A Methodology for Applying Decision Policies Into Smart Buildings With the Use of Computational Argumentation and IoT Technologies

1st Nikolaos I. Spanoudakis, Senior Member, IEEE
Applied Mathematics and Computers Laboratory
Technical University of Crete
Chania, Greece
nispanoudakis@tuc.gr

2nd Panteleimon Krinakis
School of Production Engineering and Management
Technical University of Crete
Chania, Greece
pkrinakis@tuc.gr

3rd Dionysia Kolokotsa
School of Chemical and Environmental Engineering
Technical University of Crete
Chania, Greece
dkolokotsa@chenveng.tuc.gr

***Abstract*—This paper reports on our work for defining a methodology for applying energy and other polices into smart buildings that use Internet of Things enabled devices. The goal is to automate decisions on a variety of issues where there is a conflict of policies between the priorities of different users and roles. We use computational argumentation and a method for compiling meta-policies to automate the decision-making process with the capability to offer human-readable explanations and, at the same time, to achieve increased energy efficiency. IoT sensors take the necessary measurements and data using the KNX standard. To validate our methodology we used an energy related scenario. The case study is applied on a building in the Technical University of Crete, where there are suitable sensors for measuring basic parameters (or data), such as temperature, lighting conditions, motion, and, carbon dioxide levels. The building's energy consumption data is imported by the Open Studio/Energy plus application. A web-based service for hierarchical argumentation, Gorgias Cloud, is used to process the data and make decisions.**

***Index Terms*—internet of things, artificial intelligence, web application, smart buildings, explainable AI**

I. Introduction

With the anticipated surge in global population, the demand for energy will persistently rise. Existing power grids face uncertainty in meeting future demands. Limited fossil fuel reserves not only contribute to harmful emissions but also prompt inevitable social and environmental repercussions. Consequently, the traditional centralized grid is evolving into a distributed hybrid energy generation system, heavily reliant on renewable sources like wind, solar, biomass, fuel cells, and tidal power [1], [2].

The concept of the smart grid merges information and communication technologies (ICT) with power systems to achieve efficient energy generation and consumption. It facilitates a bidirectional flow of electricity and information, aiming to mitigate issues like blackouts, voltage instability, and overloads. Smart grid solutions span production, transmission, distribution, and now extend to smart buildings. Smart buildings employ ICT for home control and automation, enabled by smart devices ranging from simple sensors to household appliances. The heterogeneous network emerging by such connected objects is referred to as Internet of Things (IoT). Applications for smart buildings (or homes) can help to optimize energy consumption or other areas such as comfort and safety of the residents [3], [4].

There is still significant potential for improvement at this stage. Smart buildings require customization based on specific needs, which entails a level of context awareness. This implies that the conditions of the environment and the preferences of the occupants are crucial factors in determining how a smart building should function [5].

This paper contributes in this direction. The buildings of the Technical University of Crete aim to pioneer, among University Institutions, introducing smart management of the vast data that can be collected from environmental sensors and utilized [6]. In the scenarios that we process in this work we apply energy management policies aiming to reduce the footprint of an installation and at the same time high-quality services and living standards are provided. Finally students, as well as professors and staff, benefit from appropriate security policies application.

To achieve these goals, this paper reports for the first time on our experience from using the approach proposed by Bassiliadis et al. [7]. This approach towards multipolicy argumentation is inline with recent developments in the Explainable Artificial Intelligence (xAI) area that brings forward the need of humans for adequate explanations when AI-based) systems take decisions autonomously. Argumentation is well

suited for this area as the decision making process captures the dialectic nature of arguments that is natural to humans.

In the following we outline the needed background in Section II and then we present our methodology in Section III. In Section IV we evaluate our approach with three stakeholders. Then, we discuss related works in Section V and we conclude.

II. Materials and Methods

A. Smart buildings

The field of smart buildings has shown significant growth in recent years, both in terms of the use of new sensors and in more effective decision-making. In conventional building installations, electrical-mechanical devices (e.g. for lighting, air-conditioning) are supplied by diverse vendors. This is why a standardized approach is needed in order to connect such devices, including sensors and actuators.

KNX technology is one such standardized approach that allows devices from different manufacturers to communicate with each other [8]. A KNX BAOS server is an Internet of Things component that can expose the data of diverse KNX-compatible devices using a local IP address. In Fig. 1 the reader can see the data of four sensors that are also exposed by a REST service.

Additionally, systems have been developed for energy management in building complexes, such as university campuses. One of these is the CAMP-IT platform [6], where data is collected through various sensors, and decisions are made using fuzzy logic artificial intelligence algorithms.

Another application is the Geographic Information System (GIS) platform for monitoring zero-energy buildings. Through this platform, data from sensors located in four different countries are integrated and monitored. It is worth noting that the sensors used in this system are of the KNX type [9].

SketchUp is a three-dimensional building modeling software. Due to its user-friendly environment, it has been used by a wide range of users such as architects, planners, and mechanical engineers. OpenStudio is an extension (plug-in) of the SketchUp program, which allows to perform energy modeling of a building. Fig. 2 shows a building of the School of Chemical and Environmental Engineering of the Technical University of Crete, that has been modelled with SketchUp. Moreover, the different colors correspond to different thermal zones of the building corresponding to the different capacities of the building to conserve its temperature.

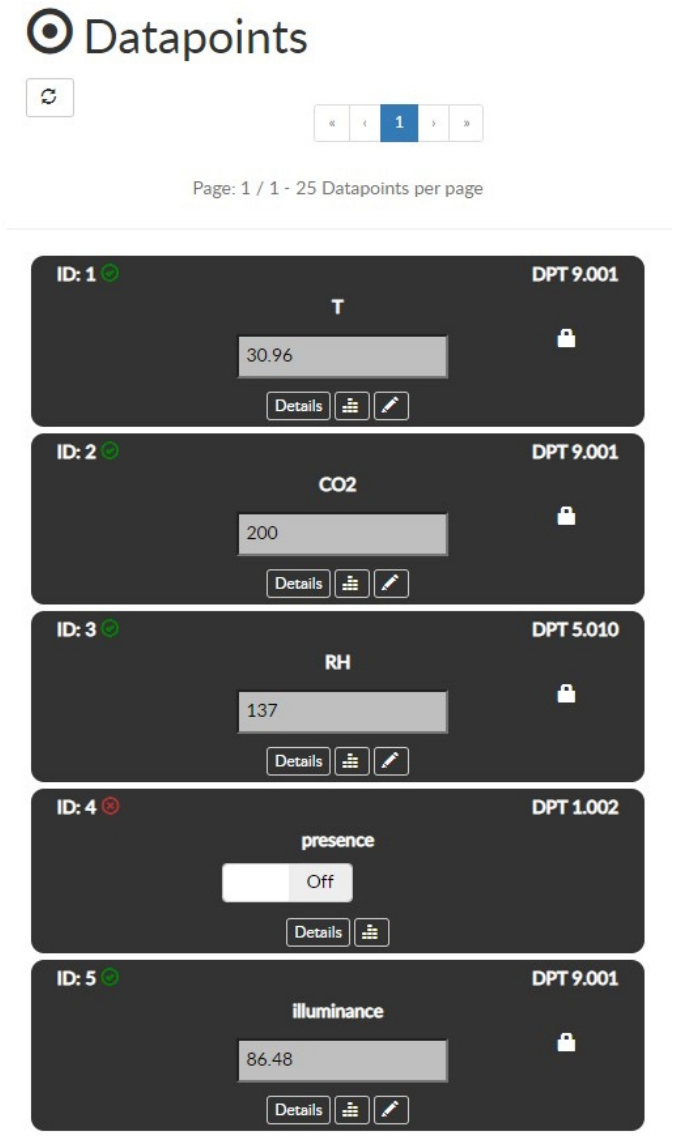

Fig. 1. Four sensors connected to a baos device, i.e. temperature (T), CO_2 level (CO2), humidity (RH) and illumination.

B. Computational Argumentation

In this section, we provide a brief overview of the Gorgias argumentation framework [10] and illustrate how application problems can be translated into an argumentation theory within this framework. Argument rules link a *conclusion* to a set of *premises* (conditions) and follow the syntax of Extended Logic Programming. According to the latter, conditions and conclusion (claim) are positive or explicit negative atomic statements, but where negation as failure is excluded from the language [11]. Arguments *attack* other arguments by supporting incompatible (complementary) conclusions.

Gorgias is an hierarchical argumentation framework in the sense that it allows to express a *priority* of an argument rule (preferred) over another argument rule (non-preferred), thus expressing a preference, or relative strength, between argument rules. This preference can be linked to a set of premises as well or not linked, in which case it can be consider to express a *default* priority. This priority expresses an attack from the preferred to the non-preferred argument rule. Argument rules supporting a conclusion (*object-level arguments*) can form composite arguments together with other object-level arguments or with priorities in order to attack other arguments.

Admissible arguments attack back their attackers. In the case that an argument (A) attacks another (B) and vice versa, the former (A) can employ a priority to "strengthen" its attack. If the latter cannot find another priority that will reverse this

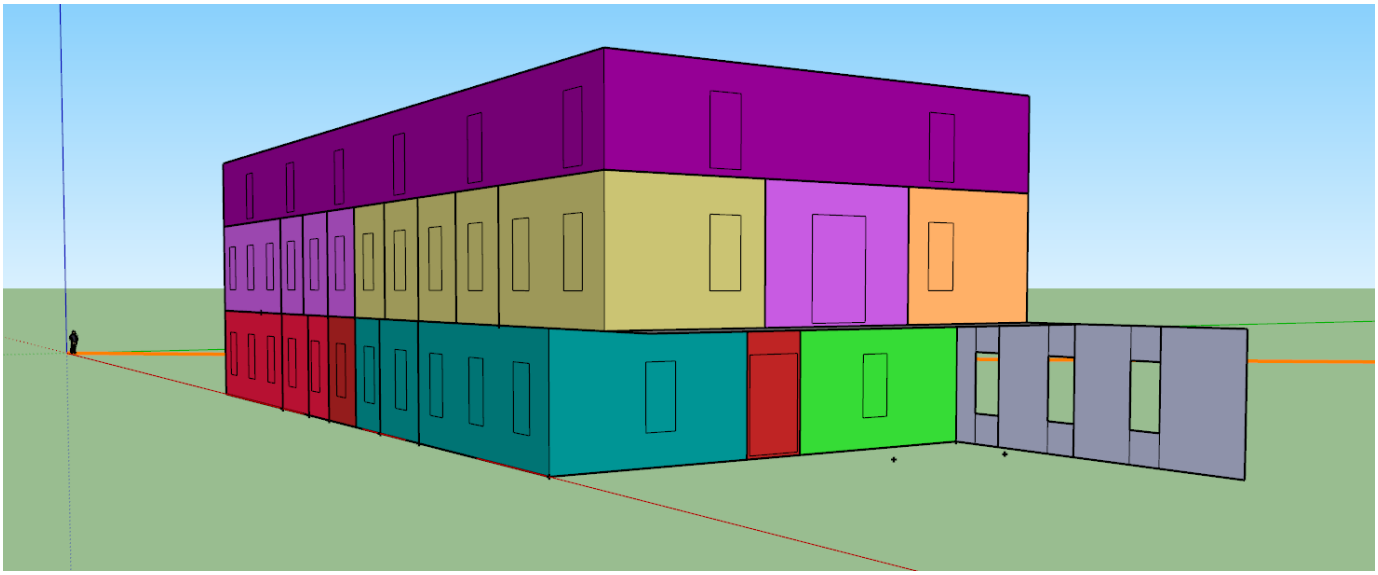

Fig. 2. The building model including the thermal zones as shown in the SketchUp tool.

attack, then it is no longer admissible. This paper does not intend to formally define this process, the interested reader can consult the Gorgias definition papers (the one by Kakas and Moraitis is a good starting point [10]).

In their work towards multipolicy argumentation [7] the authors proposed a methodology for generating meta-policies, based on policies defined by diverse stakeholders, when they interact to reach a common consensus. This work assumes that diverse stakeholders have different goals and the priorities between them are based on the relevant strength of their opinion. Not only that, this meta-policy can be influenced based on context. The methodology allows to define the meta-policy and then provides a method to combine the different policies with the meta-policy to generate a new multi-policy that can decide the desired course of action taking into account all the above data.

Recently, Spanoudakis et al. [12] offered an integrated application development environment that facilitates the development and execution of argumentation-based policies on the web. The latter is offered as a service. Thus, application developers can use this service to automate decision making, having defined their policies beforehand. Policies can be defined with the help of experts or by using a web-based tool [13] that addresses the needs of naive users that have no knowledge of argumentation or logic.

III. Proposed Methodology

A. Problem domain

This methodology applies to smart buildings that have IoT enabled devices. In the specific case of our energy-related case study, energy managers have a model of the building and have identified the thermal zones (zones of the building with different behaviour related to thermal losses).

Moreover, there are users that have diverse policies depending on their environmental sensitivity and that use different areas (offices, common spaces like amphitheaters, etc). Finally, there can be different roles that apply business rules to the different areas, e.g. a computer room, or rules to respond to different situations, e.g. in an emergency, like fire, etc.

B. Methodology

The methodology consists of the following steps:

1) *Encode the building data*. This step is about identifying the different areas in the building and encode these in logic. Then, in the case of the energy scenario, identify the different thermal zones, encode them in logic and link them to the different areas. Other relevant data to the application domain may be needed. This data is static, i.e. changes after long periods of the use of a building
2) *Encode sensors data*. Identify the different sensors and the data they provide. Link the sensors to rooms. This data is dynamic. It is provided from IoT enabled components (KNX-BAOS servers)
3) *Encode the policies of the users*. Allow the different stakeholders to define their policies
4) *Define the meta-policy*. Capture the stakeholders' policies default priorities but also possible changes under specific contexts. Combine it with the stakeholders' views
5) *Develop and install the web application*. Using this application the users can control their environment and receive explanations about the various decisions in natural language. The web server needs to have access to the building's local subnet where the BAOS servers reside and to also have a public IP address to expose the service

C. System Architecture

The developed system is composed of four sub-systems:

- *Browser*. The user monitors and controls the system through the browser. Using the browser the user can see

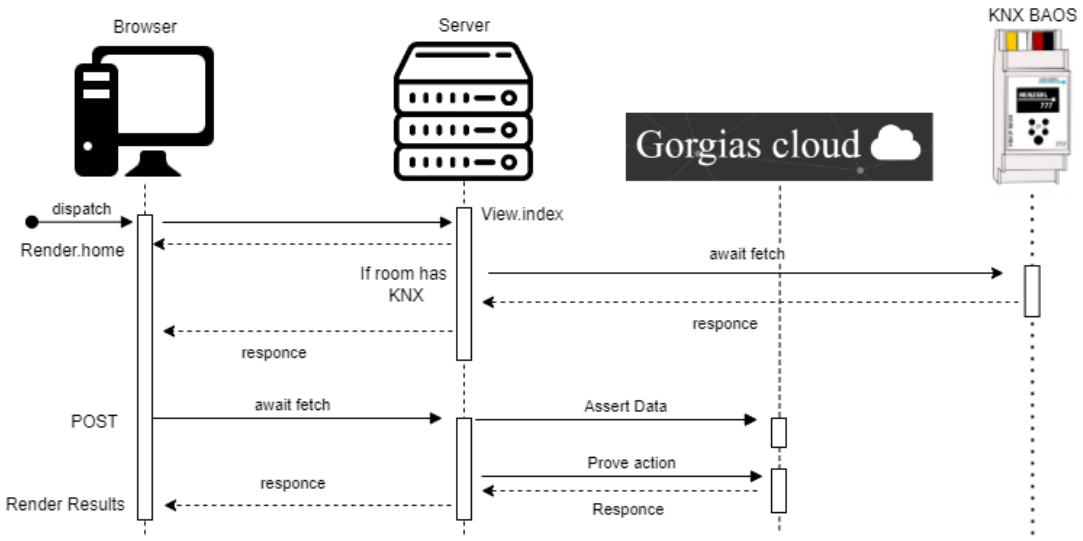

Fig. 3. The sequence diagram shows the different sub-systems and their interactions [14].

the data provided by the BAOS server and update their preferences (e.g. the desired room temperature). The user can also see an explanation of the system decision in natural language.

- *Web server*. The web server has been developed using the Python Django technology [15]
- *Gorgias Cloud server*. The Gorgias Cloud server is used for asserting the user data and context, and, subsequently, for querying about the different options (conclusions)
- *KNX BAOS server*. The KNX BAOS server is a socket that exposes the reading of the used sensors in the local subnet

The building has a local subnet where the different KNX BAOS servers are located and the system is informed about the building locations that are equipped with KNX BAOS servers. The latter expose all the devices that are connected to them.

The Gorgias Cloud server is on the Cloud and accessible through the internet[1]. The web server of the developed application is connected to the local subnet and also has a public IP to expose its service to the internet.

The interaction of the four sub-systems is shown in Fig. 3. The user opens the web-page using their *Browser*. If the user's room is equipped with a *KNX BAOS*, then the Application *Server* connects to the user room's KNX BAOS server and fetches the available data.

Then, user sees in their browser the form shown in Fig. 4, where the available data is already displayed. The user can enter the desired action with a possible update on their target temperature and click the "prove" button.

As soon as the Server gets the POST action, it connects to the *Gorgias Cloud service*, asserts the fetched data and the user inputted data and then asks whether the different possible actions are valid given the applied policy and data. The server matches the response to pre-generated natural language responses corresponding to the labels of the preferred arguments and forwards them to the browser that displays them to the user.

[1]The *Gorgias Cloud* service is offered by the Applied Mathematics and Computers Laboratory of the Technical University of Crete for free for academic use. A commercial version is offered by the start-up *Argument Theory*

IV. Evaluation

To evaluate our methodology, we developed the application outlined in section III-B. We installed it in the Technical University of Crete and connected it to the building K2 of the School of Chemical and Environmental Engineering subnet where BAOS servers operate and we have the energy model of it [6]. In the following we outline the decsion policies that we used for evaluation and then we showcase the system operation.

A. Decision Policy

The foundational element of the application is a collection of Gorgias rules files encapsulating the policies developed for the different stakeholders of our system. The stakeholders and their policies are:

- *User*: This stakeholder defines their desire regarding air-conditioning for their room (office)
- *Energy*: This stakeholder is dedicated to the implementation of adaptive energy-saving strategies, dynamically adjusting to the prevailing operational demands. A fundamental rule for the implementation of this policy is the motion sensor, which turns off the devices in the monitored space if there is no movement. Based on the thermal zones of the building, this role defines two energy saving modes, the normal *saving* mode and the *critical* saving mode
- *Emergency*: This policy specifies the protocol for device management under emergency conditions, ensuring the

safety and security of the building's occupants and infrastructure. When CO_2 levels rise (there is a fire) the alarm devices are turned on and all other devices (air-conditioners) are turned off

To elaborate on the saving modes, we would like to further explain that the normal *saving* mode describes operational parameters for temperature control on relevant devices (air conditioners). This enforces boundary conditions within the temperature thresholds set at 19°C for heating and 26°C for cooling operations. These specific bounds were based on the modeling of the building using the Open Studio tool. Subsequently, this model facilitated a comprehensive simulation, projecting the annual energy consumption metrics.

The secondary level, termed *critical* saving mode is actuated in response to high elevated energy consumption levels, signaling the necessity for the administrator to adopt immediate control measures. Under this mode, devices situated within zones identified as having disproportionately high energy consumption are automatically deactivated.

Those two proactive approaches ensure rapid mitigation of unnecessary energy expenditure, effectively reducing the operational energy footprint of the building during periods of high energy demand.

A static dataset ("Background" file) consolidates immutable parameters critical for the application's consistent execution.

As an example, we provide a snippet from the energy role decision policy in listing 1. As the Gorgias argumentation framework is an extension of the Prolog programming language the policies respect this language. It is a declarative language. Variables start with capital letters and constants with lowercase letters. The Gorgias rules (arguments) are given in the format:

$$rule(label, conclusion, []) \text{:-} premises.$$

The rule's label is used for referencing. The conclusion is a predicate and the premises an optional list of predicates. If it is empty, this means that the rule is a default rule and it applies at any time. The conclusions of the decision policy are to switch on or off a device. the "switch on" action is defined by the predicate *switchOn/2*. The */2* means that it takes two parameters, the first one defines the room and the second one a device in the room. Note that the negations of the predicate are also valid conclusions.

Returning to Listing 1, in the first line there is a comment (comment lines start with %). Line 2 contains the first argument. Its label is $r\_mose(R, D)$, note that we can use variables even in labels, the variable *R* refers to room and the variable *D* to a device. The conclusion this rule supports is $switchOff(R, D)$ and its premises are:

- $noMotion(R)$ there is no motion detected by the motion sensor in room R
- $isOn(D, R)$ the device D in room R is currently on
- $device(D, air\_cond, R)$ the device D in room R is an air condition (air_cond is a constant as it starts with lowercase letter and represents the air condition type of the device)

Then, in line 4 there is an argument for switching on the device (conclusion: $switchOn(R, D)$) with the premises:

- $isOff(D, R)$ the device D in room R is currently off
- $savingMode(R)$ the normal energy saving mode has been turned on for room R
- $tempOutOfSavingBounds(R)$ the temperature in room R is out of the saving mode bounds
- $device(D, air\_cond, R)$ the device D in room R is an air condition

In line 5 there is an argument rule for not switching on a device. Note that this is the negation of the previous conclusion (thus, these argument rules are complementary - they attack each other). The premises for this argument rule are that the room is in energy saving mode. The conflict of the argument rules in lines 4 and 5 that attack each other is resolved in the following three lines. In line 6 we have a default priority rule (as there are no premises) for switching on the device. Note that the priority is expressed with the use of the predicate *prefer/2*, whose two parameters are first the label of the preferred rule and second the label of the non-preferred rule. In line 7 we have a priority rule for not switching on the device with the premise that $notWithinHours(R)$ which implies that the room is an amphitheatre and in the School's schedule there is no organized lecture at the time. The rule in line 8 states that the rule in line 7 is stronger than the rule in line 6.

Then, in lines 9–13 we have a similar conflict between switching off and not switching off a device. Finally, the rule in line 15 puts the necessary priority to resolve a conflict that arises when both there is no motion in the room and the saving mode is applied. In that case the rule in line 2 is stronger than the one in line 11.

Listing 1. Enegry policy file snippet.

```
1  %Policy for motion sensor
2  rule(r_mose(R,D),switchOff(R,D),[]):-noMotion(R),isOn(D,
       R),device(D,air_cond,R).
3  %Policy for saving mode
4  rule(r11(R,D),switchOn(R,D),[]):-isOff(D,R),savingMode(R
       ),tempOutOfSavingBounds(R),device(D,air_cond,R).
5  rule(r12(R,D),neg(switchOn(R,D)),[]):-isOff(D,R),
       savingMode(R),device(D,air_cond,R).
6  rule(p11(R,D),prefer(r11(R,D),r12(R,D)),[]).
7  rule(p12(R,D),prefer(r12(R,D),r11(R,D)),[]):-
       notWithinHours(R).
8  rule(c12(R,D),prefer(p12(R,D),p11(R,D)),[]).
9  rule(r21(R,D),neg(switchOff(R,D)),[]):-isOn(D,R),
       savingMode(R),tempOutOfSavingBounds(R),device(D,
       air_cond,R).
10 rule(r22(R,D),switchOff(R,D),[]):-isOn(D,R),savingMode(R
       ),device(D,air_cond,R).
11 rule(p21(R,D),prefer(r21(R,D),r22(R,D)),[]).
12 rule(p22(R,D),prefer(r22(R,D),r21(R,D)),[]):-
       notWithinHours(R).
13 rule(c22(R,D),prefer(p22(R,D),p21(R,D)),[]).
14 %combination motion sensor, saving
15 rule(p_mose_saving(R,D),prefer(r_mose(R,D),r21(R,D)),[])
       .
```

The other two roles (i.e. user and emergency) have their own policies in similar files. Then, the executive officers of the building decide on the meta-policy.

The meta-policy employs three object-level rules each supporting a policy (lines 1–3 in listing 2). The emergency policy

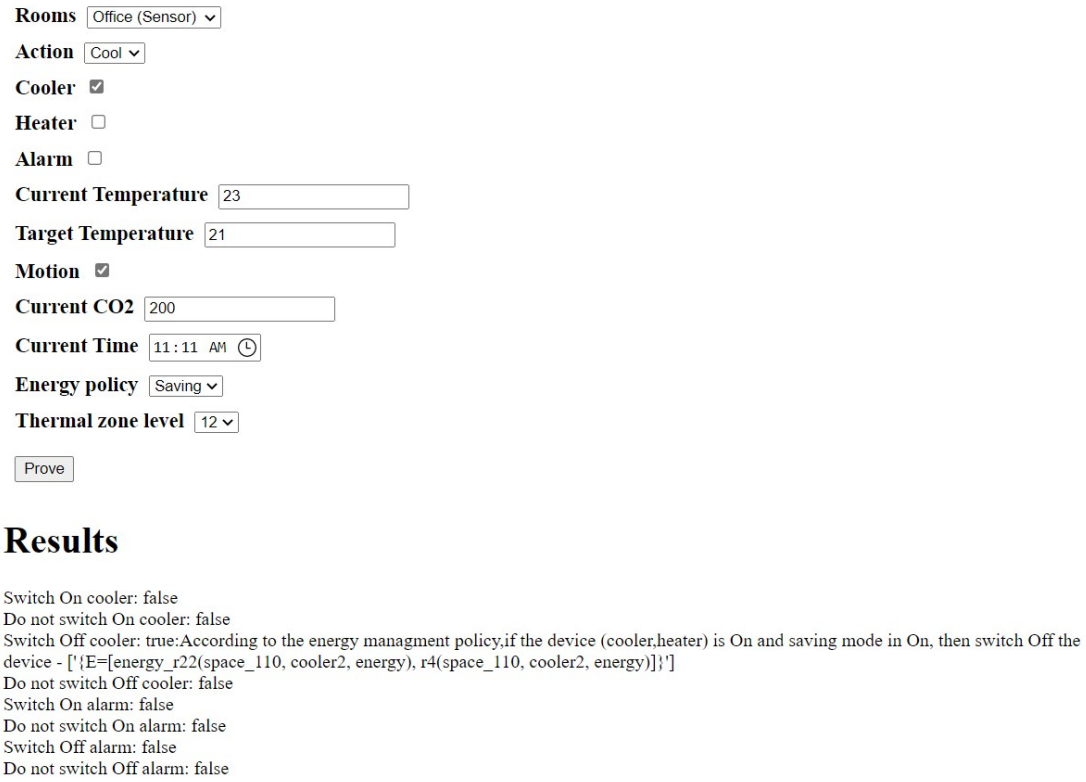

Fig. 4. The web page where the user can monitor the context of the system and press "prove" to find out the outcome of the reasoning process. [14].

is given priority over the other two (lines 4–5). The energy policy is given priority over the user policy (line 6, default preference). However, in the specific context that the user policy is about cooling a computer room, the user policy takes precedence over the energy one (line 7). This specific context priority rule is preferred over the default one (line 8).

A unified "Combined policy" file amalgamates all policy rules, providing a comprehensive rulebook that governs the application's functionality. This file is generated using the method proposed by Bassiliades et al. [7] and is uploaded to the Gorgias Cloud service and used by the web application for automated decision making.

Listing 2. Meta-policy file snippet.

```
1 rule(r1(user),policy(user),[]).
2 rule(r2(energy),policy(energy),[]).
3 rule(r3(emergency),policy(emergency),[]).
4 rule(p1(emergency),prefer(r3(emergency),r1(user)),[]).
5 rule(p2(emergency),prefer(r3(emergency),r2(energy)),[]).
6 rule(p3(energy),prefer(r2(energy),r1(user)),[]).
7 rule(p4(user),prefer(r1(user),r2(energy)),[]):-device(D,
      air_cond,R),computerRoom(R).
8 rule(c1(user),prefer(p4(user),p3(energy)),[]).
```

B. Example of System Operation

To validate our system we employed 10 scenarios that included normal operation but also energy saving and emergency scenarios. The system responded according to the meta-policy, thus validating our approach.

To give the reader an impression of the system operation we describe one of these scenarios and show its application. According to this scenario the user is located in an office in K2 and the user's desired function is to cool. The alarm device is deactivated, while the cooler is activated. The user has set the desired temperature to 21° Celcius, lower than the current one 23°. Motion sensor has detected presence in the room. The energy saving mode is activated, and the zone to which the office belongs is within the zones affected by the energy saving mode. The user can see this data when opening the Application website (see Fig. 4) and press the "prove" button to see the outcome of the system. The latter is presented under the "Results" heading in Fig. 4.

As in the user's room there are two devices (i.e. a cooler and an alarm) and four different decision types for each device (i.e. to switch on the device, to switch off the device, to not switch on the device, to not switch off the device) there are eight possibilities (number of devices times the decision types). In this case, the outcome of the system is to switch off the cooler (the only possibility that is true).

The application of the energy saving mode applicable in the room provides an argument for the cooler to switch off (line 10 in Listing 1). As the meta-policy states that the energy role is preferred over the user role (line 6 in Listing 2) the above mentioned argument is preferred over the user role policy that has an argument for not switching off the cooler until it has reached the user desired value.

V. Discussion and Related Work

In a very recent survey on Explainable Artificial Intelligence (XAI) for Internet of Things [16] the authors agree that although there are relevant works, even for energy management, explanations still suffer from several drawbacks. Among them the most important are a) the fact that the generated explanations are illogical for the non-experts, b) the lack of a guarantee that the system has been trained with data that do not contain bias (for a complete list the interested reader should consult Kök et al. [16]).

The result of argumentation typically involves a coalition of arguments that collectively support a desired conclusion. This coalition can be analyzed to provide an explanation that includes information about the support for the conclusion. Thus, in an argumentation-based approach explanations are more natural [17]–[19].

In the past, a decision-making system has been developed for smart buildings, based on defeasible logic. Through this system, a set of sensors located in a building is interconnected, and decision-making is conducted using defeasible logic [20]. Other approaches employ semantic web technologies [21] or case-based reasoning [22].

Related work on logic-based approaches loses these two qualities of the proposed approach, i.e. the possibility to merge individual policies in a meta-policy including the different roles' default and specific context priorities, and the possibility to offer an explanation in natural language.

VI. Conclusion

In this paper we propose a methodology for achieving argumentation-based automated decision making in smart buildings using IoT technology. The methodology outlines the steps from the problem statement to the installation of the servers and software. Our approach, that is based in the Gorgias Cloud system offers the unique capability to explain the decision policy in natural language (see e.g. [20]–[22]).

We consider that this work validates in the field the approach of Bassiliades et al. [7] and paves the way to the generalization of the theoretical approach and its further presentation containing also this evidence.

Our future work is on one hand to include the new explanation capabilities of the Gorgias Cloud service [12] that allow not only to get an explanation on why a course of action has been selected (attributive part) but also the reasons for which other possible actions were rejected by the system (contrastive part). Moreover, a pre-trained language model service can be used for making the response unique and more natural but also for making it available to many languages [23].

On the other hand we aim to use personal assistant agents [24] that will help users to define their policies using low-code platforms [25] such as Web-Gorgias-B [13]. These agents can interact with others and an arbitrator who will apply dynamically the meta-policy to achieve the desired effect. This way, the users can update their policies on the fly without having to combine them in a unique policy before system operation.

References

[1] P. D. Lund, J. Mikkola, and J. Ypyä, "Smart energy system design for large clean power schemes in urban areas," *Journal of Cleaner Production*, vol. 103, pp. 437–445, 2015.

[2] B. L. R. Stojkoska and K. V. Trivodaliev, "A review of Internet of Things for smart home: Challenges and solutions," *Journal of cleaner production*, vol. 140, pp. 1454–1464, 2017.

[3] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," *Future generation computer systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[4] T. Malche and P. Maheshwary, "Internet of Things (IoT) for building smart home system," in *2017 International conference on I-SMAC (IoT in social, mobile, analytics and cloud)(I-SMAC)*. IEEE, 2017, pp. 65–70.

[5] M. Jia, A. Komeily, Y. Wang, and R. S. Srinivasan, "Adopting Internet of Things for the development of smart buildings: A review of enabling technologies and applications," *Automation in Construction*, vol. 101, pp. 111–126, 2019.

[6] D. Kolokotsa, K. Gobakis, S. Papantoniou, C. Georgatou, N. Kampelis, K. Kalaitzakis, K. Vasilakopoulou, and M. Santamouris, "Development of a web based energy management system for University Campuses: The CAMP-IT platform," *Energy and Buildings*, vol. 123, pp. 119–135, 2016.

[7] N. Bassiliades, N. I. Spanoudakis, and A. C. Kakas, "Towards multipolicy argumentation," in *Proceedings of the 10th Hellenic Conference on Artificial Intelligence*, 2018, pp. 1–10.

[8] H. Merz, T. Hansemann, and C. Hübner, *Building automation*. Springer, 2009.

[9] K. Gobakis, A. Mavrigiannaki, K. Kalaitzakis, and D.-D. Kolokotsa, "Design and development of a web based gis platform for zero energy settlements monitoring," *Energy Procedia*, vol. 134, pp. 48–60, 2017.

[10] A. C. Kakas, P. Moraitis, and N. I. Spanoudakis, "*GORGIAS*: Applying argumentation," *Argument & Computation*, vol. 10, no. 1, pp. 55–81, 2019.

[11] Y. Dimopoulos and A. C. Kakas, "Logic programming without negation as failure," in *Logic Programming, Proceedings of the 1995 International Symposium, Portland, Oregon, USA, December 4-7, 1995*, 1995, pp. 369–383.

[12] N. I. Spanoudakis, G. Gligoris, A. Koumi, and A. C. Kakas, "Explainable argumentation as a service," *Journal of Web Semantics*, vol. 76, p. 100772, 2023.

[13] N. I. Spanoudakis, K. Kostis, and K. Mania, "Web-Gorgias-B: Argumentation for All." in *Proceedings of the 13th International Conference on Agents and Artificial Intelligence - Volume 2: ICAART*, INSTICC. SciTePress, 2021, pp. 286–297.

[14] P. Krinakis, "Development of an intelligent system for decision policy into smart buildings with the use of hierarchical argumentation theory," Master's thesis, Technical University of Crete, 2021.

[15] J. Forcier, P. Bissex, and W. J. Chun, *Python web development with Django*. Addison-Wesley Professional, 2008.

[16] I. Kök, F. Y. Okay, Ö. Muyanlı, and S. Özdemir, "Explainable Artificial Intelligence (XAI) for Internet of Things: A Survey," *IEEE Internet of Things Journal*, vol. 10, no. 16, pp. 14 764–14 779, 2023.

[17] A. Vassiliades, N. Bassiliades, and T. Patkos, "Argumentation and explainable artificial intelligence: a survey," *The Knowledge Engineering Review*, vol. 36, 2021.

[18] K. Čyras, A. Rago, E. Albini, P. Baroni, and F. Toni, "Argumentative XAI: A survey," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, Z.-H. Zhou, Ed. International Joint Conferences on Artificial Intelligence Organization, 2021, pp. 4392–4399.

[19] N. Spanoudakis, A. C. Kakas, and A. Koumi, "Application level explanations for argumentation-based decision making." in *1st International Workshop on Argumentation for eXplainable AI (ArgXAI 2022), Cardiff, Wales, United Kingdom, September 12th, @COMMA*, 2022.

[20] T. G. Stavropoulos, E. Kontopoulos, N. Bassiliades, J. Argyriou, A. Bikakis, D. Vrakas, and I. Vlahavas, "Rule-based approaches for energy savings in an ambient intelligence environment," *Pervasive and Mobile Computing*, vol. 19, pp. 1–23, 2015.

[21] A. Fensel, S. Tomic, V. Kumar, M. Stefanovic, S. V. Aleshin, and D. O. Novikov, "Sesame-s: Semantic smart home system for energy efficiency," *Informatik-Spektrum*, vol. 36, no. 1, pp. 46–57, 2013.

[22] S.-Y. Yang, "Developing a cloud energy-saving and case-based reasoning information agent with web service techniques," in *2012 Fifth International Symposium on Parallel Architectures, Algorithms and Programming*. IEEE, 2012, pp. 178–185.

[23] H. Wang, J. Li, H. Wu, E. Hovy, and Y. Sun, "Pre-Trained Language Models and Their Applications," *Engineering*, vol. 25, pp. 51–65, 2023.

[24] N. Spanoudakis and P. Moraitis, "Modular JADE Agents Design and Implementation Using ASEME," in *2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, vol. 2, 2010, pp. 221–228.

[25] A. Sahay, A. Indamutsa, D. Di Ruscio, and A. Pierantonio, "Supporting the understanding and comparison of low-code development platforms," in *2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. IEEE, 2020, pp. 171–178.