors.

The estimated error of branches is added and compared to sub-trees. If the result is higher than the leaf, the sub-tree is pruned.

4.2 Summary

This Chapter discussed the topic of *Decision Trees*, a supervised learning approach that aims to classify data according to similar characteristics. Three algorithms were described emphasizing their features (ID 3, C4.5, and CART).

The dataset used as input was described, differentiating the roles of the group of attributes from the label, which is the base of the distribution and refers to the prediction of the data(the classification itself). The stages that compose the classification were explained, by detailing the process of dividing the tree into two subsets, according to partial splitting. This process is repeated, aiming to get the most approximated pure separation of data as possible. The features of the dataset are used to perform this task. They are used to create questions in which the result leads to homogeneous subsets of data. The fact that examples may fit in more than one label may lead to a not perfectly pure result, i.e., items cannot be completely isolated from others.

The importance of using *Entropy* to identify the amount of information provided by the rows of the training set and the *Information gain* to define how to separate data along the tree nodes were discussed. Characteristics about these parameters were evidenced in the sense of the importance of understanding their meaning and how they influence the division of a subset. The formal definition of *Entropy* and *Information Gain* was described, highlighting how they can be applied to the creation of a decision tree.

A second dataset was briefly presented in order to be used in the explanations of the algorithms. Further details about it are discussed in Chapter 6.

ID 3 algorithm was characterized. Its efficiency was discussed by presenting an example of use. Advantages and limitations were evidenced. The fact that it does not support datasets with missing values becomes a drawback when compared to others. Also, results may not be satisfactory when using large distributions.

C 4.5 algorithm uses *gain ration* as a metric for the definition of which attribute to be considered for the splitting of data into subsets. In other words, it considers the level of impurity present in the dataset. The higher is the *gain ratio*, the more homogeneous the separation of data tends to be. Contrasting with ID 3, C 4.5 does not consider the number of different labels for the classification, and it is able to deal with datasets that contain missing values.

The third algorithm reviewed in this Chapter was CART, which was chosen to be used for the development of the proposal of this thesis. Different from the other two algorithms, it uses *GINI Impurity* to identify the level of uncertainty present in the dataset, and based on that, it calculates the *Information Gain* to find how much of it can be reduced with the categorization. The approach was applied to a subset of the second dataset aiming for more precise visualization of how the classification is created. Details about the formalization of *GINI impurity* and *Information Gain* were addressed, with the discussion of the results for that particular example.

At last, this Chapter presented a brief examination of the possibility of pruning decision trees. Characteristics of this process were given, highlighting advantages and how to calculate the adequate *Estimate error* for the pruning tree branches.

The following Chapter discusses the topic of Formal Grammar, more specifically, how Attribute Grammars can contribute to the consolidation of context-aware environments.

Chapter 5

Formal Grammars

Concepts related to formal grammars are discussed here. Context-free grammars are described with their characteristics, structure of derivation tree, and dependency graph. Attribute grammars are considered an extension of it and are also addressed by describing an algorithm, the derivation tree, and formal structure. A possibility of relation with intelligent environments and application to a context validation is discussed. An example with production and semantic rules is detailed.

Formal grammars were initially proposed by Chomsky (1956) to describe natural languages and are widely used in programming languages to validate its syntax (Slonneger and Kurtz, 1995). They allow the definition of the shape of symbols, creating restrictions, giving the language a strict notation (Henriques, 2013).

Chomsky (1956) represented formal grammar through a finite set of the terminal (Σ) and non-terminal (N) symbols, a finite set of production rules (P), and one starting symbol (S). The following tuple represents the generic form of formal grammars.

$$CFG = < \Sigma, N, P, S >;$$

The terminal symbols represent the alphabet of the language. The non-terminal represent syntactic categories, which include subsets of sentences. Production rules are used to describe the behaviour of the non-terminal symbols, i.e., from what they will be composed of or what terminals and non-terminals are allowed. The starting symbol is the most important in grammar, once it represents the structure of the sentence of the language (Slonneger and Kurtz, 1995).

The structure provided by this tuple is used as basis for the definition of four types of formal grammars (Chomsky, 1956): *unrestricted*, *context-sensitive*, *context-free* and *attribute grammars*. Considering the scope of this research, the following sections will address only *context-free* and *attribute grammars*.

Thus, this Chapter aims to introduce concepts related to Context-Free Grammars and, after that, Attributes Grammars. At last, it will be described the relation between Attributes Grammars and Intelligent Environments, highlighting how they can contribute to the evolution of such domains.

5.1 Context-Free Grammars

Context-Free Grammars (CFGs) are a formal notation to describe languages. They provide definitions for symbols, how they must be structured, and how they can be combined to represent the sentences of a language (Henriques, 2013). CFGs are used to define meta-languages, i.e., a language to define languages. It has the restriction of allowing only one non-terminal in the Left-Hand Side (LHS) of the rule. The Right-Hand Side (RHS) may be composed of terminals and/ or non-terminals (Slonneger and Kurtz, 1995). Thus, considering **P** as the set of production rules and **p** being each of them (where **p** $\in$ **P**), the following code presents the generic structure of a production rule of a CFG.

p1: S $\longrightarrow$ N;

p2: N $\longrightarrow$ N1 | Σ | N2;

p3: Σ $\longrightarrow$ <set of terminal symbols>.

Where the arrow symbol ($\longrightarrow$) represents the derivative operator.

In **p2**, it was used numbers to differ non-terminals among themselves. The productions defined by the three rules represent a complete structure of a language. It can vary from a straightforward terminal to a range of terminals and non-terminals grouped, following the specific restrictions defined for the language. The first rule (**p1**) presents the starting symbol with its derivative production. It represents the structure of the grammar's axiom (Henriques, 2013). The RHS of this rule can never be a terminal once it would restrict the language to only the terminal.

The second production (**p2**) describes how the terminal symbols can be combined through the use of non-terminals. Some variations of the RHS of this rule is the following:

p2.1: N $\longrightarrow$ N1;

p2.2: N $\longrightarrow$ N1 | Σ;

p2.3: N $\longrightarrow$ Σ | N1;

p2.4: N $\longrightarrow$ Σ;

The terminal symbols are produced in **p3**. According to Henriques (2013), they represent reserved words, signs, and terminal classes. These terminals are applied to all other rules where the symbol Σ is present in the RHS. For instance in the productions **p2.2, p2.3** and **p2.4**.

It is important to address that a CFG has as its main goal dealing with syntactic issues of a language. It does not address the semantics. The results of the productions do not validate the base case of their recursion, as presented following in the example:

N $\longrightarrow$ N1

N1 $\longrightarrow$ ((N2, Σ))

N2 $\longrightarrow$ (((N3, Σ), Σ))

N3 → (((N4, Σ), Σ), Σ))

N4 → ((((N5, Σ), Σ), Σ), Σ))

N5 → (...)

In order to finish this recursion, at some point, **Ni** must assume the value of a terminal symbol once the terminal represents the alphabet of the language. In other words, to finish the recursion, the production of **p2.4.** must be executed.

Each production of the grammar results in a Derivation Tree where the root is the LHS's symbol. The RHS's symbol(s) represent(s) the descendants of it (Henriques, 2013). Figure 5.1 presents the Derivation Trees for each of the aforementioned productions, including the starting and terminal symbols (respectively **p1, p2.4** and **p3**).

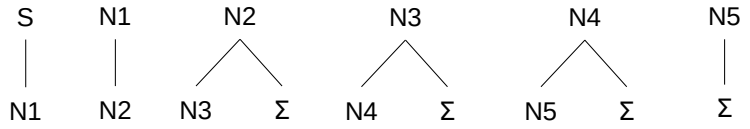


Figure 5.1 Derivation tree of the generic structure of Context-Free Grammars.

The complete Derivation Tree is the result of the combination of the productions, as shown in figure 5.2.

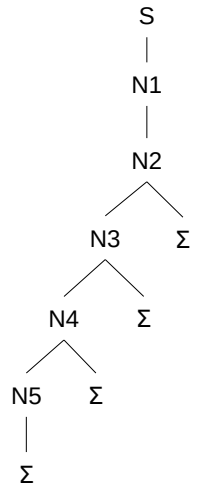


Figure 5.2 Derivation tree of the generic structure of Context-Free Grammars.

This derivation tree presents the deepness of five recursions regarding the production **2**. The verification of recursive non-terminals among the productions can be done by creating directional graphs called Dependency Graphs. The existence of full cycles involving non-terminals represent recursions, and, consequently, this must be properly treated (Slonneger and Kurtz, 1995). Figure 5.3 presents an example of this.

Considering that, according to Henriques (2013), CFGs are well-defined if all non-terminal symbols are used in the LHS in at least one production rule. Besides that, all of them must be achieved from the starting symbol. I.e., by executing the rules (beginning in S), it must be possible to reach all non-terminals

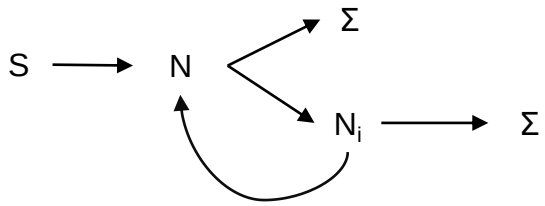


Figure 5.3 Dependency graph of Context-Free Grammars.

symbols. At last, from every non-terminal, it must be possible to reach terminal symbols in at least one production.

Thus, semantic validation still needs to be tackled. Attribute grammars address this issue, as described in the following section.

5.2 Attribute Grammars

Attribute Grammars (AGs) can be used for verification of inputs and generation of codes (Slonneger and Kurtz, 1995). They provide validation of context-sensitive aspects of a language (Knuth, 1968). For instance, a programming language allows the verification of variables declaration and if the values ascribed to them correspond to their type (Slonneger and Kurtz, 1995). Thus, while CFGs can perform syntax validation, AGs can check for the meaning of them by creating restrictions to be applied in the definition of the syntax constructions (Henriques, 2013).

Attributes Grammars extend the concept of Context-Free Grammars by adding semantic expression to the validation provided by them. This makes Attributes Grammars become an excellent approach to deal with situations where users and/ or machines may influence the behaviour of applications.

According to Slonneger and Kurtz (1995), each symbol of the grammar should have a set of attributes with rules, restrictions, and values assigned to them. AGs are composed of synthesized and inherited attributes. Assuming the sentences of a language organized in a parse tree, the synthesized attributes allow the values to be used by its parents (one higher level in the tree) and by its children through inherited attributes (one lower level in the tree). This provides the possibility of their manipulation at any node of the tree.

Code 5.1 presents an example of the generic structure of production rules of AGs, based on the definition of Slonneger and Kurtz (1995) using only synthesized attributes.

In line 1, the starting symbol represents the axiom and defines all language sentences' structure. It is composed of one or more non-terminal symbols. In lines 2 and 3, it is defined the restrictions that the non-terminals must obey in order to produce correct sentences. Lines 4 to 7 define the productions for the symbols. The LHS must be non-terminal. The RHS can be a terminal, non-terminal, or a combination of both. In the example, the first production (line 4) refers only to a terminal. If that is the case, the consequence defined in line 5 will be executed. Line 6 presents an alternative to line 4, where the same production refers to a combination of one non-terminal and one terminal. If the analysis of the input

Algorithm 5.1 Generic structure of AG with synthesized attributes

1: $S \rightarrow non-terminal_1,\ non-terminal_2,\ ...,\ non-terminal_n$
2: **Restriction** :
3: $< restrictions\ for\ the\ sentence\ to\ be\ valid >$
4: **if** $non-terminal_1 \rightarrow TERMINAL$ **then**
5: $Consequence_1$;
6: **else if** $non-terminal_1 \rightarrow (non-terminal_{1.1},\ TERMINAL)$ **then**
7: $Consequence_2$;
8: **end if**
9: **if** $non-terminal_2 \rightarrow TERMINAL$ **then**
10: $Consequence_3$;
11: **else if** $non-terminal_2 \rightarrow (non-terminal_{2.1},\ TERMINAL)$ **then**
12: $Consequence_4$;
13: **end if**
14: [...]

identifies this situation, the consequence defined in line 7 is executed. Lines 8 to 11 presents the analysis of the second non-terminal. Productions with a similar structure (represented in line 12) must be defined for each of the non-terminals present in the RHS of the sentence (line 1).

In this scenario, consequences 1 and 2 are represented by synthesized attributes. Non-terminals analysis begins in the leaves and travels through the nodes until the structure of the sentences, provided by the starting symbol. Figure 5.4 presents the same structure through the parse tree for this grammar.

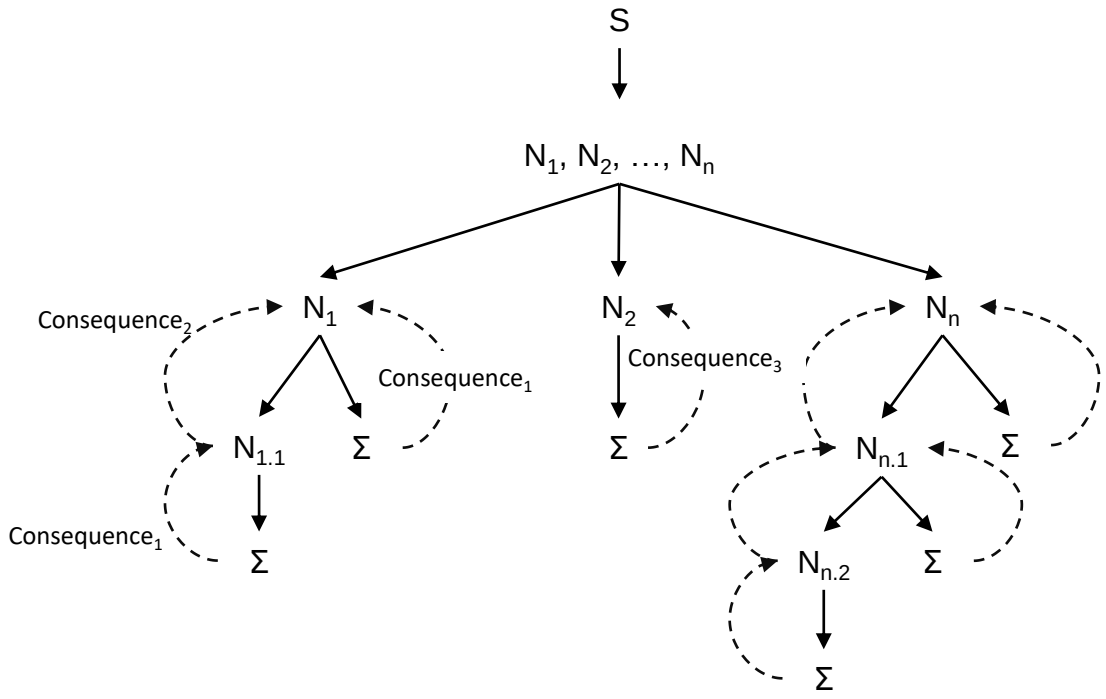


Figure 5.4 Derivative tree with synthesized attributes.

Another approach for parsing sentences of a language is through synthesized and inherited attributes. This is advised when the sentence is composed of more than one non-terminal. Thus, a relation between

the symbols must be established. The analysis of the first one is performed with synthesized attributes and the others with inherited attributes. The following code (5.2) is based on the definition presented in Slonneger and Kurtz (1995) and describes this situation.

Algorithm 5.2 Generic structure of AG with synthesized and inherited attributes

```
1:  S → non − terminal_1, non − terminal_2, ..., non − terminal_n
2:  Restriction :
3:        < restriction that validates the relation of non − terminal_1 and non − terminal_2 >
4:        < ... >
5:        < restriction that validates the relation of non − terminal_1 and non − terminal_n >
6:  if non − terminal_1 → TERMINAL then
7:      Consequence_1;
8:  else if non − terminal_1 → (non − terminal_1.1, TERMINAL) then
9:      Consequence_2;
10: end if
11: if non − terminal_2 → TERMINAL then
12:     Restriction :
13:           < Restriction analysis of non − terminal_2 >
14: else if non − terminal_2 → (non − terminal_2.1, TERMINAL) then
15:     Consequence_3;
16: end if
17: [...]
18: if non − terminal_n → TERMINAL then
19:     Restriction :
20:           < Restriction analysis of non − terminal_n >
21: else if non − terminal_n → (non − terminal_n.1, TERMINAL) then
22:     Consequence_n+1;
23: end if
```

Following the same principle of code 5.1, in line 1, the starting symbol analyses the structure of the sentences, which may be composed of one or more non-terminal symbols. The difference is that, here, the grammar associates synthesized attributes to non-terminal$_1$ and inherited attributes to all other non-terminals (lines 2 to 5). Considering that, the processing of the first non-terminal (lines 6 to 9) has the same structure as the first example. If one of the other non-terminal symbols derives a terminal, for instance, in line 10, specific restrictions related to it must be verified in order to ensure the parsing of the corresponding input's subset (lines 11 and 12). This means that different from what happens in code 5.1 where the validation of the input is performed in the root of the tree, here, the validation is done in the current node, and its result is passed to its children. Line 13 presents an alternative production composed of one non-terminal and one terminal. If the parser identifies this situation, a specific consequence must be executed (line 14). The following lines (15 to 20) represent the same logic applied to the rest of the non-terminals present in the production of the starting symbol.

Using an approach with synthesized and inherited attributes makes it possible to manipulate them in any tree node. This means that the process of validation of an input sentence becomes easier once the search for inconsistencies can be performed at any time.

5.2.1 Formal Structure of Attribute Grammars

As aforementioned, attribute grammars extend context-free grammar by providing semantic validation to the sentences of the language. Considering that, according to Slonneger and Kurtz (1995), results of productions rules (derivations) of an AG will be satisfied if, and only if, a corresponding CFG is satisfied with all conditions being true.

Using the notation provided by Henriques (2013), the following tuple represents the structure of AGs:

$$AG = <CFG, A, ER, CC, TR>;$$

Besides all the elements of the CFG, it is also composed of a set of symbols' attributes (**A**). Depending on the semantic parsing, they can assume inherited or synthesized characteristics. The set of attributes of a given symbol **X**, represented by **A(X)**, is the result of its inherited and synthesized attributes (**A(X) = Inh(X)** $\cup$ **Syn(X)**). The values of attributes must obey semantic restrictions regarding to their domain of application. Semantic rules are associated with the productions in order to define those values.

Considering that every production rule belongs to the set of productions (**p** $\in$ **P**), the evaluation rules (**ER**) are used to create restrictions in the production rules by dictating what the attribute values could assume. The set of context conditions **CC** validates the semantics of the sentences, and **TR** is the set of translation rules.

The workflow for processing sentences of a language-driven by AG is the following (Henriques, 2013):

- In the derivation tree, the symbols are associated with a set of properties, referring to the attributes. This results in the Decorated Derivation Tree (DDT);
- The meaning of sentences result from the semantic analysis of the symbols, which are defined by the properties of the grammar's axiom;
- The translation rules allow the semantic analysis of sentences and symbols to achieve an expected output.

Sentences can only be processed after been validated in all these steps.

5.2.2 Attribute Grammars Applied to Intelligent Environments

Attribute Grammar (AG) is a wide field of investigation with several relevant contributions regarding the validation of programming languages. It introduces semantic rules into Context-Free Grammars, inserting meaning to the analysis of the grammar (Knuth, 1968). Still, according to Knuth (1968), the semantics for the rules is achieved through the definition of attributes for non-terminal symbols by associating them with each production. This way, they are defined to manipulate attributes, which are associated with the grammar symbols.

The lexical and syntactical phases of processing precede the semantic. The lexical analysis has as the primary goal to analyse the inputs and convert them into a set of terminal symbols. The result of this

processing is used in the syntactical phase to generate an Abstract Syntax Tree (AST) Bürger et al. (2010). These two first phases can be applied to intelligent environments once this kind of domain should be filled with sensors sending raw data to be analysed by a context-aware system. Then, the semantic analysis is performed to take unevaluated attributes as input and return evaluated ones.

Attribute Grammars emerge as an excellent approach to define models to validate context data related to human actions. Despite having their origin coined to programming languages specification Knuth (1968), they are already used in Intelligent Environments field to analyse the structure of human tasks Freitas et al. (2019a). AG provides a formal and strict structure and the possibility of specifying semantics by defining attributes for symbols and evaluation or validation rules associated with production rules.

Another characteristic of AGs is the possibility of manipulation of the attribute values at any node of the tree Bürger et al. (2010). This feature is critical once in an intelligent environment, the states of entities (attribute values in AGs) may change frequently, and their values must be available to be used by the system at any time, with different purposes. A generic Abstract Syntax Tree of an intelligent environment is presented in figure 5.5.

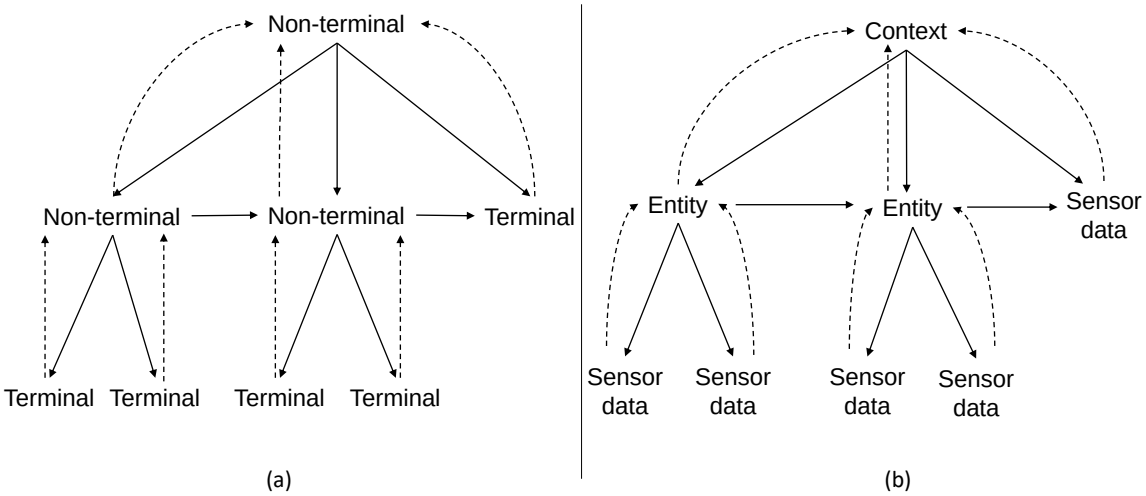


Figure 5.5 **a** General structure of AST. **b** Generic AST for intelligent environments.

The terminal symbols, in **a**, represent the raw data sent by sensors from the intelligent environment, in **b**. It refers to any data captured by the sensors (e.g., humidity, level of lightness, localization, the presence of entities in a room). It will be used to characterize new situations (new context). In the general structure **a**, it will be characterized as been synthesized or inherited attributes and can be analyzed in different nodes of the tree. The entities represent the non-terminal symbols in an AG and may assume all their characteristics. They can be composed of several data properties (attributes). Besides that, attribute values from one entity may be used by other entities. In these cases, they will assume synthesized or inherited features. From the generic AST, it is possible to define rules to validate the structure that will be applied context-aware system.

Thus, it is possible to state that AGs can contribute to the consolidation of intelligent environments through different approaches. This ph.D. work, it will address the possibility of applying AGs for the validation of activities of users.

5.2.3 Attribute Grammar for Context Validation

Intelligent environments are dynamic domains in which the state of entities (user, devices, and objects) may change frequently. Attribute grammars can easily handle this dynamicity.

The main advantage of attribute grammars is that they allow lexical, syntactic, and semantic data verification. According to Burger et al. (2010), in the first step (lexical analysis), the data is transformed into terminal symbols and will represent leaves (terminal nodes) in an Abstract Syntax Tree (AST). After this, symbols are organized into a hierarchical structure, creating the AST (syntactic analysis). At last, in the semantic analysis, this data is evaluated according to pre-established rules to ensure the complete correctness input required to produce correct output.

Attribute grammars are composed of production rules, which are defined using terminals and non-terminals symbols. The terminals represent the data itself, considering pre-established restrictions. Non-terminals can be seen as a composition of one or more terminals and are used to build the productions.

They specify the formalism that ensures that each entry passes through a verification regarding its composition. Considering human activities in smart houses, it was defined that it is essential to monitor the sensors that collect data, timestamps of the activities, local where the user performs them, devices, appliances, or objects, and a description with additional information.

Following, it is presented an example of productions that define the structure of a language (in that case, the input data structure).

p1: $context \rightarrow activity+$

p2: $activity \rightarrow id, record$

p3: $record \rightarrow date\_begin, hour\_begin, date\_end, hour\_end, local, description*, sensor\_state$

p4: $local \rightarrow place, room$

p5: $date\_begin \rightarrow sensor\_id, DATE$

p6: $hour\_begin \rightarrow sensor\_id, HOUR$

p7: $date\_end \rightarrow sensor\_id, DATE$

p8: $hour\_end \rightarrow sensor\_id, HOUR$

p9: $place \rightarrow sensor\_id, appliance$

p10: $room \rightarrow TEXT$

p11: $description \rightarrow TEXT$

p12: $sensor\_id \rightarrow TEXT$

p13: $appliance \rightarrow TEXT$

p14: $sensor\_state \rightarrow SET\_OF\_STATES$

The symbols that appear in capital letters are terminals. All of them were defined in the grammar in a way that their values were correctly validated. For instance, in **p5** the symbol DATE has restrictions to accept only valid dates in the format YYYY-MM-DD. In **p14**, the symbol SET_OF_STATES was defined to accept specific values within a pre-defined interval.

The first rule (**p1**) defines the axiom of the grammar. It means that it restrains the type of acceptable inputs. The verification performed in all other production rules will affect this axiom. Considering the domain of daily human activities, it states that a context can be composed of one or more activities. This was defined because, depending on what the user performs, it will be necessary to integrate two or more small actions to represent an activity and, consequently, the context of the user. The second rule (**p2**) states that each activity performed by the user must contain an identification code of the sensor that sent the data and a record. Identification of sensors is important since one of the phases when dealing with uncertain situations is identifying its source. Thus, the system should perform a sensor's diagnostics to find problems related to a specific one. More details about this process is given in section 6.3.1.

Rule **p3** defines the structure of record. This case states that they should be composed of timestamps that identify the beginning and ending date and hour of the occurrences. Depending on how the system was developed and the types of sensors available, they can provide additional data. Thus, this rule also has a *description* symbol to ensure that any extra information about an activity will be considered (**p11**).

Rules **p5** and **p7** verifies the structure of the dates of the beginning and ending that compose the timestamps. Rules **p6** and **p8** check the beginning and ending hours of it. Rule **p4** defines the structure of the place and room of the house where an activity is being performed. The *place* symbol in **p9** validates the specific local, furniture or appliance where sensors are installed (**p12**). It also validates its identification code. The symbol *room* validates the local of the house where the user is executing the tasks (**p10**).

Following, figure 5.6 presents an AST of an attribute grammar applied to a scenario described in Lyons et al. (2010). It was adapted for the scope of this project and refers to a situation where a woman is sleeping in her bed, as she usually does. However, in the middle of the night, at 1:00 am, she wakes up feeling bad, with a stomach ache. The user decides to take medicine to relieve the pain. Motion and door sensors provide data for the system to infer that the user is going to the medicine cabinet in the bathroom. The woman takes the proper medicine and returns to bed. This activity has preparation steps (leaving the bed and going to the bathroom) and the action itself of having the medicine.

Aiming a better visualization, the AST was divided in two parts **a** and **b**. The former contains data related to the preparation steps. In this case, it represents a scenario where the user wakes up during the night, leaves the bedroom, and goes to the local where the medicines are stored. The latter contains data related to the action of taking medicine. Together, **a** and **b**, are children of the root of the tree, represented by the *context* symbol.

The values of the attributes in this example refer to data acquired from sensors. After adequately analyzing this raw data, the system fills the structure that the grammar will use to validate the activities. Thus, the attributes of the symbols in the grammar are represented by context data.

The preparation steps that the user accomplishes before performing an activity itself are naturally necessary (in the example, represented by the process of leaving the bedroom). Thus, it is crucial for the system to understand them. Considering the structure of the grammar and the values of attributes for the activity of *takeMedicine*, the productions will have the following behavior:

p5: $date\_begin \rightarrow dt001, 2019 - 01 - 21$

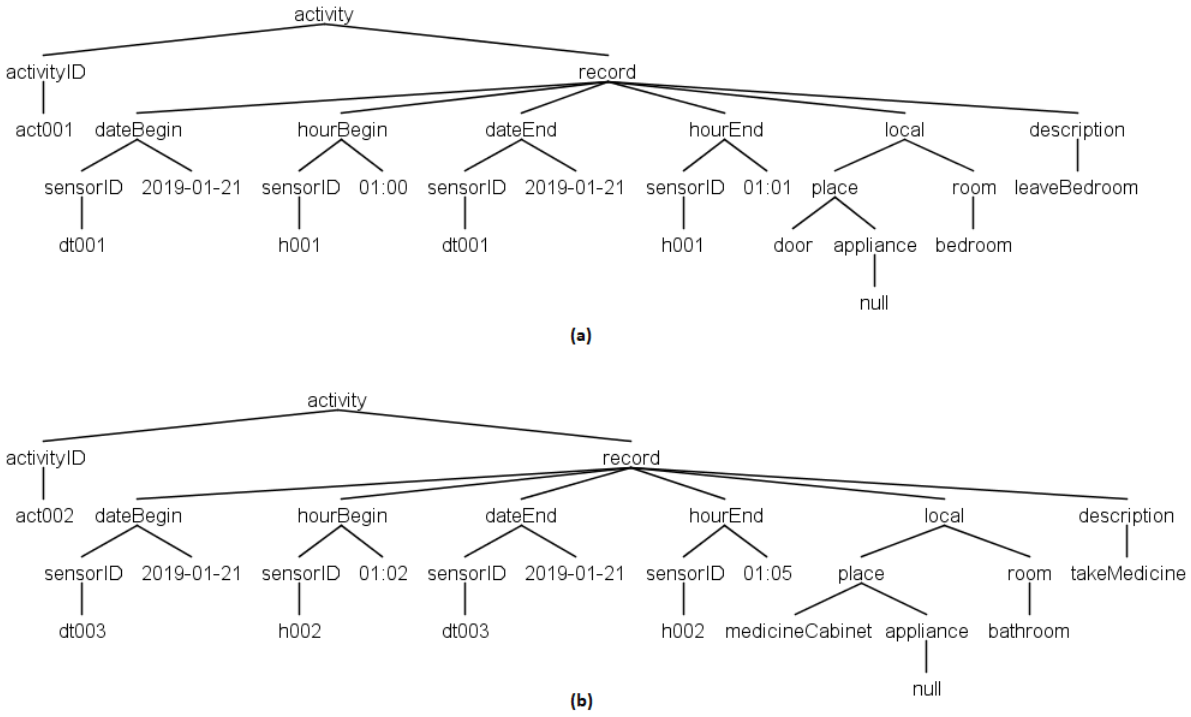


Figure 5.6 Abstract Syntax Tree: (a) leave the bedroom. (b) take the medicine.

p6: $hour\_begin \rightarrow h001, 1:00$

p6: $date\_end \rightarrow dt001, 2019-01-21$

p7: $hour\_end \rightarrow h001, 1:01$

p7: $local \rightarrow door, bedroom$

p7: $description \rightarrow leave\_bedroom$

p5: $date\_begin \rightarrow dt003, 2019-01-21$

p6: $hour\_begin \rightarrow h002, 1:02$

p6: $date\_end \rightarrow dt003, 2019-01-21$

p7: $hour\_end \rightarrow h002, 1:05$

p7: $local \rightarrow medicine\_cabinet, bathroom$

p7: $description \rightarrow take\_medicine$

The system should have a module dedicated to the identification and analysis of patterns of behavior to facilitate context interpretation. By understanding the steps that the user usually performs to accomplish an activity, it is possible to identify the data that should be considered to model new activities. For instance, based on past occurrences, the system identifies that occasionally when the user wakes up during the night, she goes to the bathroom, opens the medicine cabinet, and takes medicine. Thus, it is possible to infer that all these steps are part of the same activity. The AG will use this data as-synthesized and inherited attributes for the symbols to validate the structure of the activity.

The validation is done through the analysis of each of the steps that composed them, considering proper parameters. Besides that, the system identifies the conclusion of the activity by analyzing the behavior of the user. It considers the time used to finish the actions, the record of the activity, and any

other relevant information. If the values of these attributes are within a predefined acceptable range, the system can infer that the activity was concluded with success.

The mapping and monitoring of activities are essential because any changes identified during the execution of an activity may be related to problems with the user, for example, if considering an Ambient Assisted Living domain, the unexpected changing of behavior can characterize health problems.

5.2.3.1 Semantic Rules

AGs allow the definition of inference rules that can be used to manipulate context data properly. The execution of these rules increases the system's knowledge once the amount of available data for reasoning will be more significant. The following list presents some examples of semantic rules developed for the validation/ identification of daily human activities to be used by context-aware systems.

1. record.duration = getActivityDuration(activity.activityID, record.date_begin, record.date_end, record.hour_begin, record.hour_end);
2. activity.activityID = getActivityID(TEXT.value);
3. record.date_begin = getDate(DATE.value);
4. record.date_end = getDate(DATE.value);
5. record.hour_begin = getDate(HOUR.value);
6. record.hour_end = getDate(HOUR.value);
7. activity.identification = activityDescovery(activity.activityID, place.sensor_ID[], place.appliance[], local.room);
8. place.sensor_ID = getSensor(TEXT.value);
9. place.appliance = getAppliance(TEXT.value);
10. local.room = getRoom(TEXT.value);

Rule 1 calculates the time spent by the user to perform an activity. To accomplish that, it analyses the beginning and ending dates and hours. If the ending date and time are not available, it can use the current system's timestamps to finish the processing. To perform this calculation it is necessary to use the results from the execution of rules 2, 3, 4, 5 and 6. If the total is out of a pre-defined range of an acceptable amount of time, it may represent that the user is having problems to accomplish the task. Considering the case study of taking medicine during the night, the structure of the semantic rules would be the following:

1. record.duration = getActivityDuration(activity.activityID, record.date_begin, record.date_end, record.hour_begin, record.hour_end);

2. activity.activityID = act001;
3. record.date_begin = 2019-01-21;
4. record.date_end = 2019-01-21;
5. record.hour_begin = 01:00;
6. record.hour_end = 01:01;

1. record.duration = getActivityDuration(activity.activityID, record.date_begin, record.date_end, record.hour_begin, record.hour_end);
2. activity.activityID = act002;
3. record.date_begin = 2019-01-21;
4. record.date_end = 2019-01-21;
5. record.hour_begin = 01:02;
6. record.hour_end = 01:05;

It is possible to evidence that to calculate the time of the activity of taking medicine is necessary to execute rules 2 to 6 twice, changing only the parameters. This occurs because the activity is composed of two sets of actions. One is referring to waking up and going to the bathroom, and others referring to the process of opening the cabinet and taking medicine.

Rule 7 allows the system to identify the activity that the user is performing. The function *activityDescovery()* receives a set of identification codes referring to the sensors used during the activity, the set of appliances that the user used to perform it, and the local. It uses the resulting values of rules 8, 9 and 10 as parameters to perform this processing. The importance of 7 relays on the fact that the execution of a specific activity in a different hour or day than the usual may represent an unpredictable change of behavior, and the system should be aware of such change. In the case study, these rules would have the following structure.

2. activity.activityID = act001;
7. activity.identification = activityDescovery(activity.activityID, place.sensor_ID[], place.appliance[], local.room);
8. place.sensor_ID = door;
9. place.appliance = null;
10. local.room = bedroom;

2. activity.activityID = act002;

7. activity.identification = activityDescovery(activity.activityID, place.sensor_ID[], place.appliance[], local.room);

8. place.sensor_ID = medicineCabinet;

9. place.appliance = null;

10. local.room = bathroom;

In this example, the set of rules (2, 7, 8, 9 and 10) is also executed twice, once for each action that composes the activity (leaving the bedroom and taking the medicine). Rule 9 receives a *null* value because the user did not use any appliance to accomplish the activity. In a different scenario where the user watches TV or uses the stove to cook a meal, for instance, the rule would assume valid values.

The rules are used as the basis for the analysis of the level of accuracy of specific contexts. This means that they can contribute to the minimization of the problem of uncertainty. For instance, the system should be aware of the time spent by the user to finish the activities. Following, algorithm 5.3 describes how semantic rules could be used for context analysis.

Algorithm 5.3 Example of a semantic rule for context analysis

```
1: procedure ActivityAnalysis(expectedDurationValues[])
2:     if getActivityDuration(parameters) not in expectedDurationValues[] then
3:         analyse(record.duration)
4:     end if
5: end procedure
```

The result produced by rule 1 should be analyzed against an established set of acceptable values, referring to the time. If the result is not satisfactory, it means that there are problems related to the activity. For instance, the value of *record.hour_begin* was not captured with precision, resulting in an uncertain time spent, or it may be related to problems with the user once he is not able to finish the task as usual. To solve this problem, all the processing of the rule *record.duration* should be executed again.

Despite the low complexity applied in this set of semantic rules, it is possible to state that, depending on the application, attribute grammars create the possibility for the definition of much more complex rules.

5.3 Summary

This Chapter discussed the topic of Formal Grammars. The main goal was to relate this consolidated field of research with intelligent environments.

The main objective of using formal grammars was reviewed, and its general structure was identified. It allows the explicit definition of symbols with specific shapes and restrictions that they should obey. This is achieved through the creation of language with a strict and formal notation.

The first topic covered referred to *Context-Free Grammars*. The process of creating metalanguages through this approach was depicted by detailing the phases of creation. The importance of a rich definition of symbols to be used to represent sentences of languages was highlighted. The structure of the restrictions was deeply analysed, emphasized the difference between non-terminals to terminals and the impossibility of creating validations to case bases for recursion rules.

The generic structure of a production rule was presented, aiming to explain how they can be created for a specific language. The goals of CFG were addressed and, consequently, some limitations of it. The fact that it focuses only on syntactic validation of a language was discussed. Semantic issues can not be covered. The concept of the Derivation Tree was described presenting steps of its creation.

The second topic covered by the Chapter referred to *Attribute Grammars*. Its debate was more profound once it was used during the development of the framework that will be presented in the next Chapter. AGs are considered extensions of CFGs that provide different aspects to the validation of languages that could not be initially covered. The difference between synthesized and inherited attributes was enlightened and generic structures of them were discussed (algorithms and derivative trees).

The formal structure of AGs was reviewed with the objective of emphasizing how its tuple could facilitate the understanding of its construction. Also, the phases of processing sentences through AGs were evidenced.

As aforementioned, AGs is a consolidated field of investigation with solid contributions in the validation of formal languages. This Chapter brought a new approach describing how they can be applied to intelligent environments. The main contribution here is to show how two fields of research apparently without similarities can contribute to each other. At last, an AG to validate context data was presented. The goal was to analyse the possibility of using AG to ensure the structure of Activity Daily Living, i.e., ordinary tasks performed by a user in a smart house.

A scenario representing activities performed by a user in a smart house was described with the aim of illustrating how an AG can be applied to such a domain. The production rules were described and explained as well as the generation of the Abstract Syntax Tree of two specific activities (leaving the bedroom and taking medicine).

A group of semantic rules was created aiming to increase the knowledge about the context. A generic structure of a semantic rule for context analysis was presented and validated. The main goal of this characterization was to discuss the applicability of such rules to intelligent environments once there was no evidence of the relation between these two fields of investigation.

The next Chapter presents the approach to deal with uncertain situations faced by context-aware systems. It uses the subjects discussed so far to create a framework to tackle the problem.

Chapter 6

An Approach to Cope with Uncertainty

A framework to be applied in context-aware systems is described in this Chapter. Its four main modules, that perform specific sorts of data validation to prevent uncertain situations, are explored. Their architectures allow context data to be analysed in a fluid and logical sequence. A knowledge base is presented and is used to store data from past situations. Context is formalized, uncertain situations are identified, and a decision tree is used to cope with this problem. A new dataset is introduced to be used as a running example.

Previous Chapters described different approaches to validate the context data. They can be applied separately according to the goals of the system. Also, they can complement themselves, being used in different steps of a more comprehensive analysis, aiming to improve the accuracy of the environment's interpretation. Figure 6.1 presents an approach that encompasses the refinement of context data through attribute grammar, QoC analysis, and uncertainty identification. The main objective is to ensure that the system uses only data that successfully passed through all the modules before decision-making.

Six modules compose this framework, including the input, output, and knowledge base. The validation of the data provided by the sensors is processed sequentially in three modules. First, its structure is verified through a parser developed based on the AG. This parser aims to verify if the sequences of data are structured according to the grammar restrictions. After that, the quality of context is analysed, considering the parameters discussed in section 3.1. Before generating an output, the last verification refers to uncertainty identification, where the system searches for imperfect data that may compromise the correct functioning of the system and impact the decision-making.

The following sections describe the modules responsible for validating the data provided by sensors.

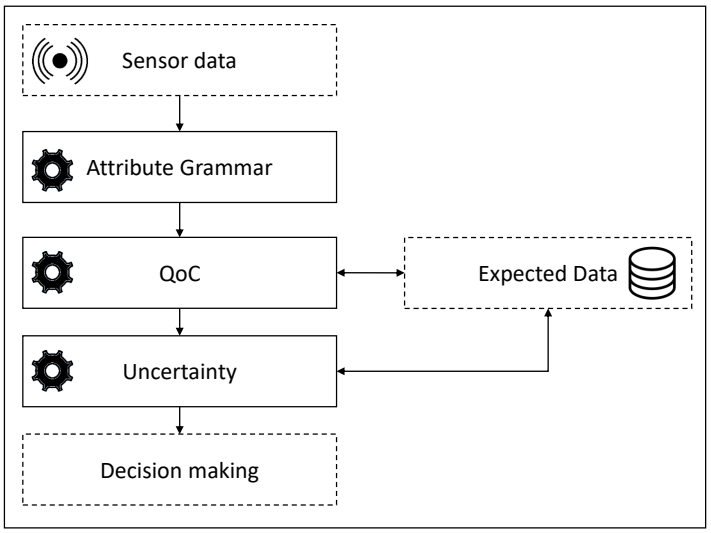


Figure 6.1 Context data refinement for decision making.

6.1 Attribute Grammar Module

This module is responsible for receiving data gathered from the environment and performs the first verification. It concerns to the validation of the data sequence structure (syntax) and static semantics. The collected data compared against the parser's production rules, where they will be analysed.

The grammar's symbols were defined following a strict and formal structure. Thus, context data should obey the restrictions that compose the axiom. After receiving the data set, an AST is generated to analyse the input values. By completing this process, the system is capable of verifying if the set of data is coherently based on the pre-defined conditions. This process is achieved by evaluating synthesized and inherited attributes to compute the values within the nodes of the AST.

The most important task in this module is the execution of inference rules. The definition of which ones will be executed depends on the type of context data. The main goal of this phase is to generate knowledge to complement the data acquired from the environment. For instance, calculating an activity's duration or determining the user's exact location based on raw data refers to coordinates. Algorithm 6.1 presents an example of the inference rule, which aims to identify an activity.

Algorithm 6.1 Inference rule to discover the activity based on timestamp and location

```
1: procedure ActivityDiscovery(TimestampBegin, TimestampEnd, room, appliances[])
2:     duration ← getActivityDuration(TimestampBegin, TimestampEnd)
3:     location ← findUserLocation(room, appliances[])
4:     Analyse possible activities based on duration and location
5: end procedure
```

The rule could be executed after getting the time-stamps for the task starting and ending events and the appliances[1] used by the user are detected. The variable *duration*, in line 2, receives data that characterizes the time spent to complete the task. The execution of the instruction in line 3 allows the system to identify

[1] Home devices used to execute daily tasks. For instance, microwave, oven, and refrigerator.

the exact location of the user, based on the room where he was spotted and the appliances that he had been using. Line 4 represent a sequence of instructions that are executed to discover the activity performed based on the data received by the rule and the inference from the above instructions.

The importance of this type of rule relies on the fact that if the system can generate more knowledge based on the AG's values, this could be used to minimize problems in the other modules (QoC and Uncertainty). Besides that, it allows the system to verify if the sensors are sending adequate data and if they follow patterns of behavior. Any significant changes related to this may configure problems with the system or with the user, and the responsible entities should be notified. If any of these scenarios is confirmed, the analysis of the context data is interrupted, and the other two modules are not executed until the problem is solved. Figure 6.2 presents the structure of this module.

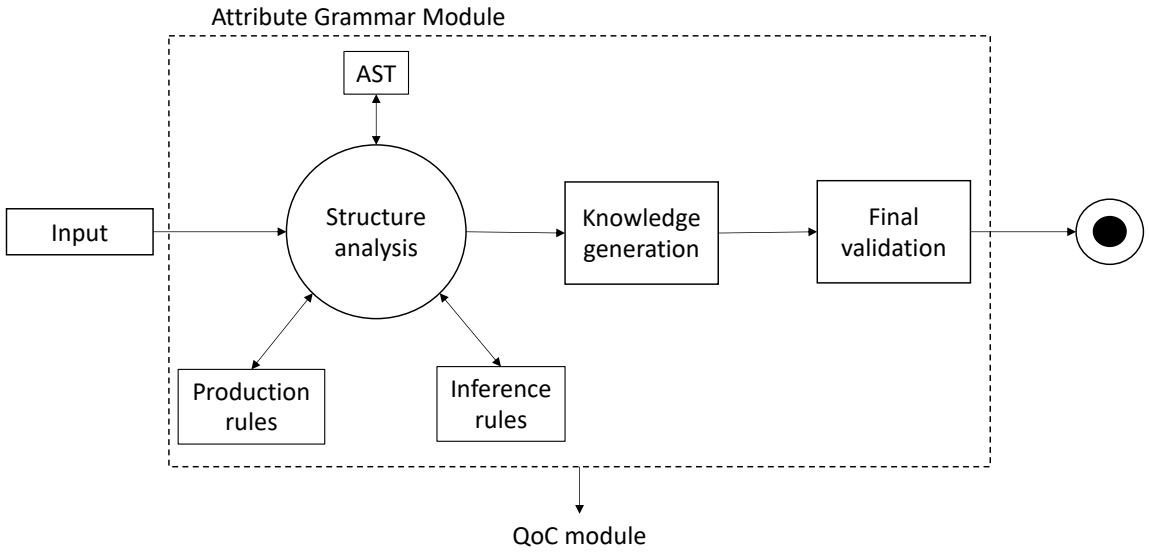


Figure 6.2 Attribute Grammar module.

The *Structure analysis* module centralizes all the phases of the validation. It receives lists of raw data captured by sensors and prepares them. In this process, the module verifies the context of the environment and uses only data that can contribute for the adaptation of services related to that context. For instance, the *input* data may contain house appliances if they were detected because they are were powered on (*STATE: ON*) and yet they not always contribute to the performance of a specific activity. Thus, this data will not be considered by the *Structure analysis* module.

After being prepared, the data passes through the analysis of the *Production rules* module. The main goal of this phase is to ensure that the resulting subset obeys the restriction imposed by the attribute grammar's axiom. This fact will avoid future inconsistencies between the input data and the requirements of software agents to execute their services. If issues are found during this process, the data should pass through a second preparation. Thus, more data may be included, deleted, or restructured in the context subset before being submitted to the *Production rules* module again. The generation of the *abstract syntax tree* results from this validation process. It will be used to verify lexical and syntactical elements of the context subset through the hierarchical structure created in the tree.

The *Inference rules* module is applied to the context subset after a successful validation in all the previous steps. The data is then processed, aiming to increase the knowledge of the system about the current state of the environment. In cases where the result of rules are able to derive new information, this is incorporated to the context subset in the *Knowledge generation* module. If the analysis of the attribute grammar is completed successfully, one *Final validation* is performed in order to verify the relevance of the new information and if it can be used allied with the existing data. If this process is completed successfully, the analysis of the context subset then continues to the second module of the framework, where the *Quality of Context* is verified. Otherwise, it is interrupted, and this context subset is not used in the analysis of the environment.

6.2 Quality of Context Module

After ensuring the structure of the data that composes a situation, it is crucial to analyze its level of quality. Sensors that do not work correctly may provide imperfect data, and its processing generates new data that is also imperfect. Thus, it is necessary to verify if the system is using relevant facts to orchestrate services.

It is essential to differentiate the verification provided by AGs and the one performed in this module. A bathroom scale that sends the random user's weight to the system can be used as an example to illustrate the difference between them. The data is well-structured but considering past measurements, it indicates a defective value.

There are several parameters that can be taken into consideration to ensure the quality of context data. New parameters can be incorporated, and/ or existing ones might be ignored, depending on the situation and the system's goals.

Considering the domain of Activity Daily Living and smart houses, and the discussion of 3.1, the parameters *accuracy, usefulness, up-to-dateness, types of acquisition and representation* and *importance* are considered as being essential to ensure high quality of the context data. Each of them is compared to a knowledge base. It stores previously validated data from past situations. It also keeps collections of acceptable values for each type of data. If any of the parameters is not successfully validated, the system should take a group of actions to avoid using imperfect data. Requisition of new data from sensors, starting an interaction with the user, sending notifications to service providers, or just ignoring a specific piece of data can be some of the precautions assumed by the system.

The process of validating the parameters does not ensure that the data has good quality. First, each parameter is verified individually. Then, they must be analyzed together, and common elements in their structure must be identified. Only data with all these characteristics are considered to have high QoC.

The analysis of QoC of a piece of data can be complicated, depending on the number of software agents and their goals. An outdated data may be irrelevant for one agent but very accurate for another. These agents may have different functions, making some of them more sensitive to context changes than others. Thus, before discarding a piece of data, the system must verify, in this module, its usefulness to all software agents. Figure 6.3 presents the structure of this module.

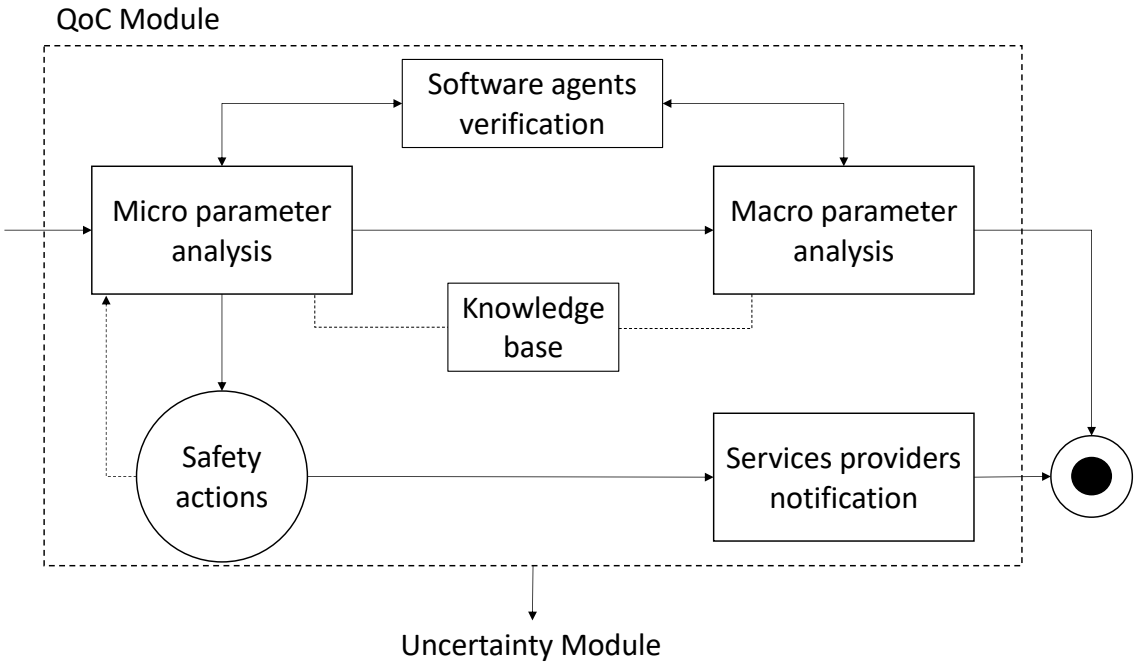


Figure 6.3 Quality of Context module.

When the context subset successfully passes through the parser in *Attribute Grammar Module* verification, the quality of its content is checked. This analysis is performed under two visions of the dataset. In the *Micro parameter analysis* each value of the dataset is compared to others of the same type. If they are consistent, it indicates that it has quality, considering the parameters mentioned above. The comparison is performed according to past situations in which the data is stored in the *knowledge base*. The *Knowledge base* receives requisitions from this module and returns pieces of data from past environmental situations. The results of queries are compared to the variables of the context subset.

The *Micro parameter analysis* module analyses the values individually. It verifies if the data collected by the sensor already appeared in other situations in the past. This comparison is essential once the prediction model applied to identify context situations is based on a training dataset that contains activities successfully identified and classified. When a piece of data does not match with values of any corresponding range in the *knowledge base*, its quality is not verified.

In cases where a piece of data is not validated, this module must take *Safety actions* to avoid it jeopardizing the context analysis. The actions depend on the piece of data and the context it belongs to. For instance, problems related to the infrastructure (e. g. hardware and communication) or anything that may compromise the health and safety of the user, the system can notify the *Service providers* and interrupt the context analysis. However, in other scenarios, where the well-being of the user is not at stake, the system could request more data from a specific sensor or start an interaction for him to provide accurate data. In these cases, the *Micro parameter analysis* is executed again.

After successfully passing by the microanalysis, the verification of the QoC follows to the *Macro parameter analysis*. In this phase, the context of the situation is built. The coherence of its structure is validated through the executions of queries to the *Knowledge base*.

Considering the diversity of objectives and services to execute, one agent uses specific pieces of context data, which may be considered irrelevant to all the others. Thus, in both phases of parameter analysis (micro and macro), before discard one piece of data, the system should verify its relevance to all software agents. If it uses helpful in any of them, the discarding process is canceled.

Context data is considered to have high quality if it passes through all these verification steps. Then, it is sent to the *Uncertainty module*, where the last verification is performed.

6.3 Uncertainty Module

The main goal of this module is to identify problems related to context data that were not previously addressed. From this identification, the system can minimize negative impacts on the user by assuming safety measures to ensure its most appropriate behavior.

The first two modules are responsible for different types of validation of the context data and complement each other. Despite the data having a well-structured (AG verification) and high quality (QoC verification), sometimes it might not be as complete as it could. Thus, this module performs additional verification.

Due to the natural dynamics of ADL environments, context data may become obsolete rapidly, as described in section 3.1. Besides that, the random behavior of sensors makes them send wrong measurements to the system, and this data can not be used for decision making.

Considering these issues, this module should maintain communication with the knowledge base to verify if the current context data is as complete and valuable as it should. Among other information, the knowledge base stores data from situations (activities) that were evidenced in the environment in the past. The system performs queries to verify if the context data is reliable. This process prevents the use of random data during the orchestration of services.

Besides that, if the system identifies that it needs specific context data that was not collected by the sensors to build a context situation, it can use data from the registers of past situations. Thus, it allows the system to complete the process of decision-making. However, users should be notified about the use of old data.

6.3.1 Formalization of Context and Uncertainty Analysis

This section aims to present a formal representation of context situations and, from this, describe how uncertain aspects of it can be identified for further analysis.

Intelligent environments analyse the current states of entities, identifying the relations among them to create a computational model representing real-world contexts. These real-world contexts will be called situations.

Definition of situation: *Refer to specific settings of the environment, in specific moments of time and involving specific sets of entities.*

The *setting of the environment* include anything that characterizes it and that a computational system could use it to adapt services (e.g., location of the user). The *moments in time* is used to identify and determine the starting and ending hours of tasks. They can help the system to identify situations and possible overlapping of sequential tasks. The *sets of entities* include anything available in the environment that is used as a source of data for other context building. This encompasses, for instance, the user, the room where he is located, and available house appliances.

Considering that, one situation is composed of a set of context values received from sensors, it can be formally represented by the following tuple:

$$S_i = \langle v_1, v_2, ..., v_k, ..., v_n \rangle$$

Where:

- i refers to a specific situation of context;
- n is the total number of context data used to compose the situation S_i;
- $1 < k < n$.

The v values represent context data provided by physical sensors or the result of any processing that involved data from them. In this case, the context-aware system itself is also seen as a source of information. The situation S_i is characterized by the set of values (context data). It may have as many as necessary. The more significant is the amount of context data in its tuple; the more accurate will be the representation of the environment and, consequently, the more accurate will be the actions that can be performed. The absence of one or more of them may compromise the resulting representation of the real-world context. For instance, if one of the missing values, v_k, refers to the user's current location in an intelligent house, it might not be possible to recognize the activity he performs if this data is imperative for this inference.

One problem that may emerge regarding the data of the situation is that, for each of them, a collection of acceptable values must be defined. They are considered acceptable once used by the system in previous analysis to validate situations. In other words, they are considered valid values because they were already used in similar situations. Hence, they are stored in the knowledge base of the framework. Additional approaches can be applied for the election of the most suitable attributes to fill the gaps of data. For instance, prediction algorithms could use the existing data from the base to provide possible values with a satisfactory confidence level. Regardless, the approach must consider the types of data and service goals. This process allows the system to ensure that only valid states are considered for the identified situation. For instance, considering that the human body temperature can vary from 35° Celsius to 41° (in case of fever). If v_k is equal to 31° Celsius and refers to the user's body temperature in an Ambient Assisted Living, the system should identify that this is not a valid number and, at least, not consider it for the analysis of the situation.

The formal specification of situations allows context-aware systems to take action. These actions are functions of the situation S_i and affect the current state of entities, such as users and devices, as well as the environment itself. Actions produce results and can be represented as follows:

$$action_1(S_1) \longrightarrow result$$
$$action_2(S_1) \longrightarrow result$$
$$...$$
$$action_n(S_1) \longrightarrow result$$

In context-aware systems, actions refer to decision-making based on environmental data. They include execution of applications or services, emission of alerts or warnings, the adaptation of appliances, and others. The result of an action represents the consequence of its execution, i.e., how the action affects the environment. One situation S can be used as the basis for the execution of one or more actions. Besides that, it is possible to create a chain of actions considering the results obtained in previous. They can be represented as follows:

$$action_1(S_1) \longrightarrow result_1$$
$$action_2(S_1|result_1) \longrightarrow result_2$$
$$action_3(S_1|result_1|result_2) \longrightarrow result_3$$
$$...$$

In this case, previous results could be considered to execute current action if they are considered relevant. I.e., if they have essential information about the environment and its context.

The context values that compose situations are validated in all the modules of the framework. Following, algorithm 6.2 describes how the verification is performed in the *Uncertainty module* and its interactions with the knowledge base in order to prepare the situation to be used for the system's adaptation and decision making.

The procedure receives as parameters the set of context data that composes the situation S and an array with the expected values for that situation. This array of expected values results from queries executed over the knowledge base, using as inputs the data collected from sensors. For each context data, a series of acceptable values is defined. An auxiliary array (*occurances[]*) is created with *False* values, in lines 3 and 4. After the verification, their states are changed to *True* if the corresponding sensor data has a value that is within the pre-defined range. In lines 5 and 6, the elements of $S[]$ are compared to the set of expected values for that specific situation. If this comparison results in a match, the state of that element, in $occurances[]$, is changed to *True* (line 7). This routine will be finished after executing these steps for all sensor data.

The existence of invalid context data is verified in line 10. If the return of the function *CheckExistanceFalseValues()* is *True*, it means that the situation S has null values or out of the pre-defined set of acceptable range (*False* state). In this case, a set of actions must be executed. First, the system starts

Algorithm 6.2 Analysis of the set of values of situation S

```
1:  procedure SituationDataAnalysis(S[], expectedValues[])
2:      j ← 0
3:      for i ← 0 to length(S[]) do
4:          occurances[i] ← False
5:          while j < length(expectedValues[]) OR occurances[i] is False do
6:              if S[i] == expectedValues[j] then
7:                  occurances[i] ← True
8:              end if
9:              j ← j + 1
10:         end while
11:         j ← 0
12:     end for
13:     if CheckExistanceFalseValues(occurances[]) then
14:         runSensorDiagnosis(occurances[], S[])
15:         NotS[] ← getFalseValues(occurances[], S[])
16:         ValidS[] ← EliminateInvalid(S[], NotS[])
17:         r ← runActions(ValidS[])
18:         if !r then
19:             messageAlert()
20:             interaction(S[], occurences[])
21:         end if
22:     end if
23: end procedure
```

diagnosis tests for the sensor(s) responsible(s) for sending the invalid context data (line 11). This function analyses hardware failures, checking if the sensor is online or not, and communication problems, starting requisitions, and verifying the data received. If problems are detected, the system sends message alerts to the responsible entities. After that, in line 12, the system removes the invalid data from the composition of S. The following tuples formalize this situation after the execution of this line code.

$$S_i = \langle v_1, v_2, ..., v_n \rangle$$
$$NotS_i = \langle v_1, v_2, ..., v_m \rangle$$

Where:

- $NotS_i$ contains only invalid data for the situation S_i;
- $NotS_i \notin S_i$;
- $m \leq n$.

A new array with only valid context data is created in line 13. This process is executed because, even with missing values, actions could be performed for the situation S (line 14). If not possible, an appropriate message is sent to the user (lines 15 and 16). The function *messageAlert* identifies the invalid values and,

depending on the type and amount, takes different decisions, such as alert the user about the missing data or request new information to fill the gaps.

The instructions of lines 14, 15, and 16 refer to actions from the *decision making* module. Based on the validation of context data from the three *verification* modules, the system analyzes which actions could be triggered to adapt to the environment. Message alerts and interactions with the user may be necessary to improve the orchestration of services.

Intelligent environments are related to the paradigm of ubiquitous computing where systems work autonomously, with minimal explicit interaction with users (Weiser, 1993). However, according to Lim and Dey (2011), the presentation of uncertain situations for them to solve helps the system to improve its knowledge. Consequently, the assistance provided by it will be more accurate and personalized. Considering this, in line 17, the specific requisition of context data through interaction with the user is performed, aiming to improve the QoC and minimize the impact that incomplete context data can have on decision-making.

6.3.2 Decision Tree to Deal with Uncertainty

Decision trees allow the separation of sets of data according to labels. These labels refer to the classification of the examples of the dataset. Even in cases where it is not possible to classify all of them, DTs provide the probability of certainty that a group of data belongs to a label (class).

Considering this, DTs can be used to deal with uncertainty problems. In the case of a situation with a lack of context data, the classification achieved from the training set could be used to analyse possible values to fill the gaps.

A dataset developed by Ordónez et al. (2013) was used to facilitate understanding the application of DT in uncertainty analysis. It presents data related to the monitoring of Activities Daily Living (ADL) of two users in their respective smart houses. Considering that, one of the goals is to identify patterns of behaviours and uncertainties related to it. For the development of this work, it was decided to use data from one user. The data was collected over 21 days through 5 different types of sensors installed in the five rooms of the house. The dataset contains ten different types of activities performed by the user in his daily routine. The dataset was properly normalized and prepared. This process included the addition of a new column to label the activities. Originally they were described in groups. As aforementioned, in intelligent environments centred in humans, one *situation* refers to the contexts that surround them. Thus, considering the structure of this dataset, each example of it can be seen as a real-world situation within the user's daily routine.

The dataset contains information related to the characteristics of the sensors used to detect activities. Besides the activity's timestamp, it stores the location of the sensors, the room of the house where they were installed, the type of data they provide, and the name of the activity the user performed. The features accept the following values:

- *Place*: Entrance,Bathroom,Bedroom,Kitchen,Living;

- *Type*: Magnetic, PIR, Flush, Pressure, Electric;
- *Location*: Maindoor, Fridge, Cupboard, Shower, Basin, Door, Toilet, Seat, Bed, Microwave;
- *Activity* (Label): Leaving, Toileting, Showering, Sleeping, Breakfast, Lunch, Dinner, Snack, Spare_Time/TV, Grooming.

No further information from the experiments developed by Ordónez et al. (2013) was used.

For each example, the label (*Activity*) was assigned based on the analysis of the data provided by the other features. The detection of a new situation in the environment means that the sensors provided a new set of values. The tuple presented in 6.3.1 applied to this dataset will have the following structure:

$$S_i = \langle Timestamp, Place, Type, Location \rangle$$

The importance of applying DTs to this dataset to find a reliable classification relies on the fact that the variety of possibilities is considerably wide. For instance, to find the most appropriate question to be done to separate the data, it is necessary to consider five possible values for *Place*, five other for *Type* and ten different for *Location*, getting to a total of 250 possibilities. If the timestamps are considered, this number grows even more. Table 6.1 shows the structure of the dataset with four samples randomly chosen.

N°	Begin	End	Location	Type	Place	Activity
1	2012-11-12 01:53:19	2012-11-12 01:53:22	Door	PIR	Living	Grooming
2	2012-11-12 01:53:27	2012-11-12 01:53:35	Basin	PIR	Bathroom	Showering
3	2012-11-12 01:53:39	2012-11-12 01:54:27	Toilet	Flush	Bathroom	Toileting
4	2012-11-12 01:53:53	2012-11-12 01:53:57	Door	PIR	Bedroom	Sleeping
5	2012-11-12 01:55:09	2012-11-12 09:30:35	Bed	Pressure	Bedroom	Sleeping

Table 6.1: Sample of dataset situations.

The dataset contains a total of 2334 registers, and it was divided into two subsets. One containing 1634 examples (70%) was used to train the classification model, and the second containing 700 (30%) to test it. The testing set simulates situations in which there are missing values, i.e., the examples do not have complete data. These gaps of data are filled with default values based on the classification provided by the DT.

Thus, the classification of the training set provided by the DT could minimize the uncertainties presented here. The attributes with known values should be used to navigate the tree nodes where the answers for the corresponding questions were true. Figure 6.4 presents one sub-tree to describe this scenario.

The timestamps of starting and finishing moment of the activity's execution were omitted in this example, aiming to visualize the tree better. However, in the actual analysis of the training set, they are taken into consideration due to their importance in differentiating similar rows of the dataset.

The questions used as examples in this image were chosen based on the scores obtained in GINI impurity and Information Gain. This example aims to discuss how the decision tree can be used to classify

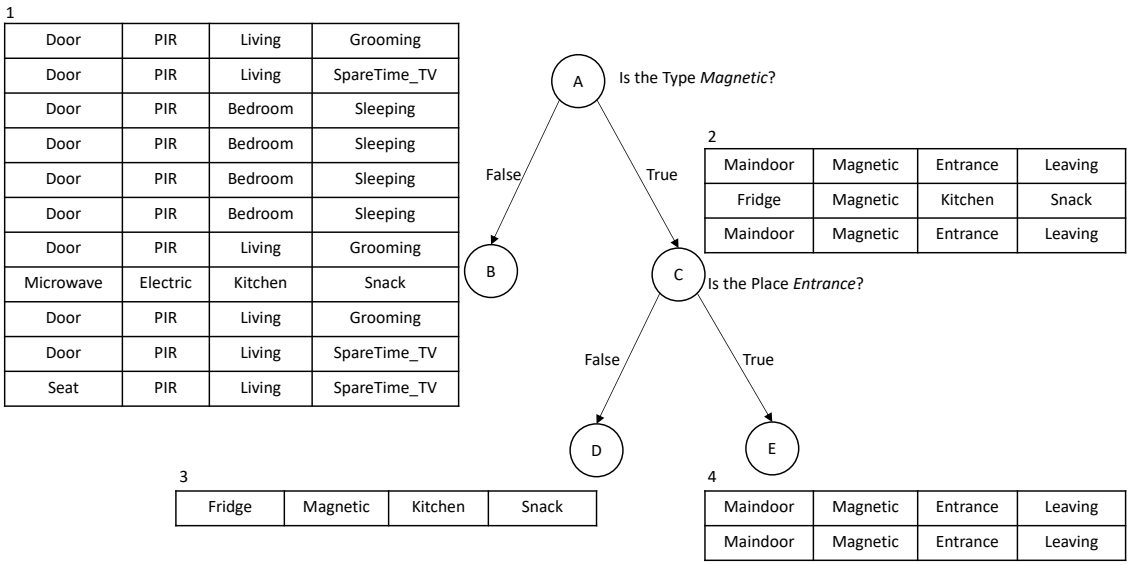


Figure 6.4 Example of Decision Tree.

the activities by analysing the values from the columns of each row. The original subset of this example refers to the union of tables 1 and 2, i.e., the table of the root node is the union of those from nodes B and C. After two rounds of questions, it was possible to classify one of the activities (*Leaving*), in table 4 (node E). The classification will not always be one hundred percent pure. Nevertheless, the distribution provided by the tree can still be used to fill gaps of data when the system cannot gather it from the sensors correctly, once it presents an acceptable level of confidence.

Table 6.2 presents a scenario with four examples of real-time situations with missing context data.

N°	Begin	End	Location	Type	Place	Activity
1	2012-11-12 16:22:22	2012-11-12 16:22:29	Maindoor	Magnetic	null	?
2	2012-11-12 01:53:27	2012-11-12 01:53:35	null	PIR	null	?
3	2012-11-12 09:47:57	2012-11-12 09:48:12	Fridge	Magnetic	null	?
4	2012-11-12 01:53:53	2012-11-12 01:53:57	null	PIR	null	?

Table 6.2: Real-time situations with missing data.

By analysing the table, it is impossible to determine the classification of the activities based only on the data provided. In case number 1, the combination *Location: Maindoor* and *Type: Magnetic* values could be enough to perform this classification, even though the *Place* attribute is missing. This inference is possible because, in this specific dataset, there is only one possible place where a sensor could be installed considering this *Location* and *Type*, which refers to *Place: Entrance*. The combination of these three values results in only one type of activity in this dataset: Leaving.

In case number 3, the same attribute is missing (*Place*). However, it is more complex to determine the activity the user is performing since there is a range of possible activities that result from the combination of *Location: Fridge* and *Type: Magnetic*. The *Place* and, consequently, the *Activity* classification could refer, for instance to *Kitchen* and *Breakfast, Lunch, Dinner* or *Snack*.

The activity classification of examples numbers 2 and 4 are more difficult to be determined because both of them have the same values for the attribute *Type* and two other attributes are missing. The complexity of classification relies on the fact that the value *PIR* when combining with, e.g., *Location: Door* represent a considerable amount of examples in the dataset. Besides that, considering that a significant number of examples in the training set has the combination of *Location: Door* and *Type: PIR*, the leaves of the tree would not present a classification with a high probability of certainty. Thus, its relevance for the definition of the activity decreases. However, the level of uncertainty would be lower when compared to the input since the number of missing values would decrease. Then, based on the possibilities provided by the DT, appropriate decision-making could be performed. For instance, request new data to the sensors, ask for more information from the user, analyse possible services that could be triggered based on what is known, and so on.

6.4 Summary

This Chapter presented the approach of this ph.D. work to cope with problems related to the uncertain situation in a context-aware system. A framework composed of six modules was introduced. The input, output, and database are represented, respectively, by the *Sensor data*, *Decision making* and the Knowledge base of *Expected data* modules. The verification of the environmental data is performed in sequential phases. The overall architecture of the framework is presented and explained. Aiming for a more profound comprehension, the architectures of three main modules are also described and explored.

The first analysis occurs in the *Attribute grammar* module. It performs a validation of the structure of the data, its syntax, and semantic. Also, it verifies if the data represents a coherent situation. Sets of inference rules are executed according to the type of data that was received. All these phases are centralized in the *Structure analysis* module. If any sort of issue is found during the process, it is interrupted, and proper actions are executed to solve the problems. Successful outputs minimize further concerns in the following steps.

The second verification is performed in the *(*QoC) module, where all the parameters discussed in Chapter 3 are taken into consideration. The validation occurs based on different perspectives. In the first one, then each piece of data is analysed individually, i.e., independently from the rest, in the *Micro parameter analysis* module. *Safety actions* are ready to be triggered if any inconsistency is found that could compromise the output. After that, the Macro parameter analysis uses the context values to compose a situation of context and verifies if it is cohesive. Communication with software agents is executed, aiming to verify the data's usefulness during all the validation processes.

The last verification aims to find uncertainties related to the context data. In order to cope with this type of problem, it was proposed a formalization of the context. Thus, it should be composed of finite values that represent pieces of information from a real-world situation. Based on this, it is possible for a system to create groups of rules. When executed, they represent actions that will have an impact on the situation. The *Uncertainty* module queries the *Knowledge base* during the validation process. When it identifies

issues from any sort group of measures are taken aiming to find solutions. If hardware failures are found, an adequate search is triggered trying to diagnoses the problem. If invalid data are encountered, they are eliminated from the subset that is being considered to build a situation. After this process, messages are sent to the user and/ or service providers, alerting them about the invalid data. If they are essential for context creation, the system explicitly asks for them for the user. All these functions were presented through an algorithm and were fully discussed.

At last, the possibility of using the distribution of data provided by a decision tree was described. A new running example was presented aiming to illustrate how the context-aware system could benefit from it. After identifying uncertain situations, the system fills gaps of missing data with default values that were considered valid in the past and classified correctly by the tree.

The next Chapter presents case studies based on the running examples aiming to validate this proposal.

Chapter 7

Case studies

The validation of the proposal is presented here. Several scenarios were created based on the two datasets that are being used along this document. Arguments that support validation via case studies are described in detail. The consequent achievement of the main objective of the research is evidenced and discussed.

This Chapter aims to present the proposal's validation by discussing case studies based on the two datasets used along with the research. The first dataset was referring to the running example scenario, as described in section 1.6 and the second referring to the dataset presented in Chapter 6.

7.1 Smart Apartment Activity Daily Living Monitoring Dataset

The following experiments were conducted based on the dataset used as a running example, described in section 1.6. The dataset was created by Cook and Schmitter-Edgecombe (2009) and contains five Activity Daily Living tasks: *phone call*, *wash hands*, *cook*, *eat*, and clean*clean*. These tasks were executed in sequence in a testbed environment. All the fifty volunteers were asked to perform the same sequence of tasks, starting with the *phone call* where they received instructions for the next steps. One file for each of the participants was created, containing data related to the tasks. The data was acquired from motion, item, and water and burn sensors. The resulting dataset contains six thousand and four hundred and ninety-seven registers. The file containing the data from all participants was chosen to be used to conduction of the experiments presented here. The dataset was properly prepared according to the goals of the research.

7.1.1 Discrepancy Among Occurrences of the Same Activity

The first scenario evidenced from the dataset's analysis refers to the time spent by the user to accomplish each of the tasks. For the development of the experiment, it was taken into consideration all executions of *phone call* activity. Figure 7.1 presents the results.

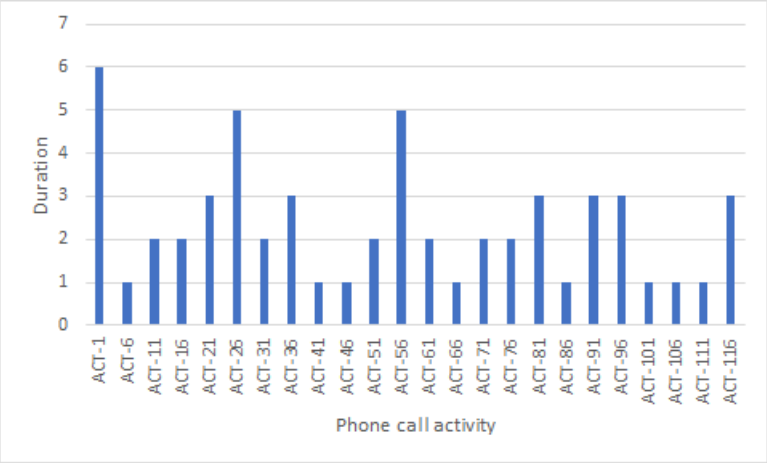


Figure 7.1: Time spent in each execution of activity *Phone call* (in minutes).

It was decided to use the *phone call* activity to represent the experiment due to its natural dynamicity in the sense that one execution of it may take much more (or less) time than the others. Even though all occurrences refer to the same type of phone call, i.e., in all of them, the user received the same information, it is possible to evidence a high discrepancy among the time to conclude it. It was necessary three minutes or less for the user to finish the activity in most of them, representing 87.5% of the occurrences. The other 12.5% represent only three occurrences. 8.33% for executions in 5 minutes and 4.17% in 6 minutes.

Considering that, it is possible to identify a pattern of behaviour of the user while answering this type of phone call. In other words, according to the results, he normally takes the same time to finish this type of call (3 minutes or less). However, in this scenario, the important analysis refers to the rest of the times. The following code (7.1) represents the algorithm from the attribute grammar to analyse this situation:

Algorithm 7.1 Analysis of activity's duration

```
1: procedure durationAnalysis()
2:     t ← getDuration(activityDuration)
3:     result ← true
4:     if t not in expectedValues[] then
5:         result ← false
6:         contextAnalysis(user, sensors)
7:     end if
8: return result
9: end procedure
```

First, the time spent by the user to finish this activity is attributed to the variable t (line 2). Then, the time spent by the user to finish the *phone call* activity is compared to a pre-defined range of expected

values (line 4). The range was defined based on the previous analysis, where it was evidenced that, in this case, the accepted time is 3 minutes or less. If this condition is *True*, a further *user*'s analysis must be performed considering parameters like his location and vital signs, including, for instance, heart rate, blood pressure, and level of oxygen. Also, a diagnosis of the correct functioning of *sensors* must be performed once this discrepancy may be related to hardware or communication problems (line 6). The result of the analysis is returned through the variable *result*. The state of this variable is *true* if the duration of the activity is within the expected time range or *false*, if otherwise.

In cases where the domain refers to an Ambient Assisted Living (AAL), where the user is a patient with a health condition that must be monitored, this variance may be related to a sudden change in his state. Thus, other types of sensors could be used. In another situation, this discrepancy may be related to communication flaws between the system and the sensors, preventing them from sending updated data to be analysed. This scenario could explain, for instance, the higher time used in the first occurrence (ACT-1), where the communication between these agents was re-established after some minutes.

Thus, the uncertainty present in this situation may have two origins. *Epistemic uncertainty* is evidenced in the case where the time spent represents unexpected changes in the user's behaviour (more time than usual to finish the task). If that is the case, the system must be able to understand it and suggest a solution once it can compromise the user's health. *Aleatoric uncertainty* is related to hardware flaws. The system must also identify this scenario once its result can affect its performance, assuming random behaviour. These types of uncertainty can be related to the definition provided by Chen et al. (2018), as aforementioned.

Assuming that the real world is naturally uncertain (Freitas et al., 2018), it is hard for humans to create long sequences of activities to perform during time intervals. Thus, creating computational representations for every possible scenario of an environment is a complicated task, and yet, it is necessary to find ways to deal with it. Uncertainty identification is the first step to face this situation. If a computational model is capable of identifying sudden changes related to user's behaviour, it can take safety precautions aiming to minimize the impact of this nebulous context.

7.1.2 Abnormal Amount of Activities per Day

In this experiment, it was analysed the total of activities performed per day. A sudden change of these values along the days may represent changes related to behaviour patterns or the user's health. Figure 7.2 presents the result.

The graph shows good consistency regarding the number of activities executed per day. The majority of the days, 84.61%, the user performed between five and ten activities. Despite having an increased number of activities on the last day of monitoring, it is reasonable to state that it does not affect his standard behaviour, once this represents an increase of five tasks (16,67%). It is essential to highlight that the definition of daily activity can vary according to the interest and goals of the system. Its deepness is directly related to its complexity. I.e., the broader is an activity, the complex it will be, and, consequently,

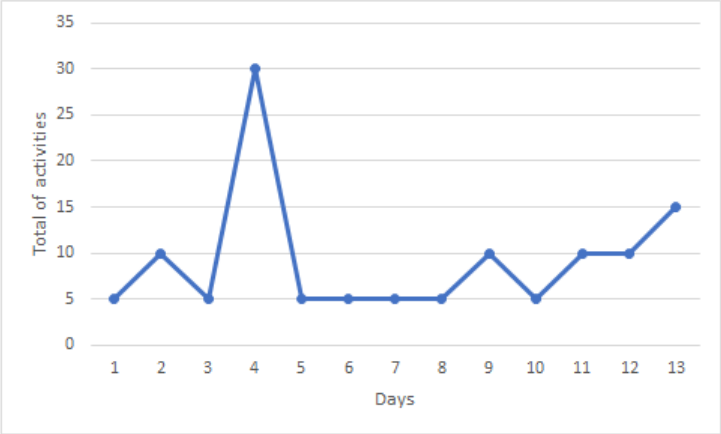


Figure 7.2: Total of activities per day

the deeper it is, the simpler it will be. Complex activities tend to represent sets of small tasks. Thus, if the user performs five activities during a day, each of them is probably composed of several minor activities.

Semantically, this graph shows that the user has a routine, probably using a constant time to perform his tasks. Considering a scenario of AAL, it is possible to infer that the patient is able to perform all his daily tasks, and his health condition is not being affected by that.

The uncertain situation is evidenced in *day 4* where the number of activities was 25, i.e., much higher than what was established as the standard amount in a regular day. There are several reasons for that. However, considering only the information from the dataset, it is impossible to state with a high level of confidence. For instance, in one scenario, it could refer to a weekend day where the user usually performs more tasks (e.g., receive visitors or clean the house). It also could mean the user's willingness to perform tasks, showing an improvement in his health condition. However, if this were the case, the following days would represent worsens of his health condition.

Regardless, as the number of activities on this particular day (4) has a high discrepancy compared to other days, the system must be capable of identifying it. This is the first step in order to minimize the impact of uncertainty in the domain. By the identification of its existence, it will be possible to circumvent the situation with security measures.

A series of security actions could be executed after identifying the source of the uncertainty. They were deeply discussed in section 6.3.1. After the proper handling, the set of data with inconsistencies may be validated by the framework modules, if necessary.

However, it is essential to emphasize that despite representing problems related to the characterization of context, it could also mean that the user performed a different number of activities than was expected. This scenario should also be taken into consideration by the system.

7.1.3 Measurement Problems in Activities with the Same Duration

The experiment was conduct based on the relation between the time spent by the user in each activity and the total of measurements of it. In order to show a clean result, only the *cook* activity was taken into

consideration. The same principle can be applied to all other activities. Figure 7.3 presents this scenario.

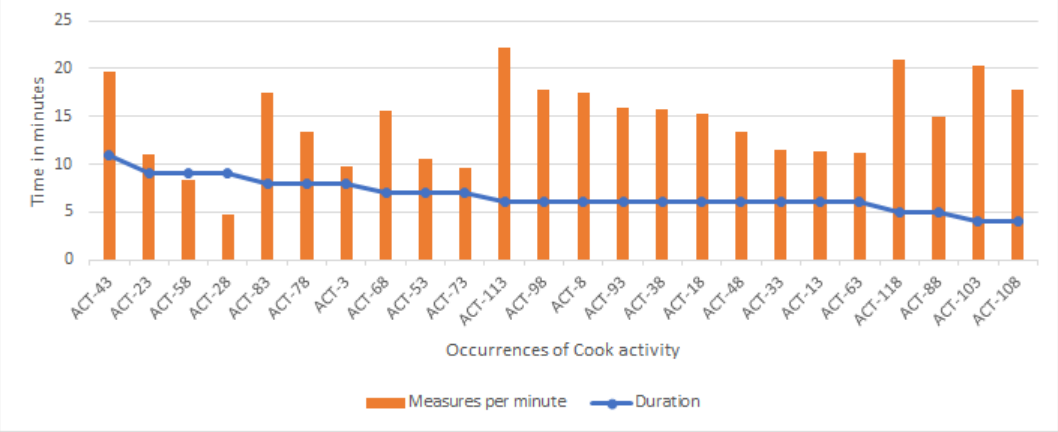


Figure 7.3: Total of measurements per minute and duration of *cook* activity occurrences.

The activities in the graph are organized from the longest duration on the left side (ACT-43) to the shortest duration in the correct size (ACT-108). The majority of the occurrences share a similar duration time (blue line), in which those that took six minutes can be highlighted (ACT-113 to ACT-63). Nevertheless, the total measurements per minute vary considerably among them (orange blocks). For instance, ACT-113 has 40.75% more measures per minute than ACT-63. This sort of discrepancy can be evidenced in other comparisons as well. For instance, ACT-83 and ACT-3 have a duration (8 minutes). However, the former has 44.28% more measures per minute than the latter. This fact means that the sensors were much more active in executing some of the activities than the others.

As aforementioned, the two main types of uncertainty are related to misinterpretation of sensor signs, including analysis of partial context data. They result in unreliable decision-making with incorrect interpretation of situations. Also, hardware failures of sensors and communication problems may influence this. Algorithm 7.2 describes how these types of uncertainty can be tackled.

Algorithm 7.2 Analysis of uncertainty

```
1:  procedure uncertaintyAnalysis()
2:      activityType ← getActivity()
3:      while durationAnalysis() is true do
4:          if getAnalysis(measurements, duration) then
5:              SD ← runSensorsDiagnosis(activityType)
6:              if SD then
7:                  messageAlert(SD)
8:              end if
9:              GD ← runGapsDiagnosis()
10:             if GD then
11:                 userInteration(GD)
12:             end if
13:         end if
14:     end while
15: end procedure
```

Line 2 defines the type of activity that is being analysed. The importance of this is that each of them has specific parameters to be analysed, namely, duration times, types of sensors, and the number of measurements. Line 3 is a call for the *durationAnalysis* method, described in code 7.1. If the return of it is *true*, it means that the user finished the activity without experiencing problems, regarding the time spent. Then, the relation between the total of measurements and the duration of the activity is analysed (line 4). The total of signs sent by sensors to the system (context data) is directly related to the time spent completing the activity and its type. This relation means that the more time the user performs an activity, the more context data the system should receive. Considering that, the importance of this analysis relies on the fact that, normally, activities of the same type that take a similar time to be completed should have a similar amount of context data once the same set of sensors are being used.

The relation between the measurements and the duration of activities can be formalized through different definitions, depending on the goals of the system. In this case study, it was used the average of measurements of all occurrences and the average of time spent in all occurrences of an activity. The following equation describes this:

$$f(Measurement, Duration) = \{\sum_{i=1}^{n} \frac{Measurement_i}{n}, \sum_{i=1}^{m} \frac{Duration_i}{m}\} \tag{7.1}$$

If a considerable discrepancy between them is evidenced, some actions must be taken to prevent the system from using inconclusive data. The first one is to run diagnosis tests aiming to find possible sensor hardware failures (line 5). If problems are found, proper actions must be taken, for instance, send warnings to the user or the service provider about it (line 7).

Services and applications of context-aware systems are executed based on the filling of pre-defined requirements. In other words, the system needs a set of context data that enables it to understand situations from the environment and trigger the most appropriate actions. Thus, when a discrepancy between the duration and the total of measurements of an activity is evidenced, the system must run a diagnosis test aiming to find possible information gaps and analyse how their absence influences the decision making (line 9). According to Lim and Dey (2011), one approach to deal with uncertainty is to ask the user for help to acquire additional data or eliminate the ambiguity of situations. Thus, if the system detects gaps of information, it could start an interaction with the user asking him for information to fill those gaps (line 11).

7.2 Dataset 2: Activity Daily Living Recognition

Dataset 1, used for the conception of the running example, explicitly presents the starting and finishing moment of time of the execution of an activity. This means that sets of lines could be used to represent one single situation. Different from that, dataset 2, described in Chapter 6, an activity can be represented in one single register. They are composed of, among other things, the type and local where the sensor was installed. Each register stores only one sensor. Thus, to identify activities where more than one sensor

was used, it is necessary to analyse sequences of lines, where the value of the attribute *ActivityName* is the same. Besides that, the interval between the *End* from a specific line must be close enough to the *Begin* of the next line. Assuming that small differences between the values of these attributes could not represent another execution of an activity of the same type, it is possible to infer that they compose the same occurrence, with sensors being used to detect sequences of actions. Following, table 7.1 describes this scenario.

N°	Begin	End	Location	Type	Place	Activity
587	2012-11-17 09:57:38	2012-11-17 09:57:42	Door	PIR	Kitchen	Breakfast
588	2012-11-17 09:58:41	2012-11-17 09:58:47	Fridge	Magnetic	Kitchen	Breakfast
589	2012-11-17 09:59:02	2012-11-17 09:59:05	Fridge	Magnetic	Kitchen	Breakfast
590	2012-11-17 09:59:26	2012-11-17 10:00:56	Microwave	Electric	Kitchen	Breakfast
591	2012-11-17 10:03:54	2012-11-17 10:03:58	Door	PIR	Kitchen	Breakfast

Table 7.1: Structure of dataset 2.

The values of attributes *End* (first line) and *Begin* (second line), highlighted in red, describe a slight difference of time-lapse between the two registers. This phenomenon must be present in all following lines. Besides that, in all registers, the value of attribute *Activity* is *Breakfast*, meaning that they refer to the same sort. Based on these pieces of evidence and the fact that different sensors were used along with the registers, it is possible to conclude that they represent the same occurrence of that activity. This inference will be used in the analysis of the following experiments of this section.

7.2.1 Duration Time Among Activities' Occurrences

The first experiment conducted with the second dataset refers to the analysis of the duration time in every activities' occurrences. Activities of the same type were grouped and the time spent by the user to perform them was calculated (in minutes). Following, figure 7.4 presents the result for the activity of cooking *breakfast*.

From the image, it is possible to state that the user follows a pattern of behaviour since the necessary time to prepare the meal was between 5 and 10 minutes in most of the occurrences. This amount of time was evidenced in 7 of the 16 occurrences (43,75%). The activity *Breakfast* was chosen due to its natural dynamicity, once sometimes the user may prepare a fast meal while in others when he has more time to enjoy it, he may prepare something more elaborated, that takes more time to be ready. Considering that, a volatile amount of time is expected during the analysis. However, some occurrences have present a duration time that could be considered way out of the expected, namely in 5, 7, 12, and 14. These cases should be analysed carefully. They could represent a scenario as aforementioned or represent uncertainty of any sort, and the Algorithm 7.1 should be applied.

This situation could evidence one kind of uncertainty, where even though all values of the registers are known, the interpretation of them is not clear compared to pre-defined standards. In this case, these standards could include, for instance, the system's knowledge about the user behaviour and the necessary

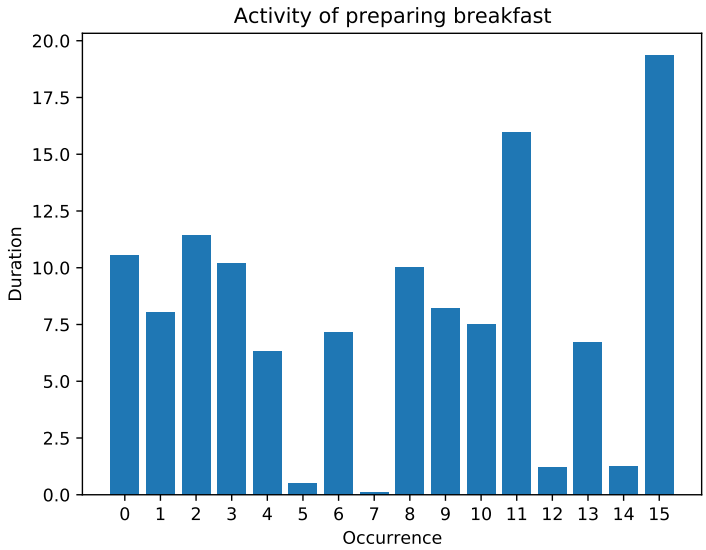


Figure 7.4: time spent for the *breakfast* activity occurrences.

time to prepare a breakfast meal. The system should also consider changes in behaviour. They may simply represent modifications related to the user's state of mind or something that requires more attention, such as the health state disturbance.

Considering that, there is apparently no missing data in the set represented by the graph, a detailed analysis of it should be performed in order to understand the discrepancy in the occurrences 5, 7, 12, and 14. If anomalies of any sort are found, the set of functions provided by the *Uncertainty module* should be used to cope with them. Algorithm 7.2 already described them in detail. After the correct identification of the activity, the relation between the total of measurements (including the five registers) and the activity's duration is checked. Then, four steps are performed in sequence. The execution of each of them depends on the results provided by the previous ones. Among them, user interaction stands out. In this case, it allows the user to inform the system of what is happening with him in the environment.

A different type of uncertainty arises when a register from the dataset presents a lack of data. In this situation, it is not possible to infer if it belongs to a subset that composes a specific activity. Following, table 7.2 presents an example.

N°	Begin	End	Location	Type	Place	Activity
16	2012-11-12 01:53:53	2012-11-12 01:53:57	Door	PIR	Bedroom	Sleeping
17	2012-11-12 01:55:09	null	null	null	null	null
18	2012-11-12 09:31:16	2012-11-12 09:31:19	Door	PIR	Bedroom	Sleeping

Table 7.2: Subset with missing data

By analysing the beginning and ending times from three tree lines, it could be possible to infer that the missing data in the second line refers to *Sleeping* activity. This deduction could be made once the first and third lines refer to this activity. However, considering that there is no *End* value for the second register, this

inference may not be the most suitable for this scenario. It also could represent another everyday activity for that time of the night, for instance, *Toileting* or preparing a *Snack*.

The most critical analysis of this table is that the uncertainty evidenced here has different characteristics from the previous example and yet, may compromise the interpretation of the environment in several forms. For instance, it would be impossible to perform the analysis presented in 7.4 without the necessary data to calculate the duration time of all the occurrences of a specific activity. Thus, to ensure that this analysis will be as accurate as possible, it is necessary to narrow this obstacle.

The classification provided by the decision tree described in 6.3 can be used to tackle the problem of missing information. Although the result will not be as accurate as actual data provided by sensors, it helps the system follow the stages of data analysis. For instance, the calculation of the activities' duration presented in 7.4 would not be possible to be done without all necessary values from attributes. However, by filling gaps of context data with values from the classification tree, it is possible to compute the time spent in each occurrence and look for any sort of discrepancy that may result in uncertain scenarios.

The context present in this example, allied with the distribution of the decision tree's activities, can help decrease the level of uncertain information. In this case, the context means the analysis of a set of rows together instead of processing them separately. For instance, row 17, when analysed isolated, may have a high level of uncertainty. However, the values of rows 16 and 18 may help to deal with this nebulous situation. Considering that the previous and following activities refer to the same type of activity, and the fact that this activity, specifically, takes a long time to be performed, one inference that could be made is that activity of row 17 is also *Sleeping*. This inference could be confirmed through the analysis of the classification provided by the decision tree. If the value of *Begin* attribute stays close to others in past occurrences, another inference could be made towards the classification as *Sleeping* activity. These two inferences complement each other, and together, help the minimization of uncertain situations.

It is important to emphasize that, besides identifying the type of activity, the rest of the values from row 17 should also be known once they may be useful for the analysis of future cases. The knowledge base of the framework that stores valid values from past situations can be queried to solve this problem. The system could retrieve sets of occurrences of this type of activity and analyse the values that compose them. Table 7.3 presents another set of occurrences of *Sleeping*, where similar values can be identified.

N°	Begin	End	Location	Type	Place	Activity
158	2012-11-13 01:29:52	2012-11-13 01:29:56	Door	PIR	Bedroom	Sleeping
159	2012-11-13 01:30:32	2012-11-13 08:51:15	Bed	Pressure	Bedroom	Sleeping
160	2012-11-13 08:52:03	2012-11-13 08:52:06	Door	PIR	Bedroom	Sleeping

Table 7.3: Occurrences of *Sleeping* activity

The training set used to build the distribution of the decision tree is used as the basis for the analysis of rows with missing data. The similarity present in the data of tables 7.2 and 7.3 also contributes to the minimization of negative impacts originated by uncertain situations.

Algorithm 7.1 from section 7.1 can be applied to this case study. If any sort of inconsistency regarding the duration, a sequence of actions is executed to cope with this problem. The system analyses the usual time the user takes to finish it from the calculation of the duration spent performing it. This value is compared against registers from the knowledge base. A further investigation is required when it does not match the set of acceptable values. A detailed diagnosis of the user's context is performed in situations like this to ensure that this sudden change is not related to his health state. Hardware and network verification are also checked.

7.2.2 Total Amount of Activities per Day

This case study analyses possible uncertainties related to the total amount of activities per day. Several aspects should be considered in the study. For instance, the user may have different routines during week and weekend days. Holidays also could affect the way he behaves. However, for the statistical analysis, these details were not considered. They were taken into account during the semantic analysis, where the reason for the possible discrepancy was verified. Figure 7.5 presents the total amount of activities performed per day along the experiment period (21 days).

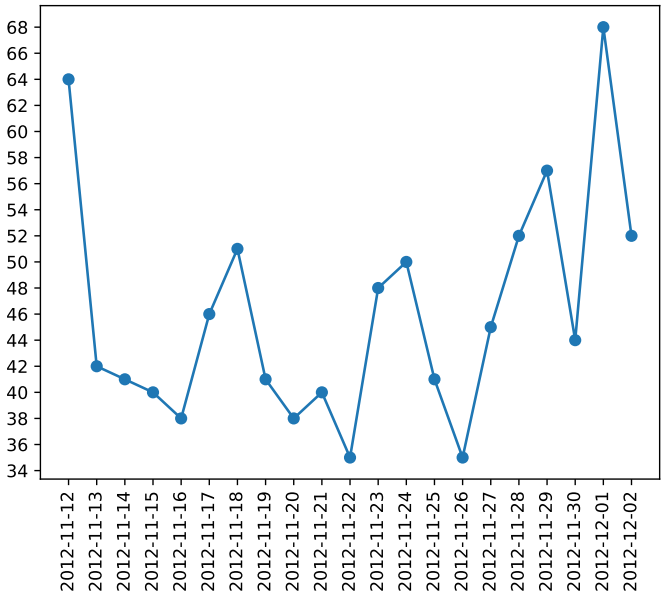


Figure 7.5: Amount of activities per day.

The graph shows considerable volatility along the days regarding the number of activities performed by the user. This scenario could difficult the process of behaviour identification once the probability of finding patterns tends to decrease. However, considering the domain of intelligent houses where ordinary activities are monitored and used to understand the user's behaviour, this volatility is natural because it is doubtful that the user would perform the exact same amount of tasks per day during a specific period.

The average of activities per day was used to analyse possible uncertainties in this case study. The methodology used in 7.2.1 was also applied here to compute the occurrences of the activities. I.e., the difference between the value of the attribute *End* in a line k and the value of the attribute *Begin* of the

next row (*k+1*) should be small enough in order to represent the same occurrence. After that, the average amount of activities per day was found, taking into account the total of 21 days of monitoring. At last, considering the average, the pattern deviation was calculated. Days where the number of activities was out of the pattern deviation, was considered to have an abnormal amount of activities performed by the user. The following data resulted from this calculation.

- Average amount of activities per day: 46.1;
- Pattern deviation: 8.7;
- Number of activities considered abnormal: 54,8 or more and 37,4 or less;
- Days with abnormal amount of activities: 2012-11-12, 2012-11-22, 2012-11-26, 2012-11-29, 2012-12-01.

It is important to emphasize that a day with an abnormal quantity of activities performed does not directly mean a problem of context sensibility nor anything related to the user's health state. As described earlier, in 7.1.2, several factors could influence the user's willingness to perform a group of tasks, for instance, holidays, week or weekend days, weather conditions and others. Nevertheless, considering the high discrepancy of the number of activities of this group (2012-11-12, 2012-11-22, 2012-11-26, 2012-11-29, 2012-12-01), the origin of this phenomenon should be carefully inspected once it also could represent problems that result in uncertain situations.

On the first day of monitoring (2012-11-12), a total of 64 activities were identified, representing 38.83% more activities than the average. This difference is even higher on day 2012-12-01, where it was evidenced 47.50% more activities (total of 68). A more subtle disparity appears in day 2012-11-29, in which the system identified 56 activities, representing 21,47% more than the average. A similar situation is witnessed in days 2012-11-22 and 2012-11-26 where, in both cases, the total of activities performed by the user was only 34, representing 26,25% less than the average.

The framework's modules deal with the cases where high discrepancies were identified. This means that additional verification is performed aiming to confirm the results. In this case study, this process is first applied to each activity individually, and after that, they are used to count the total amount of occurrences in one day. In the *Attribute Grammar* module, the structure of the activities is verified once more. In cases where inference rules were applied, they are also analysed.

The parameters of QoC, discussed in section 3.1, are used to ensure that the occurrence of a specific activity has all the requirements to be considered as of high quality. In this phase, the parameters are considered while checking the consistency of the context data that composes the activity.

The *Uncertainty* module is especially useful during the verification of cases where the number of activities in one day is considerably lower than the usual amount. This is evidenced in days 2012-11-22 and 2012-11-26. The discrepancy, in this case, may mean that the system was not able to identify a number o occurrences of the activity and, consequently, they are not included in the calculation of the total

amount. Several reasons could be related to this variation, for instance, the lack of an adequate amount of data to identify the activity or the simple fact that the user performed fewer tasks than usual. The former case could be addressed by the framework in the *Uncertainty* module. Several actions could be taken aiming to acquire more data that could help recognize the occurrence of an activity. Identifying possible failures related to the sensors and their consequent repairing would improve the acquisition of data. The explicit requisition of data through user interaction also helps the process. At last, the classification of the data provided by the decision tree also can be used to minimize the lack of certainty. The resource of the decision tree becomes a powerful asset since the dataset used to train it evolves along the time once data of current situations can be incorporated into it.

Besides facing high discrepancy regarding the metrics, the system should consider cases where the number of occurrences stays within the pre-established values but does not represent the actual amount of activities performed. I.e., when the system cannot identify an activity, the final number is within the pattern deviation. This phenomenon could represent a temporary flaw of the hardware and should be adequately dealt with.

When discrepancies remain even after the data being applied once more to the framework, it is reasonable to state that the amount of activities performed in one specific day does not present any uncertain aspects. It may reflect a change of routine.

7.3 Identification of Elements that Result in Uncertain Situations

The main objective of this thesis was the *identification of elements that result in the characterization of uncertain context in intelligent environments*. It was decided to use intelligent houses as the domain of study and the daily routine activities of users as context. This section aims the explicit elucidation of how the main objective was achieved through the case studies presented in this Chapter.

The first conclusion that can be evidenced based on the analysis of the cases is that there are two primary sources for the problems that impact the identification of context. One refers to the lack of data, i.e., datasets that do not provide complete facts for training and testing. In cases where the knowledge base is created in real-time, with sensors capturing and sending data to the context-aware system, this refers to pieces of data that are not used for some of the reasons discussed in the previous Chapters. Missing data from these two reasons impact the process of context identification. Actions can be taken to prevent or deal with this issue. The approach of ignoring missing values during the processing is one approach that is commonly adopted by context-aware systems. The result is that the environmental analysis is not as accurate as it could be, and, consequently, the service adaptation may be affected. Another way of tackling this is by filling the gaps of data with default values. They can be generated through past occurrences, where similar situations were identified and considered relevant, with complete data. In the approach used in this thesis, this refers to the dataset used as input. The classification provided by machine learning algorithms,

such as decision trees, can be consulted to find pertinent data to fill these gaps. A third method to improve the richness of the input data is through the execution of inference rules. Even with possible information gaps, they may derive extra pieces of valuable values for the context analysis.

The second source of problems that impact context analysis appears when, despite having the complete dataset values, they are not enough to provide information to be used by the system. In this type of situation, uncertainty may be more challenging to identify. The difference between relevance or not of a piece of data can be slight. It may only be identified if the goals of the system are well-defined. For instance, a timestamp that could be applied in the calculus of an interval of time depends on its value to be considered relevant, once its difference for an irrelevant timestamp is only the value that the attribute stores. This scenario evidences the importance of a careful analysis of the quality of context (QoC).

Considering that, several elements can be identified as being responsible for the characterization of uncertain contexts. The first one refers to the *changing of user's behaviour*. If the system is not prepared to deal with this variation, it may not understand that the user simply decided (or had) to perform a group of activities in a different form than the usual. The consequence of this is that it will not make the proper decisions, and the adaptation of services will not be as relevant as it could. *Hardware failures* is another element that can compromise the context analysis. This phenomenon implicates in the generation of aleatory data (e.g., random timestamp or user location) or even its absence. The use of outdated data is another issue that emerges when sensors stop working correctly.

The problems mentioned above impact another group of elements, evidenced in the case studies. For instance, in cases 7.1.1 and 7.1.2 the *time spent by the user to perform specific activities* was discussed, emphasizing the occurrences where discrepancies were identified. It is possible to infer that circumstances like this may be assumed to have possible uncertainties once they do not follow a pattern. However, this statement only stands if the data used during the analysis is valid and relevant.

In cases 7.1.2 and 7.2.2 the daily routine of the user was analysed by counting *the number of activities performed along the day*. This scenario can be seen as having a high complexity once several variables should be considered, as discussed in detail in the respective subsections. Nevertheless, the distinct aspects that they present indicate the possibility of using the user's habits as an element of analysis for the characterization of contexts.

At last, case 7.1.3 discussed a problem found in the *relation between the time used during activities and the number of measurements detected by sensors*. This element can be associated with several others described here. The variation spotted when analysing activities of the same type may indicate problems of correct functioning of sensors related to the user's health (e.g., stayed stand or seated without performing any movement).

The elements discussed in this section can influence the knowledge of the system about the context of an intelligent environment. They explicitly pointed out to accentuate how the case studies were used to achieve the primary goal of this thesis.

7.4 Summary

The proposal described in Chapter 6 was validated here through the analysis of five case studies. They were created based on the two public datasets used as running examples along with this document. The datasets refer to ordinary daily activities performed by one user within a specific interval of days in a smart house. The choice for using two independent datasets evidences the versatility of the approach once they have different structures.

For the study of the second dataset, some inferences and conventions were stated. This process is justified because it was necessary to identify the starting and finishing moment in which one occurrence of an activity happens.

Analyses were made, and the results were presented in graphs. Algorithms were presented to describe how the framework deals with uncertain situations after being identified in the case studies. The cases from section 7.2 do not have any algorithm once they use the same of section 7.1.

Two cases were created to validate the duration of specific activities, i.e., the necessary time for the user to accomplish the task. They were randomly chosen. Another two cases analyse possible discrepancies related to the number of activities performed per day. The mean number of activities and the value of the pattern deviation were used as parameters to define what could be considered an abnormal amount of occurrences. The fifth case study addressed the verification of problems related to the number of measurements collected by sensors (data capturing) and its relation with the activity duration. A formal definition of the relation *Measurements x Duration* was presented, aiming at the demonstration of how the calculation is done. Based on that, it is possible to conclude that the analyses of the cases can complement each other because some metrics are shared by two or more of them.

After evaluating the results evidenced in the case studies, the Chapter ends with a discussion of how they helped to support the hypothesis of this work. A detailed debate of how they contributed to the achievement of the main objective of the investigation was presented.

Chapter 8

Conclusion

This Chapter closes the Ph.D. work here reported. A summary of the dissertation is presented. The thesis goals and achievements are highlighted and shortly discussed. The scientific contributions are described. Collaborative links possible among universities are described here as future works. Directions for coming researches based on the results so far reached are proposed in the last section.

Context-aware systems are, gradually, becoming a reality. Intelligent devices that naturally interact with users are already available on the market. They will help the consolidation of Ambient Intelligence through intelligent environments. To build this kind of application, it is necessary to develop context-aware systems, which can sense what is happening in the environment and behave based on that. The system should use data of the surroundings to execute services aiming to assist users. However, this reasoning is not a trivial task. Moreover this complexity can be augmented due to the fact that data collected from sensors can be incomplete or incorrect. Such domains tend to be dynamic. Hence, context-aware systems may not be able to process situations due to the incompleteness of data or its reliability.

Considering this, it is possible to conclude that one of the main issues of context-aware systems is the study of approaches to handle appropriately uncertain or incomplete contexts. The assurance of correct actions in situations where the system cannot build a complete context or trustable model due to vague, partial, or outdated information is seen as an obstacle to overcome.

This document reported a doctoral thesis aimed at researching some open questions about the uncertainty of information in context-aware systems.

Researches about the Internet of Things (IoT), and Artificial Intelligence (AI), are currently in evidence.

One of the main goals is to build environments with intelligent devices acting together autonomously, sharing information, and improving users' quality of life. The results of the research, as defined in the hypothesis (section 1.3) include the increase of the system's intelligence by providing consistent and relevant data to be processed, mitigation of misinterpretation of contexts, improvement of user behaviour monitoring, and minimization of decisions based on default values. Thus, the achievement of objectives proposed in Chapter 1 allowed the improvement of the level of intelligence of autonomous systems, contributing to the evolution of the IoT paradigm.

State of the art was organized in such a way that it was possible to describe the characteristics, primary contributions, and relevance of each research topic, emphasizing how they were incorporated in the approach proposed. In Chapter 2 fundamental phases of the conception of a context-aware system were discussed (principles of design, data acquisition alternatives, modeling, and processing). A new vision of the concept of *context* was proposed, considering the domain of study. The running example presented in section 1.6 was used to illustrate the discussion. It refers to the first dataset used to build the case studies that validated the proposal.

Chapter 3 addressed the issue of uncertainty in context-aware systems. Two main types were described: Aleatory, in which the source is related to hardware failures and communication issues, and; Epistemic, which refers to problems of modeling the system. Its presence may affect the semantics of the system. Besides that, vision from authors with other classifications was discussed and related to the Aleatory and Epistemic types. The importance of the presentation of uncertainty for the user to be aware of was also addressed. Obstacles regarding this question in the Human Activity Recognition field were highlighted, emphasizing how this work tackled it.

Decision trees models for classification of context data were addressed in Chapter 4 and the most used approaches were discussed. CART algorithm was applied to the scope of this research, and it was extensively explored. A model for ADL based on this method was presented, aiming to label situations from a dataset. It was based on the identification of GINI impurity values to define the amount of uncertainty the dataset contains. Based on that, the Information Gain was calculated, aiming to maximize purifying the separation of data according to similar features. The distribution achieved by it can be used to assist the framework in facing situations where context data present flaws. Thus, ambiguities and gaps of data can be tackled and, consequently, improving the system's knowledge about the environment.

The analysis of the existing approaches for semantic modeling was considered for the proposition of a framework to validate context data. This will increase the capacity to abstract context data with new forms of mapping the entities, relations, and axioms. Thus, the frequency the system faces with uncertain contexts tends to decrease.

This Ph.D. thesis aimed to present an approach for overcoming problems related to data used by context-aware systems developed for intelligent environments. More specifically, it is believed that issues of this kind lead the system to work under uncertain situations. This directly affects the capacity of deriving conclusions based on information from the environment and, consequently, self-adaptation to context changes may be compromised. By identifying the source of uncertainty, i.e., elements that result in the

characterization of nebulous scenarios, it is possible to tackle the problem on its origin.

The integrity of data used as input for computation on such domain influences the system's behaviour. Thus, the approach presented in this thesis aimed to verify context data from three different perspectives in sequential order.

After receiving the data from sensors, the system submits it to the first layer of validation. There, its structure is checked. An attribute grammar to cope with this validation was proposed. The main goal of the grammar is to ensure that the data obey restrictions imposed by structural rules and contextual conditions. The verification is done through the use of synthesized and inherited attributes. A set of inference and semantic rules can be defined according to the domain under study. Examples of rules were described and applied to the datasets during the experiments. Inference rules allow the generation of additional knowledge based on the data and can be used in future context analysis.

Two main contributions arise from this approach. The first refers to the verification of the structure of data gathered from the ambient. This process prevents the system from working with invalid data. The second, and considered the most important, relays on the intersection of two distinct fields of investigation. Despite being considered a consolidated area, formal grammars, more specifically, attribute grammars, were not applied to context awareness solutions. By proposing this relation, a new range of possibilities of dealing with context data may emerge. Details about the attribute grammar were extensively discussed in Chapter 5.

The second layer of validation refers to the quality of data, and it is only executed in case of successful output from the previous one. In this phase, a some metrics are analysed. In the current domain used as the subject of study, they refer to the accuracy, the set of acceptable values, the source of acquisition, its format, and the level of usefulness. Different parameters could be defined for other domains. The verification was performed under two perspectives, micro, and macro analysis. In the former, each piece of data is individually verified. In this phase, the semantics of situations is not taken into consideration. In the latter, the set of data is analysed as being part of one situation, and possible semantics should be taken into account. Only successful outputs from this validation, based on the parameters, are stated as having high Quality of Context. It is important to emphasize that it should pass through verification in all software agents before discarding a piece of data. Considering that software agents have unique goals, and could use the same context data from different perspectives, one piece of data labeled as irrelevant for one agent, could be helpful to others.

The main contribution of ensuring the QoC relies on the fact that it increases the level of certainty of the system when triggering services. Erroneous interpretation of what happens in the environment is a common problem and should be overcome. It could be minimized if the system uses only data previously validated. One of the main challenges on context-aware applications relies on the misinterpretation of what the user is doing, where he is, and his intentions. For instance, virtual assistants of intelligent devices are becoming common, but still do not always understand the user's requirements or intentions properly. Mainly, this problem happens because there is a gap between what the system knows, what it is capable of executing, and what is the user's actual willingness. Besides being a system's modeling problem, it also

characterizes uncertainty. This disparity leads to equivocate definitions of situations and, consequently, the execution of incorrect services, e.g., wrong suggestions for the user about the place or bad interactions. Thus, by identifying the nature of erroneous interpretations of context data, the system behaviour will certainly improve.

The last layer of validation seeks to find other sorts of problems. The assurance of well-defined structure (layer 1) and QoC (layer 2) does not exclude the possibility of issues like the incompleteness of contexts. A formal definition of context was proposed in the first step of uncertainty analysis. Thus, relevant changes in the environment were classified as *situation*, which occurs in specific moments involving specific sets of entities. From that, it was possible to identify what should be considered for context processing and what could be ignored. An algorithm to find and deal with gaps of data was described. It provides functions to identify this problem, including hardware analysis and elimination of situations considered invalid. Security measures were defined, aiming to improve data completeness and ensure the user's safety. Then, the decision tree model was described. It classified a dataset according to the type of activity. When actions to cope with uncertain events cannot minimize them due to a lack of data, the distribution provided by the tree is used to fill the gaps. This new information tends to improve the context analysis by enriching the knowledge of situations.

The three layers of validation complement each other and compose the framework. It also has a module with a knowledge base that stores situations detected along the time. Every time a new context is used, the knowledge base is fed with its information. The framework proposed relies on the intersection of several fields of research. Along with that, it was possible to evidence, through the experiments conducted, that they can be combined each other aiming for the improvement of the intelligent environment.

The approach presented in this thesis was validated through case studies. They were created based on two public datasets that store registers from ADL in intelligent houses. It is important to emphasize that their structures are entirely different from one to another, yet the framework layers can handle them without troubles. The experiments conducted with the datasets were extensively discussed in Chapter 7 highlighting the achievements attained and how they contribute to accomplishing the main objective and hypothesis of the research.

8.1 Contributions

Considering the open research question, hypothesis and objectives proposed for this work, it is possible to evidence the following contributions for the scientific community:

- Integration of two different fields of investigation: attribute grammars and context awareness, through the proposition an approach where they complement each other to analyse data retrieved from intelligent environments;
- Situation analysis based on more relevant and complete dads, achieved through the validation of the quality of context;

- Verification of context data under three perspectives: by ensuring the quality of structure, its relevance and completeness;
- Framework capable of dealing with different structures of context data.

8.2 Validation of the Hyposthesis

The hypothesis presented in section 1.3 was addressed throughout the Chapters of this work. By validating context data regarding its structure, relevance and completeness, the intelligence of context-aware systems tends to be improved, once the decision-making will be based on more accurate data. Thus, the following list present an brief and explicit discussion of how the hypothesis was validated:

- *The perception of the system increases once the decision-making will be based on consistent and relevant data.*
  The verification of the quality of context performed by the proposed framework improves the relevance of the data. This quality was ensured by the analysis of the accuracy, usefulness, up-to-dateness, representation, and importance. By using only data with all these characteristics, the quality of computing processing tend to be improved;
- *Wrong service execution, caused by misinterpretation of context, decreases.*
  One of the causes of the wrong execution of services is the lack of context data. When the system cannot absorb sufficient data from the environment, process it to transform it into knowledge to be used to understand the situation, equivocates services are triggered. The classification of the context data provided by the decision tree allowed the system to used more complete data. I.e., when it faces situations with only a few pieces of context, it accesses the knowledge base requiring more. This extra set of data help improve the details of situations;
- *Identification of changes related to the user's behaviour is facilitated.*
  The main goal of the third module of the framework is to identify and deal with any unexpected situations. On the other hand, the decision tree classification results from the analysis of features that characterize activities. Both of them provide subsidies for the recognition of patterns of behaviours and, consequently, if the user performs an activity differently, this modification can be easily identified;
- *Decisions based on default values can be mitigated.*
  The classification provided by the decision tree is helpful in cases where there is no other possibility but to use default values front one situation. Nevertheless, the necessity of this resource tends to be minimized. The third layer of the framework comprises a series of methods to tackle cases where the lack of data leads to uncertainty. For instance, the detailed requisition for users to input new data prevents the system from considering default values in decision-making.

8.3 Academic Outcomes

The candidate was inserted in research groups at the University of Minho, where he had the opportunity to develop activities related to the Ph.D. work. Thus, this section describes achievements at the academic level from different perspectives: the scientific publications, participation in events of the field, and supervision of students.

8.3.1 Scientific Publications

The possibility of submitting partial and final achievements for the experts' analysis of the fields is a good way of validating the work. This process provides relevant insights to improve its quality. Thus, the following articles were published in scientific conferences and journal:

- *Uncertainty in Context-Aware Systems: A Case Study for Intelligent Environments.*
  Published in: World Conference on Information Systems and Technologies (WorldCIST 2018), Trends and Advances in Information Systems and Technologies, Vol. 745, pp. 225-231.
  This paper presented a case study to understand better concepts related to context awareness and dealing with inaccurate data. Through the analysis of identification of elements that result in the construction of unreliable contexts, it is aimed to identify patterns to minimize incompleteness. Concepts related to context awareness and uncertainty were discussed by providing association rules for situations' processing (Freitas et al., 2018);

- *Context Awareness and Uncertainty: Current Scenario and Challenges for the Future.*
  Published in: International Symposium on Ambient Intelligence (ISAmI 2018), Ambient Intelligence – Software and Applications, Vol. 806, pp 174-181.
  The main goal of this paper was to identify problems in context-aware systems that result in nebulous situations. Sources of uncertainty were presented, such as the lack of accurate data used by the system and misinterpretation of contexts. Speculative computing was discussed as an efficient alternative for the lack of data (Freitas et al., 2019b);

- *Attribute Grammar Applied to Human Activities Recognition in Intelligent Environments.*
  Published in: International Symposium on Ambient Intelligence (ISAmI 2019), Ambient Intelligence – Software and Applications. Vol. 1006, pp 62-70.
  This paper proposed the use of formal grammars to deal with problems of intelligent environments. An attribute grammar was described in order to create a formal specification of situations in such domains. The problem of representation of human activities is tackled through a case study to demonstrate how grammars can help the improvement of this process (Freitas et al., 2019a);

- *Knowledge Inference Through Analysis of Human Activities.*
  Published in: Intelligent Data Engineering and Automated Learning (IDEAL 2019). Part of the Lecture Notes in Computer Science, Vol. 11871, pp. 274-281.

Here, a model to validate daily human activities based on an Attribute Grammar was presented. Context data was analysed through the execution of rules that implement semantic statements. This processing, called semantic analysis, highlights problems that uncertain situations can raise. The work's main contribution is the proposal of a rigorous approach to deal with context-aware decisions (decisions that depend on the data collected from the sensors in the environment) in such a way that uncertainty can be detected and its harmful effects can be minimized. (Freitas et al., 2019c);

- *CAPAS: A Context-Aware System Architecture for Physical Activities Monitoring.*
  Published in: Hybrid Artificial Intelligent Systems (HAIS 2019). Part of the Lecture Notes in Computer Science, Vol. 11734, pp. 636-647.
  This paper is the result of a master's degree dissertation co-supervised by the candidate. It presented a Context-aware system Architecture to monitor Physical ActivitieS (CAPAS). One of the architecture's components is the attribute grammar which verifies the users' movements. According to some predefined rules, the physical activity is validated after an analysis of the grammar. Besides that, it is integrated with a system aiming at the recognition of physical activities. (Ferreira et al., 2019);

- *Analysis of human activities and identification of uncertain situations in context-aware systems.*
  To be published in: International Journal of Artificial Intelligence (IJAI 2020), Vol. 18.
  The goal of this work was to present an approach to tackle the problem of incomplete data used for decision-making. The idea for coping with this trouble was by dividing it into three parts. The first one refers to the validation of context data through Attribute Grammar. The formalism and expressiveness provided a grammar enriched with attribute evaluation rules and contextual constraints, ensures that the system will use only valid data for reasoning. Anomalous data will be detected, and the situation will be signalized. Also, the analysis of Quality of Context is provided, considering a set of characteristics, vouching that only useful information will be considered. At last, the identification of the sources of uncertain situations followed by a sequence of actions aiming to minimize the negative impacts of it helps the system work with more complete sets of data. The formalization of the approach is provided together with an algorithm to validate it. (Freitas et al., 2020). Q1 - Artificial Intelligence;

- *Uncertainty Identification in Context-Aware Systems Using Public Datasets.*
  To be published in: International Symposium on Ambient Intelligence (ISAmI 2021).
  The proposal of this paper includes the definition of a decision tree to classify context data and use it to reduce the level of uncertainty while building situations. The conduction of the experiments was through case studies created based on two public datasets containing Activity Daily Living data from smart houses.**accepted for publication**.

8.3.2 Participation in Scientific Events

During the Ph.D. course, the candidate had the opportunity to participate in several scientific events, such as summer school and conferences. Besides presenting papers with the ongoing status of the research, this type of occasion provided the possibility of the candidate getting in contact with other researchers of related fields and creating a scientific network. The following list presents these events.

- International Summer School on Deep Learning (DeepLearn2017), in 2017, Bilbao, Spain;
- 9^{th} International Symposium on Ambient Intelligence (ISAmI), in 2018, Toledo, Spain;
- 10^{th} International Symposium on Ambient Intelligence (ISAmI), in 2019, Ávila, Spain;
- ^{14th} Hybrid Artificial Intelligent Systems (HAIS), in 2019, León, Spain;
- 19^{th} Intelligent Data Engineering and Automated Learning (IDEAL), in 2019, Manchester, England.

8.3.3 Supervision of Students

Along with the development of this Ph.D. thesis, the candidate co-supervised works of students from the Master's Course in Informatics Engineering of the University of Minho. The subjects were always related to this work. The following list briefly describes the works developed by the students:

- **Paulo Alexandre Azevedo Ferreira**
  A context-aware system architecture to assist workout plans.
  Academic year: 2018/ 2019.
  In co-supervision with Professor Paulo Novais.
  An architecture for systems aiming to assist users while performing workout activities was presented in this work (Ferreira, 2019). Sensors from smartphones were used to collect data, and the results from three machine learning algorithms were analysed for activity recognition, Support Vector Machine, K-Nearest Neighbour, and Random Forest. The results were validated and tested with an overall accuracy between 61% and 99%, depending on the activity. The achievements of the dissertation were published at HAIS 2019 ((Ferreira et al., 2019);

- **Eduardo de Lima Cunha**
  Context Analysis for the Improvment of Intelligent Vehicules Networks.
  Academic year: 2018/ 2019.
  In co-supervision with Professor Pedro Rangel Henriques.
  This study analysed car accidents data from a specific region and alert drivers for possible exposure to dangers. Association Rules Learning was used for mining the data, providing the recognition of three types of alerts. They refer to a minor, medium, and high risks of car accidents. The approach integrated fields of investigation related to Intelligent Transportation Systems and context awareness to propose the concept of Context-Aware Vehicle Networks (CAVN), (Lima Cunha, 2020).

8.4 Future Work

Considering the results achieved in this thesis and the scientific contributions presented, it is possible to address some future research directions to complement and extend this project.

Decision trees provided a satisfactory level of accuracy in the labeling of data according to similar characteristics. Taking into account the wide range of approaches of supervised learning, other algorithms could be explored. This type of comparison tends to improve the research. The definition of metrics for this comparative study should be stated in advance.

The datasets selected to create the case studies, used to validate the proposal, present distinct structures. This was described as one of the pillars to ensure its relevance and efficiency. Thus, the validation could be extended to other datasets that store information about the ADL of intelligent houses. Possible adaptations to the current framework, specifically an attribute grammar, may be necessary. Despite that, the wider is the coverage of the framework, the more rigorous will be the uncertainty dealing.

The thesis presents a rich analysis of the data, allowing accurate identification of elements that result in uncertain situations. However, new layers of validation could be considered, aiming for the continuous improvement of the investigation. The architecture of the framework allows its extension without compromising what was already achieved.

Another aspect that can be explored is the possibility of building a test workbench, i.e., an intelligent house laboratory. The experiments conducted in such domains could present completely different characteristics once they would be based on new and personalized data, including, for instance, other types of activities and environments with another amount of users.

At last, future cooperation with the University of Minho (UMinho) is intended. The proponent works as an Adjunct Professor of graduation courses at the Polytechnic School of the Federal University of Santa Maria (UFSM), RS, Brazil. The proponent developed his doctoral research at the Doctoral Program in Informatics of UMinho, Portugal. Considering this, the outcomes achieved from this work shall be used by graduation and post-graduation students of all levels (master and doctoral) in scientific researches developed at UFSM and UMinho through cooperation initiatives.

Besides that, the possibility of student exchanges will be analysed, with Brazilian graduation and post-graduation students developing research at UMinho and Portuguese students contributing to projects at UFSM.

Bibliography

Aloulou, H., M. Mokhtari, T. Tiberghien, R. Endelin, and J. Biswas (Mar. 2015). "Uncertainty Handling in Semantic Reasoning for Accurate Context Understanding." In: *Know.-Based Syst.* 77(C), pp. 16–28. issn: 0950-7051. doi: 10.1016/j.knosys.2014.12.025.

Alpaydin, E. (2014). *Introduction to Machine Learning*. 3rd ed. Adaptive Computation and Machine Learning. MIT Press: Cambridge, MA. isbn: 978-0-262-02818-9.

Amdouni, S., M. Barhamgi, D. Benslimane, and R. Faiz (Apr. 2014). "Handling Uncertainty in Data Services Composition." en. In: *International Conference on Services Computing SCC*. Ed. by IEEE, pp. 653–660. isbn: 978-1-4799-5065-2. doi: 10.1109/SCC.2014.91.

Anisha, P. R. and A. Polati (2021). "A Bird Eye View on the Usage of Artificial Intelligence." In: *Communication Software and Networks*. Ed. by S. C. Satapathy, V. Bhateja, M. Ramakrishna Murty, N. Gia Nhu, and J. Kotti. Springer Singapore: Singapore, pp. 61–77. isbn: 978-981-15-5397-4.

Antifakos, S., A. Schwaninger, and B. Schiele (2004). "Evaluating the Effects of Displaying Uncertainty in Context-Aware Applications." In: *UbiComp 2004: Ubiquitous Computing: 6th International Conference, Nottingham, UK, September 7-10, 2004. Proceedings*. Ed. by N. Davies, E. D. Mynatt, and I. Siio. Springer Berlin Heidelberg: Berlin, Heidelberg, pp. 54–69. doi: 10.1007/978-3-540-30119-6_4.

Aztiria, A., J. C. Augusto, R. Basagoiti, A. Izaguirre, and D. J. Cook (2013). "Learning Frequent Behaviors of the Users in Intelligent Environments." In: *IEEE Trans. Systems, Man, and Cybernetics: Systems* 43(6). https://doi.org/10.1109/TSMC.2013.2252892, pp. 1265–1278. doi: 10.1109/TSMC.2013.2252892.

Bauer, M., T. Heiber, G. Kortuem, and Z. Segall (1998). "A Collaborative Wearable System with Remote Sensing." In: *Second International Symposium on Wearable Computers (ISWC 1998), Pittsburgh, Pennsylvania, USA, 19-20 October 1998, Proceedings*. https://doi.org/10.1109/ISWC.1998.729524, pp. 10–17. doi: 10.1109/ISWC.1998.729524.

Bellotti, V. and K. Edwards (Dec. 2001). "Intelligibility and Accountability: Human Considerations in Context-aware Systems." In: *Hum.-Comput. Interact.* 16(2), pp. 193–212. issn: 0737-0024. doi: 10.1207/S15327051HCI16234_05.

Ben Yaghlane, A., T. Denoeux, and K. Mellouli (2008). "Uncertainty and intelligent information systems." In: ed. by M. R. B. Bouchon-Meunier C. Marsala and R. R Yager. World Scientific. Chap. Elicitation of Expert Opinions for Constructing Belief Functions, pp. 75–89.

Bernardos, A. M., P. Tarrío, and J. R. Casar (2008). "A data fusion framework for context-aware mobile services." In: *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems, MFI 2008, Seoul, South Korea, August 20-22, 2008*. https://doi.org/10.1109/MFI.2008.4648011, pp. 606–613. doi: 10.1109/MFI.2008.4648011.

Bettini, C., O. Brdiczka, K. Henricksen, J. Indulska, D. Nicklas, A. Ranganathan, and D. Riboni (Apr. 2010). "A Survey of Context Modelling and Reasoning Techniques." In: *Pervasive Mob. Comput.* 6(2). http://dx.doi.org/10.1016/j.pmcj.2009.06.002, pp. 161–180. issn: 1574-1192. doi: 10.1016/j.pmcj.2009.06.002.

Bhatnagar, R. K. and L. N. Kanal (1986). "Handling Uncertain Information: A Review of Numeric and Non-Numeric Methods." In: *Uncertainty in Artificial Intelligence*. Ed. by L. N. Kanal and J. F. Lemmer. North-Holland: Amsterdam, pp. 3–26.

Bobek, S. and G. J. Nalepa (2017). "Uncertain context data management in dynamic mobile environments." In: *Future Generation Comp. Syst.* 66. https://doi.org/10.1016/j.future.2016.06.007, pp. 110–124. doi: 10.1016/j.future.2016.06.007.

Bolton, M. L., R. I. Siminiceanu, and E. J. Bass (2011). "A Systematic Approach to Model Checking Human–Automation Interaction Using Task Analytic Models." In: *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans* 41(5), pp. 961–976. issn: 1083-4427.

Boulkrinat, S., A. HadjAli, and A. A. Mokhtari (2014). "Handling preferences under uncertainty in recommender systems." In: *IEEE International Conference on Fuzzy Systems, FUZZ-IEEE 2014, Beijing, China, July 6-11, 2014*. https://doi.org/10.1109/FUZZ-IEEE.2014.6891823, pp. 2262–2269. doi: 10.1109/FUZZ-IEEE.2014.6891823.

Breiman, L., J. H. Friedman, R. A. Olshen, and C. J. Stone (1984). *Classification and Regression Trees*. Wadsworth and Brooks: Monterey, CA.

Buchholz, T. and M. Schiffers (2003). "Quality of Context: What It Is And Why We Need It." In: *In Proceedings of the 10th Workshop of the OpenView University Association: OVUA'03*.

Burger, C., S. Karol, and C. Wende (Jan. 2010). "Applying attribute grammars for metamodel semantics." In: *ECOOP 2010 Workshop Proceedings - International Workshop on Formalization of Modeling Languages, FML'10*. doi: 10.1145/1943397.1943398.

Bürger, C., S. Karol, and C. Wende (Jan. 2010). "Applying attribute grammars for metamodel semantics." In: *ECOOP 2010 Workshop Proceedings - International Workshop on Formalization of Modeling Languages, FML'10*. doi: 10.1145/1943397.1943398.

Camara, J., W. Peng, D. Garlan, and B. Schmerl (2018). "Reasoning about sensing uncertainty and its reduction in decision-making for self-adaptation." In: *Science of Computer Programming* 167, pp. 51 –69. issn: 0167-6423. doi: 10.1016/j.scico.2018.07.002.

Canepa, G. A. (2016). *What You Need to Know about Machine Learning Leveraging data for future telling and data analysis*. 1st. Packt Publishing Ltd.

Chahuara, P., F. Portet, and M. Vacher (2017). "Context-aware decision making under uncertainty for voice-based control of smart home." In: *Expert Systems with Applications* 75, pp. 63 –79. issn: 0957-4174. doi: https://doi.org/10.1016/j.eswa.2017.01.014.

Chen, H., T. W. Finin, A. Joshi, L. Kagal, F. Perich, and D. Chakraborty (2004). "Intelligent Agents Meet the Semantic Web in Smart Spaces." In: *IEEE Internet Computing* 8(6), pp. 69–79.

Chen, S., Z. Wang, J. Liang, and X. Yuan (2018). "Uncertainty-aware visual analytics for exploring human behaviors from heterogeneous spatial temporal data." In: *Journal of Visual Languages and Computing* 48, pp. 187 –198. issn: 1045-926X. doi: 10.1016/j.jvlc.2018.06.007.

Chihani, B., E. Bertin, F. Jeanne, and N. Crespi (2011). "Context-Aware Systems: A Case Study." In: *Digital Information and Communication Technology and Its Applications - International Conference, DICTAP 2011, Dijon, France, June 21-23, 2011, Proceedings, Part II*, pp. 718–732. doi: 10.1007/978-3-642-22027-2_60.

Chomsky, N. (1956). "Three models for the description of language." In: *IRE Transactions on Information Theory* 2(3), pp. 113–124. issn: 0096-1000. doi: 10.1109/TIT.1956.1056813.

Cook, D. J., J. C. Augusto, and V. R. Jakkula (2009). "Review: Ambient Intelligence: Technologies, Applications, and Opportunities." In: *Pervasive Mob. Comput.* 5(4), pp. 277–298. issn: 1574-1192. doi: 10.1016/j.pmcj.2009.04.001. url: http://dx.doi.org/10.1016/j.pmcj.2009.04.001.

Cook, D. J. and S. K. Das (Mar. 2007). "How Smart Are Our Environments? An Updated Look at the State of the Art." In: *Pervasive Mob. Comput.* 3(2). http://dx.doi.org/10.1016/j.pmcj.2006.12.001, pp. 53–73. issn: 1574-1192. doi: 10.1016/j.pmcj.2006.12.001.

Cook, D. J. and M. Schmitter-Edgecombe (2009). "Assessing the quality of activities in a smart environment." In: *Methods of information in medicine* 48 5, pp. 480–5.

D'Aniello, G., V. Loia, and F. Orciuoli (Dec. 2017). "Adaptive Goal Selection for improving Situation Awareness: the Fleet Management case study." In: *Procedia Computer Science* 109, pp. 529–536. doi: 10.1016/j.procs.2017.05.332.

Delmastro, F., V. Arnaboldi, and M. Conti (2016). "People-centric computing and communications in smart cities." In: *IEEE Communications Magazine* 54(7). https://doi.org/10.1109/MCOM.2016.7509389, pp. 122–128. doi: 10.1109/MCOM.2016.7509389.

Dey, A. K. (1998). "Context-aware computing: The CyberDesk project." In: *AAAI 1998 Spring Symposium on Intelligent Environments*. http://www.cc.gatech.edu/fce/cyberdesk/pubs/AAAI98/AAAI98.html. AAAI Press.: Palo Alto, pp. 51–54.

Dey, A. K. and G. D. Abowd (1999). "Towards a better understanding of context and context-awareness." In: *In HUC '99: Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing*. Springer-Verlag, pp. 304–307.

Dey, A. K., G. D. Abowd, and D. Salber (Dec. 2001). "A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-aware Applications." In: *Hum.-Comput. Interact.* 16(2), pp. 97–166. issn: 0737-0024. doi: 10.1207/S15327051HCI16234_02.

Djoudi, B., C. Bouanaka, and N. Zeghib (2016). "A formal framework for context-aware systems specification and verification." In: *Journal of Systems and Software* 122(Supplement C), pp. 445 –462. issn: 0164-1212. doi: https://doi.org/10.1016/j.jss.2015.11.035.

El Naqa, I. and M. J. Murphy (2015). "What Is Machine Learning?" In: *Machine Learning in Radiation Oncology: Theory and Applications*. Ed. by I. El Naqa, R. Li, and M. J. Murphy. Springer International Publishing: Cham, pp. 3–11. doi: 10.1007/978-3-319-18305-3_1. url: https://doi.org/10.1007/978-3-319-18305-3_1.

Ferreira, P., L. O. Freitas, P. R. Henriques, P. Novais, and J. Pavón (2019). "CAPAS: A Context-Aware System Architecture for Physical Activities Monitoring." In: *Hybrid Artificial Intelligent Systems - 14th International Conference, HAIS 2019, León, Spain, September 4-6, 2019, Proceedings*. Ed. by H. P. García, L. Sánchez-González, M. C. Limas, H. Quintián-Pardo, and E. S. C. Rodríguez. Vol. 11734. Lecture Notes in Computer Science. Springer, pp. 636–647. doi: 10.1007/978-3-030-29859-3_54.

Ferreira, P. A. A. (2019). "A context-aware system architecture to assist workout plans." MSc dissertation. Master's thesis. Braga, Portugal: Minho University.

Freitas, L. O., P. R. Henriques, and P. Novais (2019a). "Context-Awareness and Uncertainty: Current Scenario and Challenges for the Future." In: *Ambient Intelligence – Software and Applications –, 9th International Symposium on Ambient Intelligence*. Ed. by P. Novais, J. J. Jung, G. Villarrubia González, A. Fernández-Caballero, E. Navarro, P. González, D. Carneiro, A. Pinto, A. T. Campbell, and D. Durães. Springer International Publishing: Cham, pp. 174–181.

Freitas, L. O., P. R. Henriques, and P. Novais (2019b). "Context-Awareness and Uncertainty: Current Scenario and Challenges for the Future." In: *Ambient Intelligence – Software and Applications –, 9th International Symposium on Ambient Intelligence*. Ed. by P. Novais, J. J. Jung, G. Villarrubia González, A. Fernández-Caballero, E. Navarro, P. González, D. Carneiro, A. Pinto, A. T. Campbell, and D. Durães. Springer International Publishing: Cham, pp. 174–181. isbn: 978-3-030-01746-0.

Freitas, L. O., P. R. Henriques, and P. Novais (2019c). "Knowledge Inference Through Analysis of Human Activities." In: *Intelligent Data Engineering and Automated Learning – IDEAL 2019*. Ed. by H. Yin, D. Camacho, P. Tino, A. J. Tallón-Ballesteros, R. Menezes, and R. Allmendinger. Springer International Publishing: Cham, pp. 274–281. isbn: 978-3-030-33607-3.

Freitas, L. O., P. R. Henriques, and P. Novais (Oct. 2020). "Analysis of human activities and identification of uncertain situations in context-aware systems." In: *(JAI) International Journal of Artificial Intelligence* 18 (2). http://www.ceser.in/ceserp/index.php/ijai/article/view/6576. issn: 0974-0635.

Freitas, L. O., P. R. Henriques, and P. Novais (2018). "Uncertainty in Context-Aware Systems: A Case Study for Intelligent Environments." In: *Trends and Advances in Information Systems and Technologies*. Ed. by Á. Rocha, H. Adeli, L. P. Reis, and S. Costanzo. Springer International Publishing: Cham, pp. 225–231. isbn: 978-3-319-77703-0.

Giese, M., T. Mistrzyk, A. Pfau, G. Szwillus, and M. von Detten (2008). "AMBOSS: A Task Modeling Approach for Safety-Critical Systems." In: *Engineering Interactive Systems*. Ed. by P. Forbrig and F. Paternò. Springer Berlin Heidelberg: Berlin.

Gil, A. C. (2010). *Como elaborar projetos de pesquisa*. 5.ed. Atlas: São Paulo.

Hassani, H. (2017). "Research Methods in Computer Science: The Challenges and Issues." In: *CoRR* abs/1703.04080. http://arxiv.org/abs/1703.04080.

Helton, J. (1997). "Uncertainty and sensitivity analysis in the presence of stochastic and subjective uncertainty." In: *Journal of Statistical Computation and Simulation* 57(1-4), pp. 3–76. doi: 10.1080/00949659708811803.

Henriques, P. R. (Nov. 2013). "Brincando às Linguagens com Rigor: Engenharia Gramatical." R. da Universidade, 4710-057, Braga - Campus de Gualtar: Departamento de Informática da Escola de Engenharia da Universidade do Minho.

Hoyos, J. R., J. Garca-Molina, J. A. Bota, and D. Preuveneers (Oct. 2016). "A Model-driven Approach for Quality of Context in Pervasive Systems." In: *Comput. Electr. Eng.* 55(C), pp. 39–58. issn: 0045-7906. doi: 10.1016/j.compeleceng.2016.07.002.

Hssina, B., A. MERBOUHA, H. Ezzikouri, and M. Erritali (July 2014). "A comparative study of decision tree ID3 and C4.5." In: *(IJACSA) International Journal of Advanced Computer Science and Applications* Special Issue on Advances in Vehicular Ad Hoc Networking and Applications. doi: 10.14569/SpecialIssue.2014.040203.

Huo, Z., A. Pakbin, X. Chen, N. Hnoteey, Y. Yuan, X. Qian, Z. Wang, S. Huang, and B. Mortazavi (Mar. 2020). *Uncertainty Quantification for Deep Context-Aware Mobile Activity Recognition and Unknown Context Discovery*.

Indulska, J. and P. Sutton (2003). "Location Management in Pervasive Systems." In: *Proceedings of the Australasian Information Security Workshop Conference on ACSW Frontiers 2003 - Volume 21*. ACSW Frontiers '03. http://dl.acm.org/citation.cfm?id=827987.828003. Australian Computer Society, Inc.: Adelaide, Australia, pp. 143–151. isbn: 1-920682-00-7.

Kim, Y. and K. Lee (2006). "A Quality Measurement Method of Context Information in Ubiquitous Environments." In: *2006 International Conference on Hybrid Information Technology*. Vol. 2, pp. 576–581. doi: 10.1109/ICHIT.2006.253664.

Kittur, A., E. H. Chi, and B. Suh (2008). "Crowdsourcing User Studies with Mechanical Turk." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '08. ACM: Florence, Italy, pp. 453–456. isbn: 978-1-60558-011-1. doi: 10.1145/1357054.1357127.

Knuth, D. E. (1968). "Semantics of Context-Free Languages." In: *In Mathematical Systems Theory*, pp. 127–145.

Kwon, H., G. D. Abowd, and T. Plötz (2019). "Handling Annotation Uncertainty in Human Activity Recognition." In: *Proceedings of the 23rd International Symposium on Wearable Computers*. ISWC '19. https://doi.org/10.1145/3341163.3347744. Association for Computing Machinery: London, United Kingdom, pp. 109–117. isbn: 9781450368704. doi: 10.1145/3341163.3347744.

Lim, B. Y. and A. K. Dey (2011). "Investigating Intelligibility for Uncertain Context-aware Applications." In: *Proceedings of the 13th International Conference on Ubiquitous Computing*. UbiComp '11. ACM: Beijing, China, pp. 415–424. isbn: 978-1-4503-0630-0. doi: 10.1145/2030112.2030168.

Lima Cunha, E. de (2020). "Context Analysis for the Improvment of Intelligent Vehicules Networks." MSc dissertation. Master's thesis. Braga, Portugal: Minho University.

Liu, A., Y. Su, W. Nie, and M. Kankanhalli (2017). "Hierarchical Clustering Multi-Task Learning for Joint Human Action Grouping and Recognition." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39(1), pp. 102–114. issn: 0162-8828. doi: 10.1109/TPAMI.2016.2537337.

Liu, C., K. Chang, J. J. Chen, and S. Hung (2010). "Ontology-Based Context Representation and Reasoning Using OWL and SWRL." In: *8th Annual Conference on Communication Networks and Services Research, CNSR 2010, 11-14 May 2010, Montreal, Canada*. https://doi.org/10.1109/CNSR.2010.22, pp. 215–220. doi: 10.1109/CNSR.2010.22.

Liu, H., A. Nayak, and I. Stojmenović (2009). "Fault-Tolerant Algorithms/Protocols in Wireless Sensor Networks." In: *Guide to Wireless Sensor Networks*. Ed. by S. C. Misra, I. Woungang, and S. Misra. Springer London: London, pp. 261–291. isbn: 978-1-84882-218-4. doi: 10.1007/978-1-84882-218-4_10.

Liu, L., S. Wang, Y. Peng, Z. Huang, M. Liu, and B. Hu (2016). "Mining intricate temporal rules for recognizing complex activities of daily living under uncertainty." In: *Pattern Recognition* 60, pp. 1015 –1028. issn: 0031-3203. doi: https://doi.org/10.1016/j.patcog.2016.07.024.

Long, S., R. Kooper, G. D. Abowd, and C. G. Atkeson (1996). "Rapid Prototyping of Mobile Context-aware Applications: The Cyberguide Case Study." In: *Proceedings of the 2Nd Annual International Conference on Mobile Computing and Networking*. MobiCom '96. http://doi.acm.org/10.1145/236387.236412. ACM: Rye, New York, USA, pp. 97–107. isbn: 0-89791-872-X. doi: 10.1145/236387.236412.

Lyons, P., A. Cong, H. Steinhauer, S. Marsland, J. Dietrich, and H. Guesgen (Jan. 2010). "Exploring the responsibilities of single-inhabitant Smart Homes with Use Cases." In: *JAISE* 2, pp. 211–232. doi: 10.3233/AIS-2010-0076.

Manzoor, A., H.-L. Truong, and S. Dustdar (2008). "On the Evaluation of Quality of Context." In: *Smart Sensing and Context: Third European Conference, EuroSSC 2008, Zurich, Switzerland, October 29-31, 2008. Proceedings*. Ed. by D. Roggen, C. Lombriser, G. Tröster, G. Kortuem, and P. Havinga. Springer Berlin Heidelberg: Berlin, Heidelberg, pp. 140–153. isbn: 978-3-540-88793-5. doi: 10.1007/978-3-540-88793-5_11.

Martín, D., C. Lamsfus, and A. Alzua. *Automatic context data life cycle management framework*. Undetermined. doi: 10.1109/ICPCA.2010.5704122.

Michalakis, K., Y. Christodoulou, G. Caridakis, Y. Voutos, and P. Mylonas (2021). "A Context-Aware Middleware for Context Modeling and Reasoning: A Case-Study in Smart Cultural Spaces." In: *Applied Sciences* 11(13). issn: 2076-3417. url: https://www.mdpi.com/2076-3417/11/13/5770.

Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill: New York. isbn: 978-0-07-042807-2.

Mokhtar, S. B., A. Kaul, N. Georgantas, and V. Issarny (2006). "Efficient Semantic Service Discovery in Pervasive Computing Environments." In: *Proceedings of the ACM/IFIP/USENIX 2006 International Conference on Middleware*. Middleware '06. http://dl.acm.org/citation.cfm?id=1515984.1516003. Springer-Verlag New York, Inc.: Melbourne, Australia, pp. 240–259.

Mori, G., F. Paterno, and C. Santoro (2002). "CTTE: support for developing and analyzing task models for interactive system design." In: *IEEE Transactions on Software Engineering* 28(8), pp. 797–813. issn: 2326-3881. doi: 10.1109/TSE.2002.1027801.

Musumba, G. and H. Nyongesa (May 2013). "Context awareness in mobile computing: A review." In: 2.

Noor, M. H. M., Z. Salcic, and K. I.-K. Wang (Dec. 2016). "Enhancing Ontological Reasoning with Uncertainty Handling for Activity Recognition." In: *Know.-Based Syst.* 114(C), pp. 47–60. issn: 0950-7051. doi: 10.1016/j.knosys.2016.09.028.

Novais, P. and D. Carneiro (2016). "The role of non-intrusive approaches in the development of people-aware systems." In: *Progress in AI* 5(3). https://doi.org/10.1007/s13748-016-0085-1, pp. 215–220. doi: 10.1007/s13748-016-0085-1.

Nugroho, L. E. (2015). "Context-awareness: Connecting computing with its environment." In: *2015 2nd International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE)*, pp. 3–7. doi: 10.1109/ICITACEE.2015.7437760.

Oliveira, T. J. M. (Apr. 2017). "Clinical Decision Support: Knowledge Representation and Uncertainty Management." Doctoral dissertation. R. da Universidade, 4710-057, Braga - Campus de Gualtar: University of Minho - Doctoral Programme of Biomedical Engineering.

Ordónez, F. J., P. De Toledo, and A. Sanchis (2013). "Activity Recognition Using Hybrid Generative/Discriminative Models on Home Environments Using Binary Sensors." In: *Sensors* 13(5), pp. 5460–5477. issn: 1424-8220. doi: 10.3390/s130505460.

Panayotov, D., A. Grief, B. J. Merrill, P. W. Humrickhouse, J. T. Murgatroyd, S. Owen, and C. Saunders (2018). "Uncertainties identification and initial evaluation in the accident analyses of fusion breeder blankets." In: *Fusion Engineering and Design* 136. Special Issue: Proceedings of the 13th International Symposium on Fusion Nuclear Technology (ISFNT-13), pp. 993 –999. issn: 0920-3796. doi: https://doi.org/10.1016/j.fusengdes.2018.04.053.

Perera, C., A. B. Zaslavsky, P. Christen, and D. Georgakopoulos (2014). "Context Aware Computing for The Internet of Things: A Survey." In: *IEEE Communications Surveys and Tutorials* 16(1), pp. 414–454. doi: 10.1109/SURV.2013.042313.00197.

Pietschmann, S., A. Mitschick, R. Winkler, and K. Meißner (2008). "CroCo: Ontology-Based, Cross-Application Context Management." In: *Third International Workshop on Semantic Media Adaptation and Personalization, SMAP 2008, Prague, Czech Republic, December 15-16, 2008*, pp. 88–93. doi: 10.1109/SMAP.2008.10.

Pradeep, P., S. Krishnamoorthy, R. K. Pathinarupothi, and A. V. Vasilakos (2021). "Leveraging context-awareness for Internet of Things ecosystem: Representation, organization, and management of context." In: *Computer Communications* 177, pp. 33–50. issn: 0140-3664. doi: https://doi.org/10.

1016/j.comcom.2021.06.004. url: https://www.sciencedirect.com/science/article/pii/S0140366421002280.

Quinlan, J. R. (Mar. 1986). "Induction of Decision Trees." In: *Mach. Learn.* 1(1), pp. 81–106. issn: 0885-6125. doi: 10.1023/A:1022643204877.

Quinlan, J. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann.

Ramos, C., J. C. Augusto, and D. Shapiro (2008). "Ambient Intelligence - the Next Step for Artificial Intelligence." In: *IEEE Intelligent Systems* 23(2). https://doi.org/10.1109/MIS.2008.19, pp. 15–18. doi: 10.1109/MIS.2008.19.

Ramparany, F., R. Poortinga, M. Stikic, J. Schmalenstroeer, and T. Prante (2007). "An open context information management infrastructure the IST-amigo project." In: *3rd IET International Conference on Intelligent Environments (IE 2007)*. https://groups.uni-paderborn.de/nt/pubs/2007/RaPoStScPr07.pdf, pp. 398–403.

Regan, H. M., M. Colyvan, and M. A. Burgman (2002). "A TAXONOMY AND TREATMENT OF UNCERTAINTY FOR ECOLOGY AND CONSERVATION BIOLOGY." In: *Ecological Applications* 12(2), pp. 618–628. doi: 10.1890/1051-0761(2002)012[0618:ATATOU]2.0.CO;2.

Rukzio, E., J. Hamard, C. Noda, and A. De Luca (2006). "Visualization of Uncertainty in Context Aware Mobile Applications." In: *Proceedings of the 8th Conference on Human-computer Interaction with Mobile Devices and Services*. MobileHCI '06. http://doi.acm.org/10.1145/1152215.1152267. ACM: Helsinki, Finland, pp. 247–250. isbn: 1-59593-390-5. doi: 10.1145/1152215.1152267.

Sarker, I., A. Colman, J. Han, A. Khan, B Yoosef Abushark, and K. Salah (Nov. 2019). "BehavDT: A Behavioral Decision Tree Learning to Build User-Centric Context-Aware Predictive Model." In: *Mobile Networks and Applications*. doi: 10.1007/s11036-019-01443-z.

Sarker, I. H., Y. B. Abushark, and A. I. Khan (2020). "ContextPCA: Predicting Context-Aware Smartphone Apps Usage Based On Machine Learning Techniques." In: *Symmetry* 12(4), p. 499. issn: 2073-8994. doi: 10.3390/sym12040499.

Satoh, K. (Sept. 2005). "Speculative Computation and Abduction for an Autonomous Agent*." In: *IEICE - Trans. Inf. Syst.* E88-D(9). https://doi.org/10.1093/ietisy/e88-d.9.2031, pp. 2031–2038. issn: 0916-8532. doi: 10.1093/ietisy/e88-d.9.2031.

Schilit, B. N., N. Adams, and R. Want (1994). "Context-Aware Computing Applications." In: *IN PROCEEDINGS OF THE WORKSHOP ON MOBILE COMPUTING SYSTEMS AND APPLICATIONS*. IEEE Computer Society, pp. 85–90.

Senge, R., S. Bösner, K. Dembczynski, J. Haasenritter, O. Hirsch, N. Donner-Banzhoff, and E. Hüllermeier (2014). "Reliable classification: Learning classifiers that distinguish aleatoric and epistemic uncertainty." In: *Inf. Sci.* 255. https://doi.org/10.1016/j.ins.2013.07.030, pp. 16–29. doi: 10.1016/j.ins.2013.07.030.

Sheikh, K., M. Wegdam, and M. van Sinderen (2008). "Quality-of-Context and its use for Protecting Privacy in Context Aware Systems." In: *JSW* 3(3). https://doi.org/10.4304/jsw.3.3.83-93, pp. 83–93. doi: 10.4304/jsw.3.3.83-93.

Silveira, D. T. and F. C. Córdova (2009). “A Pesquisa Científica.” In: *Métodos de pesquisa*. Ed. by T. E. Gerhardt and T. S. Silveira. Coordenado pela Universidade Aberta do Brasil – UAB/UFRGS e pelo Curso de Graduação Tecnológica – Planejamento e Gestão para o Desenvolvimento Rural da SEAD/UFRGS. Editora da UFRGS: Porto Alegre, pp. 31–42.

Slonneger, K. and B. Kurtz (1995). *Formal Syntax and Semantics of Programming Languages: A Laboratory Based Approach*. 1st. Addison-Wesley Longman Publishing Co., Inc.: Boston, MA, USA. isbn: 0201656973.

Smola, A. and S. V. N. Vishwanathan (2008). *Introduction to Machine Learning*. 1nd. Cambridge University press. isbn: 0521825830.

Strang, T. and C. Linnhoff-Popien (2004). “A Context Modeling Survey.” In: *In: Workshop on Advanced Context Modelling, Reasoning and Management, UbiComp 2004 - The Sixth International Conference on Ubiquitous Computing, Nottingham/England*.

Tian, W., Y. Heo, P. de Wilde, Z. Li, D. Yan, C. S. Park, X. Feng, and G. Augenbroe (2018). “A review of uncertainty analysis in building energy assessment.” In: *Renewable and Sustainable Energy Reviews* 93, pp. 285 –301. issn: 1364-0321. doi: 10.1016/j.rser.2018.05.029.

Uusitalo, L., A. Lehikoinen, I. Helle, and K. Myrberg (Jan. 2015). “An Overview of Methods to Evaluate Uncertainty of Deterministic Models in Decision Support.” In: *Environ. Model. Softw.* 63(C). http://dx.doi.org/10.1016/j.envsoft.2014.09.017, pp. 24–31. issn: 1364-8152. doi: 10.1016/j.envsoft.2014.09.017.

Vanrompay, Y., M. Kirsch-Pinheiro, and Y. Berbers (2009). “Context-Aware Service Selection with Uncertain Context Information.” In: *ECEASST* 19. http://journal.ub.tu-berlin.de/index.php/eceasst/article/view/244.

Villanueva, D., I. González-Carrasco, J. L. L. Cuadrado, and N. Lado (2016). “SMORE: Towards a semantic modeling for knowledge representation on social media.” In: *Sci. Comput. Program.* 121. https://doi.org/10.1016/j.scico.2015.06.008, pp. 16–33. doi: 10.1016/j.scico.2015.06.008.

Weiser, M. (Oct. 1993). “Ubiquitous Computing.” In: *Computer* 26(10). https://doi.org/10.1109/2.237456, pp. 71–72. issn: 0018-9162. doi: 10.1109/2.237456.

Weiser, M. and J. S. Brown (1997). “The Coming Age of Calm Technology.” In: *Beyond Calculation: The Next Fifty Years of Computing*. Springer New York: New York, NY, pp. 75–85. isbn: 978-1-4612-0685-9. doi: 10.1007/978-1-4612-0685-9_6.

Xu, C. and S.-C. Cheung (Feb. 15, 2006). “Inconsistency detection and resolution for context-aware middleware support.” In: *ESEC/SIGSOFT FSE*. Ed. by M. Wermelinger and H. Gall. ACM, pp. 336–345. isbn: 1-59593-014-0.

Yan, Z., C. Liu, V. Niemi, and G. Yu (2010). “Effects of Displaying Trust Information on Mobile Application Usage.” In: *Autonomic and Trusted Computing: 7th International Conference, ATC 2010, Xi’an, China, October 26-29, 2010. Proceedings*. Ed. by B. Xie, J. Branke, S. M. Sadjadi, D. Zhang, and X. Zhou. Springer Berlin Heidelberg: Berlin, Heidelberg, pp. 107–121. isbn: 978-3-642-16576-4. doi: 10.1007/978-3-642-16576-4_8.

Yanwei, S., Z. Guangzhou, and P. Haitao (2011). “Research on the context model of intelligent interaction system in the Internet of Things.” In: *2011 IEEE International Symposium on IT in Medicine and Education*. Vol. 2, pp. 379–382. doi: 10.1109/ITiME.2011.6132129.