



---

# Reasoning with User's Preferences in Ambient Assisted Living Environments

---

FINAL THESIS

*Author name:* Chimezie Leonard Oguego

*Author no.:* M00385993

FACULTY OF SCIENCE AND TECHNOLOGY.

A THESIS SUBMITTED TO MIDDLESEX UNIVERSITY IN PARTIAL  
FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF DOCTOR OF  
PHILOSOPHY.

SUBMITTED ON NOVEMBER 6, 2020

# Contents

<b>Acknowledgement</b>	<b>vii</b>
<b>Abstract</b>	<b>viii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Problem statement . . . . .	4
1.2 Research aim and objectives . . . . .	6
1.3 Motivation . . . . .	7
1.4 Case Studies . . . . .	8
1.4.1 Light management case study . . . . .	8
1.4.2 Healthy eating case study . . . . .	9
1.4.3 Key research contributions . . . . .	10
<b>2 Literature Review</b>	<b>11</b>
2.1 Overview . . . . .	11
2.2 Preferences . . . . .	13
2.2.1 Preference in Classical in AI . . . . .	15
2.2.2 Related systems and approach . . . . .	22
2.2.3 Research Context . . . . .	29
2.3 Conflicts in preferences . . . . .	31
2.3.1 Identifying conflict and potential conflict . . . . .	31
<b>3 Methodology</b>	<b>33</b>
3.1 Temporal Argumentation . . . . .	33
3.1.1 Lighting management case illustrated using Argumentation . . . . .	40
3.1.2 Argumentation in AI . . . . .	51
3.1.3 Preferences in AI . . . . .	52
3.2 Users preference architecture for argumentation . . . . .	52
3.3 Modelling different users . . . . .	55

3.3.1	Modelling Sara's preferences . . . . .	55
3.3.2	Modelling Joe's Preferences . . . . .	63
<b>4</b>	<b>Research work and validation</b>	<b>66</b>
4.1	Hybrid System (A system for real-time Decision Making) . . . . .	67
4.1.1	Reasoning System (MReasoner) . . . . .	68
4.1.2	Specification File . . . . .	69
4.2	Argumentation Language Translation . . . . .	71
4.2.1	Modelling Argumentation Language to Implementation Language ( $\mathcal{L}^T$ to $M$ ) . . . . .	72
4.2.2	Modelling Sample (Light Scenario) . . . . .	73
4.3	Smart-Home Infrastructure . . . . .	76
4.3.1	Smart Spaces Lab . . . . .	77
4.3.2	Equipment . . . . .	78
4.4	Preference Management . . . . .	79
4.4.1	Preference Management Tool (Interface) . . . . .	79
4.4.2	Managing Users' Preferences . . . . .	80
4.4.3	Using Preference Interface To Affect System Behaviour. . . . .	82
4.4.4	System Specification for MReasoner . . . . .	83
4.5	System Illustrations (Demos) . . . . .	86
4.5.1	Hybrid System Illustration . . . . .	86
4.5.2	Solving conflicts using three preference criteria (Specificity, User Preferences and Persistency). . . . .	91
4.5.3	Supermarket Chain Store (Tesco) API . . . . .	96
<b>5</b>	<b>Discussion</b>	<b>101</b>
<b>6</b>	<b>Conclusion and Future Work</b>	<b>106</b>

# List of Tables

1.1	Input-output to the smart lighting system . . . . .	5
1.2	Summary of Scenarios . . . . .	9
2.1	Table summarizing the pros and cons of preferences in classical AI . . . . .	30
3.1	Interval-Interval relations (where X and Y represent two intervals). [3]	36
3.2	Dynamics evolution of the Light Case Scenario as regards time . . . . .	41
3.3	Knowledge Representation for First Scenario $\neg$ <i>LightsOn</i> and <i>LightsOn</i> . . . . .	42
3.4	Knowledge Representation for Second Scenario $\neg$ <i>LightsOn</i> and <i>LightsOn</i> . . . . .	44
3.5	Knowledge Representation for Third Scenario $\neg$ <i>LightsOn</i> and <i>LightsOn</i> . . . . .	47
3.6	Comparison of Classical Preferences in AI and Argumentation . . . . .	50
3.7	Lighting Scenario World Dynamics . . . . .	57
3.8	Television Scenario World Dynamics . . . . .	59
3.9	Health Scenario World Dynamics . . . . .	61
3.10	Joe's Health Scenario World Dynamics . . . . .	63
4.1	Sara Lighting Scenario Dynamics . . . . .	74
4.2	Converting Argumentation Rules to Implemented System Rules. . . . .	76

# List of Figures

1.1	Main interactions among user, system and real world affecting the dynamics of preference. [18]	7
2.1	CP-net for Light Choice: Bulb and Room.	16
2.2	A Strict, Complete, Binary, Acyclic CP-net.	17
2.3	Illustrations for Example TCP-Nets	19
2.4	The Imaging We service QoS preferences example using LCP-nets.	21
2.5	Procedural Reasoning System's Architecture, [48].	23
2.6	MonA abstract system architecture, [72].	24
2.7	Argumentative multi-agent architecture for AAL systems, [66].	26
2.8	AALFI customisation interface for profile requirement, [65]	26
2.9	AALF spoken dialogue customisation, [65].	26
2.10	The virtual butler sub system, [38]	27
2.11	Case-based conflict resolution system architecture, [79].	28
2.12	Lamp is on when the user is at the reading table.	32
2.13	Lamp is off when the user is not at the reading table.	32
3.1	First Scenario Argument A Tree for <i>LightsOn</i> .	44
3.2	First Scenario Argument B Tree for $\neg$ <i>LightsOn</i> .	45
3.3	Second Scenario Argument A Tree for <i>LightsOn</i>	46
3.4	Second Scenario Argument B Tree for $\neg$ <i>LightsOn</i> .	46
3.5	Third Scenario Argument A Tree for <i>LightsOn</i> .	48
3.6	Third Scenario Argument B Tree for $\neg$ <i>LightsOn</i> .	49
3.7	Overall preference Architecture	53
3.8	Ranking of Sara's Preferences	56
3.9	Argumentation Trees for Sara's Light Scenario	58
3.10	Argumentation Trees for Sara's Television Scenario	60
3.11	Argumentation Trees for Sara's Health Scenario	62
3.12	Ranking of Joe's Preferences	63

3.13	Argumentation Tree for Joe’s Health Scenario . . . . .	64
4.1	Hybrid Interface . . . . .	68
4.2	Reasoning System Interface. . . . .	69
4.3	Specification File Format Sample . . . . .	70
4.4	Argumentation Trees for Sara’s Light scenario. . . . .	75
4.5	Layout of the Smart Spaces Lab. . . . .	77
4.6	Living-Room of the Smart Spaces Lab. . . . .	78
4.7	Bedroom of the Smart Spaces Lab. . . . .	78
4.8	Smart devices and equipment for experiments. . . . .	78
4.9	Simple setting up of new users’ Preference (Bobby). . . . .	82
4.10	Retrieving and modifying existing preferences (Bobby). . . . .	82
4.11	Reasoning system (MReasoner) screen-shot with system specification details, to illustrate change in response based on preference ranking.	85
4.12	Overall architecture of preference management system . . . . .	86
4.13	Modeled Bobby’s situation of keeping the light “on” . . . . .	86
4.14	Browsing to select Specification File . . . . .	87
4.15	The MReasoner button is enabled as ‘NO’ potential conflict is detected	87
4.16	The Argumentation Solver button is enabled as potential conflict is detected . . . . .	88
4.17	Hybrid System highlighting the columns where conflict was detected and solved . . . . .	89
4.18	Interface showing that the user prioritised <b>Comfort</b> over <b>Light</b> . . .	90
4.19	Using Sara’s preference ranking to solve bedroom light conflict . . .	90
4.20	Database showing how the conflict was solved using the preference criterion, “ <b>User Preferences</b> ” . . . . .	91
4.21	Argument tree for Corridor-Light “on” or “off”. . . . .	93
4.22	Argument for BedRoom-Light. . . . .	93
4.23	Argument for ShowerRoom-Light. . . . .	93
4.24	Identified Potential Conflicts for Sara . . . . .	94
4.25	Hybrid System showing all three detected and solved of conflicts; “Specificity”, “Preference” and “Persistency” . . . . .	95
4.26	Database records of the three types of conflicts; “Specificity”, “User Preferences” and “Persistency” . . . . .	96
4.27	Requesting for Tesco URL to search for Cake Product . . . . .	98
4.28	System Advice Sara Not to buy cake since her “Health” has higher priority over “Pleasure” . . . . .	99

4.29 Database showing some of the 50 filtered “**Cake**” products, with last column indicating the product with “**Sugar**” or “**No Sugar**” . . . . 100

# Acknowledgement

*Firstly, I dedicate this achievement to my loving parents, Mr. Basil Emefionachuwku Oguego and Mrs. Grace Anulinwa Oguego. Thank you for all the hard-work and sacrifice made towards me and also in pushing me to achieve this dream. I have always wanted to make my a contribution in the Computer Science community, you both made this dream a reality. However, the achievement for this Ph.D goes beyond the produces work, as my problem solving technique and way of reasoning in the state of art and in general, has forever improved immensely.*

*I would also like use this opportunity to express my sincere gratitude to my supervisory team, especially Professor Juan Carlos Augusto for his continuous patience, motivation, full support (academically and non-academically) and whom his immense knowledge in the research area played a key factor in completing the research. Also big thank you to Dr. Mark Springett and Dr. Andrés Munõz Ortega for their guidance, advise, patience, support and mentorship. I could not have imagined a better supervisory team, am grateful.*

*To my colleagues and lab-mates I encountered during this process, especially Mario, Gines, and Murad whom I had the utmost privileged to share thoughts, ideas, experience and worked closely with, THANK YOU. Our team efforts in projects, how we handle challenges and deliver results, is something I will not forget, especially how we worked collaboratively and tirelessly towards winning the 2019 British Computer Society award, it was one of the highlights of my Ph.D research.*

*To my siblings, friends and acquaintances, am grateful. I cannot express my gratitude enough for the unconditional support, advice and courage expressed towards me, especially during times I was lacking motivation, I truly appreciate.*

*Most importantly, am forever grateful to God, whom without, this dream would have just been a dream.*



# Abstract

Understanding the importance of preference management in ambient intelligent environments is key to providing systems that are better prepared to meet users' expectations. Preferences are fundamental in decision making, so it is an essential element in developing systems that guides the choices of the users. These choices can be decided through argument(s) which are known to have various strengths, as one argument can rely on more certain or vital information than the other. The analysis of survey conducted on preferences handling techniques in Artificial Intelligence (AmI), indicates that most of existing techniques lack the ability to handle ambiguity and/or the evolution of preferences over time. Further investigation identified argumentation technique as a feasible solution to complement existing work.

Argumentation provides a means to deal with inconsistent knowledge and we explored its potentials to handle conflicting users preferences by applying to it several real world scenarios. The exploration demonstrates the usefulness of argumentation in handling conflicting preferences and inconsistencies, and provides effective ways to manage, reason and represents user's preferences. Using argumentation technique, this research provide a practical implementation of a system to manage conflicting situations, along with a simple interface that aids the flow of preferences from users to the system, so as to provide services that are better aligned with the users' behaviour. This thesis also describes the functionalities of the implemented system, and illustrates the functions by solving some of the complexities in users' preferences in a real smart home. The system detects potential conflict(s), and solves them using a redefined precedence order among some preference criteria.

The research further show how the implemented Hybrid System is capable of interacting with external source's data. The system was used to access and filter live data (groceries products) of a UK supermarket chain store, through their application programming interface (API), and advise users on their eating habits, based on their set preference(s).

# Chapter 1

## Introduction

Most decisions humans make are based on choice(s), even refraining from choosing is a choice. Preferences guide our choices, so it is important to understand the various aspects of preference handling in order to develop a system that supports users' decisions or acts on behalf of users [31], especially in an intelligent environment[12]. One key factor in designing a successful ambient intelligence (AmI) system is balancing users' preferences [51, 55]. This is particularly important in ambient assisted living (AAL) [19].

AmI refers to the environments which are responsive and sensitive to people's presence, allowing humans to interact in the physical space in an intelligent and unobtrusive manner. One sub area of AmI is AAL, which consist of products, concepts and services, and combines the social environment and new technologies in order to improve quality of life.

AAL systems rely on sensing technology deployed in physical space to gather real time contextual information, which the system uses in decision making to benefit the users of that space. On a daily basis we enter sensorised spaces such as cars and homes and bring sensors with us in our smartphones. Examples of current wireless sensors are passive infrared sensors (PIR) which track movement within a room, and pressure sensors which sense whether someone is in bed or sitting in a chair. There are sensors which allow for the control of lights, knowing when they are 'on' or 'off' and actuators to turn them 'on' or 'off'. There is now a wide range of devices, including wearables, which provide data about an individual's vital signs, e.g. blood pressure and glucose levels, and this information is available in digital form. Also important is information gathered from the outside world, for example, public transport timetables, doctors' appointments, offers or details of products from on-line stores, all of which may allow a system to support a human's life in a practical way. However, these systems cannot handle users' preferences in a dynamic

way, which is the focus of this research. For a system to be effective in supporting a user's needs, it needs to know about user expectations. The research aims to understand how to enhance user benefits from AAL technology through effective handling of preferences. Due to the impact of AAL on human lives [56, 19] these systems require complex problem solving and intelligent decision making capabilities. When a system is expected to act on behalf of humans, it needs to understand and respond to the preferences of users and should have the ability to resolve conflicting preferences. Preferences present a number of complexities. They may change over time, clash or conflict, or be modified by experience. For example, watching movies or listening to music may make us change our opinion about a product and we may decide to consume more or less of it. Preferences can even be imposed to some extent, such as a lifestyle adjustment requested by a doctor or insurance company, e.g. the need to take medicines [18].

These changes are what the produced solution is able to handle, as the system observes changes in the user's behaviour and adapts to those changes. The system receives input from the user and various other sources (e.g. sensors and internet services), and, if it needs to provide feedback or have help in making decisions, some real-time mechanism is required to keep the system updated and react appropriately. This research identifies the preference handling techniques that exist, and investigation (discussed in Section 2.2.1) shows that the classical preferences of AI are not capable of handling real time problems concerning user preferences. However, a feasible method has been identified during the investigation process, as the contributions of other studies indicate the usefulness of the argumentation technique.

Additional findings identify other relevant systems proposed in the state of the art, that adopt the argumentation technique. For example, [5] formalises a problem of multiple criteria decision making within a logical argumentation system, designing logical machinery that directly manipulates arguments with their strengths and returns preferred decisions, enabling users to compute with justification preferred decision choices. Following the same line of research, an argumentation framework is presented by [14, 16, 95], to reason about qualitative interest-based preferences. The same authors present a further argumentation-based framework [96] to model and automate reasoning using multi-attribute preferences of a qualitative nature, showing how to reason about preferences when information is incomplete or uncertain. A perspective on practical reasoning is proposed by [9] as probable justification for a course of action. This is based on an argumentation scheme that supports the decision making processes in multi-agent systems. Collaborative research con-

ducted by a computer scientist and a psychologist [28], presents seven procedures to help choose from among options represented as a bipolar set of arguments, with their evaluation ranked according to their importance. The authors of [100] employ multi-attribute decision theory, and introduce several argumentation schemes, in order to provide an agent that makes the best decision based on preferences over outcome. However, these studies are unable to manage preferences over time. Our findings and experience in the development of AAL systems enable us to conclude that argumentation is a technique that has advantages that the classical preferences in AI do not. Argumentation is basically concerned with the exchange of proposals and their justification [34]. These sets of arguments may come either from dialogue between several agents or from available pieces of information (which may be contradictory) at the disposal of one unique agent. Argumentation develops as a reasoning process [92] that can help make decisions by handling the conflicting situations expressed within a discussion among participants (or agents) with different goals. During the 1980s, argumentation started to attract attention within Computer Science (CS) as a branch of AI focused on ways of representing the processes humans follow when using common sense reasoning, taking into account the influx of new information [37, 26]. Time is an important factor in various areas of CS, AI [15] and, in particular, AmI [21, 68].

Conflict can occur in preferences, for example the desire to keep the bedroom light ‘off’ while asleep, can conflict with the need for the light to be ‘on’, for safety reasons. A conclusion has to be reached, and that conclusion needs to be decided depending on what the user prefers. A conclusion can change if a new reason or fact becomes available. Knowledge of new facts can lead to the preferring of a new conclusion, reliance on a previous conclusion, or a consideration that the previous conclusion is no longer correct. When new information becomes available, it might provide a better reason to stay with the previous conclusion or a new reason to come to a different conclusion. Providing a system with the ability to react in such a manner, balancing user preferences, is key to designing a successful AAL system.

The research explore the potential of argumentation for handling inconsistent knowledge and conflicts in our previous work [78], also discussed in Chapter 3 of this thesis. In the rest of this chapter, we examine some of the problems that exist in currently available systems and outlined other specific problem. This chapter also highlight the research aim and questions, discuss the motivation behind the research, and describe some case studies adopted in validating the practical system.

## 1.1 Problem statement

Modern sensors can respond to human movement [63]. For example, some systems, typically used in offices, turn lights ‘off’ when there has been no movement for some time. However, this is unhelpful if a person is absorbed in reading when suddenly the lights go ‘off’, breaking their concentration and forcing them to wave their arms to turn the lights back ‘on’. Conversely, as soon as movement is detected the system turns the lights ‘on’, which is fine for an office but not for a bedroom, as moving during the night would cause the lights to go ‘on’ and ‘off’ intermittently.

There are two problems with this type of system which the present work addresses. The first is that office systems are set in such a way that, while not impossible to change, modifying the waiting time is usually beyond typical users’ capabilities. The other is that the system’s notion of context is very limited. The only context it recognises is time without movement. Research this study conducted into these systems, aims to provide easy ways for users to personalise the behaviour of the system through parameters which represent their preferences. The parameters which facilitate this personalisation, depend on the technology available in the given environment. This research keeps the system functionality, technology and type of personalisation simple, but hope to demonstrate that the proposed system is more intelligent and capable of detecting whether the person is sleeping and whether the lights should be turned ‘on’ or ‘off’ in a sensible and flexible way.

There are other specific problems/limitations identified during the course of this research (labelled from L-1 to L-4), they include:

- L-1 The need for a simple interface for AmI, which should allow users to interact with their environment easily and with fewer clicks, and also represent effectively user’s preferences.
- L-2 Existing systems limit users’ preferences and needs with the system in some way. Intelligent Environment community operates under strong user-centred principles [12] which secure humans rights over the system and reassures the human to be in control of the system and not the other way around [40].
- L-3 Existing smart home systems are either hard-wired [79] or do not provide the flexibility of user’s involvement [57].
- L-4 The lack of intelligent environment systems that have the ability to handle inconsistencies, solve conflicting preferences and deal with time related challenges in AmI.

These specific related problems were aimed to be addressed (labelled A-1 to A-4) in the following ways:

- A-1 Provide an effective way to represent users' preference by producing an interface, which users utilise to prioritize their preferences.
- A-2 The interface allow users to create, access, and modify their preference ranking, which will affect the system behaviour automatically, in other-words giving users some level of freedom. For example, the office light stays 'on' until the user says otherwise by using the interface.
- A-3 The interface should enable multiple users' profiles to co-exist in the same system, which will allow the flexibility of multiple users to set different settings of how they want the system to function.
- A-4 Using temporal reasoning approach, the research aim to provide a system that is intelligent to handle inconsistencies, deal with time related problems and solve conflicting user's preferences in an intelligent environment.

A system such as the one initially described can be created with current technology based on wireless sensors [63]. For example, the movement within a room can be perceived by a system using passive infrared sensors (PIRs) which measure spatial variations of heat. These sensors are commonly used in domestic alarm systems as a way to detect the movement of intruders. The type of system this research aims to provide enables the user to perform their usual activities of moving around, getting in or out of bed, etc. without turning the lights 'on' or 'off'. Users can also set up preferences which affect the way the system reacts, for example how long the system waits to turn the lights 'off' when there is no movement. A similar system is used in [20], although that system is centred on measuring quality of sleep and detecting dangerous situations that threaten the safety of the user. The system does not allow for preference personalisation.

Table 1.1: Input-output to the smart lighting system

	<b>Input</b>				<b>Output</b>
	<b>Sensors</b>		<b>Human</b>		<b>Actuator</b>
<b>Type</b>	Pressure Pad	PIR	Bob	Bob	Light Bulb
<b>Values</b>	on-off	on-off	actions	preferences	on-off dimmed

Table 1.1 summarises the main parameters of the environment which can be perceived by the system and feed the context-awareness module, along with the main ways the system can act upon the environment.

## 1.2 Research aim and objectives

The research aim of developing an effective system to reason and represent user preferences, along with a flexible interface that has the ability to change its behaviour according to changes made by the user has been provided by this research. The research also provides some practical demonstration process of how the system functions, validating its effectiveness. One of the validation requires smart home infrastructure and equipment such as sensors, actuators and so on. The validation conducted in a smart space within Middlesex University premises is discussed and illustrated in Chapter 4, however there was no user involvement in the validation process due to some limitation in using the smart space lab. The preference management tool is used to automate and effect system behaviour in the smart home, and the lighting scenario was used to illustrate how the home reacts to changes in user preferences.

Prior to reaching this goal, there were some specific challenges dealt with during the research, for example gaining a deeper understanding of how preferences can be applied in intelligent environments, and solving the problem of dealing with potential conflictive situations created by opposing preferences.

Reflecting the importance of ambient intelligent systems being able to handle user preferences, the research explores the possibility of providing a reasoning system to handle inconsistencies and conflicting user's preferences. The research also aim to provide a simple interface which users will to interact with the system, but there are few objectives that the research aims to answer in providing the solution:

1. How can one or more partial order of user preferences be most effectively represented in intelligent environment?
2. Can the proposed intelligent system be friendly enough to handle user preferences and needs, and be dynamic enough to assume changes over time?
3. How flexible is the interface provided? Can the interface be used with ease?
4. Can the system change its behaviour according to changes made by the user?

Addressing these research questions will lead to providing a suitable AAL system to manage user's preferences in a smart home. Nonetheless, there is needs to

deliberate on the motivation behind undertaking this research.

### 1.3 Motivation

A key mission of AmI is to enhance the way individuals interact with their environment, promote safety and enrich their lives [17]. AmI systems are meant to act proactively to anticipate preferences, in order to support users in making decisions [22]. Users should be empowered to personalise systems according to their preferences and this should be reasonably easy to do [12].

Preference handling can naturally lead to conflict, such as when we have feelings or desires that conflict with what needs to be done (such as a diabetic patient tempted to take up an offer to buy their favourite chocolate cake). These needs can be resolved if the system has the ability to understand such situations and present solutions to users which are perceived as natural. In addition, these preferences change with time, such as temperature preferences change with the seasons or lighting preferences during day and night.

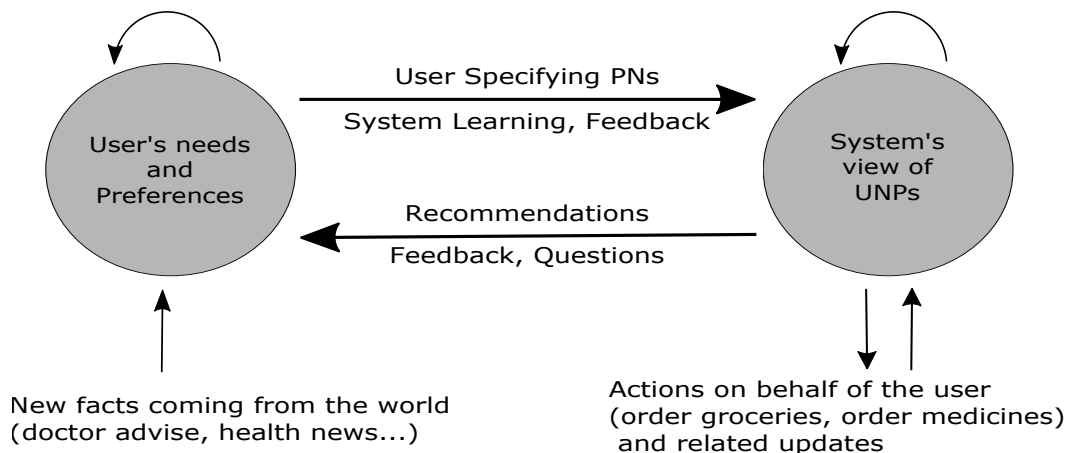


Figure 1.1: Main interactions among user, system and real world affecting the dynamics of preference. [18]

Motivated by earlier reflection on the importance of preferences and the challenge they pose technically [18], (see Figure 1.1), the research make first attempt to find a specific way to manage this concept in our survey paper [77].

Given that the initial survey [77] focuses on finding the most suitable approach to handling preferences, the analysis were simplify as much as possible to illustrate some important points. In doing so the research focus on managing the preferences of one user. Various works focus on one user and so this study follow this line, leaving as further work the consideration of more than one user [66]. The next



section discusses some informal scenarios used throughout the thesis for illustration and demonstration on the produced system.

## 1.4 Case Studies

The first scenario is about a light management system capable of understanding the activities in a smart home, and make decisions for the user (Bob). The other is for another user (Sara), who needs the system to be aware of her health situation, and provide appropriate results.

### 1.4.1 Light management case study

*Bob is a 65 year old man who lives alone and loves reading at night. He usually falls asleep during reading, leaving the lights on. Bob does not have any problem sleeping with the lights on, but knows that it increases the electricity bills and can lead to other risks, such as electrical issues, which he does not want.*

This description implies specifiable preferences such as stating how long he wants the light to be on when the system detects that he is asleep. The idea is to have a system intelligent enough to understand and react to significant changes. Three scenarios are created based on the above description, to illustrate and compare possible solutions to this type of situation.

- **Scenario 1 (Bob comes home and prepares to go to bed):** *The light can be on until the system detects that the user is asleep, and then it turns the light off after some time specified by the user.*
- **Scenario 2 (Bob wakes up in the middle of the night):** *The user is asleep (light should be off at this point), then if the user wakes up (e.g. to use the toilet, etc.), the light should come on. If the user goes back to sleep, the light goes off after some time (e.g. 10 minutes).*
- **Scenario 3 (Bob leaves home):** *The user wakes up from sleep and the lights come on. Then the user leaves home (e.g. to go to work). The system should turn the lights off after some time, if the user forgets to switch them off before leaving home.*

Table 1.2 summarises the highlights of the scenario above, with added sample times associated with its main stages.

Table 1.2: Summary of Scenarios

Scenarios	Times	Significant Developments
Scenario 1	10 p.m.	Bob enters the room and the light comes on (system detects movement and detects that it's dark outside)
	11 p.m.	Bob goes to bed (light goes off after some time (e.g. 10 minutes), if no movement is detected and pressure is detected on the bed sensor)
Scenario 2	2 a.m.	Bob wakes up in the middle of the night to use the toilet
	2.05 a.m.	Bob goes back to bed (lights go off after detecting that Bob is asleep).
Scenario 3	7 a.m.	Bob wakes up in the morning (lights come on gradually when movement is detected and Bob is out of bed).
	8 a.m.	Bob leaves home for work (light goes off automatically after a specified time, if Bob forgets to switch off the lights).

### 1.4.2 Healthy eating case study

*Sara wants the system to be aware of her health circumstances, and provide her with information on food consumption. Since she is diabetic, she wants to know the sugar content of her food, especially her favourite grocery, cake, which she usually buys from her local chain store in the UK (known as Tesco).*

The above scenarios has be used in various ways in this thesis. The light management case description in section 1.4.1 which three scenarios were produced, was theoretically explored in section 3.1.1, showing the dynamic evolution of light case as regards time. The research carefully explored each of the light scenario demonstrating the basic idea behind each argument, which was also illustrated in a smart home. This was published in first article (see i, in section 6). The healthy eating case study was applied in section 4.5.3, where the developed system was used to query external data and advised the user accordingly. This was also illustrated in a current article (see i, in section 6) which is currently under review.

The scenarios are used to evaluate the system within and outside a smart home, so as to demonstrates its abilities in managing conflicting user preferences. However, before demonstrating the evaluations, it is appropriate to discuss some theoretical

aspects of the system implemented.

### 1.4.3 Key research contributions

- We extended a previous many-sorted system with a new preference sort ( $\mathcal{P}ref$ ) [78] which we use to specify *User's Preferences* and defined as part of preference criterion used to establish the preference of an argument over another.
- Adapted the argumentation technique (along with the introduced sort,  $\mathcal{P}ref$ ) to extend an existing monotonic reasoning system, into a system that has the ability to identify inconsistencies and solve conflicting situations. The system was referred to as Hybrid System, which retained the monotonic reasoner and if required by the context allowed non-monotonic reasoning as well.
- Produced a preference management tool for the  $\mathcal{P}ref$  sort, which consist of an interface that users work with to indicate and rank their preferences.
- Demonstration of the developed Hybrid System was conducted within an intelligent environment, as various scenarios were applied to validate its correctness. The different scenarios depicts how the intelligent environment was able to quickly align its behaviour with the change of preferences from the user. The system demonstrations with the scenarios can be found in Chapter 4 of this thesis.

## Chapter 2

# Literature Review

This chapter introduces the literature review and survey of the state of the art. First this thesis gave an overview, then discuss the importance of preferences. Overall concept, definition and importance of preferences in general were provide, before discussing preferences in classical AI. The thesis highlights and explains some of the known classical preference management techniques related to the solution produced in this research, as the aim is to analyse them and understand their limitations as they relate to our produced system. Related approaches are discussed, along with a table summary of the pros and cons of the classical AI techniques.

This chapter also briefly emphasises the conflicts in preferences, as the study discuss how our practical solution is able to identify conflicts (which needs to be solved) and potential conflicts (which do not need to be solved as the event occurrence happens at intervals). The explanation is supported by a short illustration.

### 2.1 Overview

During the course of the study, with the aim of tackling the existing problem by providing a practical solution for representing and reasoning with the user's preference over time. Extensive research has been conducted into the various ways of understanding the problem in order to come up with a suitable method to tackle it, and temporal argumentation was identified as a suitable solution to handle the existing problem.

The formalization of temporal reasoning has proven to be a great task as the literature in philosophy, logic, AI, linguistics and Computer Science attest [16]. Several ways to represent and use temporal knowledge have been suggested since it was considered as a subject of study in artificial intelligence in the late 70's [14].

The Temporal Language ( $\mathcal{L}^T$ ) allows representation of time, properties, events

and actions, which have been considered in AI literature as key concepts to model a rational agent in a dynamic world.  $\mathcal{L}^T$  allows association of knowledge to either “instant” and “interval” which are used to express time in the real world. Properties express the state of the world under observation in an “instant” or “interval” in the real-world, while events are occurrences in the real-world (for example, sensor triggering), that can have effect on a certain situation. Actions will be ascribed to humans, whom acting on their free-will, perform actions that typically causes some events to occur, which in turn potentially change properties of the real-world.

This research provides a practical solution, applying these key concepts and interval logics. The practical solution also consists of the introduce preference sort ( $\mathcal{P}ref$ ), along with the argumentation technique we initially identified as a feasible solution to handle conflicting preferences and deal with time related problems.

In recent years, argumentation received momentous interest in the multi-agent system (MAS) community, as it provide ways to allow an agent to reconcile conflicting information from multiple agents through communication; conflicting information within itself; and its information state with new perception of the environment [80].

Argumentation contributes to two main sort of problems encountered in MAS. First in focusing on (non-monotonic) reasoning over uncertain or incomplete information, where arguments for and against certain conclusion (goal, belief etc.) are constructed and compared. Secondly, on interaction among agents, allowing the exchange of arguments to justify a stance and provide to defend claims [80].

Previous research ([13]) presented a theoretical exploration of the argumentation system (without the preference sort  $\mathcal{P}ref$ ), that allows temporal reasoning using the notions of “instant” and “interval”, which was one of the well-known logics for temporal reasoning presented in [3].

This thesis emphasised in section 3.1, providing more details, to the key concepts and fundamental logics around the argumentation technique, and presented some theoretically solutions where some real-world scenario were applied to argumentation along with the introduce preference sort ( $\mathcal{P}ref$ ). The thesis also provides discussions on other classical preference handling techniques that exist in AI (Section 2.2.1), and provided appropriate illustrations related to smart environments to explore the abilities of these methods to handle the complexities in users’ preferences.

Background reading and understanding is conducted as part of the research process and various findings surface, which include identifying other problems and the shortcomings of the research area. However, the study focuses on the problem of inconsistencies and handling of user preferences over time, which is a problem in

the state of the art. The research explore various ways to resolve the problem, which include researching existing models and identifying a suitable preference handling method.

The idea is to understand these models and test whether they are suitable to achieve the aim of the research. However, during the course of the research, the findings indicate that these methods are not suitable to handle conflicting situations or represent users' preferences over time. The research provide an illustration of the classical preference techniques identified, using the lighting scenario, and summarise the pros and cons of the handling techniques. Our illustration shows that, of the identified methods, a conditional preference network (CP-Nets) is the better handling technique for representing user preferences, however it is not good enough for handling conflicting preferences. A study from [41] adds that, among the various formalisms proposed in their literature to represent preferences, the most promising is CP-nets, which is a prominent qualitative approach for representing user preferences. The clear graphical structure of the model gives an easy representation of the user's desires with nice computational properties for computing the best outcome [41].

Current work in preference modelling and decision theory aims to create compact preference models, achieving a good compromise between two conflicting aspects, on one hand, "the need for sufficiently flexible models to describe sophisticated decision behaviour", and on the other, "the practical necessity of keeping the elicitation effort at an admissible level as well as the need for efficient procedures to solve preference-based optimisation problems".

## 2.2 Preferences

Intelligent systems as studied and developed in artificial intelligence either support or assist humans in the world or act in the world to accomplish tasks like humans agents [50]. The system must choose from various means of expression or actions in order to act autonomously. For systems to intelligently support human actions, they have to understand and respond to the choices of humans. To make decisions in the desired way, agents/systems need a policy, whether pre-computed or not, for choosing. The policy may take into account the short or long term effects of the choices, but it requires a way of comparing and evaluating these effects. Preferences achieve this and are crucial for systems to make decisions in a rational and desirable way.

Preferences are, for assistants, key to understanding and supporting the deci-

sions of human users. They are fundamental to decision making and have been extensively studied in various disciplines [50]. There are still various shortcomings that exist in AAL systems, related to the difficulty of systems to easily capture and manage something as essential as the user's preferences and needs. Classical preference models are utility functions which map possible outcomes of the decisions to numerical values allowing the sorting and comparison of these outcomes([51]), making it difficult manage easily the preferences of users over time.

Artificial intelligence (AI) brings new application fields to these classical preference models. Preferences are essential in recommender systems, multi-agent systems, configuration and design, planning and scheduling, and other tasks concerning autonomous decision making or intelligent decision support [50]. However, the cross-fertilisation between preference handling and AI also goes in the other direction. Existing AI methods for knowledge representation and problem solving have led to new preference handling methods. There are questions that need to be asked about preferences in AI. The primary ones identified by [50] are:

- What kinds of preference model are of interest?
- How are they represented?
- How are the preferences obtained?
- How can they be used in reasoning?
- How can we actually compute with them?

These questions arise for applications of preferences to AI problems, and also for new preferences handling methods developed in AI [51]. This research further investigated some existing preferences handling methods in a previous study [77], and discussed some of the methods related to the practical solution provided in this research.

Preference handling is one of the core issues in the design of any system that automates and supports decision making [31]. There have been various preference handling techniques proposed in artificial intelligence (e.g. CP-Net, UCP-Net, etc.) that address preference recommendation and preference-based representation problems. These techniques are to some extent useful in expressing users' preference and they have been implemented in various ways. However, they lack certain core aspects, such as not having the ability to reason and represent users' preference over time and not being able to handle inconsistencies. The limitations were illustrated in a previous study ([77]) using an AmI system case study that deals with the automatic control of lights.

Preferences are not only significant in making decisions for users in AmI, but also vital in understanding and supporting decisions made by users [51]. Evidence from [31] illustrates how preferences guide the choices of the user, and how preferences have a number of complexities that clash or produce conflicts. For example, listening to the radio or watching movies might change the user's opinion about a product, and make the user want more or less of the product.

Various preference handling models have been proposed in Artificial Intelligence (AI) to address preference recommendation problems. These techniques are not well equipped to reason and represent changes in users' preferences over time, nor do they deal with inconsistent preferences. Section 2.2.1 of this thesis emphasised in details some of the various classical handling preference technique in AI, and these techniques in AI have been investigated because they closely relate to the problem this research aim to addressed.

### 2.2.1 Preference in Classical in AI

Preferences guide the choices of the user. So understanding several aspects of preference handling is important both for supporting active user control and designing systems that act on behalf of users. Preference is known as a core issue in the design of automated systems that aims to support the decision making of the users. It is therefore crucial to understand preference handling and the tools needed to help develop a system that can handle inconsistencies and deal with time.

One of the main aims of this section is to address some existing classical preferences in AI and then investigate their ability to deal with conflicting situations and represent user's preference over time. These classical AI models will be discussed and analysed from the perspective of a smart environment (applied light scenarios), so as to assess whether they are suitable for addressing the problem described. The classical preferences techniques include:

- CP-nets
- UCP-nets
- TCP-nets
- LCP-nets

Note that these are not all the preference handling techniques that exist in AI. However, the research focus on these because they relate closely to the solution provided by this study.



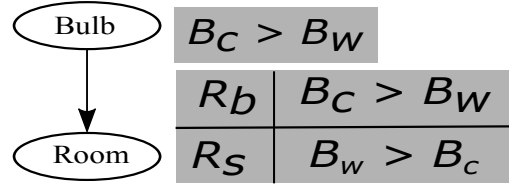


Figure 2.1: CP-net for Light Choice: Bulb and Room.

### Conditional Preference Networks (CP-Nets)

CP-net is known to be the most prominent qualitative approach for presenting preferences. Its clear graphical structure unifies an easy representation of user desires with cordial computational properties when computing the best outcome [41].

CP-nets is a directed graph representation of conditional preferences, where nodes represent variables and edges express preference links between variables. CP-nets exploits the power of conditional ceteris paribus rules [4] which enables a compact representation of human preferences. CP-net is naturally suited to simple applications (e.g. recommender systems to buy books on the web) in which preferences can easily be approximated by lexicographic rules on attributes with small domains [52]. It represents a complex preference over objects, using a set of atomic preferences each of which is a preference over a single object attributes given that the values of the other attributes are equal (the ceteris paribus principles). For example: “*Bob prefers  $X = x_1$  to  $X = x_2$* ”.

An example of how CP-net expresses preferences could be that of light choice. Figure 2.1 expresses preference of light choice in a house. This network consists of two variables B and R, standing for Bulb and Room respectively. A user might prefer coloured Bulb ( $B_c$ ) to white Bulb ( $B_w$ ), and their preference of whether the user wants the white bulb or a coloured one, could be conditioned based on the sitting room ( $R_s$ ) or the bedroom ( $R_b$ ): Bob prefers coloured bulb than white bulb in his bedroom and the white bulb in his sitting room to the coloured one.

According to [30], “tools for representing and reasoning about Ceteris paribus preferences are important because they should aid in elicitation process for naive users”.

Various studies of CP-nets are restricted to preferences that are strict, binary, known and complete [4]. This means that all the features which an outcomes depends on are known. For instance, an individual who lives alone may prefer the light to be off during the day and may wants the light on at night as long as it is not their bed time (which can vary for users). These are strict preferences because this is a user who works during the day and sleeps at night. An example of a complete, strict

and acyclic CP-net is illustrated in Figure 2.2. The diagram illustrates a user (e.g. student) who prefers to have the light on when she studies at night and off when she studies during the day.

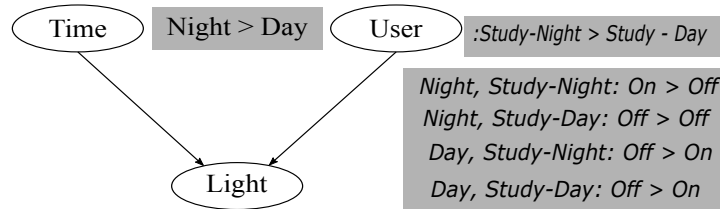


Figure 2.2: A Strict, Complete, Binary, Acyclic CP-net.

However, when the user's preference is unknown, especially given that user preferences do change more often, a method that has the ability to handle the change over time and resolve conflicting situations will be needed, and these capabilities are not present in CP-net.

Formally, a preference relation is a partial pre-order on a set of alternatives (or outcomes)  $O$ . The expression  $O > P$  means that  $O$  is preferred to  $P$ . If neither outcome is preferred to the other, they are said to be incomparable. CP-nets have been developed for such problems, rather than to compare alternatives in bits, as decision makers consider how the preference over one feature depends on the values of the other in the decision domain.

Let us consider the example of a student who lives alone and studies every night to prepare for an up-coming exam. She falls asleep almost every night without turning the lights off. This means that she falls asleep any time during studying the night, so it is unknown when the student actually falls asleep. It will be difficult to represent this using CP-net of *ceteris paribus* statements as the time when the student falls asleep is unknown.

As shown above, CP-net has not advanced in a sufficient way for widespread use in complex, real world engineering applications [4] like AAL systems this research produced. Considering this, using CP-nets to represent the preferences of a user over time to help in dealing with conflicting situations will not be feasible.

### Utility Conditional Preference Networks (UCP-Nets)

This model was proposed by [29] in 2001 by combining the appealing aspects of two existing preference models, namely: GAI (a graphical model used to represent and manage independences among attributes [52]) and CP-nets. UCP-nets can be viewed as an extension of the CP-network that allows representation of qualitative

utility information rather than simple preference ordering. UCP-nets facilitate an incremental elicitation process, as they have a number of conceptual and computational advantages over GAI and CP-nets models, providing leverage with respect to interference and elicitation. The model is directed like CP-nets though preferences are quantified with utilities and by extending CP-nets with quantitative utility information. The expressive power is enhanced and dominance queries become computationally efficient. By introducing directionality and a *ceteris paribus* semantics to GAI, it allows utility functions to be expressed more naturally and optimization queries to be answered much effectively. Furthermore, this model allows for more powerful statements that are often more natural. This leads to more effective inference, and can be used in interactive elicitation processes in determining relevant parameters of UCP models in a specific decision scenario.

Despite identifying how UCP-nets have various conceptual and computation advantages over CP-nets and the GAI model, the authors emphasised in the concluding part of their study that “practical experience and empirical studies are needed to gauge the ultimate effectiveness of UCP-nets [29]. This model has not currently been applied to the type of problem (light scenario). In addition, one of the crucial problems faced in the use of a decision theoretical model is the elicitation of preference information [29]. This is one key motivation behind the development of the UCP-nets model. However, the problem this research aims to address goes beyond eliciting and representing qualitative utility information. The present research aims to resolve conflicts and represent users’ preference over time.

### **Tradeoffs-Enhanced Conditional Preference Networks (TCP-Nets)**

This is another extension of CP-nets that can be referred to as a *relative important* statement for conditional preference networks with trade-off [32]. It is a graph based representation that encodes statements of (conditional) preferential independence and (conditional) relative importance [35]. To better understand this, using our light scenario, a Bed-Room ( $B_R$ ) can consists of both a White-Bulb ( $W_B$ ) and a Coloured-Bulb ( $C_B$ ) (as values) and the Sitting-Room ( $S_R$ ) consists of White-Fluorescent ( $W_F$ ) and Coloured-Fluorescent ( $C_F$ ) lights. The user may prefer to read in the Bed-Room than the Sitting-Room and wants to use Brighter Light (BL), (knowing that Bed-Room and Sitting-Room are preferentially independent), then the preference order over Bed-Room can be specified as White Bulb  $>$  Coloured Bulb, independently of the value of the Sitting-Room. In a similar way preference values over the Sitting-Room (if the user wants to read in the Sitting-Room), will be of the form White-Fluorescent  $>$  Coloured-Fluorescent, independent of the values of

the Bed-Room. It can infer from this that  $W_B$  and  $W_F$  is a more preferred outcome than  $C_B$  and  $C_F$ .

The TCP-net model basically empowers users to express trade-offs, which they are willing to concede among various preference criteria. The idea of conditional relative importance complements the one of conditional ceteris paribus independence [32] so as to provide for a richer conceptual framework and reason about the user's preferences. Figure 2.3 illustrates how TCP-nets extends CP-net by adding an  $i$ -arc from  $(B_R)$  to  $(BL)$  and  $(S_R)$  to  $(BL)$  (which describes the relative importance from  $B_R$  to  $BL$  and  $S_R$  to  $BL$ ) and also  $ci$ -arc (which is for *conditional importance*) between  $(W_B)$  and  $(C_B)$  as well as  $(W_F)$  and  $(C_F)$ . The relative importance of  $(W_B)$  and  $(C_B)$  or  $(W_F)$  and  $(C_F)$  depends on the assignment to  $(B_R)$  and  $(BL)$  or  $(S_R)$  and  $(BL)$  respectively.

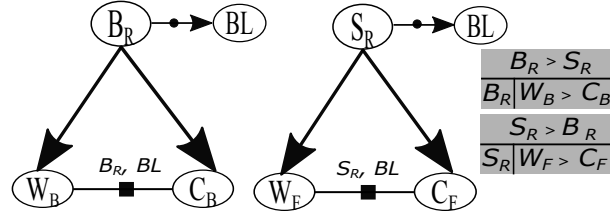


Figure 2.3: Illustrations for Example TCP-Nets

TCP-net has been used to propose a heuristic for estimating the preference ordering over the different choices at each stage in the composition to improve the efficiency of an algorithm (TCP-Compose\*) [87]. This algorithm was presented to generate a set of composite services that achieve the desired functionality and constitute a non-dominated set of solutions with respect to user specified preferences and trade-offs over non-functional attributes [87]. Given that preference elicitation can be a bottleneck in many applications, TCP-net was suggested [32] as an enhancement of CP-nets for structuring, representing and reasoning about quality preference statements. It helps to make an optimally desirable solution for users who lack the knowledge, time or expert support required to specify complex multi-attribute functions.

In other words, TCP-nets provide a richer framework for representing users preferences, allowing stronger conclusions to be drawn among two variables. However, this research aims for more, such as providing a solution to resolve conflict, as well as representing and reasoning with users' preference over time rather than trading-off a less preferred outcome among two attributes.

### Linguistic Conditional Preference Networks (LCP-Nets)

This model was proposed as a result of two important weaknesses spotted in \*CP-nets models (including extended ones) [36], in expressing preferences in a Quality of Service setting (QoS). QoS dimensions are defined on continuous domain and \*CP-nets only deal with finite domain variables. Using fuzzy linguistic terms [101], LCP-net was proposed to discretize continuous domains instead of crisp sets, so as to better capture user intentions, eliminating the need for the user to express preferences among values of a continuous domain. The other limitation is that, getting precise utility from non-speciality users is difficult, so giving numbers to express preferences is not always feasible. Current \*CP-net models provide two alternatives in this case. The original CP-nets model expresses preferences through a more simple and intuitive relation, although suffers from low performance when comparing two assignments. On the other hand, UCP-nets perform the comparison more efficiently, though it is harder to get precise numeric utility values.

This version of the CP-nets model (LCP) was developed to address the problem of expressing preferences, including non-functional properties. It provides programmers with an intuitive tool to express their preferences among services via their various qualities of services monitored at run-time. The advantage of fuzzy linguistic approaches in LCP-nets was acquired by combining UCP-nets and TCP-nets techniques, allowing preference modelling of more qualitative statements such as *I prefer the more or less V1 value for property X over exactly V2 if properties Y equals approximately VY and Z equals a bit more than VZ*. [35] expresses how LCP nets are easier to establish than writing several sets of fuzzy rules that can be interdependent but qualitative to deal with user or QoS sensor imprecision. They further stated that LCP-nets allow users to express trade-offs among variables using i-arcs from TCP-nets and have CPTs (conditional preference table) similar to that of UCP-nets, however they express utilities with linguistic terms rather than numeric values. With LCP-nets it is possible to:

- Reveal relative importance of non-functional properties;
- Elicit preferred assignment for specific QoS domains;
- Indicate trade-offs between non-functional properties.

Consider Figure 2.4 in which the user preference for having the lights on (such as for security purposes) is detailed. The main goal here is for the user to have the light on at night. The goal is translated into preferences according to three of its quality of service (QoS) properties, security ( $S$ ), bright-light ( $B$ ) and colour-bulb

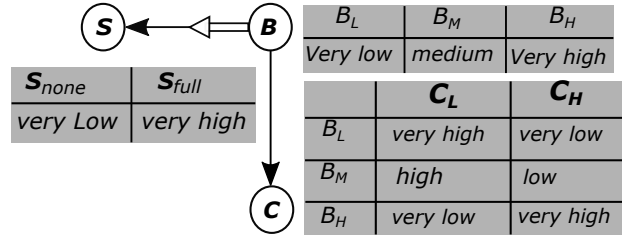


Figure 2.4: The Imaging We service QoS preferences example using LCP-nets.

( $C$ ). The user would always prefer bright-light over security, but if the light is low colour-bulb would be preferred so as to still have light at night.

In another study [36], the same authors who introduced the LCP-nets framework, apply the framework to multi-criteria decision making. This arises from runtime choice among candidate services and several unrelated QoS properties. This is done to select the best service from among a set of offers, given their dynamic non-functional properties. Generally, this new variant of CP-nets helps non-specialist programmers express preferences in a qualitative way through the values of the QoS properties in this multi-criteria decision making process. The process does not include resolving conflicting situations or dealing with time, both of which are crucial in developing a system that reasons with users to assist in making vital decisions. Furthermore, according to the conclusion of [36], one of the limitations of LCP-nets is that they do not have the flexibility to share common preferences among complex business process decision sites, which indicates that this method cannot address the complex scenario provided by this research.

Having explored these classical preference techniques in AI, our analysis found them suitable for simpler application. However, these classical preference techniques, CP-Nets for example, are less scalable and less helpful as the number of scenarios, preferences and triggers increase in a sensorized environment, and with that, the number of node combinations and probabilities need adding and revising.

Another aspect CP-Nets did not handle well was the temporal aspects of enabling conditions to support a decision, for example to turn ‘on’ and ‘off’ the light, some activities need to happen, in certain order and with some minimum or maximum duration. CP-Nets technique lacks the temporal logic representation, as the approach seems to assume as if everything happens simultaneously and instantaneously. Other system features we thought were better considered in the Argumentation Systems approach than in CP-Nets. For Argumentation Systems, the handling of inconsistencies was more explicitly addressed and flexibly processed based on the preference mechanisms and their hierarchies, which combined several high level assessments,

based on logical soundness of arguments then on (real-time current) preferences, and if necessary, on persistency. We do not rely on probabilities which have to be provided somehow and adjusted periodically if we want the CP-Net to remain current. We further provide a comparison of the classical preference technique in AI and argumentation technique, shown in Table 3.6.

There are other handling models (e.g. GAI network, Bayesian network etc.), referred to as utilities models, but these models are not explored in this research due not having the qualitative nature of representing users' preferences. However, qualitative models (e.g. CP-nets, UCP-nets etc.) which are naturally suited to simple application (such as a recommender system), are significantly outperformed by utilities models in addressing decision problems, due to their higher descriptive power [29].

The research also explore produce systems related to this study, so as to comprehend existing systems and identify their limitations. This guided the research to address the limitations of our practical solution. These systems were discussed in the next section.

### 2.2.2 Related systems and approach

Various investigations have been conducted in the area of ambient assisted living to provide suitable intelligent systems, and various developments have been introduced, or proposed, using a variety of related tools.

Procedural reasoning systems (PRSs) for instance [48], consist of a database which contains beliefs or facts about the world; a set of current goals that need to be realised; a set of plans known as knowledge areas which describe how certain sequences of actions and tests may be performed to achieve given goals or react to a particular situation; and, lastly, an intention structure containing those plans that have been chosen for execution.

The development of reasoning systems which can reason and plan in a continuously changing environment is emerging as an important area of artificial intelligence. An embedded procedural reasoning system (PRS) consist of these features which enable it to operate effectively in an intelligent environment. PRSs are designed to continue to function in an acceptable way as embedded reasoning systems in the absence of any decision knowledge. The nature and role of intention in the PRS is emphasised and the future discussed in terms of how responsive it is in real time and its ability to operate under a well-defined measure of reactivity.

The system is implemented on a Symbolic 3600 Series LISP machine and used to detect, and recover from, most possible malfunctions of the RCS , including

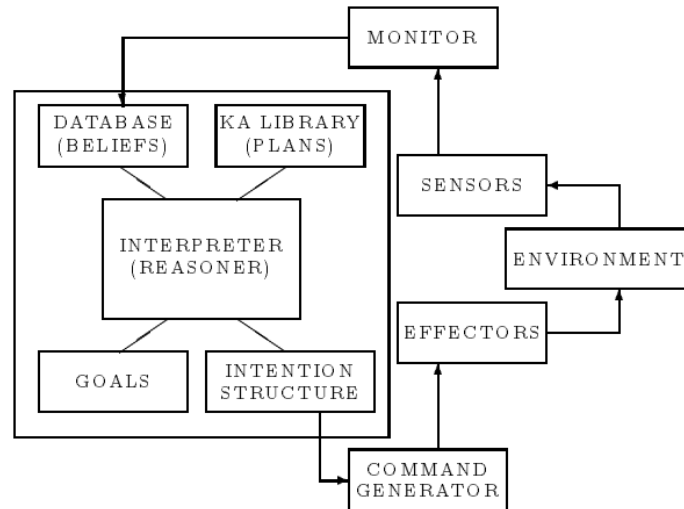


Figure 2.5: Procedural Reasoning System's Architecture, [48].

sensor faults, leaking components or regulators and jet failures. The experiment provides a severe and positive test of the system's ability to operate proficiently in real time, weigh alternative courses of action, coordinate its activities, and modify its intentions in response to a continuously changing environment.

In addition, the PRS meets the criteria for evaluating real-time reasoning systems: high performance, guaranteed response, temporal reasoning capabilities, support for asynchronous inputs, interrupt handling, continuous operation, handling of noisy (possibly inaccurate) data, and shift of focus of attention. The features of the PRS that, the creators believe, contribute most to its success at this task are:

- Its partial planning strategy
- Its reactivity
- Its use of procedural knowledge
- Its meta-level (reflective) capabilities.

In particular, the manner in which the system integrates its means-ends reasoning with the use of decision knowledge is considered an important component of rational activity.

Another research from [72] focuses on supporting people with special needs in their daily routines, which needs to be monitored continuously so as to assist them in an appropriate way. Revealing some of the shortcomings of IT based living solutions (such as not considering the current situation of the user), their project (BelAmI)



has developed a hybrid reasoner for realising monitoring and assistance services in assisted living environments. The hybrid reasoner constitutes the core of the monitoring and assistance component (MonA). The MonA has the ability to adapt planned and running treatments according to the current situation and context. This solution (project) basically renders living assistance at home in a sensitive and responsive way, as it involves the embedding of various ambient sensors into the environment or around the body.

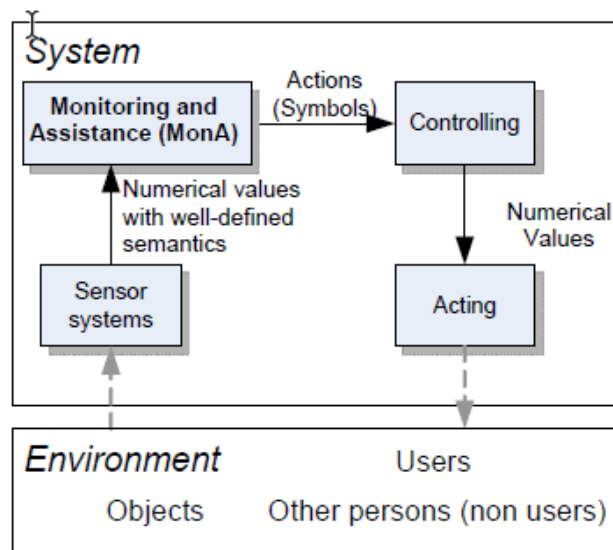


Figure 2.6: MonA abstract system architecture, [72].

The BELAmI project, which focuses on emergency assistance, is driven by some of the following scenarios:

- Monitored drinking (reminding the person to drink enough)
- Monitored eating (avoiding food poisoning by avoiding expired food)
- Fall detection (detecting falls and triggering a staged emergency reaction of the AAL system)
- Habit-observation (monitoring the person continuously for any unusual behaviour, and checking whether the person is ok by asking questions).

The MonA component receives numerical data as input and provides symbolic values for treatments or actions that needs to be executed by the controllers, such as establishing a communication link to relatives or friends. MonA consist of other functional components, one of which is the “*conflict handling*” aspect that adapts

the proposed treatment to the current context (monitored person). In this sense, the “*conflict handling*” component mainly prevents known conflicts.

Further research identifies another system that facilitates web service selection when dealing with incomplete or inconsistent user preferences [99]. The system explores the information on historical users to modify the active users’ preferences, improving the results for the selected services. Simulation certifies the efficiency and effectiveness of the technique in conflict removal. The approach uses a CP-net model, similar to LCP-nets, for the same reason, to provide QoS-based late-binding of service innovations, adding extra agility to business process execution [35]. However, at the time of this thesis, there is no evidence of the work having the ability to manage user preferences over time.

A study by [66] explains how sensors can provide an incomplete, occasionally ambiguous, picture of the world which often leads to inconsistent context and unreliability in the whole AAL system. It proposes a design for an AAL system that deals with inconsistency and ambiguous information by taking a qualitative approach based on multi-agent architecture, where each agent supports the context of the occupant’s point of view through argument. These arguments enable the development of a well-structured and sound reasoning process once inconsistent contexts are detected, and offer an alternative that is easier to validate and understand than a quantitative approach focused on complex models or intrinsic algorithms.

Another related piece of research [65], focuses on using a multi-agent system (MAS) to care for an elderly individual with assistance and support through an ambient assisted living flexible interface (AALFI). The study identifies and underlines several issues with ambient assisted living (AAL) systems that might result in the support and assistance provided being inappropriate or misunderstood by the subject. For example an AAL system cannot provide ways to tailor the support to the subject’s requirements, meaning the user may not understand the feedback. This leads to the development of the ambient assisted living flexible interface system which provides users with assistance and support tailored to their specific requirements. It ensures they are able to read messages and interact with the interface visually or by saying simple commands and listening to simple prompts when auditory interaction is being used. Basically, the system gives users the ability to control and interact with the AALFI system based on their requirement profile.

The flexibility displayed by AALFI personalises the subject interaction for an elderly person with changing requirements. The context aware characteristics displayed by MAS include the ability to select the correct interactions, and adapt the assistance to the current time, the activity carried out by the older person, the event

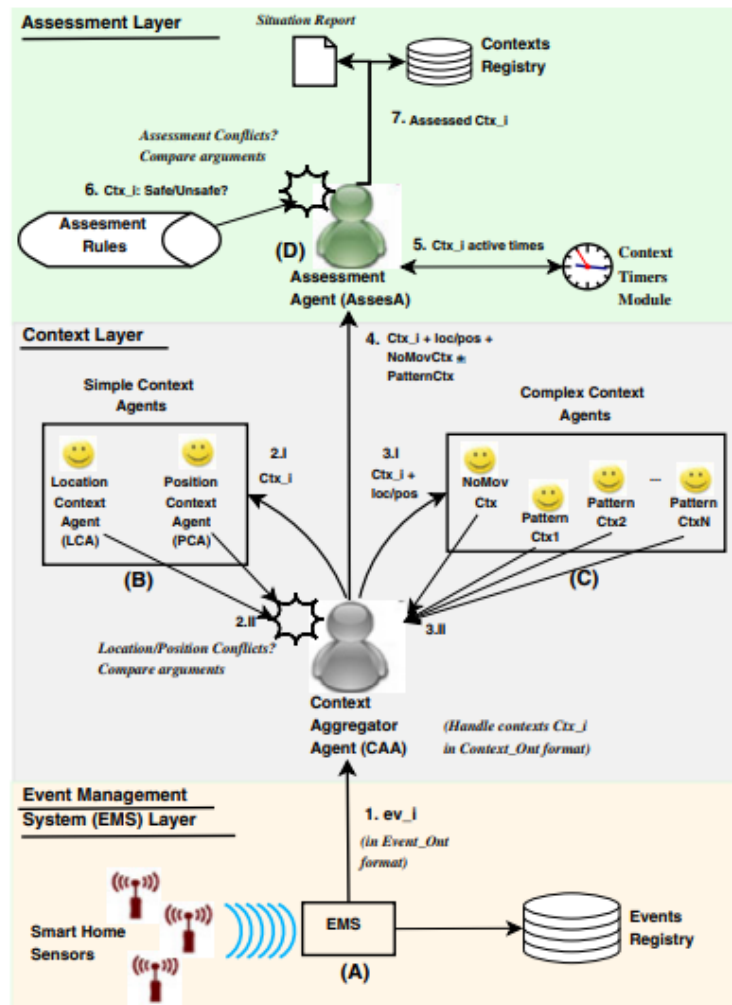


Figure 2.7: Argumentative multi-agent architecture for AAL systems, [66].



Figure 2.8: AALFI customisation interface for profile requirement, [65]

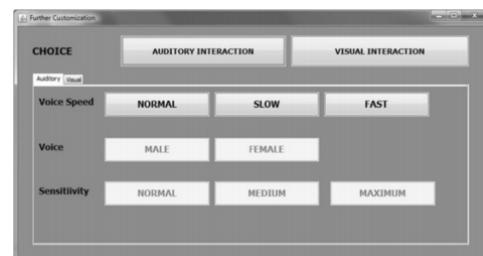


Figure 2.9: AALF spoken dialogue customisation, [65].

detected or the number of times an event has occurred, helping the AALFI provide

flexible interactions and assistance. This flexibility of AALFI provides the older person with accurate assistance based on their requirement profile.

In 2014, Costa [38] developed a mobile virtual butler (VB), which is believed to be an improved version of AALFI, that provides an interface between elderly people and smart home infrastructure. The VB is receptive to user questions and answers them in the context of knowledge. It is capable of interacting with the user when it senses something is wrong, notifying the next of kin, medical services etc. The VB is aware of the user's location and moves to the computing device closest to the user so as to always be present. The VB is a simple affordable voice interface that uses vocal commands which are correctly interpreted and executed. The VB can interact with the user through voice synthesis, either by responding to the user's command ("Turn TV on") or interacting with the user by checking if he/she is alright.

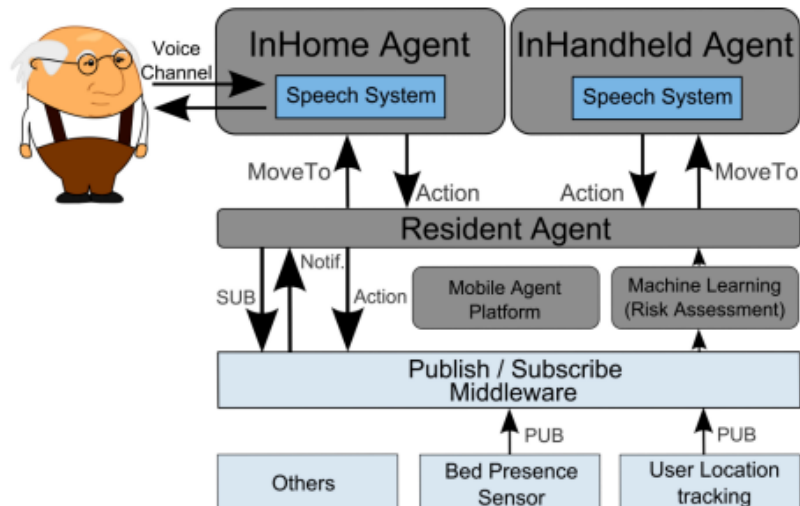


Figure 2.10: The virtual butler sub system, [38]

The VB aims to fulfil two major requirements:

- Location-awareness, behaving according to the user's location in the AISH
- Having the ability to fulfil all human machine interactions of the AISH set-up through computer-generated speech and voice recognition.

The VB device fulfils the first requirement based on the assessment and feedback gathered from interviews with volunteers, but no definite conclusion is drawn on the second requirement as the volunteers barely interacted with the VB. However, two valuable lessons are identified: firstly people regard the wireless microphones as strange if they have not used them before, and are uncomfortable using them; and

secondly, various issues impair the functionality of the seamless voice-based interface, such as it being a battery operated device with a low capacity that hinders the operability of the system.

These problems are listed in the research as future work, intended to be improved on by providing alternatives to wireless microphones that are more acceptable for users, not depending on batteries, and requiring little or no maintenance. But this does not grant the user the flexibility to manage their preferences, especially making decisions in conflicting circumstances.

A more recent system in the multi-user smart home environment was developed by [79], and is known as the virtual assistant. The main purpose of the research is the creation of a user friendly interface with an adaptive context-aware case-based conflict resolution system. A case-based reasoning approach provides a context-aware virtual assistant, and the main functionality of the system is giving multiple users the ability to control appliances by voice or text commands. However, the system does not have any form of reasoning or the ability to resolve conflicting situations based on user preferences. The system does not give users the ability to manage their preferences or control how the smart home reacts, as the house reactions are hard-coded by the engineer.

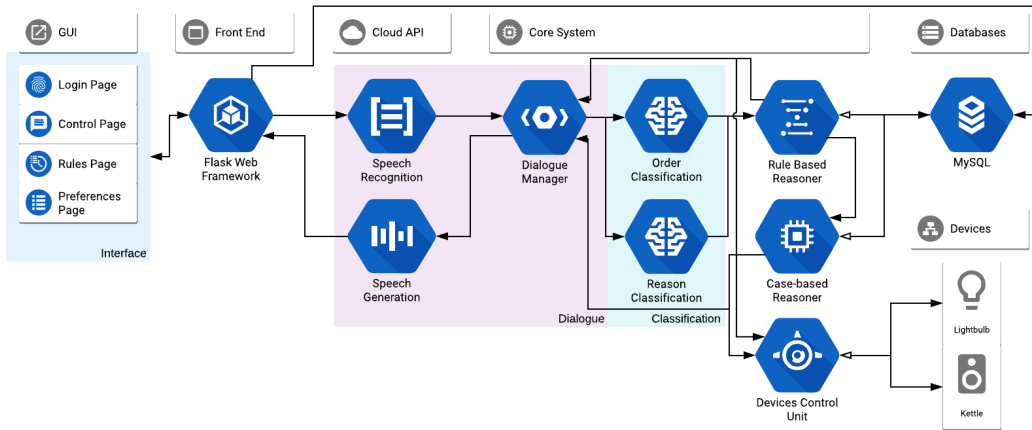


Figure 2.11: Case-based conflict resolution system architecture, [79].

These related works are not all the proposed or developed solutions that exist in the state of the art, however they are AAL solutions that aim to support users intelligently. This thesis investigate and discuss these systems sequentially by the year the work was carried out, indicating whether the works are in line with our practical system. This helps create insight into the limitations of existing applications, and enables us to provide not only an effective solution, but also a suitable system

that supports and enhances user benefits while addressing some of the limitations of exiting systems.

### 2.2.3 Research Context

This research aligns with the current works of the Research GrOup On Development of Intelligent EnvironmentS (GOODIES), at the department of Computer Science of Science of Technology, Middlesex University, which I am part of. The group focuses on areas such as Context Awareness, Smart Home, Ambient Assisted Living, Intelligent Environment, Smart Environment, Pervasive Computing, Ubiquitous Computing, Person-Centric Computing and Smart cities. The innovations developed by the group are guided by some principles Intelligent Environment application should follow, some of those principles can be found here: <http://ie.cs.mdx.ac.uk/explaining-our-work/>. The research group has several completed and ongoing projects. “Managing User’s preferences in Ambient Assisted Living” is one of our ongoing projects, which this research is about. Additional projects can be found here: <http://ie.cs.mdx.ac.uk/projects/>.

Table 2.1: Table summarizing the pros and cons of preferences in classical AI

AI Preference Models	Pros	Cons
CP-Nets: Conditional Preference Networks	<p>The promising approach for representing preferences in a qualitative and quantitative way is CP-nets [41].</p> <p>Offers a compact and arguably natural representation of preference information, necessary for solving many simple real world problems. Partial order can be created from small set of alternatives.</p> <p>Aids elicitation process for naive/non-expert. users</p>	<p>Consistency of cyclic CP-net is not guaranteed.</p> <p>CP-nets are restricted to preferences that are strict, complete and binary and the dependency graph are usually assumed to be acyclic. It will not be practical to create a partial order from large number of features. Does not allows for the comparison or the ordering of all its alternatives.</p>
UCP-Nets: Utility Conditional Preferences Networks	<p>Facilitates an incremental elicitation process.</p> <p>Has a number of conceptual and computational advantages over the CP-nets model, providing leverage as regards to inference and elicitation. Allows one to make more powerful statements that are often more natural and lead to more effective inferences.</p>	<p>Practical experience and empirical studies are needed as to gauge its effectiveness.</p> <p>To the best of our knowledge, there is no implementation of UCP-nets.</p>
TCP-Nets: Tradeoffs-enhanced Conditional Preference Network	<p>With the limitation in CP-nets that do not express preferences over the variables themselves, TCP-nets was introduced to represent relatively importance between variables.</p> <p>The model adds more important relations and conditional relative importance statement to the ceteris paribus statement.</p>	<p>There is no research work reporting on the implementation of TCP-net as a solver [102].</p> <p>To the best of our knowledge and that of [102], there is no implementation of TCP-nets. TCP-nets only deal with preferences (soft constraints) as hard constraints are not considered explicitly, which can be a real limitation when dealing with real life problems that include both constraints and preferences. The challenge of consistency of TCP-nets that is not conditionally acyclic.</p>
LCP-Nets: Linguistic Conditional Preference Networks	<p>This is a variant of CP-nets that has the same service ranking with all CP-nets extensions, however expressing CP-nets is easier with LCP-nets.</p> <p>Applies to select the best service among a set of offers. Indicates a trade-off between non-functional property and revealing relative important of non-functional property.</p>	<p>Focuses more on the mathematical modelling, allowing to aggregate the LCP-nets compared to CP-nets and TCP-nets that catch the eye due to their simplicity and expressiveness ([93]).</p>
Service Selection Framework	<p>This system was developed to utilize the information of historical users to enhance the preferences of the active users, improving the service selection results as the simulation results verified the effectiveness and efficiency in conflict removal [99].</p>	<p>Using CP-nets models, the approach tends to handle incomplete and inconsistent user preferences although it does not demonstrate the ability to handle users' preferences over time.</p>

## 2.3 Conflicts in preferences

Preferences can naturally lead to conflicts. Preferences also are sometimes in conflict with each other, as there may be reasons to keep the lights on and also reasons to keep them off. Processing and representing desires in terms of preferences, so that a more desirable choice precedes a less desirable one, is appealing as it allows one to specify desires in a declarative way and deal with inconsistencies and exceptions in a quite flexible manner [71]. However, to deliver a system that is intelligent enough to specify desires in a declarative and flexible way, while addressing conflicting situations, the system need to identify all the situations that potentially lead to conflict. The system should have the capability to differentiate actual conflict from potential conflict, and solve conflicts where necessary. The implemented system has the ability to detect all the possible conflict that might occur among arguments and, using the preferences criteria algorithm, the system addresses only the conflicts.

### 2.3.1 Identifying conflict and potential conflict

One key factor of the implemented system is the ability to identify potential conflicts first, before resolving the conflicts within the potential conflict. The system have been programmed to identify potential conflict first, by scanning the rules in the specification file. When two opposing consequences are detected, the system identifies a potential conflict which is the first step to identifying conflicts. An example of a potential conflict consisting of two opposing consequences could be:

```
ssr((#pressurepad) -> #lamp0n);  
ssr((pressurepad) -> lamp0n);
```

The above is detected as a potential conflict, because it is not a conflict, as it means that if the pressure pad (*pressurepad*) is active, it turns the lamp on. But if no pressure is detected on the pad (*#pressurepad*) the lamp is turned off.

Figure 2.13 illustrates that when the user is not sitting on the chair, the property state (pressure pad) remains inactive, thereby keeping the lights off.

Figure 2.12 illustrates that the lights are turned on when the user is on the chair, as this signifies that the pressure pad is active, activating the lamp. So the above rules are not a conflict but a potential conflict, as there is no clash and the lamp can either go off or on, depending on whether the user activates the pressure pad. This brief illustration shows how the implemented system differentiates potential conflict from actual conflict. More discussion of the specification file, conflicting rules and demonstration of the system can be found in Chapter 4. This thesis extensively discuss the methodological approach of this research in the next chapter.





Figure 2.12: Lamp is on when the user is at the reading table.



Figure 2.13: Lamp is off when the user is not at the reading table.

## Chapter 3

# Methodology

This chapter introduces the methodology adopted by this research for developing a practical solution. Analysis is conducted into the state of the art to identify exiting methods that relate to the solution provided, as the research aim to provide a system that is effective in responding to users. However, to ensure that the right approach was adopted, despite argumentation, which several pieces of literature mentioned as having the ability to handle conflicting preferences, the method was explored theoretically by applying it to several complex scenarios.

The thesis first introduce one element of argumentation, temporal argumentation, and show how the argumentation system is used, with example scenarios, to naturally capture the desirable features of ambient intelligence. This chapter uses the lighting management scenario in three ways to explore the potential of argumentation, and illustrates arguments for each, using appropriate argumentation trees. The chapter briefly discusses argumentation in AI and preferences in AI, before discussing the overall preference architecture used to compare two arguments. The research introduce a new preference sort (*Pref*) used to represent user preferences.

Various users are modelled in the last section of the chapter, classified into two groups, one that cares about their health and safety, and one that cares more about their pleasure and fun. The various scenarios used illustrate these modelled users, demonstrating different results for user's different user preferences.

### 3.1 Temporal Argumentation

Time is ubiquitous to any activity that requires intelligence, as some important notions, such as action, causality and change, are related to time [94]. Artificial intelligence is an area where the concepts of time and event are essential, as agents usually have to reason about a dynamic environment.

Section 2.2.1 provides a list of several theoretical methods which, to some extent, address the role of preferences in decision making. However, from the point of view of ambient intelligence there are further dimensions which are not explicitly addressed by those methods. Preferences are sometimes in conflict with each other. For example, sometimes there may be reasons to keep the lights on and reasons to turn them off. Time plays an important practical role, in particular preferences changing over time. For example, humans prefer different levels of lighting at night and day, and through different seasons they prefer different ambient temperatures. Computer science (CS) has long investigated both these features of handling conflicts and time in argumentation systems [68, 23, 67, 24]. This research believe time-based argumentation is an option worth exploring, offering advantages that the methods outlined in the previous section do not. This section introduce the basics of argumentation, in particular temporal argumentation. This thesis later show, with example scenarios, how desirable features in AmI are naturally captured by the argumentation system. CS has long investigated both conflicts and inconsistencies, and this constitutes an interesting feature of argumentation systems [23, 67, 24]. Time is also an important topic in various areas of CS and AI [14], in particular in AmI [21, 68].

Argumentation started to attract attention within CS during the 80's as a branch of AI focusing on finding ways to represent the processes humans follow when using common sense reasoning, particularly, taking into account exceptions and the way our conclusions adapt to the continuous influx of new information. Previously, Argumentation Systems appeared as an alternative to so-called 'non-monotonic reasoning', 'default reasoning' and 'defeasible reasoning' [37] [26].

The basic idea of argumentation is to create arguments in favour of and against a statement in order to determine if that statement can be acceptable or not and why [33]. Amongst other features argumentation offers a way to represent defeasible reasoning, characterizing the skill that allows us to reason about a changing world where available information is incomplete, or not very reliable. Argumentation systems have the ability to change conclusions in response to new information that comes to the system. The conclusions obtained by the system are "justified" through arguments supporting their consideration. In addition, an argument could be seen as a "defeasible proof" for a conclusion. The knowledge of new facts can lead to a change in preference, or to consider a previous inference no longer correct. In particular, there could exist an argument for a conclusion C and a "counter-argument", contradicting in some way the argument for C. An argument is a valid justification for a conclusion C if it is better than any other counter-argument for C. To establish

the preference of an argument over the others, a definition of preference criteria is required. Several preference methods are possible, and one of the more widely used is “specificity” [91], favouring more specific information, i.e. better informed arguments. It is important to highlight that Argumentation Systems emphasize the role of inference justification and the dialectical process related to reasoning activities.

Given the limitations noticed in the handling of preferences by state of the art systems, including both handling of inconsistency and time-related information, this research uses an Argumentation System which allows us to explicitly refer to time [13].

The system  $\mathbb{L}(\mathbb{T})$  presented in [13] is actually an extension of *MTDR*, a previous well-known argumentation framework [89]. The extension includes addition of a temporal language  $\mathcal{L}^{\mathbb{T}}$ . As mentioned in the previous chapter, this temporal language allows reification over time, properties, events and actions, which have been considered in the AI literature as key concepts to model a rational agent in a dynamic world. The system used to represent knowledge is based on a many-sorted logic [44], where different sorts are used to formalize the different concepts represented in the system. The fundamental building blocks such as time, properties, events and actions listed above are only examples of possible sorts. Others can be added depending on need. This has been done in Section 3.2.

The temporal language ( $\mathcal{L}^{\mathbb{T}}$ ) [13] allows association of knowledge to either “instants” ( $\mathcal{T}$ ) or “intervals” ( $\mathcal{I}$ ) so that we can express developments in real-world scenarios that happen (or are perceived to happen) instantaneously as well as developments requiring a non-atomic duration to complete. An example of an instant could be something that happened in a second in a system where seconds are the minimum time granularity, and an example of an interval will be a whole minute in that system. So if a Passive Infrared Sensor (PIR) is triggered only once in a second, e.g. at 17:06PM, then we can describe that as an instantaneous occurrence. If the same sensor is activated continuously for 15 minutes it can be said that the activation of the sensor lasted for a while and those 15 minutes will become an interval of time, e.g. from 17:06PM to 17:21PM. Familiar order relationships between units of time can be defined. So for example the following relationship between instants represents the notion of ‘earlier time’  $<: \mathcal{T} \times \mathcal{T}$  such that we can say 17:06PM  $<$  17:21PM. We can also define the notion of interval as a sequence of consecutive instants  $\mathcal{I} = \{[i_1, i_2] \in \mathcal{T} \times \mathcal{T} | i_1 < i_2\}$  so that, for example, [17:06PM, 17:21PM] can be the interval where the sensor was continuously active. Auxiliary useful functions like  $begin, end : \mathcal{I} \rightarrow \mathcal{T}$  can be defined to obtain the beginning and ending points of an interval:  $begin([i_1, i_2]) =_{def} i_1$  and  $end([i_1, i_2]) =_{def} i_2$ . We will consider

a set of well-known relations in the literature as those between intervals initially explored by Hamblin [54] and later adopted by Allen [3]. (see Table 3.1). Although we have adopted Interval Logic as it is by far the most widespread way to represent and reason about time in CS, especially within AI, we understand other developers may wish to use other time handling options such as the one proposed in [90].

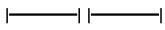
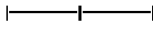





<i>Relation</i>	<i>Conditions</i>
BEFORE(X,Y)	
MEETS(X,Y)	
OVERLAP(X,Y)	
STARTS(X,Y)	
DURING(X,Y)	
FINISHES(X,Y)	
EQUAL(X,Y)	

Table 3.1: Interval-Interval relations (where X and Y represent two intervals). [3]

We considered events as noticeable occurrences of the real-world which can have an effect on a given situation. So for example the system sending a command to the light system causes it to produce light in the room. We will use a predicate  $\text{Occurs}_{at}(e, i)$  ( $\text{Occurs}_{on}(e, I)$ ) to indicate that an event  $e$  has occurred in an instant  $i$  (interval  $I$ ). For example:  $\text{Occurs}_{at}(\text{TurnOnLight}, 7 : 00\text{AM})$  or ( $\text{Occurs}_{on}(\text{Microwavecooling}, [16:10:05, 16:12:35])$ ) respectively. Mirroring explicit time references through instants and intervals, we assume non-durative and durative events defined in sorts  $\mathcal{N}$  and  $\mathcal{D}$  respectively.

We will assume the following about event instances:

$$\text{Occurs}_{on}(e, I) =_{def} \forall_{\mathcal{T}} i (\text{In}(i, I) \rightarrow \neg \text{Occurs}_{at}(e, i))$$

where  $\text{In}(i, I) =_{def} \text{Start}(i, I) \vee \text{Divides}(i, I) \vee \text{Ends}(i, I)$  where these three predicates are true when an instant is at the beginning, ‘inside’ or the end of an interval. The definition given above for  $\text{Occurs}_{on}(e, I)$  means the occurrence of a specific event in an interval implies it does not occurs inside the interval (this is usually called “non-

homogeneity”). We consider “weak negation” over durative events in the following sense:

$$\neg \text{Occurs}_{on}(e, I) =_{def} \exists_{\mathcal{T}} i (\text{In}(i, I) \wedge \neg \text{Occurs}_{at}(e, i))$$

That is, consequently with the concept of non-homogeneity explained above, an event will be considered not to have occurred if a fragment (even just an instant) of it has not occurred.

We assume the world can be described as a set of elements or entities with specific properties for which we will use the following predicate:  $\text{Holds}_{at}(p, i)$ ,  $\text{Holds}_{at} \subseteq \mathcal{P} \times \mathcal{T}$ , and  $\text{Holds}_{on}(p, I)$ ,  $\text{Holds}_{on} \subseteq \mathcal{P} \times \mathcal{I}$ , denoting that  $p$  is a property that is true in the moment  $i$  or interval  $I$  respectively.  $\text{Holds}_{on}$  and  $\text{Holds}_{at}$  are related in the following way:

$$\text{Holds}_{on}(p, I) =_{def} \forall_{\mathcal{T}} i (\text{In}(i, I) \rightarrow \text{Holds}_{at}(p, i))$$

We will assume “homogeneity” of properties over an interval, meaning that if a property holds in an interval then it also holds in any of its subintervals. For example, if a sensor was activated during 15 minutes in a row, in particular it was activated in each minute of that interval (and each second of each minute):

$$\forall_{\mathcal{T}} i \forall_{\mathcal{I}} I (\text{Holds}_{on}(p, I) \wedge \text{In}(i, I) \rightarrow \text{Holds}_{at}(p, i))$$

$$\forall_{\mathcal{I}} I, I' (\text{Holds}_{on}(p, I) \wedge I' \sqsubseteq I \rightarrow \text{Holds}_{on}(p, I'))$$

We consider “weak negation” of properties over intervals that can be obtained directly from the negation of the previous definition:

$$\neg \text{Holds}_{on}(p, I) =_{def} \exists_{\mathcal{T}} i (\text{In}(i, I) \wedge \neg \text{Holds}_{at}(p, i))$$

We will ascribe actions only to humans, so humans usually acting on their free will perform actions which typically causes some events to occur which in turn potentially change some properties of the world. We will consider that each human agent  $a$  from the sort of agents  $\mathcal{A}$  has a repertoire  $\mathcal{W}$  of possible actions  $g$ :

$$\forall_{\mathcal{A}} a \exists_{\mathcal{W}} g \text{Agent}(a, g)$$

There could be instantaneous actions  $\text{Do}_{at}$  (e.g., switching the light on) and durative actions  $\text{Do}_{on}$  (e.g., getting up from bed).

The explanations above mostly refer to the time related representation of the world. Now we focus more properly to inconsistency through the Argumentation

System. That is, how that information about a dynamic world can be grouped together to form arguments, reasons to believe or support the view of specific states of affairs in the real world we are describing.

We will assume our knowledge base is composed of a non-defeasible knowledge part  $\mathcal{K}^{\mathbb{T}}$  which in turn is organized in two subsets, one set of facts  $\mathcal{K}_G^{\mathbb{T}}$  (general knowledge) and one set of rules  $\mathcal{K}_P^{\mathbb{T}}$  (particular knowledge), where  $\mathcal{K}_P^{\mathbb{T}} \cup \mathcal{K}_G^{\mathbb{T}} = \mathcal{K}^{\mathbb{T}}$  and  $\mathcal{K}_P^{\mathbb{T}} \cap \mathcal{K}_G^{\mathbb{T}} = \emptyset$ .  $\mathcal{K}_P^{\mathbb{T}}$  represents the safe facts of the world such as the existence of a specific bedroom in a specific house and a week in the calendar having seven days, and  $\mathcal{K}_G^{\mathbb{T}}$  represents general laws, e.g. that if Monday is a day of a week then it has 24 hours. There is also a finite set  $\Delta^{\mathbb{T}}$  of *temporal defeasible rules* representing knowledge that our AmI system agent  $a^{\mathbb{T}}$  is prepared to accept unless it finds counter-evidence. Rules in  $\Delta^{\mathbb{T}}$  have the form  $\alpha \succ \beta$ , where  $\alpha$  and  $\beta$  are sets of literals of  $\mathcal{L}^{\mathbb{T}}$ .  $\Delta^{\mathbb{T}\downarrow}$  will denote the set of basic instances of members of  $\Delta^{\mathbb{T}}$ . Our simplified explanation of later sections will actually only use  $\Delta^{\mathbb{T}\downarrow}$  instead of the usually preferable  $\Delta^{\mathbb{T}}$  as we merely want to illustrate the potential of argumentation to capture certain key aspects of preference handling.

We will largely adhere to the notation used in [13] and use  $(\mathcal{K}^{\mathbb{T}}, \Delta^{\mathbb{T}})$  to denote a *temporal defeasible structure*, where  $\mathcal{K}^{\mathbb{T}}$  is a temporal context and  $\Delta^{\mathbb{T}}$  is a finite set of temporal defeasible rules. We will also adopt the same notion of *temporal defeasible consequence*, “ $\sim$ ”, and the notion of  $A$  of  $\Delta^{\mathbb{T}\downarrow}$  as a *temporal argument* for a temporal literal  $h$  and the associated notion of a subargument. Let  $(\mathcal{K}^{\mathbb{T}}, \Delta^{\mathbb{T}})$  be a temporal defeasible structure of  $a^{\mathbb{T}}$ .  $\mathbb{T}\mathbf{A}\mathbf{S}\mathbf{t}\mathbf{r}\mathbf{u}\mathbf{c}(\Delta^{\mathbb{T}\downarrow})$  will be the set of temporal arguments that can be constructed from  $(\mathcal{K}^{\mathbb{T}}, \Delta^{\mathbb{T}\downarrow})$ .

Our notion of disagreement is related to time, so given a temporal function  $\rho(\{h_1, h_2\})$  which determines whether two temporal literals  $h_1$  and  $h_2$  intersect in their time references, and given two temporal arguments  $\langle A_1, h_1 \rangle$  and  $\langle A_2, h_2 \rangle$ ,  $A_1$  for  $h_1$  and  $A_2$  for  $h_2$  are *in disagreement at least about an instant  $i$* ,  $\langle A_1, h_1 \rangle \bowtie_{\mathbb{T}} \langle A_2, h_2 \rangle$ , if and only if  $\rho(\{h_1, h_2\}) \neq \emptyset$  and  $\mathcal{K}^{\mathbb{T}} \cup \{h_1, h_2\} \vdash \perp$ . So at least a common temporal reference is required between the temporal references of the arguments involved in the conflict.

A temporal argument  $\langle A_1, h_1 \rangle$  *counterargues* another temporal argument  $\langle A_2, h_2 \rangle$  in a basic literal  $h$ , if and only if there exists a subargument  $\langle A, h \rangle$  of  $\langle A_2, h_2 \rangle$  such that  $\langle A_1, h_1 \rangle$  and  $\langle A, h \rangle$  are in disagreement (in at least an instant  $i$ ). Let  $\succ$  be a partial order defined over elements of  $\mathbb{T}\mathbf{A}\mathbf{S}\mathbf{t}\mathbf{r}\mathbf{u}\mathbf{c}(\Delta^{\mathbb{T}\downarrow})$ , we will say that a temporal argument  $\langle A_1, h_1 \rangle$  *defeats* another  $\langle A_2, h_2 \rangle$ ,  $\langle A_1, h_1 \rangle \gg_{\text{tdef}} \langle A_2, h_2 \rangle$ , if and only if there exists a subargument  $\langle A, h \rangle$  of  $\langle A_2, h_2 \rangle$  such as  $\langle A_1, h_1 \rangle$  counterargues  $\langle A_2, h_2 \rangle$  in  $h$  and  $\langle A_1, h_1 \rangle \succ \langle A, h \rangle$ .

When there is a conflict between arguments, preference criteria are used to understand whether some arguments may be preferable to others, e.g. specificity. Specificity is based on the structure of the arguments. It has the advantage of being independent from the application domain. Still, there are several other criteria which can be used to compare and select arguments. In some cases *Persistency* over time could be used as a reason to prefer an explanation over another. We assume properties persist unless we have reasons to believe otherwise. We will use predicates  $\text{Change}_{at}^{+ -}(p, i)$  and  $\text{Change}_{in}^{+ -}(p, I)$  to indicate that a proposition  $p$  changes its truth value from being true to false at an instant  $i$  or in an interval  $I$  respectively. The following axioms allow the detection of these situations:

$$\begin{aligned} \forall_{\mathcal{P}} p \forall_{\mathcal{T}} i (\text{Holds}_{at}(p, i - 1) \wedge \neg \text{Holds}_{at}(p, i) \\ \rightarrow \text{Change}_{at}^{+ -}(p, i)) \end{aligned}$$

$$\begin{aligned} \forall_{\mathcal{P}} p \forall_{\mathcal{I}} I, I' (\text{MEETS}(I, I') \wedge \text{Holds}_{on}(p, I) \wedge \neg \text{Holds}_{on}(p, I') \\ \rightarrow \text{Change}_{in}^{+ -}(p, I')) \end{aligned}$$

where “MEETS” should be considered as in [54, 3]. We can also consider analogous axioms for  $\text{Change}_{at}^{- +}$  and  $\text{Change}_{in}^{- +}$  for properties changing from being false to being true. Let  $\langle A_1, h_1 \rangle, \langle A_2, h_2 \rangle \in \mathbf{TAStruct}(\Delta^{\mathcal{T}^{\downarrow}})$ , we say that  $A_1$  for  $h_1$  is *preferred under persistency* to  $A_2$  for  $h_2$ , noted  $\langle A_1, h_1 \rangle \succ_{\text{tpers}} \langle A_2, h_2 \rangle$ , if and only if  $\langle A_2, h_2 \rangle$  use persistency and  $\langle A_1, h_1 \rangle$  does not.

In the next section we assume the following *precedence order* [83] between the preference criteria:  $\mathfrak{R} = \{\succ_{\text{tspec}}, \succ_{\text{tpers}}\}, \succ_{\text{tspec}} > \succ_{\text{tpers}}$ . This means we apply specificity first. When the arguments are incomparable under specificity or they are equi-specific we apply the persistency criteria. The next section complements this with a more user personalised preference criterion.

A conclusion  $C$  is “justified” when there is at least an argument in support of  $C$  and there are no other better counter-argument(s). For a more formal explanation of the notion of “support” as regards argumentation syntax, see [13]. The research also adopted an existing system (MReasoner), which we discuss in the next chapter. See [57] for additional details.



### 3.1.1 Lighting management case illustrated using Argumentation

The case study which has been described in Section 1.4.1 in three different scenarios has been translated into a more technical form in Table 3.2. Further below, illustrates how argumentation can handle users' preferences over time and potential conflictive scenarios. Tables 3.3, 3.4, and 3.5 show at the beginning the initial state of the world and then the evolution of the scenario through the *grounded arguments*,  $A^\downarrow$ .

At the end of each scenario, the arguments were illustrated in a tree format. However, the formal language on each table was not strictly used to create the tree, because this research only wanted to demonstrate the basic idea behind each arguments. The time measurement assumed in the three scenarios is expressed in minutes.

Table 3.2: Dynamics evolution of the Light Case Scenario as regards time

Scenario 1	Interval Relationship	$MEETS(I_0, I_1) \wedge MEETS(I_1, I_2) \wedge MEETS(I_2, I_3) \wedge MEETS(I_3, I_4)$									
	Initial Stage	$Holds_{on}(Movement, I_0) \wedge \neg Holds_{on}(Sleeping, I_0) \wedge \neg Holds_{on}(OnBed, I_0) \wedge Holds_{on}(LightsOn, I_0)$									
	Properties	Movement	Movement	$\neg$ Movement	$\neg$ Movement	$\neg$ Movement					
		$\neg$ Sleeping	$\neg$ Sleeping	$\neg$ Sleeping	Sleeping	Sleeping					
$\neg$ OnBed		OnBed	OnBed	OnBed	OnBed						
	LightsOn	LightsOn	LightsOn	LightsOn	$\neg$ LightsOn						
Transition Cause	$Do_{on}$ (GoingToBed, $I_0$ )	$\neg Occurs_{at}$ (MoveDetected, end( $I_1$ ))	$Holds_{on}(OnBed,$ $I_2) \wedge \neg Holds_{on}$ (Movement, $I_2) \wedge Length(I_2) > 10$	$Occurs_{on}$ (SystemTurns LightOff, $I_1$ )							
Intervals	$I_0$	$I_1$	$I_2$	$I_3$	$I_4$	$I_5$	$I_6$	$I_7$	$I_8$		
Scenario 2	Interval Relationship	$MEETS(I_0, I_1) \wedge MEETS(I_1, I_2) \wedge MEETS(I_2, I_3) \wedge MEETS(I_3, I_4) \wedge MEETS(I_4, I_5) \wedge MEETS(I_5, I_6) \wedge MEETS(I_6, I_7) \wedge MEETS(I_7, I_8)$									
	Initial Stage	$\neg Holds_{on}(Movement, I_0) \wedge Holds_{on}(Sleeping, I_0) \wedge Holds_{on}(OnBed, I_0) \wedge \neg Holds_{on}(LightsOn, I_0)$									
	Properties	$\neg$ Movement	Movement	Movement	Movement	Movement	Movement	Movement	$\neg$ Movement	$\neg$ Movement	$\neg$ Movement
		Sleeping	Sleeping	$\neg$ Sleeping	$\neg$ Sleeping	$\neg$ Sleeping	$\neg$ Sleeping	$\neg$ Sleeping	$\neg$ Sleeping	Sleeping	Sleeping
OnBed		OnBed	OnBed	OnBed	$\neg$ OnBed	$\neg$ OnBed	OnBed	OnBed	OnBed	OnBed	
	$\neg$ LightsOn	$\neg$ LightsOn	$\neg$ LightsOn	$\neg$ LightsOn	LightsOn	LightsOn	LightsOn	LightsOn	LightsOn	$\neg$ LightsOn	
Transition Cause	$Occurs_{at}$ (MoveDetected, begin( $I_1$ ))	$Holds_{on}$ (Movement, $I_1) \wedge Length$ ( $I_1) > 2$	$Do_{on}$ (GettingOut Of Bed, $I_2$ )	$Occurs_{on}$ (MoveDetected, $I_3) \wedge \neg Holds_{at}$ (OnBed, begin( $I_3$ ))	$Do_{on}$ (Going ToBed, $I_4$ )	$\neg Occurs_{at}$ (MoveDetected, begin( $I_6$ ))	$Holds_{on}(OnBed, I_6)$ $\wedge \neg Holds_{on}$ (Movement, $I_6)$ $\wedge Length(I_6) > 10$	$Occurs_{at}$ (SystemTurns LightOff, begin( $I_7$ ))			
Intervals	$I_0$	$I_1$	$I_2$	$I_3$	$I_4$	$I_5$	$I_6$	$I_7$	$I_8$		
Scenario 3	Interval Relationship	$MEETS(I_0, I_1) \wedge MEETS(I_1, I_2) \wedge MEETS(I_2, I_3) \wedge MEETS(I_3, I_4) \wedge MEETS(I_4, I_5) \wedge MEETS(I_5, I_6)$									
	Initial Stage	$\neg Holds_{on}(Movement, I_0) \wedge Holds_{on}(Sleeping, I_0) \wedge Holds_{on}(OnBed, I_0) \wedge \neg Holds_{on}(LightsOn, I_0)$									
	Properties	$\neg$ Movement	Movement	Movement	Movement	Movement	$\neg$ Movement	$\neg$ Movement			
		Sleeping	Sleeping	$\neg$ Sleeping	$\neg$ Sleeping	$\neg$ Sleeping	$\neg$ Sleeping	$\neg$ Sleeping			
OnBed		OnBed	OnBed	OnBed	$\neg$ OnBed	$\neg$ OnBed	$\neg$ OnBed				
	$\neg$ LightsOn	$\neg$ LightsOn	$\neg$ LightsOn	$\neg$ LightsOn	LightsOn	LightsOn	$\neg$ LightsOn				
Transition Cause	$Occurs_{at}$ (AlarmRings, end( $I_0$ ))	$Holds_{on}$ (Movement, $I_1) \wedge Length$ ( $I_1) > 2$	$Do_{at}$ (GettingOut Of Bed, begin( $I_2$ ))	$\neg Holds_{at}$ (OnBed, begin( $I_3$ )) $\wedge Holds_{on}$ (Movement, $I_3)$	$Do_{on}$ (Leaving Home, $I_4$ )	$\neg Holds_{on}(Movement,$ $I_5) \wedge Length(I_5)$ $> 15 \wedge \neg Holds_{at}$ (OnBed, $I_5)$	NotAtHome				
Intervals	$I_0$	$I_1$	$I_2$	$I_3$	$I_4$	$I_5$	$I_6$	$I_7$	$I_8$		

Table 3.3: Knowledge Representation for First Scenario  $\neg$  *LightsOn* and *LightsOn*

$MEETS(I_0, I_1) \wedge MEETS(I_1, I_2) \wedge MEETS(I_2, I_3) \wedge MEETS(I_3, I_4)$	
$Holdson(Movement, I_0) \wedge \neg Holdson(Sleeping, I_0) \wedge \neg Holdson(OnBed, I_0) \wedge Holdson(LightsOn, I_0)$	
(S1, R1)	$Do_{on}(GoingToBed, I_0) \succ Occurs_{at}(GettingOnBed, begin(I_1))$
(S1, R2)	$Occurs_{at}(GettingToBed, begin(I_1)) \succ Hold_{at}(OnBed, begin(I_1))$
(S1, R3)	$\neg Occurs_{at}(MoveDecteded, end(I_1)) \succ \neg Hold_{at}(Movement, end(I_1))$
(S1, R4)	$Holdson(OnBed, I_2) \wedge \neg Holdson(Movement, I_2) \wedge Length(I_2) > 10$ $\succ Hold_{at}(Sleeping, end(I_2))$
(S1, R5)	$Holdson(Sleeping, I_3) \succ Occurs_{on}(SystemTurnsLighsOff, I_3),$
(S1, R6)	$Occurs_{on}(TurnLigtsOff, I_3) \succ \neg Holdson(LightsOn, I_4)$

Table 3.2 shows the progression in time of the three scenarios. The time "Intervals" row at the end of each scenario states the different relevant time periods for the scenarios, for example for Scenario 1, 5 different intervals were uses  $I_0, \dots I_4$ . The first "Interval Relationship" row in each scenario states how they relate to each other in time). For the first scenario it states all the different time intervals mentioned are consecutive to each other.

The "Initial Stage" row states how the system is supposed to be at the time the scenario is considered. For Scenario 1 it states that for the interval  $I_0$  there is movement being detected by the PIR sensor, that the system believes the person is not sleeping and is not in bed and through the light sensor the system detects the lights are on in the bedroom.

The "Properties" section consists of a number of rows, one for each relevant property which depicts the state of the system under consideration. In Scenario 1 traces the evolution of movement detection (MoveDetected) as it evolves through time, and this shows that movement is detected through the PIR sensor during  $I_0$  and  $I_1$  but movement is not detected ( $\neg$  MoveDetected) in the whole of  $I_2, I_3$  and  $I_4$ .

The "Transition Cause" row explains how the world transitions from one state to the next one, it explains change. For example, Scenario 1, in  $I_0$  the system believes the person is not in bed, and then at  $I_1$  it believes the person is in bed. This is actually triggered by the action of the person going to bed ( $Do_{on}(GoingToBed, I_0)$ ).

So to understand how the scenario evolves the reader has to see the values of the properties in two consecutive states of the system of the "Properties" area of the table, and look at the Transition cause under the first state which will explain how the system transitioned to the next state. In Scenario 1, the transition from  $I_1$  to  $I_2$  is caused by an event (hence the use of an *Occurs* predicate), then the transition from  $I_2$  to  $I_3$  is caused by a condition which triggers a rule in the system modifying the current belief of the system (hence the use of an *Hold*s predicate), the transition

from  $I_3$  to  $I_4$  is caused by an event (hence the use of an  $\text{Occurs}_{on}$  predicate).

In summary the research adopted the convention that the states of the system can change due to an action of the user (Do), an event related to a sensor (Occurs) or an update in the system's beliefs (Holds). Scenarios 2 and 3 evolve in similar fashion.

### First Scenario

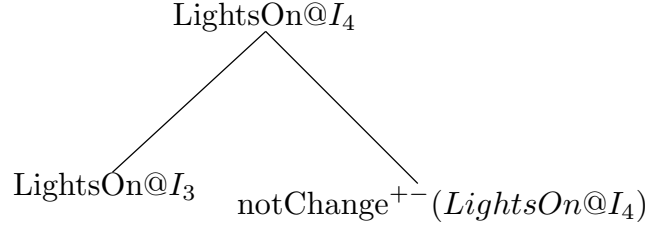
Table 3.3 focuses on the formalization of the first scenario. An informal description of what happened in the first scenario is given in Table 1.2 in Section 1.4.1, then in Table 3.2 the formalization of the evolution of that scenario in time through different states was provided as well as of the actions, events and conditions which triggered those changes. Table 3.3 focuses on the defeasible rules which allows the system to reason with the knowledge of the world as it changes so that is context-aware and can react to the right contexts with sensible actuations.

The first line of the table shows the relationship of the intervals of time, these are the same as they were stated in Table 3.2. The first column associates labels to the rules, for example (S1, R4) refers to the fourth rule of the first scenario. The interpretation of the rules is according to the syntax and semantics given for the knowledge representation language given in [13].

For example, R1 states that when the user performs the durative action of going to bed, it will have as a result the occurrence of the event getting on bed. R2 states that this event in turns has as an effect on the holding of the property of being on bed. R3 states if the system detects through sensors there is no movement detected at an instant (in this case at the end of  $I_1$ ) then the system infers there is no movement at that time. R4 states if the systems has information the person is in bed and there is no movement for more than 10 units of time (for example 10 minutes) these are reasons to believe the person is sleeping. R5 states the believe the person is sleeping is a reason for the system to turn the lights off. R6 states when lights went off the consequence is that the lights are not on anymore (it is assumed as a simplification there is not other source of light and the room is dark).

**Argument A for the first scenario:** As known from the initial facts, the user turns the lights on when he enters the room. So there is a possibility, because of persistency, that the lights will remain on as reflected in the following argumentation tree in Figure 3.1.

**Argument B for the first scenario:** There is an alternative explanation which is better informed than the previous one, given that the system has been programmed to understand when the lights are not needed ( $\neg\text{Holds}_{on}(\text{LightsOn}, \dots)$ ). The tree

Figure 3.1: First Scenario Argument A Tree for *LightsOn*.

in Figure 3.2 indicates that Bob was going to bed at  $I_0$  and at  $I_1$  Bob was in bed and stayed in bed till at  $I_2$  as seen in the lower left part of the tree. Since there was no movement detected at  $I_2$  (lower right part of the tree), the system has reasons to believe that Bob is asleep at  $I_2$ . Bob persists on sleeping all through  $I_3$ . At that moment the system infers that it is reasonable to turn the lights off. As a result, the lights are off at  $I_4$ .

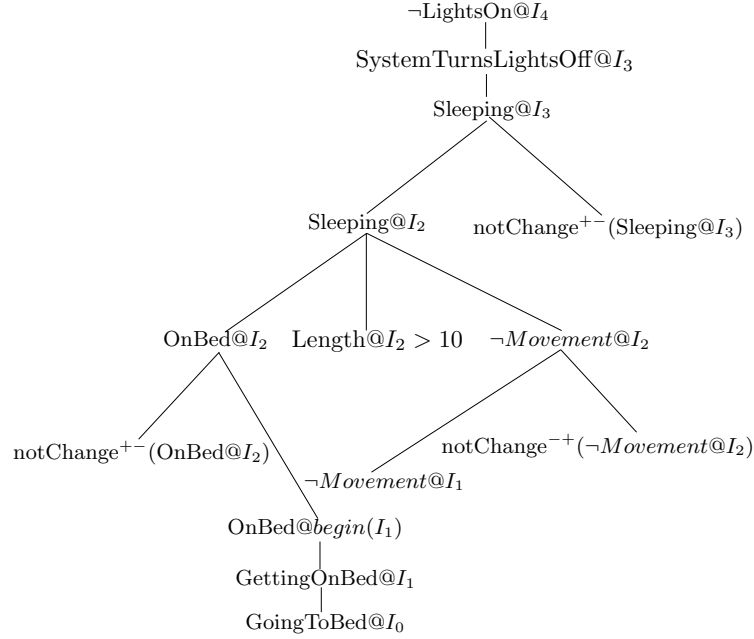
Table 3.4: Knowledge Representation for Second Scenario  $\neg$  *LightsOn* and *LightsOn*

$MEETS(I_0, I_1) \wedge MEETS(I_1, I_2) \wedge MEETS(I_2, I_3) \wedge MEETS(I_3, I_4) \wedge MEETS(I_4, I_5)$ $\wedge MEETS(I_5, I_6) \wedge MEETS(I_6, I_7) \wedge MEETS(I_7, I_8)$	
$\neg Holds_{on}(Movement, I_0) \wedge Holds_{on}(Sleeping, I_0) \wedge Holds_{on}(OnBed, I_0) \wedge \neg Holds_{on}(LightsOn, I_0)$	
(S2, R1)	$Occurs_{at}(MoveDetected, begin(I_1)) \succ Holds_{on}(Movement, I_1)$
(S2, R2)	$Holds_{on}(Movement, I_1) \wedge Length(I_1) > 2 \succ \neg Holds_{on}(Sleeping, I_1)$
(S2, R3)	$Do_{on}(GettingOutOfBed, I_2) \succ Occurs_{at}(GetsOutOfBed, end(I_2))$
(S2, R4)	$Occurs_{at}(GetsOutOfBed, end(I_2)) \succ \neg Hold_{at}(OnBed, begin(I_3))$
(S2, R5)	$Occurs_{on}(MoveDetected, I_3) \wedge \neg Hold_{at}(OnBed, begin(I_3))$ $\succ Occurs_{on}(SystemTurnLightsOn, I_3)$
(S2, R6)	$Occurs_{on}(SystemTurnLightsOn, I_3) \succ Hold_{at}(LightsOn, end(I_3))$
(S2, R7)	$Do_{on}(GoingToBed, I_4) \succ Occurs_{at}(GettingOnBed, begin(I_5))$
(S2, R8)	$Occurs_{at}(GettingToBed, begin(I_5)) \succ Hold_{at}(OnBed, end(I_5))$
(S2, R9)	$\neg Occurs_{at}(MoveDetected, begin(I_6)) \succ \neg Hold_{at}(Movement, end(I_6))$
(S2, R10)	$Holds_{on}(OnBed, I_6) \wedge \neg Holds_{on}(Movement, I_6) \wedge Length(I_6) > 10 \succ Holds_{on}(Sleeping, I_6)$
(S2, R11)	$Holds_{on}(Sleeping, I_6) \succ Occurs_{on}(SystemTurnLightsOff, I_7)$
(S2, R12)	$Occurs_{on}(SystemTurnLightsOff, I_7) \succ \neg Hold_{at}(LightsOn, end(I_8))$

From the first scenario,  $A \bowtie_{\mathbb{T}} B$  about  $I_4$ ,  $B \succ_{\text{tspec}} A$  because there is more information to support the reason that the user is asleep. Therefore,  $B \gg_{\text{tdef}} A$ , now the system can state  $\Delta^{\uparrow} \downarrow \sim \neg Holds_{on}(LightsOn, I_4)$ .

## Second Scenario

Table 3.4 focuses on the formalization of the second scenario. An informal description of what happened in the second scenario was given in Table 1.2 section 1.4.1. As previously stated, Table 3.2 provides the formalization of the evolution of that scenario in time through different states and also of actions, events and conditions

Figure 3.2: First Scenario Argument B Tree for  $\neg LightsOn$ .

that triggered the changes. Table 3.4 also focusses on defeasible rules just like Table 3.3 (same conventions apply for all rule tables).

Row labelled (S2, R1) states that when the system detects movement (maybe the user wakes up in the middle of the night to use the toilet), the property movement holds. Row labelled (S2, R2) states that if the movement continues over the next two minutes then the system believe that the user is not sleeping. Row labelled (S2, R3) states the durative action of the user getting out of bed, it will have as a result of the occurrence of the user is out of bed. This in turn has an effect in (S2, R4) that the user is not in bed anymore. S2, R5 states that if movement is detected via sensor, and if the user is not in bed, then the system turns the light on. Row labelled (S2, R6) states that when the system turns the light on, then the lights stays on. Row labelled (S2, R7) states that the durative action of going back to bed (after using the toilet) causes the event of the user being in bed. Rows labelled (S2, R7) to (S2, R12) are similar to rows labelled (S1, R1) to (S2, R6) of the first scenario.

**Argument A for the Second scenario:** As seen from the initial facts, the user turns on the light when he wakes up in the middle of the night, for example to use the toilet, so there is a possibility that the light will remain on at  $I_8$  until he turns it off again.

**Argument B for second scenario:** There is an alternative description for the second scenario which is more informed than argument A. Thus, knowing that,

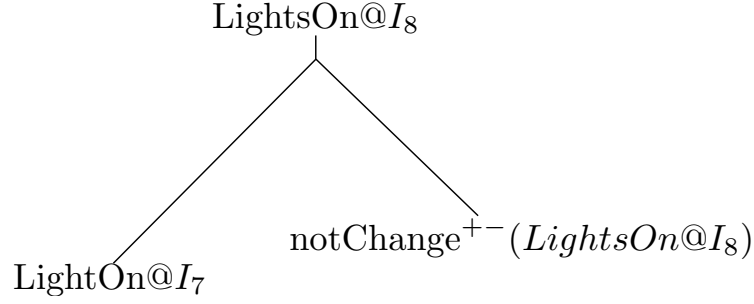


Figure 3.3: Second Scenario Argument A Tree for *LightsOn*

the system has been programmed to understand that the lights are not needed  $\neg Holds_{on}(LightsOn, \dots)$ . Figure 3.4 signifies that if Bob was going back to bed, such as at  $I_4$ , and was in bed at  $I_5$  (as seen in the lower right hand side of the tree) then Bob will be in bed from this interval onwards. Then for the system to have reasons to believe that Bob is asleep at  $I_6$ , the system will not have detected any movement at  $I_6$  and if this situation persists for the next 10 minutes, then the system concludes that Bob is now sleeping. Also if Bob persists on sleeping all through at  $I_6$ , then system assumes at  $I_7$  that it is reasonable to turn off the lights, as a result of that, the lights are off at  $I_8$ .

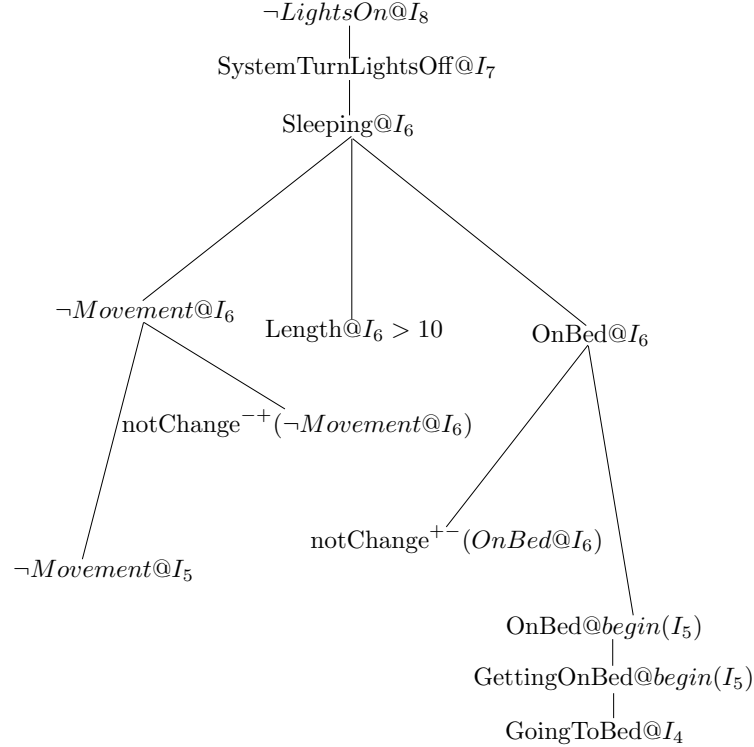


Figure 3.4: Second Scenario Argument B Tree for  $\neg LightsOn$ .

Table 3.5: Knowledge Representation for Third Scenario  $\neg$  *LightsOn* and *LightsOn*

$MEETS(I_0, I_1) \wedge MEETS(I_1, I_2) \wedge MEETS(I_2, I_3) \wedge MEETS(I_3, I_4) \wedge$ $MEETS(I_4, I_5) \wedge MEETS(I_5, I_6)$	
(S3, R1)	$Occurs_{at}(AlarmRings, end(I_0)) \succ Holds_{on}(Movement, I_1)$
(S3, R2)	$Holds_{on}(Movement, I_1) \wedge Length(I_1) > 2 \succ \neg Holds_{at}(Sleeping, end(I_1))$
(S3, R3)	$Do_{at}(GettingOutOfBed, begin(I_2)) \succ Occurs_{at}(GetsOutOfBed, end(I_2))$
(S3, R4)	$Occurs_{at}(GetsOutOfBed, end(I_2)) \succ \neg Holds_{at}(OnBed, begin(I_3))$
(S3, R5)	$\neg Holds_{at}(OnBed, begin(I_3)) \wedge Holds_{on}(Movement, I_3)$ $\succ Occurs_{on}(SystemTurnLightsOn, I_3)$
(S3, R6)	$Occurs_{on}(SystemTurnLightsOn, I_3) \succ Holds_{at}(LightsOn, end(I_3))$
(S3, R7)	$Do_{on}(LeavingHome, I_4) \succ Occurs_{at}(LeftHome, end(I_4))$
(S3, R8)	$Occurs_{at}(LeftHome, end(I_4)) \succ \neg Holds_{on}(Movement, I_5)$
(S3, R9)	$\neg Holds_{on}(Movement, I_5) \wedge Length(I_5) > 15 \wedge \neg Holds_{on}(OnBed, I_5)$ $\succ Occurs_{on}(SystemTurnLightsOff, I_6)$
(S3, R10)	$Occurs_{on}(SystemTurnLightsOff, I_6) \succ \neg Holds_{on}(LightsOn, I_6)$

From the second scenario,  $A \bowtie_{\mathbb{T}} B$  about  $I_8$ ,  $B \succ_{\mathbb{T}spec} A$  because there is more information to support the reason that the user has gone back to sleep so the system turns the light off. Therefore,  $B \gg_{\mathbb{T}def} A$ , now the system can state:

$$\Delta^{\mathbb{T}\downarrow} \sim \neg Holds_{on}(LightsOn, I_8).$$

### Third Scenario

Table 3.5 focuses on the formalization of the third scenario. An informal description of the third scenario given in Table 1.2, Section 1.4.1. Table 3.2 provides the formalization of the evolution of that scenario in time whilst Table 3.5 focusses on defeasible rules.

Row labelled (S3, R1) states the occurrence of the alarm ringing which will lead to awakening the user who will then begin to move. Row labelled (S3, R2) states that if the movement continues for more than two minutes, then the system believes that the user is not sleeping. Row labelled (S3, R3) states the durative action of getting out of bed out being performed by the user, will result in the occurrence of the event getting off bed. Row labelled (S3, R4) states that this event in turns has an effect on the holding property of not being in bed. Row labelled (S3, R5) that states when property states that user is not in bed and movement is detected with the use of sensors, then the system turns the light on. Row labelled (S3, R6) reflects the effect of the event which turns the light on. Row labelled (S3, R7) states that the durative action of the user leaving home will will lead to the occurrence of event left home. Row labelled (S3, R8) states that when the user has left no movement is expected ( $\neg Occurs_{on}(Movement, I_5)$ ). Row labelled (S3, R9) states that if the property holds no movement and this state remains the same for over 15 units of



time (for example 15 minutes) and the bed sensor does not detect anyone in bed, this will make the system to infer that the user has left home and then turns off the light. Row labelled (S3, R10) reflect the effect of the system turning the light off.

**Argument A for the third scenario:** The initial facts show that the user turns the light on at  $I_5$  when he wakes up in the morning, and as a result, there is a possibility that the light will remain on at  $I_6$  as shown in the argumentation tree in figure 3.5.

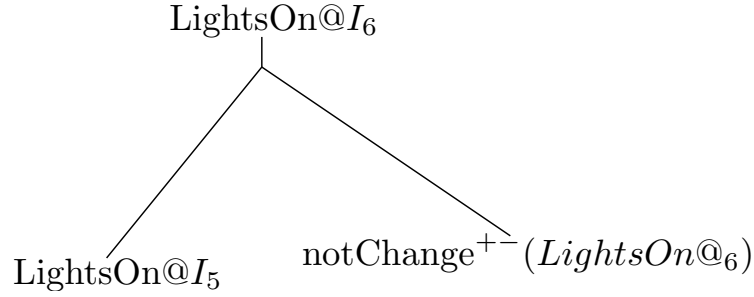


Figure 3.5: Third Scenario Argument A Tree for *LightsOn*.

**Argument B for the third scenario:** The alarm rings at  $I_0$  which will awake the user. As he begins to move, this movement is detected by the system at  $I_1$  and persists for the next 10 minutes, then the system understands that the user is awake as seen at the lower middle of the tree Figure 3.6. When the user gets out of bed at  $I_2$ , then he is no longer on bed at  $I_3$ , as shown in the low right of the argumentation tree. This informs the system which then turns the light on at  $I_3$ . As the persistence of not being in bed continues from  $I_3$  to  $I_4$  the system continues to keep the lights on (unless the user turns the light off). The user is about to leave home at  $I_4$ , then at end of  $I_4$  the user is out of home. It is possible that the user forgets to switch off the lights before he leaves home (which happened in this case).

As a result, the system turns the lights off at  $I_6$  after no movement is detected at  $I_5$  and not persistent state of  $\neg Holds_{OnBed}$  remains at  $I_5$ . The resulting argument is explained in figure 3.6.

From the second scenario,  $A \bowtie_{\mathbb{T}} B$  about  $I_6$ ,  $B \succ_{\mathbb{T}spec} A$  because there is more information to support the reason that the user has left home and then the system turns the light off. Therefore,  $B \gg_{\mathbb{T}def} A$ , now the system can state:

$$\Delta^{\mathbb{T}\downarrow} \sim \neg Holds_{on}(LightsOn, I_6).$$

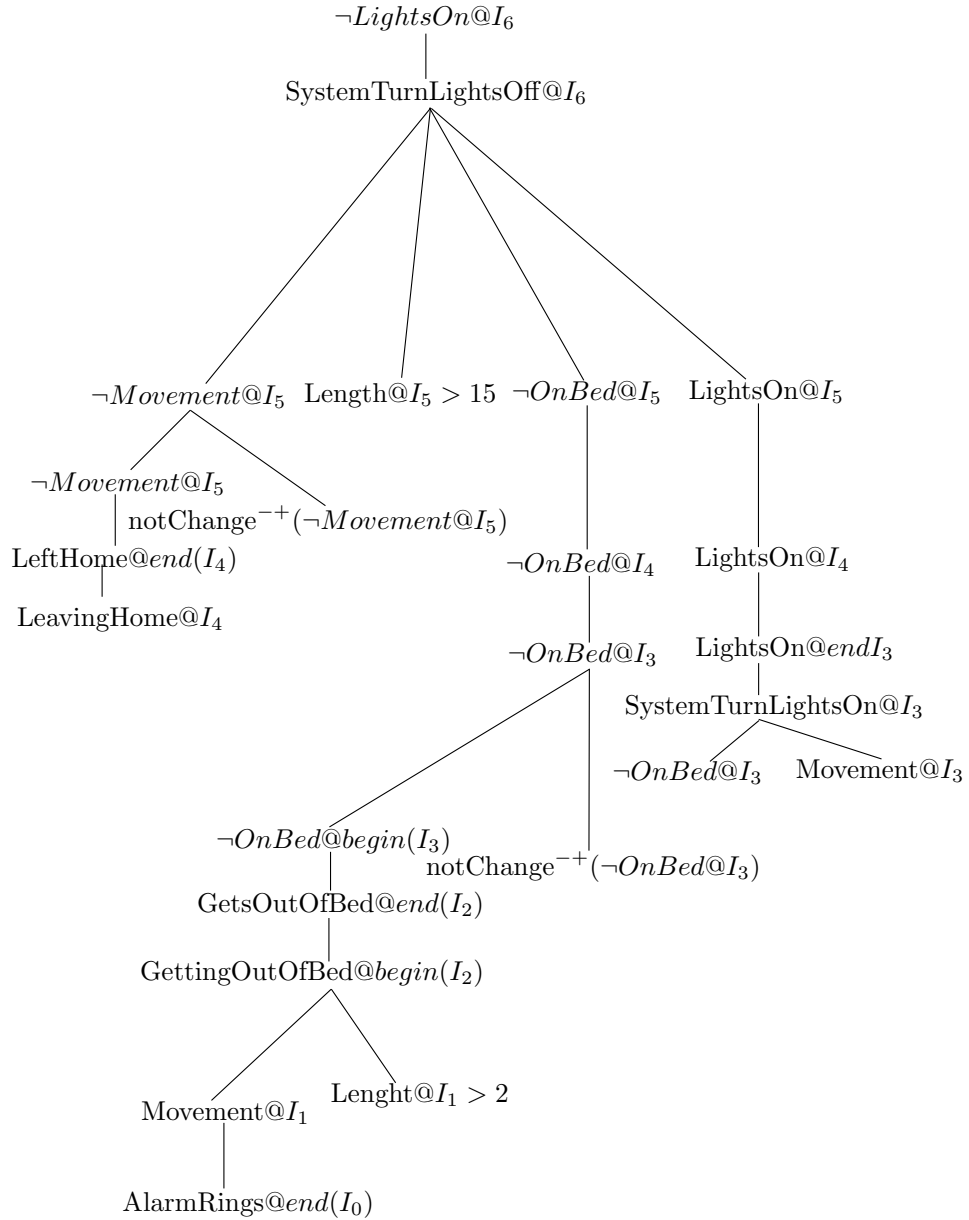


Figure 3.6: Third Scenario Argument B Tree for  $\neg LightsOn$ .

Table 3.6: Comparison of Classical Preferences in AI and Argumentation

	<b>Preferences in Classical AI</b>	<b>Argumentation</b>
Conflict Resolution	Preference methods in AI aim at decision-support systems which include web-based recommender systems, solving automated problems [81] and other interactive systems that aim to elicit and satisfy the users, preferences in order to give satisfactory recommendation.	Argumentation has been shown to handle complex situations in the previous work ([67]; [23]; [24]; [7]) especially in dealing with conflicts, and this has made researchers channel attention to this popular conflict resolution approach. Argumentation was shown to be a very relevant topic in AmI domain [55].
Application to complex problems	Most preferences handling methods in AI (CP-nets specifically) are restricted to preferences that are strict/complete (which a limitation identified by [4] in his study), as the outcome is already known. Strict or binary valued preference occurs in everyday life (such as, Bob prefers the light to be off at 10pm) though multivalued preferences are not common (Bob prefers the light to be switched off in the evening). The latter is neither strict nor complete as the term "evening" is ambiguous thereby arising conflicting questions like when in the evening?	Argumentation covers a wide range of disciplines just like preferences in AI, although it has been applied in wider domains ([27]; [69]; [66]; [43]) in AmI as a knowledge representation and reasoning paradigm, for dealing with incomplete and inconsistent (contradictory) knowledge. Though, one of its main challenges is to design a formal system that enjoys desirable semantic properties and tractable computational complexity, while being easy to understand.
Decision Making	Preferences in AI are known to express preferential dependencies between attributes [51], such as when a Bob prefers to by hard cover mathematics book (which he reads often) and a paperback survey book (which might be read not more than twice). This indicates that the choice is dependent on the book type. This limits preferences in AI in the sense that they cannot model an arbitrary preference over a combinatorial domain.	In a usual context, once a decision is made a course of action is taken leaving behind other possible choices. However, decision making in argumentation is supported by reasoning, which will account for the characteristics of the various available alternatives [42]. This shows the ability that argumentation has to reason in a changing world where information is not complete. When new information surfaces, it gives considerations to obtain new reason to further conclusions or better reasons to sustain previous one.
Ability to reason and represent users' preferences	One main important factor of preferences in AI is that they aid elicitation of preference information from non-expert users directly or indirectly. However, certain questions are yet to be addressed, including: How can these preferences be represented? How will they be used for reasoning? Can they be actually computed? [51].	Argumentation handles problems in AI which includes defeasible reasoning, (see [37]; [84]; [13]; [27]; [43]). Using the notion of instant or interval or both, [13] has demonstrated how known problems of defeasible reasoning can be solved.
Ability to handle time	Despite the apparent importance of preferences in AI, as it has been applied to handle challenges posed in AI (such as: cognitive challenges, computational challenges, conceptual challenges and representational challenges) [31], there has been no recognition of preferences in AI having the ability to represent users' preference over time	Apart from the fact that argumentation is now a popular conflict resolution approach, and has been applied successfully in [55], it has also been theoretically proven that argumentation can be used to represent users' preferences over time [13].

### 3.1.2 Argumentation in AI

The evolution of argumentation emerged as an alternative to non-monotonic formalisms based on classical logic from the mid-1980s to present [37]. Modelling common sense reasoning has long been a challenge in artificial intelligence (AI), as it mostly occurs in the face of incomplete and potentially inconsistent information [37]. Several non-monotonic reasoning formalisms emerged to match this challenge, but in this formalism, when additional information is obtained, conclusions drawn may be later withdrawn [37]. Formal logics of argument emerged as one style of formalizing nonmonotonic reasoning, as argumentation systems provide a nonmonotonic layer to reason about justification of truth [89].

The reputation of argumentation in AI has positively increased [64], which is why it has been widely used for handling inconsistent knowledge ([89], [6], [25] and [47]) and dealing with uncertainty in making decision(s) ([6] and [8]). The features of time and conflict-handling in argumentation systems have long been investigated in computer science ([66], [13], [23], [24], and [67]). Argumentation has been known as a way to implement and formalize defeasible reasoning [89], allowing us to reason about a changing world where the information available is not very reliable or incomplete.

Argumentation as a reasoning process can help in making decisions by handling conflicting situations expressed within deliberative agents [92]. The fundamental ideas behind argumentation are to construct arguments in favor of and against each decision, evaluate the arguments and apply some principle of comparing their value based on quality or strength [5]. The value of an argument can be qualified as defensible, justified, or defeated as it is determined by the importance of the rules (reasons) it contains [88]. The knowledge of new fact(s) can also lead to another conclusion being obtained. The obtained conclusions are justified through arguments to support their consideration [89].

When conflict arises among arguments, methods or preferences criteria are used to understand if some arguments may be preferred over others. Establishing the preference of an argument over another or a set of arguments over others, requires some definition of preference criteria, for example “Specificity” and “Persistency”. These criteria were adopted during our implementation process, combined with “User Preferences” which was introduced in [78].

“Specificity” as a preference criteria is based on the argument structure, and decisions can be made based on which argument is better informed than the other. “Persistency” on the other hand, assumes that properties tend to keep their truth values through time, unless there is a reason to believe otherwise.

### 3.1.3 Preferences in AI

Preferences are crucial in decision making and have been useful in areas of artificial intelligence (AI) such as scheduling, planning, combinatorial auctions, game playing and multi-agents systems [97]. AI is not the only discipline where preferences are of great interest; it has been studied extensively in various disciplines including operation research, philosophy, economy and psychology [64]. Preferences are fundamental for decision making as most areas of artificial intelligence deal with choice situation [82]. But it is important to consider that the system should be able to understand and support decisions made by users [50].

There have been various preference handling mechanism which exist in AI, and surveys have been conducted to identify the effectiveness of these classical preference techniques. One of such surveys [77] aimed to investigate the existing classical preference methods to know if they have the capability to deal with conflicting situations and represent users' preferences over time. Our study ([77]) identified and investigated (also discussed in 2.2.1) some known preference handling techniques. However, findings show that the existing methods lack the ability to handle the inconsistencies and complexities that exist in preferences, as preferences are known to change over time or clash with each other. For example, a football fan who is also a news enthusiast, may want to watch his favorite team play at 7pm, and there is an important news programme that will be televised at the same 7pm. How can the system support the user in making this decision?

## 3.2 Users preference architecture for argumentation

Figure 3.7 depicts an overall architecture of how the provided argumentation system works in handling users preferences. The system gets information from the external world, including information from sensors and information through web services. This information is represented in the knowledge base (top left area of the figure). Depending on the information the system may detect a conflict during decision making and arguments will support the different options (top right area of the figure). Argument comparison strategies will be triggered (right centre of the figure). The heuristics used to compare arguments is decided by the precedence order which defines a hierarchy amongst the different comparison criteria available to the system (left centre of the figure). If this argument comparison process resorts to user preferences then the User Preference Handling Module analyses the arguments detecting parts of the argument which directly relate to user preferences and needs (lower right part of the figure). The comparison of the arguments based

on user preferences resorts to the User Preference Order (lower left), which in turn when created or modified is based on the User Preferences Ontology (centre left). The User Preferences Ontology can be provided initially by developers. The user preference order can be changed from time to time by the user. User's preferences can be influenced by the external world.

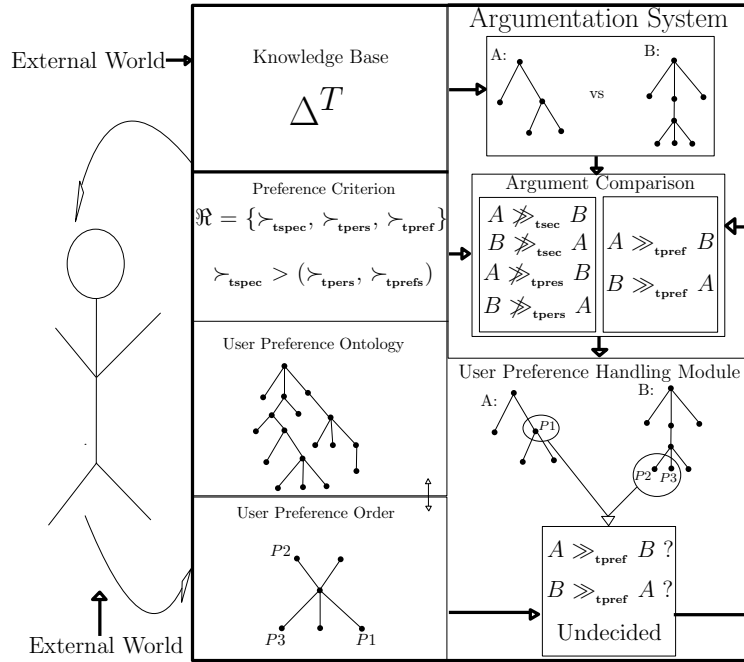


Figure 3.7: Overall preference Architecture

Imagine the argumentation system wants to compare two arguments  $A$  and  $B$  (as shown in the upper part of the diagram). The argument comparison module indicates that the arguments  $A$  and  $B$  are compared with specificity and persistency established to know which is preferred over the other. The output shows that there is no preferable outcome from the two arguments. When arguments are compared (as shown in the User Preference Handling Module), the options are that either one argument is preferred over the other, or it is undecided. One argument can be preferred over the other due to the relative value in preference. For example,  $B$  maybe preferred over  $A$  because the relative value combined of  $P2$  and  $P3$  is greater than that of  $P1$ . We assume  $P1$ ,  $P2$  and  $P3$  can be syntactically or semantically linked to the User Preference Ontology module.

The argumentation theory we introduced in the previous section included sorts  $\mathcal{T}$ ,  $\mathcal{I}$ ,  $\mathcal{N}$ ,  $\mathcal{D}$ ,  $\mathcal{P}$ , and  $\mathcal{A}$ . We introduce a new sort  $\mathcal{P}ref$  which we use to specify *user preferences*. This sort is defined through the User Preferences Ontology. Consequently we extend  $\mathcal{L}^{\mathbb{T}}$  to relate those preferences to time. We will use it in a

similar way as for other sorts, by means of a predicate  $\text{Pref}_{on}(Pr, I)(\text{Pref}_{at}(Pr, i))$  to indicate a preference which applies to a period  $I$  (to an instant  $i$ ).

An agent  $a$  can have multiple preferences, represented with a set,  $\text{Pref}_a = \{pr_1, pr_2, pr_3, \dots\}$  and we assume they can be represented in a partial order  $\mathcal{O}$ . This partial order can produce a structure  $\mathcal{O}(\text{Pref}_a)$ . For example:  $\mathcal{O}(\text{Pref}_a) = (pr_3; pr_1; pr_2)$  meaning  $pr_3$  is preferable to  $pr_1$  and this one to  $pr_2$ , and with  $\mathcal{O}(\text{Pref}_a) = ((pr_1, pr_3); pr_2)$  we can represent that  $pr_1$  and  $pr_3$  are equally preferable and these are preferable to  $pr_2$ . This order in practice will typically be partial, as sometimes we have equal preference over two or more aspects of our lives.

Personal preferences also change over time. However here we do not look in detail at these “belief dynamics”. Instead, we deal with the consequences of those changes as we show in the last example at the end of this chapter. That is, we show that in the case of a change of preferences our system can provide different results, but it does not handle changes of preference itself. We assume there is an interface where a change in preferences can be indicated for a specific agent  $a$  and it translates this change in a recalculation of  $\mathcal{O}(\text{Pref}_a)$ . We assume each agent has at least one preference criterion and the comparison of arguments taking place is for one single agent. For a system considering several users in the same environment, see [68].

We assume a function which measures the relevance of preferences, function  $f_{\text{Pref}}$ , which can be defined in various domain dependent ways. One possible definition is:  $f_{\text{Pref}} : D \rightarrow W$ , where  $D$  is a non-empty set of all possible combinations of  $\mathcal{O}(\text{Pref}_a) \times \langle A, h \rangle$ ,  $\mathcal{O}(\text{Pref}_a)$  is a partial order as explained further up,  $\langle A, h \rangle$  is an argument, and  $W$  is a weight (which can be a number or label). This function takes a set of preferences and an argument and measures the level of preference importance in the argument as follows. Let assume an argument  $\langle A, h \rangle$  where  $A = \{R_1, \dots, R_n\}$  and  $R_i = p_1^i \wedge \dots \wedge p_m^i \supset \text{head}^i$  where  $p_1^i \wedge \dots \wedge p_m^i$  and  $\text{head}^i$  are predicates, some of them possibly of type  $\text{Pref}_{on}(Pr, I)$  or  $\text{Pref}_{at}(Pr, i)$ . We define the *preference weight*  $w_{(pr_j, a)}$  for  $pr_j$  in  $\mathcal{O}(\text{Pref}_a)$  as a number reflecting its level in the partial order. For example we can transform  $((pr_1, pr_3); pr_2)$  into  $\{(2, pr_1), (2, pr_3), (1, pr_2)\}$  reflecting both  $pr_1$  and  $pr_3$  are equally preferable and rank higher in the preferences than  $pr_2$ . We define the *preference weight of a predicate*  $p_j$ ,  $W_p(p_j)$ , where  $1 \leq j \leq m$ , as the weight given for  $p_j$  in  $\mathcal{O}(\text{Pref}_a)$  as above. If  $p_j \notin (\text{Pref}_a)$ s then  $W_p(p_j) = 0$ . Then we define the *preference weight of a rule*  $R_i$ ,  $W_r(R_i)$ ; as the addition of all preference weights of the predicates in its body. Now we can define the *preference weight of an argument*  $\langle A, h \rangle$ ,  $W_a(\langle A, h \rangle)$ ; as the addition of the preference weight of the rules in the argument. That is:  $W_a(\langle A, h \rangle) = \sum_{i=1}^n W_r(R_i)$  and  $W_r(R_i) = \sum_{j=1}^m W_p(p_j^i)$ .

Based on this function which allows us to measure the importance of preferences

taking part in an argument we can define another preference criterion:

**DEFINITION 1** Let  $a$  be an agent,  $\langle A_1, h_1 \rangle$  and  $\langle A_2, h_2 \rangle \in \mathbb{T}\mathbf{A}\mathbf{Struc}(\Delta^{\uparrow\downarrow})$  two arguments and a personal preference measuring function  $f_{pref}$ . Then  $A_1$  for  $h_1$  is *user preferable* than  $A_2$  for  $h_2$  in an instant  $i$  for agent  $a$ , denoted as  $\langle A_1, h_1 \rangle \succ_{\mathbf{U}pref(a)} \langle A_2, h_2 \rangle$ , iff  $f_{Pref_a}(A_1) > f_{Pref_a}(A_2)$ .

Since we have a new way to compare arguments, we have to redefine the *precedence order* between the preference criterion:  $\mathfrak{R} = \{\succ_{\mathbf{t}spec}, \succ_{\mathbf{U}pref(a)}, \succ_{\mathbf{t}pers}\}$ ,  $\succ_{\mathbf{t}spec} > \succ_{\mathbf{U}pref(a)} > \succ_{\mathbf{t}pers}$ . This means we give priority to domain independent criteria.

As the new precedence order indicates the system considers epistemic conflicts first [26] and if no clear choices arise then it tries to disambiguate the situation looking at conflicts at a more practical level. Unusually for traditional AI approaches, the precedence order allows to change that. We discharge all responsibility of the careful use of that resource to the developers. This can be used as an exception handler in extraordinary circumstances. For example, the Intelligent Environments community operates under strong user-centred principles [12] which secure humans rights over the system and reassures the human to be in control of the system and not the other way around [40]. Similar principles have been considered for robotics. As an simple example, consider you live in a smart home or you are driving a smart car, and this Intelligent Environment is behaving erratically, or at least in a way you consider unacceptable. Then you would like to have the right to shut the system off with an order, the system may argue against it, but cannot prevent it, because humans are in control and the human preference should prevail.

### 3.3 Modelling different users

To illustrate how our system works we assume a smart home with a light management system that is capable of understanding the activities in a room, so as to make reasonable decisions for an inhabitant named Sara. This thesis will be considering a complex description involving three aspect of Sara’s life: lighting, entertainment and health management. the thesis will also be considering a description involving the health management aspect of Joe’s (Sara’s son) life.

#### 3.3.1 Modelling Sara’s preferences

*Sara is a 65 years old woman living in a smart environment. She would like the system to turn the lights off any time she leaves home and forget to switch off the light. Sara still want the system to be aware of her*



*health circumstances, and provide her with information on food consumption especially her favourite brown-cake which she buys online, despite being diabetic. The system should further manage Sara's television programmes, making suggestions on potentially interesting programmes.*

The above description provides a complex problem to deal with. The light, health and television programme scenario offers three ways of representing users' preferences. The rest of this section will illustrate how argumentation will deal with these scenarios.

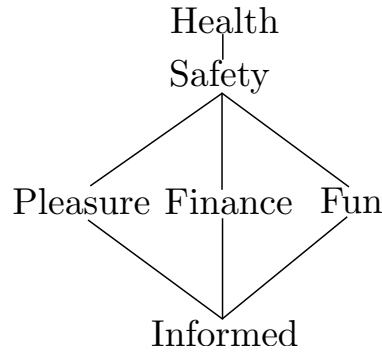


Figure 3.8: Ranking of Sara's Preferences

According to Figure 3.8 which depicts Sara's ranking of life style choices, it was assume for her that health is more important than safety, and safety more important than pleasure, finance and fun (all of them with equal level of importance) and those are more important than being informed (news). Then that can be represented in the system, using the motion introduced in Section 3.2, as follows:

$$\begin{aligned}
 Pref_{Sara} &= \{ \textit{finance}, \textit{informed}, \textit{safety}, \\
 &\quad \textit{health}, \textit{fun}, \textit{pleasure} \} \\
 \mathcal{O}(Pref_{Sara}) &= \{ (4, \textit{health}), \\
 &\quad (3, \textit{safety}), \\
 &\quad (2, \textit{pleasure}), (2, \textit{finance}), (2, \textit{fun}), \\
 &\quad (1, \textit{informed}) \}
 \end{aligned}$$

where a pair  $(N, P)$  indicates the value of preference weight  $N$  for a preference  $P$ .

### Light Scenario for Sara

Table 3.7 shows the development of the light scenario through time. The next set of rules are extracted from  $\Delta^{\text{T}\downarrow}$ :

Table 3.7: Lighting Scenario World Dynamics

$MEETS(I_0, I_1) \wedge MEETS(I_1, I_2) \wedge MEETS(I_2, I_3)$				
$ Holds_{on}(Movement, I_0) \wedge \neg Holds_{on}(Sleeping, I_0) \wedge \neg Holds_{on}(OnBed, I_0) \wedge Holds_{on}(Home, I_0) \wedge Holds_{on}(LightsOn, I_0)$				
<b>Lighting Scenario</b>	Movement	$\neg$ Movement	$\neg$ Movement	$\neg$ Movement
	$\neg$ Sleeping	$\neg$ Sleeping	$\neg$ Sleeping	$\neg$ Sleeping
	$\neg$ OnBed	$\neg$ OnBed	$\neg$ OnBed	$\neg$ OnBed
	Home	Home	$\neg$ Home	$\neg$ Home
	LightsOn	LightsOn	LightsOn	$\neg$ LightsOn
<b>Transition Cause</b>	$Do_{on}(LeavingHome, I_0)$	System Inference from: L-R3	$Occurs_{at}(\text{System TurnsLightOff}, end(I_2))$	
	$I_0$	$I_1$	$I_2$	$I_3$

$$MEETS(I_0, I_1) \wedge MEETS(I_1, I_2) \wedge MEETS(I_2, I_3)$$

$$Holds_{on}(Movement, I_0) \wedge \neg Holds_{on}(Sleeping, I_0) \wedge \neg Holds_{on}(OnBed, I_0) \wedge Holds_{on}(LightsOn, I_0)$$

- L-R1:  $Do_{on}(LeavingHome, I_0) \succ - Occurs_{at}(LeftHome, begin(I_1))$   
L-R2:  $Occurs_{at}(LeftHome, begin(I_1)) \succ - Holds_{on}(Movement, I_1)$   
L-R3:  $\neg Holds_{on}(Movement, I_1) \wedge Length(I_1) > 15 \wedge \neg Holds_{on}(OnBed, I_1) \succ - Holds_{on}(Home, I_2)$   
L-R4:  $\neg Holds_{on}(Home, I_2) \succ Pref_{on}(LightOff, I_2)$   
L-R5:  $Pref_{on}(LightOff, I_2) \succ Occurs_{at}(SystemTurnsLightOff, end(I_2))$   
L-R6:  $Occurs_{at}(SystemTurnsLightOn, end(I_2)) \succ - Holds_{on}(LightsOff, I_3)$

*Argument for LightsOn@I<sub>3</sub>*: As seen from the initial facts, the lights are on, as Sara is in the room. So because of persistency, there is a possibility that the lights will remain on.

$$L.On = \langle \{ Holds_{on}(LightsOn, I_0) \wedge \text{notChange}_{in}^{+-}(LightsOn, [end(I_0), end(I_3)]) \succ - Holds_{on}(LightsOn, I_3) \}, Holds_{on}(LightsOn, I_3) \rangle$$

The argument is reflected in figure 3.9B.

*Argument for  $\neg$ LightsOn@I<sub>3</sub>*: Considering an alternative explanation, given that the system has been programmed to understand when the lights are not needed. The argument indicates that Sara is leaving home at  $I_0$  and is out of home at beginning of  $I_1$ . As a result of this no movements were detected from there onwards. If continued for the next 15 minutes and there is no pressure on the bed at the same time, the system has reasons to believe that Sara is not at home at  $I_2$ . When Sara is not at home over that period, she usually prefers the lights off. So at that moment,

the system infers that it is reasonable to turn the lights off. As a result, the lights are off at  $I_3$ .

$$\begin{aligned}
L.Off = \langle & \{ \text{Do}_{on}(\text{LeavingHome}, I_0) \succ - \text{Occurs}_{on}(\text{LeftHome}, I_1), \\
& \text{Occurs}_{on}(\text{LeftHome}, I_1) \succ - \text{Holds}_{on}(\text{Movement}, I_1), \\
& - \text{Holds}_{on}(\text{Movement}, I_1) \wedge \text{Length}(I_1) > 15 \wedge \\
& - \text{Holds}_{on}(\text{OnBed}, I_1) \succ - \text{Holds}_{on}(\text{Home}, I_2), \\
& - \text{Holds}_{on}(\text{Home}, I_2) \succ - \text{Pref}_{on}(\text{LightsOff}, I_2), \\
& \text{Pref}_{on}(\text{LightsOff}, I_2) \\
& \succ - \text{Occurs}_{on}(\text{SystemTurnsLightOff}, I_2), \\
& \text{Occurs}_{on}(\text{SystemTurnLightOff}, I_2) \\
& \succ - \text{Holds}_{on}(\text{LightsOn}, I_3) \}, \\
& - \text{Holds}_{on}(\text{LightsOn}, I_3) \rangle
\end{aligned}$$

The argument is depicted in figure 3.9A

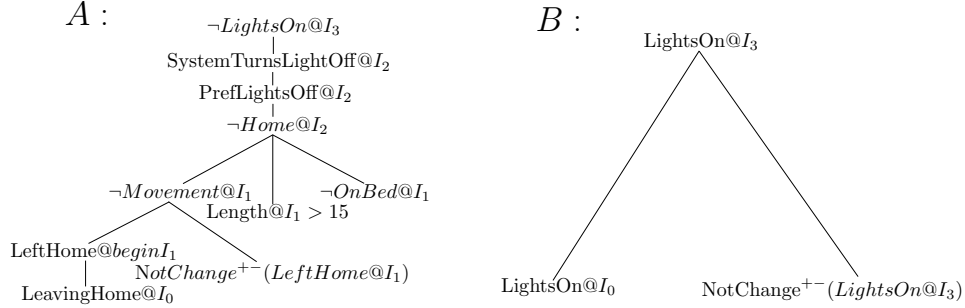


Figure 3.9: Argumentation Trees for Sara's Light Scenario

From Sara's light scenario, there are two main contending arguments,  $L.On \bowtie_T L.Off$ . Neither specificity nor persistency can be applied and we will explain how the system applies users' preferences to decide. Note  $L.On$  is based on persistency rule  $P$  and  $W_r(P) = 0$  because there is no preference predicate contained in  $P$ , therefore  $W_{Sara}(L.On) = 0$ .

Argument  $L.Off$  is based on rules L-R1, L-R2, L-R3, L-R4, L-R5, L-R6 and  $W_r(L-R1) = 0$ ,  $W_r(L-R2) = 0$ ,  $W_r(L-R3) = 0$ ,  $W_r(L-R4) = 0$ ,  $W_r(L-R6) = 0$ . Now  $W_r(L-R5) = V$ , where  $V$  indicates the value of preference of having the lights off. Lights off is not explicitly mentioned in Sara's preference ranking in Figure 3.8, we assume that the general preference ontology (as seen in lower left of figure 3.7) contains the information that connects lights off and  $Finance \subset Pref_{Sara}$ . According to  $\mathcal{O}(Pref_{Sara})$ ,  $W_p(Finance) = 2$ , so  $W_r(L-R5) = 2$ . Now we can calculate the weight for the argument which is  $W_{Sara}(L.Off) = 0 + 0 + 0 + 0 + 2 + 0 = 2$ .

$L.Off \succ_{U_{Pref}(Sara)} L.On$  because Sara is not at home and from a financial point of view she prefers the lights off. Therefore,  $L.Off \gg_{\text{tdef}} L.On$ .

Table 3.8: Television Scenario World Dynamics

$MEETS(I_0, I_1) \wedge MEETS(I_1, I_2)$			
$ Holds_{on}(WatchTV, I_0) \wedge \neg Holds_{on}(WatchingNews, I_0) \wedge \neg Holds_{on}(WatchingSports, I_0)$			
<b>Television Scenario</b>	WatchTV	WatchTV	WatchTV
	$\neg$ WatchingNews	$\neg$ WatchingNews	$\neg$ WatchingNews
	$\neg$ WatchingSports	WatchingSports	WatchingSports
<b>Transition Cause</b>	$Occurs_{on}(FootballMatch, I_0)$	$Occurs_{on}(DisastrousEvent, I_1)$	
	$I_0$	$I_1$	$I_2$

### Television Scenario for Sara

Table 3.8 shows the development of the television scenario through time. The next set of rules are extracted from  $\Delta^{\text{T}\downarrow}$  :

$MEETS(I_0, I_1) \wedge MEETS(I_1, I_2)$

$Holds_{on}(WatchTV, I_0) \wedge \neg Holds_{on}(WatchingNews, I_0) \wedge \neg Holds_{on}(WatchingSports, I_0)$

T-R1:  $Occurs_{on}(DisastrousEvent, I_0) \succ\text{---} Pref_{on}(WatchingNews, I_1)$

T-R2:  $Pref_{on}(WatchingNews, I_1) \wedge Holds_{on}(WatchTV, I_1) \succ\text{---} Do_{on}(WatchingNews, I_2)$

T-R3:  $Occurs_{on}(FootballMatch, I_0) \succ\text{---} Pref_{on}(WatchingSports, I_1)$

T-R4:  $Pref_{on}(WatchingSports, I_1) \wedge Holds_{on}(WatchTV, I_1) \succ\text{---} Do_{on}(WatchingSports, I_2)$

*Argument for Watching News at  $I_2$ :* From the initial facts, there are reasons to believe that Sara will watch the news at  $I_2$ . The reason to believe this is because, when a disastrous (important) event occurs, she will prefer to watch news. If Sara watches television at  $I_1$  and a disastrous event happens at  $I_1$ , the system infers that she prefers watching news at  $I_2$ .

$$N = \langle \{Occurs_{on}(DisastrousEvent, I_0) \succ\text{---} Pref_{on}(News, I_1), \\ Pref_{on}(News, I_1) \wedge Holds_{on}(WatchTV, I_1) \succ\text{---} \\ Do_{on}(WatchingNews, I_2)\}, \\ Do_{on}(WatchingNews, I_2) \rangle$$

Figure 3.10A represents the above argument.

*Argument for Watching Sports at  $I_2$ :* An alternative explanation shows why Sara will be watching the Sports.  $I_0$  indicates that there is a football match going on, and the system is aware that Sara is a football fan. So when Sara is watching television at  $I_1$  and prefers to watch sport because there is a football event going on, the system will believe that Sara will prefer watching sports at  $I_2$ .

$$S = \langle \{ \text{Occurs}_{on}(\text{FootballMatch}, I_0) \succ \text{Pref}_{on}(\text{Sports}, I_1), \\ \text{Pref}_{on}(\text{Sports}, I_1) \wedge \text{Holds}_{on}(\text{WatchTV}, I_1) \\ \succ \text{Do}_{on}(\text{WatchingSports}, I_2) \}, \\ \text{Do}_{on}(\text{WatchingSports}, I_2) \rangle$$

This argument is shown in figure 3.10B.

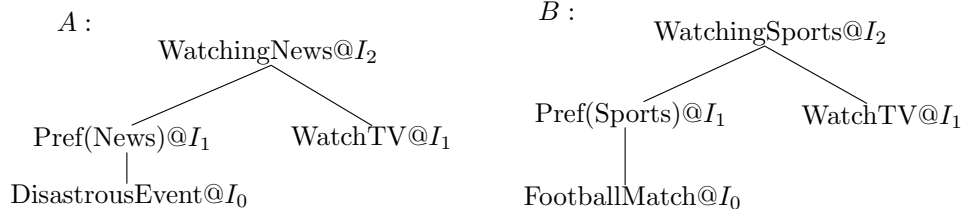


Figure 3.10: Argumentation Trees for Sara's Television Scenario

From Sara's Television scenario, there are two main contending arguments,  $N \bowtie_{\mathbb{T}} S$ . Neither specificity nor persistency can be applied and we will explain how the system applies users' preferences to decide.

$N$  is based on two rules T-R1 and T-R2,  $W_r(\text{T-R1}) = 0$  because there is no preference predicate contained in the antecedent of T-R1. However,  $W_r(\text{T-R2}) = V$  where  $V$  measures the level of preference for watching news. Watching news or watching sports is not explicitly mentioned in Sara's preference ranking in figure 3.8, we assume that the general preference ontology (as seen in lower left of figure 3.7) contains the semantic knowledge that connect watching news to being "Informed" and watching sport to "Fun", both in  $\text{Pref}_{Sara}$ . In this case  $W_r(\text{T-R2}) = 1$ , therefore  $W_{Sara}(N) = 0 + 1 = 1$ .

Argument  $S$  is based on two rules T-R3 and T-R4, so  $W_r(\text{T-R3}) = 0$  because there is no preference predicate contained in in the antecedent of T-R3. Although,  $W_r(\text{T-R4}) = V$  where  $V$  measures the level of preference for watching sports. Watching sport is not explicitly mentioned in Sara's preference ranking in figure 3.8, we assume that the general preference ontology (as seen in lower left of Figure 3.7) contain information that connects watching sport to "Fun" indicated in  $\text{Pref}_{Sara}$ . In this case  $W_r(\text{T-R4}) = 2$ , therefore  $W_{Sara}(S) = 0 + 2 = 2$ .

From Sara's television scenario,  $N \bowtie_{\mathbb{T}} S$ ,  $S \succ_{\text{Upref}(Sara)} N$  because in Sara's preference ranking, pleasure and fun have priority over being informed. Therefore,  $S \gg_{\text{tdef}} N$ .

### Health Scenario for Sara (Buying Cake Online)

Table 3.9 shows the development of the health scenario through time. The next set of rules are extracted from  $\Delta^{\mathbb{T}\downarrow}$ :

Table 3.9: Health Scenario World Dynamics

$MEETS(I_0, I_1) \wedge MEETS(I_1, I_2) \wedge MEETS(I_2, I_3)$				
$\neg Holds_{on}(BuyCake, I_0) \wedge Holds_{on}(Diabetic, I_0) \wedge \neg Holds_{on}(HighSugar, I_0) \wedge$ $\neg Holds_{on}(CakeOnSales, I_0)$				
<b>Health Scenario</b>	$\neg BuyCake$	$\neg BuyCake$	$\neg BuyCake$	$\neg BuyCake$
	Diabetic	Diabetic	Diabetic	Diabetic
	$\neg HighSugar$	$\neg HighSugar$	HighSugar	HighSugar
	$\neg CakeOnSales$	CakeOnSales	CakeOnSales	CakeOnSales
<b>Transition Cause</b>	$Occurs_{on}(CakeOnSales, I_0)$	$Occurs_{at}(HighSugarDetected, end(I_1))$	$Occurs_{on}(SystemAdvicesNotBuyCake, I_3)$	
	$I_0$	$I_1$	$I_2$	$I_3$

$MEETS(I_0, I_1) \wedge MEETS(I_1, I_2) \wedge MEETS(I_2, I_3)$   
 $\neg Holds_{on}(BuyCake, I_0) \wedge Holds_{on}(Diabetic, I_0) \wedge$   
 $\neg Holds_{on}(CakeOnSales, I_0) \wedge \neg Holds_{on}(HighSugar, I_0)$

- H1-R1:  $Occurs_{on}(CakeOnSales, I_0)$   
 $\succ \neg Holds_{on}(CakeOnSales, I_1)$   
H1-R2:  $Holds_{on}(CakeOnSales, I_1) \wedge Pref_{on}(Pleasure, I_1)$   
 $\succ Holds_{on}(BuyCake, I_1)$   
H1-R3:  $Occurs_{on}(HighSugarDetected, end(I_1))$   
 $\succ Holds_{on}(HighSugar, I_2)$   
H1-R4:  $Holds_{on}(Diabetic, I_2) \wedge Holds_{on}(HighSugar, I_2) \wedge$   
 $Holds_{on}(CakeOnSales, I_2) \wedge Pref_{on}(Health, I_2)$   
 $\succ Occurs_{on}(SystemAdvicesNotBuyCake, I_3)$   
H1-R5:  $Occurs_{on}(SystemAdvicesNotBuyCake, I_3)$   
 $\succ \neg Holds_{on}(BuyCake, I_3)$

*Argument for Buying Cake at  $I_3$ :* As seen from the initial facts, Sara is not buying cake at that moment. The Argument  $BC$  expresses the possibility of her buying cake at  $I_1$ , as the argument shows that she prefers to buy cake when on sale.

$$BC = \{ \{ Occurs_{on}(CakeOnSales, I_0) \\ \succ \neg Holds_{on}(CakeOnSales, I_1), \\ Holds_{on}(CakeOnSales, I_1) \wedge Pref_{on}(Pleasure, I_1) \\ \succ Holds_{on}(BuyCake, I_1) \}, \\ Holds_{on}(BuyCake, I_1) \}$$

Due to persistency the system will advice to buy cake at  $I_2$  and  $I_3$ . This is shown in Figure 3.11A.

*Argument for not Buying Cake at  $I_3$ :* Having considered the initial facts that the user is diabetic and this time she has a high sugar level, the system will infer that Sara will not buy cake at  $I_3$ . This is because her ranking in figure 3.8 indicates that

Sara is more concerned about her health compared to her other preferences. This will better inform the system in understanding that Sara's health is a priority and it will give the system reasons to believe that she will not buy cake and will also suggest to the user against buying the cake.

$$\begin{aligned}
\neg BC = \{ & \{ \text{Occurs}_{on}(\text{HighSugarDetected}, \text{end}(I_1)) \\
& \quad \succ \text{Holds}_{on}(\text{HighSugar}, I_2), \\
& \text{Holds}_{on}(\text{Diabetic}, I_2) \wedge \text{Holds}_{on}(\text{HighSugar}, I_2) \wedge \\
& \quad \text{Holds}_{on}(\text{CakeOnSales}, I_2) \wedge \text{Pref}_{on}(\text{Health}, I_2) \\
& \quad \succ \text{Occurs}_{on}(\text{SystemAdvicesNotBuyCake}, I_3), \\
& \text{Occurs}_{on}(\text{SystemAdvicesNotBuyCake}, I_3) \\
& \quad \succ \neg \text{Holds}_{on}(\text{BuyCake}, I_3), \\
& \quad \neg \text{Holds}_{on}(\text{BuyCake}, I_3) \} \}
\end{aligned}$$

This argument is depicted in Figure 3.11B.

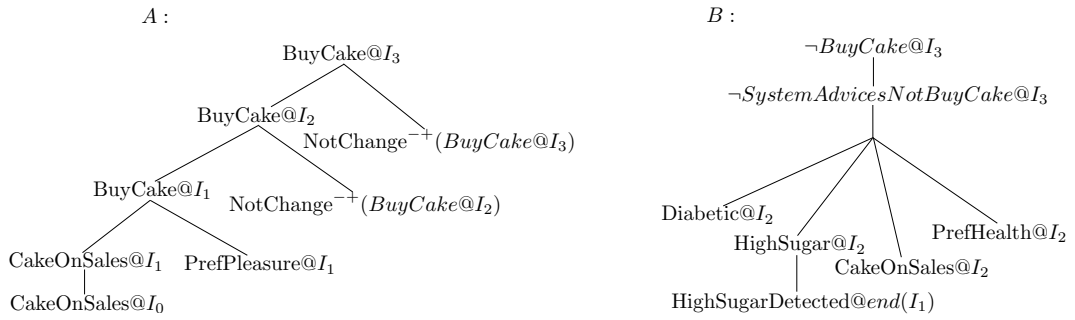


Figure 3.11: Argumentation Trees for Sara's Health Scenario

From Sara's Health scenario, there are two main contending arguments,  $BC \bowtie_{\mathbb{T}} \neg BC$ . Neither specificity nor persistency can be applied and we will explain how the system uses users preferences to decide.  $BC$  is based on two rules H1-R1 and H1-R2, so  $W_r(\text{H1-R1}) = 0$  because there is no preference predicate contained in the antecedent of H1-R1. However,  $W_r(\text{H1-R2}) = V$  where  $V$  measures the level of preference for pleasure as indicated in  $\mathcal{O}(\text{Pref}_{Sara})$ , in this case  $W_r(\text{H1-R2}) = 2$  and  $W_{Sara}(BC) = 0 + 2 = 2$ .

$\neg BC$  is based on three rules H1-R3, H1-R4 and H1-R5, with  $W_r(\text{H1-R3}) = 0$  and  $W_r(\text{H1-R5}) = 0$  because there is no preference predicate contained in H1-R3 nor in H1-R5. However,  $W_r(\text{H1-R4}) = V$  where  $V$  measures the level of preference for health as indicated in  $\mathcal{O}(\text{Pref}_{Sara})$ . In this case  $W_r(\text{H1-R4}) = 4$ , therefore  $W_{Sara}(\neg BC) = 0 + 4 + 0 = 4$ .

From Sara's health scenario,  $BC \bowtie_{\mathbb{T}} \neg BC$ ,  $\neg BC \succ_{\text{Upref}(Sara)} BC$  because in Sara's ranking preference, her health and safety are of higher priority than her other preferences. Therefore,  $\neg BC \gg_{\text{tdef}} BC$ .

Table 3.10: Joe’s Health Scenario World Dynamics

$MEETS(I_0, I_1) \wedge MEETS(I_1, I_2)$			
$\neg Holds_{on}(BuyCake, I_0) \wedge \neg Holds_{on}(Diabetic, I_0) \wedge \neg Holds_{on}(HighSugar, I_0) \wedge \neg Holds_{on}(CakeOnSales, I_0)$			
<b>Television Scenario</b>	$\neg BuyCake$	$\neg BuyCake$	$BuyCake$
	$\neg Diabetic$	$\neg Diabetic$	$\neg Diabetic$
	$\neg HighSugar$	$\neg HighSugar$	$\neg HighSugar$
	$\neg CakeOnSales$	$CakeOnSales$	$CakeOnSales$
<b>Transition Cause</b>	$Occurs_{on}(CakeOnSales, I_0)$	System Inference from: H2-R2	
	$I_0$	$I_1$	$I_2$

### 3.3.2 Modelling Joe’s Preferences

*Sara has a teenage son, Joe, who cares about pleasure and fun above everything else. He also likes being informed. He prefers being informed over health, safety and finance.*

Figure 3.12 depicts Joe’s preference ranking.

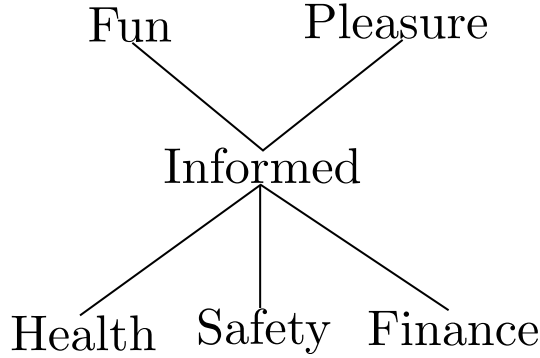


Figure 3.12: Ranking of Joe’s Preferences

Next we represent Joe’s preferences using the notation introduced in section 3.2.

Then we can represent this preference ranking in our system as follows:

$$\begin{aligned}
 Pref_{Joe} &= \{ \textit{finance}, \textit{informed}, \textit{safety}, \\
 &\quad \textit{health}, \textit{fun}, \textit{pleasure} \} \\
 \mathcal{O}(Pref_{Joe}) &= \{ (3, \textit{fun}), (3, \textit{pleasure}), \\
 &\quad (2, \textit{informed}), \\
 &\quad (1, \textit{health}), (1, \textit{safety}), (1, \textit{finance}) \}
 \end{aligned}$$

Table 3.10 shows the development of Joe’s health scenario through time. The next set of rules are extracted from  $\Delta^{\mathbb{T}\downarrow}$ :



$MEETS(I_0, I_1)$   
 $\neg Holds_{on}(BuyCake, I_0) \wedge \neg Holds_{on}(Diabetic, I_0) \wedge$   
 $Holds_{on}(CakeOnSales, I_0) \wedge \neg Holds_{on}(HighSugar, I_0)$

H2-R1:  $Holds_{on}(CakeOnSales, I_1) \wedge Pref_{on}(Finance, I_1)$   
 $\succ - \neg Holds_{on}(BuyCake, I_2)$

H2-R2:  $Holds_{on}(CakeOnSales, I_1) \wedge Pref_{on}(Pleasure, I_1)$   
 $\succ - Holds_{on}(BuyCake, I_2)$

*Arguments for Joe's not BuyingCake at I<sub>2</sub>:* From the initial facts and also his preference ranking in figure 3.12, it shows that Joe cares less about finance compared to pleasure. If the cake is on sale and buying cake requires spending money, and finance is one of the concerns for Joe this could be a reason not to buy the cake at I<sub>1</sub>.

*Arguments for Joe's BuyingCake at I<sub>2</sub>:* Figure 3.12 also indicates that Joe has a high preference for pleasure, for example eating chocolate cake is something he enjoys. This provides a reason for Joe to buy the cake.

$\neg BC_j = \langle \{ Holds_{on}(CakeOnSales, I_0) \wedge Pref_{on}(Finance, I_0)$   
 $\succ - Holds_{on}(\neg BuyCake, I_1) \},$   
 $Holds_{on}(\neg BuyCake, I_1) \rangle$

This argument is shown in Figure 3.13A.

$BC_j = \langle \{ \neg Holds_{on}(CakeOnSales, I_1) \wedge Pref_{on}(Pleasure, I_1)$   
 $\succ - Holds_{on}(BuyCake, I_2) \},$   
 $Holds_{on}(BuyCake, I_2) \rangle$

This argument is illustrated in figure 3.13B.

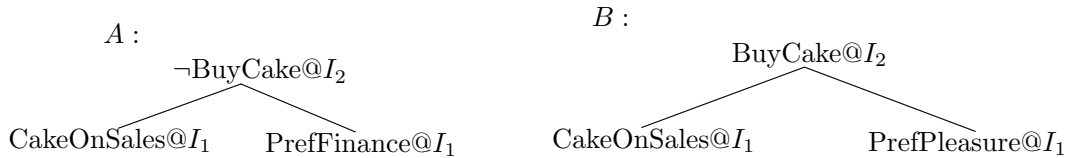


Figure 3.13: Argumentation Tree for Joe's Health Scenario

From Joe's health scenario, there are two main contending arguments,  $\neg BC_j \bowtie_{\top} BC_j$ . Neither specificity nor persistency can be applied and we will explain how the system decides using preferences.  $\neg BC_j$  is based on one

rule H2-R1, and  $W_r(H2 - R1) = 1$  according to the value of *Finance* in  $\mathcal{O}(Pref_{Joe})$ . In this case  $W_r(H2 - R1) = 1$ , then  $W_{Joe}(\neg BC_j) = 1$ .

$BC_j$  is also based on one rule  $W_r(H2 - R2) = 3$  according to the value of pleasure in  $\mathcal{O}(Pref_{Joe})$ . Therefore, in this case  $W_{Joe}(\neg BC_j) = 3$ .

From Joe's health scenario,  $BC_j \bowtie_{\mathbb{T}} \neg BC_j$ ,  $BC_j \succ_{\mathbf{U}^{pref}(Joe)} \neg BC_j$  because *Pleasure* is of higher priority for Joe compared to *Finance*. Therefore,  $BC_j \gg_{\mathbf{tdef}} \neg BC_j$ .

Useful to provide a discussion of soundness/completeness and preservation of these properties when the reasoning system is extended with preferences.

Providing the practical solution to reason with user preferences in a smart environment, the research adopted an existing monotonic reasoning system (MReasoner) and its properties [45]. The aim is to extend the MReasoner with argumentation temporal language  $\mathcal{L}^{\mathbb{T}}$  (which gives the system ability to handle inconsistencies and deal with time) and add user's preference mechanism (*Pref*) to solve conflicting user's preferences. Nonetheless, the notations and properties of the MReasoner were preserved, as it ensured less complexity in the transition and enhancement process in developing the practical solution. The properties of the existing monotonic reasoning system is equipped enough to respond soundly to requests and trigger necessary outputs to sensors and actuators in a smart home settings. The trigger happens due to an occurrence of event(s) and human actions, but the MReasoner does not in any way handle inconsistencies, complexities in user's preferences nor deal with time. To produce a system that handles such desirable features, additional system features are required. These were discussed previously in this chapter.

Next chapter introduces the produced practical system, called Hybrid System. However, the first section of the chapter, produces a brief introduction of the MReasoner, depicting how the system can execute specification file in different modes, either in simulation mode or tracking environmental conditions and acting upon them. The next section (Section 4.2.1) highlights a few modelling/transition processes of some key properties and notations required for extending the MReasoner. The modelling was validated using a lighting scenario within a smart home which was conducted in a smart space lab located within the Middlesex University premises. The chapter also introduces the produced preference management tool (preference interface), which users utilise to prioritize their preferences. The functionality of the interface has been validated, as it was used to affect the Hybrid System's behaviour. The effectiveness of the Hybrid System and the interface were also validated with few other smart environment scenarios, including using the system to access a supermarket chain store, filter their products and advise users accordingly.

## Chapter 4

# Research work and validation

This chapter consist of the practical system developed to complement an existing reasoning system that lacks some core capabilities, especially in handling conflicting situations. The features of the developed system, which we refer to as Hybrid System, are highlighted in comparison to the reasoning system (MReasoner).

The research models the argumentation language to the language our practical solution understands. The modelling process from argumentation language to implementation language is not automated, so we clarify how the modelling is done using a scenario, and applying specific guidelines.

To validate the effectiveness of the implemented system, a real environment and the necessary equipment are required. This chapter discusses some of the infrastructure and equipment used for demonstration. We also present and discuss the preference management tool (interface) which users use to rank their preferences. We use the simplified interface to provide a demonstration of how a change of preference ranking can affect the system behaviour.

The validation comprises of some real-world scenarios that were adopted in validating the system, and these scenarios were adopted for few reason. First, they were from the perspective of an intelligent environment, secondly, the scenarios aimed to deal with some specific challenges discussed earlier (section 1.1). Also, the scenarios were motivated based on the resources within the research reach, as there was limited access to bringing users to the Smart Lab for validations. Every scenario applied in the validating process uses realistic intelligent environment situations including system real-time analysis of conflicting objectives and adaptation to user's preferences.

The last section of this chapter consists of three illustrations, along with links to each video demonstration. The first illustration is of the Hybrid System, showing its overall working, from loading a specification file, to analysing the file for potential

conflict, to solving the detected conflict. The Hybrid System is used to trigger and solve the three types of potential conflict discussed in this research, that exist in the area of preference criteria. This is to show that our system is capable of identifying any of the potential conflicts at any time. Using live data from a supermarket chain store (Tesco) in the UK, we are able to validate our system’s ability to access external data in the third scenario. The Hybrid System filters Tesco’s grocery products (cakes), identifies those that contain sugar, and advise a diabetic patient accordingly, depending on their preference ranking of “Health” and “Pleasure”.

#### 4.1 Hybrid System (A system for real-time Decision Making)

One crucial aspect of providing intelligent systems is the specification of what behaviour the system needs to exhibit intelligently [57]. This is the aim of the research, and leads to the development of a suitable application for an ambient intelligent system, that represents and reasons with users’ preferences and needs. A simple interface to compliment the core system is also produced, as the interface has the ability to influence the system behaviour according to changes made by the user. Before further discussion, we want to clarify the term “*Hybrid System*”, as the name used to refer to the core practical system. This includes major integration of the “Argumentation System” (non-monotonic reasoner) and “User’s Preference” mechanism with the existing MReasoner (strictly monotonic reasoner).

Our latest Hybrid System is developed to complement MReasoner (shown in Figure 4.2) as a way of extending its capabilities, as the MReasoner tool cannot manage to solve conflicting situations in an intelligent environment in real time. The resulting Hybrid System can do both types of reasoning, monotonic and non-monotonic. When no conflicts are present in the rule base, it uses the simpler lightweight MReasoner, but when there are rules with opposing conclusions, argumentation is used. The features of the Hybrid System include:

- Selects a specification file containing rules from any location on the computer.
- Analyzes the selected specification file using a conflict analyzer algorithm to identify potential conflicts.
- Displays potential conflicts, if any are identified.
- If no potential conflicts are detected, the Hybrid System passes the file to the

existing reasoning tool (“MReasoner”), if potential conflict is detected, the Hybrid System passes the file to the conflict solver tool (“Argumentation Solver”).

- During execution, Hybrid System has the ability to generate current results of all the properties involved in the execution.
- Conflict(s) detected during the execution process will be solved by the argumentation solver. The Hybrid System will display the new results on the interface, which will take effect around the environment at run time.
- The area(s) (instant or interval) where conflict(s) were identified and solved, will be highlighted by the Hybrid System for clarity.
- The Hybrid System also has the ability to explain how the conflict(s) were solve. Clicking on any of the highlighting cell from the result table, will display the reason of how the conflict was solved in the text area of the Hybrid System interface.

Illustration of the Hybrid System at work is provided in section 4.5, as we provide some demonstration of the system in real time, applying some real scenarios and live data.

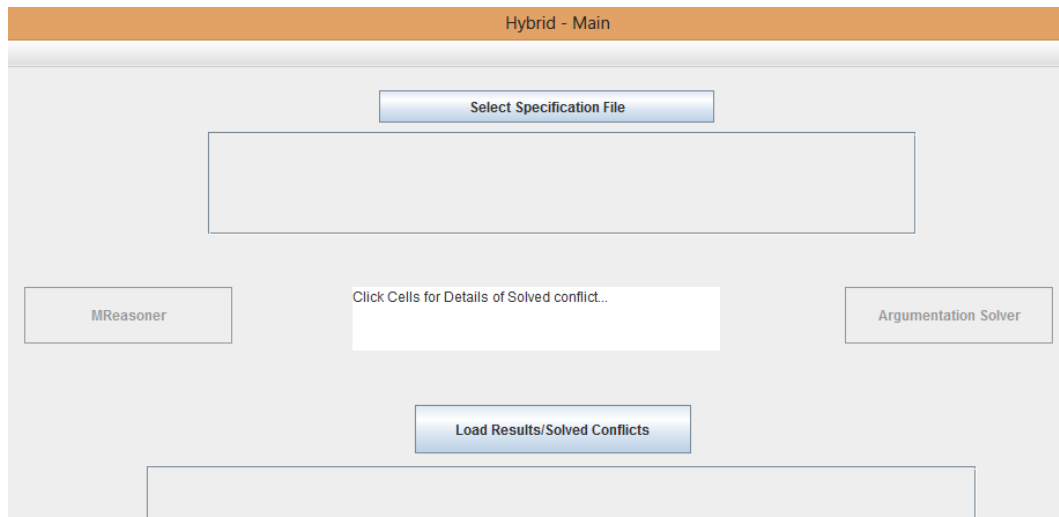


Figure 4.1: Hybrid Interface

#### 4.1.1 Reasoning System (MReasoner)

The reasoning system ( $M$ ) was developed (interface shown in Fig. 4.2) based on natural characteristics of reactive environments, as it has the ability to track certain

environment conditions and act upon them.  $M$  also has the capabilities to capture states happening during time intervals. For example if there is no movement in the last 15 seconds, turn “off” the lights in the room. However,  $M$  lacks the ability to handle conflicting outcomes. For example, if someone is doing yoga, do not turn “off” the lights.

The reasoning system ( $M$ ) is a rule based system aimed at handling simple causality, but has been extended to handle some practical uncertainties and complexities, especially conflicts in user preferences. We extended the  $M$  system by using argumentation to improve the capability of detecting and solving conflicts that occur within an intelligent environment. The argumentation solver accepts the specification file from the Hybrid System containing detected potential conflict(s). Conflicts get solved by the argumentation solver following the order of precedence of preference criteria discussed in Section 3.1.3. The specification file have to be written in an exact format that is acceptable by the Hybrid System for execution. The specification file format and the execution types are discussed in Section 4.1.2.

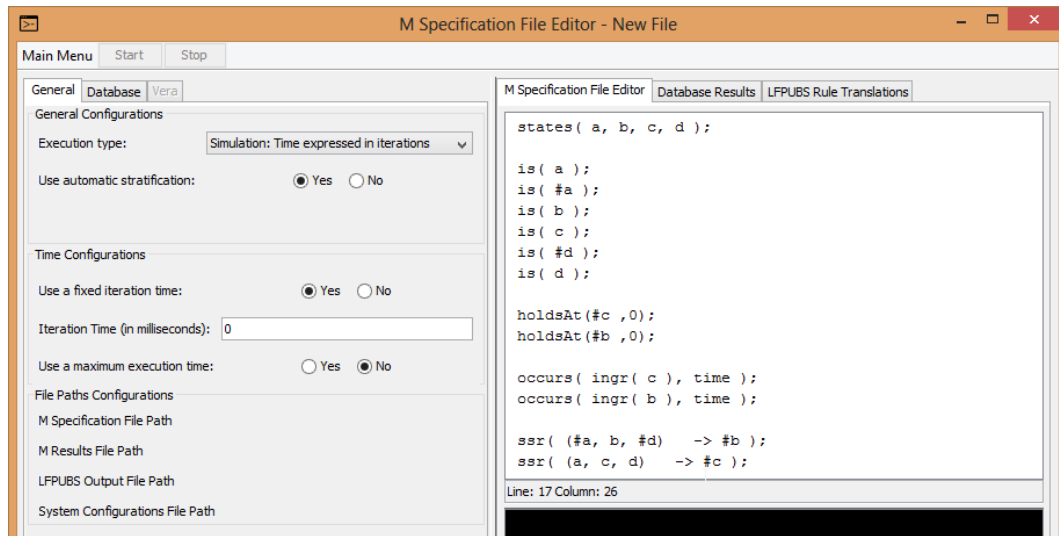


Figure 4.2: Reasoning System Interface.

## 4.1.2 Specification File

Figure 4.3 illustrates the specification file sample (and added some labels for clarity) of the reasoning system. The specification file has to be in that format, but also depends on the type of execution the reasoning system is running. The execution types that can be simulated by the reasoning system include:

- Simulation expressed in iteration

- Simulation expressed in real time
- Simulation (execution) in real environment, with sensors and actuators

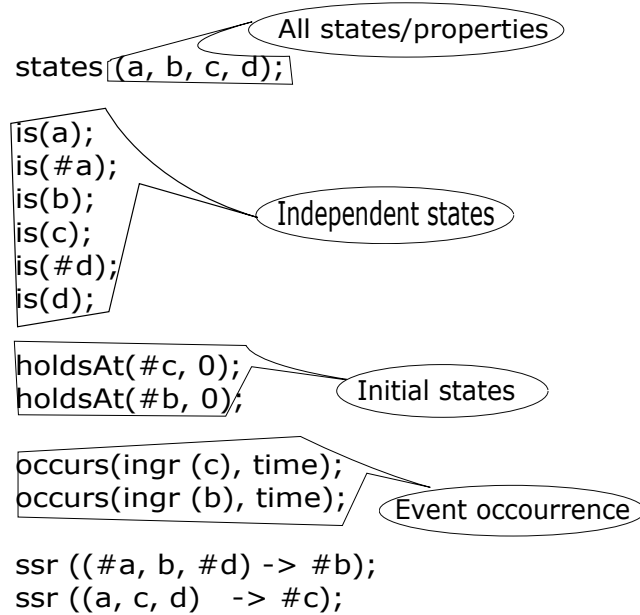


Figure 4.3: Specification File Format Sample

The first part of Fig. 4.3 consists of “all the properties (states)”, any property that will be used during execution must be specified in the first part. The second part consists of the declaration of “Independent States”, which does not depend on other states causally. The “#” symbol (when placed in front of a state) denotes that a property is false. An example of applying the # symbol can be “*is(#Movement)*”, which can be used to represent ‘no movement’ detected. The next part which is the “Initial State” (as seen in Fig. 4.3), signifies initializing the state. For instance, *holdsAt(#Movement,0)* indicate that at the start of the iteration, the movement will be “off” or false, this is absence of movement is assured.

The fourth part known as “Event Occurrence” (as shown in Fig. 4.3), are events used to impinge the system from the outside; it can be sensors being triggered or via human behaviour commands. All of this notation was first defined as part of the “*C*” language in [46], then the language was created by adding metric temporal operators to “*C*” [57]. However, the event representation (*occurs(ingr([#]s), t\*)* “*s*” signifies state and “*t*” signifies time) can only be used depending on the type of execution simulated by the reasoning system.

The “Simulation expressed in iteration” executes the specification file of the reasoning system based on the number of iterations specified, and the exe-

cution will not stop until the specified iteration. The “Simulation expressed in real-time” uses the real time specified on the specification file, as specified events are triggered at a specific time. For example:  $occurs(ingr(LightsOn), 16 : 00)$  means that the light should be triggered “on” at 4pm.

The specification file format shown in Fig. 4.3 is the format our implemented system recognizes, in order to conduct the executions. However, the theoretical argumentation language is strictly more expressive than “ $M$ ” and will need to be modeled into the specification file format (system language). This we illustrated in detail in the next section of this paper.

## 4.2 Argumentation Language Translation

The previous chapter (Chapter 3) explored the potential of argumentation to handle conflicting user preferences [78]. The study also explores a generalized framework which can be applied to handle user preferences in AAL and further provided an overall preference architecture (Fig. 3.7) which can be used to extend the current argumentation systems. A proposed system (Hybrid System) was illustrated theoretically to indicate that it can handle different users with the introduction of a personalised preference function. The illustration showed how user preferences can be handled in a realistic way in an intelligent environment.

In addition to the theoretical illustration of how the system should work, we introduced the notion of  $\mathcal{P}ref$ , ([78]) used to represent “User Preferences” in our system, allowing users to specify what part of their preferences is more important to them. This was implemented in the form of an interface, which has been discussed and illustrated in Section 4.4.1, as the interface allows users to select and rank/modify their preference(s) to effect output in a smart home.

The proposed implemented system, which we refer to as “Hybrid System”, has been discussed in Chapter 4.1 and illustrated in Section 4.5, showing its ability to handle conflicting situations in a smart home. However, executing the Hybrid System in a smart house requires a specification file containing arguments which are made of rules, that the smart home will use to act accordingly. These arguments which consist of rules, are required to make decisions, as conclusions are justified through arguments to support their consideration [13]. The argument notations in argumentation will need to be translated to the language (rules) our system (Hybrid) understands for execution. This translation will further be complemented with a simple light study of keeping the lights “off” after the user leaves home, for better explanation. However, we will illustrate in the next section how we modeled



the argumentation theoretical language to the implementation language our system understands.

#### 4.2.1 Modelling Argumentation Language to Implementation Language ( $\mathcal{L}^{\mathbb{T}}$ to $M$ )

This section illustrates how we modeled some of the notations from the argumentation theoretical language ( $\mathcal{L}^{\mathbb{T}}$ ) into the implementation language ( $M$ ). The translation of the  $\mathcal{L}^{\mathbb{T}}$  to  $M$  is not an automated process yet, it has been manually modeled by the developers of the implemented system. There are guidelines listed below, which has been followed throughout the modelling process. Some explanations have also been included for further clarifications.

The first step of the modeling process is the time frame of action(s) or/and event(s) occurred, which can be at an instant or over a period of time (interval).

- “ $I_0$ ” or “ $I_1$ ” or “ $I_2$ ” refer to “interval 0” or “interval 1” or “interval 2” in  $\mathcal{L}^{\mathbb{T}}$ . Modeling this to the  $M$  using the time interval relation “MEETS” will become “[begin( $I_0$ ), end( $I_0$ )]” or “[begin( $I_1$ ), end( $I_1$ )]” or “[begin( $I_2$ ), end( $I_2$ )]”, with an instant representing 1 unit or 1 second of time.
- [-]2 represents 2 units (2 seconds) of time (Interval).
- [-][120s.] represents 120 seconds or 2 minutes of time (Interval).
- Constraints, such as: Length ( $I_1$ ) > 15(mins), will be represented as [-][900s.] or [-]15. This indicates an action or occurrence of event taking place over the previous 15 minutes.

Events occurrence are triggered by sensors or actuators, actions are usually triggered by humans, they were modeled as follows:

- Actions triggered by humans in an interval (e.g. movement detected for 20 seconds) is represented as  $Do_{on}$  in  $\mathcal{L}^{\mathbb{T}}$ , and modeled into  $Do\_$  in the implemented system  $M$ .
- Actions triggered by human in an instant (e.g. the light gets turned “on” at 7:00PM) is represented as  $Do_{at}$ , and modeled as  $Do\_$ .
- Occurrence of event in an interval, triggered by sensor(s) or/and actuator(s) in  $\mathcal{L}^{\mathbb{T}}$  and represented as  $Occurs\_on$ ; has been modeled to  $Occ\_on$  in  $M$ .
- Occurrence of event in an instant, triggered by sensor(s) or/and actuator(s) in  $\mathcal{L}^{\mathbb{T}}$ , and represented as  $Occurs\_at$ ; has been modeled to  $Occ\_at$  in  $M$ .

Other additional notations of  $\mathcal{L}^{\mathbb{T}}$ , which were modeled to  $M$ , which is a superset of atemporal “C” language, are as follows, and we also include some explanations of  $\mathcal{L}^{\mathbb{T}}$  notations that were not modeled but used as they were in the implement system ( $M$ ).

- Negation in  $\mathcal{L}^{\mathbb{T}}$ , is represented as “ $\neg$ ”, and we modeled this to “ $\#$ ”. An example of how we applied the negation is:  $\#LivingroomLight$ , meaning the living-room light is “off”.
- The holding state of a property at an instant in  $\mathcal{L}^{\mathbb{T}}$  is represented as *holdsAt*. We use the same notation (*holdsAt*) in  $M$ . An example of how this notation can be applied is *holdsAt(#LivingroomLight,0)*, meaning at “instant 0” ( $i_0$ ), which is the starting point of the system, the living room light was “off”.
- For  $\mathcal{L}^{\mathbb{T}}$ , the rules have a name or label ID. For example, L-R1 – L-R6 indicates Light rule 1 to Light rule 6. In  $M$ , each rule is represented as “ssr” and refereed to as “same time rule”.

The sort (*Pref<sub>on</sub>*) introduced in [78] which was also applied, is represented as *pref* in the implementation language, and signifies “Users Preferences”.

We applied a light scenario example in Section 4.2.2 as regards to a user who wants the system to switch “off” the lights when s/he leaves home. This is to provide a better understanding of the guidelines for translation aforementioned.

#### 4.2.2 Modelling Sample (Light Scenario)

The notion of interval can be defined as a sequence of consecutive instant. So to translate temporal argument rules to the reasoning rules our system understands, interval relations needs to be considered. For our case, we have been adopted the interval relations defined by [54] and popularised in [3], as it has been known to be the most widespread way to reason and represent time in computer science, specifically in AI. Interval relations defined by [54], have thirteen possible relationships, one of them is “MEETS”.

MEETS( $I_1, I_2$ ) is defined as: interval  $I_1$  is before interval  $I_2$ , but there is no interval between them, i.e.,  $I_1$  ends where  $I_2$  starts. Other relations can be used, we just use MEETS for simplicity of the explanation.

Table 4.1 shows the development of the light scenario through time. The next set of rules (Sara’s lighting scenario) in the next section are extracted from  $\Delta^{\mathbb{T}\downarrow}$  [78] to model the scenario in the argumentation system.

Table 4.1: Sara Lighting Scenario Dynamics

$MEETS(I_0, I_1) \wedge MEETS(I_1, I_2) \wedge MEETS(I_2, I_3)$				
$ Holds_{on}(Movement, I_0) \wedge \neg Holds_{on}(Sleeping, I_0) \wedge \neg Holds_{on}(OnBed, I_0) \wedge Holds_{on}(Home, I_0) \wedge Holds_{on}(LightsOn, I_0)$				
<b>Lighting Scenario</b>	Movement	$\neg$ Movement	$\neg$ Movement	$\neg$ Movement
	$\neg$ Sleeping	$\neg$ Sleeping	$\neg$ Sleeping	$\neg$ Sleeping
	$\neg$ OnBed	$\neg$ OnBed	$\neg$ OnBed	$\neg$ OnBed
	Home	Home	$\neg$ Home	$\neg$ Home
	LightsOn	LightsOn	LightsOn	$\neg$ LightsOn
<b>Transition Cause</b>	$Do_{on}(LeavingHome, I_0)$	System Inference from: L-R3	$Occurs_{at}(\text{SystemTurnsLightOff}, end(I_2))$	
	$I_0$	$I_1$	$I_2$	$I_3$

### Argumentation Light Scenario for Sara

Using the MEETS interval relationship, we illustrate a lighting scenario of a user (Sara) who want the lights in her home to be switched “off”, after the system detects that she has left home.

$$\begin{aligned}
 & MEETS(I_0, I_1) \wedge MEETS(I_1, I_2) \wedge MEETS(I_2, I_3) \\
 & Holds_{on}(Movement, I_0) \wedge \neg Holds_{on}(Sleeping, I_0) \wedge \neg Holds_{on}(OnBed, I_0) \\
 & \quad \wedge Holds_{on}(LightsOn, I_0)
 \end{aligned}$$

L-R1:  $Do_{on}(LeavingHome, I_0) \succ - Occurs_{at}(LeftHome, begin(I_1))$

L-R2:  $Occurs_{at}(LeftHome, begin(I_1)) \succ - Holds_{on}(Movement, I_1)$

L-R3:  $\neg Holds_{on}(Movement, I_1) \wedge Length(I_1) \geq 15 \wedge \neg Holds_{on}(OnBed, I_1) \succ - Holds_{on}(Home, I_2)$

L-R4:  $\neg Holds_{on}(Home, I_2) \succ Pref_{on}(LightOff, I_2)$

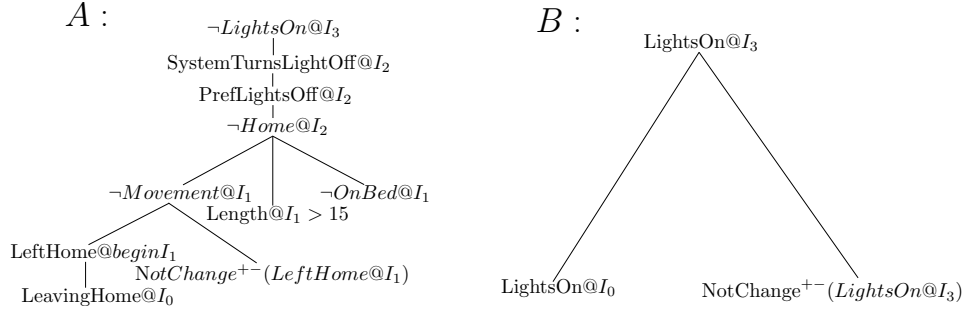
L-R5:  $Pref_{on}(LightOff, I_2) \succ Occurs_{at}(SystemTurnsLightOff, end(I_2))$

L-R6:  $Occurs_{at}(SystemTurnsLightOff, end(I_2)) \succ - Holds_{on}(LightsOff, I_3)$

The above six rules were modeled to the rules in the specification file, which is format the implemented system ( $M$ ) understands. Figure 4.4 further depicts the argumentation trees representation of the above rules, along with explanation of the argument.

*Argument for LightsOn@I<sub>3</sub>*: As seen from the initial facts, the lights are assumed to be “on”, as Sara is in the room. So because of persistency, there is a possibility that the lights will continue to remain “on”.

$$\begin{aligned}
 L.On = \langle \{ & Holds_{on}(LightsOn, I_0) \wedge \\
 & notChange_{in}^{+-}(LightsOn, [end(I_0), end(I_3)]) \\
 & \succ - Holds_{on}(LightsOn, I_3) \}, \\
 & Holds_{on}(LightsOn, I_3) \rangle
 \end{aligned}$$



The argument is reflected in figure 4.4B.

*Argument for  $\neg LightsOn@I_3$ :* Considering an alternative explanation, given that the system has been designed to understand when the lights are not needed. The argument indicates that Sara is leaving home at  $I_0$  and is not home at the beginning of  $I_1$ . As a result of this, no movements were detected from then onwards. If continued for the next 15 minutes and there is no pressure on the bed at the same time, the system has reasons to believe that Sara is not at home at  $I_2$ . When Sara is not at home, she prefers the lights “off”. So at that moment, the system infers that it is reasonable to turn the lights “off”. As a result, the lights are off at  $I_3$ , as illustrated in the argument tree shown in figure 4.4A

$$\begin{aligned}
 L.Off = & \\
 & \{ \text{Do}_{on}(LeavingHome, I_0) \succ - \text{Occurs}_{on}(LeftHome, I_1), \\
 & \text{Occurs}_{on}(LeftHome, I_1) \succ - \text{Holds}_{on}(Movement, I_1), \\
 & - \text{Holds}_{on}(Movement, I_1) \wedge Length(I_1) > 15 \wedge - \text{Holds}_{on}(OnBed, I_1) \succ - \text{Holds}_{on}(Home, I_2), \\
 & - \text{Holds}_{on}(Home, I_2) \succ - \text{Pref}_{on}(LightsOff, I_2), \\
 & \text{Pref}_{on}(LightsOff, I_2) \succ - \text{Occurs}_{on}(SystemTurnsLightOff, I_2), \\
 & \text{Occurs}_{on}(SystemTurnLightOff, I_2) \succ - \text{Holds}_{on}(LightsOn, I_3) \}, \\
 & \neg \text{Holds}_{on}(LightsOn, I_3)
 \end{aligned}$$

Table 4.2 further illustrates the translation of Sara's light scenario, following the dynamics of Table 4.1, applying the interval relationship (“MEETS”) and the modeling guidelines provided in section 4.2.1. The output of the modelling process of the light scenario is in form of the specification file we provide in section 4.2.2.

### Specification File with converted rules

Below depicts the full specification file that was used in translating the light scenario, along with the modeled rules discussed in the section 4.2.2. Other aspects of the specification file are explained in section 4.1.2.

Table 4.2: Converting Argumentation Rules to Implemented System Rules.

	Argumentation Rules ( $\mathcal{L}^T$ )	Specification File Rules ( $\mathcal{M}$ )
<b>L-R1</b>	$Do_{on}(LeavingHome, I_0)$ $\succ Occurs_{at}(LeftHome, begin(I_1))$	$ssr((Do\_LeavingHome)$ $\quad - > Occ\_LeftHome);$
<b>L-R2</b>	$Occurs_{at}(LeftHome, begin(I_1))$ $\succ \neg Holds_{on}(Movement, I_1)$	$ssr((Occ\_LeftHome) - > \#Movement);$
<b>L-R3</b>	$\neg Holds_{on}(Movement, I_1) \wedge Length(I_1) > 15 \wedge$ $\neg Holds_{on}(OnBed, I_1) \succ \neg Holds_{on}(Home, I_2)$	$ssr(([-][900s.]\#Movement \wedge \#OnBed)$ $\quad - > \#Home);$
<b>L-R4</b>	$\neg Holds_{on}(Home, I_2) \succ Pref_{on}(LightOff, I_2)$	$ssr((\#Home) - > \#prefLightOn);$
<b>L-R5</b>	$Pref_{on}(LightOff, I_2)$ $\succ Occurs_{at}(SystemTurnsLightOff, end(I_2))$	$ssr((\#prefLightOn)$ $\quad - > SystemTurnsLightOff);$
<b>L-R6</b>	$Occurs_{at}(SystemTurnsLightOn, end(I_2))$ $\succ \neg Holds_{on}(LightsOff, I_3)$	$ssr((SystemTurnsLightOff)$ $\quad - > \#LightsOn);$

```
states(Movement, OnBed, LightsOn, Home, Do_LeavingHome, Occ_LeftHome,
      SystemTurnsLightOff, prefLightOn);
```

```
is(Do_LeavingHome);
is(#OnBed);
```

```
holdsAt(#Movement, 0);
holdsAt(#OnBed, 0);
holdsAt(LightsOn, 0);
holdsAt(Home, 0);
holdsAt(Do_LeavingHome, 0);
holdsAt(#Occ_LeftHome, 0);
holdsAt(SystemTurnsLightOff, 0);
holdsAt(prefLightOn, 0);
```

```
ssr((Do_LeavingHome) -> Occ_LeftHome);
ssr((Occ_LeftHome) -> #Movement);
ssr(([-][900s.]#Movement ^ #OnBed) -> #Home);
ssr((#Home) -> #prefLightOn);
ssr((#prefLightOn) -> SystemTurnsLightOff);
ssr((SystemTurnsLightOff) -> #LightsOn);
```

Now we will discuss and show some of the infrastructure and equipment required for the demonstrations in a real environment.

### 4.3 Smart-Home Infrastructure

The research utilized a Smart Spaces Lab to conduct practical demonstration of the system. The lab is a fully functional home environment provided to support

research in AAL and specialized spaces to support research in the areas of Virtual/Mixed/Augmented reality and group decision making support. The Lab further consists of other physical equipment that was also needed for the demonstration process, the physical equipment will be explained later. Some images of the Smart Spaces Lab areas and the equipment are found in section 4.3.1 and section 4.3.2 respectively.

### 4.3.1 Smart Spaces Lab

The smart space lab is located within the Middlesex University premises. It is also known as Farm House with necessary housing facilities, giving the lab the feel of a home. Figure 4.5 depicts the layout which consists of a living room (fig. 4.6), a bedroom (fig. 4.7), a kitchen, a bathroom, a shower room and two addition rooms used for conducting meetings and research. As seen on the layout, parts of the house are wired with all types of sensors for research purposes, but we will address a few that are specific to this research.

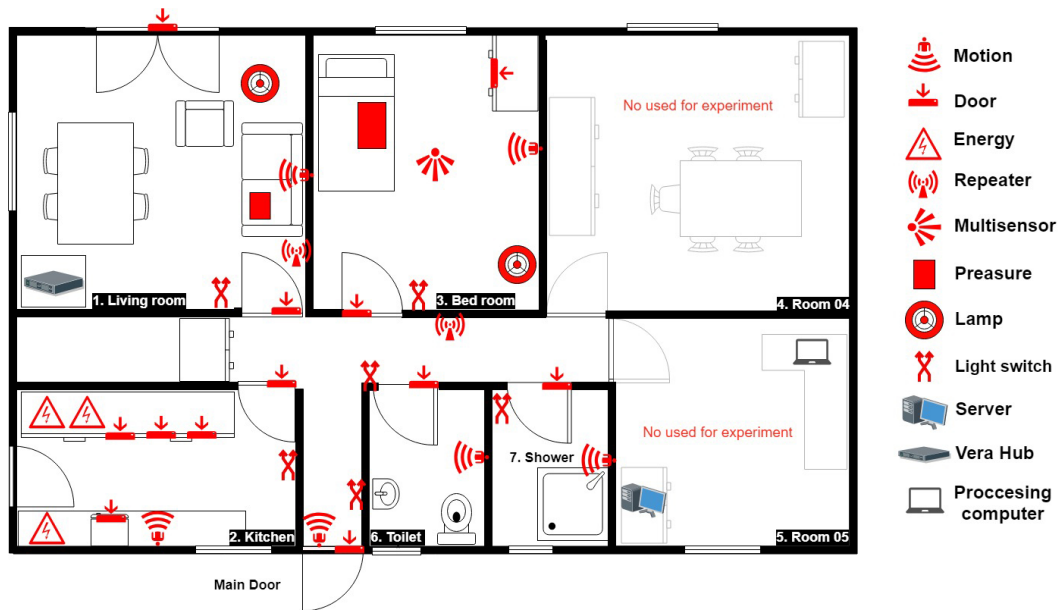


Figure 4.5: Layout of the Smart Spaces Lab.

More images of the smart home can be found here: <http://ie.cs.mdx.ac.uk/smart-spaces-lab/>



Figure 4.6: Living-Room of the Smart Spaces Lab.



Figure 4.7: Bedroom of the Smart Spaces Lab.

### 4.3.2 Equipment

The smart home requires smart devices and equipment (see right side of fig. 4.5) to conduct the experiments. However, for our demo we made use of a few, which are Motion sensor(A), Reed sensor(B), Light Switch(C), Vera Box(D) and Pressure Pad(E), as show in fig. 4.8.



(A). Movement Sensor



(B). Reed Sensor



(C). Light Switch



(D). Vera Box



(E). Bed Pressure Pad

Figure 4.8: Smart devices and equipment for experiments.

The Motion sensor (also known as the PIR) is used to detect movement in the areas placed around the house. The Reed sensor device is mostly attached by doors or windows to detect if they are open or closed. The Reed device was reconfigured along with a dance mat to produce the pressure pad (shown in fig. 4.8E), which we used to detect pressure on the bed. We can either place the pressure pad on the bed or on the sofa to detect if a user is occupying any of these positions.

Fig. 4.8C is a light switch, connected to the smart box which we refer to as

Vera. This can be used to carry out the automation process without using the switch itself. Fig. 4.8D depicts the smart hub (Vera Box) that manages the z-wave sensors and actuators connected to it through its own WiFi network. Vera accepts requests to query or modify the state of the sensors/actuators. We used Vera and the reasoning system to execute the instructions in the specification file, which will trigger the necessary outputs in the smart home.

The next section will illustrate our first demo using the preference mechanism, with our system and an informal scenario to demonstrate how different user's input can immediately impact the system's output. But first, we will introduce our preference management tools.

## 4.4 Preference Management

One key aspect of our system is to provide means which allow users to manage their preferences easily. The system uses the managed preference(s), to reason about the preferences of the user, and provides output that aligns better with the services required by the user. A simple interface has been produced to help users manage their preferences and also help to manage some of the complexities in users' preferences in a smart home. The interface consists of textual menus for simplicity, incorporating the *Pref* notion introduced in [78] to allow users to select and rank their preference(s) at their convenience. Depending on how the user ranked their preference(s), the system output will be affected.

### 4.4.1 Preference Management Tool (Interface)

As mobile devices grow in functionality and popularity, the demand for advanced mobile applications in human daily life increases [39]. Gracanin et al ([53]) emphasises how mobile technologies have the potential of connecting users with their environment, and how smart environments enhanced with technology to support better living, may improve individuals' lives. However, it is still a major issue to design and develop a flexible interface application that matches many users' needs and provide them the usability and quality experience they require [49].

#### **Some challenges on interfaces in AmI**

Preferences handling has been known to be a core issue in designing an automated system that aims to support the decision making of the users [31]. It can be more challenging when users find it difficult to handle the technology(interface) that is



supposed to manage their preferences. This is already an issue for the elderly, as it is difficult for them to be involved in technological activities [59].

One common proposed requirement (which tends to be a challenge) when designing a mobile interfaces, is for the users to be able to interact with the interface with easy, and less buttons/clicks(interactions) [60, 70]. Although, [61] developed an interface known as “Motive”, that rely on just four buttons for user inputs. Also an interesting idea was presented by [98] known as assist-robot interface, that works in Portable-Mode (when the user is not at home) and Robot-Mode (when the user is at home), and so on.

However, these interfaces might not be ideal for all users in AAL, especially for older adults whose technical experience tends to decline and it limits their ability to use and interact with technology user interfaces[86, 62]. In addition, these interface applications cannot handle the management of users’ preferences, especially when the user wants to have control over their own preferences within their environment.

We provide a simplified interface, incorporating the idea of a preference sort, *pref*, introduced by [78]. This allows users to select and rank their preference(s) at a convenient time from the developed interface. Depending on the how the users’ preferences were ranked using the simplified interface, the system output will change. The interface was developed to give the user the ability to prioritize their preferences, and give them the ability to modify it at any time, using their mobile phone and the changes will take effect immediately.

#### 4.4.2 Managing Users’ Preferences

Preferences can be imposed to a certain extent, such as doctor’s recommendation, adjustment in lifestyle, the need to do a certain activity etc.[77]. Preferences have a number of complexities as they may change over time or clash with each other. For example, sometimes there may be reasons to keep the lights ‘on’ and other reasons to keep them ‘off’ at a particular time. Therefore, balancing of these users’ preferences is a crucial factor in AmI [50], so that the system should be effective enough to support users’ needs. We give a brief scenario, followed by a demonstration of the scenario, to illustrate how the developed interface works with a reasoning system to manage the preferences set by a user.

***Scenario:** Bobby, an aged individual who lives alone, prefers the light to be ‘off’ when he is asleep at night to provide more comfort. However, he might sometimes prefer the light to be ‘on’, so it is safer for him to move around when he wakes up in the middle of the night.*

### **Setting up new user's preferences (Bobby)**

The interface has been developed in a simple manner where the existing users can easily retrieve their preference profile, or set up a new profile on the same page. The home page has two options, the drop-down list to display existing users or a text-box to enter the name of new user. Creating a new user will generate the same list of pre-defined preferences currently in the system. The users has to adhere to them, as we are currently working with strict preferences (second part of figure 4.9 depicts the strict preferences we are currently working with). Though the interface has been designed in a way to edit, delete or add more preferences directly from the database.

On the page containing the list of preferences, the user just needs to check the boxes of the preference(s) that applies to them or which preference they want to rank. The user does not need to check all the boxes from the list of the available preferences, as the idea is to give users the ability to choose and prioritize the preference(s) they want. The selected preference(s) will be transfer to the third page where the user can now prioritize and store them in the database, ready to be used immediately. Each preference can be ranked from 1-10 by the user, with 10 being the highest priority. Figure 4.9 illustrates the setting up of a new user (Bobby).

The aim is to provide a simple and easy to use interface, that will not create any form of complexity for the user, and still be effective enough to carry out the complexities in preference management with the reasoning system. Research aimed at a focus group and contextual inquiries of potential smart home inhabitants [60], indicate that, users want a control that is as simple as possible, and the interaction for usage should consist of around 2-3 buttons. The same research also found out that users want the interaction method to be consistent, easy to use and familiar, as they want to feel in control of their home environment [60].

### **Modifying Bobby's preferences**

As initially stated, preferences may change over time and can be modified based on experience or other reasons. The interface also has a simple mechanism to modify the existing users' preference(s) which will change the output/decision the reasoning system will provide for the user. Figure 4.10 illustrates two steps of modifying the ranking of existing preferences (Booby in this case) and saving it.

Despite the user setting and ranking up their preferences (which is a one-time procedure for new user), we needed to make sure the interface is easy to learn and consistent for every user regardless of the individual. The first page of figure 4.10 is



Figure 4.9: Simple setting up of new users' Preference (Bobby).  
face

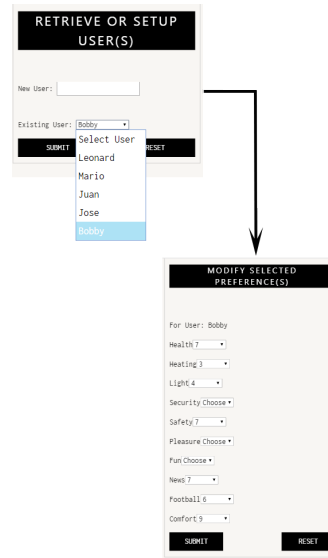


Figure 4.10: Retrieving and modifying existing preferences (Bobby).

where the user selects their name, and then it loads their profile on the next page, for modification and updating. Also, if the user has not specified any ranking for a preference at the initial stage of creating their profile, they can also do so when modifying their existing preference(s), if they choose.

#### 4.4.3 Using Preference Interface To Affect System Behaviour.

To illustrate the functionality and effectiveness of the developed interface within a smart home environment so as to manage users' preferences, other systems are required. A reasoning system (MReasoner, [57]) to run the specification file, a router, known as Vera, which provides the framework to control sensing devices (light sensor, movement sensor, pressure pad etc.) working with Z-wave sensors and the interface to manage the preferences. The reasoning system, will run the specification file (partially shown in figure 4.11), and with the occurrence of some event (e.g. *BedRoomMovement*) the required action(s) will be triggered in the smart house. Basically, the information that is entered from the interface is stored in the preference database(DB), and updated immediately anytime the user modifies their preference(s). MReasoner on the other hand, when running, continue to read the preference DB and when there is/are any update(s) in the DB (made by the user), MReasoner immediately use the current update(s) and apply the necessary changes to the system.

Using the aforementioned guiding scenario (in section 4.4.2), let us consider the user, Bobby, who expects the lighting scenario to adapt to varying circumstances.

The first sentence of the scenario indicates that Bobby wants the system to turn the bedroom light “off” when he is asleep as he prefers the comfort over keeping the light “on”. In this case Bobby has decided to rank his “Comfort” higher (probably 6) than “Light” (probably 4). When the system executes the rules (found below), which states that if Bobby is on bed for 30 seconds (*[30s.]BedPadPressure*) and there is no movement in the bedroom (*#BedRoomMotion*), the system will switch “off” the light (*#BedRoomLight*). As seen the first line of the below rules.

```
ssr((-)[30s.]BedPadPressure ^ #BedRoomMotion ^ prefComfort)-> #BedRoomLight);
ssr((-)[30s.]BedPadPressure ^ #BedRoomMotion ^ prefLight)-> BedRoomLight);
```

The second sentence in the description explains that Bobby might sometimes prefer the light “on” for safety reasons when he wakes up during the night. Let’s assume Bobby decides to change his preference and ranks “Light” higher (6) than “Comfort” (5). When he goes back to bed, after 30 seconds or more of being on the bed, the system will still continue to keep the lights “on”, as he has now indicated that he prefers “Light” over “Comfort” (shown ). The next section emphasizes on the complete specification file, along with the rules.

#### 4.4.4 System Specification for MReasoner

Below is the complete system specification that will be fed to the reasoning system, which the smart home will react to. The specification refers to the scenario in section 4.4.2.

```
states(CorridorMovement, CorridorLight, ToiletLight, ToiletMovement,
BedRoomLight, BedRoomMovement, BigPadIdle, prefLight, prefComfort, getup,
siesta, nightsleep);
```

```
is(CorridorMovement);
is(#CorridorMovement);
is(ToiletMovement);
is(#ToiletMovement);
is(BedRoomMovement);
is(#BedRoomMovement);
is(BigPadIdle);
is(#BigPadIdle);
is(prefLight);
is(prefComfort);
```

```

holdsAt(#CorridorMovement, 0);
holdsAt(#CorridorLight, 0);
holdsAt(#ToiletLight, 0);
holdsAt(#ToiletMovement, 0);
holdsAt(#BedRoomLight, 0);
holdsAt(#BedRoomMovement, 0);
holdsAt(#BigPadIdle, 0);
holdsAt(prefLight, 0);
holdsAt(prefComfort, 0);
holdsAt(#siesta, 0);
holdsAt(#nightsleep, 0);
holdsAt(#getup, 0);

ssr((<->[13:00:00-16:00:00]#BedRoomMovement ^ #BigPadIdle)-> siesta);
ssr((<->[23:00:00-06:00:00]#BedRoomMovement ^ #BigPadIdle) -> nightsleep);
ssr((siesta ^ BedRoomMovement ^ BigPadIdle) -> getup);
ssr((nightsleep ^ BedRoomMovement ^ BigPadIdle) -> getup);
ssr((getup) -> BedRoomLight);
ssr((CorridorMovement) -> CorridorLight);
ssr((#CorridorMovement)-> #CorridorLight);
ssr((ToiletMovement) -> ToiletLight);
ssr((#ToiletMovement) -> #ToiletLight);
ssr(([-][30s.]#BigPadIdle ^ #BedRoomMovement ^ prefLight)-> BedRoomLight);
ssr(([-][30s.]#BigPadIdle ^ #BedRoomMovement ^ prefComfort)-> #BedRoomLight);

```

The first part of the system specification refers to all the states in the house that are needed for the scenario mentioned above in this section. The second part of the specification (e.g. *is(BedRoomMovement)*); refers to Independent States, which do not casually depend on other states and can be either true or false. The third part are the Initial Status values for each of the states. For instance, *holdsAt(#CorridorLight, 0)*;; means the corridor light should be ‘off’ at the start of the scenario. The fourth part of the specification are the rules that triggers the actions. The selected section of the rule in figure 4.11 where we have *prefLight* and *preComfort* respectively, means the bedroom light should be ‘on’ if the user prefers *Light* or the bedroom light should be ‘off’ if the user prefers *Comfort*. Figure 4.12 depicts and overall preference management architecture of how information coming from the external world (e.g. sensors, internet) and/or from a user (through preference interface), can change the conclusion using preferences (including the ability to cope with competing and conflicting preferences. The next section of this paper provides a video link, illustrating the aforementioned scenario and how the interface interact differently with there is a change in user’s Preferences.

```

M Specification File Editor Database Results LFPUBS Rule Translations
holdsAt(#BigPadIdle, 0);
holdsAt(prefLight, 0);
holdsAt(prefComfort, 0);
holdsAt(#siesta, 0);
holdsAt(#nightsleep, 0);
holdsAt(#getup, 0);

ssr(<->[13:00:00-20:00:00]#BedRoomMovement ^ #BigPadIdle) -> siesta);
ssr(<->[23:00:00-06:00:00]#BedRoomMovement ^ #BigPadIdle) -> nightsleep);
ssr((siesta ^ BedRoomMovement ^ BigPadIdle) -> getup);
ssr((nightsleep ^ BedRoomMovement ^ BigPadIdle) -> getup);
ssr((getup) -> BedRoomLight);
ssr((CorridorMovement) -> CorridorLight);
ssr((#CorridorMovement) -> #CorridorLight);
ssr((ToiletMovement) -> ToiletLight);
ssr((#ToiletMovement) -> #ToiletLight);
ssr([[30s.]#BigPadIdle ^ #BedRoomMovement ^ prefLight) -> BedRoomLight);
ssr([[30s.]#BigPadIdle ^ #BedRoomMovement ^ prefComfort) -> #BedRoomLight);
<
Line: 36 Column: 0

```

Figure 4.11: Reasoning system (MReasoner) screen-shot with system specification details, to illustrate change in response based on preference ranking.

### Video demonstration

This research provides videos demo, illustrating how the reasoning system and interface works, and also depicting how the smart home reacts differently when there is a change in preference ranking. The link ([74]) contains two videos indicating how the house react when there is a change in preferences, such as user prioritizing Light over Comfort or vice versa. When Bobby prioritizes light over comfort, it means he wants to keep the lights ‘on’ when he is asleep and when he prioritizes comfort over light, he wants the light ‘off’ when he is asleep as its more comfortable.

Figure 4.13 indicates how we modelled part of Bobby’s scenario (from fig. 4.12), when he decided to keep the light “on” while he asleep, so it is safer for him to move around when he wakes up during the night. As seen from fig 4.13, Bobby modifies his preferences ranking to prefer “Light” over “Comfort”. The ranking order is saved in the database, and shown on the bottom right of the figure. Since Bobby ranked the “Light” higher than “Comfort”, the system provides Bobby the service of keeping the light “on”.

Further demonstration will be provided in the next section of the thesis, applying preference representation of Sara shown in Fig. 3.8. The illustration consist of how our Hybrid System works when conflicts arises.

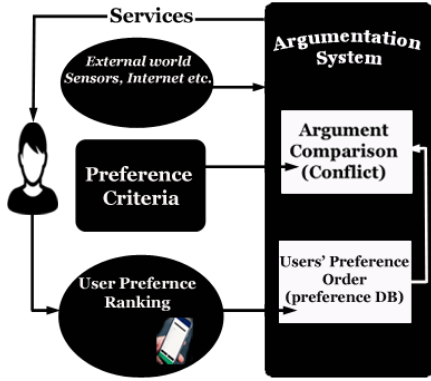


Figure 4.12: Overall architecture of preference management system

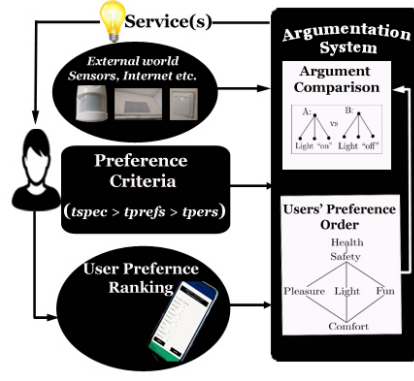


Figure 4.13: Modeled Bobby's situation of keeping the light "on"

## 4.5 System Illustrations (Demos)

The implemented system (Hybrid Main) which comprises of a reasoning system and the argumentation resolver, is used to demonstrate how the system works, applying the scenarios mentioned in Section 1.4. The demo is in three categories. The first demonstration was on the Hybrid System, which shows the overall working of the application. This includes selecting a specification file that contains rule(s), which the system compiles and check for potential conflict(s). Secondly, a light scenario to illustrate the working of the argumentation system, using preference criteria initially discussed and applied in this precedence order:  $\mathfrak{R} = \{\succ_{\text{tspec}}, \succ_{\text{Upref}(a)}, \succ_{\text{tpers}}\}$  with  $\succ_{\text{tspec}} > \succ_{\text{Upref}(a)} > \succ_{\text{tpers}}$

Lastly, the integration of a large chain store's API, as we use the system to access their data and search for a specific type of product ("Cake" in this case). This illustrates the flexibility of our research in terms of the sources of data and the type of contexts being considered.

### 4.5.1 Hybrid System Illustration

Figure 4.1 depicted the interface of the Hybrid System, which is used to load a specification file. The specification file contains a set of rules, which should be selected using the "Select Specification File" button. Depending on whether the rules in the Specification file contains potential conflict(s) or not, the system will activate/enable either the "MReasoner" button or the "Argumentation Solver" button.

Launching the Hybrid System will disable both the "MReasoner" and "Argumentation Resolver" buttons, as shown in Fig. 4.1. The specification file will need to be selected (which can be selected from any location on the computer) as shown

in Fig. 4.14. When the specification file is selected, the compiler (referred to as conflict analyser) compiles the file for potential conflicts. If no conflict is detected, the “MReasoner” button is enabled (allowing the system to run the specification file without the involvement of the conflict analyser) and the “Argumentation Solver” button stays disabled as shown in Fig 4.15. If potential conflict(s) is/are detected, the “Argumentation Solver” button is enabled and the potential conflict(s) is displayed in the text area as shown in Fig. 4.16. The system can now run the file and solve any actual conflict from the potential ones.

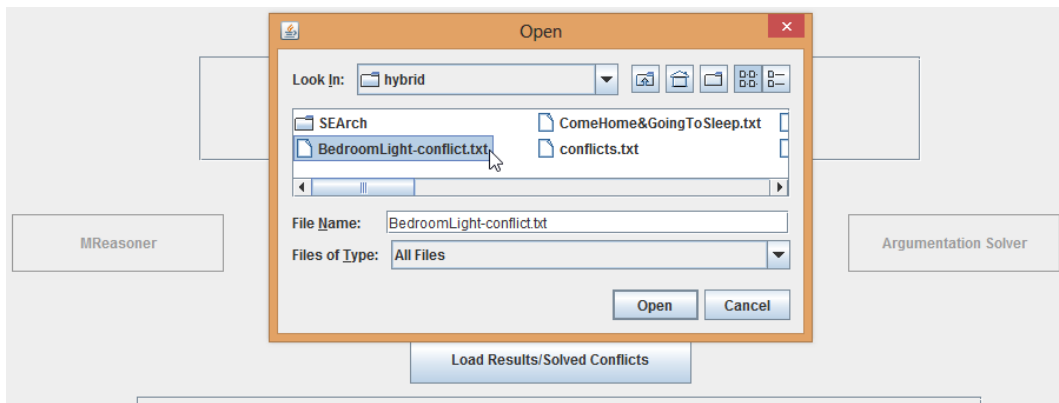


Figure 4.14: Browsing to select Specification File

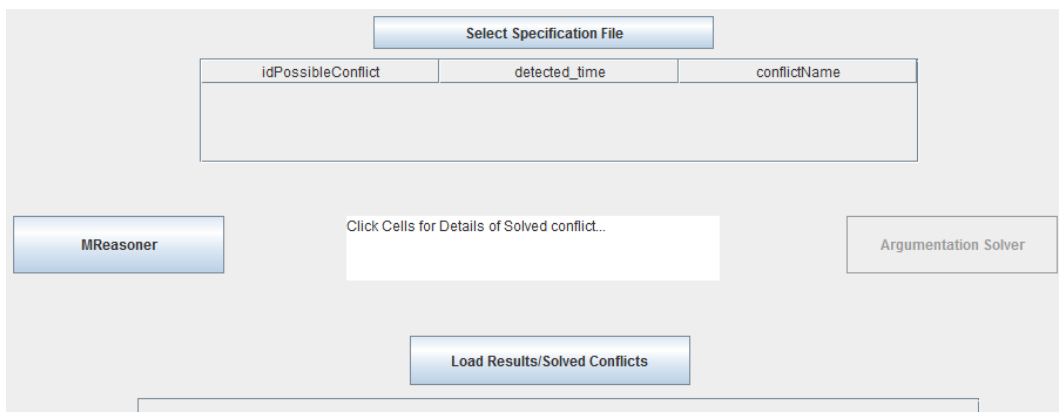


Figure 4.15: The MReasoner button is enabled as 'NO' potential conflict is detected

Figure 4.16 shows three potential conflicts, that were detected after the specification file (“BedroomLight-conflict.txt”) was selected, but only the detected conflict(s) among them was solved during execution. Meanwhile, to check for conflict(s), the compiler only compiles the last part of the specification file that consists of the rules. Below are the rules that were compiled in this case:

```
ssr((<->[12:00:00-18:00:00]BedRoomMovement ^ BigPadIdle) -> BedRoomLight);
```



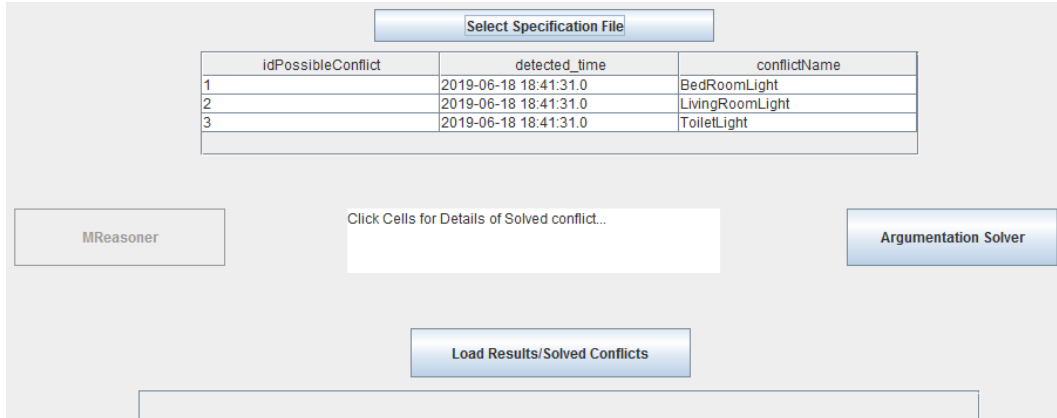


Figure 4.16: The Argumentation Solver button is enabled as potential conflict is detected

```

ssr((LivingRoomMovement) -> LivingRoomLight);
ssr((#LivingRoomMovement) -> #LivingRoomLight);
ssr((ToiletMovement) -> ToiletLight);
ssr((#ToiletMovement) -> #ToiletLight);
ssr((-)[30s.]#BigPadIdle ^ #BedRoomMovement ^ prefLight) -> BedRoomLight);
ssr((-)[30s.]#BigPadIdle ^ #BedRoomMovement ^ prefComfort) -> #BedRoomLight);

```

The three potential conflicts from the above rules are related to conclusions involving: *BedRoomLight*, *LivingRoomLight* and *ToiletLight*. However, *LivingRoomLight* and *ToiletLight* are only potential conflicting, as the consequence opposes each other. Here, the property *BedRoomLight*, has been detected as a conflict, as both rules states that if the pressure pad is “not” idle for 30 seconds ( $(-)[30s.]#BigPadIdle$ ) and no movement detected in the bedroom ( $\#BedRoomMotion$ ), then the bedroom light being either “on” or “off”, will be decided based on the preference ranking of the user. If the user ranks “Comfort” ( $prefComfort$ ) higher than “Light” ( $prefLight$ ), then the bed room light goes “off” ( $\#BedRoomLight$ ) else, the bed room light stays “on” (*BedRoomLight*).

The scenario was executed in the real environment, as the events are triggered with either sensors and/or actuators. Figure 4.18 indicates that the user has set his/her priority to prefer “Comfort” over “Light”, in this case. This means that when the conflict is detected, the system first tries to resolve the argument with “Specificity”, which will not be possible, as both rules are equally specific. The system will then try to resolve the argument using “Preference”, and from Fig 4.18, “Comfort” has higher priority over “Light”. So  $\#BedRoomLight$  wins the argument, and the system switches “off” the bedroom light when the user is on the bed for more than 30 seconds and no movement is detected in the bedroom.

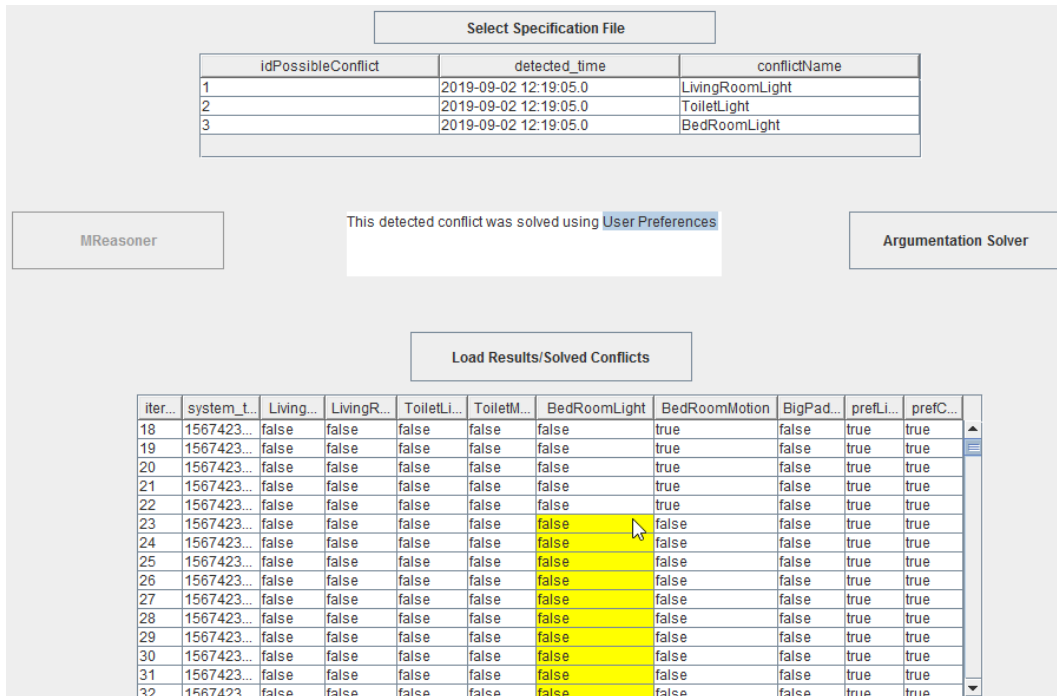


Figure 4.17: Hybrid System highlighting the columns where conflict was detected and solved

The Hybrid interface also has the ability to populate the results of the properties value, and also pinpoint the exact instant or interval the conflict(s) were identified and solved. The Hybrid System also provides the details of how the conflict(s) was/were solved. During or after execution the results are display using the“Load Results/Solved Conflicts" button. Figure 4.17 shows results which are loaded on the below text area of the Hybrid interface, and the highlighting identifies the areas where conflicts were detected and solved immediately.

Furthermore, clicking on any of the highlighted cells, additional information on how the conflict was solved at that instant will be provided. Since *BedRoomLight* was the conflicting state/property, and the argument was solved using “user preference”, the system highlights the conflicting cells within the *BedRoomLight* column. When any of the cell is clicked, the reason how the argument (*BedRoomLight*) was solved, gets displayed in the middle text-area of the Hybrid interface, as shown in Fig 4.17. If any other area (with no highlighting) is clicked, the text area will display “No conflict detected”.

We further applied this argument (*BedRoomLight*) to our preference architecture (shown in Fig. 3.7) which we introduced in our previous work [78], to illustrate how our produced system functions internally. Figure 4.19 shows how the argument was fully applied to the preference architecture, and how the bedroom light conflict



Figure 4.18: Interface showing that the user prioritised **Comfort** over **Light**.

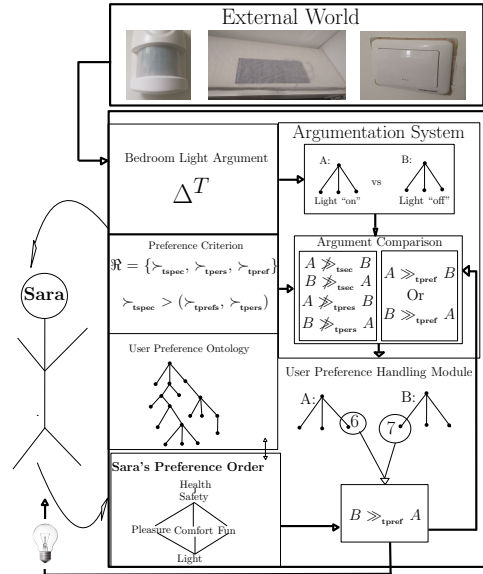


Figure 4.19: Using Sara's preference ranking to solve bedroom light conflict

was solved using the preference ranking (in Fig. 4.18) of the user (Sara).

As shown in Fig. 4.19, compared to the overall preference architecture in Fig. 3.7, the “*External world*” where information comes into the system from the outside, was replaced by the equipment in the bedroom. The equipment consists of the movement sensor which detects if the user is present in the bedroom or not. The pressure pad (known as *BigPadIdle*), placed underneath the mattress, detects that the user has been on the bed continuously for the past 30seconds ( $[-][30s.]\#BigPadIdle$ ), and the light switch automatically goes “on” or “off” depending on Sara’s preference ranking.

Since the system could not decide whether to keep the bed room light “on” or “off”, a conflict resolution process had to take place. From Fig. 4.19, the system considers the arguments as in “A” (Bedroom Light “on”) or in “B” (Bedroom Light “off”) as shown in the top right side of the architecture. The system then runs the check using Specificity (as shown in “Argument Comparison”) and from the rules “A” is not more specific than “B” ( $A \not\geq_{tsec} B$ ) and vice versa ( $B \not\geq_{tsec} A$ ). The system then moves to the next preference criterion, which according to the order of precedence, is “user preferences”. The system then runs another check, in the “User Preference Handling Module”, where the system checks the database to access the user (Sara) preference ranking order, for “Comfort” and “Light”. From the bottom left side of the figure (Fig. 4.19), it shows that Sara ranked “Comfort” higher than

“Light”, also shown on her preference profile in Fig. 4.18. The profile indicates that Sara ranked “Comfort - 7” (argument “*B*”) and “Light - 6” (argument “*A*”), which will allow the system to turn the bedroom light “off”, thereby solving the conflict with “*B*” winning the argument using user’s preferences ( $B \gg_{\text{tpref}} A$ ).

Figure 4.20 further depicts the database records, and the last column indicating the reason (“User Preferences”) the system applied in solving the bedroom light conflict.

idResolvedConflict	iteration	resolved_time	LivingRoomMovement	LivingRoomLight	ToiletLight	ToiletMovement	BedroomLight	BedroomMovement	BigPadIdle	prefLight	prefComfort	solved_reason
1	0	2019-07-11 15:21:40	NULL	NULL	NULL	0	NULL	NULL	NULL	NULL	NULL	User Preferences
2	1	2019-07-11 15:21:41	NULL	NULL	NULL	0	NULL	NULL	NULL	NULL	NULL	User Preferences
3	2	2019-07-11 15:21:43	NULL	NULL	NULL	0	NULL	NULL	NULL	NULL	NULL	User Preferences
4	3	2019-07-11 15:21:44	NULL	NULL	NULL	0	NULL	NULL	NULL	NULL	NULL	User Preferences
5	4	2019-07-11 15:21:45	NULL	NULL	NULL	0	NULL	NULL	NULL	NULL	NULL	User Preferences
6	5	2019-07-11 15:21:47	NULL	NULL	NULL	0	NULL	NULL	NULL	NULL	NULL	User Preferences
7	6	2019-07-11 15:21:48	NULL	NULL	NULL	0	NULL	NULL	NULL	NULL	NULL	User Preferences
8	7	2019-07-11 15:21:49	NULL	NULL	NULL	0	NULL	NULL	NULL	NULL	NULL	User Preferences
9	8	2019-07-11 15:21:50	NULL	NULL	NULL	0	NULL	NULL	NULL	NULL	NULL	User Preferences
10	9	2019-07-11 15:21:51	NULL	NULL	NULL	0	NULL	NULL	NULL	NULL	NULL	User Preferences
11	10	2019-07-11 15:21:52	NULL	NULL	NULL	0	NULL	NULL	NULL	NULL	NULL	User Preferences
12	11	2019-07-11 15:21:53	NULL	NULL	NULL	0	NULL	NULL	NULL	NULL	NULL	User Preferences
13	12	2019-07-11 15:21:56	NULL	NULL	NULL	0	NULL	NULL	NULL	NULL	NULL	User Preferences
14	13	2019-07-11 15:21:57	NULL	NULL	NULL	0	NULL	NULL	NULL	NULL	NULL	User Preferences
15	14	2019-07-11 15:21:58	NULL	NULL	NULL	0	NULL	NULL	NULL	NULL	NULL	User Preferences

Figure 4.20: Database showing how the conflict was solved using the preference criterion, “User Preferences”

The following link ([73]) contains a video demonstration of the Bedroom conflict scenario, as a supporting evidence of the explanation and illustration made in this section. We have also provide the data set result of the experiment, to indicate details of the full output of the validation process.

This research further conducted a supplementary demonstration in Section 4.5.2, to show that our system is able to detect and solve all three preference criteria earlier discussed.

#### 4.5.2 Solving conflicts using three preference criteria (Specificity, User Preferences and Persistency).

A specification file was written to trigger potential conflicts in all three areas of the preferences criteria, as the intention was to illustrate that our system is capable enough to detect conflicts at any time, even at the same interval and solve them using any of the preference criteria. Below is the specification file with the rules, which consist of three potential conflicts in relation to the preference criteria:

```
states(BedroomMotion, BedroomLight, ShowerMotion, ShowerRoomLight,
ToiletMotion, ToiletLight, CorridorMotion, CorridorLight, BigPadIdle,
prefComfort, prefLight);
```

```
is(CorridorMotion);
```

```

is(ShowerMotion);
is(BigPadIdle);
is(ToiletMotion);
is(BedroomMotion);
is(prefComfort);
is(prefLight);

holdsAt(#CorridorMotion, 0);
holdsAt(#CorridorLight, 0);
holdsAt(#BedRoomLight, 0);
holdsAt(#BedroomMotion, 0);
holdsAt(BigPadIdle, 0);
holdsAt(#ToiletLight, 0);
holdsAt(#ToiletMotion, 0);
holdsAt(#ShowerRoomLight, 0);
holdsAt(#ShowerMotion, 0);
holdsAt(prefComfort, 0);
holdsAt(prefLight, 0);

ssr((CorridorMotion ^ prefLight) -> CorridorLight);
ssr((CorridorMotion ^ prefComfort) -> #CorridorLight);
ssr((BedroomMotion ^ BigPadIdle) -> BedRoomLight);
ssr((BedroomMotion) -> #BedRoomLight);
ssr((ToiletMotion) -> ToiletLight);
ssr((ShowerMotion) -> ShowerRoomLight);
ssr((CorridorMotion) -> #ShowerRoomLight);

```

The rules were created to check for the “*User Preference*” aspect of conflict, as the term *prefLight* and *prefComfort* indicate the preference aspect which triggers either the corridor light “on” (*CorridorLight*) or the corridor light “off” (*#CorridorLight*). The rules also checked for the “*Specificity*” aspect of the conflict, as the potential conflict of bedroom light is determined based on which of the arguments (“*BedRoomLight*” or “*#BedRoomLight*”) is more specific or informed. The system also checked for “*Persistency*” notion, to know if the property (state) keeps the true value over time when there is no reason for the property to change its value, unless there is/are reason(s) to believe otherwise.

We applied Sara’s preference ranking order (discussed in Section 3.3.1, Fig. 3.8) for the demonstration of this scenario. Meanwhile, Figure [4.21] [4.22] [4.23] illus-

trates the argument of all three potential conflicts in argumentation tree form, using the “*MEETS*” interval relation initially discussed.

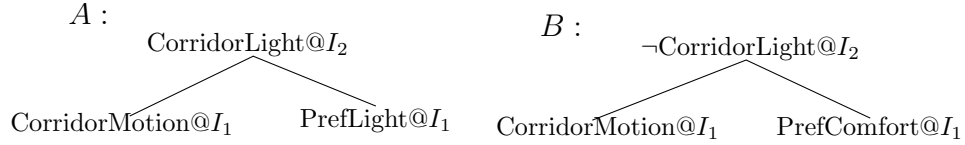


Figure 4.21: Argument tree for Corridor-Light “on” or “off”.

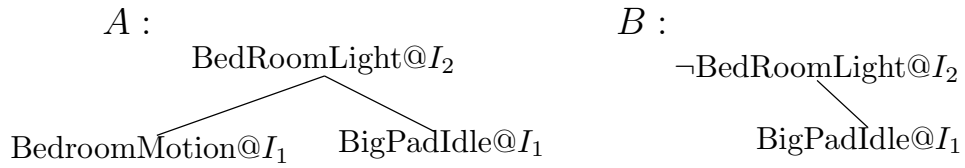


Figure 4.22: Argument for BedRoom-Light.

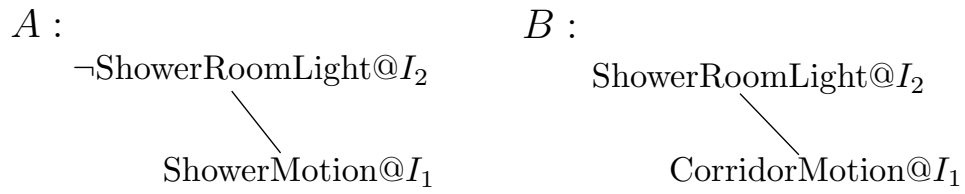


Figure 4.23: Argument for ShowerRoom-Light.

When the system is in execution, it processes the specification file and checks for potential conflict(s). From the rules, the potential conflicts are *CorridorLight*, *BedRoomLight* and *ShowerRoomLight* and are stored in the potential conflict table, and display on the Hybrid Interface (shown in Fig. 4.24). Each time a new specification file is processed, it erases the previous record(s) in the potential conflict table and saves the current potential conflict(s) identified (if any, otherwise the potential conflict table will remain empty).

The first potential conflict (*CorridorLight*) is an actual conflict, as the system does not know whether to turn “on” or turn “off” the corridor light. However, this depends on which of the preferences (“*Light*” or “*Comfort*”) has higher priority. For this scenario, Sara’s preference ranking order shown Fig. 3.8 was adopted, *asprefLight* was assigned the value 6 and *preComfort* was assigned the value 4. The corridor light was turned “on” as *CorridorLight* won the argument based on *prefLight* having higher priority over *preComfort*.

The second potential conflict (*bedroom light*) was decided based on “Specificity”, so *BedRoomLight* won the argument over *¬BedRoomLight*. This is because the argument (*BedRoomLight*) had additional information (“*BedroomMotion*”) that should

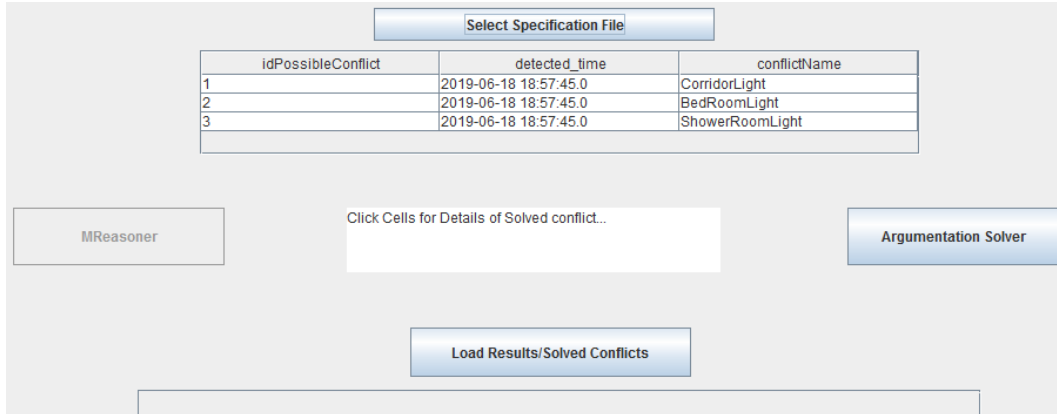


Figure 4.24: Identified Potential Conflicts for Sara

supports the argument of turning the light “on”.

The argument representation tree of the “*Bedroom Light*” (as shown in Fig. 4.22A and 4.22B), further explains why argument “A” wins the argument based on specificity, with tree “A” having additional information than tree “B”. “*BedroomMotion*” is a motion sensor (Fig. 4.8A) which is used to detect movement around the bedroom, along with the pressure pad being idle (*BigPadIdle*, shown in Fig. 4.8E), will keep the light “on”.

The current value for the shower room light persists which is *#ShowerRoomLight*, meaning that the shower room light remains “off”. Since both “Specificity” and “User Preferences” cannot solve the conflict, the property (“Shower-room Light”) retains the previous value of keeping the light “off”, unless there is an inference of new information into the system. The previous value in this case is “off” (*holdsAt(#ShowerRoom – Light,0)*;) as shown in the specification file. This signifies that the value of the “Shower-Room Light” property at the starting point or initial state, was “off”.

Note, all rules follows the order of precedence ( $\succ_{\text{tspec}} > \succ_{\text{Upref}(a)} > \succ_{\text{tpers}}$ ) in trying to solve any conflict, regardless of how the specification file is written. This means any detected conflict first tries to be solved using “Specificity” and if it cannot be solved, the system then tries to use “User Preferences”. If the conflict cannot be solved using “User Preferences” (maybe because the Preference properties are equally ranked), it then continues to keep the property’s true(initial) value (“Persistency”).

Figure 4.25 shows the intervals (highlighted) where conflicts were detected and solved for this scenario. As seen from the screen-shot, selecting a column from the bedroom light row, the reason (“Specificity”) used in solving the conflict is displayed in the middle text area. The figure also illustrates that conflicts were solved on other

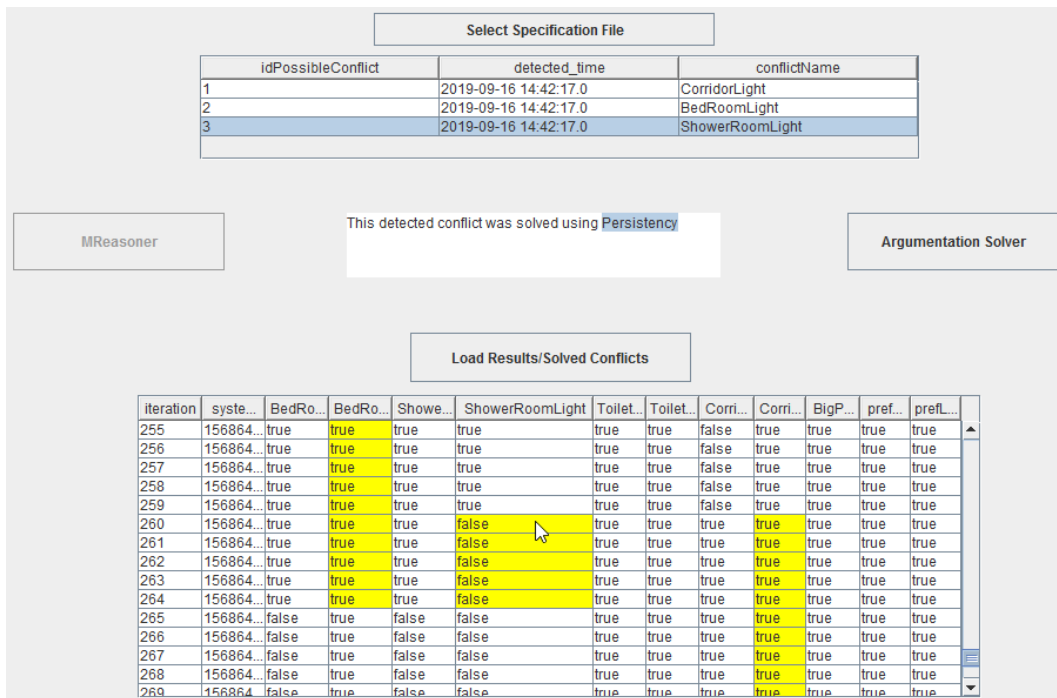


Figure 4.25: Hybrid System showing all three detected and solved of conflicts; “Specificity”, “Preference” and “Persistency”

properties (“Shower-room Light and Corridor Light”) as well, which were solved with persistency and user preferences, respectively.

Figure 4.26 illustrate some of the the database log of the solved conflict. The “iteration” column states the exact iterations where the conflicts were detected and solved, the properties columns (Bedroom Light, Shower room Light and Corridor Light) display either a new conclusion or retain the previous value. The values in the database indicating 1 or 0, which represent true or false displayed on the Hybrid interface. The last column (“resolve\_reason”), depicts the reason the Hybrid System was used to resolve the conflict. In addition, the system is able to solve multiple conflicting preferences at the same time, using any or all of the preference criteria in the iteration.

The demonstration link ([75]) indicates the illustration discussed above, using the aforementioned specification file in this section. Attached in the same link is the complete data set, showing more logs of the detected conflicts and how they were solved, applying the preference criteria where necessary. The validation was conducted for 2 hours.



idResolve	iteration	resolved_time	BedRoomMotion	BedRoomLight	ShowerMotion	ShowerRoomLight	ToiletMotion	ToiletLight	CorridorMotion	CorridorLight	BigPadIdle	prefComfort	prefLight	solved_reason
85	255	2019-09-16 14:47:18	NULL	1	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	Specificity
86	256	2019-09-16 14:47:19	NULL	1	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	Specificity
87	257	2019-09-16 14:47:20	NULL	1	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	Specificity
88	258	2019-09-16 14:47:21	NULL	1	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	Specificity
89	259	2019-09-16 14:47:23	NULL	1	NULL	NULL	NULL	NULL	1	NULL	NULL	NULL	NULL	User Preferences
90	259	2019-09-16 14:47:23	NULL	1	NULL	NULL	NULL	NULL	1	NULL	NULL	NULL	NULL	Specificity
91	259	2019-09-16 14:47:23	NULL	NULL	0	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	Persistency
92	260	2019-09-16 14:47:24	NULL	NULL	NULL	NULL	NULL	NULL	1	NULL	NULL	NULL	NULL	User Preferences
93	260	2019-09-16 14:47:24	NULL	1	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	Specificity
94	260	2019-09-16 14:47:24	NULL	NULL	0	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	Persistency
95	261	2019-09-16 14:47:25	NULL	1	NULL	NULL	NULL	NULL	1	NULL	NULL	NULL	NULL	User Preferences
96	261	2019-09-16 14:47:26	NULL	1	NULL	NULL	NULL	NULL	1	NULL	NULL	NULL	NULL	User Preferences
97	261	2019-09-16 14:47:26	NULL	NULL	0	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	Persistency
98	262	2019-09-16 14:47:28	NULL	NULL	NULL	NULL	NULL	NULL	1	NULL	NULL	NULL	NULL	User Preferences
99	262	2019-09-16 14:47:28	NULL	1	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	Specificity
100	262	2019-09-16 14:47:28	NULL	NULL	0	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	Persistency
101	263	2019-09-16 14:47:29	NULL	NULL	NULL	NULL	NULL	NULL	1	NULL	NULL	NULL	NULL	User Preferences
102	263	2019-09-16 14:47:29	NULL	1	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	Specificity
103	263	2019-09-16 14:47:29	NULL	NULL	0	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	Persistency
104	264	2019-09-16 14:47:30	NULL	NULL	NULL	NULL	NULL	NULL	1	NULL	NULL	NULL	NULL	User Preferences
105	265	2019-09-16 14:47:31	NULL	NULL	NULL	NULL	NULL	NULL	1	NULL	NULL	NULL	NULL	User Preferences
106	266	2019-09-16 14:47:32	NULL	NULL	NULL	NULL	NULL	NULL	1	NULL	NULL	NULL	NULL	User Preferences

Figure 4.26: Database records of the three types of conflicts; “Specificity”, “User Preferences” and “Persistency”

### 4.5.3 Supermarket Chain Store (Tesco) API

The research took another step to validate the effectiveness of the Hybrid System using live data. The live data was from one of the top supermarkets in the United Kingdom, known as Tesco. We requested for the API on their grocery products, which was used to filter “Cake” product, and check if the product description contains sugar. The aim was to warn the user, Sara, who is known to be diabetic, about the content of the Cake product, but it is Sara’s decision to buy the Cake or not. The system also identifies the Cake products that do not contain sugar, which gives Sara more options of deciding to buy them or not.

So based on the users’ ranking preference (Sara in this case), since she prefers health (*prefHealth*) over pleasure (*prefPleasure*) as seen in Fig. 3.8, the system should advise her “not” to buy the cake (*#Occ\_SystemAdvicesBuyCake*). If for some reason Sara changes her preference ranking of preferring “Pleasure” over “Health”, the system will then advise the user to buy the cake (*Occ\_SystemAdvicesBuyCake*). In addition, if it happens that all the filtered cake product do not contain sugar, Sara can equally choose to (or not to) order from any of the available cake products that do not contain sugar and vice versa.

Considering the healthy eating case study in Section 1.4.2, the below specification file with rules was developed to check for the availability of a particular product, “Cake”. If found, the rules check the product description (details of the cake) for sugar, and then advises the user depending on her preference ranking.

```
states (BuyCake, Diabetic, CakeOnSales, Sugar, Occ_CakeAvaliable,
Occ_SugarDetected, Occ_SystemAdvicesBuyCake, prefPleasure, prefHealth);

is(Occ_CakeAvaliable);
```

```

is(Occ_SugarDetected);
is(Diabetic);
is(prefPleasure);
is(prefHealth);

holdsAt(#BuyCake, 0);
holdsAt(Diabetic, 0);
holdsAt(#CakeOnSales, 0);
holdsAt(#Sugar, 0);
holdsAt(Occ_CakeAvaliable, 0);
holdsAt(#Occ_SugarDetected, 0);
holdsAt(#Occ_SystemAdvicesBuyCake, 0);
holdsAt(prefHealth, 0);
holdsAt(prefPleasure, 0);

ssr((Occ_CakeAvaliable) -> CakeOnSales);
ssr((CakeOnSales ^ prefPleasure) -> Occ_SystemAdvicesBuyCake);
ssr((Occ_SugarDetected) -> Sugar);
ssr((Diabetic^Sugar^CakeOnSales^prefHealth) -> #Occ_SystemAdvicesBuyCake);

```

Figure 4.27 depicts how the link to the data is generated from Tesco Labs. According to the search parameter, the product to be queried needs to be entered (Cake in this case), the “offset” indicates where the search should commence from. If the “offset” is 10, the search result is produced from the 11th product, and the “limit” is how many products you want to limit the search to. This can be any number, 12, 50, 67 or 500 (which is the maximum at a time). When the these parameters have been set, it will generate a url which will be used (along with a private subscription key) to access the filtered product.

Figure 4.28 illustrates the Hybrid interface after system’s execution. The specification file is first compiled to check for potential conflict(s) (*Occ\_SystemAdvicesBuyCake* in this case) as shown in the Fig. 4.28, but, the system is yet to advise Sara to buy the Cake or “not”. During execution, the Hybrid System accesses the URL online to check for the availability of the product, Cake. If Cake is available, it means the Cake is up for sale at that moment. The system then checks from the list of Cakes available, to know if the product description contains sugar. If sugar is found in the description, the system advises Sara “not” to buy the product (as seen from the scenario) due to her health condition, in addition to her preference priority of Health(*prefHealth*) over Pleasure (*prefPleasure*), as seen in Fig. 3.8.

Considering the rules on the specification file for the Tesco API, one might ask

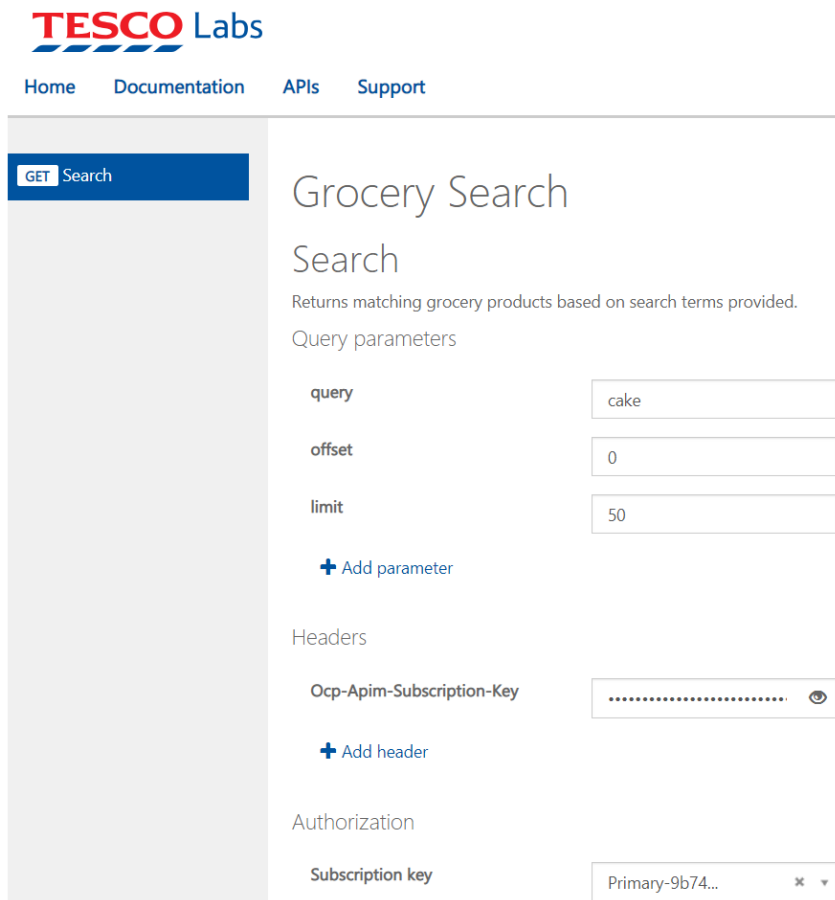


Figure 4.27: Requesting for Tesco URL to search for Cake Product

why “User Preference” criteria was used to solve the conflict instead of “Specificity”. “Specificity” as we know (when comparing arguments), is a way of preferring the best-informed argument. Specificity is also based on the structure of the arguments, and when the argument is incomparable or equi-specific [13], the system will then apply the next preference criterion (user preference in this case). The argument *Occ\_SystemAdvicesBuyCake*, is incomparable because one of the arguments has a unique property the other argument does not have. So this cannot be used to decide which one is preferable. However, if both arguments were as follows:

```
ssr((CakeOnSales ^ prefHealth) -> #Occ_SystemAdvicesBuyCake);
ssr((Diabetic^Sugar^CakeOnSales^prefHealth) -> Occ_SystemAdvicesBuyCake);
```

The notion “Specificity” will be applied in this case, as *Occ\_SystemAdvicesBuyCake* will win, as the argument is more informed than the other argument. Since the argument for *#Occ\_SystemAdvicesBuyCake* does not

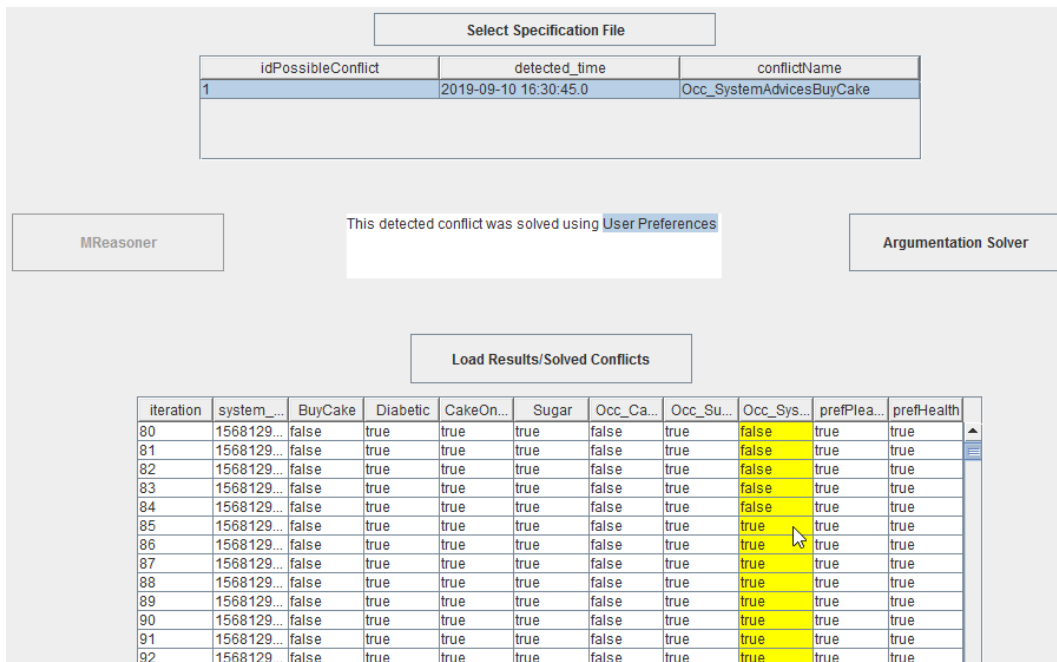


Figure 4.28: System Advice Sara Not to buy cake since her “Health” has higher priority over “Pleasure”

contain any supporting property the argument for *Occ\_SystemAdvicesBuyCake* does not have, “Specificity” criterion can be used in solving this conflict. Figure 4.29 consists of all Cake products that were extracted from the filter, with an additional column to inform the user (Sara) which of the products contains “Sugar” or “not”.

A video demonstration of the Tesco API illustration is found here: [76], along with data sets extracted from the validation process.

S/N	product_id	product_name	super_department	department	description	content_qua	unit_price	price	retrieved_date	type
55	283390081	Cadburv Chocolate Mini Roll 5 Pack	Bakerv	Cakes, Cake Bars...	/Chocolate flavoured sponoe with a vanilla flavour creme. cove...	5	0.16	0.8	2019-09-10 16:31:42	No suoar
56	264369712	Tesco Almond Fingers 5 Pack	Bakerv	Cakes, Cake Bars...	/5 Almond flavoured sponoe cakes topped with half an almond....	5	0.2	1	2019-09-10 16:31:42	No suoar
57	268322208	Tesco Mini Chocolate Cornflake Bit...	Bakerv	Cakes, Cake Bars...	/15 Cornflake clusters covered in milk chocolate. /Made with Milk...	15	0.12	1.8	2019-09-10 16:31:42	No suoar
58	262780947	Almondv Daim Chocolate Cake 400G	Frozen Food	Frozen Desserts, I...	/Chocolate cake with Daim /milk chocolate coated almond caram...	0	0.75	3	2019-09-10 16:31:42	No suoar
59	296852241	Mcvitie's Dioestive Caramel Slice 5 ...	Bakerv	Cakes, Cake Bars...	/Dioestives Slices Topped with Caramel & Milk Chocolate /www. 1...	124	0.579	0.72	2019-09-10 16:31:42	No suoar
60	295848818	Cadburv Rasoberrv Mini Roll 10 Pack	Bakerv	Cakes, Cake Bars...	/Golden sponoe with plum and rasoberrv iam and a vanilla flavo...	10	0.25	2.5	2019-09-10 16:31:42	No suoar
61	287504132	Cadburv Rasoberrv Mini Roll 5 Pack	Bakerv	Cakes, Cake Bars...	/Golden sponoe with plum and rasoberrv iam and a vanilla flavo...	5	0.16	0.8	2019-09-10 16:31:42	No suoar
62	268770593	Tesco Crispv Caramel Bites 20 Pack	Bakerv	Cakes, Cake Bars...	/Caramel and crisped rice bites part coated in milk chocolate. /Mi...	20	0.09	1.8	2019-09-10 16:31:42	No suoar
63	297571030	Tesco Cookies & Cream Cake	Bakerv	Cakes, Cake Bars...	/Chocolate and madeira cake filled and covered with cookie cru...	1	13.0	13	2019-09-10 16:31:42	No suoar
64	297544495	Mcvitie's Jaffa Cakes 10 Pack	Food Cupboard	Biscuits & Cereal Bars	/10 Light Sponoe Cakes with Dark Cracklv Chocolate and a Sma...	10	0.05	0.5	2019-09-10 16:31:42	No suoar
65	251816699	Mr Kiolino Mini Battenbero Cakes 5...	Bakerv	Cakes, Cake Bars...	/Chequered Sponoe Sandwiched Together with an Apricot Fillin...	5	0.164	0.82	2019-09-10 16:31:42	No suoar
66	285313228	Tesco Rose Bouquet Cake	Bakerv	Cakes, Cake Bars...	/Sponoe cake filled with rasoberrv iam and frosting, covered an...	1	11.0	11	2019-09-10 16:31:42	No suoar
67	282051974	Hoopers Chocolat Mini Rolls 10 Pack	Bakerv	Cakes, Cake Bars...	/Chocolate flavour sponoe rolls with vanilla flavour fillno and co...	10	0.1	1	2019-09-10 16:31:43	No suoar
68	300120975	Ms Mollv's 12 Iced Fairv Cakes	Bakerv	Cakes, Cake Bars...	/12 Sponoe cakes with plain, lemon or strawberrv flavour icino...	12	0.063	0.75	2019-09-10 16:31:43	No suoar
69	300903245	M&M's Chocolate B...ool Celebrati...	Bakerv	Cakes, Cake Bars...	/Chocolate sponoe filled and covered with a chocolate flavour f...	1	11.0	11	2019-09-10 16:31:43	Suoar
70	301516206	Mcvitie's Jaffa Cakes Strawberrv 1...	Food Cupboard	Biscuits & Cereal Bars	/10 Light Sponoe Cakes with Dark Cracklv Chocolate and a Stra...	10	0.05	0.5	2019-09-10 16:31:43	No suoar
71	284508478	Mcvitie's Jaffa Cake Bars 10 Pack	Bakerv	Cakes, Cake Bars...	/A scrumbtious blend of luscious dark cracklv chocolate, light sp...	10	0.25	2.5	2019-09-10 16:31:43	No suoar
72	300120998	Ms Mollv's 12 Fairv Cakes	Bakerv	Cakes, Cake Bars...	/12 Sponoe cakes. /Lip smackindlv lovely sponoe cakes perfect ...	12	0.063	0.75	2019-09-10 16:31:43	No suoar
73	299847773	Ms Mollv's Chocolate Fairv Cakes 1...	Bakerv	Cakes, Cake Bars...	/12 Chocolate sponoe cakes /Lip smackindlv lovely sponoe cakes...	12	0.063	0.75	2019-09-10 16:31:43	No suoar
74	251558848	Galaxv Cake Bars 5 Pack	Bakerv	Cakes, Cake Bars...	/Sponoe cake bars with a chocolate cream centre covered in mil...	5	0.3	1.5	2019-09-10 16:31:43	No suoar
75	252479773	Real Lancashire Eccles Cakes	Bakerv	Cakes, Cake Bars...	/Eccles cakes /For further details please oo to our web site www...	4	0.4	1.6	2019-09-10 16:31:43	No suoar
76	257617038	Tesco Small Chocolate Celebration...	Bakerv	Cakes, Cake Bars...	/Chocolate sponoe cake filled and covered with chocolate oana...	1	6.0	6	2019-09-10 16:31:43	No suoar
77	300795325	Tesco Jaffa Cakes Twin Pack 282G	Food Cupboard	Biscuits & Cereal Bars	/24 Soft baked cakes and an orange centre, coated in dark cho...	282	0.337	0.95	2019-09-10 16:31:43	No suoar
78	291396990	Tesco Jaffa Cake 430G	Frozen Food	Frozen Desserts, I...	/Baked orange fillno on a sponoe base, topped with orange fla...	430	0.465	2	2019-09-10 16:31:43	No suoar
79	257373377	Tesco Haovv Birthdav Cake	Bakerv	Cakes, Cake Bars...	/Madeira sponoe cake filled with rasoberrv iam and buttercrea...	1	8.5	8.5	2019-09-10 16:31:43	No suoar
80	252657128	Mr Kiolino Battenbero Cake	Bakerv	Cakes, Cake Bars...	/Chequered Sponoe Sandwiched Together with an Apricot Fillin...	1	1.0	1	2019-09-10 16:31:43	No suoar
81	259404984	Tesco Stars Partv Cake	Bakerv	Cakes, Cake Bars...	/Madeira sponoe cake filled with buttercream and rasoberrv ia...	1	6.7	6.7	2019-09-10 16:31:44	Suoar
82	261722967	Tesco Triole Laver Chocolate Cake	Bakerv	Cakes, Cake Bars...	/Three lavers of chocolate sponoe cake, filled and covered with...	1	12.0	12	2019-09-10 16:31:44	No suoar
83	250315487	Tesco Football Cake	Bakerv	Cakes, Cake Bars...	/Madeira sponoe cake filled with rasoberrv iam and buttercrea...	1	8.0	8	2019-09-10 16:31:44	Suoar
84	267207059	Tesco Partv Cake Selection 12 Pack	Bakerv	Cakes, Cake Bars...	/4 Vanilla flavoured sponoe cakes topped with icino and decorat...	12	0.15	1.8	2019-09-10 16:31:44	Suoar
85	250221275	Galaxv Caramel Cake Bars 5 Pack	Bakerv	Cakes, Cake Bars...	/Sponoe cake bars with a caramel centre covered in milk chocol...	5	0.3	1.5	2019-09-10 16:31:44	No suoar
86	299370938	Mcvitie's Jaffa Cakes Twin Pack	Food Cupboard	Biscuits & Cereal Bars	/20 Light Sponoe Cakes with Dark Cracklv Chocolate and a Sma...	244	0.656	1.6	2019-09-10 16:31:44	No suoar
87	263501840	Norfolk Cake Co. Mixed Fruit Loaf ...	Bakerv	Cakes, Cake Bars...	/Made with apple juice (from concentrate).	1	2.25	2.25	2019-09-10 16:31:44	No suoar
88	300795204	Tesco Jaffa Cakes 141G	Food Cupboard	Biscuits & Cereal Bars	/Soft baked cake and a zinov orange centre, coated in rich dark...	141	0.426	0.6	2019-09-10 16:31:44	No suoar
89	258871840	Mcvitie's Jaffa Cake Bars 5 Pack	Bakerv	Cakes, Cake Bars...	/A blend of dark cracklv chocolate, light sponoe and smashing o...	5	0.29	1.45	2019-09-10 16:31:44	No suoar
90	291548490	Tesco Free From Carl The Caterpill...	Bakerv	Cakes, Cake Bars...	/Free from chocolate cake filled with chocolate flavoured frostin...	1	6.0	6	2019-09-10 16:31:44	Suoar
91	265217995	Tesco Madeira Partv Cake	Bakerv	Cakes, Cake Bars...	/Madeira sponoe cake with rasoberrv iam, filled and topped wit...	1	6.0	6	2019-09-10 16:31:45	No suoar
92	271188921	Dr. Oetker Partv Candles 18	Food Cupboard	Home Bakino	/18 Partv Candles /Join our Webake Community to showcase vo...	18	0.056	1	2019-09-10 16:31:45	No suoar
93	297570993	Tesco Rainbow Cake	Bakerv	Cakes, Cake Bars...	/Madeira sponoe cake filled and coated with multi coloured frost...	1	12.0	12	2019-09-10 16:31:45	Suoar
94	302088595	Maltesers Treat Cake	Bakerv	Cakes, Cake Bars...	/Chocolate sponoe cake covered with a chocolate frosting and ...	1	8.0	8	2019-09-10 16:31:45	No suoar
95	284515383	Tesco Birthdav Cake Cubes 15 Pack	Bakerv	Cakes, Cake Bars...	/15 Cubes of chocolate flavoured sponoe cake filled with chocol...	15	0.667	10	2019-09-10 16:31:45	No suoar
96	300016978	Tesco Pink Flaminoo Cake	Bakerv	Cakes, Cake Bars...	/Pink sponoe cake laved with strawberrv iam, covered with cr...	1	12.0	12	2019-09-10 16:31:45	Suoar
97	264983216	Tesco Celebration Cake	Bakerv	Cakes, Cake Bars...	/Madeira sponoe cake filled with buttercream and rasoberrv ia...	1	8.5	8.5	2019-09-10 16:31:45	Suoar
98	272080844	Thorntons Celebration Cake	Bakerv	Cakes, Cake Bars...	/Chocolate Sponoe Filled and Covered with Chocolate Buttercre...	1	12.0	12	2019-09-10 16:31:45	No suoar
99	293943455	Emotv Celebration Cake	Bakerv	Cakes, Cake Bars...	/Celebration Cake - Sponoe with a laver of rasoberrv iam and s...	1	9.0	9	2019-09-10 16:31:45	No suoar
100	265391781	Cadburv Flake Cake	Bakerv	Cakes, Cake Bars...	/Chocolate sponoe laved with chocolate flavour creme and a ...	1	11.0	11	2019-09-10 16:31:45	No suoar

Figure 4.29: Database showing some of the 50 filtered “Cake” products, with last column indicating the product with “Sugar” or “No Sugar”

## Chapter 5

# Discussion

Ambient assisted living (AAL) research aims to enhance quality of life, especially for older adults who might face some level of challenges at home, either due to age or any other life circumstance. The aim is to offer appropriate technological solutions to mitigate or overcome some of the constraints felt by people, who often live alone, with some kind of physical limitation [38].

Various investigations have been conducted in the early stages of this research, as the aim was to carefully explore the state of the art, so the study can properly understand what it ought to produce, and present the best solution possible (see published survey journal of the investigation i). The research survey reflects on existing studies focused on theoretical and practical solutions for preference management systems, and investigates studies that emphasise the importance of preferences in ambient intelligence. This investigation, along with the research specific aim and questions, gives good insight into the kind of practical solution provided by this research.

It was challenging addressing most of the research objectives highlighted in section 1.2. Here, the thesis discuss some of the research questions that were addressed and how, and those that could not be completely addressed due either time, resources or complexity.

The research addressed the first research question (objective 1.), of how to effectively represent one or more partial order of preference. The research provided a number scale method which allows users rank their preferences on a scale of 1-10. This method was adopted as it was deemed an effective way to rank and represent user's preferences. Other methods were considered (for example, "Low", "Medium" and "High"), but the number scale method provides more flexibility to compare preference ranking that has been prioritized by the user. Also, when multiple conflicting preferences occur, the ranking scale enables the conflict analyser algorithm in the Hybrid System to conduct the necessary computation with less complexity.

The Hybrid System has the intelligence to handle conflicting preferences as highlighted in next research question (objective 2.), and solve them. The system first conduct checks of all possible conflicts and then analyses them to further detect any actual conflict, before solving them. The system does not have the capability to address both user preferences and needs, as the research focused more on addressing conflicts in preferences. In addition, time factor was taken into consideration, as the research provided a system with the ability to use time in identifying clashes in preferences, in an instance or within an interval, but the system does not have the ability to assume changes over time. So the current system acts automatically, according to changes made by the user, which still gives the user control. This is linked to the next research question listed (objective 4.), as the Hybrid System does have the ability to change its behaviour according to changes made by the user. Users can manage their preferences by using an interface, and when there is a change in their preference ranking it affects the system's behaviour.

The interface (objective 3.) provided does not have the most appealing look, as the aim was to keep the interface as simple as possible. The interface was mainly to effect changes on the Hybrid System. This part of the research focused on simplicity and ease of use, as the aim was for users (including older adults) to access, modify and update their preferences with fewer clicks. More details of the interface can be found in our 2019 intelligent environment conference paper (see. iii, in section 6).

Investigations were conducted on existing multi-agent tools and how researchers have adapted them to tackle problems related to the handling of user preferences. However, most of the findings have limitations in managing user preferences. They are either hard-coded/hard-wired, do not have the flexibility to allow users to manage their preferences, or do not have the sort of reasoning capability to deal with user preferences over time. Supplementary research was undertaken into the various preference handling mechanisms in the state of the art, as a technique capable of handling inconsistency and knowledge in relation to time was needed. Apart from trying to produce the practical solution for AAL in smart environments, the research explored modern sensor systems in smart spaces such as offices etc. This enabled the research to identify some additional problems of smart spaces and pinpoint areas that needs improvements.

Furthermore,  $M$  language in the MReasoner only implements a subset of  $\mathcal{L}^{\mathbb{T}}$  in the Temporal Argumentation System.  $M$  is only a fragment of the past and present time fragments of  $\mathcal{L}^{\mathbb{T}}$ . The current expressiveness has been achieved in a bottom up approach as a compromise between expressiveness and efficiency, capable to process in real-time input from 30 or so sensors and checking with this input data whether

any of the dozens of rules can be fired, and if that is the case whether there are conflicts, and if that is the case which conclusion the system should rely most on. All of this in a matter of very few seconds to make practical sense in the real home. In theory both the  $M$  and the  $\mathcal{L}^{\mathbb{T}}$  languages expressiveness can be extended to also consider more reasoning capabilities about the past events stored in the system, and perhaps add something of the future fragment (however this is not so useful for the subset of problems we investigated). Although any of these additions need to be carefully engineered and tested to see whether the impact in real-time efficiency justifies the extra expressiveness.

The research conclude that temporal ( $\mathcal{L}^{\mathbb{T}}$ ) argumentation is the best formalism to achieve the practical aim of our research. Exploration was further conducted with the argumentation techniques theoretically using complex scenarios, so as to validate its effectiveness. However, before adopting the argumentation technique, the research also investigated various classical preference handling techniques in AI. The exploration and investigation on the exiting preference handling mechanisms enable the research realise the feasibility of argumentation for the practical system. Other related systems were discussed and reviewed briefly to find out what systems have been produced, what methods are used, and how these systems relates to our work (section 2.2.2).

Despite considering argumentation a feasible solution, and theoretically exploring the technique in a published journal, the research made some comparison analysis between classical preference techniques in AI and the argumentation technique [77]. The journal emphasise argumentation in AI, establishing the reputation of argumentation and the definition of the previous preference criteria (“specificity” and “persistency”) used to solve possible conflicts. Next, there were discussion on preferences in AI and how preferences are crucial to the decision making process, both in AI and other disciplines. The research produces an overall preference architecture, depicting and explaining how the produced practical system handles user preferences. The architecture shows how the system compares two conflicting arguments, and how the dilemma can be solved using the order of preference criteria. One of the order of preference criteria is the new sort ( $\mathcal{P}ref$ ) introduced in our second publication (see ii, in section 6), which is an essential contribution to the research as it involves the user in the process, enabling them manage their preferences via a developed interface.

The interface developed is an important element of this research, which is crucial for managing user preferences. Its role in the whole practical solution, is not as major as the Hybrid reasoning system itself, but vital in connecting the users with



the environment. An essential concept of AmI is promoting automation, though user interaction is still needed [38], as user interaction is required to provide unobtrusively for the needs of the users. The interface is not only vital due to the user's involvement, but also in the manner which any changes in user's preference ranking effects the system's output immediately. This validates the effectiveness and quality of the whole system (preference interface and the Hybrid System), as Hybrid System uses user's preference ranking records to provide services that better align with the user's behaviour.

The system's effectiveness is validated in other ways. Demonstration of the Hybrid System was produce, showing the functionality and illustrating how the system is able to handle a conflicting situation (of either turning the bedroom light 'on' or 'off' within the smart home), depending on the preferences set by the user. The system is validated against the three preference criteria discussed in the study, and the aim is to show that the system is capable and flexible enough to detect any conflict which might arise while the Hybrid System is running. The system does not only detect the conflict, it also solves the conflict without any disruption, as a smart home which consist of a network of sensors and actuators should not be disruptive regardless of the conflict [79]. Further validation also shows that the produce system is able to handle information from the outside world, as the system was used to access live data from a chain store, which the system filters and informs the user accordingly.

These validations demonstrate the usefulness of the argumentation tool for reasoning about inconsistent information and time, and illustrate its potential to handle conflicting preferences, as the desire for a product might conflict with health considerations. As mentioned, preferences have many complexities, and preferences change over time, so having a system like the Hybrid System, to handle such complexity and help users make more intelligent decisions through effective management of their preferences is crucial in AAL.

Furthermore, the research adopted a scenario-based evaluation as a means to validate the Hybrid System, and the scenarios were developed to fit into an intelligent environment context as well as to test the ability of the system in resolving conflicting preferences. The scenario-based evaluation process follows the general strategies for testing and validation in improving the reliability of intelligent environment systems published in [10, 11]. The validation process for this research would have been much better should we have been allowed to have users stay over at the lab, which was not feasible due to the University's regulations. However, the audience of the Real-AI competition in August 2019, where the Hybrid System was presented and won

the competition, representing Middlesex University, showed notable appreciation on how the system can sensibly adapt decision-making based on real-time contextual changes.

Having the system validated with end users with certain conditions (people with dementia, for example) would have added more quality to the research. However, managers from the “dementia health department” in Finchley and Croydon councils (specialised on elder population) visited the lab, and were impressed with what they saw and how the system can handle scenarios of poor eating, poor sleeping, wandering, and eloping, all typical in citizens with early signs of dementia. In addition, it is typical to use contexts as the main guide for development of systems in this area and these contexts are usually a finite tractable number identified with the scenarios tested and validated. Even the use of exhaustive verification techniques will have to partition the domain in discrete number of tractable cases (which we identify here with contexts and scenarios).

The last chapter of this thesis concludes the research, and discuss additional ways the research can be further improved.

## Chapter 6

# Conclusion and Future Work

The world population is aging rapidly [85] and a rapid surge in ambient assisted living (AAL) technology is being seen due to the increasingly aging society, hence the need to develop innovative assisted living technology for safer and more independent aging individuals.

As mentioned earlier, AmI is the environments which are responsive and sensitive to human presence, allowing people to interact with the physical world in an intelligence and unobtrusive way. While AAL is a sub area that consist of products, concepts and services, and combines the social environment and new technologies in order to improve quality of life.

AAL systems are considered one of the most active research lines within the ambient intelligence community, as their services are becoming more essential and expected to improve the satisfaction of users. To develop an AAL system for a smart home that increases user satisfaction, it needs to understand and respond to the preferences of the user.

Although significant ambient intelligence research has been conducted, and despite being an area which is in essence user-centred, it has not been enough to facilitate a fluent inter-relation between AmI systems and user preferences.

The research presented in this thesis investigates ways to improve the understanding and management of preferences, analysing existing work in preference handling and looking at various strategies to represent and reason with partial orders of various types in order to explain how humans choose among alternatives. The research looks at several well-known alternatives such as CP-nets and UCP-nets, which are seen as promising in other applications.

My experience, based on the development of real AmI systems, highlights the importance of some aspects which are not well supported in AI formalisms for preference handling. One feature which is naturally expected in dealing with human

preferences is the tension of wrestling with “we would like but we can’t have”. A preference linked to tasty food may also be associated with a health preference advising against its consumption. Another feature of preferences is that they are dynamic, and change with time. It could be that we internally change our preferences based on repeated experience, or that a change of preference is imposed externally upon us, for example by health professionals or the weather.

This leads to consideration of argumentation as a possible formalism with the ability to handle inconsistent information and knowledge in relation to time. Chapter 3 emphasise on how the research theoretically explore the argumentation technique, emphasising how it can be applied to managing users’ preferences. The exploration enables us to validate the usefulness of argumentation, as illustrated by applying it to several scenarios.

Section 4.5.1 of this thesis illustrates and validates the Hybrid System, produced using the argumentation technique. The validation depicts how the system functions in executing selected specification files containing conflicting rules. Part of the validation involved using the scenario in Section 3.3.1 (lighting aspect of the scenario), for the user, Sara. The illustration and video demo provided, indicates that the system has the ability to exhibit the necessary behaviour, while solving conflicting preferences.

This thesis applied the scenario to our overall preference architecture (Fig. 3.7), Fig. 4.19 signifies how it successfully applied in a theoretically approach to solve a conflicting problem. If the CP-Net approach was to be applied to the same complex scenario, applying the logic of, for example, “*Sara prefers  $X = x1$  to  $X = x2$* ”, which means that she prefers the light ( $X$ ), to be ‘off’ ( $x1$ ) than ‘on’ ( $x2$ ) when she is not home, the CP-Nets system will not be able to have the information of when particularly Sara was not home. Since CP-nets technique is restricted to known and complete information, it does not have the information of when Sara is not at home. A system developed using the CP-Net logic can be able to express that Sara prefers the bedroom light ‘on’ or ‘off’ when she is not a home, but the question remains, when will she not be at home. We also believe a formal translation is beyond the scope of topic and time available for this project and that the analysis provided in [78] is compelling enough highlighting CP-Nets absence of time and inconsistency handling capabilities.

Furthermore, this research complement previous argumentation frameworks with a user preference architecture (Figure 3.7), showing how the produced system handles the reference to users’ preferences within arguments. This architecture consists of various modules. Part of the architecture detects preferences, another compares

preferences, and another links user specific preferences with more general ontologies.

Given that argumentation is a powerful tool for reasoning with inconsistent knowledge [27] and time [13], and our initial investigations [78] are positive, argumentation considered a useful tool for studying the computational management of preferences. As a subsequent step, work is carried out exploring ways to generalise these findings as well as to create suitable bridges between users and systems, which leads to the interface developed that facilitates the flow of preferences from user to system and vice versa. Ambient assisted living (AAL) is a crucial research and development field, where usability, learning and accessibility play important roles, and interfaces are important for applied engineering [59]. Therefore, delivering a smart system for AmI is not just about providing an effective and efficient system. Simplicity and ease of use have to be considered when developing systems that meet the needs of users, thereby reducing complexity. ISO/IEC [58] describe usability as “the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use”. The interface provided (shown in Figure 4.9 and 4.10) is simplified, enabling users to easily manage their preferences in a smart home and rank their preferences according to their lifestyle choices, as users should be empowered to personalise systems according to their preferences, and this should be reasonably easy to do [12].

As mentioned earlier, this research have provided a practical solution using argumentation to manage user preferences in a real smart home. Argumentation is integrated with a reasoning system (MReasoner) and user preference interface, to provide a useful tool (Hybrid System) that resolves detected conflicts in a smart home. The implemented AAL system for smart homes aims to increase user satisfaction, which is why it is developed to understand and respond to the preferences of users. The system is designed to automate and provide viable decisions for users through the effective management of user preferences, which is managed via the interface, enabling users to manage their preferences easily in an intelligent environment. Users are entitled to systems personalised according to their preferences, which should be reasonably easy for them. Nonetheless, there is need to continue to improve current AAL systems and keep developing a better solution for safer homes, so therefore, our current solution could still be further improved in various ways.

One of such ways to further improve this research, is to investigate further on how to balance preferences and needs (especially “Weak Needs” vs “High Preferences”). In other words, dealing with specific related challenges such as the system having the ability to handle and represent subject’s preferences and needs as a partial order, thereby understanding the difference of a user’s need (subject that has high blood

pressure that needs to control salt, coffee, Alcohol and sugar intake) and preference (subject that like, cheese, olives, and whisky), and would like the system to notify them when there are special offers on these product.

On the technical side of the produced system (Hybrid System), there are few things that the research would have done better, and should be considered for future work. The current implemented system do not have the ability to assume changes over time. It would be more better and more flexible if the system have this ability. However, for the system to have such capability, more time and resources would be required as the system would need to analyse and understand pattern, routine, preference changes and so on, in order to act by itself. Also, modelling automatically the argumentation language to the augmentation language for the Hybrid System can be improved, as the modelling process is currently done manually. In addition, further work can be done on improving the Hybrid System to have that ability to handle multi users simultaneously in the same environment as the system currently handles for one use at a time.

On the interface side of the research, automating the preference names on the interface is another way to improve this research. The interface can be designed to give users a way to answer few simple and clear questions while setting up their profile. The question should relate to what they prefer in a smart home, which the system will use to generate automatically list of preference names for the user before allowing them to prioritize those preferences. This will add more flexibility and provide for various users, difference preferences, rather than the current hard-coded version which users have to strictly adhere to, and only gets modify in the database.

Further ways to improve the research in general include, developing a mobile application version of the interface with better navigation, and collaborating with end users in the validation process. Additionally, it will be ideal to investigate how to manage multiple users' conflicting preferences in the same environment, simultaneously.

This research can also be furthered to cover the area of multi-user systems, as the produced system can be enhanced to consider multiple users' simultaneously, in the same environment. User priorities will then be an important dimension, and this has been initially explored in [69, 79]. The data driven preference aspect of the system can also be enhanced, as the system currently provides strict preferences. This can be improved by allowing users to complete a short survey, that will be fed to the Hybrid system and the system will automatically generate preferences list which will be specific to that user, which has been explored in [2]. In addition, the behavioural

activities of previous users, can be used by the system to infer or assume change over time for another user with similar preference priorities. The Hybrid system currently track the activities of users, but it can be improved by analysing these activities or pattern, in other to generate rules automatically for another user with similar behavioural activities, which is similar to the approach presented in [1].

The above highlighted topics, are relevant areas that will make the produced system more effective in real life systems, if this research is taken further. However, there are some complexities of systems already in these areas, for example [79] is hard-wired, and [57] do not provide the flexibility of user's involvement, and will require confluence of other system(s) to create effective assistance in dynamic environments, for various users with potentially different preferences, which are potentially conflicting. Given theses complexities, the area will require additional work, and I believe this research is a starting point towards a more efficient system for reasoning with User's Preferences, within the Ambient Assisted Living community.

# Bibliography

- [1] Ali, S.M., Augusto, J.C., Windridge, D.: Improving the adaptation process for a new smart home user. In: International Conference on Innovative Techniques and Applications of Artificial Intelligence. pp. 421–434. Springer (2019)
- [2] Ali, S.M., Augusto, J.C., Windridge, D.: A survey of user-centred approaches for smart home transfer learning and new user home automation adaptation. *Applied Artificial Intelligence* **33**(8), 747–774 (2019)
- [3] Allen, J.F.: Towards a general theory of action and time. *Artificial intelligence* **23**(2), 123–154 (1984)
- [4] Allen, T.E.: Making cp-nets (more) useful. In: AAI. pp. 3057–3058 (2014)
- [5] Amgoud, L., Bonnefon, J.F., Prade, H.: An argumentation-based approach to multiple criteria decision. In: European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty. pp. 269–280. Springer (2005)
- [6] Amgoud, L., Cayrol, C.: On the acceptability of arguments in preference-based argumentation. In: Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence. pp. 1–7. Morgan Kaufmann Publishers Inc. (1998)
- [7] Amgoud, L., Maudet, N., Parsons, S.: Modelling dialogues using argumentation. In: MultiAgent Systems, 2000. Proceedings. Fourth International Conference on. pp. 31–38. IEEE (2000)
- [8] Amgoud, L., Prade, H.: Using arguments for making and explaining decisions. *Artificial Intelligence* **173**(3-4), 413–436 (2009)
- [9] Atkinson, K., Bench-Capon, T., McBurney, P.: Computational representation of practical argument. *Synthese* **152**(2), 157–206 (2006)
- [10] Augusto, J.C., Jose Quinde, M., Oguego, C.L.: Context-aware systems testing and validation. In: 2019 10th International Conference on Dependable Systems, Services and Technologies (DESSERT). pp. 7–12 (2019)



- [11] Augusto, J.C., Jose Quinde, M., Oguego, C.L., Gimenez Manuel, J.G.: Context-aware systems architecture - (casa). Springer, Cybernetics and Systems. In-press (2021)
- [12] Augusto, J.C., Callaghan, V., Cook, D., Kameas, A., Satoh, I.: Intelligent environments: a manifesto. *Human-centric Computing and Information Sciences* **3**(1), 1–18 (2013)
- [13] Augusto, J.C., Simari, G.R.: Temporal defeasible reasoning. *Knowledge and information systems* **3**(3), 287–318 (2001)
- [14] Augusto, J.C.: The logical approach to temporal reasoning. *Artificial Intelligence Review* **16**(4), 301–333 (2001)
- [15] Augusto, J.C.: The logical approach to temporal reasoning. *Artif. Intell. Rev.* **16**(4), 301–333 (2001)
- [16] Augusto, J.C.: Temporal reasoning for decision support in medicine. *Artificial intelligence in medicine* **33**(1), 1–24 (2005)
- [17] Augusto, J.C.: *Intelligent Computing Everywhere*, chap. Ambient Intelligence: The Confluence of Ubiquitous/Pervasive Computing and Artificial Intelligence (2007)
- [18] Augusto, J.C.: Reflections on ambient intelligence systems handling of user preferences and needs. In: *Intelligent Environments (IE), 2014 International Conference on*. pp. 369–371. IEEE (2014)
- [19] Augusto, J.C., Huch, M., Kameas, A., Maitland, J., McCullagh, P.J., Roberts, J., Sixsmith, A., Wichert, R. (eds.): *Handbook of Ambient Assisted Living - Technology for Healthcare, Rehabilitation and Well-being, Ambient Intelligence and Smart Environments*, vol. 11. IOS Press (2012)
- [20] Augusto, J.C., Mulvenna, M.D., Zheng, H., Wang, H., Martin, S., McCullagh, P.J., Wallace, J.G.: Night optimised care technology for users needing assisted lifestyles. *Behaviour & IT* **33**(12), 1261–1277 (2014)
- [21] Aztiria, A., Izaguirre, A., Basagoiti, R., Augusto, J.C.: Learning about preferences and common behaviours of the user in an intelligent environment. In: *Behaviour Monitoring and Interpretation - BMI - Smart Environments [an outgrow of BMI workshops]*. pp. 289–315 (2009)

- [22] Aztiria, A., Izaguirre, A., Basagoiti, R., Augusto, J.C.: Learning about preferences and common behaviours of the user in an intelligent environment. In: BMI Book. pp. 289–315 (2009)
- [23] Bandara, A.K., Kakas, A., Lupu, E.C., Russo, A.: Using argumentation logic for firewall policy specification and analysis. In: Large Scale Management of Distributed Systems, pp. 185–196. Springer (2006)
- [24] Bentahar, J., Alam, R., Maamar, Z., Narendra, N.C.: Using argumentation to model and deploy agent-based b2b applications. Knowledge-Based Systems **23**(7), 677–692 (2010)
- [25] Besnard, P., Hunter, A.: A logic-based theory of deductive arguments. Artificial Intelligence **128**(1-2), 203–235 (2001)
- [26] Besnard, P., Hunter, A.: Elements of Argumentation. MIT Press (2008)
- [27] Bikakis, A., Antoniou, G.: Defeasible contextual reasoning with arguments in ambient intelligence. Knowledge and Data Engineering, IEEE Transactions on **22**(11), 1492–1506 (2010)
- [28] Bonnefon, J.F., Fargier, H.: Comparing sets of positive and negative arguments: Empirical assessment of seven qualitative rules. Frontiers in Artificial Intelligence and Applications **141**, 16 (2006)
- [29] Boutilier, C., Bacchus, F., Brafman, R.I.: Ucp-networks: A directed graphical representation of conditional utilities. In: Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence. pp. 56–64. Morgan Kaufmann Publishers Inc. (2001)
- [30] Boutilier, C., Brafman, R.I., Hoos, H.H., Poole, D.: Reasoning with conditional ceteris paribus preference statements. In: Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence. pp. 71–80. Morgan Kaufmann Publishers Inc. (1999)
- [31] Brafman, R., Domshlak, C.: Preference handling-an introductory tutorial. AI magazine **30**(1), 58–58 (2009)
- [32] Brafman, R.I., Domshlak, C.: Tcp-nets for preference-based product configuration. In: Proceedings of the Forth Workshop on Configuration (in ECAI-02). pp. 101–106 (2002)

- [33] Caminada, M., Amgoud, L.: On the evaluation of argumentation formalisms. *Artificial Intelligence* **171**(5), 286–310 (2007)
- [34] Cayrol, C., de Saint-Cyr, F.D., Lagasquie-Schiex, M.C.: Revision of an argumentation system. In: *KR*. pp. 124–134 (2008)
- [35] Châtel, P., Malenfant, J., Truck, I.: Qos-based late-binding of service invocations in adaptive business processes. In: *Web Services (ICWS), 2010 IEEE International Conference on*. pp. 227–234. IEEE (2010)
- [36] Châtel, P., Truck, I., Malenfant, J.: A linguistic approach for non-functional constraints in a semantic soa environment. In: *8th International FLINS Conference on Computational Intelligence in Decision and Control (FLINS08)*. pp. 889–894 (2008)
- [37] Chesñevar, C.I., Maguitman, A.G., Loui, R.P.: Logical models of argument. *ACM Computing Surveys (CSUR)* **32**(4), 337–383 (2000)
- [38] Costa, N., Domingues, P., Fdez-Riverola, F., Pereira, A.: A mobile virtual butler to bridge the gap between users and ambient assisted living: a smart home case study. *Sensors* **14**(8), 14302–14329 (2014)
- [39] Das, S.R., Chita, S., Peterson, N., Shirazi, B.A., Bhadkamkar, M.: Home automation and security for mobile devices. In: *2011 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*. pp. 141–146. IEEE (2011)
- [40] Dertouzos, M.L.: The invisible future. chap. *Human-centered Systems*, pp. 181–191. McGraw-Hill, Inc., New York, NY, USA (2002)
- [41] Di Noia, T., Lukasiewicz, T.: *Introducing ontological cp-nets* (2012)
- [42] Dix, J., Parsons, S., Prakken, H., Simari, G.: Research challenges for argumentation. *Computer Science-Research and Development* **23**(1), 27–34 (2009)
- [43] Ferrando, S.P., Onaindia, E.: Defeasible argumentation for multi-agent planning in ambient intelligence applications. In: *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*. pp. 509–516. International Foundation for Autonomous Agents and Multiagent Systems (2012)
- [44] Gallier, J.H.: *Logic for Computer Science: Foundations of Automatic Theorem Proving*. Wiley (1987)

- [45] Galton, A., Augusto, J.C.: Stratified causal theories for reasoning about deterministic devices and protocols. In: Proceedings Ninth International Symposium on Temporal Representation and Reasoning. pp. 52–54. IEEE (2002)
- [46] Galton, A., Augusto, J.C.: Two approaches to event definition. In: International Conference on Database and Expert Systems Applications. pp. 547–556. Springer (2002)
- [47] García, A.J., Simari, G.R.: Defeasible logic programming: An argumentative approach. arXiv preprint cs/0302029 (2003)
- [48] Georgeff, M., Ingrand, F.: Decision-making in an embedded reasoning system (1989)
- [49] Ghiani, G., Manca, M., Paternò, F., Santoro, C.: End-user personalization of context-dependent applications in aal scenarios. In: Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct. pp. 1081–1084. ACM (2016)
- [50] Goldsmith, J., Junker, U.: Preference handling for artificial intelligence. *AI Magazine* **29**(4), 9–9 (2008)
- [51] Goldsmith, J., Junker, U.: Preference handling for artificial intelligence. *AI Magazine* **29**(4), 9 (2009)
- [52] Gonzales, C., Perny, P.: Gai networks for decision making under certainty. In: IJCAI'05–Workshop on Advances in Preference Handling. Citeseer (2005)
- [53] Gračanin, D., McCrickard, D.S., Billingsley, A., Cooper, R., Gatling, T., Irvin-Williams, E.J., Osborne, F., Doswell, F.: Mobile interfaces for better living: supporting awareness in a smart home environment. In: International conference on universal access in human-computer interaction. pp. 163–172. Springer (2011)
- [54] Hamblin, C.L.: Instants and intervals. In: *The Study of Time*, pp. 324–331. Springer (1972)
- [55] Homola, M., Patkos, T., Flouris, G., Šefránek, J., Šimko, A., Frtús, J., Zografistou, D., Baláž, M.: Resolving conflicts in knowledge for ambient intelligence. *The Knowledge Engineering Review* **30**(05), 455–513 (2015)

- [56] Huyck, C., Augusto, J., Gao, X., Botía, J.A.: Advancing ambient assisted living with caution. In: International Conference on Information and Communication Technologies for Ageing Well and e-Health. pp. 19–32. Springer (2015)
- [57] Ibarra, U.A., Augusto, J.C., Goenaga, A.A.: Temporal reasoning for intuitive specification of context-awareness. In: 2014 International Conference on Intelligent Environments. pp. 234–241. IEEE (2014)
- [58] ISO-IEC, .: Ergonomics requirements for office with visual display terminals (vdts) (1998)
- [59] Kleinberger, T., Becker, M., Ras, E., Holzinger, A., Müller, P.: Ambient intelligence in assisted living: enable elderly people to handle future interfaces. In: International conference on universal access in human-computer interaction. pp. 103–112. Springer (2007)
- [60] Koskela, T., Väänänen-Vainio-Mattila, K., Lehti, L.: Home is where your phone is: Usability evaluation of mobile phone ui for a smart home. In: International Conference on Mobile Human-Computer Interaction. pp. 74–85. Springer (2004)
- [61] Kristoffersen, S., Ljungberg, F.: Designing interaction styles for a mobile use context. In: International Symposium on Handheld and Ubiquitous Computing. pp. 281–288. Springer (1999)
- [62] Lee, Y.E., Benbasat, I.: Interface design for mobile commerce. *Communications of the ACM* **46**(12), 48–52 (2003)
- [63] Madrid, N.M., Fernández, J.M., Seepold, R., Augusto, J.: Sensors for ambient assisted living (aal) and smart homes
- [64] Mahesar, Q.a.: Computing argument preferences and explanations in abstract argumentation. In: 2018 IEEE 30th International Conference on Tools with Artificial Intelligence (ICTAI). pp. 281–285. IEEE (2018)
- [65] McNaull, J., Augusto, J.C., Mulvenna, M., McCullagh, P.: Flexible context aware interface for ambient assisted living. *Human-Centric Computing and Information Sciences* **4**(1), 1 (2014)
- [66] Muñoz, A., Augusto, J.C., Villa, A., Botía, J.A.: Design and evaluation of an ambient assisted living system based on an argumentative multi-agent system. *Personal and Ubiquitous Computing* **15**(4), 377–387 (2011)

- [67] Muñoz, A., Botía, J.A.: Developing an intelligent parking management application based on multi-agent systems and semantic web technologies. In: Hybrid Artificial Intelligence Systems, pp. 64–72. Springer (2010)
- [68] Muñoz, A., Botía, J.A., Augusto, J.C.: Using argumentation to understand ambiguous situations in intelligent environments. In: Ambient Intelligence Perspectives II - Selected Papers from the Second International Ambient Intelligence Forum 2009, Hradec Králové, Czech Republic, 16-17 September 2009. pp. 35–42 (2009)
- [69] Munoz, A., Botía, J.A., Augusto, J.C.: Intelligent decision-making for a smart home environment with multiple occupants. In: Computational Intelligence in Complex Decision Systems, pp. 325–371. Springer (2010)
- [70] Nazir, M., Iqbal, I., Shakir, H., Raza, A., Rasheed, H.: Future of mobile human computer interaction research-a review. In: 17th IEEE International Multi Topic Conference 2014. pp. 20–25. IEEE (2014)
- [71] Nedbal, R.: Handling possibly conflicting preferences. In: Proceedings of the Third International Conference on Intelligent Human Computer Interaction (IHCI 2011), Prague, Czech Republic, August, 2011. pp. 207–219. Springer (2013)
- [72] Nick, M., Becker, M.: A hybrid approach to intelligent living assistance. In: 7th International Conference on Hybrid Intelligent Systems (HIS 2007). pp. 283–289. IEEE (2007)
- [73] Oguego, C.L.: Bedroom Scenario (Hybrid section 8.1) (10 2019). <https://doi.org/10.22023/mdx.9944603.v1>, [https://mdx.figshare.com/articles/Bedroom\\_Scenario\\_Hybrid\\_section\\_8\\_1\\_/9944603](https://mdx.figshare.com/articles/Bedroom_Scenario_Hybrid_section_8_1_/9944603)
- [74] Oguego, C.L.: Different users' preferences effecting system output (section 7.2) (10 2019). <https://doi.org/10.22023/mdx.9944681.v1>, [https://mdx.figshare.com/articles/Different\\_users\\_preferences\\_effecting\\_system\\_output\\_section\\_7\\_2\\_/9944681](https://mdx.figshare.com/articles/Different_users_preferences_effecting_system_output_section_7_2_/9944681)
- [75] Oguego, C.L.: Solving conflicts (section 8.2) (10 2019). <https://doi.org/10.22023/mdx.9944711.v1>, [https://mdx.figshare.com/articles/Solving\\_conflicts\\_section\\_8\\_2\\_/9944711](https://mdx.figshare.com/articles/Solving_conflicts_section_8_2_/9944711)

- [76] Oguego, C.L.: Supermarket Store (Tesco) API (section 8.3) (10 2019). <https://doi.org/10.22023/mdx.9944750.v1>, [https://mdx.figshare.com/articles/Supermarket\\_Store\\_Tesco\\_API\\_section\\_8\\_3\\_/9944750](https://mdx.figshare.com/articles/Supermarket_Store_Tesco_API_section_8_3_/9944750)
- [77] Oguego, C.L., Augusto, J.C., Muñoz, A., Springett, M.: A survey on managing users' preferences in ambient intelligence. *Universal Access in the Information Society* **17**(1), 97–114 (2018)
- [78] Oguego, C.L., Augusto, J.C., Muñoz, A., Springett, M.: Using argumentation to manage users' preferences. *Future Generation Computer Systems* **81**, 235–243 (2018)
- [79] Ospan, B., Khan, N., Augusto, J., Quinde, M., Nurgaliyev, K.: Context aware virtual assistant with case-based conflict resolution in multi-user smart home environment. In: *2018 International Conference on Computing and Network Communications (CoCoNet)*. pp. 36–44. IEEE (2018)
- [80] Panisson, A.R., Meneguzzi, F., Vieira, R., Bordini, R.H.: An approach for argumentation-based reasoning using defeasible logic in multi-agent programming languages. In: *11th International Workshop on Argumentation in Multiagent Systems*. pp. 1–15 (2014)
- [81] Pigozzi, G., Tsoukiàs, A., Viappiani, P.: Preferences in artificial intelligence. *Annals of Mathematics and Artificial Intelligence* pp. 1–41 (2014)
- [82] Pigozzi, G., Tsoukias, A., Viappiani, P.: Preferences in artificial intelligence. *Annals of Mathematics and Artificial Intelligence* **77**(3-4), 361–401 (2016)
- [83] Prakken, H.: A logical framework for modelling legal argument. In: *Proceedings of the 4th international conference on Artificial intelligence and law*. pp. 1–9. ACM (1993)
- [84] Prakken, H., Vreeswijk, G.: Logics for defeasible argumentation. In: *Handbook of philosophical logic*, pp. 219–318. Springer (2001)
- [85] Rashidi, P., Mihailidis, A.: A survey on ambient-assisted living tools for older adults. *IEEE journal of biomedical and health informatics* **17**(3), 579–590 (2012)
- [86] Ruzic, L., Lee, S.T., Liu, Y.E., Sanford, J.A.: Development of universal design mobile interface guidelines (udmig) for aging population. In: *International Conference on Universal Access in Human-Computer Interaction*. pp. 98–108. Springer (2016)

- [87] Santhanam, G.R., Basu, S., Honavar, V.: Tcp- compose★—a tcp-net based algorithm for efficient composition of web services using qualitative preferences. In: Service-Oriented Computing—ICSOC 2008, pp. 453–467. Springer (2008)
- [88] Sartor, G.: A formal model of legal argumentation. *Ratio Juris* **7**(2), 177–211 (1994)
- [89] Simari, G.R., Loui, R.P.: A mathematical treatment of defeasible reasoning and its implementation. *Artificial intelligence* **53**(2-3), 125–157 (1992)
- [90] Stary, C., Pasztor, A.: Luis—a logic for task-oriented user interface specification. *International journal of intelligent systems* **10**(2), 201–231 (1995)
- [91] Stolzenburg, F., García, A.J., Chesnevar, C.I., Simari, G.R.: Computing generalized specificity. *Journal of Applied Non-Classical Logics* **13**(1), 87–113 (2003)
- [92] Tamani, N., Croitoru, M.: A quantitative preference-based structured argumentation system for decision support. In: 2014 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE). pp. 1408–1415. IEEE (2014)
- [93] Truck, I., Schmid, W.: A new proposal to represent the linguistic conditional preference networks. In: Intelligent Systems and Knowledge Engineering (ISKE), 2015 10th International Conference on. pp. 514–520. IEEE (2015)
- [94] Vila, L.: A survey on temporal reasoning in artificial intelligence. *Ai Communications* **7**(1), 4–28 (1994)
- [95] Visser, W., Hindriks, K.V., Jonker, C.M.: Reasoning about interest-based preferences. In: International Conference on Agents and Artificial Intelligence. pp. 115–130. Springer (2011)
- [96] Visser, W., Hindriks, K.V., Jonker, C.M.: Argumentation-based qualitative preference modelling with incomplete and uncertain information. *Group Decision and Negotiation* **21**(1), 99–127 (2012)
- [97] Walsh, T.: Representing and reasoning with preferences. *AI Magazine* **28**(4), 59–59 (2007)
- [98] Wang, H., Saboune, J., El Saddik, A.: Control your smart home with an autonomously mobile smartphone. In: 2013 IEEE international conference on multimedia and expo workshops (ICMEW). pp. 1–6. IEEE (2013)



- [99] Wang, H., Shao, S., Zhou, X., Wan, C., Bouguettaya, A.: Web service selection with incomplete or inconsistent user preferences. In: *Service-Oriented Computing*, pp. 83–98. Springer (2009)
- [100] van der Weide, T.L., Dignum, F., Meyer, J.J.C., Prakken, H., Vreeswijk, G.: Arguing about preferences and decisions. In: *International Workshop on Argumentation in Multi-Agent Systems*. pp. 68–85. Springer (2010)
- [101] Zadeh, L.A.: The concept of a linguistic variable and its application to approximate reasoning-i. *Information sciences* **8**(3), 199–249 (1975)
- [102] Zhang, S., Mouhoub, M., Sadaoui, S.: Integrating tcp-nets and csps: The constrained tcp-net (ctcp-net) model. In: *Current Approaches in Applied Artificial Intelligence*, pp. 201–211. Springer (2015)

# Appendix: Publications

## Authored publications

- i Oguego, C.L., Augusto, J.C., Muñoz, A. and Springett, M., 2018. A survey on managing users' preferences in ambient intelligence. *Universal Access in the Information Society*, 17(1), pp.97-114. (<https://doi.org/10.1007/s10209-017-0527-y>) (Published)
- ii Oguego, C.L., Augusto, J.C., Muñoz, A. and Springett, M., 2018. Using argumentation to manage users' preferences. *Future Generation Computer Systems*, 81, pp.235-243. (<https://doi.org/10.1016/j.future.2017.09.040>) (Published)
- iii C. L. Oguego, J. C. Augusto, M. Springett, M. Quinde and C. J. Reynolds, "An Interface for Managing users' Preferences in AmI," 2019 15th International Conference on Intelligent Environments (IE), Rabat, Morocco, 2019, pp. 56-59, doi: 10.1109/IE.2019.00009.

## Authored publication under review

- i Using argumentation to solve conflicting situations in users' preferences in ambient assisted living (under review)

## Co-authored publications

- i J. C. Augusto, M. Jose Quinde and C. L. Oguego, "Context-aware Systems Testing and Validation," 2019 10th International Conference on Dependable Systems, Services and Technologies (DESSERT), Leeds, United Kingdom, 2019, pp. 7-12. (<https://doi.org/10.1109/DESSERT.2019.8770048>)

- ii J. C. Augusto, M. Quinde, J. G. Giménez Manuel, S. M. M. Ali, C. L. Oguego and C. James-Reynolds, “The SEArch Smart Environments Architecture,” 2019 15th International Conference on Intelligent Environments (IE), Rabat, Morocco, 2019, pp. 60-63, doi: 10.1109/IE.2019.00010.
- iii J. Augusto, J. Giménez-Manuel, M. Quinde, Ch. Oguego, M. Ali and C. James-Reynolds (2020) A Smart Environments Architecture (Search), Applied Artificial Intelligence, (<https://doi.org/10.1080/08839514.2020.1712778>)
- iv Quinde, M., Giménez-Manuel, J., Oguego, C. L., and Augusto, J. C. (2020, July). Achieving multi-user capabilities through an indoor positioning system based on BLE beacons. In 2020 16th International Conference on Intelligent Environments (IE) (pp. 13-20). IEEE. ([10.1109/IE49459.2020.9155011](https://doi.org/10.1109/IE49459.2020.9155011))
- v J. Augusto, M. Quinde, Ch. Oguego, and J. Giménez-Manuel (2021) Context-aware Systems Architecture (CaSA), Cybernetics and Systems. Springer.